ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ИРКУТСКОЙ ОБЛАСТИ «ЧЕРЕМХОВСКИЙ ГОРНОТЕХНИЧЕСКИЙ КОЛЛЕДЖ ИМ. М.И. ЩАДОВА»

РАССМОТРЕНО
на заседании ЦК
«Информатики и ВТ»
«31» июнь 2022 г.
Протокол № 10
Председатель: Окладникова Т.В.

УТВЕРЖДАЮ
И.о. зам. директора по УР
О.В. Папанова
«15» июнь 2022 г.

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

для выполнения
практических работ студентов
по учебной дисциплине

ПМ. 02 Осуществление интеграции профессионального модуля

09.02.07 Информационные системы и программирование

Разработал:
Литвинцева Е.А. преподаватель спец. дисциплин ГБПОУ «ЧГТК им. М.И. Щадова»

2022г.

СОДЕРЖАНИЕ

1.ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Методические указания по выполнению практических (лабораторных) работ по ПМ 08 **«Осуществление интеграции профессионального модуля»** предназначены для студентов специальности 09.02.07 Информационные системы и программирование, составлены в соответствии с рабочей программой ПМ 02 «**Осуществление интеграции профессионального модуля»**» с учетом рекомендаций **требований Мин. обр. (**помещение лаборатория Программного обеспечения и сопровождения компьютерных систем должна удовлетворять требованиям санитарно-эпидемиологических правил и нормативов (СанПиН 2.4.2 № 178-02), и оснащено типовым оборудованием, указанным в настоящих требованиях, в том числе специализированной учебной мебелью и средствами обучения, достаточными для выполнения требований к уровню подготовки обучающихся[1]) и направлены на овладение следующими компетенциями:

ПК 2.1 Разрабатывать требования к программным модулям на основе анализа проектной и технической документации на предмет взаимодействия компонент.

ПК 2.2 Выполнять интеграцию модулей в программное обеспечение.

ПК 2.3 Выполнять отладку программного модуля с использованием специализированных программных средств.

ПК 2.4 Осуществлять разработку тестовых наборов и тестовых сценариев для программного обеспечения.

ПК 2.5 Производить инспектирование компонент программного обеспечения на предмет соответствия стандартам кодирования.

ОК 1. Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам

ОК 2. Осуществлять поиск, анализ и интерпретацию информации, необходимой для выполнения задач профессиональной деятельности

ОК 3. Планировать и реализовывать собственное профессиональное и личностное развитие.

ОК 4. Работать в коллективе и команде, эффективно взаимодействовать с коллегами, руководством, клиентами.

ОК 5. Осуществлять устную и письменную коммуникацию на государственном языке с учетом особенностей социального и культурного контекста.

ОК 6. Проявлять гражданско-патриотическую позицию, демонстрировать осознанное поведение на основе традиционных общечеловеческих ценностей.

ОК 7. Содействовать сохранению окружающей среды, ресурсосбережению, эффективно действовать в чрезвычайных ситуациях.

ОК 8. Использовать средства физической культуры для сохранения и укрепления здоровья в процессе профессиональной деятельности и поддержания необходимого уровня физической подготовленности.

ОК 9. Использовать информационные технологии в профессиональной деятельности

[1] См. Письмо Минобрнауки РФ от 24 ноября 2011 г. N МД-1552/03 «Об оснащении общеобразовательных учреждений учебным и учебно-лабораторным оборудованием»

ОК 10. Пользоваться профессиональной документацией на государственном и иностранном языках.

ОК 11. Планировать предпринимательскую деятельность в профессиональной сфере

Методические указания являются частью учебно-методического комплекса по ПМ 02 «**Осуществление интеграции профессионального модуля»**» и содержат задания, указания для выполнения практических (лабораторных) работ, теоретический минимум и т.п. Перед выполнением практической работы каждый студент обязан показать свою готовность к выполнению работы:

- пройти инструктаж по техники безопасности;
- ответить на теоретические вопросы преподавателя.

По окончанию работы студент оформляет отчет в тетради и защищает свою работу.

В результате выполнения полного объема практических работ студент должен у**меть:**

- использовать выбранную систему контроля версий;
- использовать методы для получения кода с заданной функциональностью и степенью качества
- использовать современные технологии разработки программного обеспечения, инструментальные средства разработки программного обеспечения при решении ситуационных задач
- применять методы математического моделирования при решении задач

При проведении практических работ применяются следующие технологии и методы обучения:

1. проблемно-поисковых технологий
2. тестовые технологии

Порядок выполнения работы:

1. Изучить инструкцию к практической работе.
2. Выполнить задание.
3. Оформить отчет.

Содержание отчета:

1. Тема.
2. Цель.
3. Материальное обеспечение.
4. Практическое задание.

Требования к рабочему месту:

1. Посадочное место по количеству обучающихся
2. В состав лаборатории должна быть включена одна машина для преподавателя с соответствующим периферийным оборудованием.

Лаборатория Программного обеспечения и сопровождения компьютерных систем

Критерии оценки:

Оценки «5» (отлично) заслуживает студент, обнаруживший при выполнении заданий всестороннее, систематическое и глубокое знание учебно - программного материала, учения свободно выполнять профессиональные задачи с всесторонним творческим подходом, обнаруживший познания с использованием основной и дополнительной литературы, рекомендованной программой, усвоивший взаимосвязь изучаемых и изученных дисциплин в их значении для приобретаемой специальности, проявивший творческие способности в понимании, изложении и использовании учебно- программного материала, проявивший высокий профессионализм, индивидуальность в решении поставленной перед собой

задачи, проявивший неординарность при выполнении практических заданий.

Оценки «4» (хорошо) заслуживает студент, обнаруживший при выполнении заданий полное знание учебно- программного материала, успешно выполняющий профессиональную задачу или проблемную ситуацию, усвоивший основную литературу, рекомендованную в программе, показавший систематический характер знаний, умений и навыков при выполнении теоретических и практических заданий по дисциплине «Информатика».

Оценки «3» (удовлетворительно) заслуживает студент, обнаруживший при выполнении практических и теоретических заданий знания основного учебно-программного материала в объеме, необходимом для дальнейшей учебной и профессиональной деятельности, справляющийся с выполнением заданий, предусмотренных программой, допустивший погрешности в ответе при защите и выполнении теоретических и практических заданий, но обладающий необходимыми знаниями для их устранения под руководством преподавателя, проявивший какую-то долю творчества и индивидуальность в решении поставленных задач.

Оценки «2» (неудовлетворительно) заслуживает студент, обнаруживший при выполнении практических и теоретических заданий проблемы в знаниях основного учебного материала, допустивший основные принципиальные ошибки в выполнении задания или ситуативной задачи, которую он желал бы решить или предложить варианты решения, который не проявил творческого подхода, индивидуальности.

В соответствии с учебным планом программы подготовки специалистов среднего звена по специальности **09.02.07 Информационные системы и программирование** и рабочей программой на практические (лабораторные) работы по ПМ 02 **«Осуществление интеграции профессионального модуля»** отводится 168 часов.

2. ПЕРЕЧЕНЬ ПРАКТИЧЕСКИХ РАБОТ

№ п/п	Название практической работы (указать раздел программы, если это необходимо)	Количество часов
Раздел 1. Разработка программного обеспечения		
1	Практическая работа № 1. Анализ предметной области	2
2	Практическая работа № 2. Анализ предметной области	2
3	Практическая работа №3. Разработка и оформление технического задания.	2
4	Практическая работа №4. Разработка и оформление технического задания.	2
5	Практическая работа № 5. Построение архитектуры программного средства.	2
6	Практическая работа № 6. Изучение работы в системе контроля версий.	2
7	Практическая работа № 7. Построение диаграммы Вариантов использования и диаграммы последовательности	2
8	Практическая работа № 8. Построение диаграммы Кооперации использования и диаграммы Развертывания	2
9	Практическая работа № 9. Построение диаграммы Деятельности, диаграммы Состояний и диаграммы Классов	2
10	Практическая работа № 10. Построение диаграммы компонентов	2
11	Практическая работа № 11. Построение диаграммы IDEFO	2
12	Практическая работа № 12. Построение диаграммыIDEF1X	2
13	Практическая работа № 13. Построение диаграмм потоков данных	2
14	Практическая работа № 14. Разработка тестового сценария	2
15	Практическая работа № 15. Разработка тестового сценария	2
16	Практическая работа № 16. Оценка необходимого количества тестов	2
17	Практическая работа № 17. Оценка необходимого количества тестов	2
18	Практическая работа № 18. Разработка тестовых пакетов	2
19	Практическая работа № 19. Разработка тестовых пакетов	2
20	Практическая работа № 20. Оценка программных средств с помощью метрик	2
21	Практическая работа № 21. Оценка программных средств с помощью метрик	2
22	Практическая работа № 22. Инспекция программного кода на предмет соответствия стандартам кодирования	2
Раздел 2. Средства разработки программного обеспечения		
1	Практическая работа № 1 Разработка структуры проекта. Разработка модульной структуры проекта (диаграммы модулей)	2

2	Практическая работа № 2 Разработка перечня артефактов и протоколов проекта.	2
3	Практическая работа №3 Настройка работы системы контроля версий. Настройка типов импортируемых файлов, путей, фильтров и др. параметров импорта в репозиторий. Разработка и интеграция модулей проекта (командная работа)	2
4	Практическая работа №4 Создание базы данных в MS Sql Server. Загрузка таблиц и данных. Создание таблиц спецификаций. Импортирование данных и SQL сценариев.	2
5	Практическая работа № 5 Разработка приложения для однотабличной базы данных	2
6	Практическая работа № 6 Разработка приложения для многотабличной базы данных	2
7	Практическая работа № 7 Разработка приложения для многотабличной базы данных	2
8	Практическая работа № 8 Элементы управления	2
9	Практическая работа № 9 Оформление веб страниц	2
10	Практическая работа № 10 Работа с базами данных при разработке веб приложений	2
11	Практическая работа № 11 Разработка веб приложения	2
12	Практическая работа № 12 Подключения к источнику данных	2
13	Практическая работа № 13 Создание и выполнение команд ад источникам данных	2
14	Практическая работа № 14 Работа с таблицами	2
15	Практическая работа № 15 Строки и DataAdapter	2
16	Практическая работа № 16 Отношения между таблицами	2
17	Практическая работа № 17 Фильтрация и поиск	2
18	Практическая работа № 18 DataSet со строгим контролем типов	2
19	Практическая работа № 19 Обновление данных	2
20	Практическая работа № 20 Модуль автоматизации приложения	2
21	Практическая работа № 21 Создание проекта по юнит тестированию	2
22	Практическая работа № 22 Модуль администратора приложения	2
23	Практическая работа № 23 Отладка отдельных модулей программного проекта	2
24	Практическая работа № 24 Организация обработки исключений	2

25	Практическая работа № 25 Применение отладочных классов в проекте	2
26	Практическая работа № 26 Отладка проекта	2
27	Практическая работа № 27 Инспекция кода модулей проекта	2
28	Практическая работа № 28 Тестирование интерфейса пользователя средствами инструментальной среды разработки	2
29	Практическая работа № 29 Разработка тестовых модулей проекта для тестирования отдельных модулей	2
30	Практическая работа № 30 Выполнение функционального тестирования	2
31	Практическая работа № 31 Тестирование интеграции	2
Раздел 3. Моделирование в программных системах		
1	Практическая работа № 1 Построение простейших математических моделей	2
2	Практическая работа № 2 Построение простейших статистических моделей	2
3	Практическая работа № 3 Интервальная оценка параметров. Приближенные методы построения доверительных интервалов	2
4	Практическая работа № 4 Решение простейших однокритериальных задач.	2
5	Практическая работа № 5 Решение простейших многокритериальных задач	2
6	Практическая работа № 6 Задача Коши для уравнения теплопроводности	2
7	Практическая работа № 7 Преобразование задач линейного программирования. Сведение произвольной задачи линейного программирования к основной задаче линейного программирования	2
8	Практическая работа № 8 Графический метод решения задач линейного программирования	2
9	Практическая работа № 9 Решение задач линейного программирования симплекс - методом	2
10	Практическая работа № 10 Нахождение начального решения транспортной задачи	2
11	Практическая работа № 11 Решение транспортной задачи методом потенциалов	2
12	Практическая работа № 12 Применение метод стрельбы для решения линейной краевой задачи	2
13	Практическая работа № 13 Задача о распределении средств между предприятиями	2
14	Практическая работа № 14 Задача о замене оборудования	2
15	Практическая работа № 15 Нахождение кратчайших путей на графе	2

16	Практическая работа № 16 Решение задачи о максимальном потоке	2
17	Практическая работа № 17 Составление систем уравнений Колмогорова. Нахождение финальных вероятностей	2
18	Практическая работа № 18 Чистый доход о эксплуатации в стационарном режиме на основе нахождения финальных вероятностей	2
19	Практическая работа № 19 Нахождение характеристик простейших систем массового обслуживания	2
20	Практическая работа № 20 Алгоритм поиска решения матричной антагонистической игры	2
21	Практическая работа № 21 Сведение игры mxn к задаче линейного программирования	2
22	Практическая работа № 22 Решение задач массового обслуживания методами имитационного моделирования	2
23	Практическая работа № 23 Многоканальная система массового обслуживания с ожиданием и ограничением на длину очереди	2
24	Практическая работа № 24 Многоканальная система массового обслуживания с неограниченной очередью	2
25	Практическая работа № 25 Прогнозирование с помощью методов экстраполяции	2
26	Практическая работа № 26 Прогнозирование на основе временных рядов	2
27	Практическая работа № 27 Сглаживание временных рядов с помощью скользящей средней	2
28	Практическая работа № 28 Оценка параметров нормальной модели множественной регрессии	2
29	Практическая работа № 29 Решение матричной игры методом итераций	2
30	Практическая работа № 30 Рассмотрение решения матричных игр симплексным методом	2
31	Практическая работа № 31 Решение матричных игр графическим методом	2

3.СОДЕРЖАНИЕ ПРАКТИЧЕСКИХ РАБОТ

Раздел 1. Разработка программного обеспечения

Практическая работа № 1

Цель: описать и проанализировать ИС, необходимые элементы ИС и системного ПО ИС

Задание 1: ознакомьтесь с информацией

Проблемы управления программными проектами впервые появились в 60-х– начале70-хгодов прошлого века, когда провалились многие большие проекты по разработке программных продуктов. Были зафиксированы задержки в создании ПО, программное обеспечение было ненадежным, затраты на разработку в несколько раз превосходили первоначальные оценки и т.д. Провалы этих проектов обуславливались не только некомпетентностью руководителей и программистов. Напротив, в этих больших поисковых проектах принимали участие люди, уровень квалификации которых был явно выше среднего. Причины провалов коренились в тех подходах, которые использовались в управлении проектами. Применяемая методика была основана на опыте управления техническими проектами и оказалась неэффективной при разработке программных проектов.

Руководители программных проектов выполняют такую же работу, что и руководители технических проектов. Вместе с тем процесс разработки ПО существенно отличается от процессов реализации технических проектов. Ниже приведен небольшой список этих отличий.

1. Программный продукт нематериален. Менеджер судостроительного проекта или проекта постройки здания видит результат выполнения своего проекта. Если реализация проекта отстает от графика,то это видно по незавершенности конструкции. В противоположность этому процент незавершенности программного проекта нельзя увидеть или потрогать. Менеджер программного проекта может полагаться только на документацию, которая фиксирует процесс разработки программного продукта.

2. Не существует стандартных процессов разработки программного обеспечения. На сегодняшний день не существует четкой зависимости между процессом создания ПО и типом создаваемого программного продукта. Другие технические дисциплины имеют длительную историю, процессы разработки технических изделий многократно опробованы и проверены. Изучением же процессов создания ПО специалисты занимаются только последние несколько лет. Поэтому прока нельзя точно предсказать, на каком этапе процесса разработки ПО могут возникнуть проблемы, угрожающие всему проекту.

3. Большие программные проекты – это часто одноразовые проекты. Большие программные проекты, как правило, значительно отличаются от проектов, реализованных ранее. Поэтому, чтобы уменьшить неопределенность в планировании проекта, руководители проектов должны обладать очень большим практическим опытом. Но постоянные технологические изменения в компьютерной технике обесценивают предыдущий опыт. Перечисленные особенности могут привести к тому, что реализация проекта выйдет за рамки временного графика или бюджетных ассигнований. Об этом всегда нужно помнить.

Процессы управления программными проектами

Невозможно описать и стандартизировать все работы, выполняемые менеджером проекта по созданию ПО, но в большинстве случаев к ним относятся.

- Написание предложений по созданию ПО.
- Планирование и составление графика работ проекта.
- Оценивание стоимости проекта.
- Контроль процессов выполнения работ.
- Подбор персонала.
- Написание отчетов и представлений.

Время выполнения больших программных проектов может занимать несколько лет. В течение этого времени цели и намерения организации, оказавшей программный проект, могут существенно измениться. Может оказаться, что разрабатываемый программный продукт стал уже ненужным либо исходные требования к ПО устарели и их нужно кардинально менять. В такой ситуации руководство организации-разработчика может принять решение о прекращении разработки ПО или об изменении проекта в целом.

Планирование проекта

Эффективное управление проектами напрямую зависит от правильного планирования работ. План, разработанный на начальном этапе проекта, рассматривается всеми его участниками как руководящий документ, выполнение которого должно привести к успешному завершению

проекта. Этот первоначальный план должен максимально подробно описывать все этапы реализации проекта.

План проекта должен показать ресурсы, необходимые для реализации проекта, разделение работ на этапы и временной график выполнения этих этапов. Детализация планов проектов очень разнится в зависимости от типа разрабатываемого программного продукта и организации-

Разработчика. Но в любом случае большинство планов содержит следующие разделы.

1. Введение. Краткое описание целей проекта и проектных ограничений (бюджетных, временных и т.д.).

2. Организация выполнения проекта. Описание способа подбора команды разработчиков и распределение обязанностей между членами команды.

3. Анализ рисков. Описание возможных проектных рисков, вероятность их проявления и стратегий, направленных на их уменьшение.

4. Аппаратные и программные ресурсы для реализации проекта. Перечень аппаратных средств и программного обеспечения, необходимого для разработки программного продукта.

5. Разбиение работ на этапы. Проект разбивается на отдельные процессы, определяются этапы выполнения проекта, приводится описание результатов каждого этапа и контрольные отметки.

6. График работ. В графике работ отображаются зависимости между отдельными этапами разработки по, оценки времени их выполнения и распределение членов команды проекта по отдельным этапам.

7. Механизмы контроля и мониторинга за ходом выполнения проекта. Описываются механизмы и сроки предоставления отчетов о ходе работ, а также механизмы мониторинга всего проекта.

При планировании проекта разработки ПО определяются контрольные точки – ***вехи***, отмечающие окончание определенного этапа работ. Для каждой вехи создается отчет, который предоставляется руководству проекта.

При определении контрольных точек весь процесс создания ПО должен быть разбит на отдельные этапы с указанным «выходом» (результатом) каждого этапа. Например, на рис. 1 показаны этапы разработки спецификации требований в случае, когда для ее проверки используется прототип системы.

Рисунок 1. Этапы процесса разработки спецификации

Информационный процесс - это осуществление всей совокупности следующих элементарных информационных актов: прием или создание информации, ее хранение, передача и использование. Информационная система - это совокупность механизмов, обеспечивающих полное осуществление информационного процесса.

Вне ИС информация может лишь сохраняться в виде записей на тех или иных физических носителях, но не может быть ни принятой, ни переданной, ни использованной.

Информационная система - организационно-техническая система, которая предназначена для выполнения информационно-вычислительных работ или предоставления информационно-вычислительных работ или предоставления информационно-вычислительных услуг, удовлетворяющих потребности системы управления и ее пользователей - управленческого персонала, внешних пользователей путем использования и/или создания информационных продуктов. Информационные системы существуют в рамках системы управления и полностью подчинены целям функционирования этих систем.

Информационно-вычислительная работа - деятельность, связанная с использованием информационных продуктов. Типичным примером информационной работы является поддержка информационных технологий управления.

Информационно-вычислительная услуга - это разовая информационно-вычислительная работа.Под информационным продуктом понимается вещественный или нематериальный результат интеллектуального человеческого труда, обычно материализованный на определенном носителе, например разнообразных программных продуктов, выходной информации в виде документов управления, баз данных, хранилищ данных, баз знаний, проектов ИС и ИТ.

Методологическую основу изучения ИС составляет системный подход, в соответствии с которым любая система представляет собой совокупность взаимосвязанных объектов, функционирующих совместно для достижения общей цели.

Информационная система представляет собой совокупность функциональной структуры, информационного, математического, технического, организационного и кадрового обеспечения, которые объединены в единую системы в целях сбора, хранения, обработки и выдачи необходимой информации для выполнения функций управления. Она обеспечивает информационные потоки:

I-1 - информационный поток из внешней среды в систему управления, который, с одной стороны, представляет собой поток нормативной информации, создаваемый государственными учреждениями в части законодательства, а с другой стороны - поток информации о конъюнктуре рынка, создаваемый конкурентами, потребителями, поставщиками;

I-2 - информационный поток из системы управления во внешнюю среду (отчетная информация, прежде всего финансовая в государственные органы, инвесторам, кредиторам, потребителям; маркетинговая информация потенциальным потребителям);

I-3 - информационный поток из системы управления на объект, представляет собой совокупность плановой, нормативной и распорядительной информации для осуществления хозяйственных процессов;

I-4 - информационный поток от объекта в систему управления, который отражает учетную информацию о состоянии объекта управления экономической системой (сырья, материалов, денежных, энергетических, трудовых ресурсов, готовой продукции и выполненных услугах) в результате выполнения хозяйственных процессов.

Задачи информационных систем

Корпоративные системы позволяют решить следующие задачи:

- гарантировать требуемое качество управления предприятием;
- повысить оперативность и эффективность взаимодействия между подразделениями;
- обеспечить управляемость качеством выпускаемой продукции;
- увеличить экономическую эффективность деятельности предприятия;
- создать систему статистического учета на предприятии;
- осуществлять прогноз развития предприятия;
- создать систему стратегического и оперативного планирования, систему прогнозирования.

Задание 2:

1. Выберите предметную область
2. Выберите название ИС в рамках предметной области.
3. Определите цель ИС
4. Проведите анализ осуществимости ИС

4.1. Что произойдет с организацией, если система не будет введена в эксплуатацию?

4.2. Какие текущие проблемы существуют в организации и как новая система поможет их решить?

4.3. Каким образом (и будет ли) ИС способствовать целям бизнеса?

4.4. Требует ли разработка ИС технологии, которая до этого раньше не использовалась в организации?

5. Где будет размещена ИС? Кто является пользователем ИС?
6. Комплекс технических средств ИТ

6.1. Какие средства компьютерной техники необходимы для ИС?

6.2. Какие средства коммуникационной техники необходимы для ИС?

6.3. Какие средства организационной техники необходимы для ИС?

6.4. Какие средства оперативной полиграфии необходимы для ИС?

Итог работы: отчет, защита работы.

Практическая работа № 2

Цель: описать и проанализировать ИС, необходимые элементы ИС и системного ПО ИС

Задание 1:

Опишите системное ПО ИТ

Предметная область	Сущность задачи
Страховая медицинская компания	Страховая медицинская компания (СМК) заключает договоры добровольного медицинского страхования с населением и договоры с лечебными учреждениями на лечение застрахованных клиентов. При возникновении страхового случая клиент подает заявку на оказание медицинских услуг по условиям договора инспектору, который работает с данным клиентом. Инспектор направляет данного клиента в лечебное учреждение. Отчеты о своей деятельности инспектор предоставляет в бухгалтерию. Бухгалтерия проверяет оплату договоров, перечисляет денежные средства за оказанные услуги лечебным учреждениям, производит отчисления в налоговые органы и предоставляет отчетность в органы государственной статистики. СМК не только оплачивает лечение застрахованного лица при возникновении с ним страхового случая, но и, при возникновении каких-либо осложнений после лечения, оплачивает лечение этих осложнений
Агентство недвижимости	Агентство недвижимости занимается покупкой, продажей, сдачей в аренду объектов недвижимости по договорам с их собственниками. Агентство управляет объектами недвижимости как физических, так и юридических лиц. Собственник может иметь несколько объектов. В случае покупки или аренды клиент может произвести осмотр объекта. В качестве одной из услуг, предлагаемых агентством, является проведение инспектирования текущего состояния объекта для адекватного определения его рыночной цены. По результатам своей деятельности агентство производит отчисления в налоговые органы и предоставляет отчетность в органы государственной статистики
Кадровое агентство	Кадровое агентство способствует трудоустройству безработных граждан. Агентство ведет учет и классификацию данных о безработных на основании резюме от них. От предприятий города поступают данные о свободных вакансиях, на основании которых агентство предлагает различные варианты трудоустройства соискателям. В случае положительного исхода поиска вакансия считается заполненной, а безработный становится трудоустроенным. По результатам своей деятельности кадровое агентство производит отчисления в налоговые органы и предоставляет отчетность в органы государственной статистики
Компания по разработке программных продуктов	Компания заключает договор с клиентом на разработку программного продукта согласно техническому заданию. После утверждения технического задания определяется состав и объем работ, составляется предварительная смета. На каждый проект назначается ответственный за его выполнение – куратор проекта, который распределяет нагрузку между программистами и следит за выполнением технического задания. Когда программный продукт готов, то его внедряют, производят обучение клиента и осуществляют дальнейшее сопровождение. По результатам своей деятельности компания производит отчисления в налоговые органы и предоставляет отчетность в органы государственной статистики
Туроператор	Туроператор предоставляет возможность своим клиентам осуществить туристическую или деловую поездку в различные города России и мира. При разработке нового тура сначала анализируется текущая ситуация на рынке туризма и выбирается направление тура. После этого определяется статус тура, бронируются места в гостиницах и билеты на переезд к месту тура, разрабатывается культурная/ деловая/ развлекательная программа, утверждаются сроки тура. На каждый тур назначается ответственное лицо от туроператора, которое будет вести данный тур для улаживания проблем в случае возникновения каких-нибудь чрезвычайных или форс-мажорных ситуаций. Клиент приходит в офис туроператора, где вместе с менеджером выбирает уже разработанный тур и оформляет путевку. После возвращения из тура клиент может высказать свои замечания или пожелания, которые будут учтены при доработке существующих туров или при разработке новых. Также, для дальнейшего улучшения тура, туроператор проводит анализ отчетов от посредников (гостиница, гиды и т.д.). По результатам своей деятельности туроператор производит отчисления в налоговые органы и предоставляет отчетность в органы государственной статистики

Итог работы: отчет, защита работы.

Практическая работа № 3

Цель: приобретения навыков разработки технического заданиям на программный продукт, ознакомиться с правилами написания технического задания

Задание 1:

Ознакомьтесь с информацией

Техни́ческое зада́ние (ТЗ, техзада́ние) - исходный документ для проектирования сооружения или промышленного комплекса, конструирования технического устройства (прибора, машины, системы управления и т. д.), разработки информационных систем, стандартов либо проведения научно-исследовательских работ (НИР).

ТЗ содержит основные технические требования, предъявляемые к сооружению, изделию или услуге и исходные данные для разработки. В ТЗ указываются назначение объекта, область его применения, стадии разработки конструкторской (проектной, технологической, программной и т.п.) документации, её состав, сроки исполнения и т. д., а также особые требования, обусловленные спецификой самого объекта либо условиями его эксплуатации. Как правило, ТЗ составляют на основе анализа результатов предварительных исследований, расчётов и моделирования. Типовые требования к составу и содержанию технического задания приведены в таблице 1.

№ пп	Раздел	Содержание
1	Общие сведения	- полное наименование системы и ее условное обозначение - шифр темы или шифр (номер) договора; - наименование предприятий разработчика и заказчика системы, их реквизиты - перечень документов, на основании которых создается ИС - плановые сроки начала и окончания работ - сведения об источниках и порядке финансирования работ - порядок оформления и предъявления заказчику результатов работ по созданию системы, ее частей и отдельных средств
2	Назначение и цели создания (развития) системы	- вид автоматизируемой деятельности - перечень объектов, на которых предполагается использование системы - наименования и требуемые значения технических, технологических, производственно-экономических и др. показателей объекта, которые должны быть достигнуты при внедрении ИС
3	Характеристика объектов автоматизации	- краткие сведения об объекте автоматизации - сведения об условиях эксплуатации и характеристиках окружающей среды
4	Требования к системе	Требования к системе в целом: - требования к структуре и функционированию системы (перечень подсистем, уровни иерархии, степень централизации, способы информационного обмена, режимы функционирования, взаимодействие со смежными системами, перспективы развития системы) - требования к персоналу (численность пользователей, квалификация, режим работы, порядок подготовки) - показатели назначения (степень приспособляемости системы к изменениям процессов управления и значений параметров) - требования к надежности, безопасности, эргономике, транспортабельности, эксплуатации, техническому обслуживанию и ремонту, защите и сохранности

		информации, защите от внешних воздействий, к патентной чистоте, по стандартизации и унификации Требования к функциям (по подсистемам) : - перечень подлежащих автоматизации задач - временной регламент реализации каждой функции - требования к качеству реализации каждой функции, к форме представления выходной информации, характеристики точности, достоверности выдачи результатов - перечень и критерии отказов Требования к видам обеспечения: - математическому (состав и область применения мат. моделей и методов, типовых и разрабатываемых алгоритмов) - информационному (состав, структура и организация данных, обмен данными между компонентами системы, информационная совместимость со смежными системами, используемые классификаторы, СУБД, контроль данных и ведение информационных массивов, процедуры придания юридической силы выходным документам) - лингвистическому (языки программирования, языки взаимодействия пользователей с системой, системы кодирования, языки ввода- вывода) - программному (независимость программных средств от платформы, качество программных средств и способы его контроля, использование фондов алгоритмов и программ) - техническому - метрологическому - организационному (структура и функции эксплуатирующих подразделений, защита от ошибочных действий персонала) - методическому (состав нормативно- технической документации
5	Состав и содержание работ по созданию системы	- перечень стадий и этапов работ - сроки исполнения - состав организаций — исполнителей работ - вид и порядок экспертизы технической документации - программа обеспечения надежности - программа метрологического обеспечения
6	Порядок контроля и приемки системы	- виды, состав, объем и методы испытаний системы - общие требования к приемке работ по стадиям - статус приемной комиссии
7	Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие	- преобразование входной информации к машиночитаемому виду - изменения в объекте автоматизации - сроки и порядок комплектования и обучения персонала
8	Требования к документированию	- перечень подлежащих разработке документов - перечень документов на машинных носителях
9	Источники разработки	- документы и информационные материалы, на основании которых разрабатывается ТЗ и система

Порядок разработки технического задания

Разработка технического задания выполняется в следующей последовательности. Прежде всего, устанавливают набор выполняемых функций, а также перечень и характеристики исходных данных.

Затем определяют перечень результатов, их характеристики и способы представления.

Далее уточняют среду функционирования программною обеспечения: конкретную комплектацию и параметры технических средств, версию используемой операционной системы и, возможно, версии и параметры другого установленного программного обеспечения, с которым предстоит взаимодействовать будущему программному продукту.

В случаях, когда разрабатываемое программное обеспеченно собирает и хранит некоторую информацию или включается в управление каким-либо техническим процессом, необходимо также четко регламентировать действия программы в случае сбоев оборудования и энергоснабжения.

1. Общие положения

1.1 Техническое задание оформляют в соответствии с ГОСТ 19.106-78 на листах формата А4 и А3 по ГОСТ 2.301-68, как правило, без заполнения полей листа. Номера листов (страниц) проставляют в верхней части листа над текстом.

1.2 Лист утверждения и титульный лист оформляют в соответствии с ГОСТ 19.104-78. Информационную часть (аннотацию и содержание), лист регистрации изменений допускается и в документ не включать.

1.3 Для внесения изменений и дополнений в техническое задание на последующих стадиях разработки программы или программного изделия выпускают дополнение к нему. Согласование и утверждение дополнения к техническому заданию проводят в том же порядке, который установлен для технического задания.

1.4. Техническое задание должно содержать следующие разделы:

- введение;
- наименование и область применения;
- основание для разработки;
- назначение разработки;
- технические требования к программе или программному изделию;
- технико-экономические показатели;
- стадии и этапы разработки;
- порядок контроля и приемки;
- приложения.

В зависимости от особенностей программы или программного изделия допускается уточнять содержание разделов, вводить новые разделы или объединять отдельные из них. При необходимости допускается в техническое задание включать приложения.

2. Содержание разделов

2.1 Введение должно включать краткую характеристику области применения программы или программного продукта, а также объекта (например, системы), в котором предполагается их использовать. Основное назначение введения - продемонстрировать актуальность данной разработки и показать, какое место эта разработка занимает в ряду подобных.

2.2 В разделе «Наименование и область применения» указывают наименование, краткую характеристику области применения программы или программного изделия и объекта, в котором используют программу или программное изделие.

2.3 В разделе «Основание для разработки» должны быть указаны:

- документ (документы), на основании которых ведется разработка. Таким документом может служить план, приказ, договор и т. п.;
- организация, утвердившая этот документ, и дата его утверждения;
- наименование и (или) условное обозначение темы разработки.

2.4 В разделе «Назначение разработки» должно быть указано функциональное и эксплуатационное назначение программы или программного изделия.

2.5 Раздел «Технические требования к программе или программному изделию» должен содержать следующие подразделы:

- требования к функциональным характеристикам;
- требования к надежности;
- условия эксплуатации;
- требования к составу и параметрам технических средств;

- требования к информационной и программной совместимости;
- требования к маркировке и упаковке;
- требования к транспортированию и хранению;
- специальные требования.

В подразделе «Требования к функциональным характеристикам» должны быть указаны требования к составу выполняемых функций, организации входных и выходных данных, временным характеристикам и т. п.

2.5.2 В подразделе «Требования к надежности» должны быть указаны требования к обеспечению надежного функционирования (обеспечение устойчивого функционирования, контроль входной и выходной информации, время восстановления после отказа и т. п.).

2.5.3 В подразделе «Условия эксплуатации» должны быть указаны условия эксплуатации (температура окружающего воздуха, относительная влажность и т. п. для выбранных типов носителей данных), при которых должны обеспечиваться заданные характеристики, а также вид обслуживания, необходимое количество и квалификация персонала.

2.5.4 В подразделе «Требования к составу и параметрам технических средств» указывают необходимый состав технических средств с указанием их технических характеристик.

2.5.5 В подразделе «Требования к информационной и программной совместимости» должны быть указаны требования к информационным структурам на входе и выходе и методам решения, исходным кодам, языкам программирования. При необходимости должна обеспечиваться защита информации и программ.

2.5.6 В подразделе «Требования к маркировке и упаковке» в общем случае указывают требования к маркировке программного изделия, варианты и способы упаковки.

2.5.7 В подразделе «Требования к транспортированию и хранению» должны быть указаны для программного изделия условия транспортирования, места хранения, условия хранения, условия складирования, сроки хранения в различных условиях.

2.5.8 В разделе «Технико-экономические показатели» должны быть указаны: ориентировочная экономическая эффективность, предполагаемая годовая потребность, экономические преимущества разработки по сравнению с лучшими отечественными и зарубежными образцами или аналогами.

2.6 В разделе «Стадии и этапы разработки» устанавливают необходимые стадии разработки, этапы и содержание работ (перечень программных документов, которые должны быть разработаны, согласованы и утверждены), а также, как правило, сроки разработки и определяют исполнителей.

2.7 В разделе «Порядок контроля и приемки» должны быть указаны виды испытаний и общие требования к приемке работы.

2.8 В приложениях к техническому заданию при необходимости приводят:
- перечень научно-исследовательских и других работ, обосновывающих разработку;
- схемы алгоритмов, таблицы, описания, обоснования, расчеты и другие документы, которые могут быть использованы при разработке;
- другие источники разработки.

В случаях, если какие-либо требования, предусмотренные техническим заданием, заказчик не предъявляет, следует в соответствующем месте указать «Требования не предъявляются».

Задание 2:

1. Разработать техническое задание по варианту выбранному в практической работе №1
2. Оформить отчет

Порядок выполнения отчета по практической работе

1.Разработать техническое задание на программный продукт

2. Оформить работу в соответствии с ГОСТ 19.106-78. При оформлении использовать MS Office.

Итог работы: отчет, защита работы.

Практическая работа № 4

Цель: приобретения навыков разработки технического заданиям на программный продукт, ознакомиться с правилами написания технического задания

Задание 1.

Разработайте техническое задание по продукту- сайт ЧГТК им М.И. Щадова

Итог работы: тетрадь, защита работы.

Практическая работа № 5

Цель: приобретение навыков создания формальных моделей и на их основе определение спецификаций разрабатываемого ПО, приобретение навыков проектирования ПО

Задание 1. Ознакомьтесь с информацией

Эскизный проект

Эскизный проект предусматривает разработку предварительных проектных решений по системе и ее частям.

Выполнение стадии эскизного проектирования не является строго обязательной. Если основные проектные решения определены ранее или достаточно очевидны для конкретной ИС и объекта автоматизации, то эта стадия может быть исключена из общей последовательности работ.

Содержание эскизного проекта задается в ТЗ на систему. Как правило, на этапе эскизного проектирования определяются:

- функции ИС;
- функции подсистем, их цели и ожидаемый эффект от внедрения;
- состав комплексов задач и отдельных задач;
- концепция информационной базы и ее укрупненная структура;
- функции системы управления базой данных;
- состав вычислительной системы и других технических средств;
- функции и параметры основных программных средств.

По результатам проделанной работы оформляется, согласовывается и утверждается документация в объеме, необходимом для описания полной совокупности принятых проектных решений и достаточном для дальнейшего выполнения работ по созданию системы.

На основе технического задания (и эскизного проекта) разрабатывается технический проект.

Разработка эскизного проекта программы. Этапы выполнения эскизного проекта.

Эскизный проект	Разработка эскизного проекта	Предварительная разработка структуры входных и выходных данных.
		Уточнение методов решения задачи.
		Разработка общего описания алгоритма решения задачи
		Разработка технико-экономического обоснования.
	Утверждение эскизного проекта	Разработка пояснительной записки.
		Согласование и утверждение эскизного проекта.

Основная задача эскизного проекта – создать прообраз будущей автоматизированной системы. При разработке эскизного проекта разработчик определяет основные контуры будущей системы, а заказчик в свою очередь получает представление об основных чертах будущего объекта автоматизации и анализирует их возможную применимость в последующей работе.

При разработке эскизного проекта составляются:

- Ведомость эскизного проекта. Общая информация по проекту.
- Пояснительная записка к эскизному проекту. Вводная информация, позволяющая ее потребителю быстро освоить данные по конкретному проекту.
- Схема организационной структуры. Описание организационной структуры организации, которая будет использовать создаваемую автоматизированную систему в практической работе.
- Структурная схема комплекса технических средств. Техническая составляющая автоматизированной системы, включающая в себя набор серверов, рабочих станций, схему локальной вычислительной сети и структурированной кабельной системы.
- Схема функциональной структуры. Описание задач, которые будут использоваться в работе подсистем. Видение участков информационной системы и порядок и их взаимодействия.
- Схема автоматизации. Логический процесс создания автоматизированной системы

- Согласно ГОСТ 34.201-89, дополнительно в эскизный проект по необходимости может быть включено техническое задание на разработку новых технических средств.

Эскизный проект чаще всего не разделяют, он выполняется в рамках общего (первоначального) этапа всего проекта. Перечень работ, составляющих эскизный проект, может варьироваться в зависимости от конкретного технического задания заказчика (его пожеланий) и сложности проектируемого проекта. Соответственно варьируется и цена этого этапа.

Эскизный проект не всегда создается под конкретного заказчика. Нередко разработчики с помощью эскизного проекта стремятся показать свой творческий потенциал и найти потенциальных заказчиков. Не случайно на различные конкурсы представляются именно эскизные проекты.

Разработка спецификаций

Разработка программного обеспечения начинается с анализа требований к нему. В результате анализа получают спецификации разрабатываемого программного обеспечения, строят общую модель его взаимодействия с пользователем или другими программами и конкретизируют его основные функции.

При структурном подходе к программированию на этапе анализа и определения спецификаций разрабатывают три типа моделей: модели функций, модели данных и модели потоков данных. Поскольку разные модели описывают проектируемое программное обеспечение с разных сторон, рекомендуется использовать сразу несколько моделей, разрабатываемых в виде диаграмм, и пояснять их текстовыми описаниями, словарями и т. п.

Структурный анализ предполагает использование следующих видов моделей:

- диаграмм потоков данных (DFD - Data Flow Diagrams), описывающих взаимодействие источников и потребителей информации через процессы, которые должны быть реализованы в системе;
- диаграмм «сущность-связь» (ERD Entity-Relationship Diagrams), описывающих базы данных разрабатываемой системы;
- диаграмм переходов состояний (STD - State Transition Diagrams), характеризующих поведение системы во времени;
- функциональных диаграмм (методика SADT);
- спецификаций процессов;
- словаря терминов.

Спецификации процессов

Спецификации процессов обычно представляют в виде краткого текстового описания, схем алгоритмов, псевдокодов, Flow-форм или диаграмм Насси - Шнейдермана.

Словарь терминов

Словарь терминов представляет собой краткое описание основных понятий, используемых при составлении спецификации. Он должен включать определение основных понятий предметной области, описание структур элементов данных, их типом и форматов, а также всех сокращений и условных обозначении.

Диаграммы переходов состояний

С помощью диаграмм переходов состояний можно моделировать последующее функционирование системы на основе ее предыдущего и текущего функционирования. Моделируемая система в любой заданный момент времени находится точно в одном из конечного множества состояний. С течением времени она может изменить свое состояние, при этом переходы между состояниями должны быть точно определены.

Функциональные диаграммы

Функциональные диаграммы отражают взаимосвязи функций разрабатываемого программного обеспечения.

Они создаются на ранних этапах проектирования систем, для того чтобы помочь проектировщику выявить основные функции и составные части проектируемой системы и, по возможности, обнаружить и устранить существенные ошибки. Для создания функциональных диаграмм предлагается использовать методологию SADT.

Диаграммы потоков данных

Для описания потоков информации в системе применяются диаграммы потоков данных (DFD – Data Flow Diagrams). DFD позволяет описать требуемое поведение системы в виде

совокупности процессов, взаимодействующих посредством связывающих их потоков данных. DFD показывает, как каждый из процессов преобразует свои входные потоки данных в выходные потоки данных и как процессы взаимодействуют между собой.

Диаграммы «сущность - связь»

Диаграмма сущность-связь - инструмент разработки моделей данных, обеспечивающий стандартный способ определения данных и отношений между ними. Она включает сущности и взаимосвязи, отражающие основные бизнес-правила предметной области. Такая диаграмма не слишком детализирована, в нее включаются основные сущности и связи между ними, которые удовлетворяют требованиям, предъявляемым к ИС.

Разработка документации. Стадия «Технический проект».

Проект технический - образ намеченного к созданию объекта, представленный в виде его описания, схем, чертежей, расчетов, обоснований, числовых показателей.

Цель технического проекта - определение основных методов, используемых при создании информационной системы, и окончательное определение ее сметной стоимости.

Техническое проектирование подсистем осуществляется в соответствии с утвержденным техническим заданием.

Технический проект программной системы подробно описывает:

- выполняемые функции и варианты их использования;
- соответствующие им документы;
- структуры обрабатываемых баз данных;
- взаимосвязи данных;
- алгоритмы их обработки.

Технический проект должен включать данные об объемах и интенсивности потоков обрабатываемой информации, количестве пользователей программной системы, характеристиках оборудования и программного обеспечения, взаимодействующего с проектируемым программным продуктом.

При разработке технического проекта оформляются:

- ведомость технического проекта. Общая информация по проекту;
- пояснительная записка к техническому проекту. Вводная информация, позволяющая ее потребителю быстро освоить данные по конкретному проекту;
- описание систем классификации и кодирования;
- перечень входных данных (документов). Перечень информации, которая используется как входящий поток и служит источником накопления;
- перечень выходных данных (документов). Перечень информации, которая используется для анализа накопленных данных;
- описание используемого программного обеспечения. Перечень программного обеспечения и СУБД, которые планируется использовать для создания информационной системы;
- описание используемых технических средств. Перечень аппаратных средств, на которых планируется работа проектируемого программного продукта;
- проектная оценка надежности системы. Экспертная оценки надежности с выявлением наиболее благополучных участков программной системы и ее узких мест;
- ведомость оборудования и материалов. Перечень оборудования и материалов, которые потребуются в ходе реализации проекта.

Структурная схема

Структурной называют схему, отражающую состав и взаимодействие по управлению частями разрабатываемого программного обеспечения. Структурная схема определяется архитектурой разрабатываемого ПО.

Функциональная схема

Функциональная схема - это схема взаимодействия компонентов программного обеспечения с описанием информационных потоков, состава данных в потоках и указанием используемых файлов и устройств.

Разработка алгоритмов

Метод пошаговой детализации реализует нисходящий подход к программированию и предполагает пошаговую разработку алгоритма.

Структурные карты

Методика структурных карт используется на этапе проектирования ПО для того, чтобы продемонстрировать, каким образом программный продукт выполняет системные требования. Структурные карты Константайна предназначены для описания отношений между модулями.

Техника структурных карт Джексона основана на методе структурного программирования Джексона, который выявляет соответствие между структурой потоков данных и структурой программы. Основное внимание в методе сконцентрировано на соответствии входных и выходных потоков данных.

Задание 2.

Задание №1

1. На основе технического задания из практической работы №2 выполнить анализ функциональных и эксплуатационных требований к программному продукту.
2. Определить основные технические решения (выбор языка программирования, структура программного продукта, состав функций ПП, режимы функционирования) и занести результаты в документ, называемый «Эскизным проектом».
3. Определить диаграммы потоков данных для решаемой задачи.
4. Определить диаграммы «сущность-связь», если программный продукт содержит базу данных.
5. Добавить словарь терминов.
6. Оформить результаты, используя MS Office или MS Visio в виде эскизного проекта.
7. Сдать и защитить работу.

Порядок выполнения отчета по практической работе

Отчет по лабораторной работе должен состоять из:

1. Постановки задачи.
2. Документа «Эскизный проект», содержащего:

- выбор метода решения и языка программирования;
- спецификации процессов;
- все полученные диаграммы;
- словарь терминов.

Защита отчета по практической работе заключается в предъявлении преподавателю полученных результатов (на экране монитора), демонстрации полученных навыков и ответах на вопросы преподавателя.

Задание №2

1. Разработать функциональную схему программного продукта.
2. Представить структурную схему в виде структурных карт Константайна.
3. Представить структурную схему в виде структурных карт Джексона.
4. Оформить результаты, используя MS Office или MS Visio в виде технического проекта.
5. Сдать и защитить работу.

Порядок выполнения отчета по практической работе

Отчет по практической работе должен состоять из:

1. Структурной схемы программного продукта.
2. Функциональной схемы.
3. Алгоритма программы.
4. Структурной карты Константайна.
5. Структурной карты Джексона.
6. Законченного технического проекта программного модуля.

Итог работы: тетрадь, защита работы.

Практическая работа № 6

Цель: применение методов ООП

Задание 1.

Ознакомьтесь с информацией

Сущность объектно-ориентированного подхода к программированию заключается в том, что основные идеи объектно-ориентированного подхода опираются на следующие положения:

- программа представляет собой модель некоторого реального процесса, части реального мира;
- модель реального мира или его части может быть описана как совокупность взаимодействующих между собой объектов;
- объект описывается набором параметров, значения которых определяют состояние объекта, и набором операций (действий), которые может выполнять объект;
- взаимодействие между объектами осуществляется посылкой специальных сообщений от одного объекта к другому. Сообщение, полученное объектом, может потребовать выполнения определенных действий, например, изменения состояния объекта;
- объекты, описанные одним и тем же набором параметров и способные выполнять один и тот же набор действий представляют собой класс однотипных объектов.

С точки зрения языка программирования класс объектов можно рассматривать как тип данного, а отдельный объект - как данное этого типа. Определение программистом собственных классов объектов для конкретного набора задач должно позволить описывать отдельные задачи в терминах самого класса задач (при соответствующем выборе имен типов и имен объектов, их параметров и выполняемых действий).

Таким образом, объектно-ориентированный подход предполагает, что при разработке программы должны быть определены классы используемых в программе объектов и построены их описания, затем созданы экземпляры необходимых объектов и определено взаимодействие между ними.

Классы объектов часто удобно строить так, чтобы они образовывали иерархическую структуру. Например, класс «Студент», описывающий абстрактного студента, может служить основой для построения классов «Студент 1 курса», «Студент 2 курса» и т.д., которые обладают всеми свойствами студента вообще и некоторыми дополнительными свойствами, характеризующими студента конкретного курса. При разработке интерфейса с пользователем программы могут использовать объекты общего класса «Окно» и объекты классов специальных окон, например, окон информационных сообщений, окон ввода данных и т.п. В таких иерархических структурах один класс может рассматриваться как базовый для других, производных от него классов. Объект производного класса обладает всеми свойствами базового класса и некоторыми собственными свойствами, он может реагировать на те же типы сообщений от других объектов, что и объект базового класса и на сообщения, имеющие смысл только для производного класса. Обычно говорят, что объект производного класса наследует все свойства своего базового класса.

Некоторые параметры объекта могут быть локализованы внутри объекта и недоступны для прямого воздействия извне объекта. Например, во время движения объекта-автомобиля объект-водитель может воздействовать только на ограниченный набор органов управления (рулевое колесо, педали газа, сцепления и тормоза, рычаг переключения передач) и ему недоступен целый ряд параметров, характеризующих состояние двигателя и автомобиля в целом.

Очевидно, для того, чтобы продуктивно применять объектный подход для разработки программ, необходимы языки программирования, поддерживающие этот подход, т.е. позволяющие строить описание классов объектов, образовывать данные объектных типов, выполнять операции над объектами. Одним из первых таких языков стал язык SmallTalk в котором все данные являются объектами некоторых классов, а общая система классов строится как иерархическая структура на основе предопределенных базовых классов.

Опыт программирования показывает, что любой методический подход в технологии программирования не должен применяться слепо с игнорированием других подходов. Это относится и к объектно-ориентированному подходу. Существует ряд типовых проблем, для которых его полезность наиболее очевидна, к таким проблемам относятся, в частности, задачи имитационного моделирования, программирование диалогов с пользователем. Существуют и задачи, в которых применение объектного подхода ни к чему, кроме излишних затрат труда, не приведет. В связи с этим наибольшее распространение получили объектно-ориентированные языки программирования, позволяющие сочетать объектный подход с другими методологиями. В

некоторых языках и системах программирования применение объектного подхода ограничивается средствами интерфейса с пользователем (например, Visual FoxPro ранних версий).

Наиболее используемыми в настоящее время объектно-ориентированными языками являются Паскаль с объектами и Си++, причем наиболее развитые средства для работы с объектами содержатся в Си++.

Объектно-базирующееся программирование - это методология разработки программ, основанная на использовании совокупности объектов, каждый из которых является реализацией определенного класса. Программный код и данные структурируются так, чтобы имитировалось поведение фактически существующих объектов. Содержимое объекта защищено от внешнего мира посредством инкапсуляции. Благодаря наследованию уже запрограммированные функциональные возможности можно использовать и для других объектов. Объекты являются программным представлением физических и/или логических сущностей реального мира. Они необходимы для моделирования поведения физических или логических объектов, которые они представляют. Для изменения поведения и состояния элементов управления используются их свойства, методы, поля и события. Классы задают структуру объектов. При программировании создаются объекты - представители классов. С другой стороны, классы составляют группы одноименных объектов. Внутренняя структура класса в Visual Basic передается объекту с использованием модуля класса. С использованием команды Project →Add Class Module модуль класса можно добавить в проект. После добавления модуля класса выводится окно кода, в котором можно реализовать компоненты (свойства, поля, методы, события) класса.

Задание 2.

Пример использования методики объектно–ориентированного программирования

Создать в предметной области «Автомобили» класс с требуемой функциональностью (использовать компоненты класса: методы, поля и т.д.). Создать объект - экземпляр класса. Создать пример использования объектом компонентов класса.

Реализация задания

Приводится проект, дающий справку желающим приобрести автомобиль. Создан класс Class1, содержащий компоненты, определяющие название фирмы-изготовителя, модель автомобиля, его стоимость, изображение автомобиля и следующие технические характеристики:

- тип двигателя (бензин/дизель),
- число цилиндров/рабочий объём,
- система питания (карбюратор/впрыскивание),
- мощность (л.с),
- максимальная скорость (км/час),
- разгон 0 - 100 (км/час)/сек,
- привод (передний/задний/4x4).

Далее создаётся экземпляр класса: Dim av As New Class1, использующий компоненты класса.

Пользователю предлагается решить вопрос о необходимости покупки, выбрать фирму-изготовителя, ответить на вопрос о выводе изображения покупаемого автомобиля, либо его технических характеристик, либо обеих категорий одновременно (используются процедуры Property Get и Property Let, созданные в классе Class1), после чего программа адекватно реагирует: либо выводятся вышеперечисленные данные, либо выводится некоторое сообщение.

Для реализации проекта нужно выполнить следующую последовательность действий:

1. добавить в стандартный проект модуль класса (Project →Add Class Module →Class Module →Открыть),
2. создать:

- методы класса. Четыре метода создаются в процедурах: Public Function Met1(), Public Function Met2(), Public Function Met3(), Public Function Met4() (Tools →Add Procedure →ввести имя { Met1, Met2, Met3, Met4} → выбрать Function →выбрать Public →OK),
- свойства класса. Свойства задаются с использованием процедур Property Get и Property Let (Tools →Add Procedure ?ввести имя (здесь - varian) →выбрать Property →выбрать Public →OK),
- поля класса - avto, firma, model, stoim, pict, var, см. ниже.

3. создать на форме:

- два элемента управления ComboBox с именами Combo1 и Combo2,
- два элемента управления CommandButton с именами Command1 и Command2; значению свойства Caption объекта Command1 присвоить значение "OK", Command2 - "Exit",
- элементы управления Label1 - Label4, значениям свойств Caption присвоить: Label1 - "Хотите ли Вы купить машину?", Label2 - "Выберите фирму-изготовитель", Label3 - "Хотите ли Вы увидеть изображение выбранного автомобиля или его технические характеристики ?", Label4- "", свойству Visible объекта Label4 присвоить False,
- массив элементов управления OptionButton (присвоить значения свойствам - Option1(0).Caption= "да", Option1(1).Caption= "нет"),
- массивы элементов управления PictureBox: Picture1(0) - Picture1(12) и Picture2(0) - Picture(12). Свойству Visible всех элементов управления присвоить значение False. Свойству Picture каждого элемента управления присвоить значение изображения соответствующего автомобиля и списка технических характеристик (эти списки создаются в приложении Excel, далее таблицы передаются в приложение Paint и сохраняются как рисунки).

4. ввести код в область класса (см. ниже "область проекта Class1"),
5. ввести код, данный ниже, в области:

- General Declarations формы,
- Combo1, событие Click,
- Combo2, событие Click,
- Command1, событие Click,
- Command, событие Click,
- Form, событие Load,
- Form, событие Unload

6. стартовать проект, получить справку о предполагаемой покупке.

```
//////////////////////////область проекта Class1////////////////////////////
Public avto As Boolean
Public firma As String
Public model As String
Public stoim As String
Public pict As String
Dim var As String
Private Sub Class_Initialize() ' инициализация полей класса
avto = False: firma = "": model = "": stoim = "": var = ""
End Sub
Public Function Met1()
' Если пользователь нажал кнопку (OptionButton) - Да, то выполнить процедуры
' Met2, Met3, Met4, результатом выполнения которых является вывод данных:
' марка, стоимость, изображение и технические характеристики, иначе
' Met1 = False и выводится сообщение "Приносим свои извинения, мы даем
' информацию для желающих купить автомобиль"
If avto = True Then
model = Met2()
stoim = Met3()
pict = Met4() ' поле pict определяет номера элементов массивов
' PictureBox, см. Met4
Met1 = True
Else
Met1 = False
End If
End Function
' после щелчка на кнопках Да/Нет (два переключателя OptionButton) и выбора
' фирмы из списка ComboBox с именем Combo1 определить марку автомобиля
Public Function Met2()
Select Case firma
```

```
Case "AUDI": Met2 = "A6"
Case "CITROEN": Met2 = "C5"
Case "FORD": Met2 = "Focus"
Case "HONDA": Met2 = "Accord"
Case "HYUNDAI": Met2 = "Elanta"
Case "JEEP": Met2 = "Grand Cherokee LTD"
Case "LAND ROVER": Met2 = "Land Rover Discovery"
Case "LEXSUS": Met2 = "RX330"
Case "MITSUBISHI": Met2 = "Pajero III"
Case "NISSAN": Met2 = "Primera(1.8)"
Case "PEUGEOT": Met2 = "307 XR"
Case "PORSCHE": Met2 = "Cayenne Turbo"
Case "RENAULT": Met2 = "Laguna II"
End Select
End Function
' определить стоимость автомобиля в долларах США
Public Function Met3()
Select Case firma
Case "AUDI": Met3 = "41500"
Case "CITROEN": Met3 = "20100"
Case "FORD": Met3 = "12430"
Case "HONDA": Met3 = "33900"
Case "HYUNDAI": Met3 = "13790"
Case "JEEP": Met3 = "41690"
Case "LAND ROVER": Met3 = "40850"
Case "LEXSUS": Met3 = "65500"
Case "MITSUBISHI": Met3 = "56640"
Case "NISSAN": Met3 = "25100"
Case "PEUGEOT": Met3 = "13808"
Case "PORSCHE": Met3 = "140500"
Case "RENAULT": Met3 = "22900"
End Select
End Function
Public Function Met4()
' при выборе данных из списка ComboBox с именем Combo2
' (после щелчка на кнопке "OK" ) определяется номер элемента массива
' PictureBox, соответствующий выбранной фирме-изготовителю и
' на экран позднее выводится соответствующая фотография
' и/или технические характеристики автомобиля
  Select Case firma
  Case "AUDI": Met4 = "0"
  Case "CITROEN": Met4 = "1"
  Case "FORD": Met4 = "2"
  Case "HONDA": Met4 = "3"
  Case "HYUNDAI": Met4 = "4"
  Case "JEEP": Met4 = "5"
  Case "LAND ROVER": Met4 = "6"
  Case "LEXSUS": Met4 = "7"
  Case "MITSUBISHI": Met4 = "8"
  Case "NISSAN": Met4 = "9"
  Case "PEUGEOT": Met4 = "10"
  Case "PORSCHE": Met4 = "11"
  Case "RENAULT": Met4 = "12"
  End Select
  End Function
  ' процедура Property Let используется для задания значения свойства,
  ' Property Get - для считывания значения свойства
```

```
Public Property Get varian() As String
Select Case var
Case Is = 0: varian = "pict"
Case Is = 1: varian = "texn"
Case Is = 2: varian = "all"
End Select
End Property
Public Property Let varian(ByVal vNewValue As String)
Select Case vNewValue
Case "изображение": var = 0
Case "технические параметры": var = 1
Case Else: var = 2
End Select
End Property
////////////////////////////область проекта Form1/////////////////////////
Dim av As Class1 ' av - экземпляр класса
Dim v As String
Dim i As Integer, j As Integer
Private Sub Combo1_Click()
' сделать невидимыми элементы управления Label и Picture
' (формирующие фотографии, технические характеристики, фирму,
' марку и стоимость), для того, чтобы впоследствии на форму
' выводились только те из них, которые определяет своими
' действиями покупатель
Label5.Visible = False
For i = 0 To 12
Picture1(i).Visible = False
Picture2(i).Visible = False
Next
Dim ot As String ' переменная для хранения сообщения программы
av.firma = Combo1.Text ' значение поля firma объекта av взять из
' списка ComboBox с именем Combo1
av.avto = Option1(0).Value ' значение поля avto объекта av взять
' из поля массива OptionButton
If av.Met1 = True Then
ot = " " & CStr(av.firma) & vbCrLf: ot = ot & " " & vbCrLf
ot = ot & " модель " & CStr(av.model) & vbCrLf: ot = ot & " " & vbCrLf
ot = ot & " цена в $ " & CStr(av.stoim) & vbCrLf: ot = ot & " " & vbCrLf
ot = ot & "Для получения более полной информации обращайтесь_
по телефону 7077888"
MsgBox Title:="Мы можем предложить", Prompt:=ot
Else
Label5.Visible = False
Picture1(Val(av.pict)).Visible = False ' аргумент Picture1: (av.pict)
' определяет индекс элемента массива PictureBox
ot = "Приносим свои извинения, мы даем информацию для желающих_
купить автомобиль"
MsgBox Title:="Автосалон START", Prompt:=ot
End If
End Sub
Private Sub Combo2_Click()
av.varian = Combo2.Text ' см. процедуру Property Let. Присваиваем
' свойству varian значение выбранные из списка ComboBox с именем Combo2
End Sub
Private Sub Command1_Click()
Label5.Visible = False
Label5.Caption = ""
```

```
For i = 0 To 12
Picture1(i).Visible = False
Picture2(i).Visible = False
Next
v = av.varian ' см. процедуру Property Get. Переменной v присваиваем
' значение свойства varian объекта av
av.avto = Option1(0).Value
If av.Met1 = True Then
Select Case v
Case "pict"
Picture1(Val(av.pict)).Visible = True
Case "texn"
Picture2(Val(av.pict)).Visible = True ' технические характеристики
' хранятся как изображения в соответствующих элементах
' массива PictureBox2
Case "all"
Picture1(Val(av.pict)).Visible = True
Picture2(Val(av.pict)).Visible = True
Label5.Visible = True
Label5.Caption = CStr(av.firma) & " " & CStr(av.model) & _
vbCrLf & "цена в $ " & CStr(av.stoim)
End Select
Else
Picture1(Val(av.pict)).Visible = False
Picture2(Val(av.pict)).Visible = False
MsgBox Title:="Автосалон START", Prompt:="Приносим свои_
извинения, мы даем информацию для желающих купить автомобиль"
End If
End Sub
Private Sub Command2_Click()
MsgBox Title:="Автосалон START", Prompt: = "Мы всегда рады помочь!_
Будем рады новой встрече!"
End ' выход из программы после сообщения MsgBox.
End Sub
Private Sub Form_Load()
Set av = New Class1 ' описание объектной переменной дано выше
' заполнение списка ComboBox с именем Combo1 названиями фирм
Combo1.AddItem "AUDI"
Combo1.AddItem "CITROEN"
Combo1.AddItem "FORD"
Combo1.AddItem "HONDA"
Combo1.AddItem "HYUNDAI"
Combo1.AddItem "JEEP"
Combo1.AddItem "LAND ROVER"
Combo1.AddItem "LEXSUS"
Combo1.AddItem "MITSUBISHI"
Combo1.AddItem "NISSAN"
Combo1.AddItem "PEUGEOT"
Combo1.AddItem "PORSCHE"
' заполнение списка ComboBox с именем Combo2 предложениями для
' выбора данных в процедурах Property Get и Property Let
Combo2.AddItem "изображение"
Combo2.AddItem "технические параметры"
Combo2.AddItem "все данные"
End Sub
Private Sub Form_Unload(Cancel As Integer)
Set av = Nothing ' удалить объект из памяти
End Sub
```

Инструкция пользователя

После старта проекта при отрицательном ответе на вопрос «Хотите ли Вы купить машину?» выводится сообщение: «Приносим свои извинения, мы даем информацию для желающих купить автомобиль».

При положительном ответе на вопрос и выборе фирмы-изготовителя из списка на экран выводится сообщение о фирме, марке и стоимости автомобиля.

Далее покупатель может просмотреть или внешний вид, или технические характеристики, или одновременно обе категории, выбрав соответствующую строку во втором списке и сделав щелчок на кнопке ОК (используются процедуры Property Get и Property Let), где дан результат работы программы при выборе строки «все данные».

При щелчке на кнопке Exit выводится сообщение: «Мы всегда рады помочь! Будем рады новой встрече!» и проводится выход из программы.

Заключение (выводы)

Созданный программный продукт позволяет клиенту получить справочные данные при покупке автомобиля. Представленная программа является лишь небольшим примером использования классов, в реальности же сфера применения свойств объектно-базирующегося программирования гораздо шире.

Задания для выполнения:

1. Для предметной области (выбранной в практической работе №1) выполнить объектно-ориентированное проектирование программного продукта.

Итог работы: тетрадь, защита работы.

Практическая работа № 7

Цель: исследование процесса построения диаграммы прецендентов и диаграмм взаимодействий в заданной предметной области

Задание 1. Ознакомьтесь с материалом

Постановка задачи (описание предметной области).

Магазин осуществляет продажу товаров клиенту путем оформления документов «Заказ». Директор магазина- Антон, принял решение автоматизировать документооборот продаж товара и пригласил для выполнения работ программиста Павла. Поговорив с Антоном, в соответствие с концепцией жизненного цикла (ЖЦ) программы Павел приступил к описанию бизнес процессов, сопровождающих продажу товара. Взяв за основу язык UML, он начал с построения контекстной диаграммы процессов- Use Case diagram. Диаграмма должна ответить на вопрос-«что должно делаться в системе и кто участник этих процессов».

Задание 1. Создание диаграммы вариантов использования и действующих лиц.

Окончательный вид диаграммы показан на рис. 1.

Рис. 1 Диаграмма вариантов использования задачи о заказе товара.

Рис. 1 Диаграмма вариантов использования задачи о заказе товара.

Этапы выполнения

1. Дважды щелкнув мышью на Главной диаграмме Вариантов Использования (Main) в браузере, откройте ее.

2.С помощью кнопки Use Case (Вариант использования) панели инструментов поместите на диаграмму новый вариант использования. Назовите его "Ввести новый заказ".

3. Повторив этапы 2 и 3, поместите на диаграмму остальные варианты использования:

Изменить существующий заказ
Напечатать инвентарную опись
Обновить инвентарную опись
Оформить заказ
Отклонить заказ
Выполнить поставку заказа

4. С помощью кнопки Actor (Действующее лицо) панели инструментов поместите на диаграмму новое действующее лицо.

5. Назовите его "*Продавец*".

6. Повторив шаги 4 и 5, поместите на диаграмму остальных действующих лиц:

Управляющий магазином
Клерк магазина
Бухгалтерская система

7. Создание абстрактного варианта использования (не требующего дальнейшей декомпозиции).

Щелкните правой кнопкой мыши на варианте использования "*Отклонить заказ*" на диаграмме.

В открывшемся меню выберите пункт Open Specification (Открыть спецификацию).

Установите флажок Abstract (Абстрактный), чтобы сделать этот вариант использования абстрактным.

Добавление ассоциаций

1. С помощью кнопки Unidirectional Association (Однонаправленная ассоциация) панели инструменте нарисуйте ассоциацию между действующим лицом *Продавец* и вариантом использования "*Ввести заказ*".

2. Повторив шаг 1, поместите на диаграмму остальные ассоциации, согласно рис. 1.

Добавление связи расширения

С помощью кнопки Generalization (Обобщение) панели инструментов нарисуйте связь между вариантом использования "*Отклонить заказ*" и вариантом использования "*Оформить заказ*". Стрелка должна быть направлена от первого варианта использования ко второму. Связь расширения означает, что вариант использования "*Отклонить заказ*" при необходимости дополняет функциональные возможности варианта использования "*Оформить заказ*".

Щелкните правой кнопкой мыши на новой связи между вариантами использования "*Отклонить заказ*" и "*Оформить заказ*".

В открывшемся меню выберите пункт Open Specification (Открыть спецификацию).

В раскрывающемся списке стереотипов введите слово extends (расширение), затем нажмите ОК.

Надпись «extends» появится на линии данной связи.

Добавление описаний к вариантам использования

Выделите в браузере вариант использования "*Ввести новый заказ*".

В окне документации введите следующее описание: " Этот вариант использования дает клиенту возможность ввести новый заказ в систему".

С помощью окна документации добавьте описания ко всем остальным вариантам использования.

Добавление описаний к действующему лицу

Выделите в браузере действующее лицо *Продавец*.

В окне документации введите следующее описание: "Продавец — это служащий, старающийся продать товар".

С помощью окна документации добавьте описания к остальным действующим лицам.

Задание 2.

Согласовав основные бизнес процессы с Антоном, Павел приступил к построению модели бизнес - процессов, что бы ответить на вопрос - «как это должно делаться в системе». Для начала он выбрал наиболее важный Вариант использования - «Ввод нового заказа» и построил для него диаграммы взаимодействия.

Диаграммы взаимодействия включают в себя два типа диаграмм - Последовательности и Кооперативную.

Этапы выполнения

Настройка программной среды

1. В меню модели выберите пункт Tools >- Options (Инструменты >- Параметры).
2. Перейдите на вкладку Diagram (Диаграмма).
3. Установите флажки Sequence numbering, Collaboration numbering и Focus of control.
4. Нажмите OK, чтобы выйти из окна параметров.

Создание диаграммы Последовательности

1. Щелкните правой кнопкой мыши на Логическом представлении браузера.
2. В открывшемся меню выберите пункт New > Sequence Diagram (Создать >Диаграмма Последовательности).
3. Назовите новую диаграмму: Ввод заказа.
4. Дважды щелкнув на этой диаграмме, откройте ее.

Добавление на диаграмму действующего лица и объектов

1. Перетащите действующее лицо *Продавец* из браузера на диаграмму.
2. Нажмите кнопку Object (Объект) панели инструментов.
3. Щелкните мышью в верхней части диаграммы, чтобы поместить туда новый объект.
4. Назовите объект *Выбор варианта заказа*.
5. Повторив шаги 3 и 4, поместите на диаграмму объекты:

- *Форма деталей заказа*
- *Заказ №1234*

Добавление сообщений на диаграмму

1. На панели инструментов нажмите кнопку Object Message (Сообщение объекта).
2. Проведите мышью от линии жизни действующего лица *Продавец* к линии жизни объекта *Выбор варианта заказа*.
3. Выделив сообщение, введите его имя — *Создать новый заказ*.
4. Повторив шаги 2 и 3, поместите на диаграмму сообщения:

- *Открыть форму* — между *Выбор Варианта Заказа* и *Форма деталей Заказа*
- *Ввести номер заказа, заказчика и число заказываемых предметов* — между *Продавец* и *Форма Деталей Заказа*
- *Сохранить заказ* — между *Продавец* и *Форма Деталей Заказа*
- *Создать пустой заказ* — между *Форма Деталей Заказа* и *Заказ N1234*
- *Ввести номер заказа, заказчика и число заказываемых предметов* — между *Форма Деталей Заказа* и *Заказ N1234*
- *Сохранить заказ* — между *Форма Деталей Заказа* и *Заказ N1234*

Завершен первый этап работы. Готовая диаграмма Последовательности представлена на рис. 2.

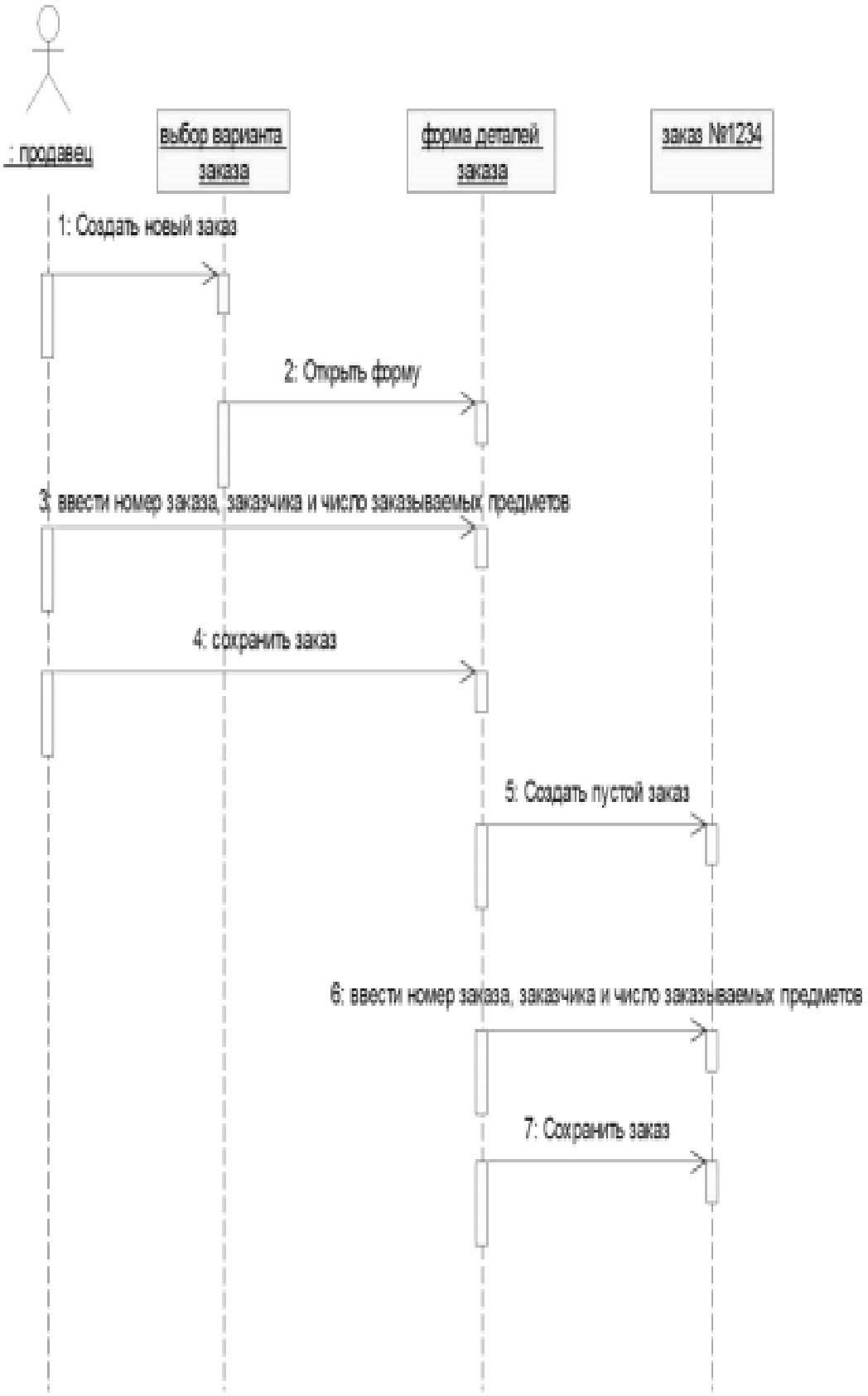


Рис. 2. Диаграмма последовательности без управляющих элементов.

Теперь нужно позаботиться об управляющих объектах и о взаимодействии с базой данных. Как видно из диаграммы, объект *Форма Деталей Заказа* имеет множество ответственностей, с которыми лучше всего мог бы справиться управляющий объект. Кроме того, новый заказ должен сохранять себя в базе данных сам. Вероятно, эту обязанность лучше было бы переложить на другой объект.

Добавление на диаграмму дополнительных объектов

1. Нажмите кнопку Object панели инструментов.

2. Щелкните мышью между объектами *Форма Деталей Заказа* и *Заказ №1234*, чтобы поместить туда новый объект.

3. Введите имя объекта — *Управляющий заказами*.

4. Нажмите кнопку Object панели инструментов.

5. Новый объект расположите справа от *Заказ №1234*.

6. Введите его имя- *Управляющий транзакциями*.

Назначение ответственностей объектам

1. Выделите сообщение 5: *Создать пустой заказ*.

2. Нажав комбинацию клавиш CTRL+D, удалите это сообщение.

3. Повторите шаги 1 и 2 для удаления двух последних сообщений:

- *Вести номер заказа, заказчика и число заказываемых предметов*
- *Сохранить заказ*

4. Нажмите кнопку Object Message панели инструментов.

5. Поместите на диаграмму новое сообщение, расположив его под сообщением 4 между *Форма деталей заказа* и *Управляющий заказами*.

6. Назовите его *Сохранить заказ*.

7. Повторите шаги 4 — 6, добавив сообщения с шестого по девятое и назвав их:

- *Создать новый заказ* — между *Управляющий заказами* и *Заказ №1234*
- *Ввести номер заказа, заказчика и число заказываемых предметов*- между *Управляющий заказами* и *Заказ №1234*
- *Сохранить заказ*- между *Управляющий заказами* и *Управляющий транзакциями*
- *Информация о заказе* —между *Управляющий транзакциями* и *Заказ №1234*

8. На панели инструментов нажмите кнопку Message to Self (Сообщение себе).

9. Щелкните на линии жизни объекта *Управляющий транзакциями* ниже сообщения 9, добавив туда рефлексивное сообщение.

10. Назовите его *Сохранить информацию о заказе в базе данных*.

Соотнесение объектов с классами

1. Щелкните правой кнопкой мыши на объекте *Выбор варианта заказа*.

2. В открывшемся меню выберите пункт Open Specification (Открыть спецификацию).

В раскрывающемся списке классов выберите пункт <New> (Создать). Появится окно спецификации классов.

4. В поле Name введите *Выбор заказа*.

5. Щелкните на кнопке OK. Вы вернетесь в окно спецификации объекта.

6. В списке классов выберите класс *Выбор Заказа*.

7. Щелкните на кнопке OK, чтобы вернуться к диаграмме. Теперь объект называется *Выбор варианта заказа: Выбор Заказа*

8. Для соотнесения остальных объектов с классами повторите шаги с 1 по 7:

- Класс *Детали заказа* соотнесите с объектом *Форма деталей заказа*
- Класс *Упр_заказами* — с объектом *Управляющий заказами*
- Класс *Заказ* — с объектом *Заказ N 1234*
- Класс *Упр_ транзакциями* — с объектом *Управляющий транзакциями*

Соотнесение сообщений с операциями

1. Щелкните правой кнопкой мыши на сообщении 1: *Создать новый заказ*.

2. В открывшемся меню выберите пункт <new operation> (создать операцию). Появится окно спецификации операции.

3. В поле Name введите имя операции —*Создать*.

4. Нажмите на кнопку OK, чтобы закрыть окно спецификации операции и вернуться к диаграмме.

5. Еще раз щелкните правой кнопкой мыши на сообщении 1.

6. В открывшемся меню выберите новую операцию *Создать()*.

7. Повторите шаги с 1 по 6, чтобы соотнести с операциями все остальные сообщения:

- Сообщение 2: *Открыть форму* соотнесите с операцией *Открыть()*
- Сообщение 3: *Ввести номер заказа, заказчика и число заказываемых предметов* — операцией *Ввести номер заказа, заказчика и число заказываемых предметов()*
- Сообщение 4: *Сохранить заказ* — с операцией *Сохранить заказ()*
- Сообщение 5: *Сохранить заказ* — с операцией *Сохранить заказ()*
- Сообщение 6: *Создать пустой заказ* – с операцией *Создать пустой заказ()*
- Сообщение 7: *Ввести номер заказа, заказчика и число заказоваемых предметов*- одноименной операцией.
- Сообщение 8 *Сохранить заказ* – с операцией *Сохранить заказ()*
- Сообщение 9 *Информация о заказе* – с одноименной операцией
- Сообщение 10 *Сохранить информацию о заказе* с одноименной операцией.

Диаграмма должна выглядеть, как на рис. 3.

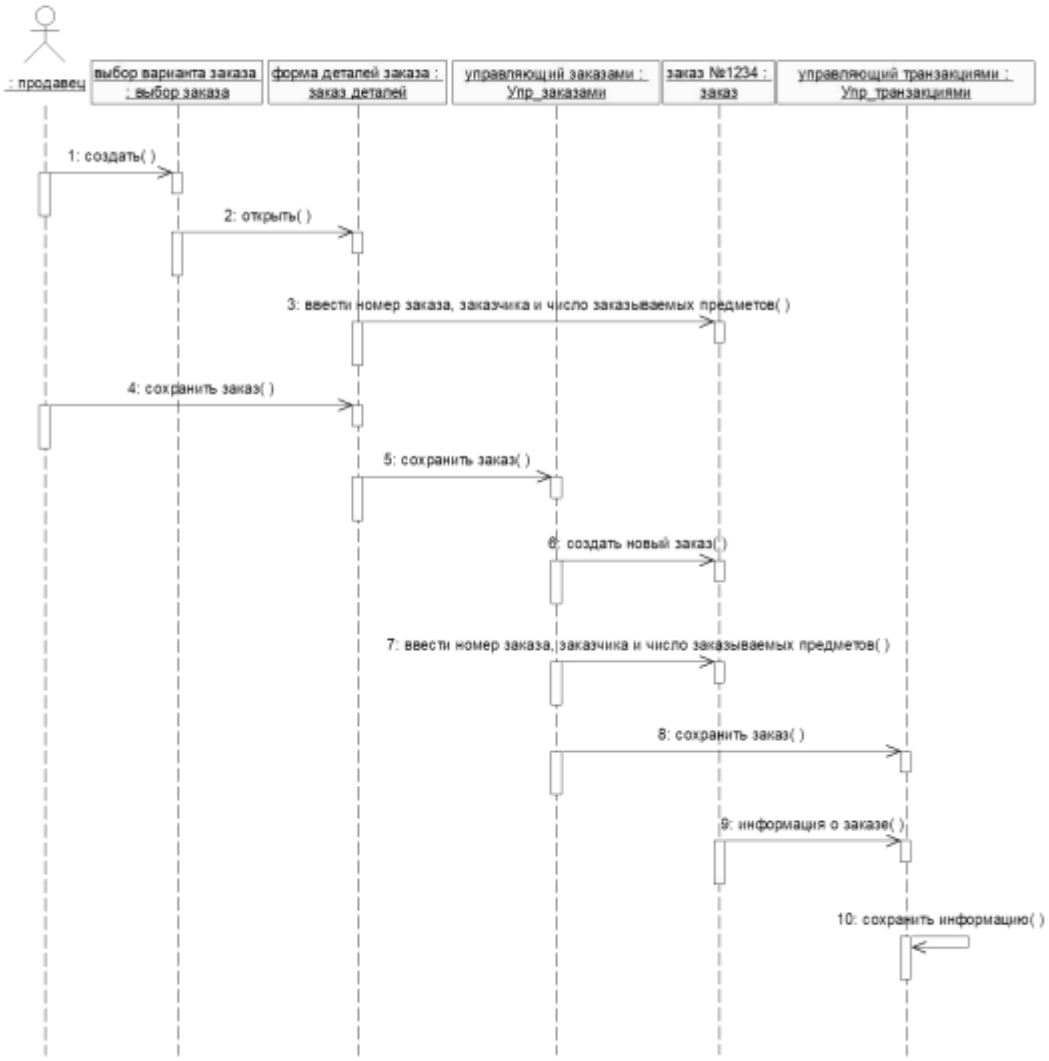


Рис. 3. Окончательный вид диаграммы последовательности

Итог работы: тетрадь, защита работы.

Практическая работа № 8

Цель: исследование моделирования процессов, описывающих взаимодействие объектов в диаграмме кооперации и диаграмме развёртывания в заданной предметной области

Задание 1.

Задание 1. Построение Создание Кооперативной диаграммы

Конечный вид диаграммы представлен на рис. 4.

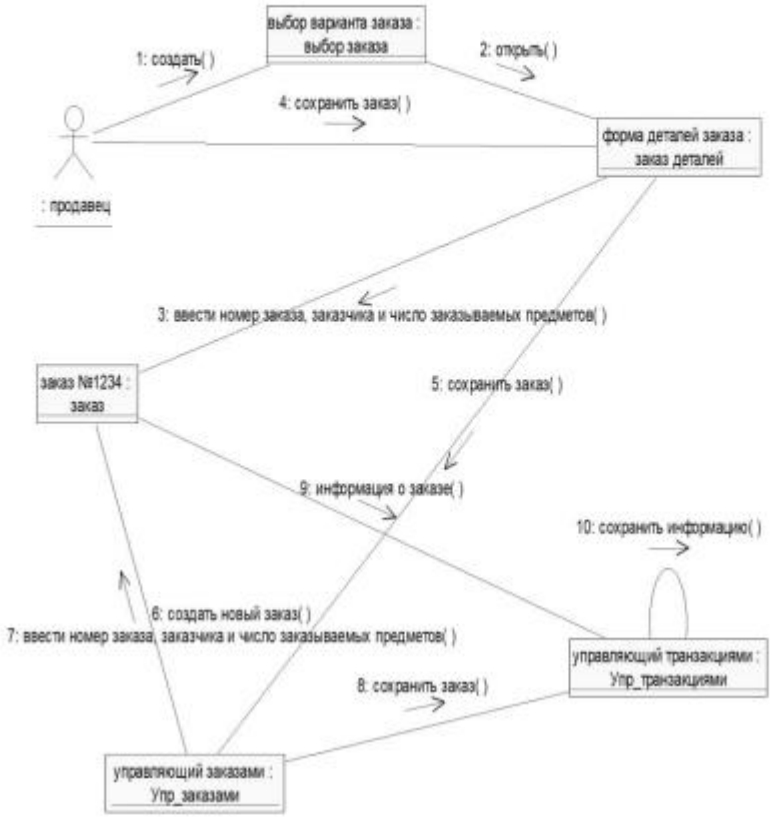


Рис. 4. Окончательный вид кооперативной диаграммы.

1. Щелкните правой кнопкой мыши на Логическом представлении в браузере.
2. В открывшемся меню выберите пункт New > Collaboration Diagram (Создать > Кооперативная диаграмма).
3. Назовите эту диаграмму *Ввод заказа*.
4. Дважды щелкнув мышью на диаграмме, откройте ее.

Добавление действующего лица и объектов на диаграмму

1. Перетащите действующее лицо *Продавец* из браузера на диаграмму.
2. Нажмите кнопку Object (Объект) панели инструментов.
3. Щелкните мышью где-нибудь внутри диаграммы, чтобы поместить туда новый объект.
4. Назовите объект *Выбор варианта заказа*.
5. Повторив шаги 3 и 4, поместите на диаграмму объекты:

- *Форма деталей заказа*
- *Заказ №1234*

Добавление сообщений на диаграмму

1. На панели инструментов нажмите кнопку Object Link (Связь объекта).
2. Проведите мышью от действующего лица *Продавец* к объекту *Выбор варианта заказа*.
3. Повторите шаги 1 и 2, соединив связями следующие объекты:
   - Действующее лицо *Продавец* и объект *Форма деталей Заказа*
   - Объект *Форма деталей Заказа* и объект *Выбор Варианта Заказа*
   - Объект *Форма деталей Заказа* объект *Заказ N1234*
4. На панели инструментов нажмите кнопку Link Message (Сообщение связи).
5. Щелкните мышью на связи между *Продавец* и *Форма деталей Заказа.*

6. Выделив сообщение, введите его имя —*Создать новый заказ*;

7. Повторив шаги с 4 по 6, поместите на диаграмму сообщения:

- *Открыть форму* — между *Выбор Варианта Заказа* и *Форма Деталей Заказа*.

- *Ввести номер заказа, заказчика и число заказываемых предметов* — между *Продавец* и *Форма Деталей Заказа*

- *Сохранить заказ* — между *Продавец* и *Форма деталей Заказа*

- *Создать пустой заказ* — между *Форма деталей Заказа* и *Заказ №1234*

- Ввести номер заказа, заказчика и число заказываемых предметов — между *Форма деталей Заказа* и *Заказ №1234*

- Сохранить заказ — между *Форма деталей Заказа* и *Заказ №1234*

Теперь нужно поместить на диаграмму дополнительные элементы, а также рассмотреть ответственности объектов.

Добавление на диаграмму дополнительных объектов

1. Нажмите кнопку Object панели инструментов.

2. Щелкните мышью где-нибудь на диаграмме, чтобы поместить туда новый объект.

3. Введите имя объекта — *Управляющий заказами*.

4. На панели инструментов нажмите кнопку Object.

5. Поместите на диаграмму еще один объект.

6. Введите его имя — *Управляющий транзакциями*.

Назначение ответственностей объектам

1. Выделите сообщение 5: *Создать пустой заказ*. Выделяйте слова, а не стрелку.

2. Нажав комбинацию клавиш CTRL+D, удалите это сообщение.

3. Повторите шаги 1 и 2 для удаления сообщений 6 и 7:

- *Ввести номер заказа, заказчика и число заказываемых предметов*

- *Сохранить заказ*

4. Выделите связь между объектами *Форма деталей Заказа* и *Заказ №1234*

5. Нажав комбинацию клавиш CTRL+D, удалите эту связь

6. На панели инструментов нажмите кнопку Object Link (Связь объекта).

7. Нарисуйте связь между *Форма деталей Заказа* и *Управляющий Заказами*.

8. На панели инструментов нажмите кнопку Object Link (Связь объекта).

9. Нарисуйте связь между *Управляющий Заказами* и *Заказ №1234*

10. На панели инструментов нажмите кнопку Object Link (Связь объекта).

11. Нарисуйте связь между *Заказ №1234* и *Управляющий Транзакцией*.

12. На панели инструментов нажмите кнопку Object Link (Связь объекта).

13. Нарисуйте связь между *Управляющий Заказами* и *Управляющий Транзакцией*.

14. На панели инструментов нажмите кнопку Link Message (Сообщение связи).

15. Щелкните мышью на связи между объектами *Форма деталей Заказа* и *Управляющий Заказами*, чтобы ввести новое сообщение.

16. Назовите это сообщение Сохранить заказ.

17. Повторите шаги 14 — 16, добавив сообщения с шестого по девятое, и назвав их:

17. Повторите шаги 14 — 16, добавив сообщения с шестого по девятое, и назвав их:

- *Создать новый заказ* — между *Управляющий Заказами* и *Заказ №1234*

- *Ввести номер заказа, заказчика и число заказываемых предметов* — между *Управляющий Заказами* и *Заказ №1234*

- *Сохранить заказ* — между *Управляющий Заказами* и *Управляющий Транзакцией*

- *Информация о заказе* — между *Управляющий Транзакцией* и *Заказ №1234*

18. На панели инструментов нажмите кнопку Link to Self (Связь с собой).

19. Щелкнув на объекте *Управляющий Транзакцией*, добавьте к нему рефлексивное сообщение.

20. На панели инструментов нажмите кнопку Link Message (Сообщение связи).

21. Щелкните мышью на рефлексивной связи *Управляющий Транзакциями*, чтобы ввести туда сообщение.

22. Назовите новое *Сохранить информацию о заказе в базе данных*.

Соотнесение объектов с классами (если классы были созданы при разработке описанной выше диаграммы Последовательности)

1. Найдите в браузере класс *Выбор Заказа*.

2. Перетащите его на объект *Выбор варианта заказа* на диаграмме.

3. Повторите шаги 1 и 2 соотнеся остальные объекты и соответствующие им классы:

- Класс *заказ деталей* соотнесите с объектом *Форма деталей заказа*
- Класс *Упр_заказами* — с объектом *Управляющий Заказами*
- Класс *Заказ* — с объектом *Заказ №1234*
- Класс *Упр_транзакциями* — с объектом *Управляющий транзакциями*

Соотнесение объектов с классами (если вы не создавали описанную выше диаграмму Последовательности)

1. Щелкните правой кнопкой мыши на объекте *Форма деталей Заказа*.
2. В открывшемся меню выберите пункт Open Specification (Открыть спецификацию).
3. В раскрывающемся списке классов выберите пункт <New> (Создать). Появится окно спецификации классов.
4. В поле имени введите *Выбор заказа*.
5. Щелкните на кнопке ОК. Вы вернетесь в окно спецификации объекта.
6. В списке классов выберите класс *Выбор заказа.*
7. Щелкните на кнопке ОК, чтобы вернуться к диаграмме. Теперь объект называется *Выбор варианта заказа: Выбор Заказа*
8. Для соотнесения остальных объектов с классами повторите шаги с 1 по 7:

- Класс *Детали заказа* соотнесите с объектом *Форма деталей заказа*
- Класс *Упр_заказами* — с объектом *Управляющий заказами*
- Класс *Заказ* — с объектом *Заказ N 1234*
- Класс *Упр_ транзакциями* — с объектом *Управляющий транзакциями*

Соотнесение сообщений с операциями (если операции были созданы при разработке описанной выше диаграммы Последовательности)

1. Щелкните правой кнопкой мыши на сообщении 1: Создать новый заказ.
2. В открывшемся меню выберите пункт Open Specification (Открыть спецификацию).
3. В раскрывающемся списке имен укажите имя операции — *Создать()*.
4. Нажмите на кнопку ОК.
5. Повторите шаги 1—4 для соотнесения с операциями остальных сообщений:

- Сообщение 2: *Открыть форму* соотнесите с операцией *Открыть()*
- Сообщение 3: *Ввести номер заказа, заказчика и число заказываемых предметов* — с операцией *Ввести номер заказа, заказчика и число заказываемых предметов()*
- Сообщение 4: *Сохранить заказ* — с операцией *Сохранить заказ()*
- Сообщение 5: *Сохранить заказ* — с операцией Сохранить заказ()

Сообщение 6: *Создать пустой заказ* – с операцией Создать пустой заказ()

Сообщение 7: *Ввести номер заказа, заказчика и число заказоваемых предметов-* с одноименной операцией.

Сообщение 8 Сохранить заказ – с операцией Сохранить заказ()

Сообщение 9 *Информация о заказе* – с одноименной операцией

Сообщение 10 *Сохранить информацию о заказе* с одноименной операцией

Соотнесение сообщений с операциями (если вы не создавали описанную выше диаграмму Последовательности)

1. Щелкните правой кнопкой мыши на сообщении 1: *Создать новый заказ()*.
2. В открывшемся меню выберите пункт <new operation> (создать операцию). Появится окно спецификации операции.
3. В поле имени введите имя операции —*Создать()*.
4. Нажмите на кнопку ОК, чтобы закрыть окно спецификации операции и вернуться к диаграмме.
5. Еще раз щелкните правой кнопкой мыши на сообщении 1.
6. В открывшемся меню выберите пункт Open Specification (Открыть спецификацию).
7. В раскрывающемся списке Name (Имя) укажите имя новой операции.
8. Нажмите на кнопку ОК.
9. Повторите шаги 1—8, чтобы создать новые операции и соотнести с ними остальные

- Сообщение 2: *Открыть форму* соотнесите с операцией *Открыть()*
- Сообщение 3: *Ввести номер заказа, заказчика и число заказываемых предметов* — с операцией *Ввести номер заказа, заказчика и число заказываемых предметов()*
- Сообщение 4: *Сохранить заказ* — с операцией *Сохранить заказ()*
- Сообщение 5: *Сохранить заказ* — с операцией *Сохранить заказ()*

Сообщение 6: *Создать пустой заказ* – с операцией *Создать пустой заказ()*

Сообщение 7: *Ввести номер заказа, заказчика и число заказоваемых предметов-* с одноименной операцией.

Сообщение 8 Сохранить заказ – с операцией *Сохранить заказ()*

Сообщение 9 *Информация о заказе* – с одноименной операцией

Сообщение 10 *Сохранить информацию о заказе* с одноименной операцией

Ваша диаграмма должна выглядеть, как показано на рис. 4

Итог работы: тетрадь, защита работы.

Практическая работа № 9

Цель: исследование процесса построения диаграммы состояний и диаграммы классов в заданной предметной области

Задание 1. Создайте страницу следующего вида:

Задание 1. Постройте диаграмму Состояний

Постройте диаграмму Состояний для класса *Заказ*, показанную на рис. 5.

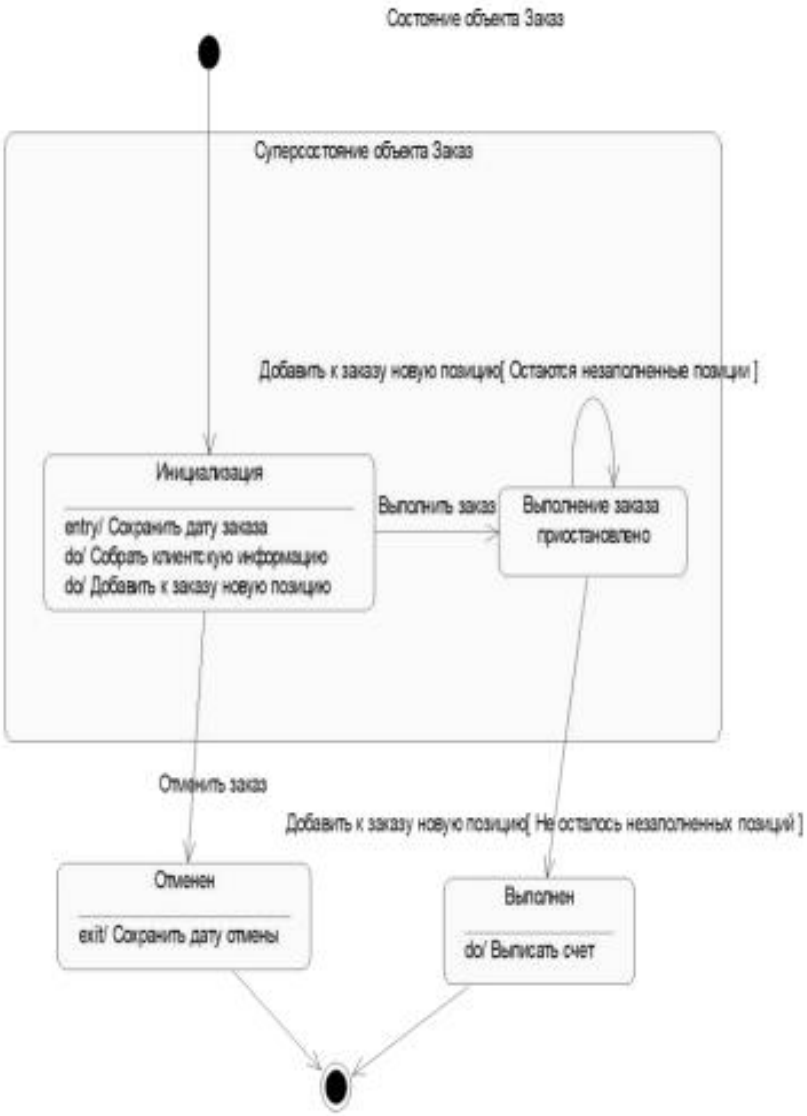


Рис 5. Диаграмма состояний для класса *Заказ*

Этапы выполнения

Создание диаграммы

1. Найдите в браузере класс *Заказ*.
2. Щелкните на классе правой кнопкой мыши и в открывшемся меню укажите пункт New > Statechart Diagram (Создать диаграмму состояний).

Добавление начального и конечного состояний

1. Нажмите кнопку Start State (Начальное состояние) панели инструментов.
2. Поместите это состояние на диаграмму.
3. Нажмите кнопку End State (Конечное состояние) панели инструментов.
4. Поместите это состояние на диаграмму.

Добавление суперсостояния

1. Нажмите кнопку State (Состояние) панели инструментов.
2. Поместите это состояние на диаграмму.

Добавление оставшихся состояний

1. На панели инструментов нажмите кнопку State (Состояние).
2. Поместите состояние на диаграмму.
3. Назовите состояние *Отменен*.
4. На панели инструментов нажмите кнопку State(Состояние).
5. Поместите состояние на диаграмму.
6. Назовите состояние *Выполнен*.
7. На панели инструментов нажмите кнопку State(Состояние).
8. Поместите состояние на диаграмму внутрь суперсостояния.
9. Назовите состояние *Инициализация*.
10. На панели инструментов нажмите кнопку State (Состояние).
11. Поместите состояние на диаграмму внутрь суперсостояния.
12. Назовите состояние *Выполнение заказа приостановлено*.

Описание состояний

1. Дважды щелкните мышью на состоянии *Инициализация*.
2. Перейдите на вкладку Detail (Подробно).
3. Щелкните правой кнопкой мыши в окне Actions(Действия).
4. В открывшемся меню выберите пункт Insert(Вставить).
5. Дважды щелкните мышью на новом действии.
6. Назовите его *Сохранить дату заказа*.
7. Убедитесь, что в окне When (Когда) указан пункт On Entry (На входе).
8. Повторив шаги 3—7, добавьте следующие действия:

- *Собрать клиентскую информацию*, в окне When укажите DO (Выполнять между входом и выходом)
- *Добавить к заказу новые позиции*, укажите DO

9. Нажмите два раза на OK, чтобы закрыть спецификацию.
10. Дважды щелкните мышью на состоянии *Отменен*.
11. Повторив шаги 2—7, добавьте действия:

Сохранить дату отмены, укажите On Exit (На выходе)

12. Нажмите два раза на OK, чтобы закрыть спецификацию.
13. Дважды щелкните мышью на состоянии *Выполнен*.
14. Повторив шаги 2—7, добавьте действие:

- *Выписать счет*, укажите On Exit

15. Нажмите два раза на OK, чтобы закрыть спецификацию.

Добавление переходов

1. Нажмите кнопку Transition (Переход) панели инструментов.
2. Щелкните мышью на начальном состоянии.
3. Проведите линию перехода к состоянию *Инициализация*.
4. Повторив шаги с первого по третий, создайте следующие переходы:

- От состояния Инициализация к состоянию *Выполнение заказа приостановлено*
- От состояния *Выполнение заказа приостановлено* к состоянию *Выполнен*
- От суперсостояния к состоянию *Отменен*
- От состояния *Отменен* к конечному состоянию
- От состояния *Выполнен* к конечному состоянию

5. На панели инструментов нажмите кнопку Transition to Self (Переход к себе).
6. Щелкните мышью на *состоянии Выполнение заказа приостановлено*

Описание переходов

1. Дважды щелкнув мышью на переходе от состояния *Инициализация* к состоянию *Выполнение заказа приостановлено*, откройте окно спецификации перехода.

2. В поле Event (Событие) введите фразу *Выполнить заказ*.

3. Щелкнув на кнопке OK, закройте окно спецификации.

4. Повторив шаги с первого по третий, добавьте событие *Отменить заказ* к переходу между суперсостоянием и состоянием *Отменен*.

5. Дважды щелкнув мышью на переходе от состояния *Выполнение заказа приостановлено* к состоянию *Выполнен*, откройте окно его спецификации.

6. В поле Event (Событие) введите фразу *Добавить к заказу новую позицию*.

7. Перейдите на вкладку Detail (Подробно).

8. В поле Guard Condition (Сторожевое Условие) введите *Не осталось незаполненных позиций*.

9. Щелкнув на кнопке OK, закройте окно спецификации.

10. Дважды щелкните мышью на рефлексивном переходе (Transition to Self) состояния *Выполнение заказа приостановлено*.

11. В поле Event (Событие) введите фразу *Добавить к заказу новую позицию*.

12. Перейдите на вкладку Detail (Подробно).

13. В поле Guard Condition (Сторожевое Условие) введите *Остаются незаполненные позиции*.

14. Щелкнув на кнопке OK, закройте окно спецификации.

Задание 2. Построение диаграммы Активности для варианта использования «Выполнить поставку Заказа».

Побеседовав с Павлом, Антон понял, что необходимо согласовать логику реализации еще одного варианта использования «Выполнить поставку заказа». Стало ясно, что здесь возможны несколько альтернативных потоков управления. Для таких ситуаций более удобно использовать не диаграммы взаимодействия, приспособленные для единственного потока, а диаграмму активности.

Описание варианта использования.

При оформлении заказа проверяют каждую содержащуюся в нем позицию, чтобы убедиться в наличии соответствующих товаров на складе. После этого выписываются товары для реализации заказа. Во время выполнения этих процедур одновременно проверяется прохождение платежа. Если платеж прошел, и товары имеются на складе, то осуществляется их поставка. Если платеж прошел, но товары на складе отсутствуют, то заказ ставится в ожидание. Если платеж не прошел, то заказ аннулируется.

Этапы выполнения

1. Найдите в браузере вариант использования «Выполнить поставку заказа»

2. Щелкните на классе правой кнопкой мыши и в открывшемся меню укажите пункт New > Activity Diagram (Создать диаграмму активности).

3. Назвите диаграмму «Выполнить поставку» и откройте ее двукратным щелчком мыши

4. На панели инструментов щелкните мышкой на элементе Swimline, затем на поле диаграммы. На диаграмме появится разделительная линия («водная дорожка»).

5. Установите курсор на заголовок NewSwinlane и нажмите правую клавишу мыши. В выпадающем списке нажмите Select in brousere. В браузере выделится этот объект. Нажав правую клавишу мыши в выпадающем списке выберете Open Specefication и откройте спецификацию. Измените поле Name на *Клерк*. Выберите в поле Class *Клерк в магазине*.

6. Выполните заново пункты 5-6 и присвойте полю Name *Система*, Class- *Бухгалтерская система*.

7. Найдите в браузере сплошной черный кружок (начальное состояние). Перенесите его на дорожку *Клерк*.

8. Выберете из панели инструментов объект Activity и поместите его на диаграмму в "дорожку" *Клерк*. Измените имя объекта на "*Получить заказ*".

9. Повторите предыдущий этап, создайте на «дорожке» *Клерк* 4 новых Activity и присвойте им имена *Проверить позицию заказа, закрепить позицию за заказом, Поставить заказ в ожидание, Скомплектовать заказ*

10. Поместите на «дорожку» 2 новых объекта End State (конечное состояние). Одному из них измените поле Name на «*Выполнить поставку*»

11. На дорожку *Система* поместите новый объект Activity и присвойте полю Name "*Проверить платеж*". На эту же дорожку поместите новый объект End State и измените в его спецификации поле Name на «*Отменить заказ*».

12. Поместить на «дорожку» Клерк 2 объекта Horisontal Sinhronization (горизонтальная синхронизация). Присвойте полю Name спецификации одного объекта «*1*», другого- «*2*».

13. Поместить на «дорожку» *Клерк* объект Dicision (выбор). Через спецификацию присвойте полю Name «*Позиция имеется?*».

14. Поместить на «дорожку» *Система* объект Dicision. Присвойте полю Name «*Деньги поступили?*».

15. Щелкните мышкой на панели инструментов объекте- стрелке State Transition (состояние перехода). Затем щелкните мышкой на диаграмме объекта начальное состояние. Удерживая кнопку мыши перенесите курсор на активность "*Получить заказ*" и лишь затем отпустить курсор. В результате два объекта будут соединены стрелкой.

16. Выполните этап 14, соединив стрелкой объект Активность «*Получить заказ*» с объектом Horisontal Sinhronization 1.

17. Соедините этими же стрелками объекты 1 и «Проверить платеж», 1 и «Проверить позицию заказа», «Проверить заказ» и «Деньги подступили?», «Деньги поступили?» и «Отменить заказ», «Проверить позицию заказа» и «Позиция имеется», «Позиция имеется» и «Закрепить

позицию за заказом», «Деньги получены?» и 2, «Закрепить позицию за заказом» и 2, «Позиция имеется?» и «Поставить заказ в ожидание», 2 и «Скомплектовать заказ», «Скомплектовать заказ» и «Выполнить поставку», «Поставить заказ в ожидание» и объект Конечное состояние (без имени).

18. Присвоим некоторым стрелкам наименование полю Event (условие перехода).Для этого, установим курсор на стрелке, соединяющей «Деньги получены?» и «Отменить заказ». Двукратным щелчком мыши откроем окно спецификации. В поле Event введем «Нет».

19. Выполним пункт 18 для стрелки, соединяющей «Деньги получены?» и 2 и присвойте Event «Да». Аналогично для стрелки соединяющей «Позиция имеется?» и «Закрепить позицию за заказом» присвоить Event «Да». Стрелке, соединяющей «Позиция имеется?» и «Поставить заказ в ожидание» - «Нет».

20. Добавим элементарные действия (Actions) к активности “Проверить позицию заказа». Установим курсор на «Проверить позицию заказа» и двукратным щелчком мыши откроем окно спецификации. Откроем закладку Actions. Установим курсор на свободное поле и нажмем правую клавиши мыши. В выпадающем меню нажмем Insert. В появившейся заставке в поле When выберем Entry(на входе в активность), В поле Name введем «Просмотреть спецификацию к заказу». Нажать Ok. Вновь нажмем курсор правой мыши и введем новое действие. Полю When прсвоим Do(промежуток между входом и выходом), а полю Name «Найти новую позицию». При вводе третьей активности полю When присвойте Exit (выход), а полю Name «Передать результаты поиска».

21. Путем перемещения объектов (установить курсор мыши- нажать- тащить- отпустить) привести диаграмму к виду, показанному на рис. 6.

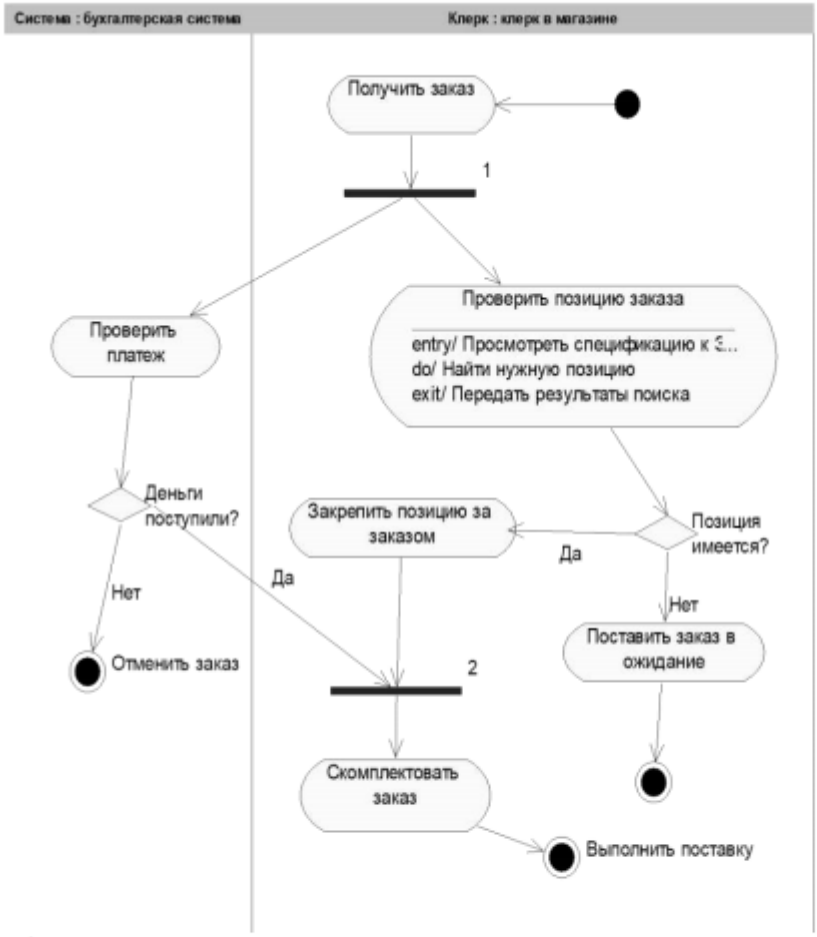


Рис. 6 Диаграмма активности для варианта использования «Выполнить поставку заказа»

Задание 3. Пакеты и классы

В этом упражнении необходимо сгруппировать в пакеты классы, созданные при выполнении предыдущих работ. Затем нужно будет построить несколько диаграмм Классов и показать на них классы и пакеты системы.

Создание диаграммы Классов

Объедините обнаруженные классы в пакеты. Создайте диаграмму Классов для отображения пакетов, диаграммы Классов, для представления классов в каждом пакете и диаграмму Классов для представления всех классов варианта использования "Ввести новый заказ".

Этапы выполнения

Создание пакетов

1. Щелкните правой кнопкой мыши на Логическом представлении браузера.
2. В открывшемся меню выберите пункт New > Package (Создать >Пакет).
3. Назовите новый пакет Сущности.
4. Повторив шаги 1—3, создайте пакеты Границы и Управление.

Создание Главной диаграммы Классов

1. Дважды щелкнув мышью на Главной диаграмме Классов, находящейся под Логическим представлением браузера, откройте ее.
2. Перетащите пакет *Сущности* из браузера на диаграмму.
3. Перетащите пакеты *Границы* и *Управление* из браузера на диаграмму.

Главная диаграмма Классов должна выглядеть, как показано на рис. 7

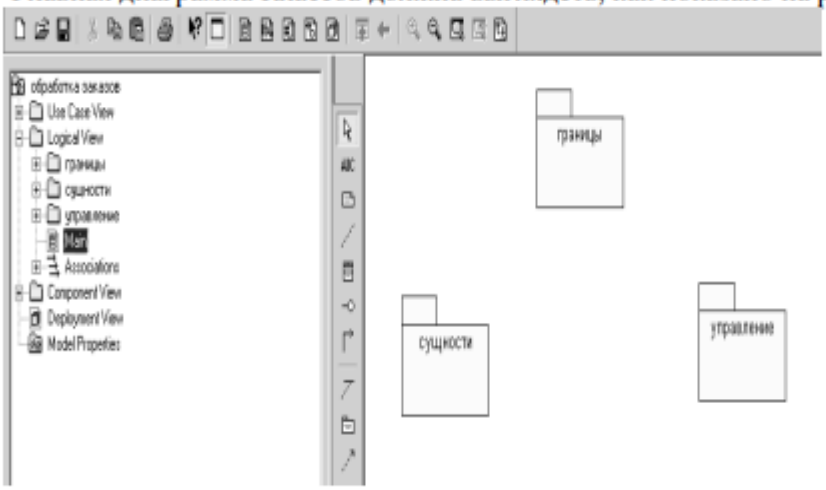


Рис. 7 Главная диаграмма классов в логическом представлении браузера.

Создание диаграммы Классов для сценария "Ввести новый заказ" с отображением всех классов

1. Щелкните правой кнопкой мыши на Логическом представлении браузера.
2. В открывшемся меню выберите пункт New > Class Diagram (Создать > Диаграмма Классов).
3. Назовите новую диаграмму Классов: Ввод нового заказа.
4. Дважды щелкнув мышью на этой диаграмме в браузере, откройте ее.
5. Перетащите из браузера все классы (*Выбор_заказа, Заказ_деталей, упр_заказами, Заказ, Упр_транзакциями.*

Объединение классов в пакеты

1. В браузере перетащите класс *выбор_заказа* на пакет Границы.
2. Перетащите класс *заказ_деталей* на пакет Границы.

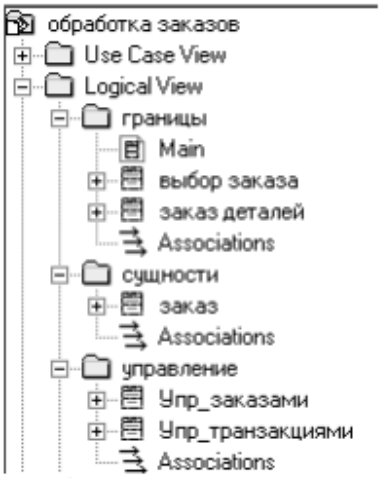


Рис. 8 Представление пакетов и классов

Добавление диаграмм Классов к каждому пакету

1. В браузере щелкните правой кнопкой мыши на пакете *Границы*.

2. В открывшемся меню выберите пункт New > Class Diagram (Создать > Диаграмма Классов).

3. Введите имя новой диаграммы — Main (Главная).

4. Дважды щелкнув мышью на этой диаграмме, откройте ее.

5. Перетащите на нее из браузера классы *выбор_заказа* и *заказ_деталей*.

6. Закройте диаграмму.

В браузере щелкните правой кнопкой мыши на пакете *Сущности.*

8. В открывшемся меню выберите пункт New > Class Diagram (Создать > Диаграмма Классов).

9. Введите имя новой диаграммы — Main (Главная).

10. Дважды щелкнув мышью на этой диаграмме, откройте ее.

11. Перетащите на нее из браузера класс *Заказ*.

12. Закройте диаграмму

13. В браузере щелкните правой кнопкой мыши на пакете *Управление*

14. В открывшемся меню выберите пункт New > Class Diagram (Создать > Диаграмма Классов).

15. Введите имя новой диаграммы — Main (Главная).

16. Дважды щелкнув мышью на этой диаграмме, откройте ее.

17. Перетащите на нее из браузера классы *Упр_заказами* и *Упр_транзакциями*

18. Закройте диаграмму

Задание 4. Уточнение методов и свойств классов.

В этом упражнении к описаниям операций будут добавлены детали, включая параметры и типы возвращаемых значений, и определены атрибуты классов

Постановка проблемы

Для определения атрибутов классов был проанализирован поток событий. В результате к классу *Заказ* диаграммы Классов были добавлены атрибуты *Номер заказа* и *Имя клиента*. Так как в одном заказе можно указать большое количество товаров и у каждого из них имеются свои собственные данные и поведение, было решено моделировать товары как самостоятельные классы, а не как атрибуты класса *Заказ*.

Добавление атрибутов и операций

Добавим атрибуты и операции к классам диаграммы Классов "Ввод нового заказа". При этом используем специфические для языка особенности. Установим параметры так, чтобы показывать все атрибуты, все операции и их сигнатуры. Применим нотацию UML.

Этапы выполнения

Настройка

1. В меню модели выберите пункт Tools > Options (Инструменты >Параметры).

2. Перейдите на вкладку Diagram.

3. Убедитесь, что флажок Show visibility (Показать видимость) установлен.

4. Убедитесь, что флажок Show stereotyps (Показать стереотипы) установлен.

5. Убедитесь, что флажок Show operation signatures (Показать сигнатуры операций) установлен.

6. Убедитесь, что флажки Show all attributes (Показать все атрибуты) и Show all operations (Показать вое операции) установлены.

7. Убедитесь, что флажки Suppress attributes (Подавить атрибуты) и Suppress operations (Подавить операции) сброшены.

8. Перейдите на вкладку Notation (Нотация).

9. Убедитесь, что флажок Visibility as icons (Отображать пиктограммы) сброшен.

Добавление нового класса

1. Найдите в браузере диаграмму Классов варианта использования "Ввести новый заказ".
2. Дважды щелкнув мышью на диаграмме, откройте ее.
3. Нажмите кнопку Class панели инструментов.
4. Щелкните мышью внутри диаграммы, чтобы поместить туда новый класс.
5. Назовите его *Позиц_заказа*.
6. Назначьте этому классу стереотип Entity.
7. В браузере перетащите класс в пакет *Сущности.*

Добавление атрибутов

1. Щелкните правой кнопкой мыши на классе *Заказ*.
2. В открывшемся меню выберите пункт New Attribute (Создать атрибут),
3. Введите новый атрибут:
OrderNumber : Integer
4. Нажмите клавишу Enter
5. Введите следующий атрибут:
CustomerName : String.
6. Повторив шаги 4 и 5, добавьте атрибуты:
OrderDate : Date
OrderFillDate : Date

Если тип атрибута не появляется в выпадающем списке, то введите его от руки и он далее будет появляться.

7. Щелкните правой кнопкой мыши на классе *Позиц_заказа*.
8. В открывшемся меню выберите пункт New Attribute (Создать атрибут).
9. Введите новый атрибут:
ItemID : Integer.
10. Нажмите клавишу Enter.
11. Введите следующий атрибут:
ItemDescription : String.

Добавление операций к классу *Позиц_заказа*

1. Щелкните правой кнопкой мыши на классе *Позиц_заказа*.
2. В открывшемся меню выберите пункт New Opration (Создать операцию).
3. Введите новую операцию:
Создать()
4. Нажмите клавишу Enter.
5. Введите следующую операцию:
Взять_информацию()

Boolean

5. Отредактируйте операцию *Дать_информацию*;
Дать_информацию(): String

Подробное описание операций с помощью браузера

1. Найдите в браузере класс *Позиц_заказа*.

2. Раскройте этот класс, щелкнув на значке "+" рядом с ним. В браузере появятся атрибуты и операции класса.

3. Дважды щелкнув мышью на операции *Дать_информацию()* , откройте окно ее спецификации:

4. В раскрывающемся списке Return class (Возвращаемый класс) укажите String.

5. Щелкнув мышью на кнопке ОК, закройте окно спецификации операции.

6. Дважды щелкните в браузере на операции *Дать_информацию* () класса *Позиц_заказа*, чтобы открыть окно ее спецификации.

7. В раскрывающемся списке Return class укажите Boolean.

8. Перейдите на вкладку Detail(Подробно).

9. Щелкните правой кнопкой мыши в области аргументов, чтобы добавить туда новый параметр:

10. В открывшемся меню выберите пункт Insert (Вставить). Rose добавит аргумент под названием argname.

11. Щелкнув один раз на этом слове, выделите его и измените имя аргумента на ID.

12. Щелкните на колонке Type (Тип). В раскрывающемся списке типов выберите Integer (Если этого либо иного необходимого типа не будет- введите его вручную).

13. Щелкните на колонке Default (По умолчанию), чтобы добавить значение аргумента по умолчанию. Введите число 0.

14. Нажав на кнопку ОК, закройте окно спецификации операции.

15. Дважды щелкните на операции *Создать()* класса *Позиц_заказа*, чтобы открыть окно ее спецификации.

16. В раскрывающемся списке Return class укажите Boolean.

17. Нажав на кнопку ОК, закройте окно спецификации операции.

Подробное описание операций

1. Используя браузер или диаграмму Классов, введите следующие сигнатуры операций класса *Заказ_деталей*:

Открыть() : Boolean

Сохранить заказ() : Boolean

2. Используя браузер или диаграмму Классов, введите сигнатуру операций класса *Выбор_заказа*:

Создать() : Boolean

3. Используя браузер или диаграмму Классов, введите сигнатуру операций класса *Упр_заказами*:

Сохранить заказ(OrderID : Integer) : Boolean

4. Используя браузер или диаграмму Классов, введите сигнатуры операций класса *Упр_транзакциями*:

Сохранить заказ(OrderID : Integer) : Boolean

Сохранить информацию() : Integer

Задание 5. Описание связей между классами

В этом упражнении определяются связи между классами, участвующими в варианте использования "Ввести новый заказ".

Постановка задачи

Чтобы найти связи, были просмотрены диаграммы Последовательности. Все взаимодействующие там классы нуждались в определении соответствующих связей на диаграммах Классов. После обнаружения связи были добавлены на диаграммы классов.

Добавление связей

Добавим связи к классам, принимающим участие в варианте использования "Ввести новый заказ".

Этапы выполнения упражнения

Настройка

1. Найдите в браузере диаграмму Классов "Ввод нового заказа",

2. Дважды щелкнув на диаграмме, откройте ее.

3. Проверьте, имеется ли в панели инструментов диаграммы кнопка Unidirectional Association (Однонаправленная ассоциация). Если ее нет, продолжите настройку, выполнив шаги 4 и 5. Если есть, приступайте к выполнению самого упражнения.

4. Щелкните правой кнопкой мыши на панели инструментов диаграммы и в открывшемся меню выберите пункт Customize(Настроить),

5. Добавьте на панель кнопку Creates A Unidirectional Association (Создать однонаправленную ассоциацию).

Добавление ассоциаций

1. Нажмите кнопку Unidirectional Association панели инструментов.

2. Проведите ассоциацию от класса *выбор_заказа* к классу *заказ_деталей*.

3. Повторите шаги 1 и 2, создав ассоциации:

• От класса *заказ_деталей* к классу *упр_заказами*

• От класса *упр_заказами* к классу *Заказ*

• От класса *упр_заказами* к классу *упр_транзакциями*

• От класса *упр_транзакциями* к классу Заказ

• От класса *упр_транзакциями* к классу *Позиц_заказа*

• От класса *Заказ* к классу *Позиц_заказа*

4. Щелкните правой кнопкой мыши на однонаправленной ассоциации между классами *выбор_заказа* и *заказ_деталей* класса *выбор_заказа*.

5. В открывшемся меню выберите пункт Multiplicity > Zero or One (Множественность >- Нуль или один),

6. Щелкните правой кнопкой мыши на другом конце однонаправленной ассоциации.

7. В открывшемся меню выберите пункт Multiplicity > Zero or One (Множественность > Нуль или один),

8. Повторите шаги 4—7, добавив на диаграмму значения множественности для остальных ассоциаций, как показано на рис. 10

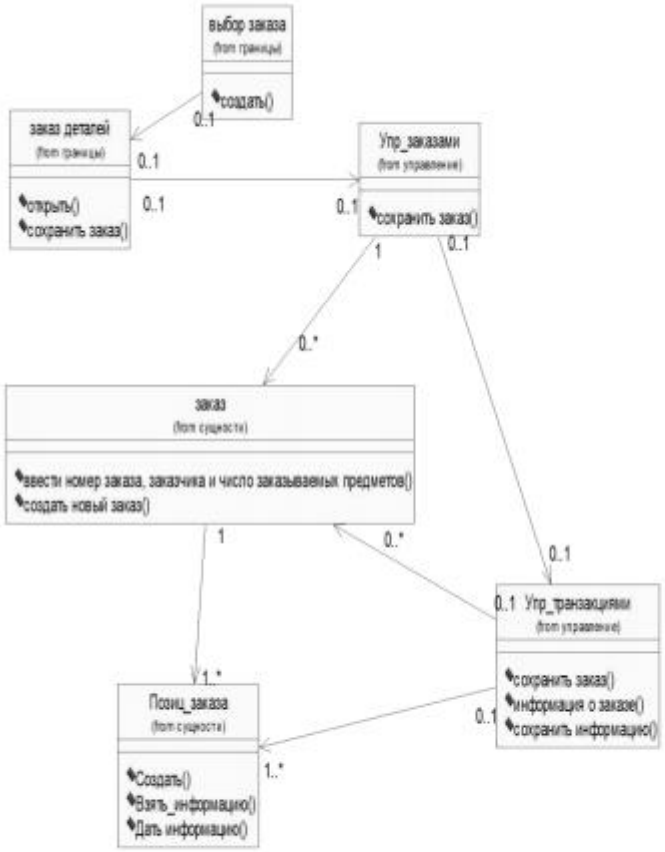


Рис. 10 Ассоциации сценария "Ввести новый заказ"

Задание 6. Исключение кириллизованного текста в информации классов.

Разработанные ранее модели, предназначенные для описания предметной области используют кириллизованную символику, недопустимую для большинства языков программирования. Выполните замену русского текста на латинский шрифт. Для этой цели сохраните предыдущую модель под другим именем и далее работайте с новым файлом (что бы при необходимости можно было бы вернуться к бизнес- процессам, описанным русским шрифтом).

Этапы выполнения

Этап 1. Используя меню (Файл-> Сохранить как) сохраните данную модель под другим именем (например Заказ1) в той же папке, что и исходная модель.

Работайте далее с копией модели (то есть Заказ1).

Этап 2. Переименуйте классы и их спецификации таким образом, чтобы использовался только латинский шрифт. Замените имя класса

Заказ_деталей на *OrderDetail*
Выбор_заказа на *OrderOptions*
Заказ на *Order*
Упр_заказами на *OrderMgr*
Позиц_заказа на *OrderItem*
Упр_транзакциями на *TransactionMgr*

Измените имена операций таким образом, чтобы рис.10 преобразовался в рис. 11. Для этого, измените операцию класса *OrderOptions*

Открыть() на *Open()*
Класса *OrderDetail*
Открыть() на *Open()*
Сохранить заказ() на *Save()*
Класса *Order*
Ввести номер заказа, заказчика и число заказываемых предметов() на *SetInfo()*
Сохранить_заказ() на *Save()*
Класса *OrderMgr*
Сохранить заказ() на *SaveOrder()*
Класса *TransactionMgr*
Сохранить заказ() на *SaveOrder()*
Сохранить информацию о заказе() на *Commit()*
Создать_заказ() на *SubmitInfo()*
Класса *OrderItem*
Создать() на *Create()*
Взять_информацию() на *GetInfo()*
Дать_информацию на *SetInfo()*
Переименуйте имена пакетов
Границы на *Boundaries*
Сущности на *Entity*
Контроль на *Control*
Добавление стереотипов к классам

1. Щелкните правой кнопкой мыши на классе OrderOptions диаграммы.
2. В открывшемся меню выберите пункт Open Specification (Открыть спецификацию).
3. В поле стереотипа выберите из выпадающего списка слово Boundary (если его нет, то введите).
4. Нажмите на кнопку ОК.
5. Повторив шаги 1—4, свяжите классы OrderDetail со стереотипом Boundary, OrderMgr и TransactionMgr со стереотипом Control, а класс Order и OderItem— со стереотипом Entity.

Теперь диаграмма Классов должна иметь вид, показанный на рис. 11.

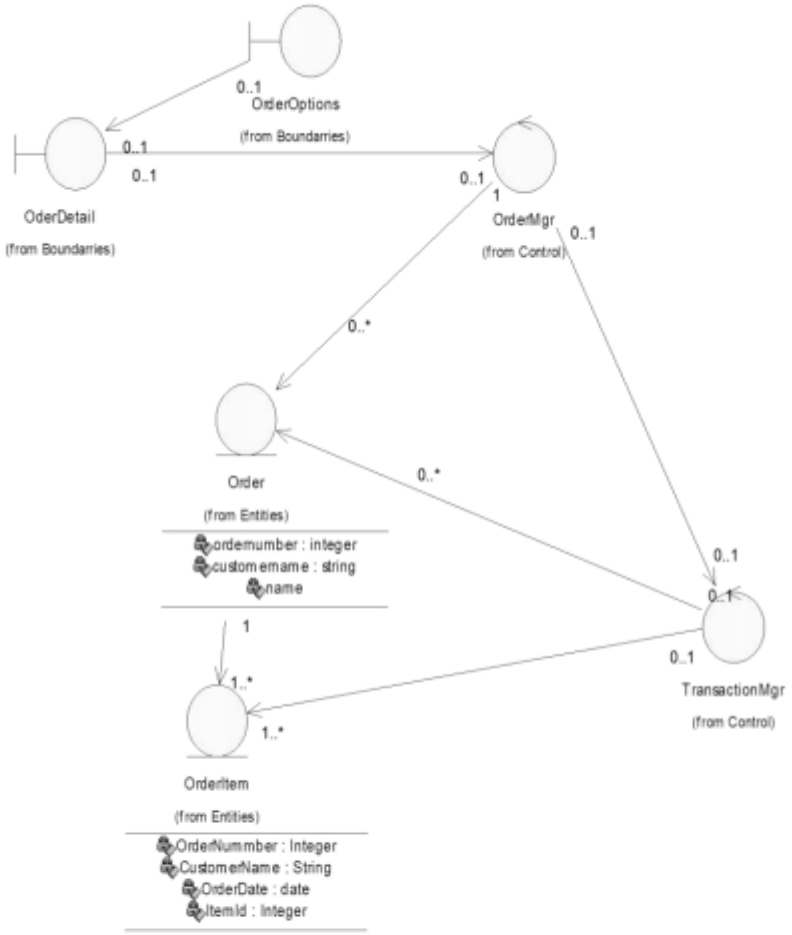


Рис. 11 Основная диаграмма классов

Замечание. На диаграмме рис. 11 возможно визуальное представление классов не в виде иконок, а в виде дополнительной строки текста с именем стереотипа. За этот вид отвечает метка установленная либо на icon либо на label (Class> Open Specefication> Options> Label)

Итог работы: тетрадь, защита работы.

Практическая работа № 10

Цель: исследование процесса построение диаграммы компонентов в заданной предметной области

Задание 1.

Задание 1. Построение диаграммы Компонентов

Этапы выполнения

Так как эта модель связана с конкретным языком программирования, то в настройках это необходимо отметить. Выполнить Tools>Options>Notations>Default Language и из выпадающего списка языков программирования выбрать Delphi.

Создание пакетов компонентов

1. Щелкните правой кнопкой мыши на представлении компонентов в браузере.
2. В открывшемся меню выберите пункт New > Package (Создать > Пакет}.
3. Назовите пакет Entities (Сущности).
4. Повторив шаги с первого по третий, создайте пакеты Boundaries (Границы) и Control (Управление).

Добавление пакетов на Главную диаграмму Компонентов

1. Откройте Главную диаграмму Компонентов, дважды щелкнув на ней мышью,
2. Перетащите пакеты Entities, Boundary и Control из браузера на Главную диаграмму.

Отображение зависимостей между пакетами

1. Нажмите кнопку Dependency (Зависимость) панели инструментов.
2. Щелкните мышью на пакете Boundary Главной диаграммы Компонентов.
3. Проведите линию зависимости к пакету Control.
4. Повторив шаги 1 — 3, проведите зависимость от пакета Control к пакету Entities.

В результате диаграмма примет вид рис. 12

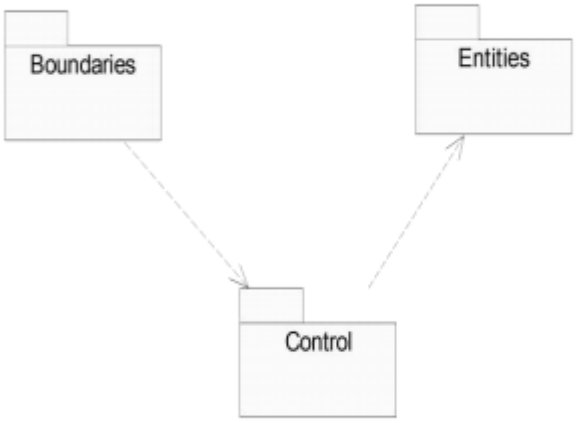


Рис. 12 Зависимости между пакетами

Добавление компонентов к пакетам и отображение зависимостей

1. Дважды щелкнув мышью на пакете Entities Главной диаграммы Компонентов, откройте Главную диаграмму Компонентов этого пакета.
2. Нажмите кнопку Package Specification (Спецификация пакета) панели инструментов.
3. Поместите спецификацию пакета на диаграмму.
4. Введите имя спецификации пакета — *OrderItem_*.
5. Повторив шаги 2—4, добавьте спецификацию пакета *Order_*.
6. Нажмите кнопку Dependency (Зависимость) панели инструментов.
7. Щелкните мышью на спецификации пакета *OrderItem_*.
8. Проведите линию зависимости к спецификации пакета *OrderItem_*.
9. С помощью описанного метода создайте следующие компоненты и зависимости:

Для пакета Boundaries:

• Спецификацию пакета *Orderoptions_*
• Спецификацию пакета *OrderDetail_*

Зависимости в пакете Boundaries:

• От спецификации пакета *Orderoptions_*
к спецификации пакета *OrderDetai_l*

Для пакета Control:

• Спецификацию пакета *OrderMgr_*

• Спецификацию пакета *TransactionMgr_*

Зависимости в пакете Control:

• От спецификации пакета *OrderMgr_*

к спецификации пакета *TransactionMg_*

Создание диаграммы Компонентов системы

1. Щелкните правой кнопкой мыши на представлении Компонентов в браузере.

2. В открывшемся меню выберите пункт New > Component Diagram (Создать > Диаграмма Компонентов).

3. Назовите новую диаграмму System.

4. Дважды щелкните на этой диаграмме мышью.

Размещение компонентов на диаграмме Компонентов системы

1. Разверните в браузере пакет компонентов *Entities*, чтобы открыть его.

2. Щелкните мышью на спецификации пакета *Order_* в пакете компонентов *Entities*.

3. Перетащите эту спецификацию на диаграмму.

4. Повторив шаги 2 и 3, поместите на диаграмму спецификацию пакета *OrderItem_*.

5. С помощью этого метода поместите на диаграмму следующие компоненты:

Из пакета компонентов *Boundaries*:

• Спецификацию пакета *Orderoptions_*

• Спецификацию пакета *OrderDetail_*

Из пакета компонентов Control:

• Спецификацию пакета *OrderMgr_*

• Спецификацию пакета *TransactionMgr_*

6. Нажмите кнопку Task Specification (Спецификация задачи) панели инструментов.

7. Поместите на диаграмму спецификацию задачи и назовите ее *OrderClientExe*.

8. Повторите шаги 6 и 7 для спецификации задачи *OrderServerExe*.

Добавление оставшихся зависимостей на диаграмму Компонентов системы

Уже существующие зависимости будут автоматически показаны на диаграмме Компонентов системы после добавления туда соответствующих компонентов. Теперь нужно добавить остальные зависимости.

1. Нажмите кнопку Dependency (Зависимость) панели инструментов.

2. Щелкните мышью на спецификации пакета *OrderDetail_*

3. Проведите линию зависимости к спецификации пакета *OrderDetail_*

4. Повторив шаги 1 — 3, создайте следующие зависимости:

• От спецификации пакета *OrderMgr_*

к спецификации пакета *Order_*

• От спецификации пакета *TransactionMgr_*

к спецификации пакета *OrderItem_*

• От спецификации пакета *TransactionMgr_*

к спецификации пакета *Order_*

• От спецификации задачи *OrderClientExe* к спецификации пакета *Orderoptions_*

От спецификации задачи *OrderServerExe* к спецификации пакета *OrderMg_r*

Соотнесение классов с компонентами

1. В Логическом представлении браузера найдите класс *Order* пакета *Entities*.

2. Перетащите этот класс на спецификацию пакета компонента *Order_* в представлении Компонентов браузера, В результате класс *Order* будет соотнесен со спецификацией пакета компонента *Order_*.

3. Повторив шаги 1 — 2, соотнесите классы со следующими компонентами:

• Класс *OrderItem* со спецификацией пакета *OrderItem_*

• Класс *Orderoptions* со спецификацией пакета *Orderoptions_*

• Класс *OrderDetail* со спецификацией пакета *OrderDetai_l*

• Класс *OrderMgr* со спецификацией пакета *OrderMgr_*

• Класс *TransactionMgr* со спецификацией пакета *TransactionMgr_*

В результате в браузере после имени класса, в скобках появятся имена компонентов, с которыми этот класс связан (рис. 13)

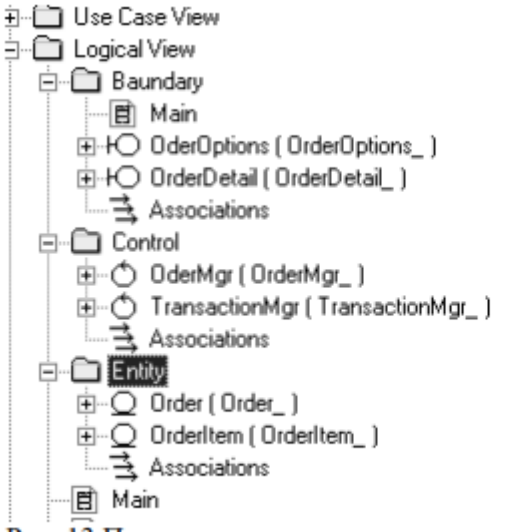


Рис. 13 Представление компонентов и классов в браузере

Итог работы: тетрадь, защита работы.

Практическая работа № 11

Цель: получить навыки создания и редактирования функциональных моделей в Microsoft Visio 2010

Задание 1: ознакомьтесь с материалом

Основные сведения по методологии IDEF0

Модель в нотации IDEF0 представляет собой совокупность иерархически упорядоченных и взаимосвязанных диаграмм. Каждая диаграмма является единицей описания системы и располагается на отдельном листе.

Цель моделирования Модель не может быть построена без четко сформулированной цели. Пример цели: «Описать функциональность предприятия с целью написания спецификаций ИС».

Точка зрения Точку зрения можно представить как взгляд человека, который видит систему в нужном для моделирования аспекте. Как правило, выбирается точка зрения человека, ответственного за моделируемую работу в целом. Цель и точка зрения документируются.

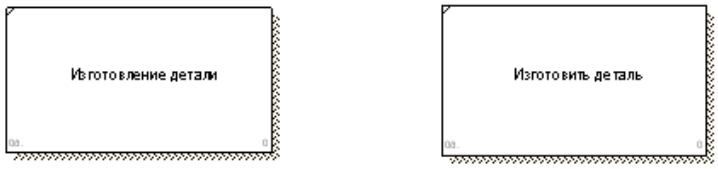
Основные элементы IDEF0-модели

В основе методологии IDEF0 лежат 4 основных понятия:

- функциональный блок;
- интерфейсная дуга (стрелка);
- декомпозиция;
- глоссарий.

Функциональный блок

Функциональные блоки обозначают *поименованные процессы, функции или задачи, которые происходят в течение определенного времени и имеют распознаваемые результаты*. Графически функциональные блоки *изображаются в виде прямоугольников*. Все блоки должны быть названы и определены. *Имя функционального блока* должно быть выражено сочетанием *отглагольного существительного, обозначающего процесс, или глаголом* (Рис. 1):



а) б)

Рисунок 1 - Примеры работ

Определение функционального блока заносится в глоссарий или словарь работ (Activity Dictionary).

Все функциональные блоки модели нумеруются. Номер состоит из префикса и числа. Может использоваться префикс любой длины, но обычно используется префикс А. Контекстная (корневая) работа (функциональный блок) имеет номер А0.

2. Интерфейсная дуга (стрелка - Arrow)

Взаимодействие функциональных блоков с внешним миром и между собой описывается в виде интерфейсных дуг (стрелок). Стрелки представляют собой *некую информацию и обозначаются существительными* (например, «Заготовка», «Изделие») или *именуемыми сочетаниями* (например, «Готовое изделие»). Все стрелки должны быть определены. Определения заносятся в словарь стрелок – глоссарий (Arrow Dictionary).

В IDEF0 различают 4 типа стрелок (рис.2).

Каждая стрелка имеет свое расположение относительно функционального блока.

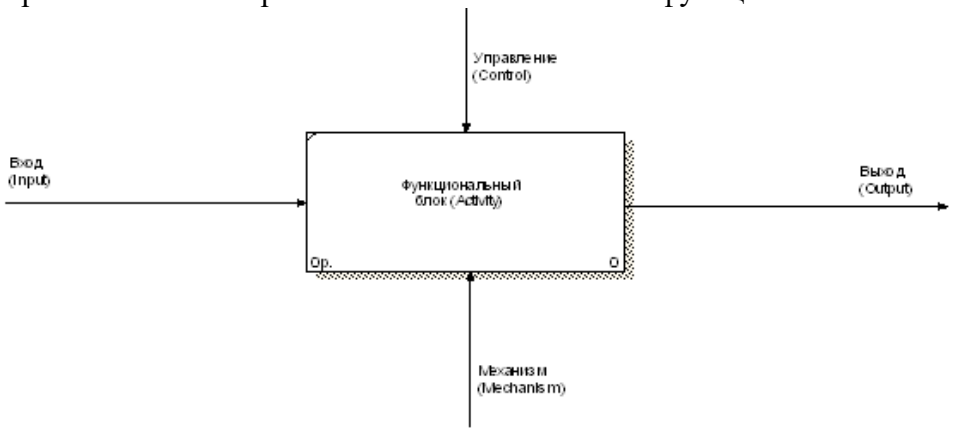


Рисунок 2 - Типы стрелок

Вход (Input) – материал или информация, которые используются или преобразуются работой для получения результата (выхода). Стрелка *Input* рисуется входящей в левую грань работы.

Управление (Control) – правила, стратегии, процедуры или стандарты, которыми руководствуется работа. Каждая работа должна иметь хотя бы одну стрелку управления. Рисуется как входящая в верхнюю грань работы.

Выход (Output) – материал или информация, которые производятся работой. Каждая работа должна иметь хотя бы одну стрелку выхода. Работа без результата не имеет смысла и не должна моделироваться. Изображается исходящей из правой грани работы.

Механизм (Mechanism) – ресурсы, которые выполняют работу, например, персонал предприятия, станки, устройства и т.д. Рисуется как входящая в нижнюю грань работы.

3. Глоссарий *– набор определений, ключевых слов* и т.д., *которые характеризуют каждый объект модели*.

4. Декомпозиция – это *разбиение системы на крупные фрагменты – функции, функции – на подфункции и т.д. до конкретных процедур.*

Модель может содержать 4 типа диаграмм:

- контекстную (в каждой модели может быть только 1 контекстная диаграмма);
- декомпозиции;
- дерева узлов;
- только для экспозиции (FEO).

Контекстная диаграмма является вершиной древовидной структуры диаграмм и *представляет собой общее описание системы и ее взаимодействия с внешней средой*.

После описания системы в целом проводится разбиение ее на крупные фрагменты. Этот процесс называется функциональной декомпозицией, а диаграммы, которые описывают каждый фрагмент и взаимодействие фрагментов – *диаграммами декомпозиции*. После декомпозиции контекстной диаграммы проводится декомпозиция каждого большого фрагмента системы на более мелкие и т.д., до достижения нужного уровня подробности описания.

Диаграмма дерева узлов показывает *иерархическую зависимость работ*, но не взаимосвязи между работами.

Диаграммы для экспозиции (FEO) строятся для *иллюстрации отдельных фрагментов модели,* для иллюстрации альтернативной точки зрения либо для специальных целей.

Все диаграммы имеют нумерацию. Контекстная диаграмма имеет номер А-0, декомпозиция контекстной диаграммы – номер А), остальные диаграммы-декомпозиции – номера по соответствующему узлу (например, А1, А2, А21 и т.д.).

Особенности Microsoft Visio 2010

Для построения функциональной модели бизнес-процесса, используя MS Office Visio 2010, необходимо в меню Пуск выбрать: Microsoft Office - Microsoft Office Visio 2010.

В открывшейся программе выбрать: Файл – Фигуры – Блок-схема – Фигуры схемы IDEF 0.

Используемые блоки для построения функциональной модели:

Блок заголовка – рамка, которую необходимо установить на весь лист и оформить в соответствии с правилами оформления диаграмм в нотации IDEF0 Блок текста необходим для описания точки зрения и цели на контекстной диаграмме.

Блок действия – для описания работ, рассматриваемых в процессе.

Одностороннее соединение – элемент изображения интерфейсных дуг, таких как вход/выход, механизм/управление.

Соединительная линия IDEF 0 – объект для изображения интерфейсных дуг между работами в модели.

Задание 2:

1. Создание контекстной диаграммы.

1. Запустите Microsoft Office Visio 2007.

2. В меню выбрать:

а) Файл – Создать – создать документ

б) Файл – Фигуры – Блок-схема – Фигуры схемы IDEF 0

Окно программы примет вид, подобный (Рис. 3)

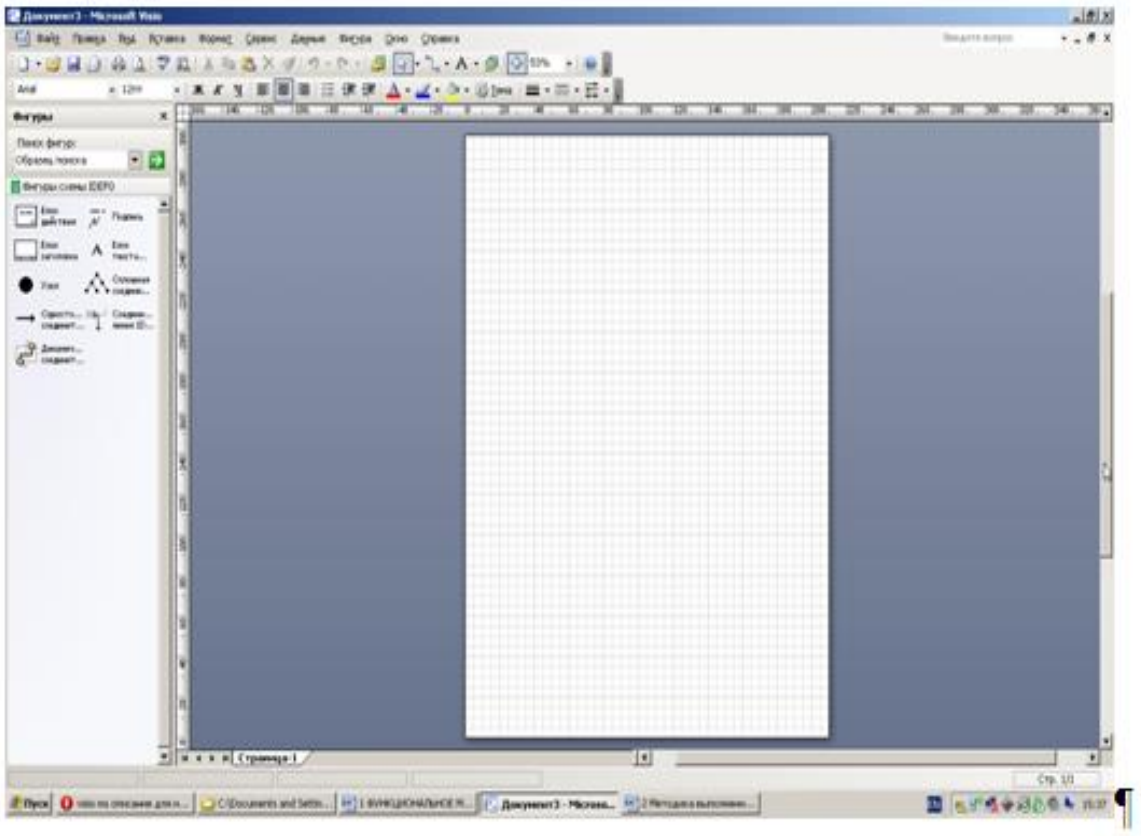


Рисунок 3 - Окно программы

3. *Создание мастерской страницы*

1) Для удобства переведите страницу в альбомный вид: Файл – Параметры страницы – Альбомная;

2) Перетащите Блок заголовка на пустую страницу, удерживая нажатой правую кнопку мыши;

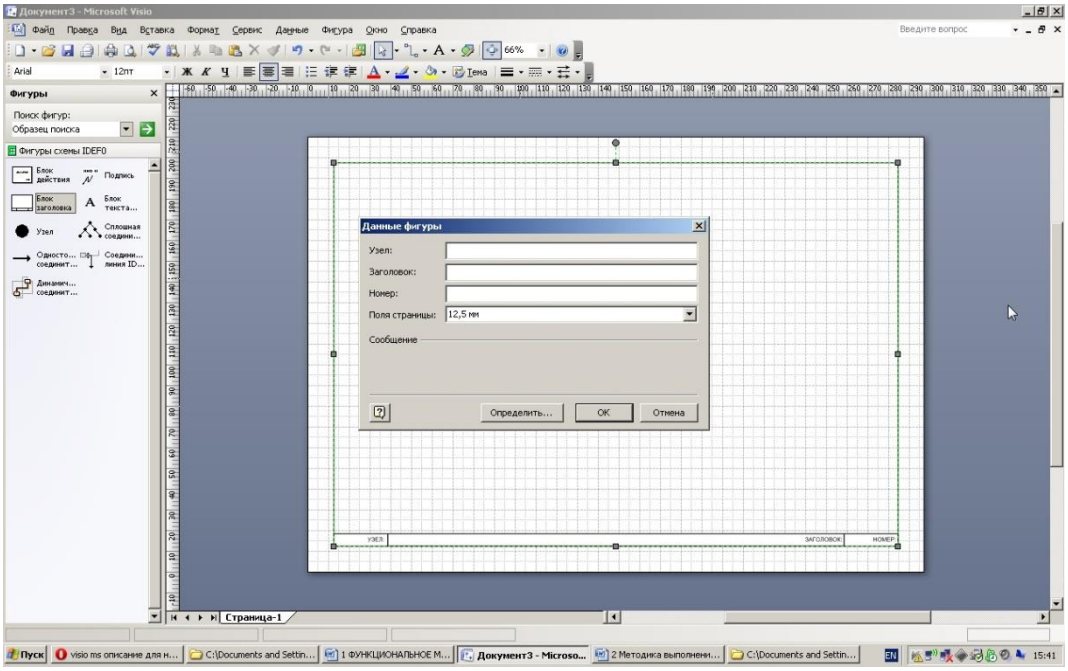


Рисунок 4 - Мастерская страница

3) Заполнить поле «Заголовок», предложенное в открывшемся окне: внести номер контекстной диаграммы и имя рассматриваемого процесса, в данном случае: *А-0 Выполнить курсовую работу*;

Далее, имя заголовка фигуры «Блок заголовка» должно соответствовать номеру и названию задачи, декомпозиция которой будет изображена в данной области. Например: *А1 Получить задание*.

4. *Определение цели и точки зрения*

С помощью кнопки *Блока текста* внесите текст в поле диаграммы – точку зрения и цель (Рис. 5).

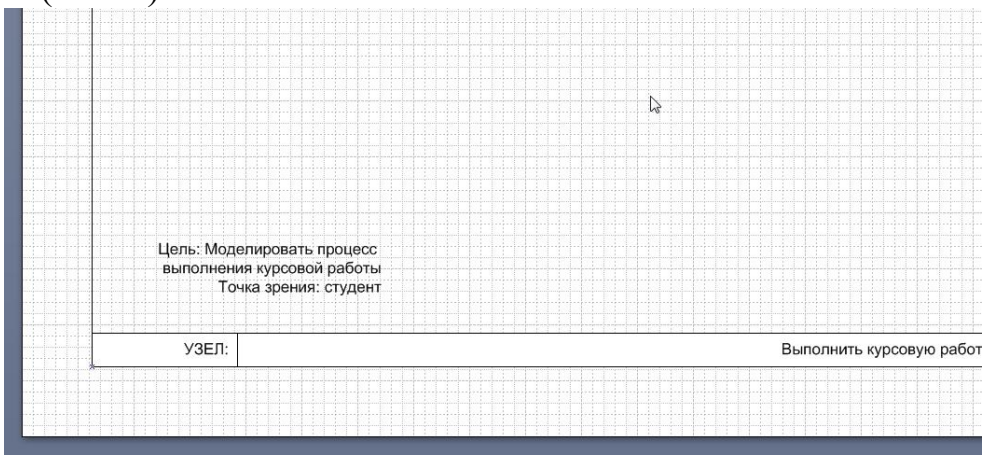


Рисунок 5 - Цель и точка зрения

5. В поле диаграммы (поле Блока заголовка) внесите *Блок действия.* В открывшемся окне «Данные фигуры» внесите *имя процесса* и *идентификатор процесса*.

6. С использованием блока *Одностороннее соединение* создайте стрелки на контекстной диаграмме (Табл. 1).

Таблица 1 – Стрелки контекстной диаграммы

Имя стрелки (Arrow Name)	***Определение стрелки (Arrow Definition)***	***Тип стрелки (Arrow Type)***
График	График консультаций и сроки сдачи	Input
Список литературы	Источники информации для выполнения курсовой работы	Input

Варианты заданий	Список заданий на курсовую работу, подлежащий распределению между студентами	Input
Методические указания	Документ, содержащий указания по выполнению курсовой работы, описывающий содержание ее частей и основные требования	Control
Положение о курсовом проектировании	Документ, отражающий организационные требования по выполнению и сдаче курсовой работы	Control
Курсовая работа	Документ, являющийся основанием для получения оценки	Output
Оценка за курсовую работу	Результат выполнения курсовой работы	Output
Студент	Тот, кто выполняет курсовую работу	Mechanism

7. Результат выполнения предыдущих пунктов представлен на Рис. 6

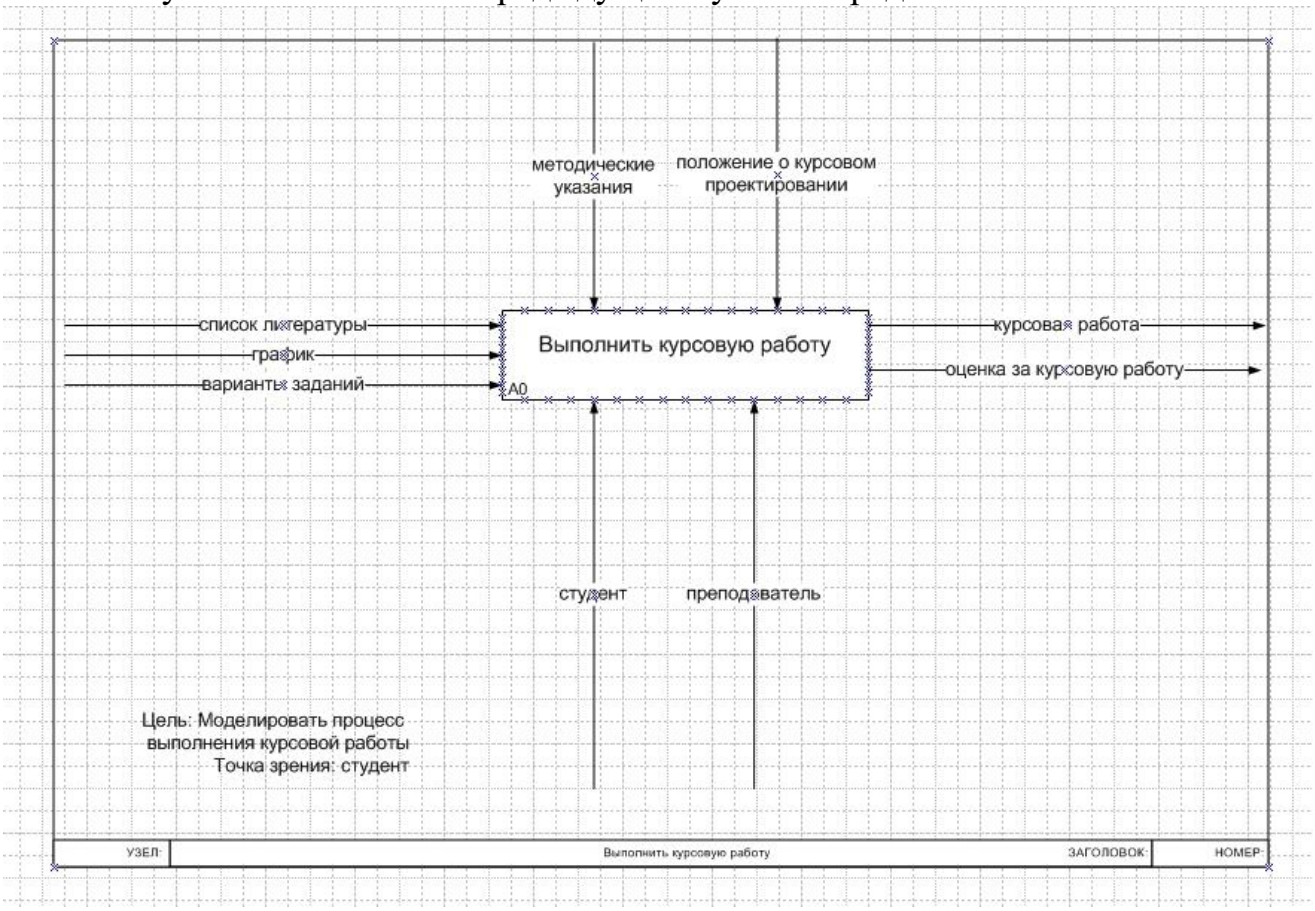


Рисунок 6 - Контекстная диаграмма

2. Создание диаграммы декомпозиции

1. Для построения декомпозиции диаграммы создайте новую страницу путем нажатия правой кнопкой мыши в нижнем левом углу окна на ярлык *Страница 1*. Выбрать пункт *Добавить страницу* (Рис. 7)

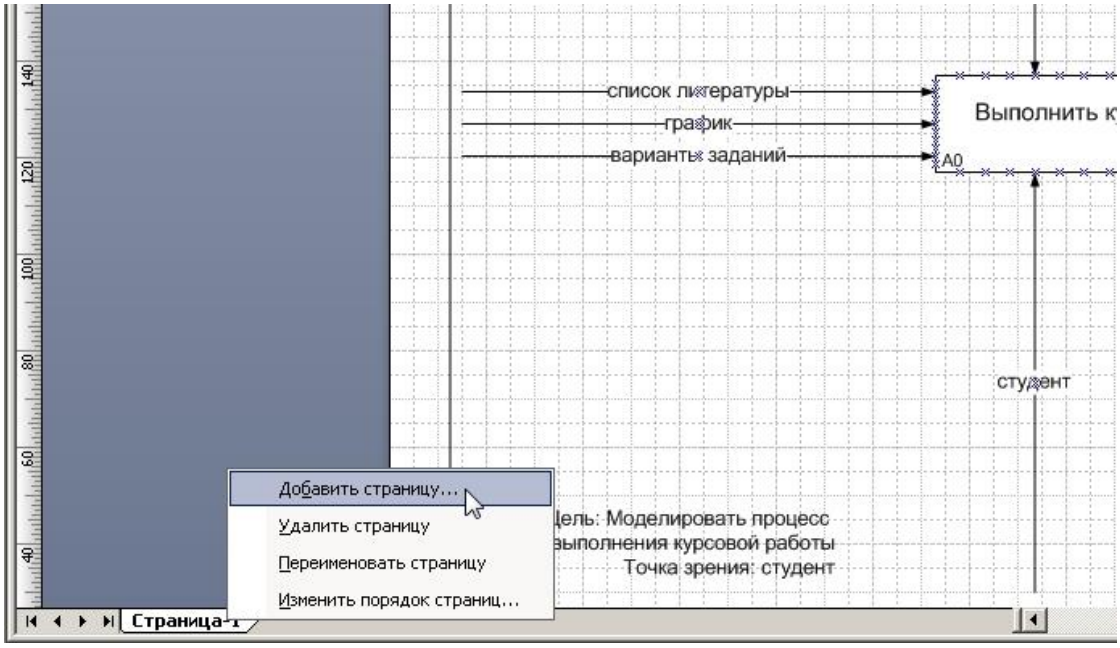


Рисунок 7 - Добавление страницы

2. Переименуйте страницы в соответствии с уровнем декомпозиции, например: A-0, A1 и т.д.

3. Распределите работы диаграммы декомпозиции в области *Блока заголовка* в соответствии с Табл. 2

Таблица 2 – Работы диаграммы декомпозиции A0

Имя работы (Activity Name)	**Определение (Definition)**
Получить задание	Выбрать задание из списка, согласовать его с преподавателем
Подобрать литературу	Выбрать из списка литературы подходящие источники
Сделать расчеты	Выполнить (если необходимо) расчетную часть курсовой работы согласно заданию
Сделать графическую часть	При необходимости сделать графики и чертежи
Оформить пояснительную записку	Оформить текстовую часть и объединить все сделанные части в единое целое
Получить консультацию	Получить консультацию у преподавателя перед защитой, выявить неточности и недостатки
Защитить курсовую работу	Сдать готовую курсовую работу и ответить на вопросы преподавателя

4. Распределите стрелки для диаграммы декомпозиции в соответствии с контекстной диаграммой. Для этого «перенесите» входные и выходные стрелки, связанные с декомпозируемой работой, в поле декомпозиции.

Итог выполнения вышеописанных шагов представлен на Рис. 8

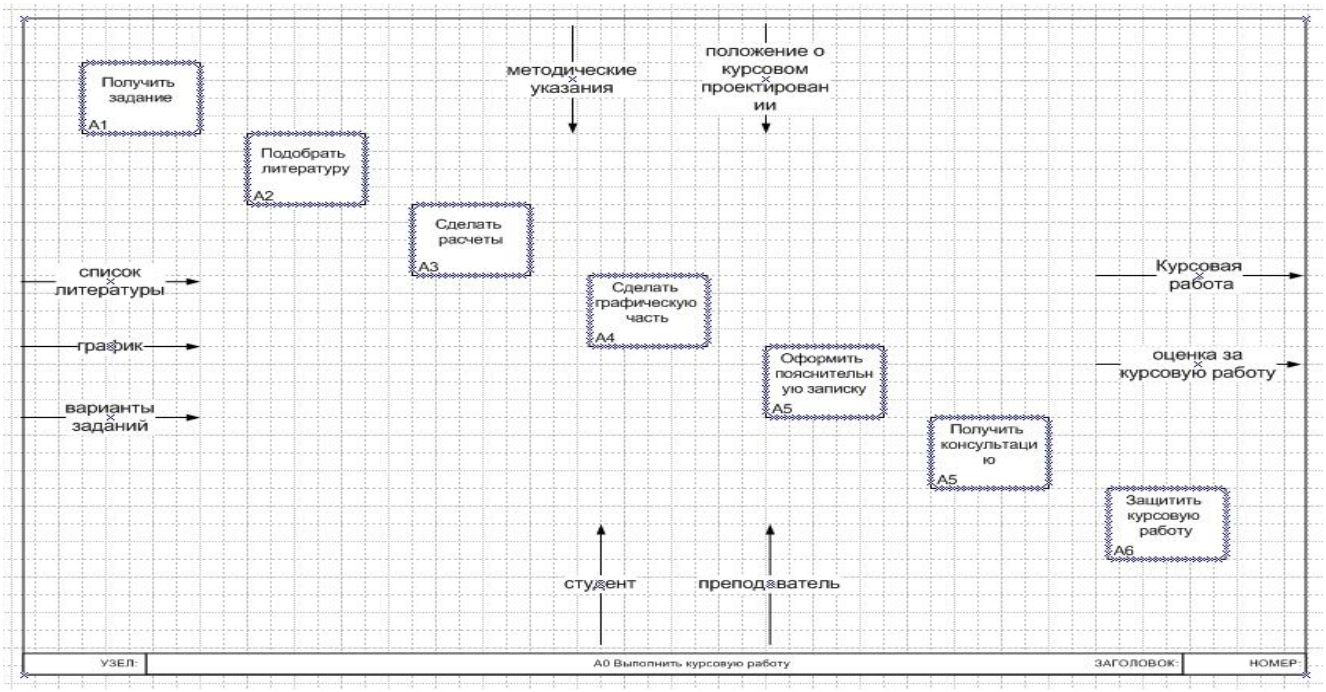


Рисунок 8 - Диаграмма декомпозиции

Разветвление стрелок. График (расписание) необходимо для того, чтобы прийти на консультацию и на защиту, т.е. необходимо подвести одноименную стрелку к 2 работам. Для разветвления стрелки необходимо от фрагмента стрелки до сегмента работы провести стрелку, состоящую из нескольких блоков *Однонаправленное соединение*.

Слияние стрелок. Для слияния двух стрелок выхода необходимо провести работы аналогичные разветвлению.

ICOM-метки. Используя *блок текста,* расставьте ICOM метки.

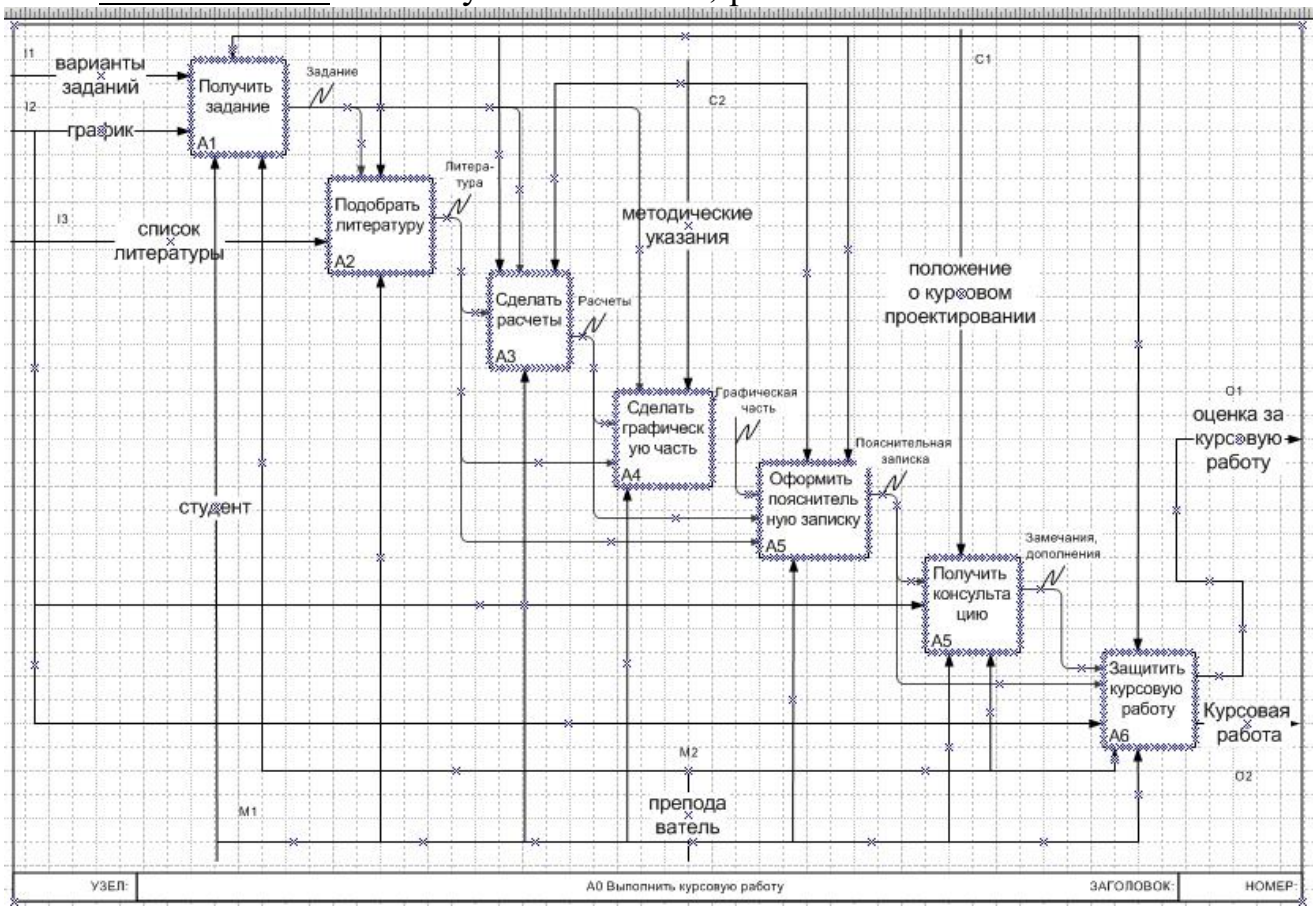


Рисунок 9 - Диаграмма декомпозиция блока A0

Результат выполнения предыдущих пунктов представлен на рисунке (рис. 9).

3. Создание дерева узлов

4. Дерево узлов – это диаграмма, отображающая иерархию работ процесса (Рис. 10)

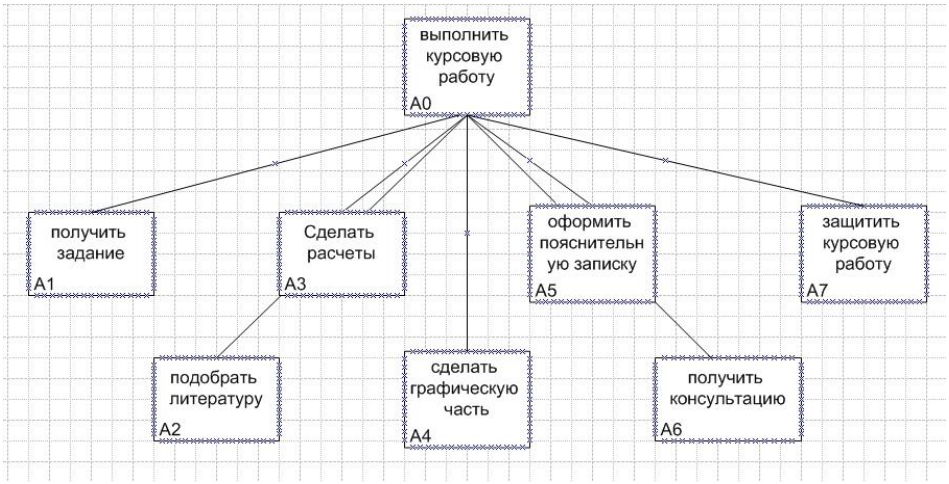


Рисунок 10 - Диаграмма узлов

Для построения диаграммы:

- создайте новую страницу;
- присвойте имя странице: дерево узлов;
- постройте дерево узлов, используя фигуры схемы IDEF0.

4. Создание глоссария

Глоссарий – это словарь ключевых слов, повествований, изложений, используемых при описании процесса (Рис. 11, 12).

Для построения глоссария:

- создайте документ Microsoft Office Word;
- создайте 2 таблицы: описание работ процесса, описание интерфейсных дуг процесса;
- наименование столбцов таблиц: имя (работы/дуги, описание);
- заполните таблицы в соответствии с ранее разработанной моделью процесса.
-

Name	Definition
Выполнить курсовую работу	Текущие процессы выполнения курсовой работы
Защитить курсовую работу	Сдать готовую курсовую работу и ответить на вопросы преподавателя
Оформить пояснительную записку	Оформить текстовую часть и объединить все сделанные части в единое целое
Подобрать литературу	Выбрать из списка литературы подходящие
Получить задание	Выбрать задание из списка, согласовать его с
Получить консультацию	Получить консультацию у преподавателя перед
Сделать графическую часть	При необходимости сделать графики и чертежи
Сделать расчеты	Выполнить (если необходимо) расчетную часть курсовой

Рисунок 11 - Словарь работ

Name	Definition
Варианты заданий	Список заданий на курсовую работу, подлежа
График	График консультаций и сроки сдачи
Графическая часть	Выполненная графическая часть курсовой раб
Задание	Выдается на консультации преподавателем, чт
Замечания, дополнения	Замечания преподавателя, полученные на кон
Курсовая работа	Документ, являющийся основанием для получ
Литература	Выбранные источники, необходимые для выпо
Методические указания	Документ, содержащий указания по выполнен
Оценка за курсовую раб	Результат выполнения курсовой работы
Положение о курсовом п	Документ, отражающий организационные треб
Пояснительная записка	Теоретическая часть + расчеты + графическая
Преподаватель	Тот, кто оценивает курсовую работу
Расчеты	Выполненная расчетная часть курсовой работ
Список литературы	Источники информации для выполнения курсо
Студент	Тот, кто выполняет курсовую работу

Рисунок 12 - Словарь стрелок

Задание

На основе мнемосхемы процесса, составить функциональную модель в нотации IDEF 0.

Задача 1. Необходимо рассмотреть процесс обработки персональных данных о школьниках. В контекстной диаграмме входной информацией являются данные: принятое заявление, личные дела, успеваемость, учебные планы. Выходная информация – сформированные журналы, различные отчеты. Механизмами являются секретарь, администрация. Управляющие стрелки – нормативные документы.

Процесс обработки персональных данных о школьниках состоит из четырех работ: обработка заявления, регистрация личного дела и формирование класса, контроль успеваемости, обработка журналов.

Блок «Обработать заявление». Входными блока являются принятое заявление учащегося, секретарь учебной части производит его обработку, т.е. занесение данных в систему. На выходе функции будет обработанное заявление и данные для регистрации школьника.

Блок «Зарегистрировать личное дело и сформировать класс». Входными данными блока являются обработанное заявление, секретарь учебной части регистрирует личное дело в бумажной форме, одновременно формируется класс. На выходе функции будут списки классов с учащимися и сформированные по классам журналы.

Блок «Контролировать успеваемость». Входными данными блока являются данные о классе с учащимися, которые подвергаются контролю успеваемости. На выходе будут заполненные журналы.

Блок «Обработать журналы». Входными данными блока являются журналы, которые обрабатываются секретарем учебной части для составления отчетов. На выходе функции формирование и печать документов, отчетов.

Задача 2. Необходимо рассмотреть процесс приема на работу нового сотрудника. В контекстной диаграмме входной информацией являются данные: заявление о приеме на работу, резюме. Выходная информация – приказ о зачислении. Механизмами являются сотрудники отдела кадров. Управляющие стрелки – устав предприятия, трудовое законодательство РФ.

Процесс приема сотрудника состоит из четырех работ: рассмотрение резюме, проведение собеседования, рассмотрение заявления о приеме на работу, подписание приказа о зачислении.

Процесс рассмотрения резюме состоит из четырех работ: анализ резюме, анализ вакансий, сопоставление резюме с существующими вакансиями, принятие решения о проведении собеседования. В диаграмме процесса «Рассмотрение резюме» входной информацией является резюме. Выходная информация – решение о назначении собеседования.

Процесс подписания приказа о зачислении состоит из трех работ: формирование приказа о зачислении, рассмотрение приказа, утверждение приказа. В диаграмме процесса «Подписание приказа о зачислении» входной информацией является подписанное заявление.

Задача 3. Рассмотреть функционирование системы, которая описывает порядок получения водительских прав.

Экзамен в ГИБДД состоит из трех частей: теория, площадка, город. Теорию сдают на компьютере в виде теста из 20 вопросов. Необходимо не допустить более 2-х ошибок. Если теория сдана, то курсанта допускают до сдачи площадки. Там надо будет выполнить трогание в подъем, параллельную парковку и разворот в три приема. Пройдя и площадку, курсант выходит на последний этап - город. Там необходимо проехать определенный маршрут, соблюдая правила дорожного движения. Если все пройдет без ошибок, то экзамен будет сдан, и курсант получит водительское удостоверение.

Для сдачи экзаменов необходимо предоставить:

- паспорт;
- мед. справку;
- документ о прохождении обучения;
- квитанции об уплате сборов.

Итог работы: тетрадь, защита работы.

Практическая работа № 12

Цель: Построить логическую модель данных выбранной предметной области в нотации IDEF1X

Задание 1: ознакомьтесь с материалом

IDEF1X основан на подходе Чена и позволяет построить модель данных, эквивалентную реляционной модели в третьей нормальной форме. Нотация Чена и сам процесс построения диаграмм сущность-связь изучалась в курсе "Организация баз данных и знаний", поэтому здесь мы рассмотрим только отличия IDFE1X от нотации Чена.

Сущность (Entity) - реальный либо воображаемый объект, имеющий существенное значение для рассматриваемой предметной области. Каждая сущность должна иметь наименование, выраженное существительным в единственном числе. Каждая сущность должна обладать уникальным идентификатором. Каждый экземпляр сущности должен однозначно идентифицироваться и отличаться от всех других экземпляров данного типа сущности.

Атрибут (Attribute) - любая характеристика сущности, значимая для рассматриваемой предметной области и предназначенная для квалификации, идентификации, классификации, количественной характеристики или выражения состояния сущности. Наименование атрибута должно быть выражено существительным в единственном числе.

Связь (Relationship) - поименованная ассоциация между двумя сущностями, значимая для рассматриваемой предметной области.

В методе IDEF1X все сущности делятся на зависимые и независимые от идентификаторов. Сущность является независимой от идентификаторов или просто независимой, если каждый экземпляр сущности может быть однозначно идентифицирован без определения его отношений с другими сущностями. Сущность называется зависимой от идентификаторов или просто

зависимой, если однозначная идентификация экземпляра сущности зависит от его отношения к другой сущности. Независимая сущность изображается в виде обычного прямоугольника, зависимая - в виде прямоугольника с закругленными углами.

В IDEF1X существуют следующие виды мощностей связей:

- N мощность - каждый экземпляр сущности-родителя может иметь ноль, один или более одного связанного с ним экземпляра сущности-потомка (по умолчанию);
- P мощность - каждый экземпляр сущности-родителя должен иметь не менее одного связанного с ним экземпляра сущности-потомка;
- Z мощность - каждый экземпляр сущности-родителя должен иметь не более одного связанного с ним экземпляра сущности-потомка;
- конкретное число - каждый экземпляр сущности-родителя связан с некоторым фиксированным числом экземпляров сущности-потомка.

Связь изображается линией, проводимой между сущностью-родителем и сущностью-потомком, с точкой на конце линии у сущности-потомка. По умолчанию мощность связи принимается равной N. Если экземпляр сущности-потомка однозначно определяется своей связью с сущностью-родителем, то связь называется идентифицирующей, в противном случае — неидентифицирующей. Идентифицирующая связь изображается сплошной линией, неидентифицирующая - пунктирной линией.

В ERwin'e при установлении идентифицирующей связи атрибуты первичного ключа родительской сущности автоматически переносятся в состав первичного ключа дочерней сущности. Эта операция называется миграцией атрибутов. В дочерней сущности новые атрибуты помечаются как внешний ключ (FK). При установке неидентифицирующей связи атрибуты первичного ключа родительской сущности мигрируют в состав неключевых полей дочерней сущности.

Построение логической модели данных предприятия по сборке и продаже компьютеров и ноутбуков. Построение модели данных начинается с выделения сущностей данной предметной области. В нашем случае были выделены следующие сущности:

- клиент - человек, который покупает компьютеры
- заказ - список компьютеров, которые покупает клиент
- компьютер
- комплектующие - то, из чего собирают компьютеры
- сотрудник - сотрудник предприятия, собирающий конкретный компьютер

Далее рассмотрим связи между сущностями:

- Клиент - Заказ. Один клиент может делать несколько заказов. При этом если данные о клиенте имеются в базе данных, то он сделал минимум один заказ. Поэтому мощность связи - P. Связь идентифицирующая, т.к. заказ без клиента существовать не может;
- Заказ - Компьютер. В рамках одного заказа клиент может заказать несколько компьютеров, но как минимум заказ должен состоять из одного компьютера. Поэтому мощность связи - P. Связь идентифицирующая, т.к. компьютер без заказа существовать не может;
- Компьютер - Комплектующие. В состав одного компьютера входит много различных комплектующих; один и тот же тип комплектующего может входить в состав разных компьютеров. Мощность связи - много ко многим. В IDEF1X такой тип связи отсутствует, поэтому вводим промежуточную (ассоциативную) сущность - Конфигурация. Мощность связи между сущностями Компьютер и Конфигурация - P, поскольку у любого компьютера должна быть конфигурация, мощность между сущностями Комплектующие и Конфигурация - N, поскольку какие-то комплектующие еще могут быть не установлены ни в один компьютер. Связь в обоих случаях идентифицирующая, т.к. конфигурация компьютера не может существовать без привязки к самому компьютеру и к комплектующим;
- Комплектующие - Тип комплектующих. Поскольку перечень типов комплектующих, которые могут быть установлены в компьютер, ограничен, но используется очень часто, то мы приняли решение создать еще одну сущность - Тип комплектующих. Мощность связи - P. Связь идентифицирующая;
- Компьютер - Сотрудник. Каждый компьютер собирается каким-то одним сотрудником. Какие-то сотрудники могут собирать множество компьютеров. Мощность связи - N. Тип связи - неидентифицирующая, поскольку экземпляр сущности Компьютер уже может существовать, но за ним еще может быть не закреплен ни один сотрудник. Именно из этих же

соображений в свойствах этой связи мы выбрали переключатель "Nulls Allowed" (на диаграмме это отображается в виде незакрашенного ромбика со стороны сущности-родителя.

Задание 2:

Необходимо построить в нотации IDEF1X в CASE-средстве ERwin Data Modeler логическую схему данных предметной области. Примечание. При построении модели можно ограничиться 5-6 сущностями.

Итоговая диаграмма показана на Рис. 1:

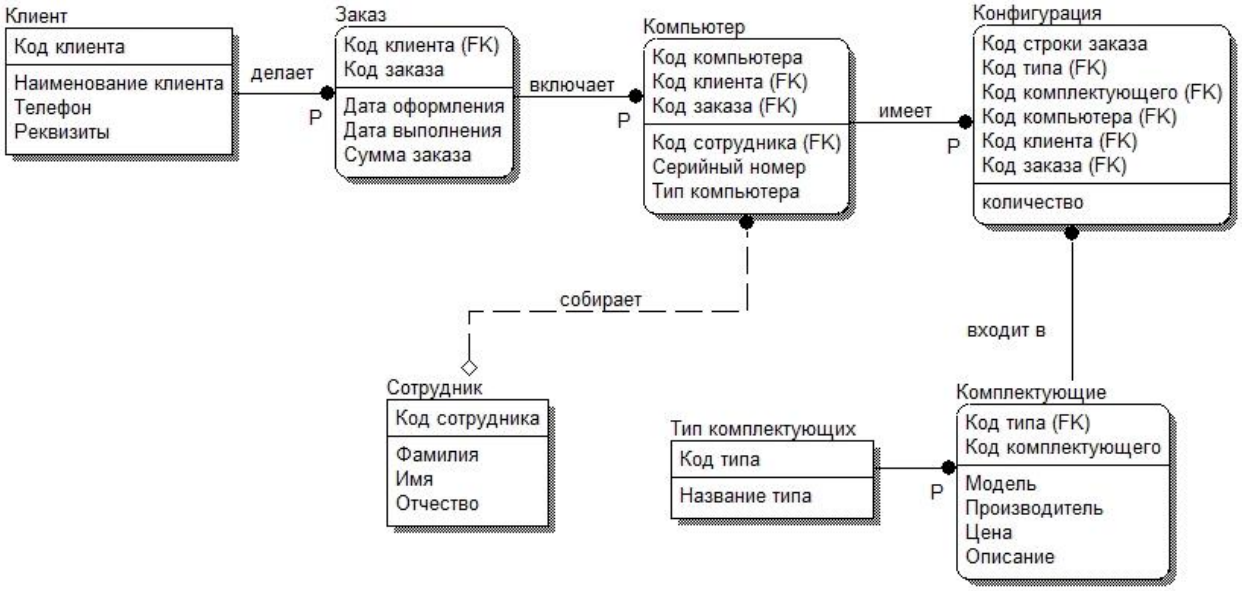


Рисунок 1-Логическая модель данных предприятия по сборке компьютеров и ноутбуков

Варианты заданий:

1) Налоговая инспекция

Таблицы:

- налогоплательщик: ФИО, адрес, категория, …
- налоги: наименование, процентная ставка, дата и размер платежа, …
- льготы: вид льготы, сумма льготы, процентная ставка, …

Правила:

- каждый налогоплательщик платит несколько разновидностей налогов
- каждый налогоплательщик может иметь несколько льгот

2) Земельный реестр

Таблицы:

- землевладелец: наименование, адрес, …
- участок земли: расположение, площадь, категория земельного участка, …
- строения на участке земли: наименование, площадь, стоимость, …

Правила:

- один землевладелец может иметь несколько участков, одним участком может распоряжаться несколько землевладельцев;
- на каждом участке может быть несколько строений;

3) Бухгалтерия

Таблицы:

- контрагенты: плательщики и получатели – наименование, адрес, банк, …
- платежи: дата, сумма, контрагент, …

Правила:

- каждый контрагент может выполнять несколько платежей
- в одном банке может быть зарегистрировано несколько контрагентов

4) Банк

Таблицы:

- счета: ФИО или наименование клиента, адрес, другие сведения, …
- операции по счету: зачисление или снятие средств, начисление процентов, …
- вклады: наименование, срок вклада, порядок начисления процентов, …

Правила:

- каждый клиент может иметь несколько вкладов
- с каждым счетом выполняется множество операций

5) Инвестиционный фонд

Таблицы:

- объект инвестиций: наименование, срок инвестиций, срок возврата, процент, …
- фирма-получатель инвестиций: наименование, адрес, …

Правила:

- фонд одновременно может инвестировать несколько проектов
- фонд получает платежи в несколько приемов
- фонд может осуществлять инвестирование в несколько этапов
- одна фирма может получать инвестиции на несколько различных объектов.

Итог работы: тетрадь, защита работы.

Практическая работа № 13

Цель: исследование процесса построения диаграмм потоков данных и диаграммы размещения в заданной предметной области

Задание 1.

Задание 1. Кодогенерация проекта в Delphi.

Теперь вся информация подготовлена к тому, чтобы запрограммировать классы с их методами и операциями.

Для выполнения кодогенерации в среде Delphi необходимо выполнить следующую последовательность действий:

- протестировать модель на логические непротиворечия;
- настроить (или проверить настройки) среду на законы кодогенерации (соответствие элемента модели Rose элементу кода Delphi);
- создать имя проекта Delphi и выполнить кодогенерацию.

Этапы выполнения упражнения.

1) Протестируйте модель Tools->Ceck Model. Просмотрите log файл на наличие ошибок. Если файл не виден- выполните команду file->Save Log As и введите имя файла (по умолчанию error.log). Затем его просмотрите и, при необходимости, исправьте ошибки. К наиболее распространенным ошибкам относятся такие, как неотображение сообщений на операции или несоотнесение объектов с классами на диаграммах взаимодействия. С помощью пункта меню Check Model можно выявить большую часть неточностей и ошибок в модели.

2) Пункт меню Access Violations позволяет обнаружить нарушения правил доступа, возникающие тогда, когда существует связь между двумя классами разных пакетов. При этом связи между самими пакетами нет. Например, если существует связь между классами Order пакета Entities и OrderManager пакета Control, то обязательно должна существовать и связь между пакетами Entities и Control. Если последняя связь не установлена, Rose выявит нарушение правил доступа. Чтобы обнаружить нарушение правил доступа:

Выберите в меню Report > Show Access Violations.

Проанализируйте все нарушения правил доступа.

3) Выполните Tools>Options>Notation>Default Language и из выпадающего списка выберите язык программирования Delphi.

4) Проверьте правильность установок кодогенерации по умолчанию (default). Для этого выполните Tools->Options->Delphi и последовательно переберите из выпадающего списка поля Type все элементы. Сравните установки в поле Model Propities с данными (default) из таблицы Приложения А. В случае несоответствия- исправьте.

5) Выполните Tools> Ensemble Tools>Rose Delphi Link (рис.14)

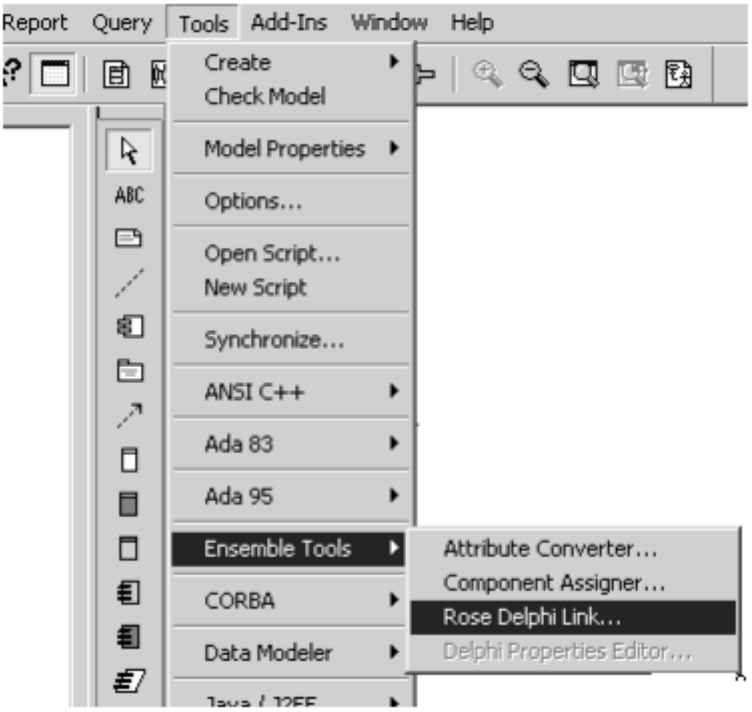


Рис. 14. Меню для выбора процесса кодогенерации

В результате появится соответствующая экранная форма. Выполните на этой форме File>New Proect. Появится форма с браузером. Введите имя файла и место на диске, куда будет сохранено имя сгенерированного проекта в Delphi. Например, NewProect.dpr и нажмите Открыть. В результате форма примет вид (рис. 15)

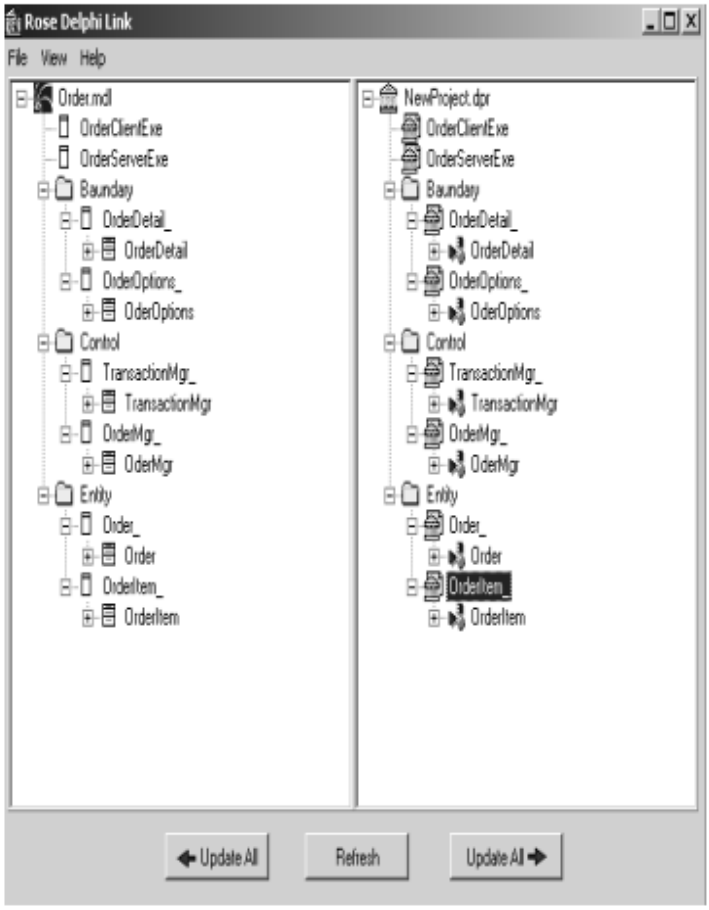


Рис. 15. Представление результатов кодогенерации в окне Rose Delphi Link

5. Через проводник Windows найдите папку проекта Delphi. С помощью программы Блокнот просмотрите содержимое всех файлов. В приложении В к руководству приведено содержимое всех файлов проекта.

Задание 2. Анализ Delphi проекта, добавление визуальных объектов, реинжениринг в Rose

1) Запустите на выполнение программу Delphi и загрузите сгенерированный проект (Proect1.dpr). Проверьте, что проект содержит все модули и присмотрите их содержимое через редактор Delphi.

2) Создайте в проекте Delphi новую форму с Name Form1. Поместите на форму компонент MainMenu (главное меню)

3) С помощью Menu Disigner введите две позиции горизонтального меню с названиями (полями Caption) Oder и OderItem.

4) Для Oder введите две строки вертикального меню с Caption Create и SubmitInfo. Для OderItem введите одну строку вертикального меню с названием GetInfo.

5) Сохраните проект в Delphi.

Реинжениринг Delphi проекта в модель Rose

1) Вернитесь в проект Rose и откройте окно проектов Rose Delphi Link. Проверьте, что открыт именно тот проект, для которого выполнялась кодогенерация.

2) Курсором мыши нажмите клавишу Update ALL со стрелкой влево (обновление модели Rose на основе изменений проекта Delphi). В результате в моделе Rose должны произойти определенные изменения (рис. 16):

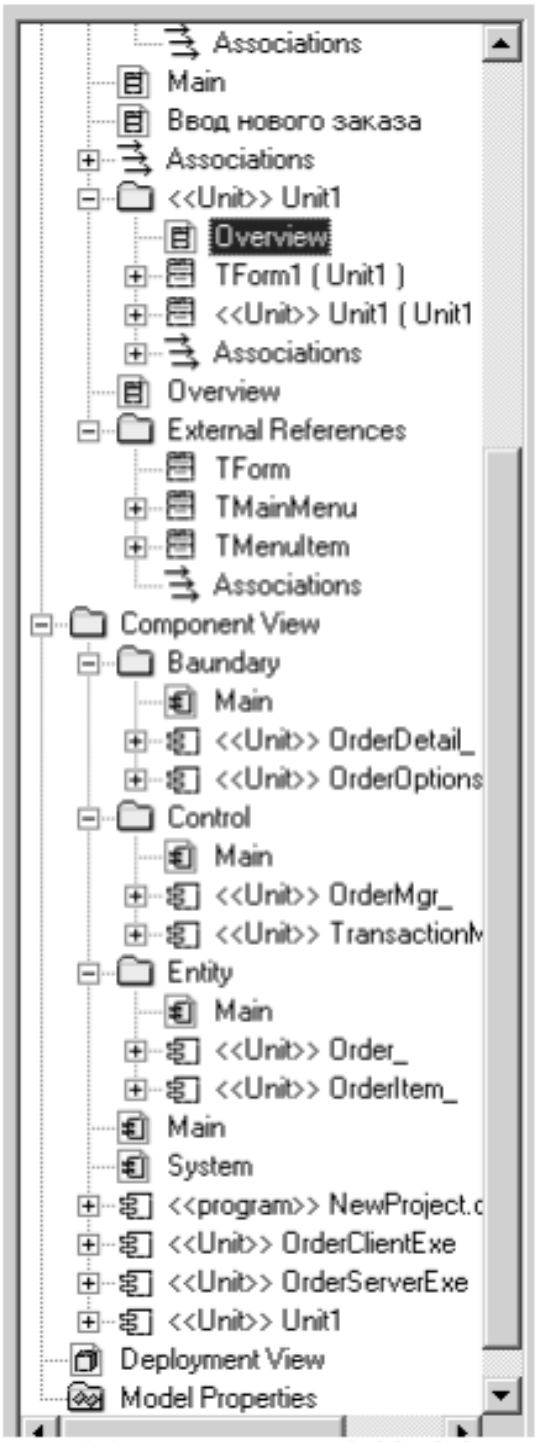


Рис. 16. Окно Rose Delphi Link после кодогенерации

- в представление Logic View создался новый пакет Unit1 и External References (Внешние ссылки). Внутри второго пакета созданы три класса TForm, TMainMenu и TMenuItem, которые использовались при развитии проекта Delphi. Отметим, что эта папка не создалась бы, если бы мы при первоначальном создании проекта включили в него пакет классов Delphi FreimWork.

- в этом же представлении в пакете Unit1 создан класс TForm1 и Unit1 оба соотнесенные с вновь созданным компонентом Unit1. Кроме того, в этом же пакете создалась диаграмма классов Overview, содержимое которой показано на рис.17

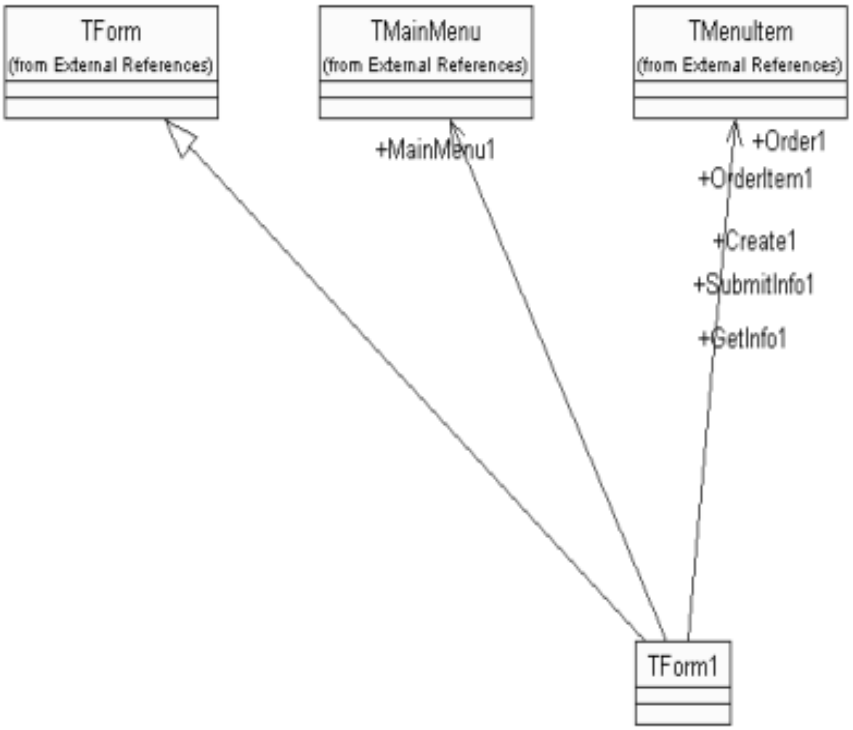


Рис. 17. Результаты реинжениринга проекта Delphi в Rose

Задание 3. Построение диаграммы размещения

В этом задании создается диаграмма Размещения для системы обработки заказов.

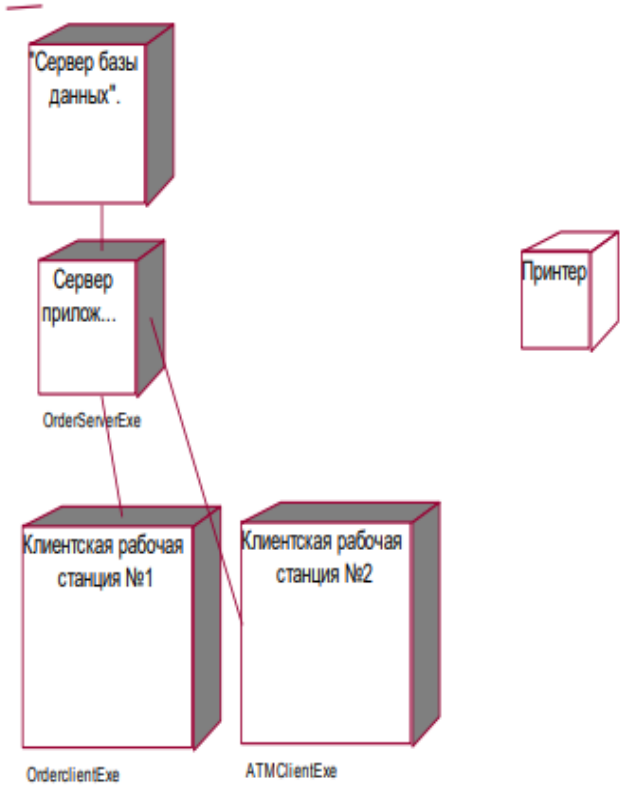


Рис. 18 Диаграмма размещения для модельной задачи

Этапы выполнения

Добавление узлов к диаграмме Размещения

1. Дважды щелкнув мышью на представлении Размещения в браузере, откройте диаграмму Размещения.
2. Нажмите кнопку Processor (Процессор) панели инструментов.
3. Щелкнув мышью на диаграмме, поместите туда процессор.
4. Введите имя процессора "Сервер базы данных".
5. Повторив шаги 2—4, добавьте следующие процессоры:

-Сервер приложения
- Клиентская рабочая станция №1
- Клиентская рабочая станция №2

6. На панели инструментов нажмите кнопку Devices (Устройство).
7. Щелкнув мышью на диаграмме, поместите туда устройство.
8. Назовите его "Принтер".

Добавление связей

1. Нажмите кнопку Connection (Связь) панели инструментов.
2. Щелкните мышью на процессоре "Сервер базы данных".
3. Проведите линию связи к процессору "Сервер приложения".
4. Повторив шаги 1 — 3, добавьте следующие связи;

- От процессора "Сервер приложения" к процессору "Клиентская рабочая станция №1"
- От процессора "Сервер приложения" к процессору "Клиентская рабочая станция №2"
- От процессора "Сервер приложения" к устройству "Принтер"

Добавление процессов

1. Щелкните правой кнопкой мыши на процессоре "Сервер приложения" в браузере.
2. В открывшемся меню выберите пункт New > Process (Создать > Процесс),
3. Введите имя процесса — OrderServerExe.
4. Повторив шаги 1 —- 3, добавьте процессы:

- Процесс OrderclientExe на процессоре "Клиентская рабочая станция №1"
- Процесс ATMClientExe на процессоре "Клиентская рабочая станция №2"

Показ процессов на диаграмме

1. Щелкните правой кнопкой мыши на процессоре "Сервер приложения".
2. В открывшемся меню выберите пункт Show Process (Показать процессы).
3. Повторив шаги 1 и 2, покажите процессы на следующих процессорах:

- Клиентская рабочая станция №1
- Клиентская рабочая станция №2

Итог работы: тетрадь, защита работы.

Практическая работа № 14

Цель: усвоить знания о видах тестирования, способы обнаружения и фиксирования ошибок

Задание 1.

Ознакомьтесь со сведениями

Общепринятая практика состоит в том, что после завершения продукта и до передачи его заказчику независимой группой тестировщиков проводится тестирование ПО.

Уровни тестирования:

Модульное тестирование. Тестируется минимально возможный для тестирования компонент, например отдельный класс или функция;

Интеграционное тестирование. Проверяется, есть ли какие-либо проблемы в интерфейсах и взаимодействии между интегрируемыми компонентами, например, не передается информация, передается некорректная информация;

Системное тестирование. Тестируется интегрированная система на ее соответствие исходным требованиям.

Таблица 1. Виды некоторых ошибок и способы их обнаружения

Виды программных ошибок	Способы их обнаружения
Ошибки выполнения, выявляемые автоматически: а) переполнение, защита памяти; б) несоответствие типов; в) зацикливание	Динамический контроль: аппаратурой процессора; run-time системы программирования; операционной системой – по превышению лимита времени

Тест – это набор контрольных входных данных совместно с ожидаемыми результатами. Тесты должны обладать определенными свойствами.

Детективность: тест должен с большой вероятностью обнаруживать возможные ошибки.

Покрывающая способность: один тест должен выявлять как можно больше ошибок.

Воспроизводимость: ошибка должна выявляться независимо от изменяющихся условий.

С помощью тестирования разных видов обнаруживаются ошибки в разрабатываемом программном обеспечении. После обнаружения ошибок проводится их устранение.

Задание 2:

Задание

Создать приложение Простой калькулятор, в котором реализовать выполнение простых операций с вводимыми двумя операндами. Выполнить тестирование приложения на различных данных, отличающихся по типу и значению.

Программа работы

1. Разработать интерфейс приложения и написать программные коды для событий кнопок.
2. Сохранить проект в отдельной папке, скопировать исполняемый файл на рабочий стол.
3. Составить тесты для проверки работы приложения.
4. Провести тестирование исполняемого файла

Составить отчет по итогам тестирования и рекомендации по устранению выявленных ошибок

Итог работы: тетрадь, защита работы.

Практическая работа № 15

Цель: усвоить знания о видах тестирования, способы обнаружения и фиксирования ошибок

Задание 1. Разработайте и создайте приложение по любой тематике, реализующее выполнение простых операций

Итог работы: тетрадь, защита работы.

Практическая работа № 16

Цель: получить бавыки разработки текстовых сценариев

Задание 1:

Задание № 1

Написать программу решения квадратного уравнения ax2 + bx + c = 0.

Задание № 2

Найти минимальный набор тестов для программы нахождения веще-ственных корней квадратного уравнения ax2 + bx + c = 0. Решение представлено в таблице.

Номер теста	a	b	c	Ожидаемый результат	Что проверяется
1	2	-5	2	x_1=2, x_2=0,5	Случай вещественных корней
2	3	2	5	Сообщение	Случай комплексных корней
3	3	-12	0	x_1=4, x_2=0	Нулевой корень
4	0	0	10	Сообщение	Неразрешимое уравнение
5	0	0	0	Сообщение	Неразрешимое уравнение
6	0	5	17	Сообщение	Неквадратное уравнение
7	9	0	0	x_1=x_2=0	Нулевые корни

Таким образом, для этой программы предлагается минимальный набор функциональных тестов, исходя из 7 классов выходных данных.

Заповеди по отладки программного средства, предложенные Г. Майерсом.

Заповедь 1. Считайте тестирование ключевой задачей разработки ПС, поручайте его самым квалифицированным и одаренным программистам, нежелательно тестировать свою собственную программу.

Заповедь 2. Хорош тот тест, для которого высока вероятность обнаружить ошибку, а не тот, который демонстрирует правильную работу программы.

Заповедь 3. Готовьте тесты как для правильных, так и для неправильных данных.

Заповедь 4. Документируйте пропуск тестов через компьютер, детально изучайте результаты каждого теста, избегайте тестов, пропуск которых нельзя повторить. Заповедь 5. Каждый модуль подключайте к программе только один раз, никогда не изменяйте программу, чтобы облегчить ее тестирование.

Заповедь 6. Пропускайте заново все тесты, связанные с проверкой работы какой-либо программы ПС или ее взаимодействия с другими программами, если в нее были внесены изменения (например, в результате устранения ошибки).

Итог работы: тетрадь, защита работы.

Практическая работа № 17

Цель: получить бавыки разработки текстовых сценариев

Задание 1: ответьте на вопросы

- Оценка стоимости и причины ошибок в программном обеспечении.
- Виды и методы тестирования.
- Понятие теста.
- Требования к разработке тестовых сценариев.
- Правила разработки тестовых сценариев.

Задание 2:

Разработайте набор тестовых сценариев (как позитивных, так и негативных) для следующей программы:

Имеется консольное приложение (разработайте самостоятельно). Ему на вход подается 2 строки. На выходе приложение выдает число вхождений второй строки в первую. Например:

Строка 1	Строка 2	Вывод
абвгабвг	аб	2
стстсап	стс	2

Набор тестовых сценариев запишите в виде таблицы, приведенной выше.

Итог работы: тетрадь, защита работы.

Практическая работа № 18

Цель: получить навыки разработки текстовых пакетов

Задание 1:

Задание № 1

В Древней Греции (II в. до н.э.) был известен шифр, называемый "квадрат Полибия". Шифровальная таблица представляла собой квадрат с пятью столбцами и пятью строками, которые нумеровались цифрами от 1 до 5. В каждую клетку такого квадрата записывалась одна буква. В результате каждой букве соответствовала пара чисел, и шифрование сводилось к замене буквы парой чисел. Для латинского алфавита квадрат Полибия имеет вид:

	1	2	3	4	5
1	A	B	C	D	E
2	F	G	H	I, J	K
3	L	M	N	O	P
4	Q	R	S	T	U
5	V	W	X	Y	Z

Пользуясь изложенным способом создать программу, которая:

а) зашифрует введенный текст и сохранит его в файл;

б) считает зашифрованный текст из файла и расшифрует данный текст.

Задание № 2

Спроектировать тесты по принципу «белого ящика» для программы, разработанной в задании № 1. Выбрать несколько алгоритмов для тестирования и обозначить буквами или цифрами ветви этих алгоритмов. Выписать пути алгоритма, которые должны быть проверены тестами для выбранного метода тестирования. Записать тесты, которые позволят пройти по путям алгоритма. Протестировать разработанную вами программу. Результаты оформить в виде таблиц:

Тест	Ожидаемый результат	Фактический результат	Результат тестирования
…	…	…	…

Задание № 3

Проверить все виды тестов и сделать выводы об их эффективности

Задание № 4

Итог работы: тетрадь, защита работы.

Практическая работа № 19

Цель: получить навыки разработки текстовых пакетов

Задание 1: ответьте на вопросы

- Системные основы разработки требований к сложным комплексам программ.
- Формализация эталонов требований и характеристик комплекса программ.
- Формирование требований компонентов и модулей путем декомпозиции функций комплексов программ.
- Тестирование по принципу «белого ящика».

Задание 2:

Разработайте задание, демонстрирующее возможности тестовых пакетов, реализуйте его

Итог работы: тетрадь, защита работы.

Практическая работа № 20

Цель:

знакомство с ГОСТ 28.195-89 «Оценка качества программных средств. Общие положения»; определить способы получения информации о ПС; формирование информационно - правовых компетенции обучающихся.

Задание 1:

Ознакомьтесь с материалом

Необходимая документация: ГОСТ 28.195-89

Одной из важнейших проблем обеспечения качества программных средств является формализация характеристик качества и методология их оценки. Для определения адекватности качества функционирования, наличия технических возможностей программных средств к взаимодействию, совершенствованию и развитию необходимо использовать стандарты в области оценки характеристик их качества.

Показатели качества программного обеспечения устанавливают ГОСТ 28.195-89 «Оценка качества программных средств. Общие положения» и ГОСТ Р ИСО/МЭК 9126 «Информационная технология. Оценка программной продукции. Характеристика качества и руководства по их применению». Одновременное существование двух действующих стандартов, нормирующих одни и те же показатели, ставит вопрос об их гармонизации. Ниже рассмотрим каждый из перечисленных стандартов.

ГОСТ 28.195-89 «Оценка качества программных средств. Общие положения» устанавливает общие положения по оценке качества программных средств, номенклатуру и применяемость показателей качества.

Оценка качества ПС представляет собой совокупность операций, включающих выбор номенклатуры показателей качества оцениваемого ПС, определение значений этих показателей и сравнение их с базовыми значениями.

Методы определения показателей качества ПС различаются: по способам получения информации о ПС – измерительный, регистрационный, органолептический, расчетный; по источникам получения информации – экспертный, социологический.

Измерительный метод основан на получении информации о свойствах и характеристиках ПС с использованием инструментальных средств. Например, с использованием этого метода определяется объем ПС - число строк исходного текста программ и число строк - комментариев, число операторов и операндов, число исполненных операторов, число ветвей в программе, число точек входа (выхода), время выполнения ветви программы, время реакции и другие показатели.

Регистрационный метод основан на получении информации во время испытаний или функционирования ПС, когда регистрируются и подсчитываются определенные события, например, время и число сбоев и отказов, время передачи управления другим модулям, время начала и окончания работы.

Органолептический метод основан на использовании информации, получаемой в результате анализа восприятия органов чувств (зрения, слуха), и применяется для определения таких показателей как удобство применения, эффективность и т. п.

Расчетный метод основан на использовании теоретических и эмпирических зависимостей (на ранних этапах разработки), статистических данных, накапливаемых при испытаниях, эксплуатации и сопровождении ПС. При помощи расчетного метода определяются длительность и точность вычислений, время реакции, необходимые ресурсы.

Определение значений показателей качества ПС *экспертным методом* осуществляется группой экспертов-специалистов, компетентных в решении данной задачи, на базе их опыта и интуиции. Экспертный метод применяется в случаях, когда задача не может быть решена никаким другим из существующих способов или другие способы являются значительно более трудоемкими. Экспертный метод рекомендуется применять при определении показателей наглядности, полноты и доступности программной документации, легкости освоения, структурности.

Социологические методы основаны на обработке специальных анкет-вопросников.

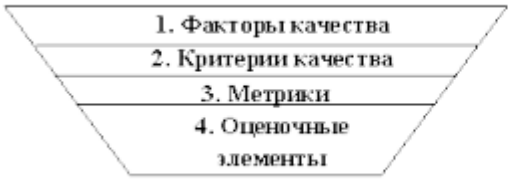


Рис. 1 – Уровни системы показателей качества

Показатели качества объединены в систему из четырех уровней. Каждый вышестоящий уровень содержит в качестве составляющих показатели нижестоящих уровней (рисунок 1).

Стандарт ИСО 9126 (ГОСТ Р ИСО/МЭК 9126) «Информационная технология. Оценка программной продукции. Характеристика качества и руководства по их применению».

Определенные настоящим стандартом характеристики дополнены рядом требований по выбору метрик и их измерению для различных проектов ПС. Они применимы к любому типу ПС, включая компьютерные программы и данные, содержащиеся в программируемом оборудовании. Эти характеристики обеспечивают согласованную терминологию для анализа качества ПС. Кроме того, они определяют схему для выбора и специфицирования требований к качеству ПС, а также для сопоставления возможностей различных программных продуктов, таких как функциональные возможности, надежность, практичность и эффективность.

Все множество атрибутов качества ПС может быть классифицировано в структуру иерархического дерева характеристик и субхарактеристик. Самый высший уровень этой структуры состоит из характеристик качества, а самый нижний уровень – из их атрибутов. Эта иерархия не строгая, поскольку некоторые атрибуты могут быть связаны с более чем одной субхарактеристикой. Таким же образом, внешние свойства (такие, как пригодность, корректность, устойчивость к ошибкам или временная эффективность) влияют на наблюдаемое качество. Недостаток качества в использовании (например, пользователь не может закончить задачу) может быть прослежен к внешнему качеству (например, функциональная пригодность или простота использования) и связанным с ним внутренним атрибутам, которые необходимо изменить.

Внутренние метрики могут применяться в ходе проектирования и программирования к неисполняемым компонентам ПС (таким, как спецификация или исходный программный текст). При разработке ПС промежуточные продукты следует оценивать с использованием внутренних метрик, которые измеряют свойства программ, и могут быть выведены из моделируемого поведения. Основная цель внутренних метрик – обеспечивать, чтобы было достигнуто требуемое внешнее качество. Внутренние метрики дают возможность пользователям, испытателям и разработчикам оценивать качество ЖЦ программ и заниматься вопросами технологического обеспечения качества задолго до того, как ПС становится готовым исполняемым продуктом.

Внутренние метрики позволяют измерять внутренние атрибуты или формировать признаки внешних атрибутов путем анализа статических свойств промежуточных или поставляемых программных компонентов. Измерения внутренних метрик используют категории, числа или характеристики элементов из состава ПС, которые, например, имеются в процедурах исходного программного текста, в графе потока управления, в потоке данных и в представлениях изменения состояний памяти. Документация также может оцениваться с использованием внутренних метрик.

Внешние метрики используют меры ПС, выведенные из поведения системы, частью которых они являются, путем испытаний, эксплуатации или наблюдения исполняемого ПС или системы. Перед приобретением или использованием ПС его следует оценить с использованием метрик, основанных на деловых и профессиональных целях, связанных с использованием, эксплуатацией и управлением продуктом в определенной организационной и технической среде. Внешние метрики обеспечивают заказчикам, пользователям, испытателям и разработчикам возможность определять качество ПС в ходе испытаний или эксплуатации.

Когда требования к качеству ПС определены, в них должны быть перечислены характеристики и субхарактеристики, которые составляют полный набор показателей качества. Затем определяются подходящие внешние метрики и их приемлемые диапазоны значений, устанавливающие количественные и качественные критерии, которые подтверждают, что ПС удовлетворяет потребностям заказчика и пользователя. Далее определяются и специфицируются внутренние атрибуты качества, чтобы спланировать удовлетворение требуемых внешних характеристик качества в конечном продукте и обеспечивать их в промежуточных продуктах в ходе разработки. Подходящие внутренние метрики и приемлемые диапазоны специфицируются для получения числовых значений или категорий внутренних характеристик качества, чтобы их можно было использовать для проверки того, что промежуточные продукты в процессе разработки удовлетворяют внутренним спецификациям качества. Рекомендуется использовать внутренние метрики, которые имеют наиболее сильные связи с целевыми внешними метриками, чтобы они могли помогать при прогнозировании значений внешних метрик.

Метрики качества в использовании измеряют, в какой степени продукт удовлетворяет потребности конкретных пользователей в достижении заданных целей с результативностью,

продуктивностью и удовлетворением в заданном контексте использования. При этом результативность подразумевает точность и полноту достижения определенных целей пользователями при применении ПС; продуктивность соответствует соотношению израсходованных ресурсов и результативности при эксплуатации ПС, а удовлетворенность – психологическое отношение к качеству использования продукта. Эта метрика не входит в число шести базовых характеристик ПС, регламентируемых стандартом ИСО 9126, однако рекомендуется для интегральной оценки результатов функционирования комплексов программ.

Оценивание качества в использовании должно подтверждать его для определенных сценариев и задач, оно составляет полный объединенный эффект характеристик качества ПС для пользователя. *Качество в использовании* – это восприятие пользователем качества системы, содержащей ПС, и оно измеряется скорее в терминах результатов использования комплекса программ, чем собственных внутренних свойств ПС. Связь качества в использовании с другими характеристиками качества ПС зависит от типа пользователя, так, например, для конечного пользователя качество в использовании обусловливают, в основном, характеристики функциональных возможностей, надежности, практичности и эффективности, а для персонала сопровождения ПС качество в использовании определяет сопровождаемость. На качество в использовании могут влиять любые характеристики качества, и это понятие шире, чем практичность, которая связана с простотой использования и привлекательностью. Качество в использовании, в той или иной степени, характеризуется сложностью применения комплекса программ, которую можно описать трудоемкостью использования с требуемой результативностью. Многие характеристики и субхарактеристики ПС обобщенно отражаются неявными технико-экономическими показателями, которые поддерживают функциональную пригодность конкретного ПС. Однако их измерение и оценка влияния на показатели качества, представляет сложную проблему.

Задание 2:

Задание №1. Провести сравнение понятий «качество» государственным и международным стандартами. Выписать документы, в которых даны данные определения.

Задание №2. Опишите методы получения информации о ПС по ГОСТу. Для каждого метода выделите источник информации.

Задание №3. Выберите стандарты для оценки качества ПС. Перечислите критерии надежности ПС по ГОСТу.

Задание №4. Методика оценки качественных показателей ПП основана на составлении метрики ПП. В лабораторной работе необходимо выполнить следующее:

1. Выбрать показатели качества (не менее 5) и сформулировать их сущность. Каждый показатель должен быть существенным, т. е. должны быть ясны потенциальные выгоды его использования. Показатели представить в виде таблицы (таблица 1).

Показатели качества	Сущность показателя	Экспертная оценка (вес) wi	Оценка, установленная экспериментом ri

2. Установить веса показателей wi (?wi =1).

3. Для каждого показателя установить конкретную численную оценку ri от 0 до 1, исходя из следующего:

§ 0 – свойство в ПП присутствует, но качество его неприемлемо;

§ 0.5 — 1 – свойство в ПП присутствует и обладает приемлемым качеством;

§ 1 – свойство в ПП присутствует и обладает очень высоким качеством.

§ Возможно, присвоение промежуточных значений в соответствии с мнением оценивающего лица относительно полезности того или иного свойства ПП.

$$K = \frac{\sum w_i \cdot r_i}{общее\_количество\_показателей}$$

Результатом выполнения данной работы является отчет

Итог работы: тетрадь, защита работы.

Практическая работа № 21

Цель:

знакомство с ГОСТ 28.195-89 «Оценка качества программных средств. Общие положения»; определить способы получения информации о ПС; формирование информационно - правовых компетенции обучающихся.

Задание 1:

Разработайте задание, демонстрирующее возможности оценки программных средств с помощью метрик, реализуйте его

Итог работы: тетрадь, защита работы.

Практическая работа № 22

Цель: Научиться выполнять реорганизацию программного кода на основе шаблонов рефакторинга

Задание 1:

Задание №1

Выполнить анализ программного кода для разрабатываемого ПО и модульных тестов с целью плохо организованного кода.

Задание №2

Используя шаблоны рефакторинга, выполнить реорганизацию программного кода разрабатываемого ПО.

Задание №3

Выполнить описание произведенных операций рефакторинга (было-стало-шаблон рефакторинга).

Задание №4

В случае необходимости скорректировать проектную документацию.

Задание №5

Сделать выводы по результатам выполнения работ.

Класс Main

Было:

```
пакетная игра;
импорт игры. Персонажи. *;
импорт игры. Персонажи. Персонажи;
импорт игры. Энергетика. Энергетика;
импорт игры. Энергетика. Освещение;
импорт игры. Уровни. Блок;
импорт игры. Уровни. Уровень;
импортировать game.Levels.Level_data;
импорт игры. Weapon.Bullet;
импорт игры. Оружие. Оружие;
import javafx.animation.AnimationTimer;
импорт javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.image.Image;
import javafx.scene.input.KeyCode;
import javafx.scene.layout.Pane;
import javafx.stage.Stage;
import java.io.DataInputStream;
import java.io.FileInputStream;
импорт java.io.IOException;
import java.util.ArrayList;
import java.util.HashMap;
публичный класс Main расширяет приложение {
public static ArrayList <Block> blocks = new ArrayList <> ();
public static ArrayList <Bullet> bullets = new ArrayList <> ();
public static ArrayList <Bullet> врагаBullets = новый ArrayList <> ();
public static ArrayList <EnemyBase> враги = новый ArrayList <> ();
static HashMap <KeyCode, Boolean> keys = new HashMap <> ();
публичная статическая сцена этапа;
публичная статическая сцена;
public static Pane gameRoot = new Pane ();
public static Pane appRoot = new Pane ();
публичное статическое меню;
публичный статический Персонаж букера;
публичная статическая HUD HUD;
статическое оружие общего назначения;
публичная статика елизавета елизавета;
статический VendingMachine vendingMachine;
статическое учебное пособие;
```

```
частные статические CutScenes cutScene;
публичная статика Энергетика энергетика;
публичная статическая молния молнии;
public static int levelNumber;
уровень статического уровня;
public static AnimationTimer timer = new AnimationTimer () {
@Override
public void handle (давно) {
Обновить();
}
};
приватное статическое void update () {
для (EnemyBase враг: враги) {
enemy.update ();
if (врага.getDelete ()) {
enemies.remove (враг);
перемена;
}
}
Bullet.update ();
Controller.update ();
booker.update ();
if (! energetic.getName (). equals (""))
energetic.update ();
if (levelNumber> 0)
elizabeth.update ();
если (молния! = ноль) {
lightning.update ();
if (lightning.getDelete ())
молния = ноль;
}
menu.update ();
hud.update ();
weapon.update ();
if (booker.getTranslateX ()> Level_data.BLOCK_SIZE * 295)
cutScene = новые CutScenes (levelNumber);
}
@Override
public void start (Stage primaryStage) выдает исключение {
stage = primaryStage;
сцена = новая сцена (appRoot, 1280, 720);
. AppRoot.getChildren () добавить (gameRoot);
   уровень = новый уровень ();
   try (DataInputStream dataInputStream = new DataInputStream (новый FileInputStream ("C:
/DeadShock/saves/data.dat"))) {
   levelNumber = dataInputStream.readInt ();
   level.createLevels (levelNumber);
   level.changeImageView (levelNumber);
   vendingMachine = new VendingMachine ();
   букер = новый персонаж ();
   booker.setMoney (dataInputStream.readInt ());
   booker.setSalt (dataInputStream.readByte ());
   booker.setCountLives (2);
   оружие = новое оружие ();
   weapon.setWeaponClip (dataInputStream.readInt ());
   weapon.setBullets (dataInputStream.readInt ());
   hud = новый HUD ();
```

```
Елизавета = новая Елизавета ();
энергичный = новый Энергетический ();
} catch (IOException e) {
levelNumber = 0;
level.createLevels (levelNumber);
vendingMachine = new VendingMachine ();
букер = новый персонаж ();
оружие = новое оружие ();
hud = новый HUD ();
энергичный = новый Энергетический ();
tutorial = new Tutorial (levelNumber);
}
switch (levelNumber) {
случай 0:
        враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 117, 200));
        враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 127, 200));
        враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 148, 200));
        враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 161, 200));
        враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 171, 200));
        враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 185, 200));
        враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 204, 200));
        враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 215, 200));
        враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 228, 200));
        враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 233, 200));
        враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 243, 200));
        враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 252, 200));
        враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 262, 200));
        враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 277, 200));
        враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 280, 200));
        враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 286, 200));
        перемена;
        Дело 1:
        враги.адд (новый EnemyRedEye (Level_data.BLOCK_SIZE * 57, 200));
        враги.адд (новый EnemyRedEye (Level_data.BLOCK_SIZE * 67, 200));
        враги.адд (новый EnemyRedEye (Level_data.BLOCK_SIZE * 74, 200));
        враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 87, 200));
        враги.адд (новый EnemyRedEye (Level_data.BLOCK_SIZE * 104, 150));
        враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 117, 200));
        враги.адд (новый EnemyRedEye (Level_data.BLOCK_SIZE * 133, 200));
        враги.адд (новый EnemyRedEye (Level_data.BLOCK_SIZE * 156, 200));
        враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 177, 200));
        враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 193, 200));
        враги.адд (новый EnemyRedEye (Level_data.BLOCK_SIZE * 201, 200));
        враги.адд (новый EnemyRedEye (Level_data.BLOCK_SIZE * 216, 200));
        враги.адд (новый EnemyRedEye (Level_data.BLOCK_SIZE * 224, 200));
        враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 246, 200));
        враги.адд (новый EnemyRedEye (Level_data.BLOCK_SIZE * 260, 200));
        враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 277, 100));
        Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 34,
Level_data.BLOCK_SIZE * 13));
        Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 36,
Level_data.BLOCK_SIZE * 13));
        Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 60,
Level_data.BLOCK_SIZE * 9));
        Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 61,
Level_data.BLOCK_SIZE * 9));
```

```
        Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 106,
Level_data.BLOCK_SIZE * 7));
        Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 107,
Level_data.BLOCK_SIZE * 7));
        Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 168,
Level_data.BLOCK_SIZE * 13));
        Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 170,
Level_data.BLOCK_SIZE * 13));
        Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 196,
Level_data.BLOCK_SIZE * 13));
        Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 197,
Level_data.BLOCK_SIZE * 13));
        Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 232,
Level_data.BLOCK_SIZE * 8));
        Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 233,
Level_data.BLOCK_SIZE * 8));
        перемена;
        }
        меню = новое меню ();
        appRoot.getChildren () добавить (menu.menuBox).
        booker.translateXProperty (). addListener (((наблюдаемый, oldValue, newValue) -> {
        int offset = newValue.intValue ();
        if (offset> 600 && offset <gameRoot.getWidth () - 680) {
        gameRoot.setLayoutX (- (смещение - 600));
        level.getBackground (). setLayoutX ((смещение - 600) / 1,5);
        }
        если (смещение <= 100)
        level.getBackground () setLayoutX (0).
        }));
        vendingMachine.createButtons ();
        stage.getIcons (). add (новое изображение ("файл: / C: /DeadShock/images/icon.jpg"));
        stage.setTitle ( "DeadShock");
        stage.setResizable (ложь);
        stage.setWidth (scene.getWidth ());
        stage.setHeight (scene.getHeight ());
        stage.setScene (сцены);
        stage.show ();
        timer.start ();
        }
        public static void main (String [] args) {
        запуск (арг);
        }
        }
```

Стало:

```
пакетная игра;
импорт игры. Персонажи. *;
импорт игры. Персонажи. Персонажи;
импорт игры. Энергетика. Энергетика;
импорт игры. Энергетика. Освещение;
импорт игры. Уровни. Блок;
импорт игры. Уровни. Уровень;
импортировать game.Levels.Level_data;
импорт игры. Weapon.Bullet;
импорт игры. Оружие. Оружие;
import javafx.animation.AnimationTimer;
импорт javafx.application.Application;
import javafx.scene.Scene;
```

```
import javafx.scene.image.Image;
import javafx.scene.input.KeyCode;
import javafx.scene.layout.Pane;
import javafx.stage.Stage;
import java.io.DataInputStream;
import java.io.FileInputStream;
импорт java.io.IOException;
import java.util.ArrayList;
import java.util.HashMap;
публичный класс Main расширяет приложение {
public static ArrayList <Block> blocks = new ArrayList <> ();
public static ArrayList <Bullet> bullets = new ArrayList <> ();
public static ArrayList <Bullet> врагаBullets = новый ArrayList <> ();
public static ArrayList <EnemyBase> враги = новый ArrayList <> ();
static HashMap <KeyCode, Boolean> keys = new HashMap <> ();
публичная статическая сцена этапа;
публичная статическая сцена;
public static Pane gameRoot = new Pane ();
public static Pane appRoot = new Pane ();
публичное статическое меню;
публичный статический Персонаж букера;
публичная статическая HUD HUD;
статическое оружие общего назначения;
публичная статика елизавета елизавета;
статический VendingMachine vendingMachine;
статическое учебное пособие;
частные статические CutScenes cutScene;
публичная статика Энергетика энергетика;
публичная статическая молния молнии;
public static int levelNumber;
уровень статического уровня;
public static AnimationTimer timer = new AnimationTimer () {
@Override
public void handle (давно) {
Обновить();
}
};
private void initContent () {
. AppRoot.getChildren () добавить (gameRoot);
уровень = новый уровень ();
try (DataInputStream dataInputStream = new DataInputStream (новый FileInputStream ("C:
/DeadShock/saves/data.dat"))) {
levelNumber = dataInputStream.readInt ();
level.createLevels (levelNumber);
level.changeImageView (levelNumber);
vendingMachine = new VendingMachine ();
букер = новый персонаж ();
booker.setMoney (dataInputStream.readInt ());
booker.setSalt (dataInputStream.readByte ());
booker.setCountLives (2);
оружие = новое оружие ();
weapon.setWeaponClip (dataInputStream.readInt ()
weapon.setBullets (dataInputStream.readInt ());
hud = новый HUD ();
Елизавета = новая Елизавета ();
энергичный = новый Энергетический ();
} catch (IOException e) {
```

```
levelNumber = 0;
level.createLevels (levelNumber);
vendingMachine = new VendingMachine ();
букер = новый персонаж ();
оружие = новое оружие ();
hud = новый HUD ();
энергичный = новый Энергетический ();
tutorial = new Tutorial (levelNumber);
}
createEnemies ();
меню = новое меню ();
appRoot.getChildren () добавить (menu.menuBox).
booker.translateXProperty (). addListener (((наблюдаемый, oldValue, newVal
int offset = newValue.intValue ();
if (offset> 600 && offset <gameRoot.getWidth () - 680) {
gameRoot.setLayoutX (- (смещение - 600));
level.getBackground (). setLayoutX ((смещение - 600) / 1,5);
}
если (смещение <= 100)
level.getBackground () setLayoutX (0).
}));
vendingMachine.createButtons ();
}
public static void createEnemies () {
switch (levelNumber) {
случай 0:
враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 117, 200));
враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 127, 200));
враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 148, 200));
враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 161, 200));
враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 171, 200));
враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 185, 200));
враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 204, 200));
враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 215, 200));
враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 228, 200));
враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 233, 200));
враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 243, 200));
враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 252, 200));
враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 262, 200));
враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 277, 200));
враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 280, 200));
враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 286, 200));
перемена;
Дело 1:
враги.адд (новый EnemyRedEye (Level_data.BLOCK_SIZE * 57, 200));
враги.адд (новый EnemyRedEye (Level_data.BLOCK_SIZE * 67, 200));
враги.адд (новый EnemyRedEye (Level_data.BLOCK_SIZE * 74, 200));
враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 87, 200));
враги.адд (новый EnemyRedEye (Level_data.BLOCK_SIZE * 104, 150));
враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 117, 200));
враги.адд (новый EnemyRedEye (Level_data.BLOCK_SIZE * 133, 200));
враги.адд (новый EnemyRedEye (Level_data.BLOCK_SIZE * 156, 200));
враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 177, 200));
враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 193, 200));
враги.адд (новый EnemyRedEye (Level_data.BLOCK_SIZE * 201, 200));
враги.адд (новый EnemyRedEye (Level_data.BLOCK_SIZE * 216, 200));
враги.адд (новый EnemyRedEye (Level_data.BLOCK_SIZE * 224, 200));
```

```
            враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 246, 200));
            враги.адд (новый EnemyRedEye (Level_data.BLOCK_SIZE * 260, 200));
            враги.адд (новый EnemyComstock (Level_data.BLOCK_SIZE * 277, 100));
            Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 34,
  Level_data.BLOCK_SIZE * 13));
            Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 36,
  Level_data.BLOCK_SIZE * 13));
            Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 60,
  Level_data.BLOCK_SIZE * 9));
            Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 61,
  Level_data.BLOCK_SIZE * 9));
            Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 106,
  Level_data.BLOCK_SIZE * 7));
            Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 107,
  Level_data.BLOCK_SIZE * 7));
            Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 168,
  Level_data.BLOCK_SIZE * 13));
            Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 170,
  Level_data.BLOCK_SIZE * 13));
            Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 196,
  Level_data.BLOCK_SIZE * 13));
            Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 197,
  Level_data.BLOCK_SIZE * 13));
            Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 232,
  Level_data.BLOCK_SIZE * 8));
            Level_data.enemyBlocks.add (новый блок («невидимый», Level_data.BLOCK_SIZE * 233,
  Level_data.BLOCK_SIZE * 8));
            перемена;
            }
            }
            приватное статическое void update () {
            для (EnemyBase враг: враги) {
            enemy.update ();
            If (enemy.GetDelete()) {
            enemies.remove(enemy);
            break;
}
}
Bullet.update();
Controller.update();
booker.update();
If (!energetic.GetName().Equals(""))
energetic.update();
if (levelNumber > 0)
elizabeth.update();
if (lightning != null) {
lightning.update();
If (lightning.GetDelete())
lightning = null;
}
menu.update();
hud.update();
weapon.update();
if (booker.getTranslateX() > Level_data.BLOCK_SIZE * 295)
cutScene = new CutScenes(levelNumber);
}
@Override
```

```
public void start(Stage primaryStage) throws Exception {
stage = primaryStage;
scene = new Scene(appRoot, 1280, 720);
initContent();
stage.getIcons().add(new Image("file:/C:/DeadShock/images/icon.jpg"));
stage.setTitle("DeadShock");
stage.setResizable(false);
stage.setWidth(scene.getWidth());
stage.setHeight(scene.getHeight());
stage.setScene(scene);
stage.show();
timer.start();
}
public static void main(String[] args) {
launch(args);
}
}
```

Шаблон рефакторинга: Выделение метода(Extract Method)

Итог работы: отчет, защита работы.

Раздел 2. Средства разработки программного обеспечения

Практическая работа № 1

Цель: формирование навыков постановки задачи и разработки технического задания на программный продукт

Задание1: площадь квадрата по заданной стороне.

1. Выбрать вариант задания на проектирование и разработку учебной программы.
2. В соответствии с вариантом выполнить разработку технического задания, которое должно включать:
   - введение;
   - основание для разработки;
   - назначение;
   - требования к программе и программному продукту;
   - требования к программной документации.
3. Оформить отчет. Содержание отчета:
   - тема лабораторной работы
   - цель лабораторной работы
   - ответы на контрольные вопросы
   - задание на лабораторную работу
   - разработанное техническое задание
   - выводы по проделанной работе.

Варианты заданий

1. Ввести вещественную матрицу размерности n * m построчно, а вывести по столбцам.
2. Выяснить сколько положительных элементов содержит матрица размерности n * m , если a ij = sin(i+j/2).
3. Дана квадратная вещественная матрица размерности n. Является ли матрица симметричной относительно главной диагонали.
4. Дана квадратная вещественная матрица размерности n. Транспонировать матрицу.
5. Дана квадратная вещественная матрица размерности n. Сравнить сумму элементов матрицы на главной и побочной диагоналях.
6. Дана квадратная вещественная матрица размерности n. Найти количество нулевых элементов, стоящих:
   выше главной диагонали;
   ниже главной диагонали;
   выше и ниже побочной.
7. Дана вещественная матрица размерности n * m. По матрице получить логический вектор, присвоив его k-ому элементу значение True , если выполнено указанное условие и значение False иначе:
   все элементы k столбца нулевые;
   элементы k строки матрицы упорядочены по убыванию;
   k строка массива симметрична.
8. Дана вещественная матрица размерности n * m. Сформировать вектор b, в котором элементы вычисляются как:
   произведение элементов соответствующих строк;
   среднее арифметическое соответствующих столбцов;
   разность наибольших и наименьших элементов соответствующих строк;
   значения первых отрицательных элементов в столбце.
9. Дана вещественная матрица размерности n * m. Вывести номера столбцов, содержащих только отрицательные элементы.
10. Дана вещественная матрица размерности n * m. Вывести номера строк, содержащих больше положительных элементов, чем отрицательных.
11. Дана вещественная матрица размерности n * m. Найти общую сумму элементов только тех столбцов, которые имеют хотя бы один нулевой элемент.
12. Дана вещественная матрица размерности n * m. Поменять местами строки с максимальным и минимальным элементами.
13. Дана вещественная матрица размерности n * m. Удалить k столбец матрицы.
14. Дана вещественная квадратная матрица размерности n. Поменять местами элементы главной и побочной диагоналей матрицы:

    по строкам;
    по столбцам.
15. Дана вещественная матрица размерности m * n. Упорядочить элементы каждой четной строки по возрастанию.
16. Дана вещественная матрица размерности m * n. Расположить все элементы матрицы по убыванию. Обход матрицы осуществлять по строкам.
17. Дана вещественная матрица размерности m * n. Определить индексы первого нулевого элемента матрицы. Обход матрицы осуществлять по столбцам.
18. Известно положение двух ферзей на шахматной доске. Бьют ли они друг друга?

Контрольные вопросы

1. Перечислите этапы разработки программных продуктов.
2. Для чего необходимо техническое задание?
3. Кто занимается разработкой технического задания?
4. Какие пункты включает техническое задание?

Итог работы: отчет, защита работы.

Практическая работа № 2

Цель: Научиться разрабатывать перечни артефактов и протоколы проекта

Задание1:

Системный анализ и пути решения задачи

При разработке ПС человек имеет дело с системами. Под *системой* будем понимать совокупность взаимодействующих (находящихся в отношениях) друг с другом элементов. ПС можно рассматривать как пример системы. Логически связанный набор программ является другим примером системы. Любая отдельная программа также является системой. Понять систему — значит осмысленно перебрать все пути взаимодействия между ее элементами.

Целью системного анализа в наиболее общем виде является описание и исследование систем, определение путей и методов разработки ПО. Система характеризуется структурой и поведением. Применительно к разработке ПО системный анализ представляет собой анализ существующей структуры отношений в рамках конкретной предметной области, выявление роли и места будущей программной системы, ее основных функций и свойств. В этой связи системный анализ также можно назвать *внешним проектированием*.

Этап системного анализа состоит из следующих трех стадий:

1. обоснование необходимости разработки программы;
2. научно-исследовательские работы (НИР);
3. разработка и утверждение технического задания.

На первой стадии выполняются постановка задачи, сбор исходных материалов, Выбор и обоснование критериев эффективности и качества разрабатываемой программы, обоснование необходимости проведения научно-исследовательских работ.

На стадии научно-исследовательских работ решаются следующие задачи: определяется структура входных и выходных данных, осуществляется предварительный выбор методов решения задач, обосновывается целесообразность применения ранее разработанных программ, определяются требования к техническим средствам, обосновывается принципиальная возможность решения поставленной задачи.

На стадии разработки и утверждения технического задания определяются требования к программе, разрабатываются технико-экономического обоснования разработки программ, определяются стадии, этапы и сроки разработки программы и документации на нее, согласовывается и утверждается*техническое задание*.

Результат системного анализа — **спецификация** (техническое задание) как самостоятельный документ имеет очень важное значение. Этот документ является формальным соглашением между заказчиком продукта и его разработчиками.

В настоящее время можно выделить пять основных подходов к организации процесса создания и использования программного обеспечения.

1. *Водопадный подход*. При таком подходе разработка ПС состоит из цепочки этапов. На каждом этапе создаются документы, используемые на последующем этапе. В исходном документе фиксируются требования к ПС. В конце этой цепочки создаются программы, включаемые в ПС.
2. *Исследовательское программирование*. Этот подход предполагает быструю (насколько это возможно) реализацию рабочих версий программ ПС, выполняющих лишь в первом приближении требуемые функции. После экспериментального применения реализованных программ производится их модификация с целью сделать их более полезными для пользователей. Этот процесс повторяется до тех пор, пока ПС не будет достаточно приемлемо для пользователей. Такой подход применялся на ранних этапах развития программирования, когда технологии программирования не придавали большого значения (использовалась интуитивная технология). В настоящее время этот подход применяется для разработки таких ПС, для которых пользователи не могут точно сформулировать требования (например, для разработки систем искусственного интеллекта).
3. *Прототипирование*. Этот подход моделирует начальную фазу исследовательского программирования вплоть до создания рабочих версий программ, предназначенных для проведения экспериментов с целью установить требования к ПС. В дальнейшем должна последовать разработка ПС по установленным требованиям в рамках какого-либо другого подхода (например, водопадного).
4. *Формальные преобразования*. Этот подход включает разработку формальных спецификаций ПС и превращение их в программы путем корректных преобразований.

На этом подходе базируется компьютерная технология (CASE-технология) разработки ПС.

5. *Сборочное программирование*. Этот подход предполагает, что ПС конструируется, главным образом, из компонентов, которые уже существуют. Должно быть некоторое хранилище (библиотека) таких компонентов, каждая из которых может многократно использоваться в разных ПС. Такие компоненты называются *повторно используемыми(reusable)*. Процесс разработки ПС при данном подходе состоит скорее из сборки программ из компонентов, чем из их программирования.

Задание

1. В соответствии с подготовленным техническим заданием выполнить разработку спецификаций на программный продукт, которые должны включать:
   - спецификации процессов;
   - словарь терминов;
   - диаграммы переходов состояний;
   - диаграммы потоков с детализацией.
2. Оформить отчет. Содержание отчета:
   - тема лабораторной работы;
   - цель лабораторной работы;
   - ответы на контрольные вопросы;
   - задание на лабораторную работу;
   - разработанные спецификации процессов;
   - словарь терминов;
   - диаграммы переходов состояний;
   - диаграммы потоков с детализацией;
   - выводы по проделанной работе.

Контрольные вопросы

1. Для чего разрабатываются спецификации на программный продукт?
2. Что должны включать спецификации на программный продукт?
3. Что должна содержать спецификация процессов
4. Что такое словарь терминов и для чего он используется?
5. Что такое диаграмма переходов состояний и для чего ее используют?
6. Что такое диаграмма потоков и для чего ее используют?

Итог работы: отчет, защита работы.

Практическая работа № 3

Цель: отработать навыки настройки работы системы контроля версий

Задание1:

Система управления/контроля версиями (от англ. Version Control System или Revision Control System) — программное обеспечение для облегчения работы с изменяющейся информацией. Система управления версиями позволяет хранить несколько версий одного и того же документа, при необходимости, возвращаться к более ранним версиям, определять, кто и когда сделал то или иное изменение и многое другое.

Такие системы наиболее широко применяются при разработке программного обеспечения, для хранения исходных кодов разрабатываемой программы. Однако, они могут с успехом применяться и в других областях, в которых ведётся работа с большим количеством непрерывно изменяющихся электронных документов, в частности, они всё чаще применяются в САПР, обычно, в составе систем управления данными об изделии (PDM). Управление версиями используется в инструментах конфигурационного управления (Software Configuration Management Tools).

Распространённые системы управления версиями

- Subversion
- Darcs
- Microsoft Visual SourceSafe
- Bazaar
- Rational ClearCase
- Perforce
- BitKeeper
- Mercurial
- Git
- GNU Arch
- CVS — устаревшая. Потомок: Subversion
- RCS — устаревшая. Потомок: CVS

Основные понятия

Репозиторий (repository) – центральное хранилище, которое содержит версии файлов. Очень часто репозиторий организуется средствами какой-нибудь СУБД.

Версия файла (revision) – состояние файла в определенный момент времени. Репозиторий предоставляет возможность хранить неограниченное число версий одного и того же файла.

Актуальная версия файла – обычно это самая последняя версия файла, размещенного в репозитории.

Рабочая версия файла (working copy) – версия файла, с которой в текущий момент ведется работа, и которая не загружена в репозиторий.

Загрузка (Upload) – размещение файла в репозитории. В процессе загрузки в репозиторий помещается рабочая версия файла.

Выгрузка (Checkout) – получение файла из репозитория. В процессе выгрузки осуществляется получение из репозитория необходимой версии файла.

Синхронизация (update, sync) – приведение в соответствие рабочих версий файлов с актуальными версиями в репозитории. В процессе синхронизации в репозиторий загружаются те файлы, рабочие копии которых являются более "свежими" (т.е. имеют более поздние версии), по сравнению с файлами в репозитории, и выгружаются те файлы, рабочие копии которых устарели по сравнению с копиями в репозитории.

Borland StarTeam

Borland StarTeam – очень мощный и функциональный кросс-платформенный продукт, разрабатываемый в прошлом фирмой StarBase, которую Borland приобрела в конце 2002 г. Заметное преимущество данного решения состоит в том, что версия 2005 выступает центральным элементом стратегии управления жизненным циклом приложений (Application Lifecycle Management, ALM) компании Borland и обладает расширенными возможностями интеграции со всеми ее ключевыми пакетами, используемыми при разработке программного обеспечения.

MS SourceSafe

Microsoft Visual SourceSafe (Visual SourceSafe, VSS) — программный продукт компании Майкрософт, файл-серверная система управления версиями, предназначенная для небольших команд разработчиков. VSS позволяет хранить в общем хранилище файлы, разделяемые несколькими пользователями, для каждого файла хранится история версий. VSS входит в состав

- Subversion
  - https://github.com/irgups/project_2015_01
  - https://github.com/irgups/project_2015_02

Итог работы: отчет, защита работы.

Практическая работа № 4

Цель: Приобретение навыков проектирования структур данных.

Задание1:

1. Начало работы в Microsoft SQL Server Management Studio

Для создания баз данных используем среду Microsoft SQL Server Management Studio.

На запрос соединения с сервером выбираем (рис. 1):

Тип сервера: **Компонент Database Engine**

Имя сервера: **SQL-MS.**

Под таким именем в домене fizmat.vspu.ru. доступна машина, на которой установлены серверные компоненты MS SQL Server 2005. Можно попробовать выбрать сервер из выпадающего списка серверов. Можно также обратиться к этой машине по IP-адресу 192.168.10.152 из локальной сети.

Проверка подлинности: **Проверка подлинности SQL Server.**

Такая настройка позволяет создавать пользователей данного экземпляра SQL Server независимо от компьютера, с которого производится вход.

Имя входа: **studentMBS21.**

Пароль: **student.**

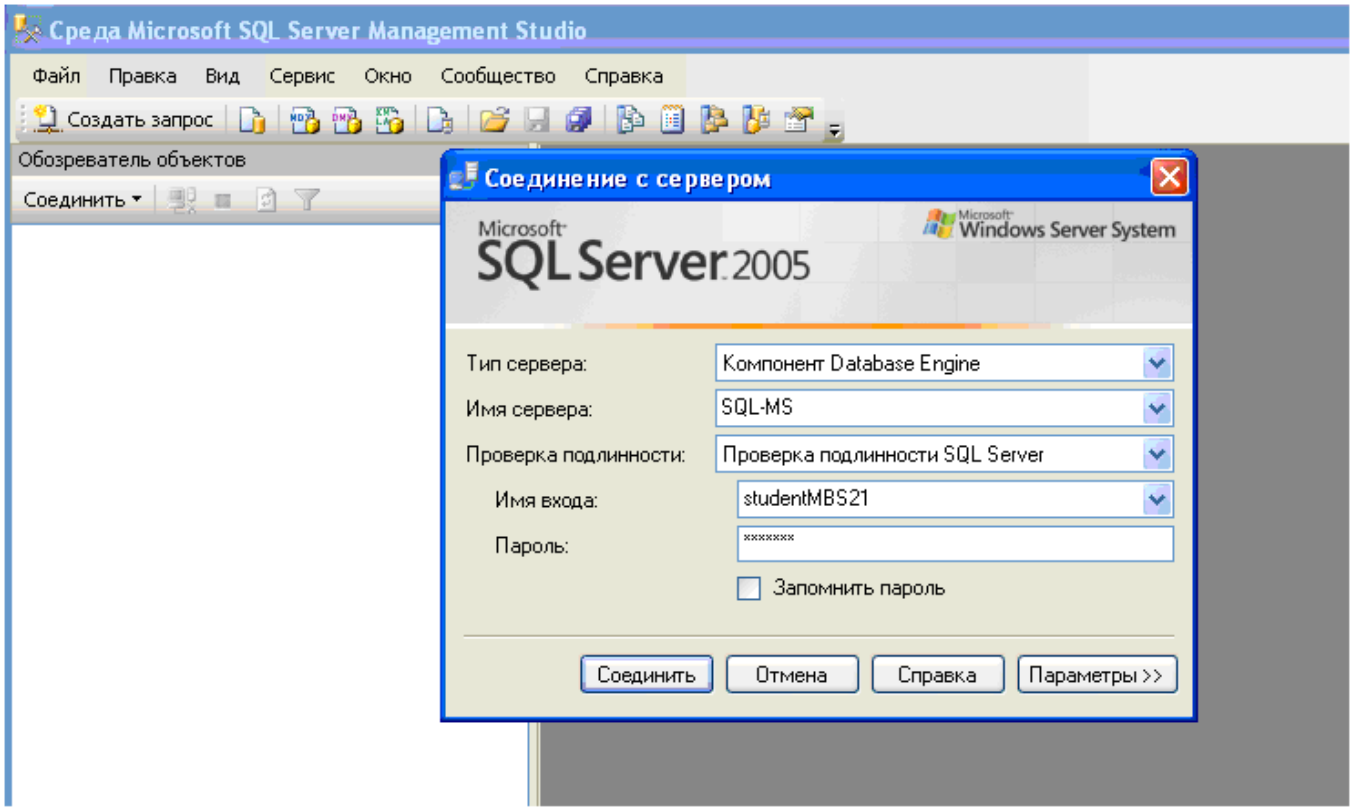


Рисунок 1. Окно входа в Microsoft SQL Server Management Studio 2005

Примечание. Пользователь studentMBS21 обладает большими полномочиями на этом сервере, поэтому пользоваться им надо очень аккуратно. Под этим пользователем мы создадим базу данных, а заполнять её и производить поиск по ней мы будем под другими пользователями. Предпочтительнее всего использовать свою учетную запись в домене fizmat.vspu.ru. В этом случае надо выбирать проверку подлинности Windows.

Теперь нажимаем кнопку «Параметры» и выбираем (рис. 2):

Соединение с базой данных → Обзор сервера… → Пользовательские базы данных → trial_base.

Сетевой протокол → TCP/IP

Нажимаем кнопку «Соединить».

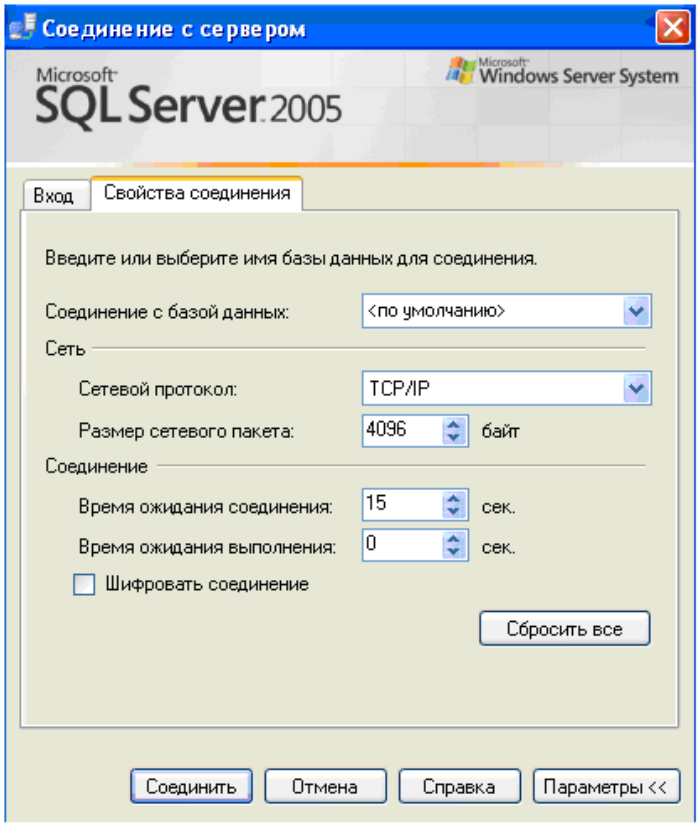


Рисунок 2. Окно входа в Microsoft SQL Server Management Studio 2005 (вкладка Параметры)

Примечание. База данных trial_base является базой данной по умолчанию для пользователя studentMBS21, она была создана при регистрации этого пользователя. В случае, когда права доступа пользователя не ограничены (как в рассматриваемом случае), вкладку Параметры можно не открывать. Если же пользователь имеет доступ только к определенным базам данных, при подключении к серверу нужно одну из этих баз указывать.

После успешного соединения с базой данных на экране видим следующую картинку (рис. 3):

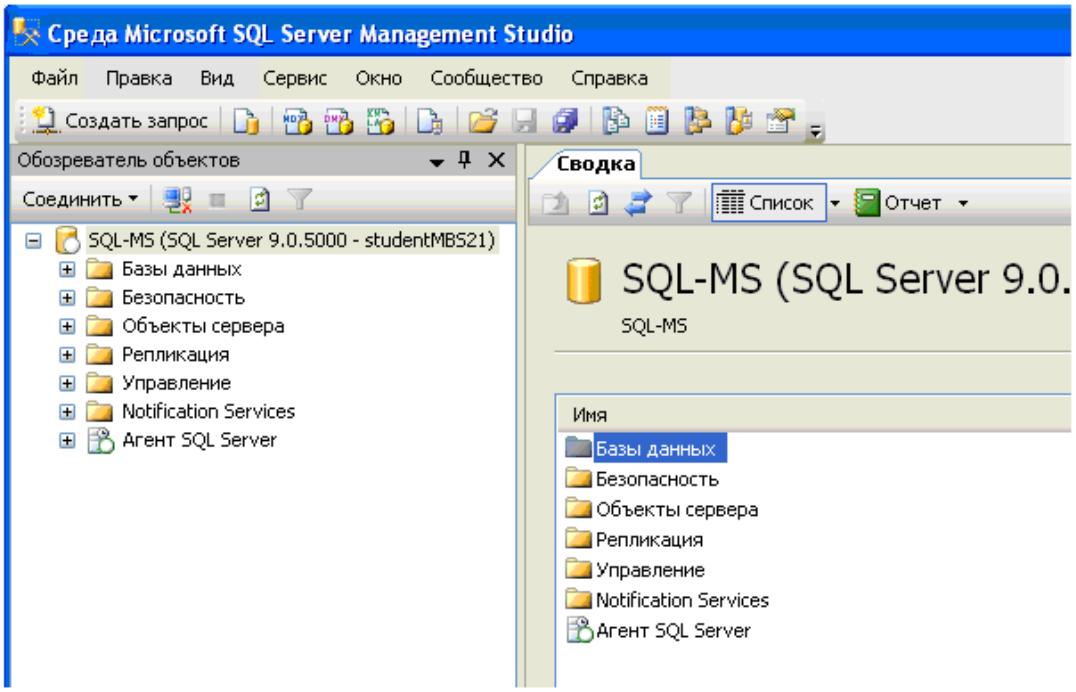


Рисунок 3. Подключение к SQL - серверу установлено

Среда MS SQL Management Studio предоставляет удобный инструментарий для создания, редактирования, заполнения баз данных. Но настоящие профессионалы в своей работе редко пользуются этой средой, а для выполнения своих задач используют SQL-запросы. Мы будем пользоваться, когда это удобно и наглядно, графическим режимом, но основной упор будем делать на освоении базы языка SQL.

2. Создание базы данных в среде Microsoft SQL Server Management Studio

В разделе «Базы данных» правой кнопкой выбираем «Создать базу данных…» (рис. 4). Назовем базу данных по индексу группы – mbs21. Владельцем базы данных назначим пользователя, под именем которого был произведен вход – studentMBS21. В разделе «Параметры» выбираем тип сортировки Cyrillic_General_BIN (для примера), нажимаем ОК.

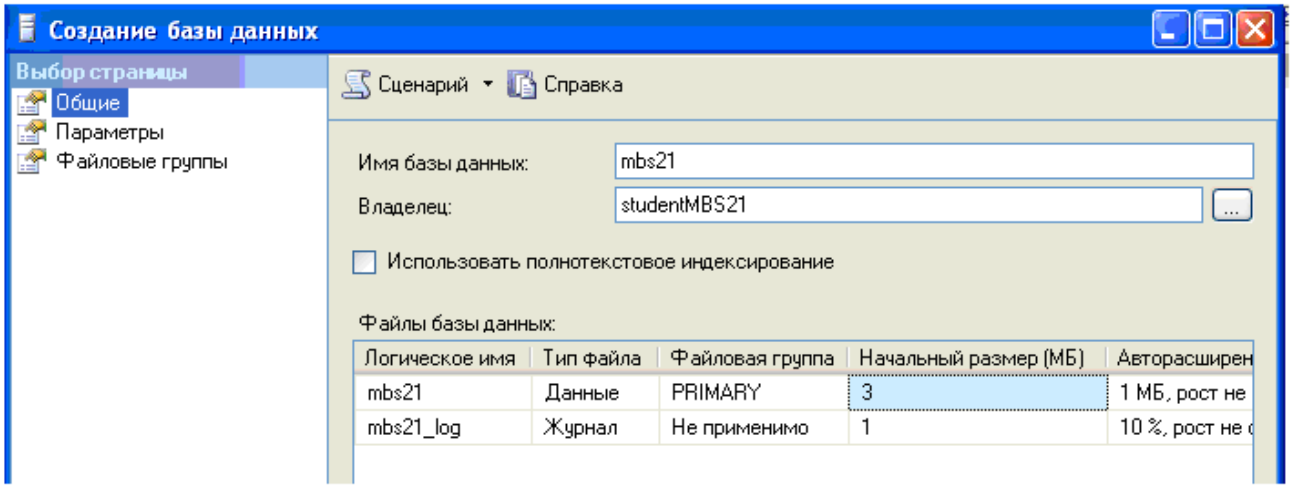


Рисунок 4. Создание базы данных

В разделе «Базы данных» Обозревателя объектов появилась вновь созданная mbs21 (проверьте!):

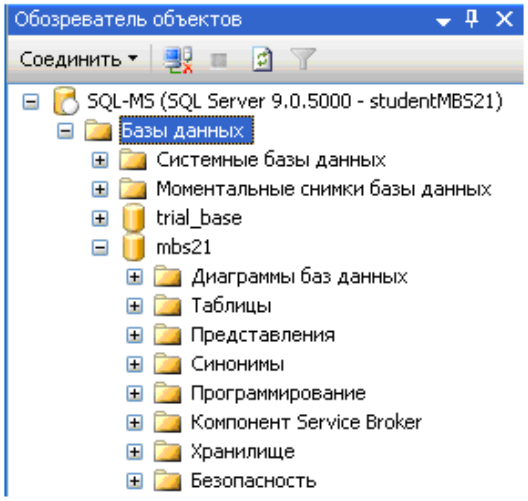


Рисунок 5. Обозреватель объектов

3. Создание таблиц базы данных в среде Microsoft SQL Server Management Studio

Начнем с создания таблицы Speciality. Структура таблицы приведена ниже:

Имя поля (столбца)	Содержание	Тип данных	Возможность содержать NULL
Num	Первичный ключ	int	нет
Name	Название	varchar(60)	нет

	специальности		

В реляционных базах данных первичный ключ используется как уникальный идентификатор записи. Это поле является обязательным, оно используется для связи таблиц по внешним ключам (примеры такого связывания будут рассмотрены далее). Первичный ключ должен иметь целочисленный тип (в данном случае - *int*). Во втором поле будет храниться название специальности - некоторая строка, поэтому мы выбираем для этого поля тип *varchar(60)*. Число в скобках означает максимальное число символов в строке. Детальную информацию об этих типах можно посмотреть в справке.

Простейшим образом можно создавать таблицы средствами MS SQL Server Management Studio (правая кнопка мыши на заголовке «Таблицы» > Создать таблицу.). Получаем следующее:

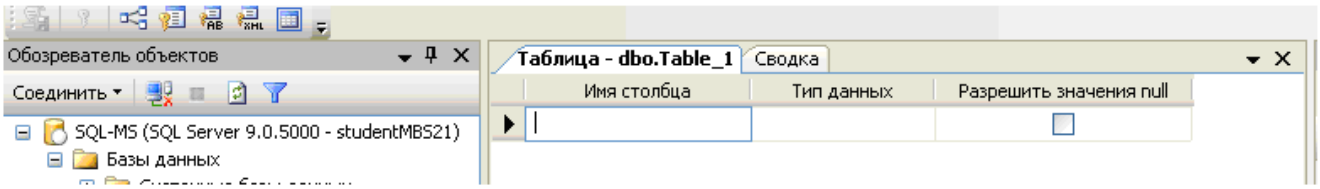


Рисунок 6. Создание таблицы

Вводим имя первого столбца Num (первичный ключ – в том столбце хранится номер записи), выбираем из выпадающего списка тип данных int. Первичный ключ не может быть пустым, поэтому и оставляем неотмеченным поле «Разрешить значения null». Затем аналогичным образом вводим имя второго столбца, задаем тип, запрещаем полю иметь значение null. Таблица принимает следующий вид:

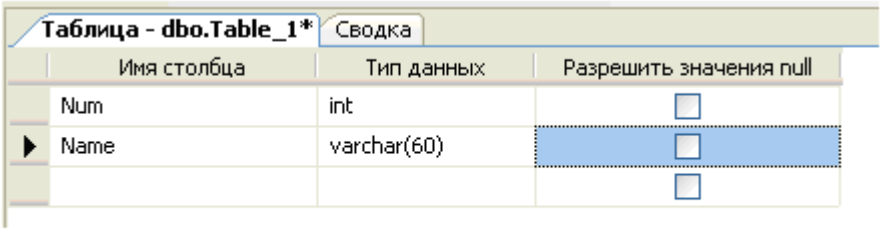
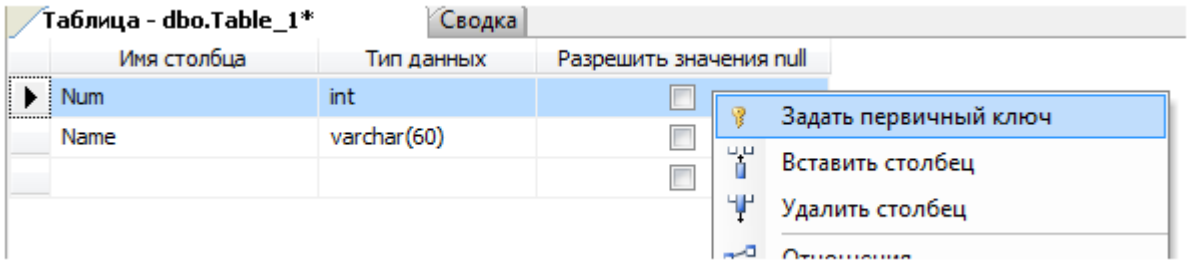


Рисунок 7.

Теперь необходимо указать, что поле *Num* будет являться первичным ключом. Правой кнопкой мыши щелкаем по этому полю и выбираем «Задать первичный ключ»:



Сохраняем таблицу под именем Speciality (после этого таблица должна появиться в обозревателе объектов). Теперь можно перейти к заполнению этой таблицы (для этого нужно в обозревателе объектов выбрать эту таблицу и в контекстном меню нажать «Открыть таблицу»):

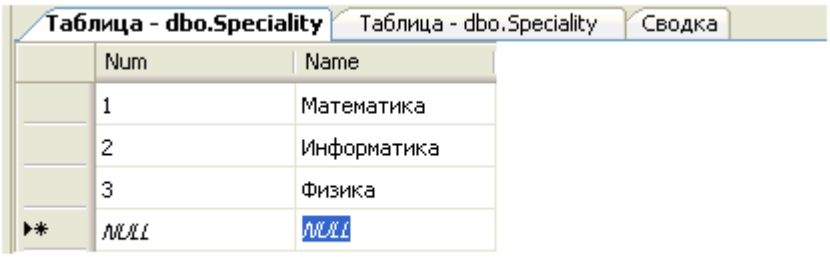


Рисунок 9.

При заполнении вы обнаружите, что каждый раз приходится вводить не только полезную информацию (название специальности), но и номер записи. Чтобы вводить номер записи автоматически, нужно задать спецификацию идентифицирующего столбца. Для этого необходимо в свойствах столбца указать, что данный столбец является идентифицирующим (рис. 10):

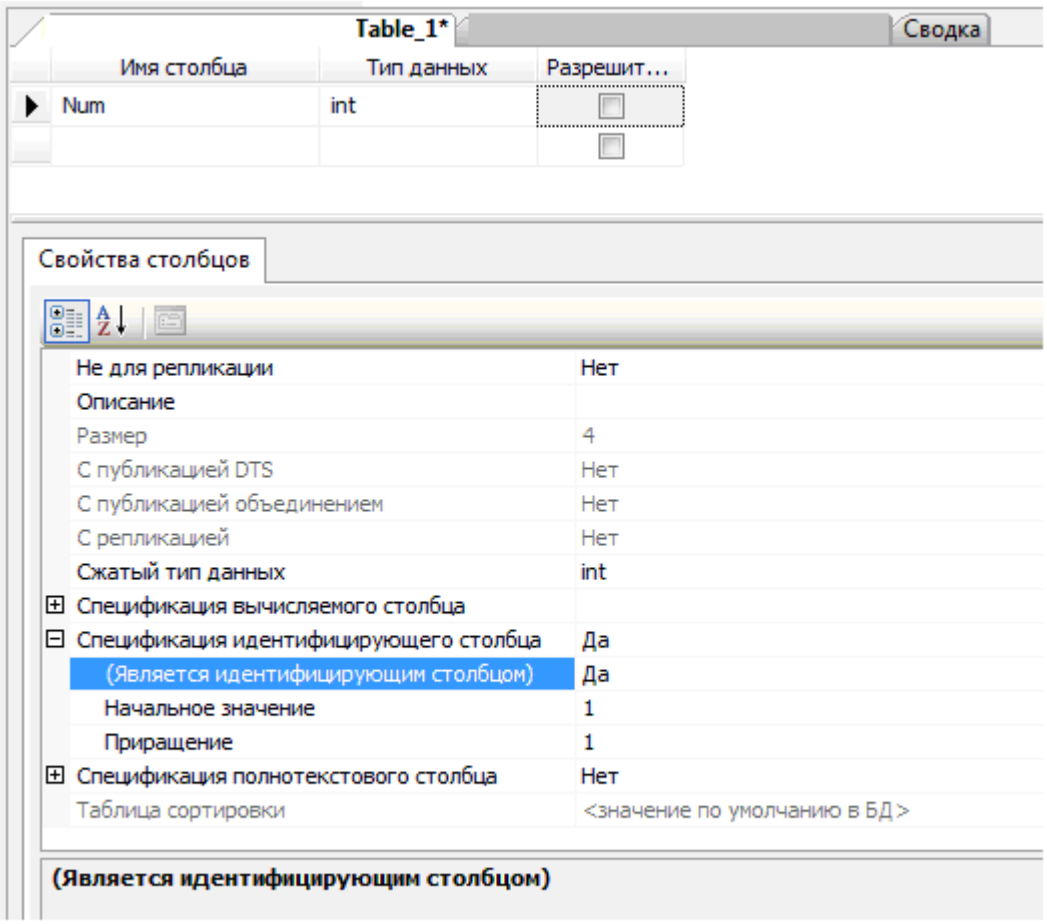


Рисунок 10. Определение свойств идентифицирующего столбца

4. Создание таблиц базы данных с помощью SQL-запроса

Создание таблиц в графическом режиме, безусловно, удобно, однако не универсально. При использовании других средств разработки баз данных (например, IBM DB2) придется привыкать к новым приемам работы. Использование конструкций языка SQL позволяет работать с базами данных, исходя из единого подхода, в любой среде управления базами данных.

Выберите на панели инструментов «Создать запрос»:

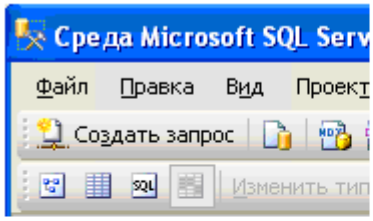


Рисунок 11.

Создадим новую базу данных запросом. Напишем

```
CREATE DATABASE mbs21_query
```

и нажмем F5. В обозревателе объектов должна появиться новая база (если сразу не появилась, то надо выделить мышью раздел «Базы данных» и в контекстном меню выбрать «Обновить»).

Теперь создадим таблицу Speciality. Упрощенный синтаксис создания таблиц следующий:

```
CREATE TABLE <имя таблицы> (
    <имя столбца 1> <тип данных> [NOT NULL] [DEFAULT <значение по умолчанию>],
    <имя столбца 2> <тип данных> [NOT NULL] [DEFAULT <значение по умолчанию>],
    ...
)
```

Введем новый запрос:

```
/* создание таблицы Специальность*/
USE mbs21_query                                  -- определяем базу данных, в которую входит таблица
CREATE  TABLE  Speciality(
    Num  INT IDENTITY(1,1) PRIMARY KEY NOT NULL, -- первичный ключ
    NameSpec VARCHAR(60)                         -- название специальности
)
```

В обозревателе объектов видим, что таблица действительно создана. Файл с SQL-запросом сохраняем в своей папке (в конце работы необходимо показать запросы, которые были выполнены, преподавателю). Слово IDENTITY(1,1) добавлено, чтобы поле первичного ключа Num автоматически нумеровалось начиная с единицы (фактически, эта конструкция определяет спецификацию идентифицирующего столбца).

Таким же образом необходимо создать остальные таблицы. Рассмотрим таблицу Course.

Таблица **Course** (курс)

Имя поля (столбца)	Содержание	Тип данных	Возможность содержать NULL
Num	Первичный ключ	int	нет
Name	Название специальности	varchar(60)	нет
YearEntry	Год поступления	int	нет
YearFinal	Год выпуска	int	да
Speciality	Специальность (внешний ключ ссылается на первичный ключ таблицы Speciality)	int	нет

Эта таблица содержит поле *Speciality*, которое ссылается на первичный ключ таблицы Speciality. Чтобы создать такую таблицу, необходимо выполнить запрос:

```
/* создание таблицы Курс */
USE mbs21_query                              -- определяем базу данных, в которую входит таблица
CREATE  TABLE  Course(
    Num  INT IDENTITY(1,1) PRIMARY KEY NOT NULL, -- первичный ключ
    YearEntry INT NOT NULL,                  -- год поступления
    YearFinal INT,                           -- год окончания
    Speciality INT FOREIGN KEY REFERENCES Speciality(Num) -- специальность,
    -- ссылка по внешнему ключу на поле Num таблицы Speciality
)
```

Примечание. Ссылку можно создать только на существующую таблицу. Задать ссылку по внешнему ключу можно и после создания таблицы

Задание 2:

Создайте все остальные таблицы, указанные в Приложении, используя SQL – запросы.

Таблица **Speciality** (специальность)

Имя поля (столбца)	Содержание	Тип данных	Возможность содержать NULL
Num	Первичный ключ	int	Нет
Name	Название	varchar(60)	Нет

Таблица **Course** (курс)

Имя поля (столбца)	Содержание	Тип данных	Возможность содержать NULL
Num	Первичный ключ	int	нет
Name	Название специальности	varchar(60)	нет
YearEntry	Год поступления	int	нет
YearFinal	Год выпуска	int	да
Speciality	Специальность (внешний ключ ссылается на первичный ключ таблицы Speciality)	int	нет

Таблица **Group** (группа)

Имя поля (столбца)	Содержание	Тип данных	Возможность содержать NULL
Num	Первичный ключ	int	нет
Name	Название специальности	varchar(60)	нет
Course	Курс (внешний ключ ссылается на первичный ключ таблицы Course)	int	нет

Таблица **Discipline** (дисциплина)

Имя поля (столбца)	Содержание	Тип данных	Возможность содержать NULL
Num	Первичный ключ	int	Нет
Name	Название (возможные значения: программирование, алгебра...)	varchar(60)	Нет

Таблица **Account** (тип отчетности)

Имя поля (столбца)	Содержание	Тип данных	Возможность содержать NULL
Num	Первичный ключ	int	Нет
Name	Название (возможные значения: экзамен, зачет, дифференцированный зачет...)	varchar(30)	Нет

Таблица **Mark** (отметка)

Имя поля (столбца)	Содержание	Тип данных	Возможность содержать NULL
Num	Первичный ключ	int	Нет
Name	Название (возможные значения: зачтено, не зачтено, отлично, хорошо...)	varchar(30)	Нет
Value	Значение (возможные значения: 0, 1, ..., 5)	int	Нет

Таблица **Status** (академический статус студента)

Имя поля (столбца)	Содержание	Тип данных	Возможность содержать NULL
Num	Первичный ключ	int	Нет
Name	Название (возможные значения: обучается, отчислен, в академическом отпуске, в отпуске по уходу за ребенком)	varchar(60)	Нет

Таблица **Position** (должность)

Имя поля (столбца)	Содержание	Тип данных	Возможность содержать NULL
Num	Первичный ключ	int	Нет
Name	Название (возможные значения: ассистент, старший преподаватель, доцент...)	varchar(60)	Нет

Таблица **People** (люди)

Имя поля (столбца)	Содержание	Тип данных	Возможность содержать NULL
Num	Первичный ключ	int	Нет
LastName	Фамилия	varchar(30)	Нет
FirstName	Имя	varchar(30)	Нет
MiddleName	Отчество	varchar(30)	Да
Male	Пол	char(1)	Нет
BrthDate	День рождения	datetime	Да
Addr	Адрес	varchar(100)	Да

Таблица **Student** (студент)

Имя поля (столбца)	Содержание	Тип данных	Возможность содержать NULL
Num	Первичный ключ	int	Нет
People	Человек (внешний ключ ссылается на первичный ключ таблицы People)	int	Нет
Group	Группа (внешний ключ ссылается на первичный ключ таблицы Group)	int	Нет
StudNum	Номер студенческого билета	varchar(30)	Нет
Status	Академический статус студента (внешний ключ ссылается на первичный ключ таблицы Status)	int	Нет

Таблица **Teacher** (преподаватель)

Имя поля (столбца)	Содержание	Тип данных	Возможность содержать NULL
Num	Первичный ключ (табельный номер сотрудника)	int	Нет
People	Человек (внешний ключ ссылается на первичный ключ таблицы People)	int	Нет
Position	Должность (внешний ключ ссылается на первичный ключ таблицы Position)	int	Нет

Таблица **SemesterResults** (результаты сессии)

Имя поля (столбца)	Содержание	Тип данных	Возможность содержать NULL
Num	Первичный ключ	int	Нет
Student	Студент (внешний ключ ссылается на первичный ключ таблицы Student)	int	Нет
Semester	Порядковый номер семестра	int	Нет
Account	Тип отчетности (внешний ключ ссылается на первичный ключ таблицы Account)	int	Нет
Discipline	Дисциплина (внешний ключ ссылается на первичный ключ таблицы Discipline)	int	Нет
Teacher	Преподаватель (внешний ключ ссылается на первичный ключ таблицы Teacher)	int	Нет
Mark	Отметка (внешний ключ ссылается на первичный ключ таблицы Mark)	int	Нет
Date	Дата сдачи отчетности	DateTime	Нет

Задание 3:

Ответьте на вопросы:

Каким образом можно получить доступ к MS SQL Server 2005?

С помощью каких средств можно создать таблицу для MS SQL Server 2005?

Что такое первичный ключ?

Каким образом можно создать автоматическую нумерацию строк таблицы?

Что означают Not Null?

Итог работы: отчет, защита работы.

Практическая работа № 5

Цель: освоение технологии создания однотабличной БД.

Задание 1:

1. **Создать** структуру таблицы базы данных **«Ученик»**, содержащую следующие поля: **фамилия, имя, школа, класс, дата рождения, вес**. **Типы** и **форматы** полей определить самостоятельно.
2. Определить **первичный ключ** таблицы.
3. В режиме **таблицы ввести** в базу данных **пять** любых записей.
4. **Добавить** в структуру таблицы после поля **«дата рождения»** поле **«рост»**.
5. **Заполнить** поле **«рост»**.
6. С помощью **мастера форм** создать **форму** для редактирования таблицы.
7. В режиме **формы ввести** в таблицу **пять** любых записей.
8. **Удалить** из структуры таблицы поле «вес».
9. **Удалить** из таблицы **вторую** и **пятую** записи.

Итог работы: отчет, защита работы.

Практическая работа №6

Цель: обобщить и систематизировать знания и умения по теме "Базы данных"

Задание 1:

1. Создайте базу данных **«Воскресная школа программирования»**, содержащую три таблицы: **Группы, Списки, Личные данные**.

1. Создайте структуру таблицы **Группы**. Поле «Учебная группа» сделайте ключевым. Сохраните таблицу, не заполняя её.
2. Создайте структуру таблицу **Списки**. Ключевое поле Код студента (тип числовой). Сохраните таблицу, не заполняя её.
3. Создайте структуру таблицу **Личные данные** по образцу. Сохраните таблицу, не заполняя её.
4. Установите связи между таблицами
5. Заполните таблицу Группы
6. Для заполнения двух других таблиц создайте соответствующие формы.

Таблица Группы

Учебная группа	Преподаватель
101	Верзаков С.А.
102	Белоусов А.И.
103	Масалова В.А.
104	Новикова Е.В.
105	Зачёсова Т.П.

Таблица Списки

Код студента	Фамилия	Имя	Отчество	Год рождения	Школа	Класс	Учебная группа
1	Иванова	Анна	Ивановна	2001	1	9	101
2	Баранова	Ирина	Алексеевна	2000	3	10	102
3	Корнилова	Ольга	Владимировна	2001	5	9	103
4	Воробьёв	Алексей	Петрович	2000	1	10	101
5	Воробьёв	Алексей	Иванович	2001	3	9	104
6	Воробьёв	Олег	Григорьевич	2002	5	8	105
7	Скоркин	Александр	Евгеньевич	1999	1	11	101
8	Володина	Анна	Алексеевна	2001	3	9	102
9	Новосёлов	Алексей	Антонович	2000	5	10	103
10	Александрова	Елена	Алексеевна	2001	1	9	101

Таблица Личные данные

Код студента	Адрес	Номер телефона
1	Центральная 11-15	5-17-22
2	Солнечная 8-117	5-18-22
3	Сиреневый 7-16	5-19-22
4	Центральная 14-81	5-20-22
5	Сиреневый 7-16	5-21-22
6	Солнечная 2-121	5-22-22
7	Школьная 5-34	5-23-22
8	Центральная 30-7	5-24-22
9	Сиреневый 7-16	5-25-22
10	Солнечная 6-34	5-26-22

2. С помощью запросов выполнить следующие задания:

1. Вывести учащихся, год рождения которых 2001.
2. Вывести список учащихся школы №5, 10 класса.
3. Вывести учащихся, родившихся позднее 2000 года.
4. Вывести список учащихся с номерами телефонов (показать поля «Фамилия», «Имя», «Отчество», «Номер телефона»).
5. Вывести информацию о девочках, имя которых Анна (показать поля «Фамилия», «Имя», «Отчество», «Адрес», «Школа», «Класс»).

Итог работы: отчет, защита работы.

Практическая работа №7

Цель: обобщить и систематизировать знания и умения по теме "Базы данных"

Задание 1:

Имеются следующие данные по регистрации заказов на товары:

Код товара	Тип товара	Производитель	Модель	Дата заказа	Цена	Количество	№ заказа
851365125874	Весы	Marta	M-150	12.12.12	980	20	948
680587369521	Кофемолка	Bork	B-56	19.12.12	650	12	1001
460123654987	Комбайн	LG	L1200	12.12.12	4320	10	1002
680587369521	Кофемолка	Bork	B-56	16.12.12	650	5	1011
461357951852	Нагреватель	Омск	H-1500	13.12.12	680	5	1003
481258741951	Кипятильник	Казань	ЭК-11	12.11.12	62	25	900
851365125874	Весы	Marta	M-150	16.12.12	980	10	1007
851365125874	Весы	Marta	M-150	2.12.12	980	8	1008
680587369521	Кофемолка	Bork	B-56	14.12.12	650	10	1005
460123654987	Комбайн	LG	L1200	16.12.12	4320	10	1006
569258147456	Утюг	Braun	DT-16	12.12.12	3240	5	948
481258741951	Кипятильник	Казань	ЭК-11	17.12.12	62	23	1012
461357951852	Нагреватель	Омск	H-1500	18.12.12	680	19	1013
569258147458	Утюг	Bork	BT-16	12.11.12	1320	15	904
460123654988	Комбайн	LG	G500	19.12.12	3240	5	1001
654321987597	Утюг	LG	L569	14.12.12	3240	3	1005
481258741951	Кипятильник	Казань	ЭК-11	12.12.12	62	25	1002
680587369527	Кофемолка	LG	G196	2.12.12	1650	7	1008

1. В Access создать базу данных «Регистрация заказов», состоящую из двух таблиц («Товары» и «Заказы»). Состав полей каждой таблицы, типы данных, ключевые поля назначить самостоятельно. Учесть, что одинаковые товары имеют одинаковые коды, не должно быть дублирующих записей, один и тот же товар за день может быть заказан более одного раза (по разным заказам). По одному заказу – только разные товары.
2. Заполнить созданную базу имеющимися данными.
   Отсортировать таблицу «Товары» по стоимости - в тетрадь зафиксировать самый дешёвый и самый дорогой товар.
3. Создать и сохранить следующие запросы:
   a. Все данные из обеих таблиц (как в исходной таблице в начале задания).
   b. Данные о заказах товаров с количеством больше 8 единиц в заказе, с информацией о цене и производителе товара.
   c. Данные о заказах товаров на 12 декабря.
   d. Данные о стоимости заказов по каждому товару (вычисляемое поле - произведение цены на количество) с указанием типа, модели и производителя товара, даты заказа.
   Отсортировать таблицу «Заказы» по номеру в тетрадь зафиксировать самый дешёвый и самый дорогой заказ.
   e. Данные о том, когда и сколько заказывали товар фирмы LG, а также их стоимость и информацию о модели.
4. Создать форму для ввода, просмотра и редактирования данных таблиц.

При создании формы учесть: в таблице «Товары» - поле для фото товара (изображение необходимо отыскать в интернете), каждой форме присвоить заголовок,

в формах применить размер шрифтов более 10 пт.

Итог работы: отчет, защита работы.

Практическая работа № 8

Цель: Изучение объектов Visual Basic for Application на примере линейной программы.

Задание 1: ознакомьтесь с материалом

Форма (**UserForm**) представляет собой диалоговое окно, в котором можно размещать различные элементы управления. В приложении может быть как одна, так и несколько форм. Новая форма добавляется в проект выбором команды **Insert(Вставка) → UserForm**.

В **VBA** имеется обширный набор встроенных элементов управления.

Элементы управления являются объектами. Как любые объекты, они обладают свойствами, методами и событиями.

1. Элементы управления создаются при помощи **Панели элементов,** которая отображается на экране либо выбором команды **Вид (View) →** Панель элементов **(Toolbox),** либо нажатием кнопки  панели инструментов **Standard.**

На этой панели представлены кнопки, позволяющие конструировать элементы управления. Для создания элементов управления служат все кнопки панели инструментов, за исключением кнопки **Выбор объекта**  **.**

Таблица 1. Основные элементы управления и соответствующих кнопок панели элементов.

Элемент управления	Название	Кнопка	Элемент управления	Название	Кнопка
Поле	TextBox		Переключатель	OptionButton	
Надпись	Label		Флажок	CheckBox	
Кнопка	CommandButton		Рисунок	Image	
Список	ListBox		Поле со списком	ComboBox	

Таблица 2. Основные общие свойства элементов управления.

Свойство	**Описание**
Caption	Надпись, отображаемая при элементе управления.
Visible	Допустимые значения: True (элемент управления отображается во время выполнения программы) и False (в противном случае).
Enabled	Допустимые значения: True (пользователь вручную может управлять элементом управления) и False (в противном случае).
Height, Width	Устанавливают геометрические размеры объекта (высоту и ширину).
Left и Top	Устанавливают координаты верхнего левого угла элемента управления, определяющие его местоположение в форме
BackColor, ForeColor	Устанавливают цвет заднего и переднего плана элемента управления, также его границы.
BackStyle	Устанавливает тип заднего фона.

Свойство	Описание
Picture (создание картинки)	Внедряет картинку на элемент управления. Например, на поверхности кнопки картинка отображается с помощью следующей инструкции: CommandButton1. Picture =LoadPicture("c:\my doc\Круг.bmp").
Picture (удаление картинки)	После того как картинка создана на элементе управления, иногда возникает необходимость ее удалить. Это легко достигается присвоением свойству Picture значения LoadPicture("") CommandButton1.Picture = LoadPicture("")

После размещения элементов управления на форме необходимо связать объект на форме с кодом.

Задание 2: Создать пользовательскую форму на примере простейшего диалогового окна.

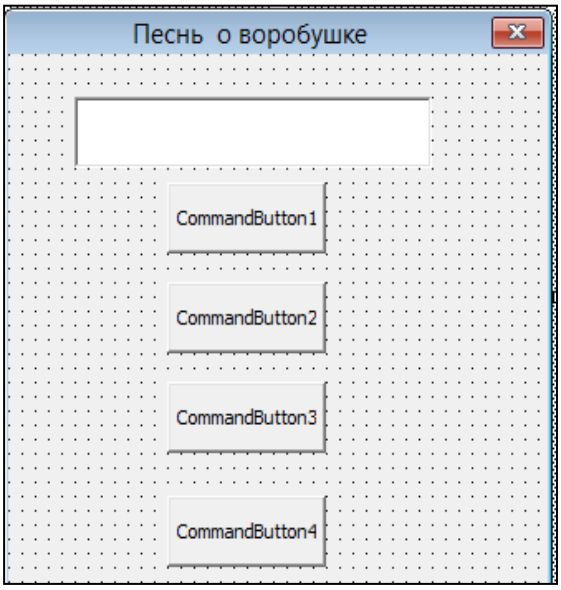


Рисунок 13

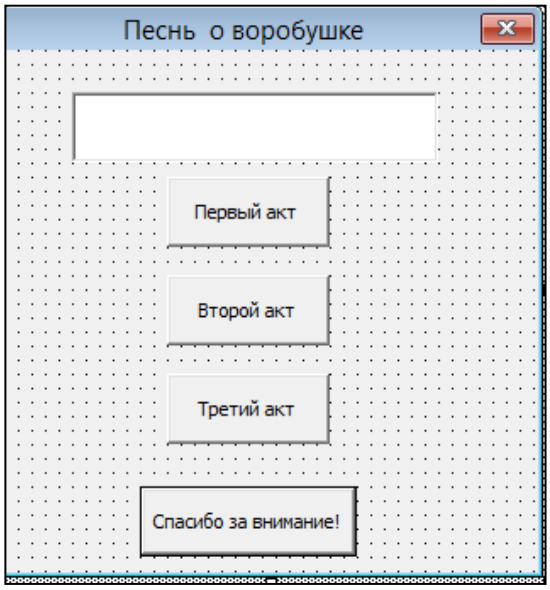


Рисунок 14

1. Откройте новую рабочую книгу.
2. Нажмите комбинацию клавиш **Alt +F11** для открытия редактора Visual Basic.
3. Выполнить команду ***Insert / UserForm***. В редакторе появятся *окно с пользовательской формой* и *панель с элементами управления*.
4. Используя панель инструментов, заполните пользовательскую форму элементами управления, создав требуемое диалоговое окно приложения.

Для этого на форму поместите элементы управления: четыре кнопки(CommandButton); одно текстовое поле (TextBox).

5. Используя диалоговое окно ***Свойства*** (***Properties***), отображаемое нажатием кнопки  , свойство *Caption* пользовательской формы определить равным *Песнь о воробушке и бабочке*, а кнопок - равными *Первый акт, Второй акт* и *Третий акт, Спасибо за внимание*.
6. Написать процедуру обработки события - нажатие кнопки.

Для написания процедуры обработки события - нажатия кнопки П*ервый акт,* дважды ее щелкнуть. Откроется редактор кода на листе модуля ***UserForm1***.

Введите текст процедуры:

Private Sub CommandButton1_Click()
TextBox1.Text = "Воробышек за бабочкой скок-скок-скок"
End Sub

Для написания процедуры обработки событий - нажатий кнопок *Второй акт* и *Третий акт* ввести аналогичные процедуры:

Private Sub CommandButton2_Click()
TextBox1.Text = "Воробышек за бабочкой прыг-прыг-прыг"

End Sub
Private Sub CommandButton3_Click()
TextBox1.Text = "Воробышек бабочку ням-ням-ням"
End Sub
Private Sub CommandButton4_Click()
End
End Sub

Процесс создания диалогового окна и процедур, связанных с ним, завершен. Для выполнения созданной программы нажать кнопку  или выполнить команды меню ***Run / Run Sub UserForm*** или нажать функциональную **клавишу *F5***.

Итог работы: отчет, защита работы.

Практическая работа № 9

Цель: Овладение основными приемами создания Web-страниц на языке HTML

Задание 1:

Создать веб- сайт, рассказывающий о башнях Московского Кремля.

1.Создать специальную папку , дать ей имя Kremlin.

2.. В ней сохранить веб-страницу и графические файлы (рисунки) Кремля.

Создание Web-сайта «Московский кремль»

1.Запустить текстовой редактор Блокнот(туда скопировать теги из файла «основные теги».

2.Дать странице название «Башни Московского Кремля»

3.Между тегами <BODI> </BODI> вставить скопированный текст из файла «История Кремля».

4.Сохранить под именем index.htm в папке Kremlin.

5.Просмотреть полученную страницу в браузере (открыть с помощью Internet Explorer)

Итог работы: отчет, защита работы.

Практическая работа № 10

Цель: отработать навыки работы с базами данных при разработке веб приложений

Задание 1:

Задание 1. Установка и настройка IIS:

1) добейтесь отображения на мониторе компьютера окна «Компоненты Windows»;

2) позиционируйте дерево настроек в окне «Компоненты Windows» таким образом, чтобы были видны первоначальные настройки пункта «службы IIS»;

3) получите скриншот окна «Компоненты Windows» с первоначальными настройками служб IIS и включите его в отчет по контрольной работе;

4) отметьте все необходимые компоненты IIS и выполните их установку;

5) перезагрузите операционную систему.

Задание 2. Проверка работоспособности IIS: 1) с помощью командной строки запустите приложение IIS Manager и убедитесь, что на мониторе отобразилось окно, аналогичное представленному на рис.1.4;

2) получите скриншот окна «Диспетчер служб IIS» и включите его в отчет по контрольной работе;

3) запустите браузер, в адресной строке введите http://localhost; убедитесь, что в браузере отобразилось окно, аналогичное представленному на рис. 1.5;

4) получите скриншот окна браузера со стартовой страницей IIS 7 и включите его в отчет по контрольной работе.

Задание 3. Контрольные вопросы:

1) поясните термины: «сервер», «клиент», «браузер», «HTTP», «HTML», «web-приложение»;
2) поясните термины: «IIS», «IIS Manager» «ASP.NET», «Web Forms», «MVC»; 3
) поясните термины: «SOAP», «web-сервис», «WCF».
Итог работы: отчет, защита работы.

Практическая работа № 11

Цель: Овладение основными приемами разработки веб приложения
Задание 1:
Задание 1. Создание простейшего приложения Web Forms ASP.NET:
1) запустите Visual Studio и создайте шаблонное приложение Web Forms ASP.NET;
2) сделайте скриншот окна «Обозреватель решений» проекта и поместите его в отчет по контрольной работе;
3) откройте в проекте файл Default.aspx и замените текст «Добро пожаловать в ASP.NET!» на текст, содержащий ваше имя, фамилию и отчество;
4) запустите созданное web-приложение на выполнение в отладочном режиме;
5) сделайте скриншот окна браузера, отображающего стартовую страницу приложения, и поместите его в отчет по контрольной работе;
6) откройте окно отладочного сервера ASP.NET Development Server, сделайте скриншот и поместите его в отчет по контрольной работе.
Задание 2. Публикация приложения на IIS:
1) запустите приложение IIS Manager и с его помощью создайте сайт;
2) опубликуйте в этот сайт созданное в задании 1 приложение методом простого копирования файлов;
3) убедитесь в работоспособности сайта;
4) с помощью приложения IIS Manager создайте еще один сайт;
5) опубликуйте в последний созданный сайт созданное в задании 1 приложение методом отличным от простого копирования файлов;
6) сделайте скриншот окна приложения IIS Manager, который бы демонстрировал наличие двух созданных в этом задании сайтов; поместите скриншот в отчет по контрольной работе.
Задание 3. Контрольные вопросы:
1) поясните термины: «шаблон приложения», «ASP.NET Development Server»;
2) поясните термины: «сайт», «публикация приложения»;
3) что такое пакет развертывания приложение, как он может быть сформирован и как применен?
4) перечислите параметры, которые необходимо указать при создании сайта с помощью приложения IIS Manager;
5) перечислите известные вам способы публикации web-приложений.
Итог работы: отчет, защита работы.

Практическая работа № 12

Цель: Овладение основными приемами разработки веб приложения
Задание 1:
Задание1. Исследование структуры приложения и модели со- бытий Web Forms ASP.NET:
1) создайте приложение Web Forms ASP.NET с помощью Visual Studio;
2) измените страницу Default.aspx так, чтобы она была такой же, как на рис. 3.4; Рис. 3.5. Файл Default.aspx

```
<%@ Page Title="Домашняя страница" Language="C#" MasterPageFile="~/Site.master" AutoEventWireup="true"
    CodeBehind="Default.aspx.cs" Inherits="WebApplication5._Default" %>

<asp:Content ID="HeaderContent" runat="server" ContentPlaceHolderID="HeadContent"></asp:Content>
<asp:Content ID="BodyContent" runat="server" ContentPlaceHolderID="MainContent">
  <h2> Практическая работа № 3 </h2>
  <p>
    <br /><br />
    <asp:Button ID="Button1" runat="server" Text="Button1"
                onclick="Button1_Click" />
    <asp:Button ID="Button2" runat="server" Text="Button2" style="margin-left: 34px"
                onclick="Button2_Click" />
    <br /><br />
    <asp:Label ID="Label1" runat="server" Text="Label"></asp:Label>
  </p>
 </asp:Content>
```

Рис. 3.5. Файл Default.aspx

3) откорректируйте программный код в файле Default.aspx.cs, чтобы он стал таким же, как на рис. 3.6; отличаться может имя пространства имен (указывается в операторе namespace), которое в шаблонном приложении совпадает с именем приложения, задаваемым при создании; 24 Рис. 3.6. Файл Default.aspx.cs

```
namespace WebApplication5
{
        public partial class _Default : System.Web.UI.Page
        {
            private int k = 0;
            protected void Page_Init(object sender, EventArgs e)
            { this.Label1.Text += "Init["+ (++k).ToString()+"]-"; }

            protected void Page_Load(object sender, EventArgs e)
            { this.Label1.Text += "Load[" + (++k).ToString() + "]-"; }

            protected void Button1_Click(object sender, EventArgs e)
            { this.Label1.Text += "Click1[" + (++k).ToString() + "]-"; }

            protected void Button2_Click(object sender, EventArgs e)
            { this.Label1.Text += "Click2[" + (++k).ToString() + "]-";}

            protected void Page_PreRender(object sender, EventArgs e)
            { this.Label1.Text += "PreRender[" + (++k).ToString() + "]-"; }

            protected void Page_Unload(object sender, EventArgs e)
            { this.Label1.Text += "Unload[" + (++k).ToString() + "]-"; }
         }
 }
```

Рис. 3.6. Файл Default.aspx.cs

4) запустите приложение на выполнение в отладочном режиме; убедитесь, что в окне браузера отображается форма такая же, как на рис. 3.7; Рис. 3.7. Отображение формы Default.aspx в окне браузера

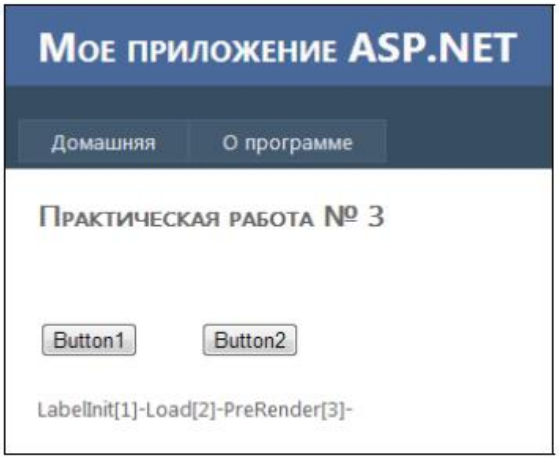


Рис. 3.7. Отображение формы Default.aspx в окне браузера

1) сделайте скриншот окна браузера, отображающего страницу Default.aspx приложения и поместите его в отчет по контрольной работе;
2) с помощью мыши нажмите клавишу Button1; обратите внимание на изменение текста строки, расположенной под кнопками;
3) сделайте скриншот окна браузера, отображающего страницу Default.aspx приложения и поместите его в отчет по контрольной работе;
4) с помощью мыши нажмите клавишу Button2; обратите внимание на изменение текста строки, расположенной под кнопками;
5) сделайте скриншот окна браузера, отображающего страницу Default.aspx приложения и поместите его в отчет по контрольной работе;
6) опубликуйте приложение на IIS и убедитесь в его работоспособности. Задание 2. Контрольные вопросы:
1) перечислите все типы файлов, которые могут входить в приложение Web Forms ASP.NET, поясните их назначение;
2) перечислите все наименования стандартных папок приложения Web Forms ASP.NET, поясните их назначение;
3) перечислите в порядке их появления все события, которые могут быть обработаны методами класса Global (файл Global.asax.cs);
4) перечислите все события страницы в порядке их появления; 5) объясните принцип формирования строки на форме приложения, разработанного в задании 1
Итог работы: отчет, защита работы.

Практическая работа № 13

Цель: отработать навыки по созданию и выполнению команд к источникам данных
Задание 1:
Задание 1. Разработка HTTP-обработчика ASP.NET:
1) разработайте HTTP-обработчик, реагирующий на GET-запрос страниц с расширением bstu; HTTP-обработчик должен принимать значения двух параметров: faculty и course; в результате выполнения в окне браузера должна отобразиться следующая строка: GET: bstu, faculty = val_parm1, course = val_parm2, где val_parm1 и val_parm2 – значения первого и второго параметров.
2) разработайте HTTP-обработчик, реагирующий на POST-запрос страниц с расширением bstu; HTTP-обработчик должен принимать значения двух параметров: faculty и course; в результате выполнения в окне браузера должна отобразиться следующая строка: POST: bstu, faculty = val_parm1, course= val_parm2, где val_parm1 и val_parm1 – значения первого и второго параметров.
3) опубликуйте HTTP-обработчики на IIS;
4) с помощью приложения IIS Manager отобразите окно «сопоставление обработчиков», найдите разработанные в предыдущих пунктах задания обработчики, сформируйте скриншот и поместите его в отчет по контрольной работе;
5) с помощью браузера выполните GET-запрос к HTTP-обработчику и убедитесь в его работоспособности;
6) разработайте web-форму, формирующую POST-запрос к HTTPобработчику; опубликуйте эту web-форму IIS; выполните с помощью браузера запрос к форме, сформируйте POST-запрос и убедитесь в работоспособности обработчика; текст разработанной web-формы поместите в отчет по контрольной работе;

7) поместите исходные коды HTTP-обработчиков в отчет по контрольной работе.
Задание 2. Разработка клиента, взаимодействующего с HTTP-обработчиком ASP.NET:
1) разработайте консольное приложение на языке C#, формирующее GET и POST-запросы к HTTP-обработчикам, разработанным в задании 1; приложение должно отображать ответ, оправленный обработчиками в окне консоли;
2) выполните приложение, разработанное в предыдущем пункте задания; убедитесь в работоспособности приложения и HTTPобработчиков; сформируйте скриншот, отражающий результат выполнения приложения; поместите скриншот и исходный код приложения в отчет по контрольной работе.
Задание 3. Контрольные вопросы:
1) поясните понятие «HTTP-обработчик ASP.NET»;
2) каким образом можно отобразить перечень HTTP-обработчиков, действующих в рамках сайта?
3) поясните структуру HTTP-обработчика (интерфейс, назначение методов);
4) где располагается информация связывающая класс HTTPобработчика, метод HTTP-запроса и расширение страниц;
5) в каких случаях целесообразно применять HTTP-обработчики?
Итог работы: отчет, защита работы.

Практическая работа № 14.

Цель: отработать навыки по созданию таблиц
Задание 1:

В таблице могут использоваться следующие способы выравнивания:

По горизонтали: - слева; **По вертикали:** - вверху.	**По горизонтали:** - слева; **По вертикали:** - по центру. (этот способ используется по умолчанию)	**По горизонтали:** - слева; **По вертикали:** - внизу.
По горизонтали: - по центру; **По вертикали:** - вверху.	**По горизонтали:** - по центру; **По вертикали:** - по центру.	**По горизонтали:** - по центру; **По вертикали:** - внизу.
По горизонтали: - справа; **По вертикали:** - вверху.	**По горизонтали:** - справа; **По вертикали:** - по центру.	**По горизонтали:** - справа; **По вертикали:** - внизу.

Сохраните страницу в личной папке в файл `table1.html`

Форматирование заднего фона таблиц, рядов и ячеек:

1	2	3
4	5	6
7	8	9

Итог работы: отчет, защита работы.

Практическая работа № 15

Цель: Научиться управлять процессом передачи данных от источника данных к DataSet.

Задание 1:

Создайте базу данных EmployeeInfo средствами Visual Studio.Net в SQL Server. Модель базы данных представлена на рис. 3.2.

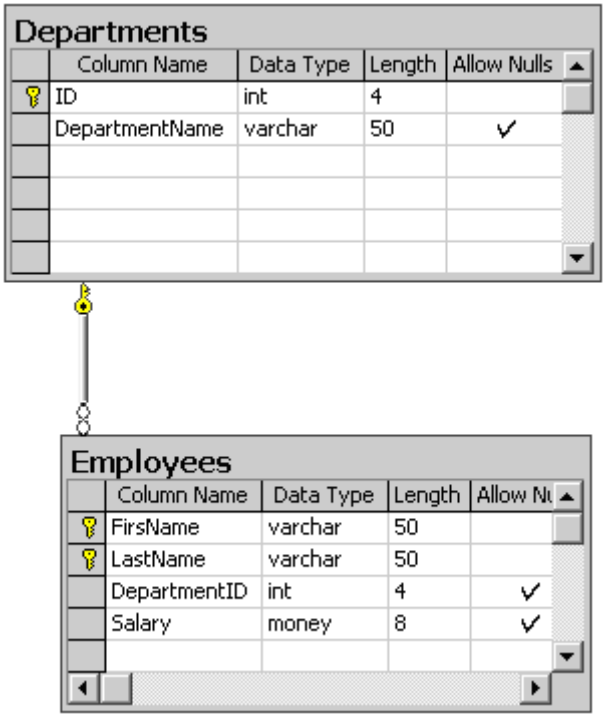


Рис. 3.2. Модель базы данных EmployeeInfo

2. Установите связи и заполните таблицы необходимой информацией.
3. Откройте форму "Data Set" проекта к практикуму по теме "Отсоединенная модель программирования".
4. Создайте новую кнопку DataAdapterFill. Вставьте в процедуру обработки щелчка по командной кнопке код, приведенный ниже. Не забудьте импортировать необходимые пространства имен:

```
Imports System.Data, System.Data.SqlClient
Private Sub Button7_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
Button7.Click
ReadData()
End Sub
```

Код процедуры общего назначения для чтения данных из источника данных ReadData():

```
Private Sub ReadData()
Dim rows As Integer
Dim daDep As SqlDataAdapter = New
SqlDataAdapter("select * from Departments",
"server=Persist Security
Info=False; Integrated Security=SSPI;
database=EmployeeInfo; server=abrzh")
dsE = New DataSet
rows = daDep.Fill(dsE, "Departments")
DisplayDataset(dsE)
End Sub
```

После создания объекта daDep с двумя аргументами, которые содержат команду Select и строку подключения, вызывается метод Fill для наполнения данными таблицы Departments набора данных dsE. Метод Fill также возвращает количество записей, которые включены (или обновлены) в набор данных dsE. При выполнении метода Fill провайдером данных неявно выполняются следующие действия:

- Для объекта SelectCommand открывается подключение к источнику данных, если оно еще не открыто.

- Выполняется команда, указанная в свойстве CommandText объекта SelectCommand (вместе с параметрами, если таковые имеются).

- Создается объект DataReader для возвращения имен полей и типов, использованных для создания нового объекта DataTable в указанном наборе данных DataSet, если этого объекта еще не существует.

- Объект DataReader используется для извлечения данных и вставки их в таблицу.

- Объект DataReader закрывается.

- Подключение к источнику данных закрывается, если оно было открыто объектом DataReader, в противном случае оно остается открытым.

При выполнении одной команды по отношению к источнику данных (наш с вами случай) обычно проще и эффективнее создавать объекты Command и Connection неявно при создании объекта DataAdapter (именно так мы с вами и поступили). Однако при выполнении нескольких команд по отношению к одному источнику данных эффективнее создавать объект Connection явно и затем присвоить его объекту DataAdapter. Это позволяет поддерживать подключение постоянно открытым, без часто повторяющихся операций его открытия и закрытия, что снижает производительность. Эквивалентный код представлен ниже.

```
Private Sub ReadData1()
Dim rows As Integer
Dim daDep As New SqlDataAdapter
Dim cnn As New SqlConnection("server=Persist Security
Info=False; Integrated Security=SSPI;
database=EmployeeInfo; server=abrzh")
Dim cmdSelect As New SqlCommand("select * from
Departments")
```

```
dsE = New DataSet
cmdSelect.Connection = cnn
daDep.SelectCommand = cmdSelect
cnn.Open()
rows = daDep.Fill(dsE, "Departments")
DisplayDataset(dsE)
cnn.Close()
End Sub
```

Конечно, для эффективного использования явно созданных объектов cnn и cmdSelect желательно, чтобы количество операций с базой данных было достаточно большим.

Еще одно дополнение к написанному коду. Методу Fill мы передаем ссылку на набор данных dsE и имя таблицы Departments, в которую вставляются данные:

```
rows = daDep.Fill(dsE, "Departments")
```

Возможны другие варианты вызова метода Fill. Вместо имени таблицы можно было бы передать ссылку на объект DataTable или передать только ссылку на объект DataSet, а метод Fill в таком случае по умолчанию загрузит данные в объект DataTable по имени Table.

```
rows = daDep.Fill(dsE)
```

Для указания информативных имен таблиц Table, Table1 и так далее, используется коллекция объектов DataTableMapping и метод Add:

```
daDep.TableMappings.Add("Table", "Отделы")
daDep.TableMappings.Add("Table1", "Сотрудники")
```

Итог работы: отчет, защита работы.

Практическая работа № 16

Цель: Научиться управлять процессом передачи данных от источника данных к DataSet.

Задание 1:

Добавьте в процедуру ReadData() описание еще одно адаптера с другой командой Select для загрузки данных из таблицы Employees. Скомпонуйте проект, щелкните по кнопке DataAdapterFill, в поле со списком, как и прежде, будет отображена информация о содержании набора данных dsE, но теперь она извлекается из базы данных под управлением SQL Server, а не генерируется локально кодом приложения (рис. 3.3).

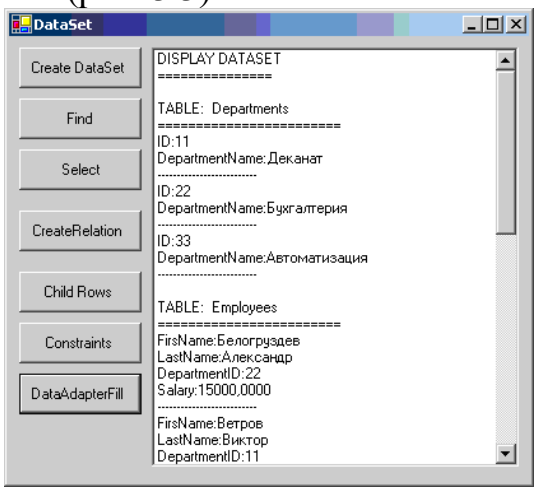


Рис. 3.3. Данные из таблиц базы данных, управляемой SQL Server (не из локальной версии, как ранее)

Для установления родительско-дочерних связей между записями в этих таблицах можно создать объект DataRelation, который служит отношением между ними.

```
dsE.Relations.Add("D_E",
dsE.Tables("Departments").Columns("ID"),
dsE.Tables("Employees").Columns("DepartmentID"))
```

Итог работы: отчет, защита работы.

Практическая работа № 17

Цель: отработать навыки фильтрации и поиска данных

Задание 1:

Исходным файлом является DVD.xls, содержащий информацию о DVD-коллекции видеофильмов (названии фильма на русском и английском языках, годе создания фильма, жанре, актерам и режиссере).

Технические требования:

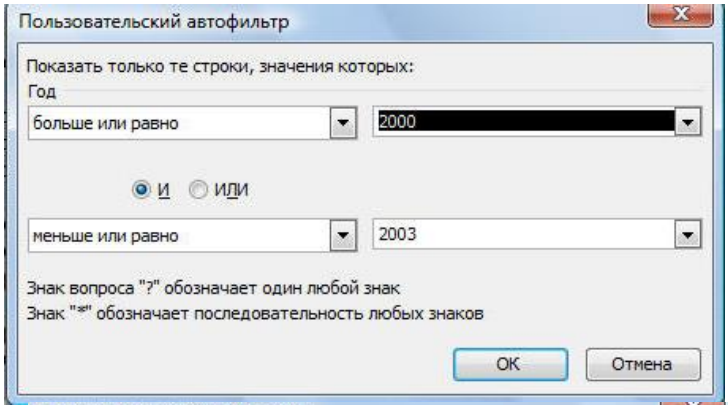
1. Каждое задание выполнять на новом листе рабочей книги с именем, указанным в поле «Имя нового листа».
2. Фильм будем считать отечественным, если название на английском языке является пустым.
3. Каждая запись списка должна соответствовать основным типам полей (числовое, текстовое, денежный…).
4. ДОБАВИТЬ запись «цена» и ввести значения.
5. Сортировать по возрастанию список 
произвести сортировку по указанному полю: Номер диска, Актеры, Режиссер.
6. Сформировать и реализовать следующие запросы:

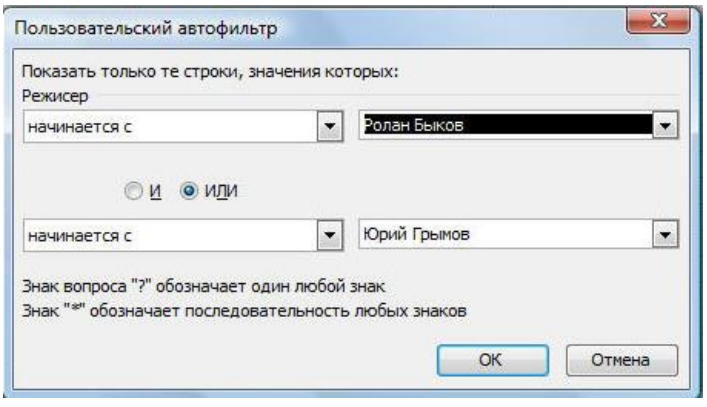
- простой (автофильтр) *Данные – фильтр – автофильтр – самостоятельно определить критерии.*

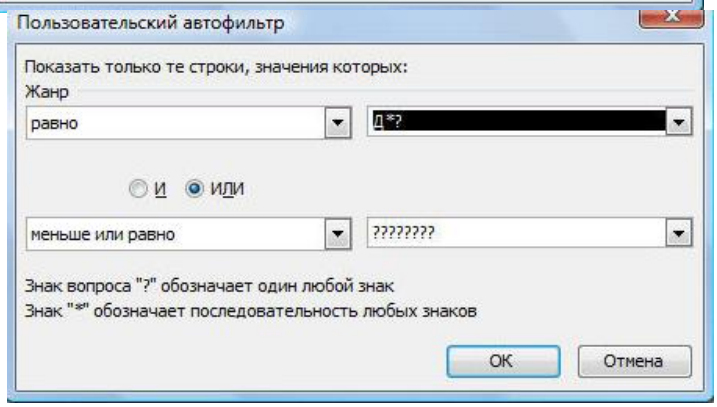
-расширенный (пользовательский)

При выборке “по году” допустимо 4 варианта: все, только указанного года, в период между двумя указанными годами, ранее или позднее указанного года.

Выборка “Режисеры”.







7. Фильтр с использованием шаблона, например Детектив-драма.
8. Найти максимальную и минимальную цену фильма.
9. Сформировать Промежуточные итоги *(Данные-итоги)- самостоятельно определить критерии.*

Итог работы: отчет, защита работы.

Практическая работа № 18

Цель: отработать навыки работы DataSet со строгим контролем.

Задание 1:

Создайте объект DataSet в режиме создания компонентов (панель Data) для модели Dogovor. Проверьте идентичность результатов создания таблиц, установления связей, заполнения таблиц для случая программного кодирования объекта DataSet и для случая визуального программирования. Воспользуйтесь разработанными процедурами отображения дочерних записей связанных таблиц.

Итог работы: отчет, защита работы.

Практическая работа № 19

Цель: отработать навыки обновления данных

Задание 1:

Примените процедуру просмотра имеющихся ограничений для таблиц объекта DataSet Dogovor. Убедитесь, что все уникальные поля таблиц и установленные вами связи между таблицами реализованы в объектах UniqueConstraint и ForeignKeyConstraint.

Итог работы: отчет, защита работы.

Практическая работа № 20

Цель: отработать навыки автоматизации приложения

Задание 1:

Ознакомьтесь с материалом

Разработка эскизного проекта

Эскизный проект возникает как результат анализа требований, предъявленных к программному продукту. В нем в общем виде формулируются указания по созданию программного продукта. Здесь ставится задача для каждого разработчика, описываются алгоритм решения задачи, способы взаимодействия создаваемого продукта с другими программами и устройствами ввода- вывода, выбираются структуры данных, определяются способы хранения данных на диске или в базе данных.

Эскизный проект не может быть слишком большим. Он должен быть обозримым, схематичным, четко показывающим основные этапы создания программного продукта. Обычно эскизный проект содержит не больше 5— 6 страниц текста. К нему прилагаются диаграммы, рисунки и чертежи, а также календарный план выполнения проекта.

После того как эскизный проект создан, он раздается всем участникам разработки для изучения и обсуждения. Каждый разработчик обдумывает свой участок проекта, вносит свои предложения и дополнения, конкретизирует план выполнения проекта.

Разработка технического проекта

После изучения эскизного проекта всеми заинтересованными лицами наступает время создания технического проекта. В его обсуждении принимает участие вся команда разработчиков под руководством менеджера проекта. Каждый разработчик вносит свои предложения по реализации и улучшению проекта, уточняет и детализирует относящиеся к нему положения проекта, согласует интерфейсы с другими разработчиками.

Технический проект будет рабочим документом на все время реализации проекта, поэтому он должен быть понятен и приемлем для всех программистов. В нем не должно быть недомолвок, двусмысленностей, не должно оставаться пробелов и недоговоренностей.

При разработке технического проекта окончательно определяется конфигурация технических средств, и вся дальнейшая работа ведется с учетом этой конфигурации. Уточняется операционная среда, в которой будет функционировать программный продукт, и системное программное обеспечение. Например, Web-приложение работает в браузере. Браузеры по-разному интерпретируют языки HTML и JavaScript, поэтому надо сразу решить, будет ли программный продукт рассчитан на определенный браузер или он должен работать в любом. В первом случае разработчики могут включить в продукт дополнительные возможности языков HTML и JavaScript, интерпретируемые данным браузером, во втором — должны использовать только стандартные конструкции, что может значительно затруднить разработку.

техническом проекте уточняются типы и структуры исходных и промежуточных данных, полностью детализируется алгоритм решения задачи. Задача разбивается на модули, которые распределяются среди программистов.

При объектно-ориентированном проектировании в техническом проекте определяются все объекты, необходимые для осуществления проекта и выявляются связи между ними. Полностью выписывается строение каждого объекта, его поля и методы. Объекты записываются в виде интерфейсов или абстрактных классов, дальнейшая разработка которых поручается конкретным программистам.

После проработки технического проекта каждым участником разработки собираются и обобщаются их уточнения и замечания. Окончательная версия проекта обсуждается командой разработчиков. Менеджер проекта выносит технический проект на утверждение руководством фирмы-разработчика и заказчиком программного продукта. После этого технический проект становится рабочим проектом для группы разработчиков.

Рабочий проект

После утверждения технического проекта он становится основным рабочим документом для команды разработчиков программного продукта. Рабочий проект — это большой, подробный документ, наиболее полно описывающий будущий программный продукт и план его создания. В нем содержатся детальные указания каждому разработчику и команде в целом, определена структура базы данных и других хранилищ данных, которой будут руководствоваться все разработчики. Короче говоря, в рабочем проекте должны содержаться все сведения, нужные каждому разработчику и команде в целом. В частности, в нем должны быть записаны этапы и сроки разработки, чтобы каждый программист твердо знал их.

При объектно-ориентированном проектировании в рабочем проекте должны быть полностью описаны все классы и связи между ними. Это описание можно сделать в виде абстрактных классов или интерфейсов, на языке разработки или на языке описания. Важно, чтобы все участники проекта правильно понимали эту запись и одинаково интерпретировали ее.

Каждому участнику проекта выдается экземпляр рабочего проекта. При всяком изменении рабочего проекта участники получают его новую версию. В настоящее время с развитием Web-технологии, как правило, создается собственный сайт для каждого проекта. Все рабочие документы публикуются на этом сайте, а при каждом их изменении участники проекта получают уведомление по электронной почте.

Задание 2:

1. Разработайте проект автоматизации библиотечного каталога.
2. Проведите анализ работы деканата и разработайте проект его автоматизации.
3. Проанализируйте информационные потоки вашего факультета и спроектируйте компьютерную систему их обработки.

Итог работы: отчет, защита работы.

Практическая работа № 21

Цель: Получение навыков разработки юнит-тестов и проведения тестирования программного обеспечения

Задание 1:

1. Откройте выбранную IDE и создайте проект на основе существующих программных кодов, реализующих алгоритм пирамидальной сортировки.
2. Подключите к проекту библиотеку JUnit.
3. Создайте каркас для юнит-тестов (Например, в IDE Eclipse можно выбрать нужный класс, открыть контекстное меню, и выбрать New->Junit test case, в появившемся диалоговом окне выбрать методы, для которых понадобятся юнит-тесты).
4. Создайте юнит-тест согласно описанным требованиям.
5. Отладьте и запустите юнит-тест.
6. Оцените результаты выполнения юнит-тестирования и сделайте соответствующие выводы.

Итог работы: отчет, защита работы

Практическая работа № 22

Цель: отработать навыки администрирования приложения

Задание 1:

1. Описать этапы проектирования модулей программы.
2. Составить в виде блок-схемы алгоритм решения задачи.
3. Составить отчет по практической работе.

Отчет по практической работе должен включать:

1. Алгоритм решения задачи.
2. Набор тестов для отладки программы.

Задача. Составить алгоритм решения задачи, приведенной ниже, с использованием структурных единиц: процедур и/или функций.

Варианты индивидуальных заданий.

1. Даны два двумерных массива вещественных элементов. Размер исходных массивов не превосходит 10x10 элементов. Для каждого из массивов указать номера столбцов, содержащих только положительные элементы. Если таковых столбцов в массиве нет, то вывести соответствующее сообщение. Проверку столбца на положительность элементов оформить в виде процедуры с передачей в нее всех элементов текущего столбца.
2. Даны два двумерных массива натуральных элементов. Размер исходных массивов не превосходит 10x10 элементов. Для каждого из массивов указать номера столбцов, содержащих только кратные 5 или 7 элементы. Если таких столбцов в массиве нет, то вывести соответствующее сообщение. Проверку столбца на наличие указанных элементов оформить в виде процедуры с передачей в нее всех элементов текущего столбца.
3. Даны пять одномерных массива вещественных элементов. Размер каждого массива не превосходит 100 элементов. Для каждого из массивов определить, составляют ли его элементы знакочередующуюся последовательность. Если да, то указать порядковый номер такого массива, в противном случае вывести отрицательный ответ. Проверку массива на выполнение условия оформить в виде процедуры с передачей в нее всех элементов рассматриваемого массива.
4. Даны два двумерных массива символьных (буквы русского алфавита) элементов. Размер исходных массивов не превосходит 10x10 элементов. Для каждого из массивов указать номера строк, содержащих элементы только строчных букв, если таких строк нет ни для какого массива, то вывести соответствующее сообщение. Проверку строки на наличие указанных элементов оформить в виде процедуры с передачей в нее всех элементов текущей строки.
5. Даны два двумерных массива вещественных элементов. Размер исходных массивов не превосходит 10x10 элементов. Для каждого из массивов указать количество столбцов, содержащих только не положительные элементы. Если таких столбцов нет ни для одного из массивов, то вывести соответствующее сообщение. Проверку столбца на наличие указанных элементов оформить в виде процедуры с передачей в нее всех элементов текущего столбца.
6. Даны пять одномерных массива вещественных элементов. Размер каждого массива не превосходит 100 элементов. Для каждого из массивов определить, составляют ли его элементы одного знака. Если да, то указать порядковый номер такого массива, в противном случае вывести отрицательный ответ. Проверку массива на выполнение условия оформить в виде процедуры с передачей в нее всех элементов рассматриваемого массива.
7. Даны два двумерных массива целочисленных элементов. Размер исходных массивов не превосходит 10x10 элементов. Для каждого из массивов указать количество строк, содержащих элементы, четность которых чередуется, а вторым в четных строках является нечетный элемент. Если таких строк нет ни для одного из массивов, то вывести соответствующее сообщение. Проверку строки на наличие указанных элементов оформить в виде процедуры с передачей в нее всех элементов текущего столбца.
8. Даны пять одномерных массива символьных (только латинские буквы) элементов. Размер каждого массива не превосходит 100 элементов. Для каждого из массивов определить, чередуются ли в нем буквы строчные и прописные. Если да, то указать

порядковый номер такого массива, в противном случае вывести отрицательный ответ. Проверку массива на выполнение условия оформить в виде процедуры с передачей в нее всех элементов рассматриваемого массива.

9. Даны два двумерных массива целочисленных элементов. Размер исходных массивов не превосходит 10x10 элементов. Для каждого из массивов указать количество строк, для которых сумма элементов, стоящих на нечетных местах в строке, является положительным числом. Если таких строк нет ни для одного из массивов, то вывести соответствующее сообщение. Проверку строки на выполнение условия и расчет оформить в виде процедуры с передачей в нее всех элементов текущей строки.
10. Даны два двумерных массива вещественных элементов. Размер исходных массивов не превосходит 10x10 элементов. Для каждого из массивов указать номера столбцов, произведение отрицательных элементов которых является положительным числом. Если таких столбцов нет ни для одного из массивов, то вывести соответствующее сообщение. Проверку столбца на выполнение условия и расчет оформить в виде процедуры с передачей в нее всех элементов текущего столбца.
11. Даны пять одномерных массива символьных (только латинские буквы) элементов. Размер каждого массива не превосходит 100 элементов. Для каждого из массивов определить, расположены ли в нем строчные буквы в алфавитном порядке. Если да, то указать порядковый номер такого массива, в противном случае вывести отрицательный ответ. Проверку массива на выполнение условия оформить в виде процедуры с передачей в нее всех элементов рассматриваемого массива.
12. Даны два двумерных массива целочисленных элементов. Размер исходных массивов не превосходит 10x10 элементов. Для каждого из массивов проверить выполнение условия: все четные строки массива таковы, что суммы их элементов образуют возрастающую последовательность. Вывести соответствующее сообщение. Вычисление суммы элементов массива и проверку последовательности чисел на выполнение условия оформить в виде процедуры с передачей в нее всех необходимых элементов.
13. Даны два двумерных массива вещественных элементов. Размер исходных массивов не превосходит 10x10 элементов. Преобразовать все нечетные строки каждого массива так, чтобы элементы составляли возрастающую по абсолютной величине последовательность. Вывести преобразованные массивы. Упорядочивание элементов оформить в виде процедуры с передачей в нее всех необходимых элементов.
14. Даны два двумерных массива целочисленных элементов. Размер исходных массивов не превосходит 10x10 элементов. Для каждого столбца массивов вычислить суммы и количества элементов, значения которых находятся в заданном диапазоне. Если чисел, удовлетворяющих этому условию нет, то вывести соответствующее сообщение. Вычисление для элементов столбца массива оформить в виде процедуры с передачей в нее всех необходимых элементов.
15. Даны пять одномерных массива символьных (только латинские буквы) элементов. Размер каждого массива не превосходит 100 элементов. Преобразовать все массивы так, чтобы все строчные буквы были расположены по алфавиту. При этом переставлять только строчные буквы, оставив прописные буквы на своих местах. Преобразование каждого массива оформить в виде процедуры с передачей в нее всех необходимых элементов. Если перестановка элементов не потребовалась, то есть исходные массивы удовлетворяют требуемому условию, то вывести соответствующее сообщение.

удовлетворяют требуемому условию, то вывести соответствующее сообщение.

Задание.

1. Разработать модули программы, спроектированные во время практического занятия
2. Отладить программу с использованием тестов, составленных во время практического занятия

Итог работы: отчет, защита работы.

Практическая работа № 23

Цель: изучить основные подходы к проектированию тестов

Задание 1: ознакомиться с материалом

Рассмотрим два основных подхода к проектированию тестов.

Первый подход ориентируется только на стратегию тестирования, называемую стратегией "черного ящика", тестированием с управлением по данным или тестированием с управлением по входу-выходу. При использовании этой стратегии программа рассматривается как черный ящик. Тестовые данные используются только в соответствии со спецификацией программы (т. е. без учета знаний о ее внутренней структуре). Недостижимый идеал сторонника первого подхода — проверить все возможные комбинации и значения на входе. Обычно их слишком много даже для простейших алгоритмов. Так, для программы расчета среднего арифметического четырех чисел надо готовить 10^7 тестовых данных.

При первом подходе обнаружение всех ошибок в программе является критерием исчерпывающего входного тестирования. Последнее может быть достигнуто, если в качестве тестовых наборов использовать все возможные наборы входных данных. Следовательно, приходим к выводу, что для исчерпывающего тестирования программы требуется бесконечное число тестов, а значит, построение исчерпывающего входного теста невозможно. Это подтверждается двумя аргументами: во-первых, нельзя создать тест, гарантирующий отсутствие ошибок; во-вторых, разработка таких тестов противоречит экономическим требованиям. Поскольку исчерпывающее тестирование исключается, нашей целью должна стать максимизация результативности капиталовложений в тестирование (максимизация числа ошибок, обнаруживаемых одним тестом). Для этого необходимо рассматривать внутреннюю структуру программы и делать некоторые разумные, но, конечно, не обладающие полной гарантией достоверности предположения.

Второй подход использует стратегию "белого ящика", или стратегию тестирования, управляемую логикой программы, которая позволяет исследовать внутреннюю структуру программы. В этом случае тестировщик получает тестовые данные путем анализа только логики программы; стремится, чтобы каждая команда была выполнена хотя бы один раз. При достаточной квалификации добивается, чтобы каждая команда условного перехода выполнялась бы в каждом направлении хотя бы один раз. Цикл должен выполняться один раз, ни разу, максимальное число раз. Цель тестирования всех путей извне также недостижима. В программе из двух последовательных циклов внутри каждого из них включено ветвление на десять путей, имеется 10^{18} путей расчета. Причем выполнение всех путей расчета не гарантирует выполнения всех спецификаций.

Сравним способ построения тестов при данной стратегии с исчерпывающим входным тестированием стратегии "черного ящика". Неверно предположение, что достаточно построить такой набор тестов, в котором каждый оператор исполняется хотя бы один раз. Исчерпывающему входному тестированию может быть поставлено в соответствие исчерпывающее тестирование маршрутов. Подразумевается, что программа проверена полностью, если с помощью тестов удается осуществить выполнение этой программы по всем возможным маршрутам ее потока (графа) передач управления.

Последнее утверждение имеет два слабых пункта: во-первых, число не повторяющих друг друга маршрутов — астрономическое; во-вторых, даже если каждый маршрут может быть проверен, сама программа может содержать ошибки (например, некоторые маршруты пропущены).

Свойство пути выполняться правильно для одних данных и неправильно для других — называемое чувствительностью к данным, наиболее часто проявляется за счет численных погрешностей и погрешностей усечения методов. Тестирование каждого из всех маршрутов одним тестом не гарантирует выявление чувствительности к данным.

В результате всех изложенных выше замечаний отметим, что ни исчерпывающее входное тестирование, ни исчерпывающее тестирование маршрутов не могут стать полезными стратегиями, потому что оба они нереализуемы. Поэтому реальным путем, который позволит создать хорошую, но, конечно, не абсолютную стратегию, является сочетание тестирования программы несколькими методами.

Рассмотрим пример тестирования оператора:

if A and B then…

при использовании разных критериев полноты тестирования.

При критерии покрытия условий требовались бы два теста: ***A = true, B = false*** и ***A = false, B = true***. Но в этом случае не выполняется then-предложение оператора if.

Существует еще один критерий, названный покрытием решений/условий. Он требует такого достаточного набора тестов, чтобы все возможные результаты каждого условия в решении выполнялись, по крайней мере, один раз; все результаты каждого решения выполнялись тоже один раз и каждой точке входа передавалось управление, по крайней мере, один раз.

Недостатком критерия покрытия решений/условий является невозможность его применения для выполнения всех результатов всех условий. Часто подобное выполнение имеет место вследствие того, что определенные условия скрыты другими условиями. Например, если условие AND есть ложь, то никакое из последующих условий в выражении не будет выполнено. Аналогично, если условие OR есть истина, то никакое из последующих условий не будет выполнено. Следовательно, критерии покрытия условий и покрытия решений/условий недостаточно чувствительны к ошибкам в логических выражениях.

Критерием, который решает эти и некоторые другие проблемы, является комбинаторное покрытие условий. Он требует создания такого числа тестов, чтобы все возможные комбинации результатов условия в каждом решении и все точки входа выполнялись, по крайней мере, один раз.

В случае циклов число тестов для удовлетворения критерию комбинаторного покрытия условий обычно больше, чем число путей.

Легко видеть, что набор тестов, удовлетворяющий критерию комбинаторного покрытия условий, удовлетворяет также и критериям покрытия решений, покрытия условий и покрытия решений/условий.

Таким образом, для программ, содержащих только одно условие на каждое решение, минимальным является критерий, набор тестов которого вызывает выполнение всех результатов каждого решения, по крайней мере, один раз; передает управление каждой точке входа (например, оператор CASE).

Для программ, содержащих решения, каждое из которых имеет более одного условия, минимальный критерий состоит из набора тестов, вызывающих все возможные комбинации результатов условий в каждом решении и передающих управление каждой точке входа программы, по крайней мере, один раз.

Деление алгоритма на типовые стандартные структуры позволяет минимизировать усилия программиста, затрачиваемые им на тестирование. Запрет на вложенные структуры как раз и объясняется излишними затратами на тестирование. Использование цепочки простых альтернатив с одним действием или структуры ВЫБОР вместо вложенных простых АЛЬТЕРНАТИВ значительно сокращает число тестов!

Задание 2:

1. Оформить внешнюю спецификацию.
2. Составить в виде блок-схемы алгоритм решения задачи.
3. Создать программу решения задачи на любом алгоритмическом языке программирования.
4. Составить набор тестов и провести тестирование созданной программы с помощью методов «белого ящика» (покрытия операторов, покрытия решений, покрытия условий, комбинаторного покрытия условий).

Задача. «Нахождение характерных точек функции». Составить алгоритм и написать программу последовательного вычисления значений заданной функции Y(X) до тех пор, пока не будет пройдена некоторая характерная точка графика функции. Значения аргумента X составляют возрастающую последовательность с шагом h. Начальное значение X_0 и шаг изменения аргумента h задаются пользователем.

Варианты индивидуальных заданий.

	Вид функции Y(X),		Вид функции Y(X),

№ п/п	характерные точки	№ п/п	характерные точки
	Локальный минимум функции $x^2+0.5-\sin(3*x)$	6	Точка (точки), в которой функция $\left\|\frac{2x^3-3}{x^2+1}\right\|$ равна 10.
	Точка (точки), для которой $\sqrt{x^2+5}-1=5$.	7	Локальный минимум функции $3(x+4)^2-\sin(x+15)$
	Пересечение графиков функций $\sqrt{x^2+1}(3+x)$ и $\frac{5}{x^2-7}$.	8	Точка (точки), в которой функция $\frac{2x^2}{\sin(x-1)}-1$ равна -500.
	Все нули функции $x^2-\sin(3x)$.	9	Локальный максимум функции $\frac{2x+15}{3+x^2}$
	Локальный минимум функции $\frac{5}{3x-x^2-3}$	0	Пересечение графиков функций $(x^2+1)\sin(3+x)$ и $\frac{x+5}{x^2+1}$.
	Точка (точки), в которой функция x^2-e^x равна -10.	1	Локальный минимум функции $x^2-3x+15$
	Пересечение графиков функций $100\sin(1+x)$ и x^3+5.	2	Все нули функции $\sqrt{x^2+0.5-\sin(3x)}$.
	Локальный минимум функции $\sqrt{e^x}-x$	3	Локальный максимум функции $200x-e^x$
	Все нули функции $\frac{x^2-15}{\ln x^2+3x-7}$.	4	Все нули функции $x-e^x\cos(x)$.
0	Локальный максимум функции $\frac{\sqrt{2x^2+1}}{3+x^2}$	5	Точка (точки), в которой функция $\frac{x^2+5}{x^2+7}$ равна 15.
1	Пересечение графиков функций $10\cos(x)$ и $x^3/3+5$.	6	Все нули функции $\frac{x^2-15}{2x^2+3x-7}$.
2	Локальный максимум функции $-(x-2)^2+\cos(x)$	7	Пересечение графиков функций $3x^2-15$ и $10\cos(x)+7$.
3	Точка (точки), в которой функция $3x^2-\frac{12x-5}{x^2+1}$ равна 5.	8	Локальный минимум функции $-100x+e^x$
4	Все нули функции $\frac{x+5}{x^2-7}$.	9	Локальный максимум функции $-\frac{x}{x^2+3}$
5	Локальный максимум функции $-e^{-x}-100x$	0	Пересечение графиков функций $4(x-5)^2-15$ и $\frac{100}{x^2+2}-7$.

Итог работы: отчет, защита работы.

Практическая работа № 24

Цель: получение практических навыков обработки исключений

Задание 1:

1. Помогите тестировщику определить возможные исключительные ситуации вашей ИС, определить экспериментально, ошибки каких классов могут быть сгенерированы в вашем приложении
2. Создать обработчики исключительных ситуаций с использованием выявленных классов и всех секций конструкции обработчика с соответствующими сообщениями, позволяющими корректно выполнить программу. Обработка исключительных ситуаций рассмотрена на примере Java. Вы пишете обработчики на выбранном языке программирования для Вашей системы.

Задание 2:

Ответьте на вопросы:

1. Что представляет собой исключение?
2. На какие части исключения позволяют разделить вычислительный процесс? Достоинства такого подхода?
3. Какой оператор используется для генерации исключительной ситуации?
4. Что представляет собой контролируемый блок? Для чего он нужен?
5. Что представляет собой секция-ловушка? Для чего она нужна?
6. Какие формы может иметь спецификация исключения в секции ловушке? В каких ситуациях используются эти формы?
7. Какой стандартный класс можно использовать для создания собственной иерархии исключений?
8. Каким образом можно создать собственную иерархию исключений?
9. В какой части программы может генерироваться исключение?

Итог работы: отчет, защита работы.

Практическая работа № 25

Цель: получение практических навыков применения отладочных классов в проекте

Задание 1:

1. Создать основной класс. Создать 3 класса наследника. Создать в основном классе 5 методов и 3 свойства, связанных с его назначением, отличных от методов и свойств других подгрупп.

2. Придумать и создать, как минимум, по 2 метода в каждом классе-наследнике (всего не меньше 6) и хотя бы по 2 свойства (всего не меньше 6). Методы и свойства должны быть связаны с особенностями класса-наследника.

3. Сделать, как минимум, 3 метода основного класса переопределёнными в классах наследниках. Сделать, как минимум, 1 метод, который нельзя переопределить.

4. Скомпилировать программу, демонстрирующую работу каждого из методов каждого из классов.

Варианты на лабораторную работу

В ходе лабораторной работы нужно разработать программу.

1,11,21. Основной класс «Автомобили», 3 наследника «Грузовики», «Легковые» и «Автобусы».

2,12,22. Основной класс «Одежда», 3 наследника «Куртки», «Шубы» и «Пуховики».

3,13,23. Основной класс «Еда», 3 наследника «Супы», «Закуски» и «Напитки».

4,14,24. Основной класс «Гаджеты», 3 наследника «Планшеты», «Смартфоны» и «Нетбуки».

5,15,25. Основной класс «Овощи», 3 наследника «Морковь», «Свёкла» и «Лук».

6,16,26. Основной класс «Песни», 3 наследника «Частушки», «Баллады» и «Романсы».

7,17,27. Основной класс «Оружие», 3 наследника «Пистолеты», «Автоматы» и «Пулеметы».

8,18,28. Основной класс «Документы», 3 наследника «Справки», «Приказы» и «Заявления».

9,19,29. Основной класс «Погода», 3 наследника «Пасмурно», «Ясно» и «Ураган».

10,20,30. Основной класс «Звери», 3 наследника «Зайцы», «Волки» и «Лисы».

Итог работы: отчет, защита работы

Практическая работа № 26

Цель: получение практических навыков тестирования и отладки программ

Задание 1: ознакомиться с материалом

Теоретические основы. Тестирование – процесс выполнения программы на наборе тестов с целью выявления ошибок.

Локализацией называют процесс определения оператора программы, выполнение которого вызвало нарушение нормального вычислительного процесса. Для исправления ошибки необходимо определить ее причину, т.е. определить оператор или фрагмент, содержащие ошибку. Причины ошибок могут быть как очевидны, так и очень глубоко скрыты. В целом сложность отладки обусловлена следующими причинами:

- требует от программиста глубоких знаний специфики управления используемыми техническими средствами, операционной системы, среды и языка программирования, реализуемых процессов, природы и специфики различных ошибок, методик отладки и соответствующих программных средств;
- психологически дискомфортна, так как необходимо искать собственные ошибки и, как правило, в условиях ограниченного времени;
- возможно взаимовлияние ошибок в разных частях программы, например, за счет затирания области памяти одного модуля другим из-за ошибок адресации;
- отсутствуют четко сформулированные методики отладки.

Отладка программы в любом случае предполагает обдумывание и логическое осмысление всей имеющейся информации об ошибке. Большинство ошибок можно обнаружить по косвенным признакам посредством тщательного анализа текстов программ и результатов тестирования без получения дополнительной информации. При этом используют различные методы:

- ручного тестирования;
- индукции;
- дедукции;
- обратного прослеживания.

Метод ручного тестирования

Это - самый простой и естественный способ данной группы. При обнаружении ошибки необходимо выполнить тестируемую программу вручную, используя тестовый набор, при работе с которыми была обнаружена ошибка. Метод очень эффективен, но не применим для больших программ, программ со сложными вычислениями и в тех случаях, когда ошибка связана с неверным представлением программиста о выполнении некоторых операций. Данный метод часто используют как составную часть других методов отладки.

Метод индукции

Метод основан на тщательном анализе симптомов ошибки, которые могут проявляться как неверные результаты вычислений или как сообщение об ошибке. Если компьютер просто "зависает", то фрагмент проявления ошибки вычисляют, исходя из последних полученных результатов и действий пользователя. Полученную таким образом информацию организуют и тщательно изучают, просматривая соответствуюший фрагмент программы. В результате этих действий выдвигают гипотезы об ошибках, каждую из которых проверяют. Если гипотеза верна, то детализируют информацию об ошибке, иначе - выдвигают другую гипотезу. Последовательность выполнения отладки методом индукции показана на рисунке в виде схемы алгоритма.

Самый ответственный этап - выявление симптомов ошибки. Организуя данные об ошибке, целесообразно записать все, что известно о её проявлениях, причем фиксируют, как ситуации, в которых фрагмент с ошибкой выполняется нормально, так и ситуации, в которых ошибка проявляется. Если в результате изучения данных никаких гипотез не появляется, то необходима дополнительная информация об ошибке. Дополнительную информацию можно получить, например, в результате выполнения схожих тестов. В процессе доказательства пытаются выяснить, все ли проявления ошибки объясняет данная гипотеза, если не все, то либо гипотеза не верна, либо ошибок несколько.

Метод дедукции

По методу дедукции вначале формируют множество причин, которые могли бы вызвать данное проявление ошибки. Затем анализируя причины, исключают те, которые противоречат имеющимся данным. Если все причины исключены, то следует выполнить дополнительное тестирование исследуемого фрагмента. В противном случае наиболее вероятную гипотезу пытаются доказать. Если гипотеза объясняет полученные признаки ошибки, то ошибка найдена, иначе - проверяют следующую причину.

Метод обратного прослеживания

Для небольших программ эффективно применение метода обратного прослеживания. Начинают с точки вывода неправильного результата. Для этой точки строится гипотеза о значениях основных переменных, которые могли бы привести к получению имеющегося результата. Далее, исходя из этой гипотезы, делают предложения о значениях переменных в предыдущей точке. Процесс продолжают, пока не обнаружат причину ошибки.

Задание 2:

Задание.

1. Составить в виде блок-схемы алгоритм решения задачи.
2. Создать программу решения задачи на любом алгоритмическом языке программирования.
3. Отладить программу.

Задача: составить список учебной группы, включающей 25 человек. Для каждого учащегося указать дату рождения, год поступления в колледж, курс, группу, оценки каждого года обучения.

Назначение задачи: получить значение определённого критерия и упорядочить список студентов по нему.

Достигаемая цель: упорядочить список студентов по среднему баллу и получить его.

Итог работы: отчет, защита работы.

Практическая работа № 27

Цель: получение практических навыков разработки модулей программной системы и интеграции этих модулей

Задание 1: ознакомиться с материалом

Теоретические сведения Термин «интеграция» относится к такой операции в процессе разработки ПО, при которой вы объединяете отдельные программные компоненты в единую систему. В небольших проектах интеграция может занять одно утро и заключаться в объединении горстки классов. В больших — могут потребоваться недели или месяцы, чтобы связать воедино весь набор программ. Независимо от размера задач в них применяются одни и те же принципы.

Тема интеграции тесно переплетается с вопросом последовательности конструирования. Порядок, в котором вы создаете классы или компоненты, влияет на порядок их интеграции: вы не можете интегрировать то, что еще не было создано. Последовательности интеграции и конструирования имеют большое значение.

Поскольку интеграция выполняется после того, как разработчик завершил модульное тестирование, и одновременно с системным тестированием, ее иногда считают операцией, относящейся к тестированию. Однако она достаточно сложна, и поэтому ее следует рассматривать как независимый вид деятельности.

Аккуратная интеграция обеспечивает:

- упрощенную диагностику дефектов;
- меньшее число ошибок;
- меньшее количество «лесов»;
- раннее создание первой работающей версии продукта;
- уменьшение общего времени разработки;
- лучшие отношения с заказчиком;
- улучшение морального климата;
- увеличение шансов завершения проекта;
- более надежные оценки графика проекта;
- более аккуратные отчеты о состоянии;
- лучшее качество кода;
- меньшее количество документации.

Интеграция программ выполняется посредством ***поэтапного*** или ***инкрементного*** подхода.

Поэтапная интеграция состоит из этапов, перечисленных ниже:

1. «Модульная разработка»: проектирование, кодирование, тестирование и отладка каждого класса.
2. «Системная интеграция»: объединение классов в одну огромную систему.
3. «Системная дезинтеграция»: тестирование и отладка всей системы.

Проблема поэтапной интеграции в том, что, когда классы в системе впервые соединяются вместе, неизбежно возникают новые проблемы и их причины могут быть в чем угодно. Поскольку у вас масса классов, которые никогда раньше не работали вместе, виновником может быть плохо протестированный класс, ошибка в интерфейсе между двумя классами или ошибка, вызванная взаимодействием двух классов. Все классы находятся под подозрением.

Неопределенность местонахождения любой из проблем сочетается с тем фактом, что все эти проблемы вдруг проявляют себя одновременно. Это заставляет вас иметь дело не только с проблемами, вызванными взаимодействием классов, но и другими ошибками, которые трудно диагностировать, так как они взаимодействуют.

Поэтому поэтапную интеграцию называют еще «интеграцией большого взрыва»

Поэтапную интеграцию нельзя начинать до начала последних стадий проекта, когда будут разработаны и протестированы все классы. Когда классы, наконец, будут объединены и проявится большое число ошибок, программисты тут же ударятся в паническую отладку вместо методического определения и исправления ошибок.

Для небольших программ — нет, а для крошечных — поэтапная интеграция может быть наилучшим подходом. Если программа состоит из двух-трех классов, поэтапная интеграция может сэкономить ваше время, если вам повезет. Но в большинстве случаев инкрементный подход будет лучше.

При **инкрементной интеграции** вы пишете и тестируете маленькие участки программы, а затем комбинируете эти кусочки друг с другом по одному. При таком подходе — по одному элементу за раз — вы выполняете перечисленные далее действия:

1. Разрабатываете небольшую, функциональную часть системы. Это может быть наименьшая функциональная часть, самая сложная часть, основная часть или их комбинация. Тщательно тестируете и отлаживаете ее. Она послужит скелетом, на котором будут наращиваться мускулы, нервы и кожа, составляющие остальные части системы.
2. Проектируете, кодируете, тестируете и отлаживаете класс.
3. Прикрепляете новый класс к скелету. Тестируете и отлаживаете соединение скелета и нового класса. Убеждаетесь, что эта комбинация работает, прежде чем переходить к добавлению нового класса. Если дело сделано, повторяете процесс, начиная с п. 2.

Инкрементный подход имеет массу преимуществ перед традиционным поэтапным подходом независимо от того, какую инкрементную стратегию вы используете:

Ошибки можно легко обнаружить Когда во время инкрементной интеграции возникает новая проблема, то очевидно, что к этому причастен новый класс. Либо его интерфейс с остальной частью программы неправилен, либо его взаимодействие с ранее интегрированными классами приводит к ошибке. В любом случае вы точно знаете, где искать проблему.

В таком проекте система раньше становится работоспособной Когда код интегрирован и способен выполняться, даже если система еще не пригодна к использованию, это выглядит так, будто это скоро произойдет. При инкрементной интеграции программисты раньше видят результаты своей работы, поэтому их моральное состояние лучше, чем в том случае, когда они подозревают, что их проект может никогда не сделать первый вдох.

Вы получаете улучшенный мониторинг состояния При частой интеграции реализованная и нереализованная функциональность видна с первого взгляда. Менеджеры будут иметь лучшее представление о состоянии проекта, видя, что 50% системы уже работает, а не слыша, что кодирование «завершено на 99%».

Вы улучшите отношения с заказчиком Если частая интеграция влияет на моральное состояние разработчиков, то она также оказывает влияние и на моральное состояние заказчика. Клиенты любят видеть признаки прогресса, а инкрементная интеграция предоставляет им такую возможность достаточно часто.

Системные модули тестируются гораздо полнее Интеграция начинается на ранних стадиях проекта. Вы интегрируете каждый класс по мере его готовности, а не ожидая одного внушительного мероприятия по интеграции в конце разработки. Программист тестирует классы в обоих случаях, но в качестве элемента общей системы они используются гораздо чаще при инкрементной, чем при поэтапной интеграции.

Вы можете создать систему за более короткое время Если интеграция тщательно спланирована, вы можете проектировать одну часть системы в то время, когда другая часть уже кодируется. Это не уменьшает общее число человеко-часов, требуемых для полного проектирования и кодирования, но позволяет выполнять часть работ параллельно, что является преимуществом в тех случаях, когда время имеет критическое значение.

При поэтапной интеграции вам не нужно планировать порядок создания компонентов проекта. Все компоненты интегрируются одновременно, поэтому вы можете разрабатывать их в любом порядке — главное, чтобы они все были готовы к часу X.

При инкрементной интеграции вы должны планировать более аккуратно. Большинство систем требует интеграции некоторых компонентов перед интеграцией других. Так что планирование интеграции влияет на планирование конструирования — порядок, в котором конструируются компоненты, должен обеспечивать порядок, в котором они будут интегрироваться.

Нисходящая интеграция

При нисходящей интеграции класс на вершине иерархии пишется и интегрируется первым. Вершина иерархии — это главное окно, управляющий цикл приложения, объект, содержащий метод main() в программе на Java, функция WinMain() в программировании для Microsoft Windows или аналогичные. Для работы этого верхнего класса пишутся заглушки. Затем, по мере интеграции классов сверху вниз, классы заглушек заменяются реальными.

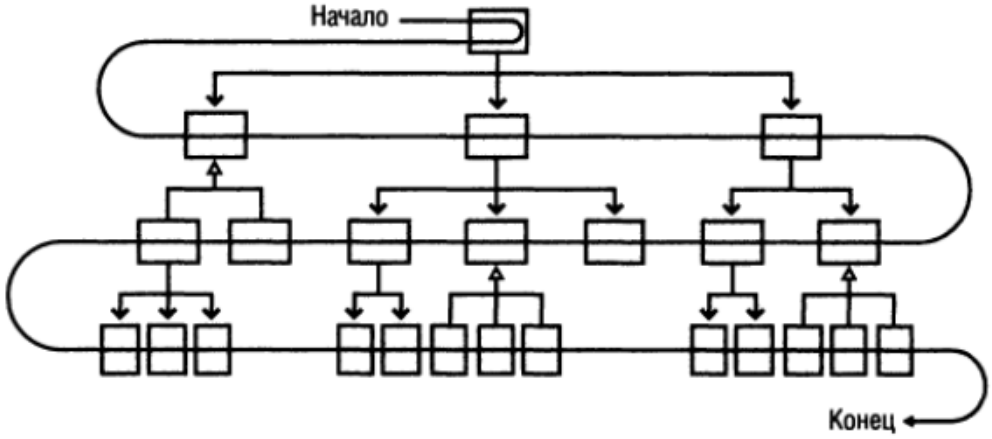
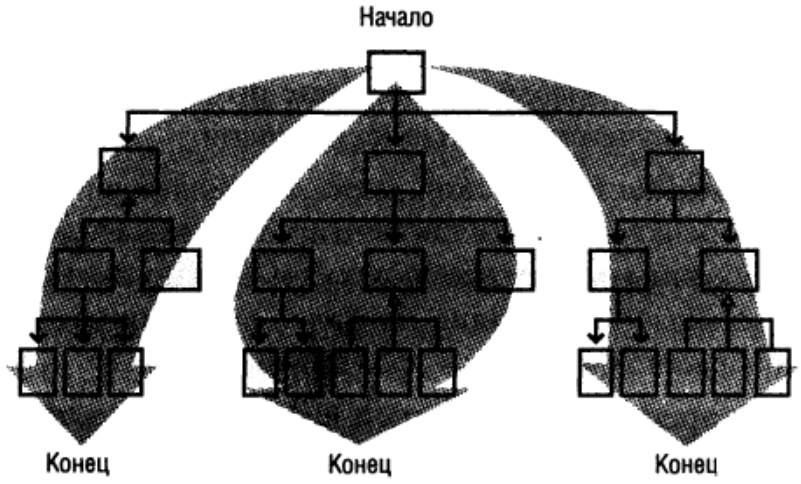


Рис.3 Нисходящая интеграция

При нисходящей интеграции вы создаете те классы, которые находятся на вершине иерархии, первыми, а те, что внизу, — последними.

Хорошей альтернативой нисходящей интеграции в чистом виде может стать подход с вертикальным секционированием.

Рис. 4 Вертикальное секционирование



—

При этом систему реализуют сверху вниз по частям, возможно, по очереди выделяя функциональные области и переходя от одной к другой.

Восходящая интеграция

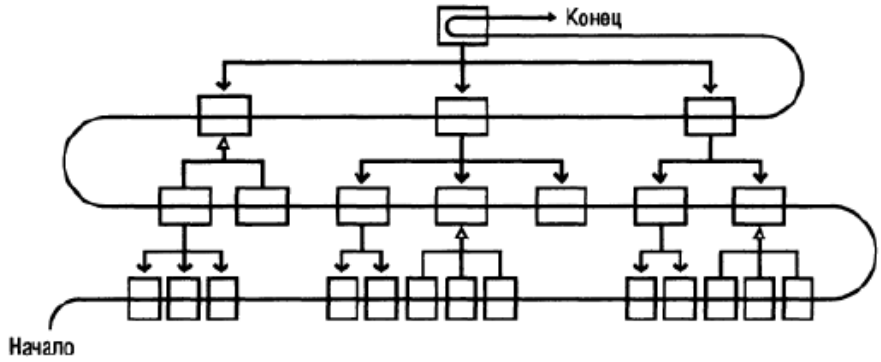


Рис.5 Восходящая интеграция

При восходящей интеграции вы пишете и интегрируете сначала классы, находящиеся в низу иерархии. Добавление низкоуровневых классов по одному, а не всех одновременно — вот что делает восходящую интеграцию инкрементной стратегией. Сначала вы пишете тестовые драйверы для выполнения низкоуровневых классов, а затем добавляете эти классы к тестовым драйверам, пристраивая их по мере готовности. Добавляя класс более высокого уровня, вы заменяете классы драйверов реальными.

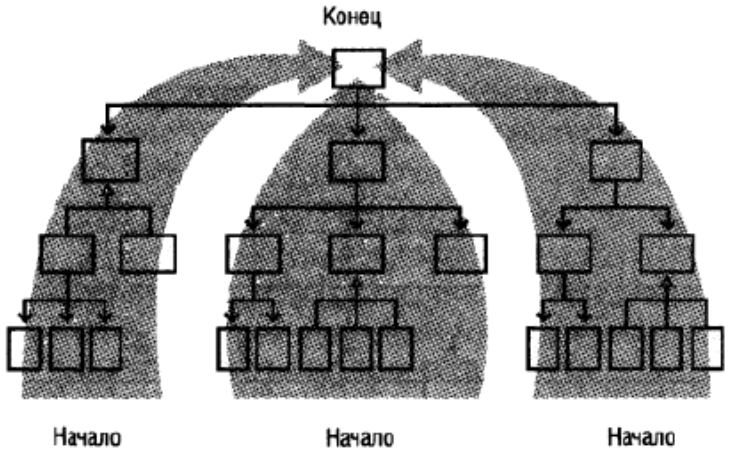


Рис. 6 Гибридный подход при восходящей интеграции

Как и нисходящую, восходящую интеграцию в чистом виде используют редко — вместо нее можно применять гибридный подход, реализующий секционную интеграцию.

Сэндвич-интеграция

Проблемы с нисходящей и восходящей интеграциями в чистом виде привели к тому, что некоторые эксперты стали рекомендовать сэндвич-подход.

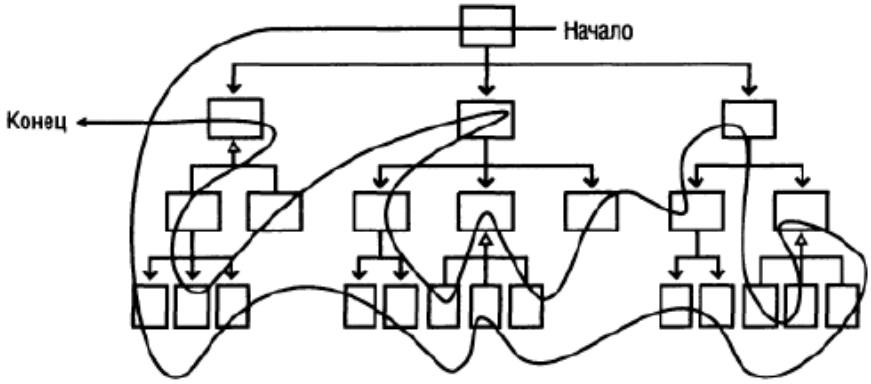


Рис. 7 Сэндвич-интеграция

Сначала вы объединяете высокоуровневые классы бизнес-объектов на вершине иерархии. Затем добавляете классы, взаимодействующие с аппаратной частью, и широко используемые вспомогательные классы в низу иерархии.

Напоследок вы оставляете классы среднего уровня.

Риск-ориентированная интеграция

Риск-ориентированную интеграцию, которую также называют «интеграцией, начиная с самых сложных частей» (hard part first integration), похожа на сэндвич-интеграцию тем, что пытается избежать проблем, присущих нисходящей или восходящей интеграциям в чистом виде. Кроме того, в ней также есть тенденция к объединению классов верхнего и нижнего уровней в первую очередь, оставляя классы среднего уровня напоследок. Однако суть в другом.

При риск-ориентированной интеграции вы определяете степень риска, связанную с каждым классом. Вы решаете, какие части системы будут самыми трудными, и реализуете их первыми.

Функционально-ориентированная интеграция

Еще один поход — интеграция одной функции в каждый момент времени. Под «функцией» понимается не нечто расплывчатое, а какое-нибудь поддающееся определению свойство системы, в которой выполняется интеграция.

Когда интегрируемая функция превышает по размерам отдельный класс, то «единица приращения» инкрементной интеграции становится больше отдельного класса. Это немного снижает преимущество инкрементного подхода в том плане, что уменьшает вашу уверенность об источнике новых ошибок. Однако если вы тщательно тестировали классы, реализующие эту функцию, перед интеграцией, то это лишь небольшой недостаток. Вы можете использовать стратегии инкрементной интеграции рекурсивно, сформировав сначала из небольших кусков отдельные свойства, а затем инкрементно объединив их в систему.

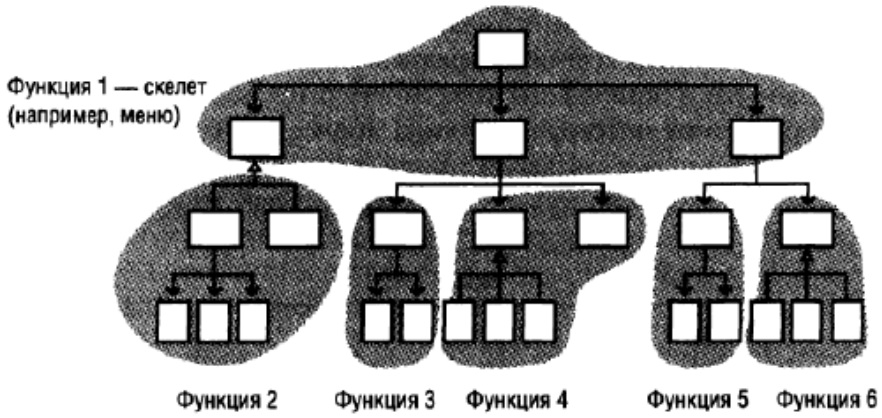


Рис. 8 Функционально-ориентированная интеграция

Обычно процесс начинается с формирования скелета, поскольку он способен поддерживать остальную функциональность. В интерактивной системе такой изначальной опцией может стать система интерактивного меню. Вы можете прикреплять остальную функциональность к той опции, которую интегрировали первой.

Т-образная интеграция

Последний подход, который часто упоминается в связи с проблемами нисходящей и восходящей методик, называется «Т-образной интеграцией». При таком подходе выбирается некоторый вертикальный слой, который разрабатывается и интегрируется раньше других. Этот слой должен проходить сквозь всю систему от начала до конца и позволять выявлять основные проблемы в допущениях, сделанных при проектировании системы. Реализовав этот вертикальный участок (и устранив все связанные с этим проблемы), можно разрабатывать основную канву системы (например, системное меню для настольного приложения). Этот подход часто комбинируют с риск-ориентированной и функционально-ориентированной интеграциями.

1-ообразная интеграция

Последний подход, который часто упоминается в связи с проблемами нисходящей и восходящей методик, называется «Т-образной интеграцией». При таком подходе выбирается некоторый вертикальный слой, который разрабатывается и интегрируется раньше других. Этот слой должен проходить сквозь всю систему от начала до конца и позволять выявлять основные проблемы в допущениях, сделанных при проектировании системы. Реализовав этот вертикальный участок (и устранив все связанные с этим проблемы), можно разрабатывать основную канву системы (например, системное меню для настольного приложения). Этот подход часто комбинируют с риск-ориентированной и функционально-ориентированной интеграциями.

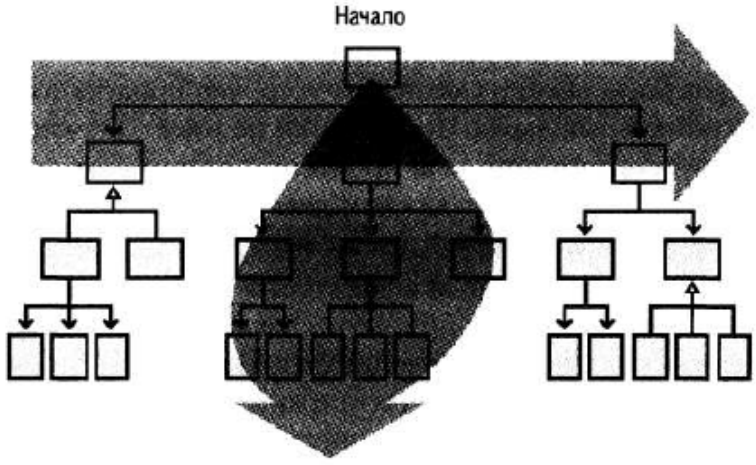


Рис. 9 Т-образная интеграция

Задание 2:

1. Оформить внешнюю спецификацию.
2. Составить в виде блок-схемы алгоритм решения задачи.
3. Спроектировать и разработать модули программы для решения задачи на любом алгоритмическом языке программирования.
4. Выполнить отладку и тестирование модулей программы.
5. Выполнить инкрементную интеграцию модулей с использованием одного из подходов.
6. Выполнить системное тестирование программы.

Задача. Задан двумерный массив размерности n×m. Отсортировать элементы строк массива по возрастанию значений, а затем отсортировать строки массива по возрастанию среднего арифметического элементов строк.

Реализовать сортировку разными способами и сравнить эффективность этих способов для разных исходных данных.

Контрольные вопросы.

1. Значение фазы интеграции программных модулей.
2. Подходы к интегрированию программных модулей.

Эффективность и оптимизация программ.

Итог работы: отчет, защита работы.

Практическая работа № 28

Цель: получение практических навыков автоматической генерации теста на основе формального описания

Задание 1: ознакомиться с материалом

Практически все программные системы предусматривают интерфейс с оператором. Практически всегда этот интерфейс – графический (GUI – Graphical User's Interface). Соответственно, актуальна и задача тестирования создаваемого графического интерфейса.

Вообще говоря, задача тестирования создаваемых программ возникла практически одновременно с самими программами. Известно, что эта задача очень трудоёмка как в смысле усилий по созданию достаточного количества тестов (отвечающих заданному критерию тестового покрытия), так и в смысле времени прогона всех этих тестов. Поэтому решение этой задачи стараются автоматизировать (в обоих смыслах).

Для решения задачи тестирования программ с программным интерфейсом (API – Application Program Interface: вызовы методов или процедур, пересылки сообщений) известны подходы – методы и инструменты – хорошо зарекомендовавшие себя в индустрии создания программного обеспечения. Основа этих подходов следующая: создается формальная спецификация программы, и по этой спецификации генерируются как сами тесты, так и тестовые оракулы – программы, проверяющие правильность поведения тестируемой программы. Спецификации, как набор требований к создаваемой программе, существовали всегда, Ключевым словом здесь является формальная спецификация. Формальная спецификация – это спецификация в форме, допускающей её формальные же преобразования и обработку компьютером. Это позволяет анализировать набор требований с точки зрения их полноты, непротиворечивости и т.п. Для задачи автоматизации тестирования эта формальная запись должна также обеспечивать возможность описания формальной связи между понятиями, используемыми в спецификации, и сущностями языка реализации программы.

Правильность функционирования системы определяется соответствием реального поведения системы эталонному поведению. Для того чтобы качественно определять это соответствие, нужно уметь формализовать эталонное поведение системы. Распространённым способом описания поведения системы является описание с помощью диаграмм UML (Unified Modeling Language). Стандарт UML предлагает использование трех видов диаграмм для описания графического интерфейса системы:

- ✓ Диаграммы сценариев использования (Use Case).
- ✓ Диаграммы конечных автоматов (State Chart).
- ✓ Диаграммы действий (Activity).

С помощью UML/Use Case diagram можно описывать на высоком уровне наборы сценариев использования, поддерживаемых системой. Данный подход имеет ряд преимуществ и недостатков по отношению к другим подходам, но существенным с точки зрения автоматизации генерации тестов является недостаточная формальная строгость описания.

Широко распространено использование UML/State Chart diagram для спецификации поведения системы, и такой подход очень удобен с точки зрения генерации тестов. Но составление спецификации поведения современных систем с помощью конечных автоматов есть очень трудоёмкое занятие, так как число состояний системы очень велико. С ростом функциональности системы спецификация становится всё менее и менее наглядной.

Перечисленные недостатки этих подходов к описанию поведения системы преодолеваются с помощью диаграмм действий (Activity). С одной стороны нотация UML/Activity diagram является более строгой, чем у сценариев использования, а с другой стороны предоставляет более широкие возможности по сравнению с диаграммами конечных автоматов.

Для создания прототипа работающей версии данного подхода используется инструмент Rational Rose. В первую очередь для спецификации графического интерфейса пользователя при помощи диаграмм действий UML.

Для прогона сгенерированных по диаграмме состояний тестов используется инструмент Rational Robot. Из возможностей инструмента в работе мы использовали следующие:

1. Возможность выполнять тестовые воздействия, соответствующие переходам между состояниями в спецификации.

2. Возможность проверять соответствие свойств объектов реальной системы и эталонных свойств, содержащихся в спецификации. Из тех возможностей, которые доступны с помощью этого инструмента, используется проверка следующих свойств объектов:

- Наличие и состояние окон (заголовок, активность, доступность, статус).
- Наличие и состояние таких объектов, как PushButton, CheckBox, RadioButton, List, Tree и др. (текст, доступность, размер).
- Значение буфера обмена.
- Наличие в оперативной памяти запущенных процессов.
- Существование файлов.

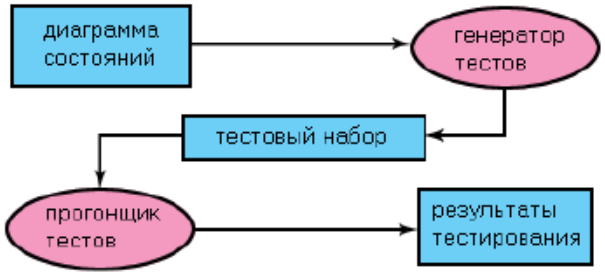


Рис. 2 Общая схема генерации и прогона тестов

Генератор строит по диаграмме состояний набор тестов. Условием окончания работы генератора является выполнение тестового покрытия. В данном случае в качестве покрытия было выбрано условие прохода по всем рёбрам графа состояний, то есть выполнения каждого доступного тестового воздействия из каждого достижимого состояния.

Автоматическая генерация тестов по диаграммам действий имеет следующие преимущества перед остальными подходами к тестированию графического интерфейса:

Генератор тестов есть программа (script), написанная на языке 'Rational Rose Scripting language' (расширение к языку Summit BasicScriptLanguage). В Rational Rose есть встроенный интерпретатор инструкций, написанных на этом языке, посредством которого можно обращаться ко всем объектам модели (диаграммы состояний).

- Спецификация автоматически интерпретируется (тем самым она проверяется и компилируется в набор тестов).
- Если какая-то функциональность системы изменилась, то диаграмму состояний достаточно изменить в соответствующем месте, и затем сгенерировать новый тестовый набор. Фактически, это снимает большую часть проблем, возникающих при организации регрессионного тестирования.
- Гарантия тестового покрытия. Эта гарантия даётся соответствующим алгоритмом обхода графа состояний.

В процессе построения обхода, генератор тестов компилирует набор тестов - инструкции на языке SQABasic. Эти инструкции есть чередование тестовых воздействий и оракула свойств объектов, соответствующих данному состоянию.

Задание 2:

1. Сформировать диаграмму вариантов использования для задачи
2. Сгенерировать набор тестов.

Итог работы: отчет, защита работы.

Практическая работа № 29

Цель: получение практических навыков использования средств автоматизации тестирования

Задание 1: ознакомиться с материалом

Для того чтобы продолжать тестирование, когда один тест не прошёл, в генератор тестов встроена возможность выбора – генерировать один большой тест или набор атомарных тестов. Атомарный тест – тот, который не требует приведения системы в состояние, отличное от начального состояния.

В связи с наличием ограничения инструмента прогона тестов на тестовую длину, в тесты после каждой законченной инструкции вставляется строка разреза. Во время прогона по этим строкам осуществляется разрез теста в случае, если его длина превышает допустимое ограничение. После прохождения части теста до строки разреза продолжается выполнение теста с первой инструкции, следующей за строкой разреза. Нарезку и сам прогон тестов осуществляет прогонщик тестов.

В качестве прогонщика тестов мы используем Rational Robot, который выполняет сгенерированные наборы инструкций. В случае удачного выполнения всех инструкций выносится вердикт – тест прошёл. В противном случае, если на каком-то этапе выполнения теста, поведение системы не соответствует требованиям, Robot прекращает его выполнение, вынося соответствующий вердикт – тест не прошёл.

Задание 2:

1. Выполнить тестовый набор л
2. Проанализировать отчёт о прохождении тестов.

Итог работы: отчет, защита работы.

Практическая работа № 30

Цель: получение практических навыков разработки тестов на основе внешней спецификации программы

Задание 1: ознакомиться с материалом

Программа в случае тестирования с управлением по данным рассматривается как "черный яшик", и целью тестирования является выяснение обстоятельств, в которых поведение программы не соответствует спецификации. Различают следующие методы формирования тестовых наборов:

- эквивалентное разбиение;
- анализ граничных значений;
- анализ причинно-следственных связей;
- предположение об ошибке.

Эквивалентное разбиение.

Область всех возможных наборов входных данных программы по каждому параметру разбивают на конечное число групп - ***классов эквивалентности***. Наборы данных такого класса объединяют по принципу обнаружения одних и тех же ошибок. Для составления классов эквивалентности нужно перебрать ограничения, установленные для каждого входного значения в техническом задании или при уточнении спецификации. Каждое ограничение разбивают на две или более групп.

Граничные значения.

Граничные значения - это значения на границах классов эквивалентности входных значений или около них.

Анализ причинно-следственных связей.

Метод ***анализа причинно-следственных связей*** позволяет системно выбирать тесты, используя алгебру логики. *Причиной* называют отдельное входное условие или класс эквивалентности. *Следствием* - выходное условие или преобразование системы. Идея заключается в отнесении всех следствий к причинам, то есть в уточнении причинно-следственных связей.

Предположение об ошибке.

Метод основан на интуиции программиста с большим опытом работы. Составляется список, в котором перечисляются возможные ошибки или ситуации, в которых они могут появиться, а затем на основе списка составляются тесты.

Задание 2:

На основе внешней спецификации задачи составить набор тестов на основе подхода черного ящика

Произвести тестирование программы

Итог работы: отчет, защита работы.

Практическая работа № 31

Цель: получение практических навыков отладки программ с помощью отладчика среды программирования

Задание 1: ознакомиться с материалом

Отладка — это процесс определения и устранения причин ошибок. Этим она отличается от тестирования, направленного на обнаружение ошибок. В некоторых проектах отладка занимает до 50% общего времени разработки. Многие программисты считают отладку самым трудным аспектом программирования.

Для сокращения времени отладки необходимо пользоваться научным подходом.

Классический научный подход включает следующие этапы:

1. Сбор данных при помощи повторяющихся экспериментов.
2. Формулирование гипотезы, объясняющей релевантные данные.
3. Разработка эксперимента, призванного подтвердить или опровергнуть гипотезу.
4. Подтверждение или опровержение гипотезы.
5. Повторение процесса в случае надобности.

Эффективный метод поиска дефектов при отладке с использованием научного подхода может быть описан следующими шагами:

1. Стабилизация ошибки.
2. Определение источника ошибки.

a. Сбор данных, приводяших к дефекту.

b. Анализ собранных данных и формулирование гипотезы, объясняющей дефект.

c. Определение способа подтверждения или опровержения гипотезы, основанного или на тестировании программы, или на изучении кода.

d. Подтверждение или опровержение гипотезы при помощи процедуры, определенной в п. 2(c).

3. Исправление дефекта.
4. Тестирование исправления.
5. Поиск похожих ошибок.

Способ подтверждения или опровержения гипотезы может быть одним из следующего списка:

1. сокращение подозрительной области кода;
2. проверка классов и методов, в которых дефекты обнаруживались ранее;
3. проверка кода, который изменялся недавно.

Отладка — это тот этап разработки программы, от которого зависит возможность се выпуска. Конечно, лучше всего вообще избегать ошибок. Однако потратить время на улучшение навыков отладки все же стоит, потому что эффективность отладки, выполняемой лучшими и худшими программистами, различается минимум в 10 раз.

Систематичный подход к поиску и исправлению ошибок — непременное условие успешности отладки. Организуйте отладку так, чтобы каждый тест приближал вас к цели. Используйте Научный Метод Отладки.

Прежде чем приступать к исправлению программы, поймите суть проблемы. Случайные предположения о причинах ошибок и случайные исправления только ухудшат программу.

Установите в настройках компилятора самый строгий уровень диагностики и устраняйте причины всех ошибок и предупреждений.

Инструменты отладки значительно облегчают разработку ПО. Найдите их и используйте. Большинство современных сред программирования (Delphi, C++ Builder, Visual Studio и т.д.) включают средства отладки, которые обеспечивают максимально эффективную отладку. Они позволяют:

- выполнять программу по шагам, причем как с заходом в подпрограммы, так и выполняя их целиком;
- предусматривать точки останова;
- выполнять программу до оператора, указанного курсором;
- отображать содержимое любых переменных при пошаговом выполнении;
- отслеживать поток сообщений и т.п.

Задание 2:

1. Составить в виде блок-схемы алгоритм решения задачи.
2. Создать программу решения задачи на любом алгоритмическом языке программирования.
3. Отладить программу с использованием инструментальных средств.

Задача: Имеется матрица размера N*M. Определить в какой строке количество положительных элементов наибольшее.

Контрольные вопросы.

1. Что такое тестирование программы?
2. Что такое отладка программы?
3. Какие стадии тестирования выделяют при разработке программного обеспечения?
4. Какие различают подходы в формировании тестовых наборов?
5. В чем суть тестирования методом “покрытия операторов”?
6. В чем суть тестирования методом “покрытия решений”?
7. В чем суть тестирования методом “покрытия условий”?
8. В чем суть тестирования методом “комбинаторного покрытия условий”?
9. В чём суть метода эквивалентных разбиений?
10. В чём суть метода анализа граничных значений?
11. В чём суть метода анализа причинно-следственных связей?

Итог работы: отчет, защита работы.

Раздел 3. Моделирование в программных системах

Практическая работа № 1

Цель: закрепить практические навыки по построению простейших математических и простейших статистических моделей

Задание 1: ознакомиться с материалом

Построение математической модели процесса, явления или объекта начинается с построения упрощенного варианта модели, в котором учитываются только основные черты. В результате прослеживаются основные связи между входными параметрами, ограничениями и показателем эффективности. Общего подхода к построению модели нет. В каждом конкретном случае при построении математической модели учитывается большое количество факторов: цель построения модели, круг решаемых задач, точность описания модели и точность выполнения вычислений. Математическая модель должна отражать все существенные факторы, определяющие ее поведение, и при этом быть простой и удобной для восприятия результатов. Каждая математическая модель процесса, явления или объекта в своей основе имеет математический количественный метод.

Применение математических количественных методов для обоснования выбора того или иного управляющего решения во всех областях человеческой деятельности называется *исследованием операций*. Целью исследования операций является нахождение с использованием специального математического аппарата решения, удовлетворяющего заданным условиям. На самом деле при решении практически любой задачи имеется неограниченное количество решений. Множество решений, удовлетворяющих заданным условиям (ограничениям), называется допустимым множеством решением. Выбор из множества допустимых решений одного решения, наилучшего в каком-либо смысле, называемого *оптимальным* решением, и есть задача исследования операций.

Модель — это материальный или идеальный объект, замещающий оригинал, наделенный основными характеристиками (чертами) оригинала и предназначенный для проведения некоторых действий над ним с целью получения новых сведений об оригинале.

Рис. 1. Классификация моделей

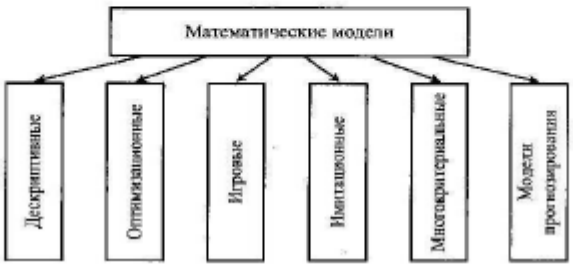


Рисунок 1. Классификация математических моделей

Рисунок 1. Классификация математических моделей

При построении математической модели необходимо обеспечить *достаточную* точность вычислений (точность решения) и *необходимую* подробность модели. Любая математическая модель включает в себя описание основных, т. е. *необходимых* для исследования свойств и законов функционирования исследуемого объекта, процесса или явления. В своей основе каждая математическая модель имеет целевую функцию, которая описывает функционирование реального объекта, процесса или явления. В зависимости от исследуемого (моделируемого) объекта, явления или процесса *целевая функция* может быть представлена одной функциональной зависимостью, системой уравнений (линейных, нелинейных, дифференциальных и т. д.), набором статистических данных и т. д. При работе с целевой функцией исследователь воздействует на нее через ***набор входных параметров*** (рис. 2).

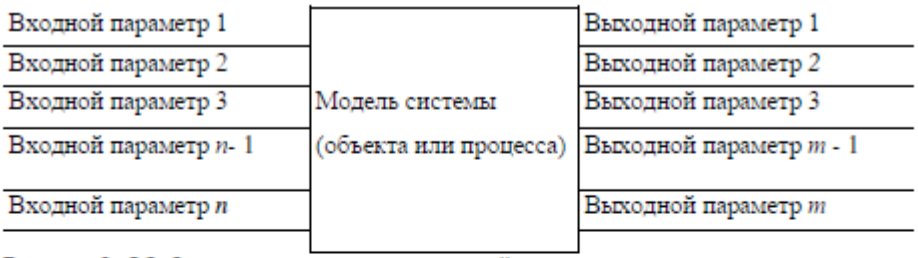


Рисунок 2. Обобщенная схема математической модели

По способу реализации математические модели можно разделить следующим образом.

1. Линейное программирование.

Математическая модель целиком (целевая функция и ограничения) описывается уравнениями первого порядка. Линейное программирование включает в себя несколько методов решения (задач):

- симплексный;
- графический;
- транспортная задача;
- целочисленное программирование.

2. Нелинейное программирование.

Целевая функция и ограничения, составляющие математическую модель, содержат хотя бы одно нелинейное уравнение (уравнение второго порядка и выше). Нелинейное программирование содержит несколько методов решения (задач):

- графический;
- регулярного симплекса;
- деформируемого многогранника (Нелдера - Мида);
- градиентный.

3. Динамическое программирование.

Ориентировано на решение задач прокладки магистралей кратчайшим путем и перераспределения различных видов ресурсов.

4. Сетевое планирование.

Решает проблему построения графика выполнения работ, распределения производственных, финансовых и людских ресурсов.

5. Принятие решений и элементы планирования.

В этом случае и качестве целевой функции выступает набор статистических данных или некоторые данные прогноза. Решением задачи являются рекомендации о способах поведения (стратегии). Решение носит рекомендательный характер (приблизительное решение). Выбор стратегии целиком остается за человеком — ответственным лицом, принимающим решение. Для принятия решения разработаны следующие теории:

- теория игр;
- системы массового обслуживания.

Задание 2:

Задание 1. Составить математическую модель следующей задачи. На складе имеется 300 кг сырья. Надо изготовить два вида продукции. На изготовление первого изделия требуется 2 кг сырья, а на изготовление второго изделия — 5 кг. Определить план выпуска двух изделий.

Решение.

Обозначим, x1 – единица первого изделия, x2 – единица второго изделия. Тогда составим математическая модель: 2x1+5x2=300.

Задание 2. Составить математическую модель следующей задачи. Предположим, что для производства продукции вида А и В можно использовать материал 3-х сортов. При этом на изготовление единицы изделия вида А расходуется 14 кг первого сорта, 12 кг второго сорта и 8 кг третьего сорта. На изготовление продукции вида В расходуется 8 кг первого сорта, 4 кг второго сорта, 2 кг третьего сорта. На складе фабрики имеется всего материала первого сорта 624 кг, второго сорта 541 кг, третьего сорта 376 кг. От реализации единицы готовой продукции вида А фабрика имеет прибыль вида 7 руб., а от реализации единицы готовой продукции вида В фабрика имеет прибыль вида 3 руб. Определить максимальную прибыль от реализации всей продукции видов А и В.

Решение.

Составим математическую модель задачи:

Пусть x_1 – единица готовой продукции вида А,

x_2 - единица готовой продукции вида В,

Цель фабрики получить максимальную прибыль от реализации всей продукции видов А и В, тогда:

$$F = 7 \cdot x_1 + 3 \cdot x_2 \to \max$$

Система ограничений:

$$\begin{cases} 14x_1 + 8x_2 \le 624 \\ 12x_1 + 4x_2 \le 541 \\ 8x_1 + 2x_2 \le 376 \end{cases}$$

$x_1 \ge 0, \quad x_2 \ge 0$ *условие неотрицательности*

Задание 3. Составить математическую модель следующей задачи. Имеются три пункта поставки однородного груза А1, А2, А3 и пять пунктов В1, В2, В3, В4, В5 потребления этого груза. На пунктах А1, А2 и А3 находится груз соответственно в количестве 200, 450, 250 тонн. В пункты В1, В2, В3, В4, В5 требуется доставить соответственно 100, 125, 325, 250, 100 тонн груза. Расстояние между пунктами поставки и пунктами потребления приведено в таблице:

Пункты поставки	**Пункты потребления**				
	B1	**B2**	**B3**	**B4**	**B5**
A1	5	8	7	10	3
A2	4	2	2	5	6
A3	7	3	5	9	2

Решение:

1. Проверка сбалансированности модели задачи. Модель является сбалансированной, т.к. суммарный объем запасов сырья равен суммарному объему потребности в ней:
   200+450+250=100+125+325+250+100.
2. Построение математической модели – неизвестными в этой задачи является объем перевозок. Пусть x_{ij} - объем перевозок с *i*-го предприятия в *j*-го пункт потребления. Суммарные транспортные расходы - это функционал качества (критерий цели): $F = \sum_{i=1}^{n} \sum_{j=1}^{m} c_{ij} x_{ij}$,

Где c_{ij} - стоимость перевозки единицы продукции с *i*-го предприятия в *j*-й пунктах потребления.

Неизвестные в этой задачи должны удовлетворять следующим ограничениям:

- Объем перевозок не могут быть отрицательными;
- Поскольку модель сбалансирована, то вся продукция должна быть вывезена с предприятия, а потребность всех пунктов потребления должна быть полностью удовлетворены.

Итак, имеем следующую задачу:

- Найти минимум функционала: $F = \sum_{i=1}^{4} \sum_{j=1}^{5} c_{ij} x_{ij} \to \min,$

- При ограничениях: $\begin{cases} \sum_{i=1}^{4} x_{ij} = 100, \\ \sum_{i=1}^{4} x_{ij} = 125, \\ \sum_{i=1}^{4} x_{ij} = 325, \\ \sum_{i=1}^{4} x_{ij} = 250, \\ \sum_{i=1}^{4} x_{ij} = 100, \end{cases} \quad \begin{cases} \sum_{j=1}^{5} x_{ij} = 200, \\ \sum_{j=1}^{5} x_{ij} = 450, \\ \sum_{j=1}^{5} x_{ij} = 250, \end{cases} \quad x_{ij} \geq 0, i \in [1,3], j \in [1,5],$

Задания для самостоятельной работы

1 вариант.

Задача 1. Составить математическую модель следующей задачи. Предположим, что для производства продукции вида А и В можно использовать материал трех сортов. При этом на изготовление единицы изделия вида А расходуется ***a1*** кг первого сорта, ***a2*** кг второго сорта и ***a3*** кг третьего сорта. На изготовление продукции вида В расходуется ***b1*** кг первого сорта, ***b2*** кг второго сорта, ***b3*** кг третьего сорта. На складе фабрики имеется всего материала первого сорта ***c1*** кг, второго сорта ***c2*** кг, третьего сорта ***c3*** кг. От реализации единицы готовой продукции вида А фабрика имеет прибыль вида α руб., а от реализации единицы готовой продукции вида В фабрика имеет прибыль вида β руб. Определить максимальную прибыль от реализации всей продукции видов А и В.

a1= 19, a2= 16, a3= 19, b1= 26, b2= 17, b3= 8, c1 = 868, c2= 638, c3= 853,
α=5, β=4.

Задача 2. Имеются три пункта поставки однородного груза **A1, A2, A3** и пять пунктов **B1, B2, B3, B4, B5** потребления этого груза. На пунктах **A1, A2 и A3** находится груз соответственно в количестве ***a1, a2 и a3*** тонн. В пункты **B1, B2, B3, B4, B5** требуется доставить соответственно ***b1, b2, b3, b4, b5*** тонн груза. Расстояние между пунктами поставки и пунктами потребления приведено в таблице:

Пункты поставки	**Пункты потребления**				
	B1	**B2**	**B3**	**B4**	**B5**
A1	D11	D12	D13	D14	D15
A2	D21	D22	D23	D24	D25
A3	D31	D32	D33	D34	D35

Найти такой план закрепления потребителей за поставщиками однородного груза, чтобы общие затраты по перевозкам были минимальными.

a1=300, a2=250, a3=200,
b1=210, b2=150, b3=120, b4=135, b5=135.

$$D = \begin{pmatrix} 4 & 8 & 13 & 2 & 7 \\ 9 & 4 & 11 & 9 & 17 \\ 3 & 16 & 10 & 1 & 4 \end{pmatrix}$$

2 вариант.

Задача 1. Составить математическую модель следующей задачи. Предположим, что для производства продукции вида А и В можно использовать материал трех сортов. При этом на изготовление единицы изделия вида А расходуется ***a1*** кг первого сорта, ***a2*** кг второго сорта и ***a3*** кг третьего сорта. На изготовление продукции вида В расходуется ***b1*** кг первого сорта, ***b2*** кг второго сорта, ***b3*** кг третьего сорта. На складе фабрики имеется всего материала первого сорта ***c1*** кг, второго сорта ***c2*** кг, третьего сорта ***c3*** кг. От реализации единицы готовой продукции вида А фабрика имеет прибыль вида α руб., а от реализации единицы готовой продукции вида В фабрика имеет прибыль вида β руб. Определить максимальную прибыль от реализации всей продукции видов А и В.

a1= 14, a2= 15, a3= 20, b1= 40, b2= 27, b3= 4, c1 = 1200, c2= 993, c3= 1097,
α=5, β=13.

Задача 2. Имеются три пункта поставки однородного груза **A1, A2, A3** и пять пунктов **B1, B2, B3, B4, B5** потребления этого груза. На пунктах **A1, A2 и A3** находится груз соответственно в количестве ***a1, a2 и a3*** тонн. В пункты **B1, B2, B3, B4, B5** требуется доставить соответственно ***b1, b2, b3, b4, b5*** тонн груза. Расстояние между пунктами поставки и пунктами потребления приведено в таблице:

Пункты поставки	**Пункты потребления**				
	B1	**B2**	**B3**	**B4**	**B5**
A1	D11	D12	D13	D14	D15
A2	D21	D22	D23	D24	D25
A3	D31	D32	D33	D34	D35

Найти такой план закрепления потребителей за поставщиками однородного груза, чтобы общие затраты по перевозкам были минимальными.

a1=350, a2=200, a3=300,

b1=170, b2=140, b3=200, b4=195, b5=145.

$$D = \begin{pmatrix} 22 & 14 & 16 & 28 & 30 \\ 19 & 17 & 26 & 36 & 36 \\ 37 & 30 & 31 & 39 & 41 \end{pmatrix}$$

3 вариант.

Задача 1. Составить математическую модель следующей задачи. Предположим, что для производства продукции вида А и В можно использовать материал трех сортов. При этом на изготовление единицы изделия вида А расходуется ***a1*** кг первого сорта, ***a2*** кг второго сорта и ***a3*** кг третьего сорта. На изготовление продукции вида В расходуется ***b1*** кг первого сорта, ***b2*** кг второго сорта, ***b3*** кг третьего сорта. На складе фабрики имеется всего материала первого сорта ***c1*** кг, второго сорта ***c2*** кг, третьего сорта ***c3*** кг. От реализации единицы готовой продукции вида А фабрика имеет прибыль вида α руб., а от реализации единицы готовой продукции вида В фабрика имеет прибыль вида β руб. Определить максимальную прибыль от реализации всей продукции видов А и В.

a1= 9, a2= 15, a3= 15, b1= 27, b2= 15, b3= 3, c1= 606, c2= 802, c3= 840,

α=11, β=6.

Задача 2. Имеются три пункта поставки однородного груза **A1, A2, A3** и пять пунктов **B1, B2, B3, B4, B5** потребления этого груза. На пунктах **A1, A2 и A3** находится груз соответственно в количестве ***a1, a2 и a3*** тонн. В пункты **B1, B2, B3, B4, B5** требуется доставить соответственно ***b1, b2, b3, b4, b5*** тонн груза. Расстояние между пунктами поставки и пунктами потребления приведено в таблице:

Пункты поставки	**Пункты потребления**				
	B1	**B2**	**B3**	**B4**	**B5**
A1	D11	D12	D13	D14	D15
A2	D21	D22	D23	D24	D25
A3	D31	D32	D33	D34	D35

Найти такой план закрепления потребителей за поставщиками однородного груза, чтобы общие затраты по перевозкам были минимальными.

a1=200, a2=250, a3=200,

b1=190, b2=100, b3=120, b4=110, b5=130.

$$D = \begin{pmatrix} 28 & 27 & 18 & 27 & 24 \\ 18 & 26 & 27 & 32 & 21 \\ 27 & 33 & 23 & 31 & 34 \end{pmatrix}$$

4 вариант.

Задача 1. Составить математическую модель следующей задачи. Предположим, что для производства продукции вида А и В можно использовать материал трех сортов. При этом на изготовление единицы изделия вида А расходуется ***a1*** кг первого сорта, ***a2*** кг второго сорта и ***a3*** кг третьего сорта. На изготовление продукции вида В расходуется ***b1*** кг первого сорта, ***b2*** кг второго сорта, ***b3*** кг третьего сорта. На складе фабрики имеется всего материала первого сорта ***c1*** кг, второго сорта ***c2*** кг, третьего сорта ***c3*** кг. От реализации единицы готовой продукции вида А фабрика имеет прибыль вида α руб., а от реализации единицы готовой продукции вида В фабрика

имеет прибыль вида β руб. Определить максимальную прибыль от реализации всей продукции видов А и В.

a1= 13, a2= 13, a3= 11, b1= 23, b2= 11, b3= 1, c1= 608, c2= 614, c3= 575,
α=5, β=7.

Задача 2. Имеются три пункта поставки однородного груза **A1, A2, A3** и пять пунктов **B1, B2, B3, B4, B5** потребления этого груза. На пунктах **A1, A2 и A3** находится груз соответственно в количестве *a1, a2 и a3* тонн. В пункты **B1, B2, B3, B4, B5** требуется доставить соответственно *b1, b2, b3, b4, b5* тонн груза. Расстояние между пунктами поставки и пунктами потребления приведено в таблице:

Пункты поставки	**Пункты потребления**				
	B1	**B2**	**B3**	**B4**	**B5**
A1	D11	D12	D13	D14	D15
A2	D21	D22	D23	D24	D25
A3	D31	D32	D33	D34	D35

Найти такой план закрепления потребителей за поставщиками однородного груза, чтобы общие затраты по перевозкам были минимальными.

a1=230, a2=250, a3=170,
b1=140, b2=90, b3=160, b4=110, b5=150.

$$D = \begin{pmatrix} 40 & 19 & 25 & 25 & 35 \\ 49 & 26 & 27 & 18 & 38 \\ 46 & 27 & 36 & 40 & 45 \end{pmatrix}$$

5 вариант.

Задача 1. Составить математическую модель следуюшей задачи. Предположим, что для производства продукции вида А и В можно использовать материал трех сортов. При этом на изготовление единицы изделия вида А расходуется *a1* кг первого сорта, *a2* кг второго сорта и *a3* кг третьего сорта. На изготовление продукции вида В расходуется *b1* кг первого сорта, *b2* кг второго сорта, *b3* кг третьего сорта. На складе фабрики имеется всего материала первого сорта *c1* кг, второго сорта *c2* кг, третьего сорта *c3* кг. От реализации единицы готовой продукции вида А фабрика имеет прибыль вида α руб., а от реализации единицы готовой продукции вида В фабрика имеет прибыль вида β руб. Определить максимальную прибыль от реализации всей продукции видов А и В.

a1= 19, a2= 16, a3= 19, b1= 31, b2= 9, b3= 1, c1= 1121, c2= 706, c3= 1066,
α=16, β=19.

Задача 2. Имеются три пункта поставки однородного груза **A1, A2, A3** и пять пунктов **B1, B2, B3, B4, B5** потребления этого груза. На пунктах **A1, A2 и A3** находится груз соответственно в количестве *a1, a2 и a3* тонн. В пункты **B1, B2, B3, B4, B5** требуется доставить соответственно *b1, b2, b3, b4, b5* тонн груза. Расстояние между пунктами поставки и пунктами потребления приведено в таблице:

Пункты поставки	**Пункты потребления**				
	B1	**B2**	**B3**	**B4**	**B5**
A1	D11	D12	D13	D14	D15
A2	D21	D22	D23	D24	D25
A3	D31	D32	D33	D34	D35

Найти такой план закрепления потребителей за поставщиками однородного груза, чтобы общие затраты по перевозкам были минимальными.

a1=200, a2=300, a3=250,
b1=210, b2=150, b3=120, b4=135, b5=135.

$$D = \begin{pmatrix} 20 & 10 & 13 & 13 & 18 \\ 27 & 19 & 20 & 16 & 22 \end{pmatrix}$$

Итог работы: отчет, защита работы.

Практическая работа № 2

Цель: закрепить практические навыки по построению простейших математических и простейших статистических моделей

Задание 1: разработать задачи на построение математических и статистических моделей (не менее 4), порешать их

Итог работы: отчет, защита работы.

Практическая работа №3

Цель: овладеть навыками вычисления точечных и интервальных оценок параметров распределения.

Задание 1:

Вариант 1

1. Вычислить точечные оценки для дискретного вариационного ряда

2	4	5	7	15
1	5	6	2	3

2. Вычислить точечные оценки для интервального вариационного ряда

-4 -2	-2 0	0 2	2 4	4 6
3	5	2	2	6

3. Найти доверительный интервал для оценки с надежностью 0,95 неизвестного математического ожидания a нормально распределенного признака X генеральной совокупности, если генеральное среднеквадратичное отклонение равно 5, выборочная средняя равна 14, объем выборки – 25.

4. Решить задачу 3 при условии, что средне квадратичное отклонение неизвестно.

Вариант 2

1. Вычислить точечные оценки для дискретного вариационного ряда

1	3	5	7	12
3	5	1	2	4

2. Вычислить точечные оценки для интервального вариационного ряда

-4 -1	-1 2	2 5	5 8	8 11
2	1	5	3	6

3. Найти доверительный интервал для оценки с надежностью 0,98 неизвестного математического ожидания a нормально распределенного признака X генеральной совокупности, если генеральное среднеквадратичное отклонение равно 4, выборочная средняя равна 12, объем выборки – 30.

4. Решить задачу 3 при условии, что средне квадратичное отклонение неизвестно.

Вариант 3

1. Вычислить точечные оценки для дискретного вариационного ряда

2	3	5	6	8
2	3	4	2	5

2. Вычислить точечные оценки для интервального вариационного ряда

-5 -2	-2 1	1 4	4 7	7 10
1	3	2	4	1

3. Найти доверительный интервал для оценки с надежностью 0,85 неизвестного математического ожидания a нормально распределенного признака X генеральной совокупности, если генеральное среднеквадратичное отклонение равно 4, выборочная средняя равна 15, объем выборки – 35.

4. Решить задачу 3 при условии, что средне квадратичное отклонение неизвестно.

Вариант 4

1. Вычислить точечные оценки для дискретного вариационного ряда

2	3	5	7	9
2	1	2	8	3

2. Вычислить точечные оценки для интервального вариационного ряда

-4 -2	-2 0	0 2	2 4	4 6
1	3	7	2	2

3. Найти доверительный интервал для оценки с надежностью 0,8 неизвестного математического ожидания a нормально распределенного признака X генеральной совокупности, если генеральное среднеквадратичное отклонение равно 7, выборочная средняя равна 16, объем выборки – 30.

4. Решить задачу3 при условии, что средне квадратичное отклонение неизвестно

Вариант 5

1. Вычислить точечные оценки для дискретного вариационного ряда

-1	0	1	3	4
10	1	6	3	3

2. Вычислить точечные оценки для интервального вариационного ряда

-4 -2	-2 0	0 2	2 4	4 6
3	5	2	2	6

3. Найти доверительный интервал для оценки с надежностью 0,9 неизвестного математического ожидания a нормально распределенного признака X генеральной совокупности, если генеральное среднеквадратичное отклонение равно 5, выборочная средняя равна 14, объем выборки – 25.

4. Решить задачу 3 при условии, что средне квадратичное отклонение неизвестно.

Вариант 6

1. Вычислить точечные оценки для дискретного вариационного ряда

-2	-1	0	2	3
4	1	1	2	4

2. Вычислить точечные оценки для интервального вариационного ряда

-4 -2	-2 0	0 2	2 4	4 6
1	4	1	2	4

3. Найти доверительный интервал для оценки с надежностью 0,9 неизвестного математического ожидания a нормально распределенного признака X генеральной совокупности, если генеральное среднеквадратичное отклонение равно 6, выборочная средняя равна 19, объем выборки – 35.

4. Решить задачу 3 при условии, что средне квадратичное отклонение неизвестно.

Итог работы: отчет, защита работы.

Практическая работа № 4

Цель: закрепить практические навыки по отработки простейших навыков решения однокритериальных задач

Задание 1: ознакомиться с материалом

В зависимости от вида показателя эффективности различают задачи принятия решений по скалярному показателю (однокритериальные задачи) и задачи принятия решений по векторному показателю (многокритериальные задачи).

Задачами **математического программирования** называют **однокритериальные задачи оптимизации**. Методы их решения оперируют с детерминированными математическими моделями. В этих моделях отражены разнообразные проблемы распределения ограниченных ресурсов в экономике, военном деле, создании новой техники и т.д. Пути решения этих проблем, так или иначе, связаны с планированием целенаправленной деятельности, т.е. с разработкой определенных установок на будущее.

Задача математического программирования формулируется следующим образом: найти значения переменных $x_1, x_2, \ldots, x_n$, доставляющие максимум (минимум) заданной целевой функции $y = f(x_1, x_2, \ldots, x_n)$ при условиях:

$$g_j(x_1, x_2, \ldots, x_n) \leq (\geq, =) b_j, (j = \overline{1, m}).$$

Различают два вида задач математического программирования:

1. Задачи линейного программирования.
2. Задачи нелинейного программирования.

В первых задачах функция y и ограничения g_j линейны относительно переменных x. Во вторых задачах целевая функция y и (или) условия g_j имеют разного рода нелинейности.

Графоаналитический метод решения задач оптимизации

Этим методом вручную решаются простые задачи оптимизации. Математические модели в этих задачах не должны быть сложными, т.к. в противном случае требуется много времени для их решения. Для начала рассмотрим однопараметрическую однокритериальную задачу оптимизации.

Постановка задачи: Дан один критерий y. Объект (процесс) описан уравнением (уравнениями), включающими один искомый параметр $y = f(x)$. Имеется система ограничений:

$1) x \geq a_1;$

$2) a_2 \leq x \leq b_1;$

и т.д.

Необходимо найти оптимальное значение параметра $x = x_{опт}$, обращающее целевую функцию $f(x)$ в максимум или минимум.

Задача решается в два этапа:

1. Построение области допустимых решений (ОДР).
2. Нахождение в пределах ОДР оптимального решения.

При построении ОДР на первом этапе рассматривается система ограничений. Все ограничения должны быть выполнены. Выполнение первого ограничения в приведенной выше постановке задачи оптимизации означает, что искомое значение параметра x должно находиться правее a_1, причем, a_1 в разрешенный интервал входит (рис.1). Выполнение второго ограничения означает, что искомое значение параметра x должно находиться в интервале (на отрезке) $[a_2 b_1]$, следует иметь в виду, что границы интервала в интервал входят.

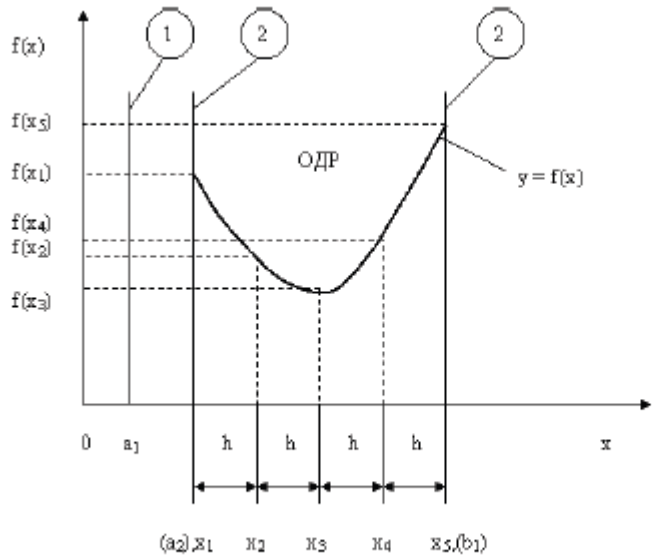


Рис.1. Графическая иллюстрация решения однопараметрической однокритериальной задачи оптимизации

Когда однопараметрическая однокритериальная задача оптимизации решается с применением графоаналитического метода вручную, то на втором этапе применяют метод перебора. Суть его заключается в следующем. В пределах ОДР через определенный интервал h выбирается ряд значений параметра x. В рассматриваемом нами случае ОДР разбита на четыре отрезка, и выбрано пять значений параметра x. Для этих значений параметра x рассчитываются соответствующие значения целевой функции. Среди них находят минимальное (максимальное) значение. Значение параметра x_i, обращающее целевую функцию в минимум (максимум), является оптимальным. Если в рассматриваемом нами случае $f(x)$ стремится к минимуму, то $x_{опт} = x_3$, если к максимуму, то $x_{опт} = x_5$.

При решении практических задач оптимизации всегда следует иметь в виду, какова целевая функция. Это значительно упрощает работу как при решении задач оптимизации вручную с применением графоаналитического метода, так и при решении таких задач с использованием компьютерных программ. Причем, это относится и к случаю использования готовых программ, и, что особенно важно, к разработке собственных программ.

Рассмотрим, например, следующий частный случай, когда целевая функция линейная (рис.2.).

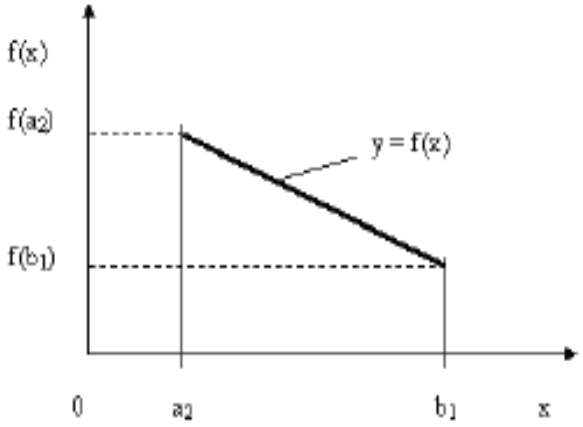


Рис.2. Графическая иллюстрация решения однопараметрической однокритериальной задачи оптимизации для случая линейной целевой функции

В данном случае на втором этапе вычисляют значения целевой функции только на границах ОДР. Эти значения сравнивают и выбирают наименьшее или наибольшее. Для примера, приведенного на рис. 2, если $f(x) \to \min$, то $x_{опт} = b_1$, если $f(x) \to \max$, то $x_{опт} = a_2$.

В задачах, как правило, присутствует не один, а несколько признаков предпочтения (критериев). Такие задачи называются **многокритериальными.** Критерии могут оказаться противоречивыми, т.е. решение, лучшее по определенному признаку, может оказаться худшим по другому признаку.

Например, минимизация стоимости и максимизация качества товара почти всегда противоречивы. В этом случае задача отыскания решения, предпочтительного по всем признакам, будет ***некорректной***, т.е. не будет иметь ни одного решения.

В случае противоречивых критериев имеются следующие подходы к отысканию подходящего решения.

1) **Замена некоторых критериев ограничениями** вида $\leq$ или $\geq$. Например, минимизация стоимости $f(x) \rightarrow \min$, может быть заменена ограничением вида $f(x) \leq A$, где A некоторая верхняя оценка стоимости, т.е. максимально допустимая стоимость.

2)**Свертка критериев.** Создается один глобальный скалярный критерий, целевая функция которого является некоторой функцией от исходных целевых функций. Наиболее употребимыми являются линейные свертки вида $\alpha f(x) + \beta g(x)$ (в случае двух критериев). Нетривиальной является задача отыскания адекватных значений коэффициентов α и β, отражающих относительную важность целевых функций $f(x)$ и $g(x)$.

3) **Ранжирование критериев.** Критерии ранжируются по степени важности.

4) **Отыскание решений, лучших хотя бы по одному критерию.**

Подходы 1) и 2) приводят к **однокритериальной задаче**.

Подход 3) приводит к ***задаче с упорядоченными критериями.***

Подход 4) приводит к ***задаче с независимыми критериями.*** В задаче с упорядоченными критериями критерии упорядочиваются по важности, и требуется найти оптимальное решение для наименее важного критерия на множестве решений, оптимальных для более важного критерия (см. рис 3). Самое большое множество

Рис 3. Множество решений.

всех допустимых решений, в него вложено множество решений, оптимальных по самому важному критерию, далее вложено множество оптимальных решений по второму по важности критерию, и т.д.

В задаче с независимыми критериями требуется найти множество ***недоминируемых (эффективных) решений.*** **Недоминируемое решение лучше любого другого допустимого решения хотя бы по одному критерию либо не хуже по всем критериям.**

Множество недоминируемых решений также называется ***множеством Парето.***

Задание 2:

Задание 1. Решить графическим способом задачу. Для производства двух видов, изделии P_1 и P_2 используется, три вида сырья S_1, S_2, S_3, запасы которого соответственно равны 100, 60, 180 единиц. Для производства одной единицы продукции P_1 используется 2 единицы сырья S_1 и по 1 единице сырья $S_2 и S_3$. Для производства одной единицы продукции P_2 используется по 1 единице сырья $S_1 и S_2$ и 4 единицы сырья S_3. Прибыль от реализации 1 единицы каждой продукции P_1 и P_2 соответственно равна 30 и 20 единиц. Необходимо составить такой план выпуска продукции P_1 и P_2, при котором суммарная прибыль будет наибольшей.

Решение.

Составим математическую модель задачи:

Пусть x_1 – единица готовой продукции вида P_1,

x_2 - единица готовой продукции вида P_2,

Цель фабрики получить максимальную прибыль от реализации всей продукции видов P_1 и P_2, тогда:

$$F = 30 \cdot x_1 + 20 \cdot x_2 \to \max$$

Система ограничений:

$$\begin{cases} 2x_1 + x_2 \le 100 \\ x_1 + x_2 \le 60 \\ x_1 + 4x_2 \le 180 \end{cases}$$

$x_1 \ge 0, \quad x_2 \ge 0$ *условие неотрицательности*

Алгоритм решения:

1. Используя систему ограничений и условия неотрицательности, строим область допустимых решений.
2. Строим линию уровня $F = 0$. Линией уровня функции двух переменных называется линия, вдоль которой функция сохраняет свое постоянное значение.
3. Строим градиент целевой функции. Градиент функции - это вектор, имеющий своими координатами частные производные функции и показывающий направление наискорейшего роста значения функции. Так как целевая функция ЗЛП линейная, то линии уровня целевой функции - прямые и $\overline{gradF} = \overline{n}$, n- вектор нормали к этим прямым. :
.
4. Перемещаем линию уровня $F = 0$ вдоль градиента функции. Если ЗЛП на минимум, то оптимальное решение находится в первой точке, принадлежащей ОДР; если ЗЛП на максимум, то оптимальное решения находится в последней точке, принадлежащей ОДР.

Замечание.

При построении ОДР возможны случаи:

1. ОДР оказалась пустым множеством. В этом случае ЗЛП не имеет решения из-за несовместности системы ограничений.

2. ОДР оказалась либо выпуклым многоугольником, либо не ограниченной выпуклой многоугольной областью. Тогда ЗЛП имеет оптимальное решение, которое совпадает по крайней мере с одной из вершин ОДР.

Используя алгоритм решения и систему ограничений и условия неотрицательности, построим ОДР. Для этого во всех неравенствах системы ограничений и условия неотрицательности знак неравенства заменим на знак равенства. В результате будем иметь уравнения прямых:

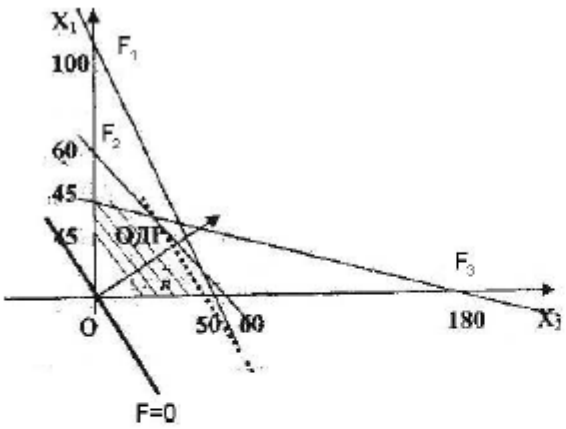
$$F_1 : 2x_1 + x_2 = 100$$

$$F_2 : x_1 + x_2 = 60$$

$$F_3 : x_1 + 4x_2 = 180$$

$$x_1 = 0, \quad x_2 = 0$$

В системе координат X_1OX_2 построим эти прямые. В результате будем иметь ОДР. В этой же системе координат строим линию уровня $F = 0$ и вектор $\overline{gradF} = \overline{n}$



Так как задача на максимум, будем перемещать линию уровня F=0 вдоль вектора *n* до тех пор, пока она не пересечет ОДР в самом крайнем своем положении, т.е. при дальнейшем перемещении она не будет с ОДР иметь общие точки. Такой точкой оказалась точка пересечения прямых F_1 и F_2.

Вычислим ее координаты.

$$\begin{cases} 2x_1 + x_2 = 100 \\ x_1 + x_2 = 60 \end{cases} \Rightarrow x_1 = 40, \quad x_2 = 20, \qquad F_{\max} = 30 \cdot 40 + 20 \cdot 20 = 1600$$

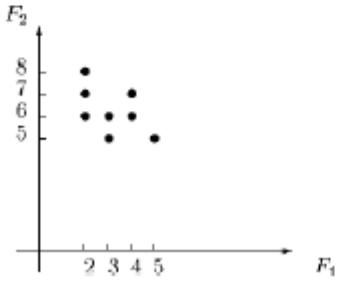
Таким образом, если предприятие будет выпускать продукцию вида P_1 и P_2, в количестве 40 и 20 единиц соответственно, то получит максимальную прибыль в размере 1600 единиц.

Задание 2. Фирме необходимо выбрать наилучший вариант закупки оборудования, если задана закупочная цена каждого из вариантов оборудования и время изготовления и доставки. Под наилучшим вариантом понимается вариант с минимальными закупочной стоимостью и временем доставки.

Обозначим, соответственно, через x_i - номер, $F_1(x_i)$ - время изготовления и доставки, $F_2(x_i)$ - закупочную стоимость варианта закупки оборудования. Значения функций $F_1(x_i)$ и $F_2(x_i)$ заданы таблицей:

x_i	1	2	3	4	5	6	7	8
$F_1(x_i)$	2	2	2	3	3	4	4	5
$F_2(x_i)$	6	7	8	5	6	6	7	5

Задача отыскания множества Парето в случае двух критериев вида $F_1(x) \to \min$ и $F_2(x) \to \min$ может быть решена графически следующим образом. Находим все точки с наименьшим значением $F_1(x)$.



Если их несколько, выбираем из них точку с наименьшим значением $F_2(x)$. Включаем ее в множество Парето. Отсекаем точки с большими либо равными значениями $F_1(x)$ и $F_2(x)$ (северо-восточный угол с вершиной в выбранной точке). Повторяем процедуру для оставшейся части допустимой области.

Из рисунка видно, что для нас представляют интерес пары $(F_1, F_2) \in \{(2,6),(3,5)\}$ и соответствующие решения $(x_1, x_2) \in \{(2,2),(1,2)\}$, которые являются недоминируемыми и образуют множество Парето рассматриваемой задачи.

Задания для самостоятельной работы

1 вариант.

Задача 1. Составить математическую модель следующей задачи. Предположим, что для производства продукции вида А и В можно использовать материал трех сортов. При этом на изготовление единицы изделия вида А расходуется ***a1*** кг первого сорта, ***a2*** кг второго сорта и ***a3*** кг третьего сорта. На изготовление продукции вида В расходуется ***b1*** кг первого сорта, ***b2*** кг второго сорта, ***b3*** кг третьего сорта. На складе фабрики имеется всего материала первого сорта ***c1*** кг, второго сорта ***c2*** кг, третьего сорта ***c3*** кг. От реализации единицы готовой продукции вида А фабрика имеет прибыль вида α руб., а от реализации единицы готовой продукции вида В фабрика имеет прибыль вида β руб. Определить максимальную прибыль от реализации всей продукции видов А и В.

a1= 19, a2= 16, a3= 19, b1= 26, b2= 17, b3= 8, c1 = 868, c2= 638, c3= 853,
α=5, β=4.

Задача 2. Фирме необходимо выбрать наилучший вариант закупки оборудования, если задана закупочная цена каждого из вариантов оборудования и время изготовления и доставки. Под наилучшим вариантом понимается вариант с минимальными закупочной стоимостью и временем доставки.

А) Для заданной двухкритериальной задачи, задавшись коэффициентами α и β провести линейную свертку критериев $F_1(x)$ и $F_2(x)$ и определить минимальное решение.

Б) Для заданной двухкритериальной задачи найти множество Парето в случае двух критериев вида $F_1(x) \to \min$ и $F_2(x) \to \min$.

Значения $F_1(x_i)$ и $F_2(x_i)$ заданы таблицей:

x_i	1	2	3	4	5	6	7	8
$F_1(x_i)$	2	2	2	3	3	4	4	5
$F_2(x_i)$	4	5	6	3	4	4	5	3

2 вариант.

Задача 1. Составить математическую модель следующей задачи. Предположим, что для производства продукции вида А и В можно использовать материал трех сортов. При этом на изготовление единицы изделия вида А расходуется ***a1*** кг первого сорта, ***a2*** кг второго сорта и ***a3*** кг третьего сорта. На изготовление продукции вида В расходуется ***b1*** кг первого сорта, ***b2*** кг второго сорта, ***b3*** кг третьего сорта. На складе фабрики имеется всего материала первого сорта ***c1*** кг, второго сорта ***c2*** кг, третьего сорта ***c3*** кг. От реализации единицы готовой продукции вида А фабрика имеет прибыль вида α руб., а от реализации единицы готовой продукции вида В фабрика имеет прибыль вида β руб. Определить максимальную прибыль от реализации всей продукции видов А и В.

a1= 14, a2= 15, a3= 20, b1= 40, b2= 27, b3= 4, c1 = 1200, c2= 995, c3= 1097,
α=5, β=13.

Задача 2. Фирме необходимо выбрать наилучший вариант закупки оборудования, если задана закупочная цена каждого из вариантов оборудования и время изготовления и доставки. Под наилучшим вариантом понимается вариант с минимальными закупочной стоимостью и временем доставки.

А) Для заданной двухкритериальной задачи, задавшись коэффициентами α и β провести линейную свертку критериев $F_1(x)$ и $F_2(x)$ и определить минимальное решение.

Б) Для заданной двухкритериальной задачи найти множество Парето в случае двух критериев вида $F_1(x) \to \min$ и $F_2(x) \to \min$.

Значения $F_1(x_i)$ и $F_2(x_i)$ заданы таблицей:

x_i	1	2	3	4	5	6	7	8
$F_1(x_i)$	3	3	3	4	4	5	5	6
$F_2(x_i)$	5	6	7	4	5	5	6	4

3 вариант.

Задача 1. Составить математическую модель следующей задачи. Предположим, что для производства продукции вида А и В можно использовать материал трех сортов. При этом на изготовление единицы изделия вида А расходуется ***a1*** кг первого сорта, ***a2*** кг второго сорта и ***a3*** кг третьего сорта. На изготовление продукции вида В расходуется ***b1*** кг первого сорта, ***b2*** кг второго сорта, ***b3*** кг третьего сорта. На складе фабрики имеется всего материала первого сорта ***c1*** кг, второго сорта ***c2*** кг, третьего сорта ***c3*** кг. От реализации единицы готовой продукции вида А фабрика имеет прибыль вида ***α*** руб., а от реализации единицы готовой продукции вида В фабрика имеет прибыль вида ***β*** руб. Определить максимальную прибыль от реализации всей продукции видов А и В.

a1= 9, a2= 15, a3= 15, b1= 27, b2= 15, b3= 3, c1= 606, c2= 802, c3= 840,
α=11, β=6.

Задача 2. Фирме необходимо выбрать наилучший вариант закупки оборудования, если задана закупочная цена каждого из вариантов оборудования и время изготовления и доставки. Под наилучшим вариантом понимается вариант с минимальными закупочной стоимостью и временем доставки.

А) Для заданной двухкритериальной задачи, задавшись коэффициентами α и β провести линейную свертку критериев $F_1(x)$ и $F_2(x)$ и определить минимальное решение.

Б) Для заданной двухкритериальной задачи найти множество Парето в случае двух критериев вида $F_1(x) \to \min$ и $F_2(x) \to \min$.

Значения $F_1(x_i)$ и $F_2(x_i)$ заданы таблицей:

x_i	1	2	3	4	5	6	7	8
$F_1(x_i)$	4	4	4	5	5	6	6	7
$F_2(x_i)$	6	7	8	5	6	6	7	5

4 вариант.

Задача 1. Составить математическую модель следующей задачи. Предположим, что для производства продукции вида А и В можно использовать материал трех сортов. При этом на изготовление единицы изделия вида А расходуется ***a1*** кг первого сорта, ***a2*** кг второго сорта и ***a3*** кг третьего сорта. На изготовление продукции вида В расходуется ***b1*** кг первого сорта, ***b2*** кг второго сорта, ***b3*** кг третьего сорта. На складе фабрики имеется всего материала первого сорта ***c1*** кг, второго сорта ***c2*** кг, третьего сорта ***c3*** кг. От реализации единицы готовой продукции вида А фабрика имеет прибыль вида ***α*** руб., а от реализации единицы готовой продукции вида В фабрика имеет прибыль вида ***β*** руб. Определить максимальную прибыль от реализации всей продукции видов А и В.

a1= 13, a2= 13, a3= 11, b1= 23, b2= 11, b3= 1, c1= 608, c2= 614, c3= 575,
α=5, β=7.

Задача 2. Фирме необходимо выбрать наилучший вариант закупки оборудования, если задана закупочная цена каждого из вариантов оборудования и время изготовления и доставки. Под наилучшим вариантом понимается вариант с минимальными закупочной стоимостью и временем доставки.

А) Для заданной двухкритериальной задачи, задавшись коэффициентами α и β провести линейную свертку критериев $F_1(x)$ и $F_2(x)$ и определить минимальное решение.

Б) Для заданной двухкритериальной задачи найти множество Парето в случае двух критериев вида $F_1(x) \to \min$ и $F_2(x) \to \min$.

Значения $F_1(x_i)$ и $F_2(x_i)$ заданы таблицей:

x_i	1	2	3	4	5	6	7	8
$F_1(x_i)$	2	2	2	3	3	4	5	5
$F_2(x_i)$	6	4	7	2	4	4	6	5

5 вариант.

Задача 1. Составить математическую модель следующей задачи. Предположим, что для производства продукции вида А и В можно использовать материал трех сортов. При этом на изготовление единицы изделия вида А расходуется ***a1*** кг первого сорта, ***a2*** кг второго сорта и ***a3*** кг третьего сорта. На изготовление продукции вида В расходуется ***b1*** кг первого сорта, ***b2*** кг второго сорта, ***b3*** кг третьего сорта. На складе фабрики имеется всего материала первого сорта ***c1*** кг, второго сорта ***c2*** кг, третьего сорта ***c3*** кг. От реализации единицы готовой продукции вида А фабрика имеет прибыль вида ***α*** руб., а от реализации единицы готовой продукции вида В фабрика имеет прибыль вида ***β*** руб. Определить максимальную прибыль от реализации всей продукции видов А и В.

a1= 19, a2= 16, a3= 19, b1= 31, b2= 9, b3= 1, c1= 1121, c2= 706, c3= 1066,
α=16, β=19.

Задача 2. Фирме необходимо выбрать наилучший вариант закупки оборудования, если задана закупочная цена каждого из вариантов оборудования и время изготовления и доставки. Под наилучшим вариантом понимается вариант с минимальными закупочной стоимостью и временем доставки.

А) Для заданной двухкритериальной задачи, задавшись коэффициентами α и β провести линейную свертку критериев $F_1(x)$ и $F_2(x)$ и определить минимальное решение.

Б) Для заданной двухкритериальной задачи найти множество Парето в случае двух критериев вида $F_1(x) \to \min$ и $F_2(x) \to \min$.

Значения $F_1(x_i)$ и $F_2(x_i)$ заданы таблицей:

x_i	1	2	3	4	5	6	7	8
$F_1(x_i)$	5	5	5	6	6	7	7	8
$F_2(x_i)$	7	8	9	6	7	7	8	6

Контрольные вопросы

1. Какие задачи называются однокритериальными?
2. Какие задачи называются многокритериальными?
3. Какие способы решения однокритериальных задач вы знаете?
4. Какие подходы к отысканию подходящего решения вы знаете у противоречивых критериев?
5. Какое множество называется множеством Парето?

Итог работы: отчет, защита работы.

Практическая работа № 5

Цель: закрепить практические навыки по отработки простейших навыков решения многокритериальных задач в оптимальных условиях

Задание 1: разработать однокритериальные многокритериальные задачи, решить их

Итог работы: отчет, защита работы.

Практическая работа № 6

Цель: закрепить практические навыки по отработки простейших навыков решения многокритериальных задач

Задание 1: ознакомиться с материалом

Многие задачи науки и техники сводятся к решению обыкновенных дифференциальных уравнений (ОДУ). ОДУ называются такие уравнения, которые содержат одну или несколько производных от искомой функции. В общем виде ОДУ можно записать следующим образом:

$F(x, y, y', y'', \ldots, y^{(n)}) = 0$, где x – независимая переменная, $y^{(i)}$ - i-ая производная от искомой функции. n - порядок уравнения. Общее решение ОДУ n–го порядка содержит n произвольных постоянных $c_1, \ldots, c_n$, т.е. общее решение имеет вид $y = \varphi(x, c_1, \ldots, c_n)$.

Для выделения единственного решения необходимо задать n дополнительных условий. В зависимости от способа задания дополнительных условий существуют два различных типа задач: задача Коши и краевая задача. Если дополнительные условия задаются в одной точке, то такая задача называется задачей Коши. Дополнительные условия в задаче Коши называются начальными условиями. Если же дополнительные условия задаются в более чем одной точке, т.е. при различных значениях независимой переменной, то такая задача называется краевой. Сами дополнительные условия называются краевыми или граничными.

Ясно, что при n=1 можно говорить только о задачи Коши.

Примеры постановки задачи Коши:

$$\frac{dy}{dx} = x^2 y^3, \quad y(1) = 1$$

$$\frac{d^2y}{dx^2} = \frac{dy}{dx} + xy^2, \quad y(1) = 1,\ y'(1) = 0$$

Примеры краевых задач:

$$\frac{d^2y}{dx^2} + 2\frac{dy}{dx} - y = \sin(x), \quad y(0) = 1,\ y(1) = 0$$

$$\frac{d^3y}{dx^3} = x + x\frac{d^2y}{dx^2} - \frac{dy}{dx}, \quad y(1) = 0,\ y'(1) = 0,\ y(3) = 2$$

Решить такие задачи аналитически удается лишь для некоторых специальных типов уравнений.

Единственность решения задачи Коши для уравнения теплопроводности в классе ограниченных функций.

Теорема 1.

Пусть D - ограниченная область в R^n. Если решение $u(x,t)$ смешанной задачи для уравнения теплопроводности

$u_t(x,t) - a^2(x,t)\Delta x u(x,t) = f(x,t)$, $(x,t) \in G = D \times (0;T)$

с начальными условиями

$u(x,0) = u_0(x)$, $u(0) \in C(\bar{D})$

с граничными условиями первого рода

$u(x,t)|_{x \in \partial D} = v(x,t)$, $v(x,t) \in C(\partial D \times [0;T])$

существует в классе функций $C^{2,1}_{x,t}(G) \cap C(\bar{G})$, то оно единственно в этом классе и непрерывно зависит от начальных и граничных данных (в равномерной метрике).

Доказательство теоремы 1. Единственность. Пусть $u_\sim$ и $u_\wedge$ - решение задачи. Тогда их разность $u = u_\sim - u_\wedge$ удовлетворяет однородному уравнению теплопроводности с однородными начальными и граничными условиями:

$u_t(x,t) = a^2(x,t)\Delta x u_x(x,t)$, $(x,t) \in G$,

$u(x,0) = 0$,

$u(x,t)|_{x \in \partial D} = 0$.

Согласно принципу максимума в ограниченной области выполняются неравенства

$0 \leqslant u(x,t) \leqslant 0$ $(x,t) \in \bar{G}$

Следовательно $u_\sim = u_\wedge$ в $\bar{G}$.

Непрерывная зависимость. Пусть теперь $u_\sim$ и $u_\wedge$ - решение задачи Коши отвечающие различным начально-краевым данным: $u_{0\sim}, u_{0\wedge}$ и $v_\sim, v_\wedge$ соответственно.

Тогда разность $u=u\sim-u\hat{}$ является решением смешанной задачи для однородного уравнения теплопроводности

$ut(x,t)=a2(x,t)\Delta xu(x,t)$, $(x,t)\in G$

с начальными условиями

$u(x,0)=u0\hat{}-u0\sim$

и граничными условиями

$u(x,t)|x\in\partial D=v\sim(x,t)-v\hat{}(x,t)$.

Воспользуемся для функции u принципом максимума, получаем оценку

$|u\sim(x,t)-u\hat{}(x,t)|\leqslant\max\{\sup D\bar{}|u0\sim-u0\hat{},\sup\partial D\times[0;T]|v\sim-v\hat{}|\}$, $(x,t)\in G\bar{}$,

что означает непрерывную зависимость решения от начальных и краевых данных в равномерной метрике. Из принципа единственности максимума в неограниченной области вытекает следующее

Теорема 2.

Если решение задачи Коши

$ut(x,t)-a2(x,t)\Delta xu(x,t)=f(x,t)$, $(x,t)\in G=Rn\times(0;T)$,

$u(x,0)=u0(x)$, $u0\in C(Rn)$

с ограниченными начальными данными $u0$ существует в классе функций $C2,1x,t(G)\cap C(G\bar{})\cap B(G\bar{})$, то оно единственно в нем и непрерывно зависит от начальных данных.

Задания для самостоятельной работы

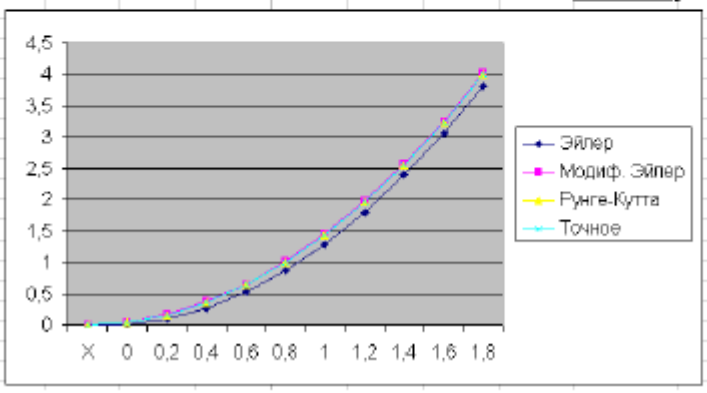
Решение обыкновенных дифференциальных уравнений (ОДУ)

В приведённом примере решается задача Коши, то есть, ищется решение дифференциального уравнения первого порядка вида $dy/dx = f(x,y)$ на интервале $x \in [x_0,x_n]$ при условии $y(x_0)=y_0$ и равномерном шаге сетки по x.

Решение выполняется методами Эйлера, "предиктор-корректор" (он же модифицированный метод Эйлера) и методом Рунге-Кутта 4 порядка точности. Пример может служить образцом для Ваших решений, правда, функцию придётся перепрограммировать несколько раз при различных значениях аргумента - поскольку без применения макросов на VBA Excel не позволяет создать полноценную функцию, которую было бы удобно вызывать с разными значениями аргументов.

Здесь решается уравнение $dy/dx = 2x-y+x^2$ на интервале [0,2], начальное значение $y(0)=0$, для оценки точности задано также точное решение в виде функции $u(x)=x^2$. Оценка погрешности делается в норме L_1, как и принято в данном случае.

	A	B	C	D	E	F	G	H	I	J	K	L
1	X	Эйлер	y~	Модиф. Эйлер	k1	k2	k3	k4	Рунге-Кутта	Точное		
2	0	0	0	0					0	0	A=	0
3	0,2	0	0	0,044	0	0,21	0,189	0,4022	0,040006667	0,04	B=	2
4	0,4	0,088	0,1232	0,16726	0,399993	0,609984	0,588994	0,802195	0,160012125	0,16	N=	10
5	0,6	0,2624	0,325624	0,36987	0,799988	1,009989	0,988989	1,20219	0,360016594	0,36	H=	0,2
6	0,8	0,52192	0,607976	0,652175	1,199983	1,409985	1,388985	1,602186	0,640020252	0,64		
7	1	0,865536	0,96874	1,013984	1,59998	1,809982	1,788982	2,002183	1,000023249	1		
8	1,2	1,292429	1,411187	1,455467	1,999977	2,209979	2,188979	2,402181	1,440025701	1,44		
9	1,4	1,801943	1,932373	1,976653	2,399974	2,609977	2,588977	2,802179	1,960027709	1,96		
10	1,6	2,393554	2,533346	2,57768	2,799972	3,009975	2,988975	3,202177	2,560029353	2,56		
11	1,8	3,066844	3,214144	3,258497	3,199971	3,409974	3,388973	3,602176	3,240030699	3,24		
12	2	3,821475	3,974798	4,019168	3,599969	3,809972	3,788972	4,002175	4,000031801	4		
13	Погрешн.	0,178525		0,019168					3,18007E-05			



Итог работы: отчет, защита работы.

Практическая работа № 7

Цель: научиться сводить произвольную задачу линейного программирования к основной задаче

Задание 1: ознакомиться с материалом

Задачи оптимального планирования, связанные с отысканием оптимума заданной целевой функции (линейной формы) при наличии ограничений в виде линейных уравнений или линейных неравенств относятся к задачам линейного программирования.

Линейное программирование – это направление математического программирования, изучающее методы решения экстремальных задач, которые характеризуются линейной зависимостью между переменными и линейным критерием.

Сущность линейного программирования состоит в нахождении точек наибольшего или наименьшего значения некоторой функции при определенном наборе ограничений, налагаемых на аргументы и образующих ***систему ограничений***, которая имеет, как правило, бесконечное множество решений. Каждая совокупность значений переменных (аргументов функции ***F***), которые удовлетворяют системе ограничений, называется ***допустимым планом*** задачи линейного программирования. Функция ***F***, максимум или минимум которой определяется, называется ***целевой функцией*** задачи. Допустимый план, на котором достигается максимум или минимум функции ***F***, называется ***оптимальным планом*** задачи.

Система ограничений, определяющая множество планов, диктуется условиями производства. Задачей линейного программирования (***ЗЛП***) является выбор из множества допустимых планов наиболее выгодного (оптимального).

Общая форма задачи линейного программирования формулируют следующим образом:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \{\leq, \geq, =\} b \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \{\leq, \geq, =\} b \\ \ldots\ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \{\leq, \geq, =\} \end{cases} \quad 1)$$

$$x_1 \geq 0,\ x_2 \geq 0,\ \ldots,\ x_n \geq 0 \quad 2)$$

$$F = c_1x_1 + c_2x_2 + \ldots + c_nx_n \rightarrow \max \quad 3)$$

Коэффициенты a_{ij}, b_i, c_j, $j = 1, 2, \ldots, n$, $i = 1, 2, \ldots, m$ – любые действительные числа (возможно 0).

Итак, решения, удовлетворяющие системе ограничений (1) условий задачи и требованиям неотрицательности (2), называются **допустимыми**, а решения, удовлетворяющие одновременно и требованиям минимизации (максимализации) (3) целевой функции, - **оптимальными**.

Выше описанная задача линейного программирования (ЗЛП) представлена в общей форме, но одна и та же (ЗЛП) может быть сформулирована в различных эквивалентных формах. Наиболее важными формами задачи линейного программирования являются **каноническая** и **стандартная**.

В **канонической форме** задача является задачей на максимум (минимум) некоторой линейной функции ***F***, ее система ограничений состоит только из равенств (уравнений). При этом переменные задачи $x_1, x_2, \ldots, x_n$ являются неотрицательными:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1, \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2, \\ \ldots\ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n = b_m. \end{cases} \quad 4)$$

$$x_1 \geq 0,\ x_2 \geq 0,\ \ldots,\ x_n \geq 0 \quad 5)$$

$$F = c_1x_1 + c_2x_2 + \ldots + c_nx_n \to \max(\min) \quad 6)$$

К канонической форме можно преобразовать любую задачу линейного программирования.

Правило приведения ЗЛП к каноническому виду:

1. Если в исходной задаче некоторое ограничение (например, первое) было неравенством, то оно преобразуется в равенство, введением в левую часть некоторой неотрицательной переменной, при чем в неравенства «≤» вводится дополнительная неотрицательная переменная со знаком «+»; в случаи неравенства «≥» - со знаком «-»

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \quad (7)$$

Вводим переменную $x_{n+1} = b_1 - a_{11}x_1 - a_{12}x_2 + \ldots + a_{1n}x_n$.

Тогда неравенство (7) запишется в виде:

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n + x_n \quad 8)$$

В каждое из неравенств вводится своя "**уравнивающая**" переменная, после чего система ограничений становится системой уравнений.

2. Если в исходной задаче некоторая переменная не подчинена условию неотрицательности, то ее заменяют (в целевой функции и во всех ограничениях) разностью неотрицательных переменных

$x_k = x_k -$
$x_k \geq 0,\ x_l$, l - свободный индекс

3. Если в ограничениях правая часть отрицательна, то следует умножить это ограничение на (-1)

4. Наконец, если исходная задача была задачей на минимум, то введением новой целевой функции $F_1 = -F$ мы преобразуем нашу задачу на минимум функции F в задачу на максимум функции F_1.

Таким образом, ***всякую задачу линейного программирования можно сформулировать в канонической форме.***

В стандартной форме задача линейного программирования является задачей на максимум (минимум) линейной целевой функции. Система ограничений ее состоит из одних линейных неравенств типа « <= » или « >= ». Все переменные задачи неотрицательны.

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \leq b_1, \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \leq b_2, \\ \ldots\ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \leq b_m. \end{cases} \quad 9)$$

$$F = c_1x_1 + c_2x_2 + \ldots + c_nx_n \to \max$$

$$x_1 \geq 0,\ x_2 \geq 0,\ \ldots,\ x_n \geq 0$$

Всякую задачу линейного программирования можно сформулировать в **стандартной форме**. Преобразование задачи на минимум в задачу на максимум, а также обеспечение не отрицательности переменных производится так же, как и раньше. Всякое равенство в системе ограничений равносильно системе взаимопротивоположных неравенств:

$$a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n = b_i \Leftrightarrow$$
$$\Leftrightarrow \begin{cases} a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n \leq b_i, \\ -a_{i1}x_1 - a_{i2}x_2 - \ldots - a_{in} \leq b_i, \end{cases}$$

Существует и другие способы преобразования системы равенств в систему неравенств, т.е. ***всякую задачу линейного программирования можно сформулировать в стандартной форме.***

Задание 2:

Порядок выполнения заданий

Задание 1. а) Привести к канонической форме задачу линейного программирования.

$$\begin{cases} 2x_1 - x_2 + 3x_3 \le 5 \\ x_1 + 2x_3 = 8 \\ -x_1 - 2x_2 \ge 1 \end{cases}$$

$$x_1 \ge 0, x_2 \ge 0, x_3 \ge 0$$

$$F = x_1 - x_2 + 3x_3 \to \min$$

б) Напишите задачу в стандартной форме.

Решение:

а) Введем дополнительные переменные x_4, x_5. Причем в первое неравенство введем переменную x_4 со знаком плюс, а в третье – неотрицательную переменную, x_5 со знаком минус запишем задачу в виде:

$$\begin{cases} 2x_1 - x_2 + 3x_3 + x_4 = 5 \\ x_1 + 2x_3 = 8 \\ -x_1 - 2x_2 - x_5 = 1 \end{cases}$$

$$x_1 \ge 0, x_2 \ge 0, x_3 \ge 0, x_4 \ge 0, x_5 \ge 0$$

Переведем *min* на *max*, домножив целевую функцию на (-1)

$$F = -x_1 + x_2 - 3x_3 + 0 \cdot x_4 + 0 \cdot x_5 \to \max$$

что и дает эквивалентную задачу в канонической форме.

б) Всякую задачу линейного программирования можно сформулировать в **стандартной форме**. Преобразование задачи на минимум в задачу на максимум, а также обеспечение не отрицательности переменных производится так же, как и раньше. Всякое равенство в системе ограничений равносильно системе взаимопротивоположных неравенств, тогда получим:

$$\begin{cases} 2x_1 - x_2 + 3x_3 \le 5 \\ x_1 + 2x_3 \le 8 \\ -x_1 - 2x_3 \le 8 \\ -x_1 - 2x_2 \ge 1 \end{cases}$$

$$x_1 \ge 0, x_2 \ge 0, x_3 \ge 0$$

$$F = -x_1 + x_2 - 3x_3 \to \max$$

Задание 2:

Вариант1

1 вариант.

Задача 1. а) Привести к канонической форме задачу линейного программирования.

$$\begin{cases} x_1 - 2x_2 + x_3 \ge 4, \\ x_1 + x_2 - 3x_3 \le 9 \\ x_1 + 3x_2 + 2x_3 = 10 \end{cases}$$

$$x_1 \ge 0, x_2 \ge 0, x_3 \ge 0$$

$$F = 2x_1 + x_2 - x_3 \to \max$$

б) Напишите задачу в стандартной форме.

Вариант 2

Задача 1. а) Привести к канонической форме задачу линейного программирования.

$$\begin{cases} 4x_1 + 2x_2 + 5x_3 \le 12, \\ 6x_1 - 3x_2 + 4x_3 = 18 \\ 3x_1 + 3x_2 - 2x_3 \ge 16 \end{cases}$$

$$x_1 \ge 0, x_2 \ge 0, x_3 \ge 0$$

$$F = -2x_1 + x_2 + 5x_3 \to \max$$

б) Напишите задачу в стандартной форме.

Итог работы: отчет, защита работы.

Практическая работа № 8

Цель: научиться решать задачу линейного программирования графическим методом

Задание 1: ознакомиться с материалом

Графический метод решения двумерной задачи линейного программирования (максимизация целевой функции)

Двумерная задача линейного программирования – задача линейного программирования, количество переменных которой равно 2.

В общем виде двумерную задачу линейного программирования можно представить следующим образом.

Определить значение переменных x_1 и x_2, при которых линейная целевая функция F достигает максимума (минимума).

$F = c_1x_1+c_2x_2 \rightarrow \max(\min)$ при ограничениях на переменные

$$\begin{cases} a_{11}x_1 + a_{12} \le (=, \ge) b_1 \\ a_{21}x_1 + a_{22}x_2 \le (=, \ge) b_2 \\ \dots \\ a_{m1}x_1 + a_{m2}x_2 \le (=, \ge) bm \\ x_1 \ge 0,\ x_2 \ge 0 \end{cases}$$

Среди ограничений могут одновременно встречаться знаки $\ge$, $\le$ и =. Коэффициенты a_{ij}, b_i, c_j (i = 1..m, j = 1,2) - любые действительные числа (возможно и 0).

Двумерные задачи линейного программирования обычно решаются графически и решение связано со свойствами выпуклых множеств.

Множество точек называется выпуклым, если оно вместе с любыми двумя точками содержит и их произвольную выпуклую комбинацию.

Геометрический смысл этого определения состоит в том, что множеству вместе с его произвольными точками полностью принадлежит и прямолинейный отрезок, их соединяющий. Примерами выпуклых множеств являются прямолинейный отрезок, полуплоскость, круг, шар, куб, полупространство и др.

Множество планов основной задачи линейного программирования является выпуклым (если оно не пусто). Непустое множество планов называется многогранником решений, а всякая угловая точка многогранника решений - вершиной.

Если основная задача линейного программирования имеет оптимальный план, то целевая функция задачи принимает максимальное значение в одной из вершин многогранника решений. Если максимальное значение достигается более чем в одной вершине, то целевая функция принимает его во всякой точке, являющейся выпуклой линейной комбинацией этих вершин.

Алгоритм решения двумерной задачи линейного программирования графическим методом

Решение задачи линейного программирования графическим методом включает следующие этапы.

1. На плоскости X_1OX_2 строят прямые, уравнения которых получаются в результате замены в ограничениях знаков неравенств на знаки точных равенств.
2. Находят полуплоскости, определяемые каждым из ограничений задачи.
3. Строят многоугольник решений.
4. Строят вектор$N(c_1, c_2)$, который указывает направление возрастания целевой функции.
5. Строят начальную прямую целевой функции $c_1x_1 + c_2x_2 =0$ и затем передвигают ее в направлении вектора N до крайней угловой точки многоугольника решений. В результате находят точку, в которой целевая функция принимает максимальное значение, либо множество точек с одинаковым максимальным значением целевой функции, если начальная прямая сливается с одной из сторон многоугольника решений, либо устанавливают неограниченность сверху функции на множестве планов.
6. Определяют координаты точки максимум функции и вычисляют значение целевой функции в этой точке.

Минимальное значение линейной функции цели находится путем передвижения начальной прямой $c_1x_1 + c_2x_2 = 0$ в направлении, противоположном вектору $N(c_1,c_2)$.

Замечание 1: *В алгоритме решения пункты 4-6 можно выполнять следующим образом:*

4. Найти значение целевой функции в угловых точках многогранника решений.
5. Точка, в которой функция принимает наибольшее значение и является точкой максимума.

Пример 1

На предприятии имеется сырье видов I, II, III. Из него можно изготавливать изделия типов A и B. Пусть запасы видов сырья на предприятии составляют $b1$, $b2$, $b3$ ед. соответственно, изделие типа A дает прибыль $c1$ ден. ед., а изделие типа B – $c2$ ден. ед. Расход сырья на изготовление одного изделия задан в условных единицах таблицей. Составить план выпуска изделий, при котором предприятие имеет наибольшую прибыль. Решить задачу графически и симплексным методом.

Изделие	Сырье			b_1	b_2	b_3	c_1	c_2
	I	II	III					
A	3	4	3	150	260	300	6	3
B	1	3	4					

Решение. Составим математическую модель задачи. Обозначим: x_1 – количество выпускаемых изделий типа A, x_2 - количество выпускаемых изделий типа B . Тогда с учетом расходов сырья на изготовление изделия каждого типа получим следующие ограничения на x_1 и x_2 , учитывающие запасы сырья каждого вида:

$$\begin{cases} 3x_1 + x_2 \le 150 \\ 4x_1 + 3x_2 \le 260 \\ 3x_1 + 4x_2 \le 300 \end{cases} \quad (1)$$

По смыслу задачи

$$x_1 \ge 0, x_2 \ge 0$$

Прибыль F предприятия при плане x_1, x_2 равна

$$F = 6x_1 + 3x_2 . \quad (3)$$

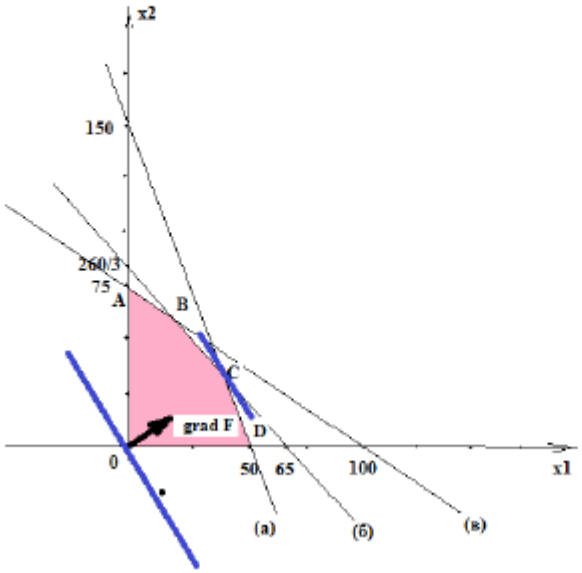
Итак, математическая модель задачи получена: необходимо найти значения x_1, x_2 , удовлетворяющие неравенствам (1), (2), для которых функция (3) достигает наибольшего значения. Полученная задача – стандартная задача линейного программирования.

Решим полученную задачу графически.

Для этого введем систему координат x_1Ox_2 и изобразим в ней множество решений систем неравенств (1), (2) (область допустимых решений - *ОДР*) в виде множества точек плоскости.

Условию (2) удовлетворяют точки первой четверти. Для получения полуплоскостей, соответствующих неравенствам системы (1), построим их границы, т.е. прямые линии:

Имя прямой	Уравнение прямой	Таблица для построения прямой		
(а)	$3x_1 + x_2 = 150$	x_1	0	50
		x_2	150	0
(б)	$4x_1 + 3x_2 = 260$	x_1	0	65
		x_2	260/3	0
(в)	$3x_1 + 4x_2 = 300$	x_1	0	100
		x_2	75	0



$$C:\begin{cases}(a)\\(б)\end{cases}\Rightarrow\begin{cases}4x_1+3x_2=260\\3x_1+x_2=150\end{cases}\Rightarrow\begin{cases}x_1=38\\x_2=36\end{cases}\Rightarrow C(38;36)$$

F(C)=6*38+3*36=336

D(50;0)

F(D)=6*50+3*0=300.

Отсюда

F_{max}= F(C)= F(38;36)=336.

Вывод: предприятию выгодно выпустить 38 изделий типа $A(x_1=38)$ и 36 изделия типа B ($x_2=36$). При этом его прибыль будет наибольшая и составит 336 ден. ед.

Задание 2:

Задание 1

На предприятии имеется сырье видов I, II, III. Из него можно изготавливать изделия типов A и B. Пусть запасы видов сырья на предприятии составляют b1, b2, b3 ед. соответственно, изделие типа A дает прибыль c1 ден.ед., а изделие типа B - c2 ден.ед. Расход сырья на изготовление одного изделия задан в словных единицах таблицей.

Составить план выпуска изделий, при котором предприятие имеет наибольшую прибыль. Решить задачу графическим методом.

1 вариант								
Изделие	Сырье			b1	b2	b3	c1	c2
	I	II	III					
A	6	3	2	102	91	105	5	9
B	3	4	5					
2вариант								
Изделие	Сырье			b1	b2	b3	c1	c2
	I	II	III					
A	1	1	3	20	36	40	2	5
B	3	2	1					
3 вариант								
Изделие	Сырье			b1	b2	b3	c1	c2
	I	II	III					
A	2	1	3	40	34	46	1	2
B	2	2	1					
4 вариант								
Изделие	Сырье			b1	b2	b3	c1	c2
	I	II	III					
A	3	4	3	300	520	600	6	3
B	1	3	4					

Задание 2

Предприятие выпускает три вида изделий (N1, N2, N3), используя три вида ресурсов (P1, P2, P3). Запасы ресурсов (З) ограничены. Прибыль от реализации (П) единицы изделия и нормы расхода ресурсов представлены в таблице. Определить ассортимент и объемы выпуска продукции, получаемую прибыль, величину остатков. Найти решение задачи симплексным методом с представлением всех симплексных таблиц и проанализировать полученные результаты.

Вариант 1				
	N1	N2	N3	З
P1	1	8	5	43
P2	4	1	6	74
P3	5	2	2	35
П	5	7	6	

Вариант 2				
	N1	N2	N3	3
P1	3	5	4	81
P2	6	1	3	74
P3	1	5	2	33
П	4	8	7	

Вариант 3				
	N1	N2	N3	3
P1	6	7	2	57
P2	6	6	1	97
P3	3	7	8	63
П	5	6	8	

Вариант 4				
	N1	N2	N3	3
P1	7	8	3	81
P2	4	1	6	68
P3	5	1	7	54
П	2	5	6	

Итог работы: отчет, защита работы.

Практическая работа №9

Цель: научиться решать задачу линейного программирования симплекс - методом

Задание 1: ознакомиться с материалом

Решение задач линейного программирования симплекс-методом.

Идея симплекс-метода заключается в последовательном улучшении первоначального плана путем упорядоченного перехода от одного опорного плана к другому и завершается нахождением оптимального плана. Симплекс-методом решаются только канонические задачи линейного программирования. Решение канонической задачи симплекс-методом существенно облегчается применением так называемых симплексных таблиц. Всякую каноническую задачу можно записать условно в виде таблицы. Таблица заполняется следующим образом: первые m строк содержат в условной форме уравнения системы ограничений, разрешенные относительно базисных переменных. В последней строке записана целевая функция, эта строка называется F -строкой. В столбцах записаны свободные переменные и свободные члены.

Условие оптимальности плана: если ЗЛП на максимум, то в F-строке не должно быть отрицательных элементов; если ЗЛП на минимум, то в F-строке не должно быть положительных элементов.

Алгоритм решения:

1. Исходную задачу линейного программирования приводим к каноническому виду путем введения базисных переменных.
2. Базисные переменные выражаем через свободные переменные.
3. Строим начальный план, полагая свободные переменные равными нулю, тогда базисные переменные будут равны свободным членам.
4. Строим первую симплекс-таблицу.
5. Проверяем план на оптимальность. Если план не оптимален, то его улучшаем.
6. Улучшение плана.

а) выбор разрешающего столбца: для этого в F- строке выбираем максимальный по абсолютной величине из отрицательных элементов, если задача на максимум, или, максимальный из положительных элементов, если задача на минимум. Пусть это будет столбец с номером s;

б) выбор разрешающей строки: выбираем строку с минимальным симплексным отношением. Симплексные отношения - это отношение свободных членов к положительным элементам разрешающего столбца. Пусть это будет строка с номером r.

в) выбор разрешающего элемента: элемент, стоящий на пересечении разрешающих строки и столбца. Пусть это будет элемент a_{rs}.

г) переменную x_s вводим в базис вместо переменной x_r.

д) элементы новой симплекс-таблицы b_{ij} пересчитываем по следующим формулам:

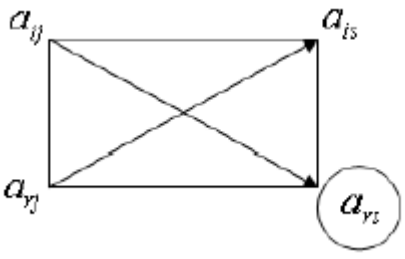
разрешающий элемент $$b_{rs} = \frac{1}{a_{rs}},$$

элементы разрешающего столбца $$b_{is} = -\frac{a_{is}}{a_{rs}}, i \neq r,$$

элементы разрешающей строки $$b_{rj} = \frac{a_{rj}}{a_{rs}}, j \neq s,$$

остальные элементы симплекс-таблицы по правилу прямоугольника:

$$b_{ij} = \frac{a_{ij} \cdot a_{rs} - a_{rj} \cdot a_{is}}{a_{rs}}$$



Задание 2. Для производства двух видов, изделии P_1 и P_2 используется, три вида сырья S_1, S_2, S_3, запасы которого соответственно равны 100, 60, 180 единиц. Для производства одной единицы продукции P_1 используется 2 единицы сырья S_1 и по 1 единице сырья $S_2 и S_3$. Для производства одной единицы продукции P_2 используется по 1 единице сырья $S_1 и S_2$ и 4 единицы сырья S_3. Прибыль от реализации 1 единицы каждой продукции P_1 и P_2 соответственно равна 30 и 20 единиц. Необходимо составить симплекс-методом такой план выпуска продукции P_1 и P_2, при котором суммарная прибыль будет наибольшей.

Решение.

1. Составим математическую модель задачи:

Пусть x_1 – единица готовой продукции вида P_1,

x_2 - единица готовой продукции вида P_2,

Цель фабрики получить максимальную прибыль от реализации всей продукции видов P_1 и P_2, тогда:

$$F = 30 \cdot x_1 + 20 \cdot x_2 \to \max$$

Система ограничений:

$$\begin{cases} 2x_1 + x_2 \le 100 \\ x_1 + x_2 \le 60 \\ x_1 + 4x_2 \le 180 \end{cases}$$

$x_1 \ge 0, \quad x_2 \ge 0$ *условие неотрицательности*

2. Задачу приводим к каноническому виду:

$$F = 30 \cdot x_1 + 20 \cdot x_2 \to \max$$

$$\begin{cases} 2x_1 + x_2 + x_3 = 100 \\ x_1 + x_2 + x_4 = 60 \\ x_1 + 4x_2 + x_5 = 180 \end{cases}$$

$x_1 \ge 0, \quad x_2 \ge 0 \quad x_3 \ge 0, \quad x_4 \ge 0, \quad x_5 \ge 0$

3. Базисные переменные выражаем через свободные:

$$\begin{cases} x_3 = 100 - 2x_1 - x_2 \\ x_4 = 60 - x_1 - x_2 \\ x_5 = 180 - x_1 - 4x_2 \end{cases}$$

4. Записываем начальный план: $X_0 = (0; \quad 0; \quad 100; \quad 60; \quad 180)$.
5. Строим первую симплекс-таблицу:

Таблица 1. Первая симплекс-таблица

Своб. перем. / Базис. перем.	$-x_1$	$-x_2$	Свободные члены	Симплексные отношения
x_3	2	1	100	$\frac{100}{2} = 50 \min$
x_4	1	1	60	$\frac{60}{1} = 60$
x_5	1	4	180	$\frac{180}{1} = 180$
F-строка	-30	-20	0	

6. Начальный план не оптимален, так как в F-строке есть отрицательные элементы.
7. Улучшение плана. Строим вторую симплекс-таблицу, элементы которой пересчитываем по соответствующим формулам.

Таблица 2. Вторая симплекс-таблица

Своб. перем. / Базис. перем.	$-x_3$	$-x_2$	Свободные члены	Симплексные отношения
x_1	$\frac{1}{2}$	$\frac{1}{2}$	50	100

▶ x_4	$-\frac{1}{2}$	$\frac{1}{2}$	10	20 min
x_5	$-\frac{1}{2}$	$\frac{7}{2}$	130	≈ 37,14
F-строка	15	-5	1500	

8. План, соответствующий таблице 2, $X_1 = (50;\ 0;\ 0;\ 10;\ 130)$ не оптимален, так как в F-строке есть отрицательные элементы. Улучшаем его.
9. Улучшение плана. Строим третью симплекс-таблицу, элементы которой пересчитываем по соответствующим формулам.

Таблица 3. Третья симплекс-таблица

Своб. перем. / Базис. перем.	$-x_3$	$-x_4$	Свободные члены	Симплексные отношения
x_1	1	-1	40	
x_2	-1	2	20	
x_5	3	-7	60	
F-строка	10	10	1600	

10. План, соответствующий таблице 3, $X_2 = (40;\ 20;\ 0;\ 0;\ 60)$ оптимален, так как в F-строке нет отрицательных элементов.

Ответ: если предприятие будет выпускать продукцию вида P_1 и P_2 в количестве 40 и 20 единиц соответственно, то получит максимальную прибыль в размере 1600 единиц, при этом сырье $S_1 и S_2$ будет израсходовано полностью, а сырье S_3 останется в количестве 60 единиц.

Задание 2:

Вариант 1

Задача 2. Предположим, что для производства продукции вида А и В можно использовать материал трех сортов. При этом на изготовление единицы изделия вида А расходуется ***a1*** кг первого сорта, ***a2*** кг второго сорта и ***a3*** кг третьего сорта. На изготовление продукции вида В расходуется ***b1*** кг первого сорта, ***b2*** кг второго сорта, ***b3*** кг третьего сорта. На складе фабрики имеется всего материала первого сорта ***c1*** кг, второго сорта ***c2*** кг, третьего сорта ***c3*** кг. От реализации единицы готовой продукции вида А фабрика имеет прибыль вида ***α*** руб., а от реализации единицы готовой продукции вида В фабрика имеет прибыль вида ***β*** руб. Определить максимальную прибыль от реализации всей продукции видов А и В симлекс-методом.

a1= 19, a2= 16, a3= 19, b1= 31, b2= 9, b3= 1, c1= 1121, c2= 706, c3= 1066,
α=16, β=19.

Вариант 2

Задача 2. Предположим, что для производства продукции вида А и В можно использовать материал трех сортов. При этом на изготовление единицы изделия вида А расходуется ***a1*** кг первого сорта, ***a2*** кг второго сорта и ***a3*** кг третьего сорта. На изготовление продукции вида В расходуется ***b1*** кг первого сорта, ***b2*** кг второго сорта, ***b3*** кг третьего сорта. На складе фабрики имеется всего материала первого сорта ***c1*** кг, второго сорта ***c2*** кг, третьего сорта ***c3*** кг. От реализации единицы готовой продукции вида А фабрика имеет прибыль вида ***α*** руб., а от реализации единицы готовой продукции вида В фабрика имеет прибыль вида ***β*** руб. Определить максимальную прибыль от реализации всей продукции видов А и В симплекс-методом.

a1= 14, a2= 15, a3= 20, b1= 40, b2= 27, b3= 4, c1= 1200, c2= 993, c3= 1097,
α=5, β=13.

Итог работы: отчет, защита работы.

Практическая работа №10

Цель: научиться находить начальное решение транспортной задачи

Задание 1: ознакомиться с материалом

Симплексный метод для решения задач линейного программирования является универсальным, он позволяет решить любую задачу, но решение иных задач связано с трудоемкими расчетами. Можно выделить класс задач, которые решаются более простыми специальными методами. К числу таких задач относятся так называемые **транспортные задачи.**

Классическая транспортная задача - о наиболее экономном плане перевозок однородного продукта или взаимозаменяемых продуктов из пунктов отправления в пункты назначения.

Классическая транспортная задача (сокращенно ТЗ) формулируется следующим образом.

В пунктах отправления $A_1, A_2, ..., A_m$, которые будем называть также поставщиками, сосредоточены запасы однородного груза в количествах $a_1, a_2, ..., a_m$ соответственно. В пункты назначения $B_1, B_2, ..., B_n$, именуемые потребителями, надлежит доставить соответственно $b_1, b_2, ..., b_n$ единиц груза.

Известен транспортный тариф c_{ij} - стоимость перевозки единицы груза из пункта A_i в пункт B_j, $i = 1,2,...,m, j = 1,2,...,n$. Требуется составить такой план перевозок груза, при котором общая стоимость F всех перевозок была бы наименьшей, при этом все заявки были бы выполнены.

В термин "транспортный тариф" вкладывается условное понимание стоимости единицы груза - это может быть себестоимость, расстояние, тариф, время, расход топлива или электроэнергии и др.

Пусть суммарные запасы грузов у поставщиков равны суммарным потребностям потребителей:

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j$$

Это условие называется условием баланса. Если для ТЗ условие баланса выполняется, то модель ТЗ называется закрытой, если условие баланса не выполнено, то модель ТЗ - открытая. Составим математическую модель ТЗ.

Пусть x_{ij} - количество груза, которое поставщик A_i отправляет потребителю B_j ($i = 1,2,...,m, j = 1,2,...,n$) со стоимостью перевозок c_{ij}. Данные задачи можно представить в виде таблицы 1.

Таблица 1.

Поставщики	Потребители				Запасы
	B_1	B_2	…	B_n	
A_1	c_{11} x_{11}	c_{12} x_{12}	…	c_{1n} x_{1n}	a_1
A_2	c_{21} x_{21}	c_{22} x_{22}	…	c_{2n} x_{2n}	a_2
…	…	…	…	…	…

A_m	c_{m1} x_{m1}	c_{m2} x_{m2}	…	c_{mn} x_{mn}	a_m
Потребности	b_1	b_2	…	b_n	$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j$

По смыслу своему величины $x_{ij} \geq 0$ и должны удовлетворять следующим ограничениям:

- Из пункта A_i все запасы должны быть вывезены (ограничения по ресурсам): $\sum_{j=1}^{n} x_{ij} = a_i (i = 1,2,...,m)$;
- Заявки потребителей B_j должны быть выполнены (ограничения по потребностям): $\sum_{i=1}^{m} x_{ij} = b_j (j = 1,2,...,n)$.

Затраты на перевозку x_{ij} единиц груза из пункта поставки A_i в пункт потребления B_j составляют $c_{ij} \cdot x_{ij}$ рублей; общая же стоимость всех перевозок x_{ij} равна сумме всех таких затрат: $F = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} \cdot x_{ij} \to \min$

Математическая постановка ТЗ состоит в следующем:

составить план перевозок x_{ij}, удовлетворяющих системе ограничении:

$$\begin{cases} \sum_{i=1}^{m} x_{ij} = b_j, (j = 1,2,...,n) \\ \sum_{j=1}^{n} x_{ij} = a_i, (i = 1,2,...,m) \\ \sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j \end{cases},$$

условию неотрицательности: $x_{ij} \geq 0$, $i = 1,2,...,m, j = 1,2,...,n$, при котором целевая функция достигает своего минимума:

$$F = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} \cdot x_{ij} \to \min$$

Из математической модели видно, что ТЗ является частным случаем общей задачи линейного программирования. В общей теории линейного программирования доказаны следующие теоремы:

Теорема 1. Транспортная задача при выполнении условия баланса всегда имеет решение.

Теорема 2. Система ограничений транспортной задачи содержит m+n-1 линейно-независимых уравнений.

При решении задач практический смысл теоремы 2 заключается в следующем: число назначенных перевозок равно *m+n-1*.

Процедура решения ТЗ будет состоять в последовательном улучшении опорных планов и проверки их на оптимальность.

Методы построения начального плана.

Существует несколько методов построения первоначального опорного плана ТЗ (опорный план - план, удовлетворяюший системе ограничений и условию неотрицательности). Рассмотрим только два из них: метод северо-западного угла и метод наименьшей стоимости.

Как уже отмечалось, в опорном плане не более $r = m + n - 1$ переменных x_{ij}, отличных от нуля. Если таких переменных равно r, то такой план называют невырожденным, в противном случае - вырожденным.

Метод северо-западного угла. Назначение перевозок начинаем с левой верхней клетки (северо-западный угол). Сравнивая ресурсы поставщика и потребности потребителя, назначаем максимально возможную перевозку. Если ресурсов поставщика недостаточно, то переходим к следующему поставщику. Если ресурсов у поставщика достаточно, то назначив нужную перевозку первому потребителю, переходим к следующему потребителю. При назначении перевозок для удобства записываем остаток ресурсов (потребностей); если ресурсы закончились или потребности удовлетворены, то ставим букву "к" (конец). Если при назначении перевозки одновременно закончились запасы ресурсов у поставщика и удовлетворены потребности потребителя, то из "игры" выводим только одного участника, другому оставляем нуль запасов или нуль потребностей.

Метод наименьшей стоимости. Выбираем клетку с наименьшей тарифной ставкой и назначаем максимально возможную перевозку. Если запасы закончились или потребности удовлетворены, то поставщика или потребителя исключаем. Среди оставшихся клеток снова выбираем клетку с наименьшей стоимостью и назначаем максимально возможную перевозку. Если в результате назначения перевозки закончились запасы поставщика или удовлетворены потребности потребителя, то его исключаем из дальнейшего рассмотрения.

Метод потенциалов построения оптимального плана.

Наиболее простым методом решения ТЗ является метод потенциалов. Потенциалами называются условные числа u_i, v_j приписанные определенным образом каждому поставщику и потребителю.

Теорема 3(условие оптимальности плана). Сумма потенциалов поставщика и потребителя равна тарифной ставке для занятых клеток; сумма потенциалов поставщика и потребителя не превышает тарифную ставку для свободных клеток:

$$\begin{cases} u_i + v_j = c_{ij}, & x_{ij} \geq 0, \\ u_i + v_j \leq c_{ij}, & x_{ij} = 0. \end{cases}$$

Замечание. Опорный план должен быть невырожденным.

Алгоритм решения транспортной задачи:

1. Строим начальные планы методом северо-западного угла и наименьшей стоимости, из них выбираем лучший.
2. Находим потенциалы поставщиков и потребителей, используя первое условие оптимальности плана: $u_i + v_j = c_{ij}$
3. Проверяем второе условие оптимальности плана для свободных клеток $u_i + v_j \leq c_{ij}$. Если оно выполнено, то план оптимален. Если не выполнено, то улучшаем план.
4. Улучшение плана.

a) при невыполнении второго условия оптимальности плана в клетку заносим нарушение $u_i + v_j - c_{ij}$ со знаком "+". Такие клетки называются потенциальными;

b) среди всех потенциальных клеток выбираем клетку с наибольшим нарушением;

c) строим для выбранной клетки замкнутый контур, состоящий из вертикальных и горизонтальных отрезков прямой, причем вершины контура лежат в занятых клетках, за исключением той клетки, для которой строится контур. Виды контуров приведены на рисунке 1;

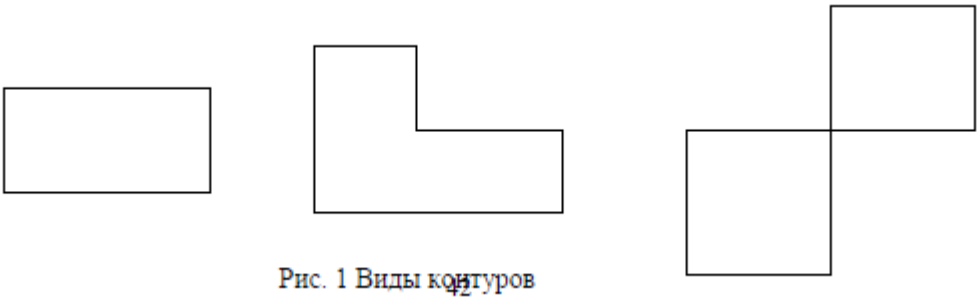


Рис. 1 Виды контуров

d) вершины контура поочередно помечаем, знаками "+","-", начиная с клетки, для которой построен контур;

е) среди клеток, помеченных знаком "-", выбираем наименьшую перевозку. На эту величину увеличиваем перевозки в клетках, помеченных знаком "+", и уменьшаем в клетках, помеченных знаком "-". В результате переназначения перевозок освобождается одна клётка.

5.Вновь полученный план проверяем на оптимальность.

Порядок выполнения заданий

Задание. Имеются три пункта поставки однородного груза **A1, A2, A3** и пять пунктов **B1, B2, B3, B4, B5** потребления этого груза. На пунктах **A1, A2 и A3** находится груз соответственно в количестве ***a1, a2 и a3*** тонн. В пункты **B1, B2, B3, B4, B5** требуется доставить соответственно ***b1, b2, b3, b4, b5*** тонн груза. Расстояние между пунктами поставки и пунктами потребления приведено в таблице:

Пункты поставки	**Пункты потребления**				
	B1	**B2**	**B3**	**B4**	**B5**
A1	D11	D12	D13	D14	D15
A2	D21	D22	D23	D24	D25
A3	D31	D32	D33	D34	D35

Найти такой план закрепления потребителей за поставщиками однородного груза, чтобы общие затраты по перевозкам были минимальными.

a1=200, a2=250, a3=200,

b1=190, b2=100, b3=120, b4=110, b5=130.

$$D=\begin{pmatrix}28 & 27 & 18 & 27 & 24\\ 18 & 26 & 27 & 32 & 21\\ 27 & 33 & 23 & 31 & 34\end{pmatrix}$$

Решение.

1. Построим начальный план двумя методами: методом северо-западного угла и методом наименьшей стоимости, и выберем тот план, который будет наилучшим, то есть получим минимальные затраты за перевозку однородного груза.

А) Строим начальный план методом северо-западного угла. Составим таблицу значений:

Потребители Поставщики	**B1**	**B2**	**B3**	**B4**	**B5**	Запасы
A1	28 190	27 10	18	27	24	200, 10, к
A2	18	26 90	27 120	32 40	21	250, 160, 40, к
A3	27	33	23	31 70	34 130	200, 130, к
Потребности	190, к	100, 90, к	120, к	110, 70, к	130, к	650=650

Число назначенных перевозок ***m+n-1=3+5-1=7***, то есть начальный план

$x_{11}=190,\quad x_{12}=10,\quad x_{22}=90,\quad x_{23}=120,\quad x_{24}=40,\quad x_{34}=70,\quad x_{35}=130$

невырожденный.

При таком плане суммарные транспортные издержки равны:

$$F=28\cdot 190+27\cdot 10+26\cdot 90+27\cdot 120+32\cdot 40+31\cdot 70+34\cdot 130=5320+270+2340+3240+$$
$$+1280+2170+4420=19040(единиц)$$

Б) Строим начальный план методом наименьшей стоимости. Составим таблицу значений:

Потребители Поставщики	**B1**	**B2**	**B3**	**B4**	**B5**	Запасы
A1	28	27 10	18 120	27	24 70	200, 80, 10, к
A2	18	26	27	32	21	250, 60, к

	190				60	
A3	27	33 90	23	31 110	34	200, 90, к
Потребности	190, к	100, 90, к	120, к	110, к	130, 70, к	650=650

Начальный план:

$$x_{12}=10,\quad x_{13}=120,\quad x_{15}=70,\quad x_{21}=190,\quad x_{25}=60,\quad x_{32}=90,\quad x_{34}=110$$

При таком плане транспортные издержки

$$F=27\cdot 10+18\cdot 120+24\cdot 70+18\cdot 190+21\cdot 60+33\cdot 90+31\cdot 110=270+2160+1680+34$$
$$+1260+2970+3410=15170(\text{единиц})$$

Сравнивая транспортные издержки, видим, что план, построенный методом наименьшей стоимости, лучший.

2. Выбираем лучший план и находим потенциалы поставщиков и потребителей, используя первое условие оптимальности плана: $u_i+v_j=c_{ij}$

Потребители, v_j Поставщики, u_i		**21**	**27**	**18**	**25**	**24**
		B1	**B2**	**B3**	**B4**	**B5**
0	**A1**	28	27 10	18 120	27	24 70
-3	**A2**	18 190	26	27	32	21 60
6	**A3**	27	33 90	23 **+1**	31 110	34

Используя первое условие оптимальности плана, составим систему линейных уравнений для определения потенциалов:

$$\begin{cases} u_1+v_2=27 \\ u_1+v_3=18 \\ u_1+v_5=24 \\ u_2+v_1=18 \\ u_2+v_5=21 \\ u_3+v_2=33 \\ u_3+v_4=31 \end{cases}$$

Система линейных уравнений содержит 7 уравнений и 8 неизвестных, т.е. она имеет множество решений. Так как нужно одно решение, то любой из неизвестных задаем значение и вычисляем остальные неизвестные.

■

Пусть $u_1=0$, тогда

$$u_2=-3,\quad u_3=6,\quad v_1=21,\quad v_2=27,\quad v_3=18,\quad v_4=25,\quad v_5=24$$

3. Проверяем второе условие оптимальности плана для свободных клеток $u_i+v_j\le c_{ij}$. Если есть нарушения, то заносим их со знаком «+». В результате проверки получили одну потенциальную клетку. Таким образом, начальный план не оптимален.

4. Улучшение плана. Выбираем клетку с максимальным нарушением и для нее строим замкнутый контур.

Потребители, v_j Поставщики, u_i						
		B1	**B2**	**B3**	**B4**	**B5**
	A1	28	27 + 10	18 - 120	27	24 70
	A2	18	26	27	32	21

		190				60
	A3	27	33 - 90	23 **+1**	31 110	34

Среди клеток, помеченных знаком «-», выбираем наименьшую перевозку:

$q = \min(90,120) = 90$

На эту величину увеличиваем перевозки в клетках, помеченных знаком «+», и уменьшаем в клетках, помеченных знаком «-». В результате переназначения перевозок имеем план:

Потребители, v_j / Поставщики, u_i		**21**	**27**	**18**	**26**	**24**
		B1	**B2**	**B3**	**B4**	**B5**
0	**A1**	28	27 100	18 30	27	24 70
-3	**A2**	18 190	26	27	32	21 60
5	**A3**	27	33	23 90	31 110	34

Используя первое условие оптимальности плана, составим систему линейных уравнений для определения потенциалов:

$$\begin{cases} u_1 + v_2 = 27 \\ u_1 + v_3 = 18 \\ u_1 + v_5 = 24 \\ u_2 + v_1 = 18 \\ u_2 + v_5 = 21 \\ u_3 + v_3 = 23 \\ u_3 + v_4 = 31 \end{cases}$$

Система линейных уравнений содержит 7 уравнений и 8 неизвестных, т.е. она имеет множество решений. Так как нужно одно решение, то любой из неизвестных задаем значение и вычисляем остальные неизвестные.

Пусть $u_1 = 0$, тогда

$u_2 = -3, \quad u_3 = 5, \quad v_1 = 21, \quad v_2 = 27, \quad v_3 = 18, \quad v_4 = 26, \quad v_5 = 24$

Проверяем второе условие оптимальности плана для свободных клеток $u_i + v_j \le c_{ij}$. Условие оптимальности выполнены, следовательно, план, соответствующий таблице, оптимален.

$x_{12} = 100, \quad x_{13} = 30, \quad x_{15} = 70, \quad x_{21} = 190, \quad x_{25} = 60, \quad x_{33} = 90, \quad x_{34} = 110$

$F = 27 \cdot 100 + 18 \cdot 30 + 24 \cdot 70 + 18 \cdot 190 + 21 \cdot 60 + 23 \cdot 90 + 31 \cdot 110 = 2700 + 540 + 1680 + 3420 +$

$+1260 + 2070 + 3410 = 15080 (\text{единиц})$

Ответ: Сравнивая три метода нахождения оптимального плана, делаем вывод, что метод потенциалов находит оптимальный план решения транспортной задачи, так как получили минимальные транспортные издержки равные 15080 единиц.

Задание 2:

1 вариант.

Задача. Имеются три пункта поставки однородного груза **A1, A2, A3** и пять пунктов **B1, B2, B3, B4, B5** потребления этого груза. На пунктах **A1, A2 и A3** находится груз соответственно в количестве ***a1, a2 и a3*** тонн. В пункты **B1, B2, B3, B4, B5** требуется доставить соответственно ***b1, b2, b3, b4, b5*** тонн груза. Расстояние между пунктами поставки и пунктами потребления приведено в таблице:

Пункты поставки	**Пункты потребления**				
	B1	**B2**	**B3**	**B4**	**B5**
A1	D11	D12	D13	D14	D15
A2	D21	D22	D23	D24	D25

A3	D31	D32	D33	D34	D35

Найти такой план закрепления потребителей за поставщиками однородного груза, чтобы общие затраты по перевозкам были минимальными.

a1=200, a2=350, a3=300,
b1=270, b2=130, b3=190, b4=150, b5=110.

$$D = \begin{pmatrix} 24 & 50 & 55 & 27 & 16 \\ 50 & 47 & 23 & 17 & 21 \\ 35 & 59 & 55 & 27 & 41 \end{pmatrix}$$

2 вариант.

Задача. Имеются три пункта поставки однородного груза **A1, A2, A3** и пять пунктов **B1, B2, B3, B4, B5** потребления этого груза. На пунктах **A1, A2 и A3** находится груз соответственно в количестве ***a1, a2 и a3*** тонн. В пункты **B1, B2, B3, B4, B5** требуется доставить соответственно ***b1, b2, b3, b4, b5*** тонн груза. Расстояние между пунктами поставки и пунктами потребления приведено в таблице:

Пункты поставки	**Пункты потребления**				
	B1	**B2**	**B3**	**B4**	**B5**
A1	D11	D12	D13	D14	D15
A2	D21	D22	D23	D24	D25
A3	D31	D32	D33	D34	D35

Найти такой план закрепления потребителей за поставщиками однородного груза, чтобы общие затраты по перевозкам были минимальными.

a1=200, a2=300, a3=250,
b1=210, b2=150, b3=120, b4=135, b5=135.

$$D = \begin{pmatrix} 20 & 10 & 13 & 13 & 18 \\ 27 & 19 & 20 & 16 & 22 \\ 26 & 17 & 19 & 21 & 23 \end{pmatrix}$$

3 вариант.

Задача. Имеются три пункта поставки однородного груза **A1, A2, A3** и пять пунктов **B1, B2, B3, B4, B5** потребления этого груза. На пунктах **A1, A2 и A3** находится груз соответственно в количестве ***a1, a2 и a3*** тонн. В пункты **B1, B2, B3, B4, B5** требуется доставить соответственно ***b1, b2, b3, b4, b5*** тонн груза. Расстояние между пунктами поставки и пунктами потребления приведено в таблице:

Пункты поставки	**Пункты потребления**				
	B1	**B2**	**B3**	**B4**	**B5**
A1	D11	D12	D13	D14	D15
A2	D21	D22	D23	D24	D25
A3	D31	D32	D33	D34	D35

Найти такой план закрепления потребителей за поставщиками однородного груза, чтобы общие затраты по перевозкам были минимальными.

a1=230, a2=250, a3=170,
b1=140, b2=90, b3=160, b4=110, b5=150.

$$D = \begin{pmatrix} 40 & 19 & 25 & 25 & 35 \\ 49 & 26 & 27 & 18 & 38 \\ 46 & 27 & 36 & 40 & 45 \end{pmatrix}$$

4 вариант.

Задача. Имеются три пункта поставки однородного груза **A1, A2, A3** и пять пунктов **B1, B2, B3, B4, B5** потребления этого груза. На пунктах **A1, A2 и A3** находится груз соответственно в количестве ***a1, a2 и a3*** тонн. В пункты **B1, B2, B3, B4, B5** требуется доставить соответственно ***b1, b2, b3, b4, b5*** тонн груза. Расстояние между пунктами поставки и пунктами потребления приведено в таблице:

Пункты поставки	**Пункты потребления**				
	B1	**B2**	**B3**	**B4**	**B5**
A1	D11	D12	D13	D14	D15
A2	D21	D22	D23	D24	D25

A3	D31	D32	D33	D34	D35

Найти такой план закрепления потребителей за поставщиками однородного груза, чтобы общие затраты по перевозкам были минимальными.

a1=350, a2=200, a3=300,

b1=170, b2=140, b3=200, b4=195, b5=145.

$$D = \begin{pmatrix} 22 & 14 & 16 & 28 & 30 \\ 19 & 17 & 26 & 36 & 36 \\ 37 & 30 & 31 & 39 & 41 \end{pmatrix}$$

5 вариант.

Задача. Имеются три пункта поставки однородного груза **A1, A2, A3** и пять пунктов **B1, B2, B3, B4, B5** потребления этого груза. На пунктах **A1, A2 и A3** находится груз соответственно в количестве ***a1, a2 и a3*** тонн. В пункты **B1, B2, B3, B4, B5** требуется доставить соответственно ***b1, b2, b3, b4, b5*** тонн груза. Расстояние между пунктами поставки и пунктами потребления приведено в таблице:

Пункты поставки	**Пункты потребления**				
	B1	**B2**	**B3**	**B4**	**B5**
A1	D11	D12	D13	D14	D15
A2	D21	D22	D23	D24	D25
A3	D31	D32	D33	D34	D35

Найти такой план закрепления потребителей за поставщиками однородного груза, чтобы общие затраты по перевозкам были минимальными.

a1=300, a2=250, a3=200,

b1=210, b2=150, b3=120, b4=135, b5=135.

$$D = \begin{pmatrix} 4 & 8 & 13 & 2 & 7 \\ 9 & 4 & 11 & 9 & 17 \\ 3 & 16 & 10 & 1 & 4 \end{pmatrix}$$

Контрольные вопросы

1. Какие задачи называются транспортными?
2. В чем суть классической транспортной задачи?
3. Что означает термин «транспортный тариф»?
4. Как записывается условие баланса?
5. Как выглядит математическая постановка транспортной задачи?
6. В чем суть метода северо-западного угла?
7. Основная идея метода наименьшей стоимости?
8. В чем суть метода потенциалов?
9. Какие клетки называются потенциальными?
10. Какие виды контуров вы знаете?

Итог работы: отчет, защита работы.

Практическая работа №11

Цель: научиться находить начальное решение методом потенциалов

Задание 1: порешать задачи предыдущего пункта метода потенциалов

Итог работы: отчет, защита работы.

Практическая работа №12

Цель: научиться находить решение линейной краевой задачи методом стрельбы

Задание 1: ознакомиться с материалом

Пример краевой задачи

Примером двухточечной краевой задачи является задача:

$$y'' = f(x, y, y'), \qquad 0 < x < 1,$$

$$y(0) = Y_0, \qquad y(1) = Y_1. \tag{8.1}$$

с граничными условиями на обоих концах отрезка $0 \le x \le 1$, на котором надо найти решение $y = y(x)$. На этом примере мы схематически изложим некоторые способы численного решения краевых задач.

Если функция $f(x, y, y')$ в (8.1) линейна по аргументам y и y', то мы имеем линейную краевую задачу, иначе — нелинейную краевую задачу.

Линейная краевая задача

Рассмотрим частную, но довольно распространенную краевую задачу следующего вида:

$$Ly = y'' - p(x)\,y = f(x), \qquad 0 < x < 1,$$

$$y(0) = Y_0, \qquad y(1) = Y_1. \tag{8.2}$$

Для этой задачи проиллюстрируем два способа решения: один основан на идее численного построения общего решения линейного дифференциального уравнения, другой (конечно-разностный) сводит исходную дифференциальную краевую задачу к системе линейных алгебраических уравнений, решение которой находится методом прогонки.

Метод численного построения общего решения

Для нахождения решения краевой задачи (8.2) можно численно построить решение дифференциального уравнения, представимое в виде

$$y(x) = C_1 y_1(x) + C_2 y_2(x) + y_0(x),$$

где $y_0(x)$ — какое-либо решение неоднородного уравнения

$$y'' - p(x)\,y = f(x),$$

а $y_1(x)$ и $y_2(x)$ — два любые линейно независимые решения однородного уравнения $y'' - p(x)\,y = 0$. Постоянные C_1 и C_2 находятся из граничных условий задачи (8.2).

Так как решения $y_0(x),\ y_1(x),\ y_2(x)$ произвольны, то их можно построить различными способами. Например, можно задать какие-то начальные условия и решить одну задачу Коши для неоднородного и две задачи Коши для однородного уравнений. Эти условия, в частности, могут быть такими:

$y_0(0) = 0, \quad y_0'(0) = 0$ — для неоднородного уравнения;

$y_1(0) = 1, \quad y_1'(0) = 0;$

$y_2(0) = 0, \quad y_2'(0) = 1$ — для однородного уравнения.

Однако при реализации этого способа, например, в случае $p(x) \gg 1$ для рассматриваемого уравнения могут возникнуть трудности, связанные с неустойчивостью задачи Коши. В этом случае можно попытаться построить $y_0(x),\ y_1(x),\ y_2(x)$ с помощью решения одной краевой задачи для неоднородного уравнения и двух краевых задач для однородного уравнения. Краевые условия для этих задач могут быть, например, следующими:

$y_0(0) = 0, \quad y_0'(1) = 0$ — для неоднородного уравнения;

$y_1(0) = 1, \quad y_1'(1) = 0;$

$y_2(0) = 0, \quad y_2'(1) = 1$ — для однородного уравнения.

Эти задачи могут быть решены методом прогонки. Условия устойчивости метода прогонки при $p(x) > 0$, как легко проверить, выполнены. Этот подход может оказаться полезным, если краевые условия таковы, что для исходной задачи (8.2) метод прогонки применен быть не может.

Отметим, что с учетом специфики краевых условий исходной задачи можно строить общее решение вида

$$y(x) = y_0(x) + Cy_1(x),$$

где $y_0(x)$ — некоторое решение неоднородного уравнения, а $y_1(x)$ — некоторое решение однородного уравнения.

Конечно-разностный метод (метод прогонки)

При нахождении решения линейной краевой задачи:

$$y'' - p(x)y = f(x), \qquad 0 < x < 1,$$

$$y(0) = Y_0, \qquad y(1) = Y_1.$$

для $p(x) >> 1$ методом построения общего решения, если оно находится с помощью решения задач Коши, могут возникнуть трудности, связанные с вычислительной неустойчивостью задачи Коши.

Для решения поставленной задачи можно воспользоваться разностной схемой:

$$\frac{y_{m+1} - 2y_m + y_{m-1}}{h^2} - p(x_m)y_m = f(x_m),$$

$$0 < m < M, \qquad Mh = 1, \qquad y_0 = Y_0, \qquad y_M = Y_1$$

и решить разностную задачу методом прогонки. Условия применимости метода прогонки при $p(x) > 0$, как легко проверить, выполнены. Подробнее о методе прогонки см. в [1–4, 17, 31]. В [17] рассмотрены различные варианты метода прогонки.

Нелинейная краевая задача

Краевая задача

$$y'' = f(x, y, y'), \qquad 0 < x < 1,$$

$$y(0) = Y_0, \qquad y(1) = Y_1. \qquad (8.3)$$

является нелинейной краевой задачей, если функция $f(x, y, y')$ нелинейна хотя бы по одному из аргументов y или y'.

В настоящей работе реализованы два способа решения нелинейных краевых задач: *метод стрельбы* и *метод линеаризации* (метод Ньютона), который сводит решение нелинейной краевой задачи к решению серии линейных краевых задач.

2.6. Метод стрельбы

Метод стрельбы для решения краевой задачи (8.3) базируется на том, что имеются удобные способы численного решения задачи Коши, т. е. задачи следующего вида

$$y'' = f(x, y, y'), \quad 0 < x < 1,$$

$$y(0) = Y_0, \qquad (8.4)$$

$$y'(0) = \operatorname{tg}\alpha,$$

где Y_0 — ордината точки $(0, Y_0)$, из которой выходит интегральная кривая; α — угол наклона интегральной кривой к оси x при выходе из точки $(0, Y_0)$ (рис. 8). При фиксированном Y_0 решение задачи (8.4) имеет вид $y = y(x, \alpha)$. При $x = 1$ решение $y(x, \alpha)$ зависит только от α:

$$y(x, \alpha)\big|_{x=1} = y(1, \alpha).$$

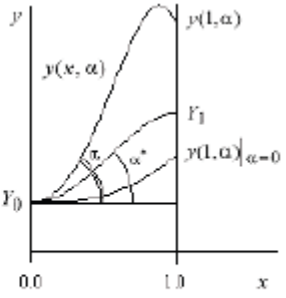


Рис. 8

Используя указанное замечание о решении задачи Коши (8.4), можно задачу (8.3) переформулировать следующим образом: найти такой угол $\alpha = \alpha^*$, при котором интегральная кривая, выходящая из точки $(0, Y_0)$ под углом α к оси абсцисс, попадет в точку $(1, Y_1)$:

$$y(1, \alpha) = Y_1. \qquad (8.5)$$

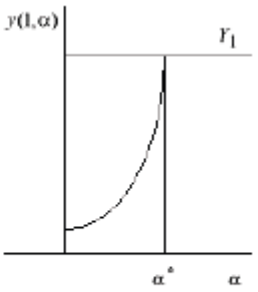


Рис. 9

Решение задачи (8.4) при этом $\alpha = \alpha^*$ совпадает с искомым решением задачи (8.3). Таким образом, дело сводится к решению

уравнения (8.5) (рис. 9). Уравнение (8.5) — это уравнение вида

$$F(\alpha) = 0,$$

где $F(\alpha) = y(1, \alpha) - Y_1$.

Оно отличается от привычных уравнений лишь тем, что функция $F(\alpha)$ задана не аналитическим выражением, а с помощью алгоритма численного решения задачи (8.4).

Для решения уравнения (8.5) можно использовать любой метод, пригодный для уточнения корней нелинейного уравнения, например, метод деления отрезка пополам, метод Ньютона (касательных) и др. Метод Ньютона здесь предпочтительнее (если имеется достаточно хорошее начальное приближение) из-за высокой стоимости вычисления одного значения функции $F(\square)$ (нужно решить задачу Коши (8.4) с данным $\square$).

Метод стрельбы, сводящий решение краевой задачи (8.3) к вычислению решений задачи Коши (8.4), хорошо работает в том случае, если решение $y(x, \alpha)$ «не слишком сильно» зависит от $\square$. В противном случае он становится вычислительно неустойчивым, даже если решение задачи (8.3) зависит от входных данных «умеренно».

При решении уравнений $F(\alpha) = 0,$ методом деления отрезка пополам, мы задаем α_0 и α_1 так, чтобы разности $y(1, \alpha_0) - Y_1$ и $y(1, \alpha_1) - Y_1$ имели разные знаки. Затем полагаем

$$\alpha_2 = \frac{\alpha_0 + \alpha_1}{2}.$$

Вычисляем $y(1, \alpha_2)$. Затем вычисляем α_3 по одной из формул:

$$\alpha_3 = \frac{\alpha_1 + \alpha_2}{2} \quad \text{или} \quad \frac{\alpha_0 + \alpha_2}{2}$$

в зависимости от того, имеют ли разности $y(1, \alpha_2) - Y_1$ и $y(1, \alpha_1) - Y_1$ соответственно разные или одинаковые знаки. Затем вычисляем $y(1, \alpha_3)$. Процесс продолжаем до тех пор, пока не будет достигнута требуемая точность $| y(1, \alpha_n) - Y_1 | < \varepsilon$.

В случае использования для решения уравнения $F(\alpha) = 0$ метода Ньютона задаем α_0, а затем последующие α_n вычисляем по рекуррентной формуле

$$\alpha_{n+1} = \alpha_n - \frac{F(\alpha_n)}{F'(\alpha_n)}, \qquad n = 0, 1, \ldots$$

Производная $F'(\alpha_n)$ может быть вычислена по одной из формул численного дифференцирования, например, первого порядка аппроксимации:

$$F'(\alpha_n) \approx \frac{F(\alpha_n + h) - F(\alpha_n)}{h}.$$

Вычислительная неустойчивость задачи Коши

Поясним причину возникновения вычислительной неустойчивости на примере следующей линейной краевой задачи:

$$y'' - p^2 y = 0, \qquad 0 < x < 1,$$
$$y(0) = Y_0, \qquad y(1) = Y_1. \tag{8.6}$$

при постоянном p^2. Выпишем решение этой задачи:

$$y(x) = \frac{e^{-px} - e^{-p(2-x)}}{1 - e^{-2p}} Y_0 + \frac{e^{-p(1-x)} - e^{-p(1+x)}}{1 - e^{-2p}} Y_1.$$

Коэффициенты при Y_0 и Y_1 с ростом p остаются ограниченными на отрезке $0 \le x \le 1$ функциями; при всех $p > 0$ они не превосходят единицу. Поэтому небольшие ошибки при задании Y_0 и Y_1 ведут к столь же небольшим погрешностям в решении, т. е. краевая задача является «хорошей».

Рассмотрим теперь задачу Коши:

$$y'' - p^2 y = 0, \qquad 0 < x < 1,$$
$$y(0) = Y_0, \qquad y'(0) = \operatorname{tg}\alpha. \tag{8.7}$$

Ее решение имеет вид:

$$y(x) = \frac{pY_0 + \operatorname{tg}\alpha}{2p} e^{px} + \frac{pY_0 - \operatorname{tg}\alpha}{2p} e^{-px}.$$

Если при задании $\operatorname{tg}\alpha$ допущена погрешность □, то значение решения при $x = 1$ получит приращение

$$\Delta y(1) = \frac{\varepsilon}{2p} e^{p} - \frac{\varepsilon}{2p} e^{-p}. \qquad (8.8)$$

При больших p вычитаемое в равенстве (8.8) пренебрежимо мало, но коэффициент в первом слагаемом $e^{p}/2p$ становится большим. Поэтому метод стрельбы при решении задачи (8.6), будучи формально приемлемой процедурой, при больших p становится практически непригодным. Подробнее о возникновении неустойчивостей см. [1, 2].

Метод линеаризации (метод Ньютона)

Метод Ньютона сводит решение нелинейной краевой задачи к решению серии линейных краевых задач и состоит в следующем.

Пусть для нелинейной краевой задачи (8.3) известна функция $y_0(x)$, удовлетворяющая граничным условиям и грубо приближенно равная искомому $y(x)$. Положим

$$y(x) = y_0(x) + v(x), \qquad (8.9)$$

где $v(x)$ — поправка к нулевому приближению $y_0(x)$. Подставим (8.9) в уравнение (8.8) и линеаризуем задачу, используя следующие равенства:

$$y''(x) = y_0''(x) + v''(x),$$

$$f(x, y_0 + v, y_0' + v') = f(x, y_0, y_0') +$$

$$+ \frac{\partial f(x, y_0, y_0')}{\partial y} v + \frac{\partial f(x, y_0, y_0')}{\partial y'} v' + O(v^2 + |v'|^2).$$

Отбрасывая остаточный член $O(v^2 + |v'|^2)$, получим линейную краевую задачу для нахождения поправки $\tilde{v}(x)$:

$$\tilde{v}'' = p(x)\tilde{v}' + g(x)\tilde{v} + r(x),$$

$$\tilde{v}(0) = 0, \qquad \tilde{v}(1) = 0, \qquad (8.10)$$

где

$$p(x) = \frac{\partial f(x, y_0, y_0')}{\partial y'}, \qquad q(x) = \frac{\partial f(x, y_0, y_0')}{\partial y},$$

$$r(x) = f(x, y_0, y_0') - y_0''.$$

Решая линейную краевую задачу (8.10) каким-либо численным методом, найдем поправку $\tilde{v}$ и примем за первое приближение

$$y_1(x) = y_0(x) + \tilde{v}.$$

Аналогично, зная приближение $y_1(x)$, положим $y(x) = y_1(x) + \tilde{v}_1$ и найдем следующее приближение. Продолжая процесс до тех пор, пока не будут выполнены неравенства

$$\max |\tilde{v}(x)| \le \varepsilon, \qquad x \in [0, 1],$$

где □ — требуемая точность, найдем приближенное решение исходной нелинейной задачи.

Задание 2:

1. Начните выполнение работы с темы «*Линейная краевая задача*». Выбрав с помощью меню один из методов решения линейной краевой задачи, перейдите к пункту меню «*Параметры*». Наберите следующую краевую задачу:

$y'' - py = -p, \quad 0 < x < 1,$

$y(0) = 1, \quad y(1) = 1.$

для $p = \text{const} > 0$. Решением этой задачи является функция $y = 1$. Установите значение шага сетки $h = 0{,}05$.

1.1. Найдите решение этой задачи методом построения общего решения и методом прогонки для разных p, начиная с умеренных значений и увеличивая их до величины порядка 1200. Сравните получаемые решения с точным и объясните наблюдаемые эффекты. Попытайтесь найти решение этой же задачи методом стрельбы. Проанализируйте, как влияет при разных p точность задания недостающего начального условия на левом конце интервала на успешное решение задачи методом стрельбы.

1.2. Объясните полученные результаты. Замените левое краевое условие (положите, например, $y'(0) = 0$) и посмотрите, как изменится характер решения.

1.3. Выполните п. 1.1, 1.2 для задачи:

$y'' + py = p, \quad 0 < x < 1,$

$y(1) = 1, \quad y'(0) = 0.$

Ее точное решение $y = 1$. Объясните полученные результаты. Найдите условие устойчивости метода прогонки для данной задачи.

2. Получите численное решение следующих нелинейных краевых задач:

2.1. $y'' + px\cos y = 0, \quad 0 < x < 1,$

$y'(0) = 0, \quad y(1) = 0, \quad p = 1, 4, 7, 25, 50, 100;$

2.2. $y'' + \frac{0{,}5}{1 - 0{,}5y} y'^2 = 0, \quad 0 < x < 1,$

$y(0) = y_0, \quad y(1) = 0,$

$y_0 = 0{,}25;\ 0{,}5;\ 1;\ 1{,}5;\ 1{,}8;\ 1{,}9;\ 1{,}95;$

2.3. $y'' + \sin y = 0, \quad 0 < x < x_k,$

$y(0) = 0, \quad y(x_k) = \pi,$

$x_k = 0{,}5;\ 1;\ 2;\ 4;\ 6.$

3. Рассмотрите следующие краевые задачи:

3.1. $y'' = e^y, \quad 0 < x < 1,$

$y(0) = 1, \quad y(1) = a;$

3.2. $y'' = -e^y, \quad 0 < x \le 1,$

$y(0) = 1, \quad y(1) = a.$

Параметр a меняется от 0 до 2. Что при этом происходит с решением задач? Почему в задаче 3.2 при значениях $a > 1{,}4999\ldots$ не работает метод линеаризации?

Замечание. Задача 3 подробно рассмотрена в [29].

4. Рассмотреть две сингулярно-возмущенные задачи (с малым параметром при старших производных):

$\varepsilon y'' = y(y^2 - 1), \quad -1 < x < 1,$

$y(-1) = y(1) = \sqrt{2}$

и

$\varepsilon y'' = -y(y^2 - 1), \quad -1 < x < 1,$

$y(-1) = y(1) = \sqrt{2}.$

Считаем $\varepsilon = 10^{-3}$. Какие численные методы позволят получить решение каждой из этих задач? Почему?

Итог работы: отчет, защита работы.

Практическая работа №13

Цель: отработать навыки решения практических задач методом динамического программирования

Задание 1:

Динамическое программирование – метод оптимизации, приспособленный, к задачам, в которых процесс принятия решения может быть разбит на отдельные этапы (шаги). Такие задачи называются многошаговыми.

Характерные особенности задач динамического программирования:

1) Неоднозначность решения.
2) Возможность деления вычислительного процесса на этапы.
3) Общий критерий – сумма частных критериев на этапах.

Динамическое программирование позволяет осуществлять оптимальное планирование многошаговых процессов, зависящих от времени. Процесс называется управляемым, если можно влиять на ход его развития. Управлением называется совокупность решений, принимаемых на каждом этапе для влияния на ход процесса. Началом этапа (шага) управляемого процесса считается момент принятия решения. Планируя многошаговый процесс, исходят из интересов всего процесса в целом, всегда необходимо иметь в виду конечную цель.

Метод динамического программирования состоит в том, что оптимальное управление строится постепенно. На каждом этапе оптимизируется управление только этого этапа, причем управление выбирается с учётом последствий, т.е. оптимальное управление для данного этапа должно учитывать весь последующий ход процесса, для чего необходимо знать все управления на последующих этапах. Поскольку процесс заканчивается на последнем этапе, оптимальное решение не должно учитывать последующего управления. Таким образом, процесс вычисления протекает в обратном направлении, от конца к началу.

Постановка задачи динамического программирования.

Пусть S_0, S_k, S_n - состояния системы на начальном, *k-ом* и конечном этапе, u_0, u_k, u_n - управления системой на начальном, *k-ом* и конечном этапах. Управление u_k переводит систему из состояния S_{k-1} в состояние S_k. Показатель эффективности на *k-ом* этапе обозначим через $W_k(S_{k-1})$.

Так как оптимизацию показателя эффективности начинаем с последнего этапа, то, зная максимум показателя эффективности на *n*-ом шаге

$$W_n^*(S_{n-1}) = \max_{u_n}(W_n(S_{n-1}, u_n))$$

найдем максимум показателя эффективности на *(n-1)*- ом шаге

$$W_{n-1}^*(S_{n-2}) = \max_{u_{n-1}}(W_{n-1}(S_{n-2}, u_{n-1}) + W_n^*(S_{n-1}))$$

,

где $W_{n-1}(S_{n-2}, u_{n-1})$ называется "сиюминутной" выгодой на *(n-1)*- ом шаге.

Основное функциональное уравнение динамического программирования (уравнение Беллмана) имеет вид:

$$W_k^*(S_{k-1}) = \max_{u_k}(W_k(S_{k-1}, u_k) + W_{k+1}^*(S_k))$$

Принцип оптимальности Беллмана можно сформулировать следующим образом: каковы бы не были начальное состояние и начальное решение, последующее решение должно быть оптимальным по отношению к состоянию, полученному в результате начального решения. Иными словами, принцип оптимальности утверждает, что если в данный момент выбрано не наилучшее решение, то последствия этого нельзя исправить в будущем.

Задача определения кратчайших расстояний по заданной сети

Пусть дано конечное число точек $P_1, P_2, ..., P_n$, соединенных всевозможными отрезками линий, называемых звеньями или связями. Тогда совокупность точек и их связей называют сетью. Сеть называется достаточно связанной, если существует путь, состоящий из звеньев и соединяющий любые две точки сети. Пусть каждому звену поставлено в соответствие

действительное неотрицательное число l_{ij} - его длина. Необходимо определить кратчайшее расстояние по сети от каждой точки до всех остальных и соответствующие пути, по которым, они проходят. Пронумеруем точки сети в любом порядке и укажем длину каждого звена. Две точки называются соседними, если они непосредственно соединены связью. Положим, $l_{ij} = l_{ji}$.

Для решения задачи используем метод динамического программирования и отыскиваем кратчайшее расстояние не от фиксированной точки до всех остальных, а от всех остальных до фиксированной через соседние точки. Связь, через которую проходит кратчайшее расстояние, после каждого шага отмечаем стрелкой. Для удобства точки сети обозначим кружками с номерами точек.

Алгоритм решения:

1. Фиксируем конечную точку P_{ii}, до которой необходимо рассчитать кратчайшее расстояние от всех остальных, и рядом с этой точкой записываем нуль, т.к. расстояние P_i от точки до ней самой равно 0.Это число, отличное от нуля, для других точек, назовём характеристикой точки.
2. Определим соседние точки по формуле $c_{ij} = 0 + l_{ij}$ и на связях, соединяющих эти точки, поставим стрелки, направленные в точку P_i. После этого точку P_i отметим символом v, обозначающим, что операции над ней закончены.
3. Переходим к любой соседней точке, для которой характеристика уже найдена. Пусть это будет точка P_j. Определяем соседние с ней точки и подсчитываем характеристики этих точек по формуле $c_{ji} = c_{ij} + l_{ji}$.
4. При определении c_{ji} для соседних P_j может оказаться, что для некоторых из них характеристики c_{ij} уже известны. В этом случае новую характеристику c_{ji} сравниваем со старой характеристикой c_{ij}.

   Если $c_{ji} \geq c_{ij}$, то старую характеристику оставляем без изменений.

   Если $c_{ji} < c_{ij}$, то старую характеристику заменяем на новую, соответственно происходит или не происходит изменение направления.

   Точку P_j отмечаем символом v, если соседняя точка, у которой изменилась характеристика, не была ранее отмечена v. Если же точка ранее была отмечена символом v, то пересчитываем характеристики соседних с ней точек.
5. Переходим к пункту 3.
6. Процесс продолжаем до тех пор, пока не будут отмечены символом v все точки сети. Ответ выписываем в виде таблицы, где указаны кратчайшее расстояние от всех точек до конечной и пункты, через которые они проходят.

Порядок выполнения заданий

Задача 1. Двум предприятиям А и В на 4 квартала выделено $S_0 = 1000$ единиц средств. Каждый квартал предприятие А получает х средств, предприятие В - у средств. При этом от выделенных средств предприятие А получает 5х единиц и остаток средств 0,3х единиц, а предприятие В - доход 4у единиц и остаток выделенных средств 0,5у единиц. Необходимо распределить средства между предприятиями поквартально таким образом, чтобы за весь год оба предприятия получили максимальный доход.

Решение. Период времени 1 год разделим на 4 квартала (4 этапа).

Введем обозначения: через x_i, y_i обозначим вклад в развитие предприятий А и В в 1-ом квартале, W_i - доход за i-ый квартал, S_i - остаток средств на конец i-ого квартала, i – 1,2,3,4.

№	Состояние	Вклад		Доход	Остаток
		A	B		
1	$S_0 - S_1$	x_1	y_1	W_1	S_1
2	$S_1 - S_2$	x_2	y_2	W_2	S_2
3	$S_2 - S_3$	x_3	y_3	W_3	S_3
4	$S_3 - S_4$	x_4	y_4	W_4	S_4

С учетом введенных обозначений составим подробную таблицу по этапам.

Предприятие	1 квартал			2 квартал			3 квартал			4 квартал	
	вклад	доход	остаток	вклад	доход	остаток	вклад	доход	остаток	вклад	доход
A	x_1	$5x_1$	$0{,}3x_1$	x_2	$5x_2$	$0{,}3x_2$	x_3	$5x_3$	$0{,}3x_3$	x_4	$5x_4$
B	y_1	$4y_1$	$0{,}5y_1$	y_2	$4y_2$	$0{,}5y_2$	y_3	$4y_3$	$0{,}5y_3$	y_4	$4y_4$
	$S_0=x_1+y_1$	$W_1=5x_1+4y_1$	$S_1=0{,}3x_1+0{,}5y_1$	$S_1=x_2+y_2$	$W_2=5x_2+4y_2$	$S_2=0{,}3x_2+0{,}5y_2$	$S_2=x_3+y_3$	$W_3=5x_3+4y_3$	$S_3=0{,}3x_3+0{,}5y_3$	$S_3=x_4+y_4$	$W_4=5x_4+4y_4$

Отыскание оптимального управления начнем с 4 квартала.

$$W_4^* = \max W_4 = \max(5x_4 + 4y_4) = \left\{ \begin{array}{c} x_4 + y_4 = S_3, \\ x_4 = S_3 - y_4, S_3 = const \end{array} \right\} = \max(5(S_3 - y_4) + 4y_4) =$$

$$= \max_{0 \le y_4 \le S_3} (5S_3 - y_4) = (5S_3 - y_4)\Big|_{y_4=0} = 5S_3.$$

3 квартал.

$$W_{3-4}^* = \max(W_3 + W_4^*) = \max(5x_3 + 4y_3 + 5S_3) = \left\{ \begin{array}{l} x_3 + y_3 = S_2, \quad x_3 = S_2 - y_3, \quad S_3 = 0{,}3x_3 + \\ +0{,}5y_3 = 0{,}3S_2 - 0{,}3y_3 + 0{,}5y_3 = 0{,}3S_2 + 0{,}2y_3, \\ S_2 = const \end{array} \right\} =$$

$$= \max_{0 \le y_3 \le S_2} (5S_2 - 5y_3 + 4y_3 + 1{,}5S_2 + y_3) = \max_{0 \le y_3 \le S_2} (6{,}5S_2) = 6{,}5S_2.$$

Так как максимум дохода за 3-4 кварталы постоянен при любом распределении средств, то пусть $x_3 = \frac{S_2}{2}, \quad y_3 = \frac{S_2}{2}$.

2 квартал.

$$W_{2-4}^* = \max(W_2 + W_{3-4}^*) = \max(5x_2 + 4y_2 + 6{,}5S_2) = \left\{ \begin{array}{l} x_2 + y_2 = S_1, \quad x_2 = S_1 - y_2, \quad S_2 = 0{,}3x_2 + \\ +0{,}5y_2 = 0{,}3S_1 - 0{,}3y_2 + 0{,}5y_2 = 0{,}3S_1 + 0{,}2y_2, \\ S_1 = const \end{array} \right\} =$$

$$= \max_{0 \le y_2 \le S_1} (5S_1 - 5y_2 + 4y_2 + 1{,}95S_1 + 1{,}3y_2) = \max_{0 \le y_2 \le S_1} (6{,}95S_1 + 0{,}3y_2) = (6{,}95S_1 + 0{,}3y_2)\Big|_{y_2 = S_1} = 7{,}25S_1$$

1 квартал.

$$W^*_{1-4} = \max(W_1 + W^*_{2-4}) = \max(5x_1 + 4y_1 + 7{,}25S_1) = \left\{\begin{matrix} x_1 + y_1 = S_0, \quad x_1 = S_0 - y_1, \quad S_1 = 0{,}3x_1 + \\ +0{,}5y_1 = 0{,}3S_0 - 0{,}3y_1 + 0{,}5y_1 = 0{,}3S_0 + 0{,}2y_1, \\ S_0 = const \end{matrix}\right\} =$$

$$= \max_{0 \le y_1 \le S_0} (5S_0 - 5y_1 + 4y_1 + 2{,}175S_0 + 1{,}45y_1) = \max_{0 \le y_1 \le S_0} (7{,}175S_0 + 0{,}45y_1) =$$

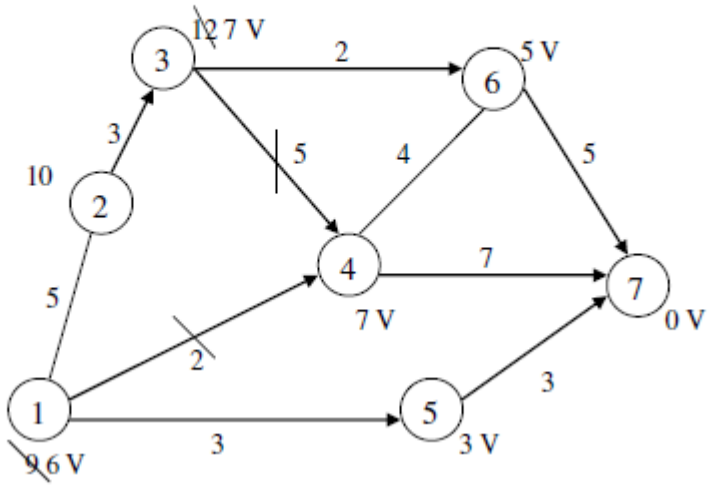
$$(7{,}175S_0 + 0{,}45y_1)\Big|_{y_1 = S_0} = 7{,}625S_0$$

По условию задачи $S_0 = 1000$ единиц, $W^*_{1-4} = 7625$ единиц, при этом будем иметь следующие распределение средств по кварталам:

Квартал	Распределяемые средства	Вклады	
		A	B
1	$S_0 = 1000$	$x_1^* = 0$	$y_1^* = 1000$
2	$S_1 = 0{,}3x_1^* + 0{,}5y_1^* = 500$	$x_2^* = 0$	$y_2^* = 500$
3	$S_2 = 0{,}3x_2^* + 0{,}5y_2^* = 250$	$x_3^* = 125$	$y_3^* = 125$
4	$S_3 = 0{,}3x_3^* + 0{,}5y_3^* = 100$	$x_4^* = 100$	$y_4^* = 0$

Задача 2. Дана сеть, состоящая из 7 точек, и известны расстояния между точками. Необходимо определить кратчайшее расстояние от любой точки до точки 7.

Решение.



1. Рассмотрим точку 7. Рядом с кружком ставим 0 характеристику этой точки.
2. Соседними с точкой 7 являются точки 6,5,4. Подсчитаем характеристики этих точек и укажем направления. Точку 7 отмечаем символом V , т.к. операции на ней закончены.
3. Рассмотрим точку 4. Соседними с ней будут точки 6,3,1,7. Находим характеристики каждой из них. Характеристики точек 1 и 3 – соответственно 9 и 12. Характеристики точек 6,7 остались без изменения, так как 7+4=11>5, 7+7=14>0. Точку 4 отметим символом V. Рассмотрим точку 6. Соседними являются точки 3,4,7. Для точки 3 новая характеристика 5+2=7>12, поэтому изменяем старую характеристику 12 на 7, и указываем новое направление. Для точек 4,7 старые характеристики остаются без изменений, т.к. 5+4=9>7, 5+5=10>0. Точку 6 отмечаем знаком V. Рассмотрим точку 5. Соседняя с ней точка 1. Новая характеристика 3+3=6<9, поэтому изменяем характеристику и направление. Точку 5 отмечаем символом V. Точка 1, характеристика которой изменилась, является соседней с точкой 4. Точка 4 отмечена символом V,

поэтому пересчитываем характеристику этой точки и проверяем соседние с ней: 7+5=12>7; 7+4=11>5; 7+7=14>0. Характеристики точек 3,6,7 остаются без изменений.

4. Рассмотрим точку 3. Соседними являются точки 2,4,6. Характеристика 2: 7+3=10, записываем эту характеристику и указываем направление. Характеристики 6,4 остались без изменения. Точку 3 отмечаем символом V.
5. Рассмотрим точку 2. Соседними являются точки 1 и 3. Характеристики точек не изменяются, т.к. 10+5=15>6, 10+3=13>7. Точку 2 отмечаем символом V.
6. Рассмотрим точку 1. Соседними являются точки 2,4,5. Характеристики точек не изменились, т.к. 6+5=11>10, 6+2=8>4, 6+3=9>3. Операции над всеми точками закончены. Ответ запишем в виде таблицы.

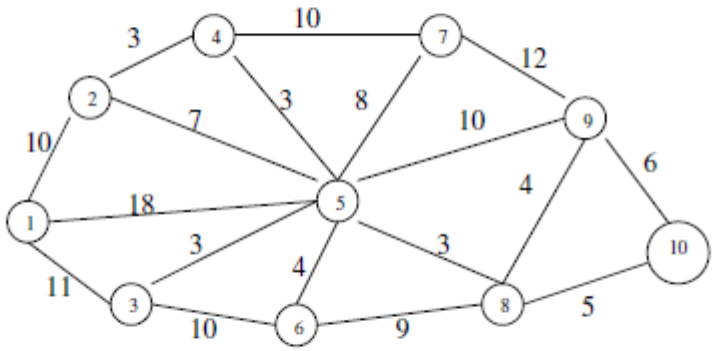
Номера точек, между которыми рассчитывается расстояние	Кратчайшее расстояние	Маршрут, по которому проходит кратчайшее расстояние
1-7	6	1-5-7
2-7	10	2-3-6-7
3-7	7	3-6-7
4-7	7	4-7
5-7	3	5-7
6-7	5	6-7
7-7	0	

Задание 2:

1 вариант.

Задача 1. Планируется работа двух отраслей производства А и В на 4 года. Количество х средств, вложенных в отрасль А, позволяет получить доход 2х и уменьшается до 0,6х. Количество у средств, вложенных в отрасль В, позволяет получить доход 3у и уменьшается до 0,2у. Необходимо распределить выделенные ресурсы в количестве $S_0 = 850$ единиц между отраслями по годам планируемого периода для получения максимальной прибыли за весь период.

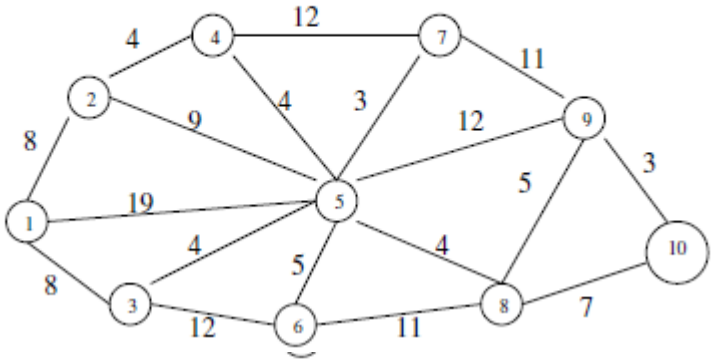
Задача 2. По заданной схеме, соединяющей 10 точек, найти кратчайшее расстояние от 1 точки до 10.



2 вариант.

Задача 1. Двум предприятиям А и В на 4 квартала выделено $S_0 = 900$ единиц средств. Каждый квартал предприятие А получает х средств, предприятие В - у средств. При этом от выделенных средств предприятие А получает 4х единиц и остаток средств 0,3х единиц, а предприятие В - доход 5у единиц и остаток выделенных средств 0,1у единиц. Необходимо распределить средства между предприятиями поквартально таким образом, чтобы за весь год оба предприятия получили максимальный доход.

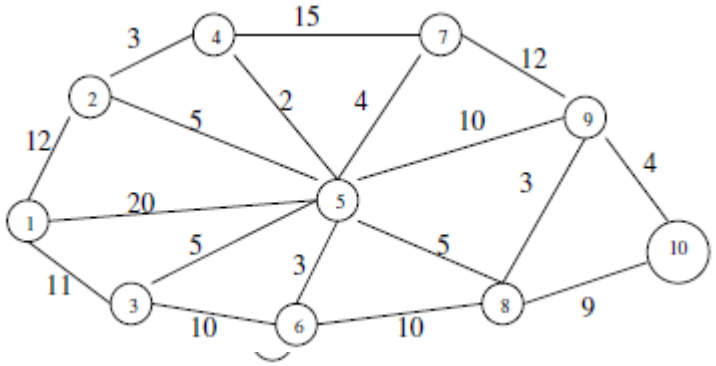
Задача 2. По заданной схеме, соединяющей 10 точек, найти кратчайшее расстояние от 1 точки до 10.



3 вариант.

Задача 1. Планируется работа двух отраслей производства А и В на 4 года. Количество х средств, вложенных в отрасль А, позволяет получить доход 5х и уменьшается до 0,1х. Количество у средств, вложенных в отрасль В, позволяет получить доход 3у и уменьшается до 0,5у. Необходимо распределить выделенные ресурсы в количестве $S_0 = 1100$ единиц между отраслями по годам планируемого периода для получения максимальной прибыли за весь период.

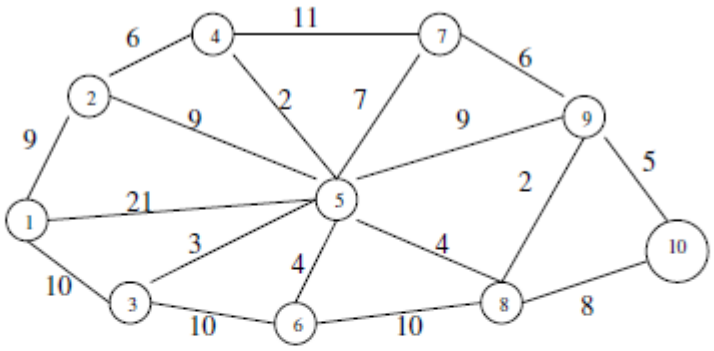
Задача 2. По заданной схеме, соединяющей 10 точек, найти кратчайшее расстояние от 1 точки до 10.



4 вариант.

Задача 1. Двум предприятиям А и В на 4 квартала выделено $S_0 = 750$ единиц средств. Каждый квартал предприятие А получает х средств, предприятие В - у средств. При этом от выделенных средств предприятие А получает 4х единиц и остаток средств 0,3х единиц, а предприятие В - доход 3у единиц и остаток выделенных средств 0,6у единиц. Необходимо распределить средства между предприятиями поквартально таким образом, чтобы за весь год оба предприятия получили максимальный доход.

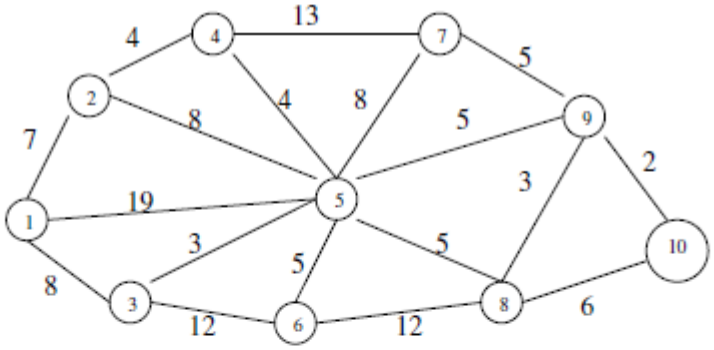
Задача 2. По заданной схеме, соединяющей 10 точек, найти кратчайшее расстояние от 1 точки до 10.



5 вариант.

Задача 1. Планируется работа двух отраслей производства А и В на 4 года. Количество x средств, вложенных в отрасль А, позволяет получить доход 5x и уменьшается до 0,3x. Количество y средств, вложенных в отрасль В, позволяет получить доход 6y и уменьшается до 0,1y. Необходимо распределить выделенные ресурсы в количестве $S_0 = 800$ единиц между отраслями по годам планируемого периода для получения максимальной прибыли за весь период.

Задача 2. По заданной схеме, соединяющей 10 точек, найти кратчайшее расстояние от 1 точки до 10.



Контрольные вопросы

1. Что называется динамическим программированием?
2. Какие характерные особенности задач динамического программирования вы знаете?
3. Что называется управлением?
4. В чем состоит метод динамического программирования?
5. Сформулируйте принцип оптимальности Беллмана?
6. Что называется сетью, звеньями?
7. Что такое характеристика точки?
8. Опишите алгоритм решения задачи определения кратчайшего расстояния по заданной сети?

Итог работы: отчет, защита работы.

Практическая работа №14

Цель: отработать навыки решения практических задач методом динамического программирования, научиться выбирать оптимальную стратегию поведения

Задание 1:

1. Понятие задачи динамического программирования

Рассматриваемые ранее задачи характеризуются тем, что в них не учитываются изменения оптимизируемых параметров во времени – процессы считаются статичными. Выбирается некоторый период времени, и для него определяются проектируемые или планируемые значения показателей. При этом предполагается, что управляемые или неуправляемые параметры системы в течение всего планового времени не будут изменяться или, по крайней мере, не претерпят серьёзных изменений, требующих пересмотра принятых решений.

Однако в реальной жизни есть задачи, в которых необходимо учитывать изменения параметров систем во времени. Эти параметры могут меняться непрерывно или дискретно – от этапа к этапу. Например, из года в год меняется возраст машин и оборудования, изменяется производственная мощность и производительность труда на предприятиях. Очевидно, что необходимо принимать оптимальные решения на год (или другой срок) и одновременно на весь рассматриваемый период в целом с учётом возможных изменений параметров. Для решения такого вида задач, которые получили название **многошаговые**, разработан соответствующий математический аппарат, который получил название **динамическое программирование**.

Задача может быть сформулирована следующим образом:

Задача динамического программирования – определить u_i^* (u_i^* не только число, а может быть вектором, функцией) на каждом шаге, i = 1, 2 …, m, и тем самым u^* всей операции в целом.

Рассмотрим подход к решению данной задачи. Характерным для динамического программирования является то, что переменные рассматриваются вместе, а не последовательно – одна за другой. При этом вычислительная тема строится таким образом, что вместо одной задачи с n переменными решается серия задач с небольшим числом, а чаще с одной переменной. Сам же вычислительный процесс производится на основе метода последовательных приближений в два круга:

1. *От последнего шага к первому.*
2. *От первого шага к последнему или же наоборот, в зависимости от исходных данных.*

На первом круге ищется так называемое условное оптимальное решение. Оно выбирается так, чтобы все предыдущие шаги обеспечили максимальную эффективность последующего. Основу такого подхода составляет принцип оптимальности Беллмана, который формулируется следующим образом:

Нельзя получить оптимальное значение целевой функции i-шагового процесса, если для любого u_i, выбранного на шаге i, значение целевой функции для оставшихся i-1 шагов не является оптимальным при этом выбранном на i-шаге значении u_i.

Такой процесс продолжается до тех пор, пока решение не потеряет свой условный характер, т. е. до первого шага или последнего. Для него решение просто оптимально. Поэтому второй круг начинают именно с этого шага и последовательно переходят от условных к оптимальным решениям, тем самым обеспечивается оптимальность операции в целом.

Оптимальное распределение ресурсов

Пример:

Капитал 40 млн.руб. инвестор должен вложить в четыре инвестиционных проекта так, чтобы получить максимальный доход. Доходность проектов дана в таблице (вложения кратны 8 млн. руб.)

u	**Прибыль от внедрения**			
	f4(u)	**f3(u)**	**f2(u)**	**f1(u)**
8	**55**	**39**	35	32
16	58	53	76	68
24	90	80	**120**	115

32	100	120	135	134
40	140	145	158	147

Решение:

Это задача динамического программирования. Решение состоит из двух этапов. На первом этапе (от конца к началу) ищем условное оптимальное решение, на втором (от начала к концу) – ищем оптимальное решение задачи.

1 этап.

Распределяем капитал между четырьмя проектами и считаем получаемую прибыль $L(i)$, i=8,16,24,32,40.

1 шаг: Денежные средства вкладываются в четвертый проект.

L(8)=**55**

L(16)=58

L(24)=90

L(32)=100

L(40)=140

2 шаг: Денежные средства вкладываются в четвертый и третий проекты.

u	**Прибыль от внедрения**	
	1 шаг	**f3(u)**
8	**55**	**39**
16	58	53
24	90	80
32	100	120
40	140	145

$$L(8) = \max\underset{8+0}{\{55}; \underset{0+8}{39}\} = 55$$

$$L(16) = \max\left\{\underset{16+0}{58}; \underset{8+8}{55+39}; \underset{0+16}{53}\right\} = \max\{58; \underline{94}; 53\} = \underline{94}$$

$$L(24) = \max\left\{\underset{24+0}{90}; \underset{16+8}{58+39}; \underset{8+16}{55+53}; \underset{0+24}{80}\right\} = \max\{90;97;108;80\} = 108$$

$$L(32) = \max\left\{\underset{32+0}{100}; \underset{24+8}{90+39}; \underset{16+16}{58+53}; \underset{8+24}{55+80}; \underset{0+32}{120}\right\} = \max\{100;129;111;135;120\} = 135$$

$$L(40) = \max\left\{\underset{40+0}{140}; \underset{32+8}{100+39}; \underset{24+16}{90+53}; \underset{16+24}{58+80}; \underset{8+32}{55+120}; \underset{0+40}{145}\right\} = \max\{140;139;143;138;175;145\} = 175$$

3 шаг: Денежные средства вкладываются в четвертый, третий (2 шаг) и второй проекты.

u	**Прибыль от внедрения**	
	2 шаг	**f2(u)**
8	55	35
16	**94**	76
24	108	**120**
32	135	135
40	175	158

$$L(8) = \max\{\underset{8+0}{55};\underset{0+8}{35}\} = 55$$

$$L(16) = \max\{\underset{16+0}{94};\underset{8+8}{55+35};\underset{0+16}{76}\} = \max\{94;90;76\} = 94$$

$$L(24) = \max\{\underset{24+0}{108};\underset{16+8}{94+35};\underset{8+16}{55+76};\underset{0+24}{120}\} = \max\{108;129;131;120\} = 131$$

$$L(32) = \max\{\underset{32+0}{135};\underset{24+8}{108+35};\underset{16+16}{94+76};\underset{8+24}{55+120};\underset{0+32}{135}\} = \max\{135;143;170;175;135\} = 175$$

$$L(40) = \max\left\{\underset{40+0}{175};\underset{32+8}{135+35};\underset{24+16}{108+76};\underset{16+24}{\underline{94+120}};\underset{8+32}{55+135};\underset{0+40}{158}\right\} = \max\{175;170;184;\underline{214};190;158\} = \underline{2}$$

4 шаг: Денежные средства вкладываются в четвертый, третий, второй (3 шаг) и первый проекты.

u	Прибыль от внедрения	
	3 шаг	**f1(u)**
8	55	32
16	94	68
24	131	115
32	175	134
40	**214**	147

$$L(8) = \max\{\underset{8+0}{55};\underset{0+8}{32}\} = 55$$

$$L(16) = \max\{\underset{16+0}{94};\underset{8+8}{55+32};\underset{0+16}{68}\} = \max\{94;87;68\} = 94$$

$$L(24) = \max\{\underset{24+0}{131};\underset{16+8}{94+32};\underset{8+16}{55+68};\underset{0+24}{115}\} = \max\{131;126;123;115\} = 131$$

$$L(32) = \max\{\underset{32+0}{175};\underset{24+8}{131+32};\underset{16+16}{94+68};\underset{8+24}{55+115};\underset{0+32}{134}\} = \max\{175;163;162;170;134\} = 175$$

$$L(40) = \max\left\{\underset{40+0}{\underline{214}};\underset{32+8}{175+32};\underset{24+16}{131+68};\underset{16+24}{94+115};\underset{8+32}{55+134};\underset{0+40}{147}\right\} = \max\{\underline{214};207;199;209;189;147\} = \underline{214}$$

2 этап:

На четвертом шаге выбираем максимальное из полученных значений прибыли L(40)=214.

И возвращаясь в обратном порядке от таблицы к таблице (от 4 шага к 1) выбираем такие значения доходов, при которых и получено значение 214.

Максимальный доход 214 млн. руб. от вложенных средств может быть получен при следующем распределении средств:

1 проект – 0 млн. руб.

2 проект – 24 млн. руб.

3 проект – 8 млн. руб.

4 проект – 8 млн. руб.

Задание 2:

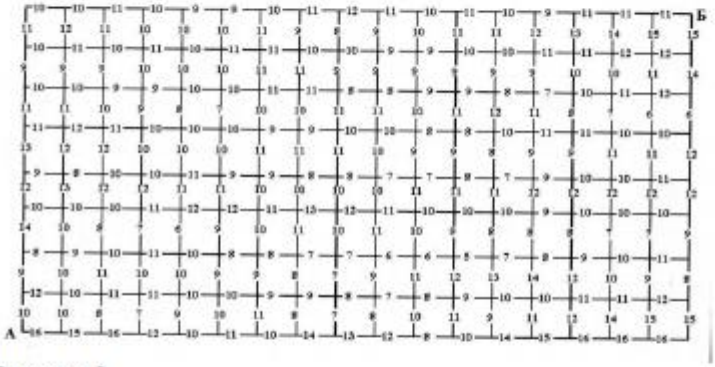
Задание 1. Распределите оптимальным образом денежные средства инвестора величиной *5 у.е.* между четырьмя предприятиями. Доход каждого предприятия от вложения в него *u* у.е.определяется функцией дохода f(u). Эти функции приведены в таблице.

u	Прибыль от внедрения по предприятиям			
	f4(u)	f3(u)	f2(u)	f1(u)
1	f4(1)	6	3	4
2	10	f3(2)	4	6
3	11	11	f2(3)	8
4	12	13	11	f1(4)
5	18	15	18	16

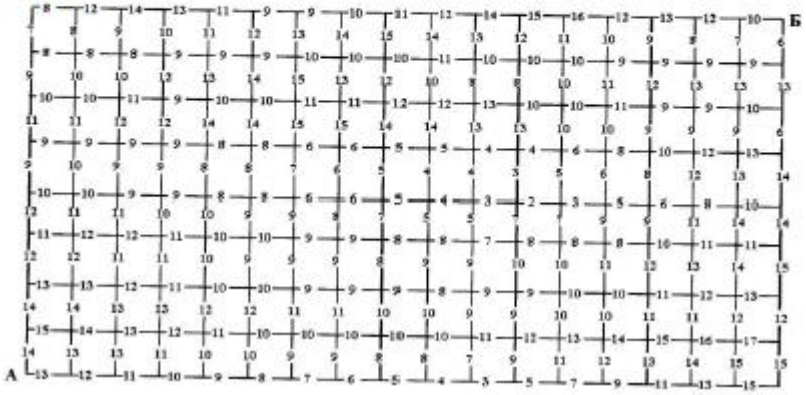
Вариант	1	2	3	4
f4(1)	9	5	6	8
f3(2)	10	10	7	7
f2(3)	7	5	8	9
f1(4)	10	9	13	15

Задание 2. Из пункта А в пункт В необходимо проложить автомобильную трассу по самому экономичному пути.

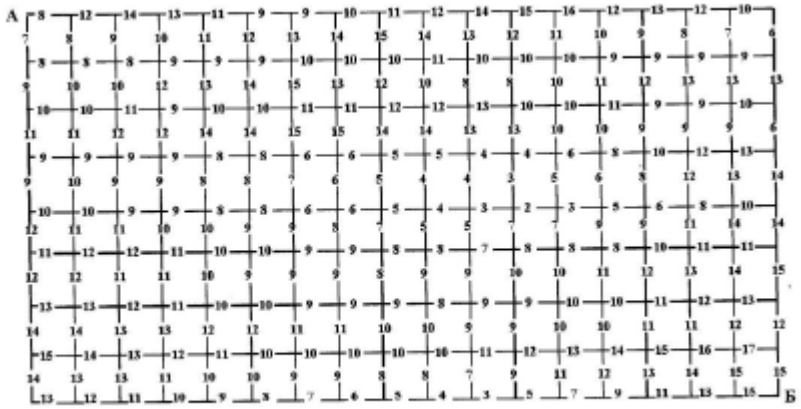
Вариант 1



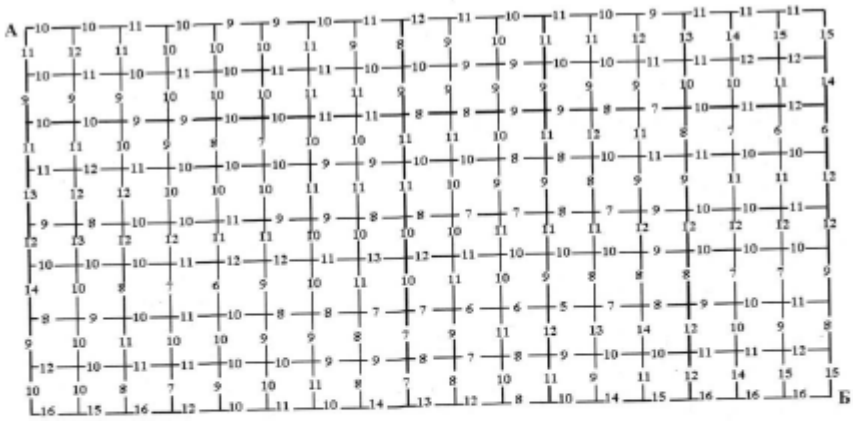
Вариант 2



Вариант 3



Вариант 4



Контрольные вопросы:

1. Какие задачи можно решать методами динамического программирования?
2. В чем заключаются достоинства и недостатки динамического программирования?
3. Объясните алгоритм решения задач динамического программирования.
4. Укажите принцип выбора направления движения.
5. В чем заключается принцип оптимальности?
6. Каков алгоритм распределения ресурсов?

Итог работы: отчет, защита работы.

Практическая работа №15

Цель: отработать навыки решения задачи о нахождении кратчайшего пути в графе

Задание 1: ознакомиться с материалом

Граф это множество точек или вершин и множество линий или ребер, соединяющих между собой все или часть этих точек. *Вершины*, прилегающие к одному и тому же ребру, называются *смежными*. Если *ребра* ориентированы, что обычно показывают *стрелками*, то они называются *дугами*, и граф с такими ребрами называется ***ориентированным графом***. Если *ребра не имеют ориентации*, граф называется ***неориентированным***.

Графы обычно изображаются в виде геометрических фигур, так что вершины графа изображаются точками, а ребра - линиями, соединяющими точки (рис. 1).

Петля это дуга, начальная и конечная вершина которой совпадают.

Простой граф - граф без кратных ребер и петель.

Степень вершины это удвоенное количество петель, находящихся у этой вершины плюс количество остальных прилегающих к ней ребер.

Пустым называется граф без ребер. *Полным* называется граф, в котором каждые две вершины смежные.

Рис. 1

Путь в ориентированном графе — это последовательность дуг, в которой конечная вершина всякой дуги, отличной от последней, является начальной вершиной следующей.

Маршрут в графе путь, ориентацией дут которого можно пренебречь.

Цепь маршрут, в котором все ребра попарно различны.

Цикл замкнутый маршрут, являющийся цепью.

Маршрут, в котором *все вершины попарно различны*, называют ***простой цепью***. Цикл, в котором *все вершины, кроме первой* и *последней*, *попарно различны*, называются ***простым циклом***.

Подграф графа это граф, являющийся *подмоделью* исходного графа, т.е. подграф содержит некоторые вершины исходного графа и некоторые ребра (только те, оба конца которых входят в подграф).

Подграф называется ***остовным*** подграфом, если множество его вершин совпадает с множеством вершин самого графа.

Граф называется ***связным***, если любая пара его вершин связана. *Связными компонентами* графа называются подграфы данного графа, вершины которых связаны.

Дерево — это связный граф без циклов. Деревья особенно часто возникают на практике при изображении различных иерархий. Например, популярны генеалогические деревья.

Граф без цикла называется ***лесом***. Вершины *степени 1* в дереве называются ***листьями***. ***Деревья*** - очень удобный инструмент представления информации самого разного вида. Деревья ***отличаются*** от простых графов тем, что ***при обходе дерева невозможны циклы***. Это делает графы очень удобной формой организации данных для различных алгоритмов.

Очевидно, что графический способ представления графов непригоден для ПК. Поэтому существуют другие способы представления графов.

В теории графов применяются

1. **Матрица инцидентности.** Это матрица ***A*** с ***n*** строками, соответствующими вершинам, и ***m*** столбцами, соответствующего рёбрам. Для ориентированного графа столбец,

соответствующий дуге (x,y) содержит - **1** в строке, соответствующей вершине x и **1**, в строке, соответствующей вершине y. Во всех остальных **0**. Петлю, т.е. дугу (x,x) можно представлять иным значением в строке x, например, **2**. Если граф неориентированный, то столбец, соответствующий ребру (x,y) содержит **1**, соответствующие x и y и нули во всех остальных строках.

2. **Матрица смежности**. Это матрица $n \times n$ где n - число вершин, где $b_{ij} = 1$, если существует ребро, идущее из вершины x в вершину y и $b_{ij} = 0$ в противном случае.

Нахождение минимального остова в графе

Алгоритм решения

1. Упорядочить ребра графа по возрастанию весов;
2. Выбрать ребро с минимальным весом, не образующее цикл с ранее выбранными ребрами. Занести выбранное ребро в список ребер строящегося остова;
3. Проверить, все ли вершины графа вошли в построенный остов. Если нет, то выполнить пункт **2**.

Нахождение кратчайшего пути в графе

Пусть дан граф, дугам которого приписаны веса. Задача о нахождении кратчайшего пути состоит в нахождении кратчайшего пути от заданной начальной вершины до заданной конечной вершины, при условии, что такой путь существует.

Данная задача может быть разбита на две:

1. для начальной заданной вершины найти все кратчайшие пути от этой вершины к другим;
2. найти кратчайшие пути между всеми парами вершин.

Рассмотрим алгоритм решения для задачи первого типа:

Необходимо найти путь от s - начальной вершины до t - конечной вершины. Каждой вершине присваиваем пометки $I(Xi)$.

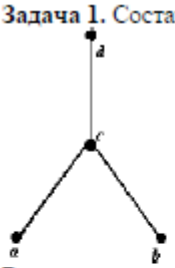
1. $I(s) = 0$, $I(Xi)$ равно бесконечности для всех Xi не равных s и считать эти пометки временными. Положить $p = s$.
2. Для всех Xi, принадлежащих $\Gamma(p)$ и пометки которых временны, изменить пометки по следующему правилу: $I(Xi) = min[I(Xi), I(p) + c(p, Xi)]$
3. среди всех вершин с временными пометками найти такую, для которой $I(Xi^*) = min[I(Xi)]$
4. считать пометку вершины Xi^* постоянной и положить $p = Xi^*$.
5. если $p = t$, то $I(p)$ является длинной кратчайшего пути, если нет, перейти к шагу **2**.

Как только все пометки расставлены, кратчайшие пути получают, используя соотношение $I(Xi') + c(Xi',Xi) = I(Xi)$ **(1)**.

Для решения задачи второго типа можно применять данный алгоритм для каждой вершины.

Задание 2:

Задача 1. Составить матрицы инцидентности и смежности для графа:



Решение.

Матрица инцидентности

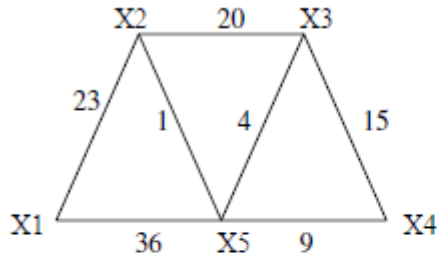
	u	*v*	*w*
a	1	0	0
b	0	0	1
c	1	1	1
d	0	1	0

Где ***u, v, w*** – ребра данного графика

Матрица смежности

	a	*b*	*c*	*d*
a	0	0	1	0
b	0	0	1	0
c	1	1	0	1
d	0	0	1	0

Задача 2. На представленном графе найдите: а) минимальный остов дерева, б) найдите кратчайший путь от начальной точки X1 до всех остальных точек.

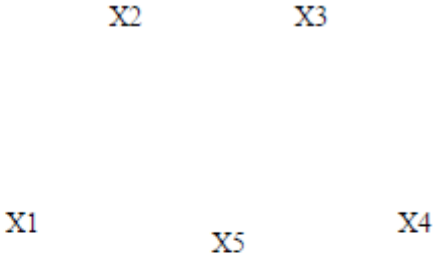


Решение. а) Найдем минимальный остов дерева представленного на рисунке. Составим таблицу значений расстояний между точками.

	X1	X2	X3	X4	X5
X1		23			36
X2	23		20		1
X3		20		15	4
X4			15		9
X5	36	1	4	9	

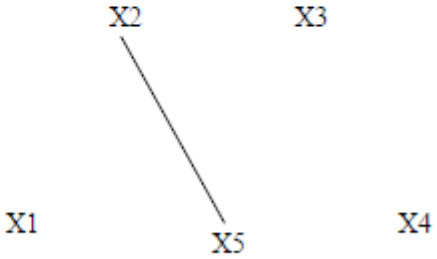
Для решения данной задачи достаточно рассмотреть или только левую или только правую часть от главной диагонали матрицы. Воспользуемся левой частью таблицы. А также изобразим исходный график без ребер, только с помощью одних вершин.

	X1	X2	X3	X4	X5
X1					
X2	23				
X3		20			
X4			15		
X5	36	1	4	9	



Из элементов матрицы выбираем минимальный - ***(X2,X5) = 1***. Обводим выбранный элемент кружком и указываем на рисунке соответствующее ребро.

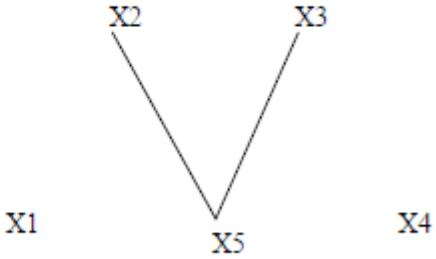
	X1	X2	X3	X4	X5
X1					
X2	23				
X3		20			
X4			15		
X5	36	1	4	9	



Из оставшихся элементов выбираем минимальный - ***(X3,X5) = 4***. Элемент обводим кружком. Чтобы выполнялось условие **2** пункты **X2** и **X3** не должны соединяться, поэтому элемент ***(X2,X3)*** зачёркивается. И т.д.

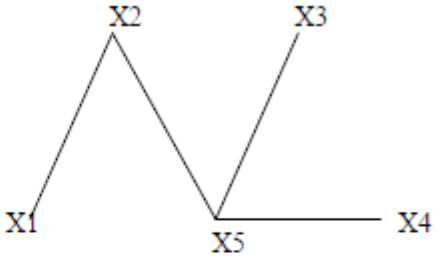
Из оставшихся элементов выбираем минимальный - ***(X3,X5) = 4***. Элемент обводим кружком. Чтобы выполнялось условие **2** пункты **X2** и **X3** не должны соединяться, поэтому элемент ***(X2,X3)*** зачёркивается. И т.д.

	X1	X2	X3	X4	X5
X1					
X2	23				
X3		~~20~~			
X4			15		
X5	36	1	4	9	



В итоге получаем:

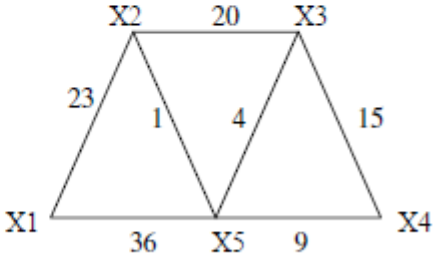
	X1	X2	X3	X4	X5
X1					
X2	23				
X3		20			
X4			15		
X5	36	1	4	9	



Длина минимального остова равна **(X1,X2)+(X2,X5)+(X3,X5)+(X4,X5)=23+1+4+9=37**

Б) Найдем кратчайший путь представленного графа от начальной точки X1 до всех остальных точек.

	X1	X2	X3	X4	X5
X1		23			36
X2	23		20		1
X3		20		15	4
X4			15		9
X5	36	1	4	9	



Начальное расстояние I(X1)=0*, I(Xi)=∞, Xi≠X1, p=X1.

Находим множество точек, соединяющиеся с точкой X1:

Г{X1}={X2,X5}

Находим минимальное расстояние каждой из этих точек:

I(X2)=min[∞,0*+23]=23,

I(X5)=min[∞,0*+36]=36,

min[I(X2), I(X3), I(X4), I(X5)]=min[23, 36, ∞, ∞]=23,

X2: I(X2)=23*, p=23, рядом с точкой X2 поставим расстояние 23.

Находим множество точек, соединяющиеся с точкой X2, точку X1 не трогаем, так как мы ее уже рассмотрели.

Г{X2}={X3,X5}

Находим минимальное расстояние каждой из этих точек:

I(X3)=min[∞,23*+20]=43,

I(X5)=min[36,23*+1]=24,

min[I(X3), I(X4), I(X5)]=min[43,∞, 24]=24,

X5: I(X5)=24*, p=24, рядом с точкой X5 поставим расстояние 24.

Аналогично находим все остальные расстояния до остальных точек:

Г{X5}={X3,X4}

Находим минимальное расстояние каждой из этих точек:

I(X3)=min[43,24*+4]=28,

I(X4)=min[∞,24*+9]=33,

min[I(X3), I(X4)]=min[28, 33]=28,

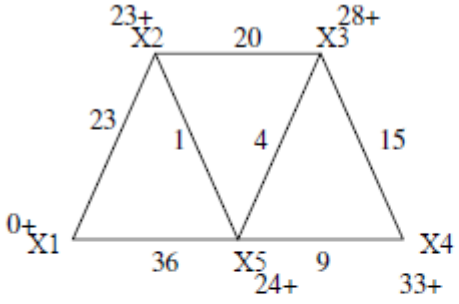
X3: I(X3)=28*, p=28, рядом с точкой X3 поставим расстояние 28.

Г{X3}={X4}

Находим минимальное расстояние до этой точки:

I(X4)=min[33,28*+15]=33,

X4: I(X4)=33*, p=33, рядом с точкой X4 поставим расстояние 33.



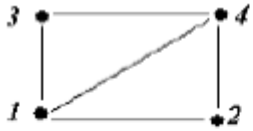
Запишем ответ в виде таблицы кратчайших расстояний от точки X1 до всех остальных точек графа.

Кратчайший путь	значение
X1-X2	23
X1-X2-X5-X3	28
X1-X2-X5-X4	33
X1-X2-X5	24

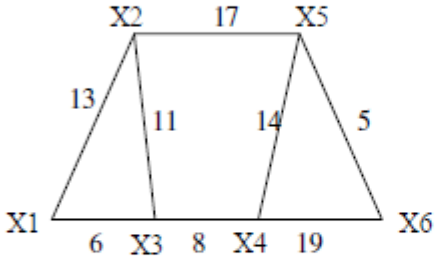
Задание 2:

1 вариант

Задача 1. Составить матрицы инцидентности и смежности для графа:



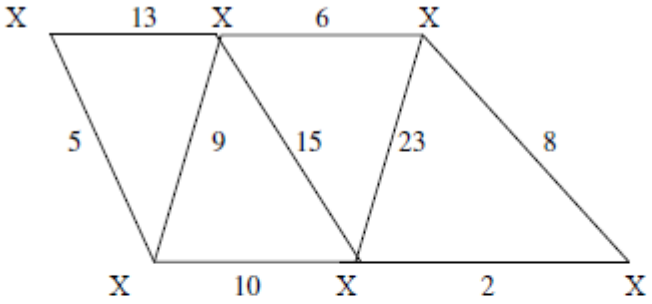
Задача 2. На представленном графе найдите: а) минимальный остов дерева, б) найдите кратчайший путь от начальной точки X1 до всех остальных точек.



2 вариант

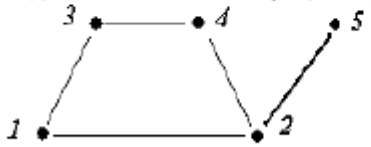
Задача 1. Составить матрицы инцидентности и смежности для графа:

Задача 2. На представленном графе найдите: а) минимальный остов дерева, б) найдите кратчайший путь от начальной точки X1 до всех остальных точек.

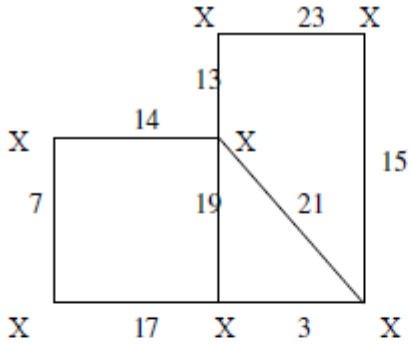


3 вариант

Задача 1. Составить матрицы инцидентности и смежности для графа:

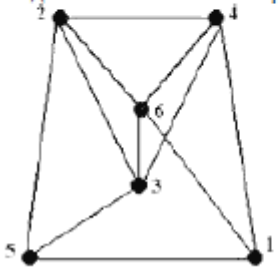


Задача 2. На представленном графе найдите: а) минимальный остов дерева, б) найдите кратчайший путь от начальной точки X1 до всех остальных точек.

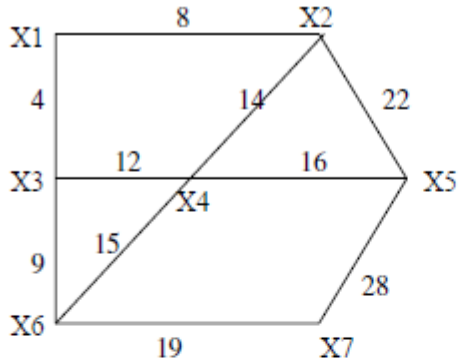


4 вариант

Задача 1. Составить матрицы инцидентности и смежности для графа:

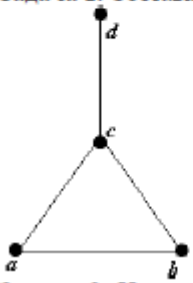


Задача 2. На представленном графе найдите: а) минимальный остов дерева, б) найдите кратчайший путь от начальной точки X1 до всех остальных точек.

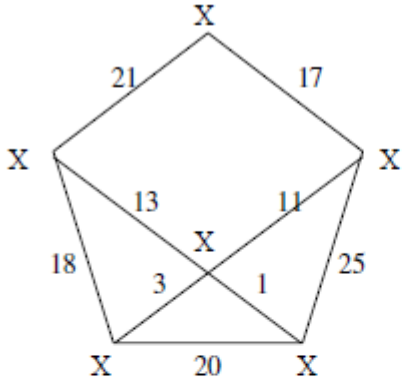


5 вариант

Задача 1. Составить матрицы инцидентности и смежности для графа:



Задача 2. На представленном графе найдите: а) минимальный остов дерева, б) найдите кратчайший путь от начальной точки X1 до всех остальных точек.



Контрольные вопросы

1. Дайте определение граф.
2. В чем состоит отличие ориентированного графа от неориентированного графа?
3. В чем отличие пустого графа от простого графа?
4. Как определить степень вершины?
5. Чем отличается цепь в графе от цикла?
6. Дайте понятие подграф графа.
7. В чем суть связанного графа?
8. Как находятся матрицы инцидентности и матрицы смежности?
9. Как найти минимальный остов дерева?
10. Как найти кратчайшее расстояние в графе?

Итог работы: отчет, защита работы.

Практическая работа №16

Цель: Решить задачи определения максимальной пропускной способности сети , с использованием MS Excel.

Задание 1:

Варианты заданий

Сеть содержит 10 вершин Рисунок 4.5. Пропускная способность дуг заданы таблицей.

Найти максимальный поток между вершинами 1 и 10.

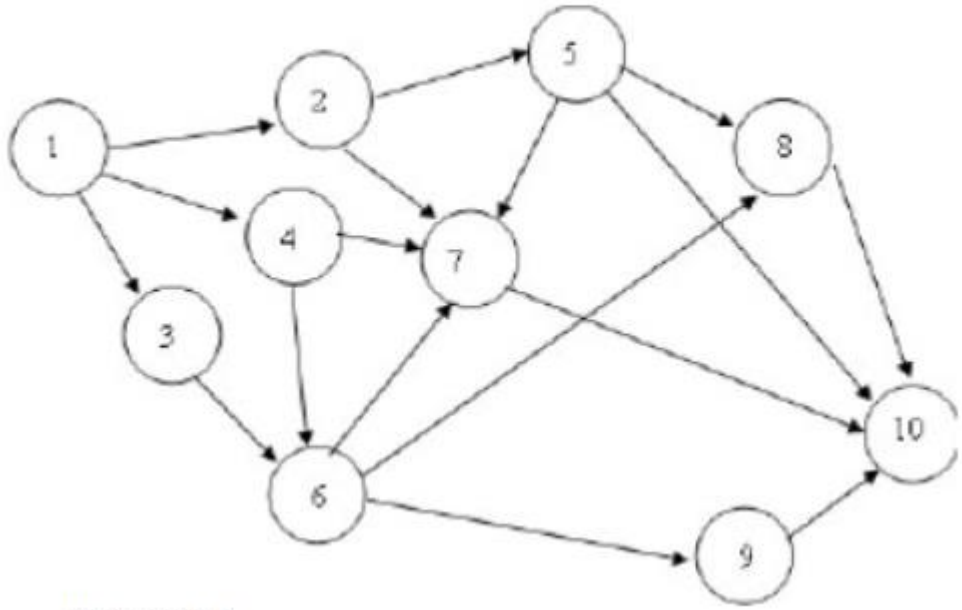


Рисунок 4.5

Таблица 4.1

начало	конец	Варианты						
		1	2	3	4	5	6	7
		длина	длина	длина	длина	длина	длина	длина
1	2	4	9	12	10	9	12	9
1	3	13	12	9	13	11	12	8
1	4	14	4	9	11	3	12	7
2	5	7	10	7	11	2	3	7
2	7	7	3	3	4	1	4	14
3	6	13	4	14	9	13	5	1
4	6	6	15	5	7	15	5	13
4	7	9	13	12	9	3	2	12
5	7	6	8	11	5	14	15	5
5	8	2	4	9	13	1	11	4
5	10	2	10	2	2	10	8	8
6	7	4	15	11	3	6	4	9
6	8	9	14	11	15	3	3	3
6	9	14	7	5	7	7	5	11
7	10	10	9	15	10	6	11	2
8	10	15	10	9	8	15	8	15
9	10	10	15	7	1	12	11	8

Итог работы: отчет, защита работы.

Практическая работа №17

Цель: отработать навыки составлять системы уравнений Колмогорова

Задание 1:

Вариант 1

1. Техническое устройство состоит из трёх узлов и в любой момент времени может находиться в одном из восьми состояний (рис. 1).Численные значения интенсивности потоков событий: $\lambda_1=2$; $\lambda_2=2$; $\lambda_3=1$; $\mu_1=4$; $\mu_2=4$; $\mu_3=2$. Найдите финальные вероятности сосотояний устройства.

2. Интенсивность потока автомобилей, поступающих на моечную станцию (одноканальная СМО) – 5 автомобиля в час, а интенсивность обслуживания – 6 автомобилей в час. Предполагая, что станция работает в стационарном режиме, найти среднее число автомобилей, находящихся на станции, среднюю длину очереди и среднее время ожидания обслуживания.
3. Какое оптимальное число линий обслуживания должна иметь СМО, если $\lambda = 3$, $\mu = 2$, $c_1 = 4$, $c_2 = 2$.
4. Определить число взлетно-посадочных полос для самолётов с учетом требования, что вероятность ожидания $P(w > 0)$ должна быть меньше, чем 0,06. Интенсивность потока равна 28 требований в сутки и интенсивность линий обслуживания – 32 самолётов в сутки.

Вариант 2

1. Техническое устройство состоит из трёх узлов и в любой момент времени может находиться в одном из восьми состояний (рис. 1).Численные значения интенсивности потоков событий: $\lambda_1=2$; $\lambda_2=1$; $\lambda_3=1$; $\mu_1=4$; $\mu_2=2$; $\mu_3=2$. Найдите финальные вероятности сосотояний устройства.
2. Интенсивность потока автомобилей, поступающих на моечную станцию (одноканальная СМО) – 6 автомобиля в час, а интенсивность обслуживания – 7 автомобилей в час. Предполагая, что станция работает в стационарном режиме, найти среднее число автомобилей, находящихся на станции, среднюю длину очереди и среднее время ожидания обслуживания.
3. Какое оптимальное число линий обслуживания должна иметь СМО, если $\lambda = 4$, $\mu = 2$, $c_1 = 5$, $c_2 = 2$.
4. Определить число взлетно-посадочных полос для самолётов с учетом требования, что вероятность ожидания $P(w > 0)$ должна быть меньше, чем 0,06. Интенсивность потока равна 30 требований в сутки и интенсивность линий обслуживания – 34 самолётов в сутки.

Вариант 3

1. Техническое устройство состоит из трёх узлов и в любой момент времени может находиться в одном из восьми состояний (рис. 1).Численные значения интенсивности потоков событий: $\lambda_1=1$; $\lambda_2=2$; $\lambda_3=2$; $\mu_1=4$; $\mu_2=4$; $\mu_3=4$. Найдите финальные вероятности сосотояний устройства.
2. Интенсивность потока автомобилей, поступающих на моечную станцию (одноканальная СМО) – 4 автомобиля в час, а интенсивность обслуживания – 5 автомобилей в час. Предполагая, что станция работает в стационарном режиме, найти среднее число автомобилей, находящихся на станции, среднюю длину очереди и среднее время ожидания обслуживания.
3. Какое оптимальное число линий обслуживания должна иметь СМО, если $\lambda = 2$, $\mu = 1$, $c_1 = 3$, $c_2 = 2$.
4. Определить число взлетно-посадочных полос для самолётов с учетом требования, что вероятность ожидания $P(w > 0)$ должна быть меньше, чем 0,08. Интенсивность потока равна 28 требований в сутки и интенсивность линий обслуживания – 32 самолётов в сутки.

Вариант 4

1. Техническое устройство состоит из трёх узлов и в любой момент времени может находиться в одном из восьми состояний (рис. 1).Численные значения интенсивности потоков событий: $\lambda_1=2$; $\lambda_2=2$; $\lambda_3=2$; $\mu_1=2$; $\mu_2=2$; $\mu_3=4$. Найдите финальные вероятности сосотояний устройства.
2. Интенсивность потока автомобилей, поступающих на моечную станцию (одноканальная СМО) – 8 автомобиля в час, а интенсивность обслуживания – 9 автомобилей в час. Предполагая, что станция работает в стационарном режиме, найти среднее число автомобилей, находящихся на станции, среднюю длину очереди и среднее время ожидания обслуживания.
3. Какое оптимальное число линий обслуживания должна иметь СМО, если $\lambda = 7$, $\mu = 8$, $c_1 = 4$, $c_2 = 2$.
4. Определить число взлетно-посадочных полос для самолётов с учетом требования, что вероятность ожидания $P(w > 0)$ должна быть меньше, чем 0,06. Интенсивность потока равна 18 требований в сутки и интенсивность линий обслуживания – 22 самолётов в сутки.

Контрольные вопросы:

1. Дайте определение марковскому процессу.
2. Какие типы неопределенностей встречаются.
3. Дайте определение потоку событий.
4. Как составить уравнения Колмогорова.
5. Какие виды СМО Вы знаете?
6. При каких предположениях изучаются одноканальные СМО с отказами?
7. Почему стационарный режим в одноканальных СМО с ожиданием существует только при условии $\varphi > 0$?
8. Какие средние характеристики можно рассчитать в одноканальных СМО с ожиданием?

Итог работы: отчет, защита работы.

Практическая работа №18

Цель: Сформировать систему знаний о результатах хозяйственной деятельности предприятия. Овладение методикой проведения анализа результатов деятельности предприятия, методикой расчета выручки, прибыли и рентабельности.

Задание 1:

Какую прибыль может получить предприятие при выпуске 14000 штук изделий, если постоянные затраты предприятия - 960,9 тыс. руб. переменные затраты на единицу продукции -185 руб. Рыночная цена единицы продукции - 280 рублей.

Задание 2. Какую прибыль может получить предприятие при выпуске 15000 штук изделий, если постоянные затраты предприятия 975 тыс. руб. переменные затраты на единицу продукции 200 руб. Рыночная цена единицы продукции 300 руб.

Задание 3. Оптовая отпускная цена изделия - 820 руб. Полная себестоимость - 412 рублей. За год выпускается 5000 штук изделий. Определить годовую прибыль промышленного предприятия.

Задание 4. Годовой выпуск изделия - 5000 шт. Прибыль от реализации годового выпуска продукции - 532 тыс. руб. Оптовая цена изделия (без НДС) - 720 руб. Определить полную себестоимость одного изделия.

Задание 5. Рассчитайте годовую прибыль предприятия, если доход за год составил 2,5 млн рублей, годовые переменные издержки составили 0,5 млн рублей, постоянные издержки составили 1,2 млн рублей. Рассчитайте рентабельность продаж.

Задание 6. В первом квартале предприятие реализовало продукции 5 тысяч штук по цене 80 рублей за единицу. Общие постоянные расходы – 70 тысяч рублей; переменные затраты на единицу - 60 рублей. Во 2 квартале изготовлено на 100 единиц больше, а постоянные расходы удалось сократить на 20%. Определить величину прибыли (убытка) от реализации продукции в 1 и во 2 квартале, а также прирост прибыли (убытка) в абсолютном и относительном выражениях**.**

Итог работы: отчет, защита работы.

Практическая работа №19

Цель: решать задачи в системах массового обслуживания.

Задание 1:

Варианты заданий.

1. Одноканальная СМО с отказами представляет собой одну телефонную линию. Заявка, пришедшая в момент, когда линия занята, получает отказ. Все потоки событий простейшие. Интенсивность потока λ =0,95 вызова в минуту. Средняя продолжительность разговора t=1мин. Определите вероятностные характеристики СМО в установившемся режиме работы.

2. В одноканальную СМО с отказами поступает простейший поток с интенсивностью λ = 0,5 заявки в минуту. Время обслуживания заявки имеет показательное распределение с t = 1,5 мин. Определите вероятностные характеристики СМО в установившемся режиме работы.

3. В вычислительном центре работают пять персональных компьютеров. Простейший поток задач, поступающих на вычислительный центр, имеет интенсивность λ =10 задач в час. Среднее время решения задачи равно 12 мин. Заявка получает отказ, если все компьютеры заняты. Найдите вероятностные характеристики системы обслуживания.

4.В аудиторскую фирму поступает простейший поток заявок на обслуживания с интенсивностью λ =1.5 заявки в день. Время обслуживания распределено по показательному закону и равно в среднем трем дням. Аудиторская фирма располагает пятью независимыми бухгалтериями, выполняющими аудиторские проверки. Очередь заявок неограниченна. Определите вероятностные характеристики аудиторской фирмы как СМО, работающей в стационарном режиме.

5.На пункт техосмотра поступает простейший поток заявок интенсивностью λ =4 машины в час. Время осмотра распределено по показательному закону и равно в среднем 17 мин, в очереди может находиться не более пяти автомобилей. Определите вероятностные характеристики пункта техосмотра в установившемся время.

6.В бухгалтерии предприятия имеются два кассира, каждый из которых может обслужить в среднем 30 сотрудников в час. Поток сотрудников, получающих заработную плату, - простейший, с интенсивностью ,равной 40 сотрудников в час. Очередь в кассе не ограничена. Время обслуживания подчинено экспоненциальному закону распределения. Вычислите вероятностные характеристики СМО в стационарном режиме и определите целесообразность приема третьего кассира на предприятие, работающего с такой же производительностью, как и первые два.

7.В инструментальном отделении сборочного цеха работают три кладовщика. В среднем за 1 мин за инструментом проходят 0,8 рабочих. Обслуживание одного рабочего занимает у кладовщика t=1,0 мин. Очередь не имеет ограничения. Известно, что поток рабочих за инструментом - пуассоновской, а время обслуживания подчинено экспонциональному закону распределения. Стоимость 1 мин работы рабочего равна 30 д.е а кладовщика – 15д.е. Найдите средние потери цеха при данной организации обслуживания в инструментальном отделении при стационарном режиме работы.

8.Билетная касса работает без перерыва. билеты продает один кассир. Среднее время обслуживания – 2 мин на каждого человека. Среднее число пассажиров, желающих приобрести билеты в кассе в течение одного часа, λ =20 пассажиров в час. Все потоки в системе простейшие. Определите характеристики СМО в условиях стационарного режима работы кассы.

9.Пост диагностики автомобилей представляет собой одноканальную СМО с отказами. Заявка на диагностику, поступающая в момент, когда пост занят, получает отказ. Интенсивность потока заявок на диагностику λ =0,5 автомобиля в час. Средняя продолжительность диагностики t=1,2 ч. Все потоки событий в системе простейшие. Определите в установившемся режиме характеристики системы.

10.Автозаправочная станция представляет собой СМО с одним каналом обслуживания и одной колонкой. Площадка при АЗС допускает пребывание в очереди на заправку не более трех автомобилей одновременно. Если в очереди уже находится три автомобиля, очередной автомобиль, прибывший к станции, в очередь не становится, а проезжает мимо. Поток автомобилей, прибывающих для заправки, имеет интенсивность λ =0,7 автомобиля в минуту. Процесс заправки продолжается в среднем 1,25 мин. Все потоки простейшие. Определите вероятностные характеристики СМО в стационарном режиме.

11.На железнодорожную сортировочную горку прибывают составы с интенсивностью λ=2 состава в час. Среднее время, в течение которого горка обслуживает состав, равно 0,4 ч. Составы, прибывающие в момент, когда занята, становятся в очередь и ожидают в парке прибытия, где имеются три запасных пути,

на каждом из которых может ожидать один состав. Состав, прибывший в момент, когда все три запасных пути в парке прибытия заняты, становятся в очередь на внешний путь. Все потоки событий простейшие. Определите характеристики СМО в условиях стационарного режима работы

12.Рассматривается работа АЗС, на которой имеются три заправочных колонки. Заправка одной машины длится в среднем 3 мин. В среднем на АЗС каждую минуту прибывает машина, нуждающаяся в заправке бензина. Число мест в очереди не ограничено. Все машины, вставшие в очередь на заправку, дожидаются своей очереди. Все потоки в системе простейшие. Определите вероятностные характеристики работы АЗС в стационарном режиме.

13.На станцию технического обслуживания автомобилей каждые два часа подъезжает в среднем одна машина. Станция имеет шесть постов обслуживания. Очередь автомобилей, ожидающих обслуживания, не ограничена. Среднее время обслуживания одной машины – 2 ч. Все потоки в системе простейшие. Определите характеристики станции технического обслуживания автомобилей.

14.Имеется двухканальная простейшая СМО с отказами. На ее вход поступает поток заявок с интенсивностью $\lambda=3$ заявки в час. Среднее время обслуживания одной заявки t= 0,5 ч. Каждая обслуженная заявка приносит доход 5 д.е./. Содержание канала обходится в 3 д.е. /ч. Решите, выгодно ли в экономическом отношении увеличить число каналов СМО до трех.

15. В магазине работает один продавец, который может обслужить в среднем 30 покупателей в час. Поток покупателей простейший с интенсивностью, равной 60 покупателей в час. Все покупатели «нетерпеливые» и уходят, если в очереди стоит пять человек (помимо обслуживаемых). Все потоки событий простейшие. Определите характеристики магазина для стационарного режима работы.

16.На вход телефонный станции, имеющей девять каналов обслуживания, поступает в среднем 120 заявок в час. Заявка получает отказ, если все каналы заняты. Среднее время обслуживания в одном канале равно 4 мин. Все потоки в системе простейшие. Определите характеристики телефонной станции.

17. Рассматривается работа АЗС, на которой имеется пять заправочных колонок. Заправка одной машины длится в среднем 4 мин. В среднем на АЗС каждую минуту прибывает машина, нуждаются в заправке бензином. Число мест в очереди не ограничено. Все машины, вставшие в очередь , дожидаются своей очереди. Все потоки событий простейшие. Определите вероятностные характеристики АЗС для стационарного режима.

18.Подсчитайте вероятностные характеристики для простейшей одноканальной СМО с тремя с тремя местами в очереди при условиях $\lambda=4$ заявки в час, t=0,5 ч. Выясните, как эти характеристики изменятся, если увеличить число мест в очереди до четырех.

19.Одноканальная СМО – ЭВМ, на которую поступают заявки на расчеты. Поток заявок простейший со среднем интервалом времени между заявками t=10 мин. Время обслуживания распределено по экспоненциальному закону с математическим ожиданием t = 8мин. Определите среднее число заявок в СМО, среднее время пребывания заявки в системе и в очереди.

20.Система массового обслуживания – билетная касса с тремя окошками (с тремя кассирами) и неограниченной очередью. Пассажиров, желающих купить билет, приходит в среднем пять человек за 20 мин. Поток пассажиров можно считать простейшим. Кассир в среднем обслуживает трех пассажиров за 10 мин.

Время обслуживания подчинено показательному закону распределения. Определите вероятностные характеристики СМО в стационарном режиме.

Контрольные вопросы.

1. *Примеры СМО.*
2. *Основные понятия СМО.*
3. *Основные характеристики СМО.*

Итог работы: отчет, защита работы.

Практическая работа №20

Цель: отработать навыки решения матричной антогонистической игры

Задание 1:

1. Привести к виду ЗЛП**.**

1. $\begin{pmatrix} 7 & 9 \\ 12 & 4 \end{pmatrix}$ 2. $\begin{pmatrix} -1 & 2 \\ 2 & -2 \end{pmatrix}$ 3. $\begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix}$

4. $\begin{pmatrix} 3 & -2 \\ 0 & 5 \end{pmatrix}$ 5. $\begin{pmatrix} 1 & 4 \\ 3 & -2 \end{pmatrix}$ 6. $\begin{pmatrix} 4 & 2 \\ -4 & 0 \end{pmatrix}$

2. Определить оптимальные чистые стратегии и цену игры.

1. $\begin{pmatrix} 1 & 2 & 1 & 3 & 7 \\ 3 & -5 & -1 & 6 & -2 \\ 3 & 2 & 1 & 4 & 2 \\ -1 & 7 & -2 & -3 & 4 \end{pmatrix}$ 2. $\begin{pmatrix} 1 & 2 & 1 & 3 & 7 \\ 2 & -5 & -1 & 6 & -5 \\ 6 & 2 & 1 & 4 & 2 \\ -1 & 3 & -4 & -1 & 4 \end{pmatrix}$ 3. $\begin{pmatrix} 5 & -3 & 2 & 2 \\ 4 & 7 & 3 & 3 \\ 4 & 0 & 1 & 5 \\ 1 & 2 & 3 & 4 \end{pmatrix}$

4. $\begin{pmatrix} 3 & 4 & 4 & 9 & 6 \\ 5 & 5 & 3 & 7 & 8 \\ 4 & 5 & 1 & 4 & 2 \\ 4 & 2 & 8 & 5 & 3 \end{pmatrix}$ 5. $\begin{pmatrix} 4 & -8 & 6 & 2 \\ 3 & 7 & 3 & 13 \\ 4 & 5 & 1 & 5 \\ 11 & 2 & 3 & 14 \end{pmatrix}$ 6. $\begin{pmatrix} 2 & 3 & 3 & 8 & 5 \\ 4 & 4 & 2 & 6 & 7 \\ 3 & 4 & 0 & 3 & 1 \\ 3 & 1 & 7 & 4 & 2 \end{pmatrix}$

3. Предприниматель собирается вложить сумму в количестве 100 тыс.руб. в совместное предприятие. У него есть четыре альтернативы выбора формы заключения договора с партнером (стратегии А1, А2, А3, А4). С другой стороны прибыль предпринимателя зависит от того , какую стратегию поведения выберет его партнер и совет директоров (у партнера контрольный пакет акций). Имеются оценки выигрышей предпринимателя для каждой пары альтернатив ($А_к$, $В_к$). Прибыль приводится в процентах годовых от вложения, которые приведены в платежной матрице А.

Определить оптимальную стратегию вложения денег. Решить задачу методом теории игр в чистых стратегиях.

1

	B1	B2	B3	B4
A1	70	90	30	70
A2	80	70	40	50
A3	30	40	20	60
A4	70	50	30	50

2

	B1	B2	B3	B4
A1	45	30	50	80
A2	75	70	90	80
A3	60	40	50	70
A4	10	20	10	40

3

	B1	B2	B3	B4
A1	40	50	90	60
A2	20	30	40	40
A3	10	20	60	30
A4	5	15	15	20

4

	B1	B2	B3	B4
A1	20	10	20	50
A2	50	40	50	60
A3	30	20	30	70
A4	40	10	20	60

5

	B1	B2	B3	B4
A1	60	50	40	30
A2	70	60	70	90
A3	60	50	80	80
A4	40	30	60	70

6

	B1	B2	B3	B4
A1	70	20	60	50
A2	90	40	80	50
A3	80	50	70	90
A4	40	10	20	60

4. Некоторая организация А собирается либо выпускать(стратегия А1), либо не выпускать (стратегия А2) новый вид продукции. При этом ей неизвестно, будет ли выпускать тот же вид продукции конкурирующая организация В.Если и А и В будут выпускать одну и ту же продукцию , то это принесет организации А убыток в а млн. руб. Если и А и В не будут выпускать продукцию, то это не принесет А ни прибыли ни убытка. Если А будет выпускать, а В нет, то прибыль А составит b млн. руб. Если В будет выпускать , а А нет, то из за прекращения конкуренции организации В по другим товарам, прибыль А составит с млн.руб.

Методами теории игр определить, с какой вероятностью следует решиться на выпуск нового товара, чтобы получить максимальную прибыль.

1. a=8 b=17 c=2.
2. a=9 b=11 c=7.
3. a=1 b=13 c=8.
4. a=5 b=18 c=7.
5. a=7 b=14 c=7.
6. a=8 b=17 c=5.

Итог работы: отчет, защита работы.

Практическая работа №21

Цель: приобретение умений представлять матричные игры в виде задач линейного программирования с последующим решением

Задание 1:

1. Привести к виду ЗЛП**.**

1. $\begin{pmatrix} 7 & 9 \\ 12 & 4 \end{pmatrix}$
2. $\begin{pmatrix} -1 & 2 \\ 2 & -2 \end{pmatrix}$
3. $\begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix}$
4. $\begin{pmatrix} 3 & -2 \\ 0 & 5 \end{pmatrix}$
5. $\begin{pmatrix} 1 & 4 \\ 3 & -2 \end{pmatrix}$
6. $\begin{pmatrix} 4 & 2 \\ -4 & 0 \end{pmatrix}$

2. Определить оптимальные чистые стратегии и цену игры.

1. $\begin{pmatrix} 1 & 2 & 1 & 3 & 7 \\ 3 & -5 & -1 & 6 & -2 \\ 3 & 2 & 1 & 4 & 2 \\ -1 & 7 & -2 & -3 & 4 \end{pmatrix}$ 2. $\begin{pmatrix} 1 & 2 & 1 & 3 & 7 \\ 2 & -5 & -1 & 6 & -5 \\ 6 & 2 & 1 & 4 & 2 \\ -1 & 3 & -4 & -1 & 4 \end{pmatrix}$ 3. $\begin{pmatrix} 5 & -3 & 2 & 2 \\ 4 & 7 & 3 & 3 \\ 4 & 0 & 1 & 5 \\ 1 & 2 & 3 & 4 \end{pmatrix}$

4. $\begin{pmatrix} 3 & 4 & 4 & 9 & 6 \\ 5 & 5 & 3 & 7 & 8 \\ 4 & 5 & 1 & 4 & 2 \\ 4 & 2 & 8 & 5 & 3 \end{pmatrix}$ 5. $\begin{pmatrix} 4 & -8 & 6 & 2 \\ 3 & 7 & 3 & 13 \\ 4 & 5 & 1 & 5 \\ 11 & 2 & 3 & 14 \end{pmatrix}$ 6. $\begin{pmatrix} 2 & 3 & 3 & 8 & 5 \\ 4 & 4 & 2 & 6 & 7 \\ 3 & 4 & 0 & 3 & 1 \\ 3 & 1 & 7 & 4 & 2 \end{pmatrix}$

3. Предприниматель собирается вложить сумму в количестве 100 тыс.руб. в совместное предприятие. У него есть четыре альтернативы выбора формы заключения договора с партнером (стратегии А1, А2, А3, А4). С другой стороны прибыль предпринимателя зависит от того , какую стратегию поведения выберет его партнер и совет директоров (у партнера контрольный пакет акций). Имеются оценки выигрышей предпринимателя для каждой пары альтернатив ($А_к$, $В_к$). Прибыль приводится в процентах годовых от вложения, которые приведены в платежной матрице А.

Определить оптимальную стратегию вложения денег. Решить задачу методом теории игр в чистых стратегиях.

1

	B1	B2	B3	B4
A1	70	90	30	70
A2	80	70	40	50
A3	30	40	20	60
A4	70	50	30	50

2

	B1	B2	B3	B4
A1	45	30	50	80
A2	75	70	90	80
A3	60	40	50	70
A4	10	20	10	40

3

	B1	B2	B3	B4
A1	40	50	90	60
A2	20	30	40	40
A3	10	20	60	30
A4	5	15	15	20

4

	B1	B2	B3	B4
A1	20	10	20	50
A2	50	40	50	60
A3	30	20	30	70
A4	40	10	20	60

5

	B1	B2	B3	B4
A1	60	50	40	30
A2	70	60	70	90
A3	60	50	80	80
A4	40	30	60	70

6

	B1	B2	B3	B4
A1	70	20	60	50
A2	90	40	80	50
A3	80	50	70	90
A4	40	10	20	60

4. Некоторая организация А собирается либо выпускать(стратегия А1), либо не выпускать (стратегия А2) новый вид продукции. При этом ей неизвестно, будет ли выпускать тот же вид продукции конкурирующая организация В.Если и А и В будут выпускать одну и ту же продукцию , то это принесет организации А убыток в а млн. руб. Если и А и В не будут выпускать продукцию, то это не принесет А ни прибыли ни убытка. Если А будет выпускать, а В нет, то прибыль А составит b

млн. руб. Если В будет выпускать , а А нет, то из за прекращения конкуренции организации В по другим товарам, прибыль А составит с млн.руб.

Методами теории игр определить, с какой вероятностью следует решиться на выпуск нового товара, чтобы получить максимальную прибыль.

1.	a=8	b=17 c=2.	2.	a=9	b=11 c=7.
3.	a=1	b=13 c=8.	4.	a=5	b=18 c=7.
5.	a=7	b=14 c=7.	6.	a=8	b=17 c=5.

Итог работы: отчет, защита работы.

Практическая работа №22

Цель: научиться оценивать надежность простейших систем

Задание 1:

Вариант 1

1. Система состоит из двух блоков, соединенных последовательно. Первый блок содержит три элемента: A, B, C, а второй- два элемента: D, E. Элементы каждого блока соединены параллельно.

а) Найти методом Монте-Карло оценку P^* надежности системы, зная вероятности безотказной работы элементов: P(A)=0,8; P(B)=0,9; P(C)=0,85; P(D)=0,7; P(E)=0,6;

б) найти абсолютную погрешность $|P-P^*|$, где P- надежность системы, вычисленная аналитически. Произвести 15 испытаний.

2. В двухканальную систему массового обслуживания с отказом поступает пуассоновский поток заявок. Время между поступлениями двух последовательных заявок распределено по показательному закону $f(\tau)=4e^{-4\tau}$. Длительность обслуживания каждой заявки равна 1 мин. Найти методом Монте-Карло математическое ожидание *a* числа обслуженных заявок за время T=8 мин.

Вариант 2

1. Система состоит из двух блоков, соединенных последовательно. Первый блок содержит два элемента: A, B, второй- три элемента: C, D, E. Элементы первого и второго блоков соединены параллельно.

а) Найти методом Монте-Карло оценку P^* надежности системы, зная вероятности безотказной работы элементов: P(A)=0,8; P(B)=0,9; P(C)=0,7; P(D)=0,75; P(E)=0,8;

б) найти абсолютную погрешность $|P-P^*|$, где P - надежность системы, вычисленная аналитически. Произвести 15 испытаний.

2. В трехканальную СМО с отказами поступает пуассоновский поток заявок. Время между моментами поступления двух последовательных заявок распределено по закону $f(\tau)=0,8e^{-0,8\tau}$; время обслуживания заявок 1,5 мин. Найти методом Монте-Карло математическое ожидание *a* числа обслуженных заявок за время T=10мин.

Вариант 3

1. Система состоит из двух блоков, соединенных последовательно. Первый блок содержит три элемента: A, B, C, а второй- два элемента: D, E. Элементы каждого блока соединены параллельно.

а) Найти методом Монте-Карло оценку P^* надежности системы, зная вероятности безотказной работы элементов: P(A)=0,9; P(B)=0,5; P(C)=0,95; P(D)=0,8; P(E)=0,7;

б) найти абсолютную погрешность $|P-P^*|$, где P- надежность системы, вычисленная аналитически. Произвести 15 испытаний.

2. В двухканальную систему массового обслуживания с отказом поступает пуассоновский поток заявок. Время между поступлениями двух последовательных заявок распределено по показательному закону $f(\tau)=5e^{-5\tau}$. Длительность обслуживания каждой заявки равна 1,5 мин. Найти методом Монте-Карло математическое ожидание *a* числа обслуженных заявок за время T=10 мин.

Вариант 4

1. Система состоит из двух блоков, соединенных последовательно. Первый блок содержит два элемента: A, B, второй- три элемента: C, D, E. Элементы первого и второго блоков соединены параллельно.

а) Найти методом Монте-Карло оценку P^* надежности системы, зная вероятности безотказной работы элементов: P(A)=0,8; P(B)=0,7; P(C)=0,8; P(D)=0,85; P(E)=0,6;

б) найти абсолютную погрешность $|P-P^*|$, где P - надежность системы, вычисленная аналитически. Произвести 15 испытаний.

2. В трехканальную СМО с отказами поступает пуассоновский поток заявок. Время между моментами поступления двух последовательных заявок распределено по закону $f(\tau)=8\ e^{-8\tau}$; время обслуживания заявок 1мин. Найти методом Монте-Карло математическое ожидание *a* числа обслуженных заявок за время T=8 мин.

Контрольные вопросы:

1. В чем заключается суть имитационного моделирования?
2. В чем заключаются достоинства и недостатки такого типа моделирования?
3. Как применяется метод Монте-Карло?
4. Какие способы получения случайных величин Вы знаете?

Итог работы: отчет, защита работы.

Практическая работа №23

Цель: отработать навыки решения задач

Задание 1:

АТС имеет k линий связи. Поток вызовов - простейший с интенсивностью λ в минуту. Среднее время переговоров составляет t минут. Время переговоров имеет показательное распределение. Найти: а) вероятность того, что все линии связи заняты; б) относительную и абсолютную пропускные способности АТС; в) среднее число занятых линий связи. Определить оптимальное число линий связи, достаточное для того, чтобы вероятность отказа не превышала α. k = 5; λ = 0.6; t = 3.5, α = 0.04.

Задание 2:

Пункт по ремонту квартир работает в режиме отказа и состоит из двух бригад. Интенсивность потока заявок λ, производительность пункта μ. Определить вероятность того, что оба каналы свободны, один канал занят, оба канала заняты, вероятность отказа, относительную и абсолютную пропускные способности, средне число занятых бригад.

Рекомендации к решению задачи: здесь n = 2; λ = 1.5 ед. в час.; μ = 1.8 ед. в час.

Задание 3:

В вычислительный центр коллективного пользования с тремя ЭВМ поступают заказы от предприятий на вычислительные работы. Если работают все три ЭВМ, то вновь поступающий заказ не принимается, и предприятие вынуждено обратиться в другой вычислительный центр. Среднее время работы с одним заказом составляет 3 ч. Интенсивность потока заявок 0,25 (1/ч). Найти предельные вероятности состояний и показатели эффективности работы вычислительного центра.

Рекомендации к решению задачи: здесь n = 3; λ = 0.25 ед. в час.; $t_{обс}$ = 3 час.

Итог работы: отчет, защита работы.

Практическая работа №24

Цель: отработать навыки решения задач

Задание 1:

Построение математической модели систем массового обслуживания с неограниченной очередью

Магазин с одним продавцом. Предполагается, что простейший поток покупателей поступает с интенсивностью λ=20 человек/ч. Время обслуживания заявки – случайная величина, которая подчиняется экспоненциальному закону распределения с параметром μ = 25 человек/ч. Определим:

1) среднее время пребывания покупателя в очереди;
2) среднюю длину очереди;
3) среднее число покупателей в магазине;
4) среднее время пребывания покупателя в магазине;
5) вероятность того, что в магазине не окажется покупателей;
6) вероятность того, что в магазине окажется ровно 4 покупателя.

Итог работы: отчет, защита работы.

Практическая работа №25

Цель: овладеть навыками прогнозирования рисковых ситуаций методом экстраполяции

Задание 1:

Произвести расчет, используя метод экстраполяции, по сложившемуся среднегодовому коэффициенту роста числа негативных происшествий, сделайте прогноз относительно вашей компании по вариантам, постройте график, сформулируйте выводы.

	Количество происшествий по годам							
Вариант	2010	2011	2012	2013	2014	2015	2016	2017
1	42	45	59	38	45	49	63	58
2	113	110	116	115	120	110	115	124
3	496	524	488	500	493	579	462	536
4	880	887	876	897	880	888	869	880
5	22	23	20	15	13	17	20	18
6	58	59	73	54	55	58	62	60
7	100	69	97	105	116	85	70	76
8	74	110	69	107	109	71	106	77
9	87	74	69	66	115	86	96	96
10	99	92	95	97	85	94	117	81
11	79	103	85	102	104	73	74	92
12	99	86	99	104	104	99	87	112
13	119	82	81	96	101	86	93	86
14	101	75	95	84	72	114	80	76
15	120	88	80	112	98	71	67	110
16	104	119	117	78	112	87	114	116
17	105	119	73	113	87	83	111	94
18	116	119	113	100	87	70	73	70
19	70	67	109	111	95	114	102	78
20	72	102	88	114	100	96	73	85
21	91	115	97	114	116	85	86	104
22	120	134	88	128	102	98	126	109
23	131	94	93	108	113	98	105	98
24	85	82	124	126	110	129	117	93
25	132	100	92	124	110	83	79	122
26	106	130	112	129	131	100	101	119
27	117	131	85	125	99	95	123	106
28	146	109	108	123	128	113	120	113
29	82	79	121	123	107	126	114	90
30	147	115	107	139	125	98	94	137

Итог работы: отчет, защита работы.

Практическая работа №26

Цель: приобретение умений представлять статистические ряды в виде линейных моделей, оценивать адекватность моделей для возможности прогнозирования

Задание 1:

1. Построить линейную модель $\tilde{Y}(t) = a_0 + a_1 t$, параметры которой оценить МНК.
2. Оценить точность моделей на основе использования средней относительной ошибки аппроксимации.
3. По двум построенным моделям осуществить прогноз спроса на следующие две недели (доверительный интервал прогноза рассчитать при доверительной вероятности = 70%.
4. Фактические значения показателя, результаты моделирования и прогнозирования представить графически.

Итог работы: отчет, защита работы.

Практическая работа №27

Цель: отработать навыки решения задач

Задание 1:

1. Смоделируйте временные ряды по следующим формулам:

$$y_t^{(1)} = \alpha^{(1)} + \beta_1^{(1)} t + \beta_2^{(1)} \cos\left(\frac{2\pi}{5} t\right) + \varepsilon_t, \quad t = \overline{1,40},$$

$$y_t^{(2)} = \alpha^{(2)} + \beta_1^{(2)} t + \beta_2^{(2)} \cos\left(\frac{2\pi}{4} t\right) + \varepsilon_t, \quad t = \overline{1,40},$$

$$y_t^{(3)} = \begin{cases} \alpha^{(3)} + \beta_1^{(3)} t + \beta_2^{(3)} \cos\left(\frac{2\pi}{3} t\right) + \varepsilon, & t = 1,\text{K}\,,20,22,\text{K}\,,29,31,\text{K}\,,40, \\ -500 + \varepsilon_{21}, & t = 21, \\ 500 + \varepsilon_{30}, & t = 30. \end{cases}$$

Параметры, входящие в выражения задайте произвольным образом с учетом условия $|\beta_2^{(1)}|, |\beta_2^{(2)}|, |\beta_2^{(3)}| \le 3$. Значения $\varepsilon_1, \varepsilon_2, \text{K}\,, \varepsilon_n$ возьмите из нормального распределения $N(0, \sigma^2)$, дисперсию которого задайте также произвольно, например, $\sigma^2 = 2;4$. Наблюдения $y_{21}^{(3)}$ и $y_{30}^{(3)}$ называют *аномальными*.

2. Проведите сглаживание построенных временных рядов с помощью простой скользящей средней, используя для каждого ряда три интервала сглаживания: $g = 3$, $g = 4$, $g = 5$.

3. Постройте на одной диаграмме графики исходных и сглаженных временных рядов. Постройте графики остатков

$$e_t^{(i)} = y_t - \hat{y}_t^{(i)}, \quad i = 1,2,3.$$

4. По результатам исследования оформите отчет, в котором приведите результаты расчетов по сглаживанию рядов и графики, указанные в п. 3.

Итог работы: отчет, защита работы.

Практическая работа №28

Цель: отработать навыки решения задач

Задание 1:

Вариант 1

x_1	32	30	36	40	41	47	56	54	60	55	61	67	69	76
x_2	33	31	41	39	46	43	34	38	42	35	39	44	40	41
x_3	30	29	35	40	40	48	50	52	59	54	60	70	70	75
y	20	24	28	30	31	33	34	37	38	40	41	43	45	48

Вариант 2

x_1	55	46	40	39	35	29	31	75	68	66	60	54	59	53
x_2	33	42	45	38	40	30	32	40	39	43	38	34	41	37
x_3	50	45	39	40	34	30	30	74	69	66	59	54	60	52
y	33	32	30	29	27	23	19	47	44	42	40	39	37	36

Вариант 3

x_1	48	57	55	61	56	62	68	70	77	42	41	37	31	33
x_2	44	35	39	43	36	40	45	41	42	47	40	42	32	34
x_3	47	56	54	62	56	62	67	70	76	42	40	37	30	32
y	34	35	38	39	41	42	44	46	49	32	31	29	25	21

Вариант 4

x_1	52	54	45	39	38	34	28	30	74	67	65	59	53	58
x_2	36	32	41	44	37	39	29	31	39	38	42	37	33	40
x_3	52	53	45	38	38	34	28	31	73	66	65	60	52	57
y	35	32	31	29	28	26	22	18	46	43	41	39	33	36

Вариант 5

x_1	43	49	58	56	62	57	63	69	71	78	34	32	38	42
x_2	48	45	36	40	44	37	41	46	42	43	35	33	43	41
x_3	42	48	58	55	61	56	62	70	70	78	35	32	38	41
y	33	35	36	39	40	42	43	45	47	50	22	26	30	32

Вариант 6

x_1	52	57	51	53	44	38	37	33	27	29	73	66	64	58
x_2	32	39	35	31	40	43	36	38	28	30	38	37	41	36
x_3	52	56	50	53	45	37	37	32	28	30	72	66	64	59
y	37	35	34	31	30	28	27	25	21	17	45	42	40	38

Вариант 7

x_1	39	43	44	50	59	57	63	58	64	70	72	79	35	33
x_2	44	42	49	46	37	41	45	38	42	47	43	44	36	34
x_3	45	42	50	46	38	40	45	39	41	48	43	44	35	34
y	31	33	34	36	37	40	41	43	44	46	48	51	23	27

Вариант 8

x_1	63	57	51	56	50	52	43	37	36	32	26	28	72	65
x_2	40	35	31	38	34	30	39	42	35	37	27	29	37	36
x_3	39	38	35	35	32	31	28	28	25	25	21	15	45	40
y	39	37	36	34	33	30	29	27	26	24	20	16	44	41

Вариант 9

x_1	64	59	65	71	73	80	36	34	40	44	45	51	60	58
x_2	46	39	43	48	44	45	37	35	45	43	50	47	38	42
x_3	50	40	50	55	50	60	35	34	42	41	48	49	50	50
y	42	44	45	47	49	52	24	28	32	34	35	37	38	41

Вариант 10

x_1	46	52	61	59	65	60	66	72	74	81	37	35	41	45
x_2	51	48	39	43	47	40	44	49	45	46	38	36	46	44
x_3	46	52	60	58	64	61	65	72	74	80	38	34	40	44
y	36	38	39	42	43	45	46	48	50	53	25	29	33	35

Итог работы: отчет, защита работы.

Практическая работа №29

Цель: научиться выбирать оптимальную стратегию игры

Задание 1:

Борьба двух предприятий за рынок в регионе (N – номер варианта)

Две компании, занимающиеся производством антивирусного программного обеспечения, практически полностью делят рынок некоторого региона. Разрабатывая новую версию программного продукта для мобильных телефонов, каждая из компаний может использовать один из четырех вариантов продвижения нового программного продукта на рынок, который влияет на конечную стоимость продукции.

В зависимости от сделанного выбора компании могут установить цену реализации единицы продукции на уровне 25, 22, 19 и 16 условных единиц соответственно. Соотношение цен реализации и себестоимость представлены в таблице:

Вариант продвижения нового продукта	Цена реализации единицы продукции, у.е.	Полная себестоимость единицы продукции, у.е.	
		Компания А	Компания В
1	25	17	21–0,1*N
2	22	15	10+0,1*N
3	19	10+0,1*N	10

N – номер варианта, предложенный преподавателем.

В результате маркетингового исследования рынка была определена функция спроса на программные продукты:

$Y = 20 – 0,5*X$,

где Y – количество продукции, которое будет реализовано в регионе (тыс. ед.), а X – средняя цена продукции компаний, у.е.

Значения долей продукции, реализованной компанией А, зависят от соотношения цен на продукцию компании А и компании В. Маркетинговое исследование позволило установить эту зависимость:

Цена реализации 1 ед. продукции, у.е.		Доля реализованной продукции компании А
Компания А	Компания В	
25	25	0,31
25	22	0,33
25	19	0,25
25	16	0,2
22	25	0,4
22	22	0,35
22	19	0,32
22	16	0,28
19	25	0,52
19	22	0,48
19	19	0,4
19	16	0,35
16	25	0,6
16	22	0,58
16	19	0,55
16	16	0,5

1. Существует ли в данной задаче ситуация равновесия при выборе варианта продвижения продукта на рынок обоими компаниями?

2. Существуют ли варианты, которые компании заведомо не будут выбирать вследствие невыгодности?

3. Сколько продукции будет реализовано в ситуации равновесия? Какая компания получит больше прибыль в ситуации равновесия? Какая компания будет иметь большую долю рынка в ситуации равновесия? Дайте краткую экономическую интерпретацию результатов решения задачи.

Итог работы: отчет, защита работы.

Практическая работа №30

Цель: Актуализация знаний о симплекс - методе и принципе его решения с помощью программы Microsoft Excel, умение применять полученные знания на практике, при выполнении заданий

Задание 1:

. Дана итерация. Укажите разрешающей столбец, разрешающую строку, разрешающую ячейку

Итерация О

Базис	B	x_1	x_2	x_3	x_4	x_5	x_6
x_4	10	0	2	3	1	0	0
x_5	4	1	-2	1	0	1	0
x_6	11	3	2	1	0	0	1
F(X0)	0	0	-2	-3	0	0	0

Как будет записана итерация 1?

Итог работы: отчет, защита работы.

Практическая работа №31

Цель: отработать навыки решения задач

Задание 1:

1. Решить игру, заданную платежной матрицей , графоаналитическим способом.
2. Решить матричную игру, в которой один из игроков имеет две чистые стратегии, или игру, которая сводится к таковой после отбрасывания доминируемых строк и столбцов. Для нахождения цены игры и оптимальной стратегии игрока, имеющего две чистые стратегии, применяется графический метод.

Итог работы: отчет, защита работы.

4. ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ПРАКТИЧЕСКИХ РАБОТ

Основные:

0-1 Зубкова, Т. М. Технология разработки программного обеспечения : учебное пособие / Т. М. Зубкова. — Санкт-Петербург : Лань, 2019. — 324 с

Дополнительные:

Д-1 Немцова Т.И.. Практикум по информатике: учебное пособие / Т.И. Немцова, Ю.В. Назарова. — М : ИД "ФОРУМ"-ИНФРА-М — 2009. -437 с.

Интернет ресурсы

1. Зубкова, Т. М. Технология разработки программного обеспечения : учебное пособие / Т. М. Зубкова. — Санкт-Петербург : Лань, 2019. — 324 с— Текст : электронный // Лань : электронно-библиотечная система
2. Ганичева, А. В. Математическое моделирование и проектирование : учебное пособие / А. В. Ганичева. — Тверь : Тверская ГСХА, 2020. — Текст : электронный // Лань : электронно-библиотечная система

5. ЛИСТ ИЗМЕНЕНИЙ И ДОПОЛНЕНИЙ, ВНЕСЕННЫХ В МЕТОДИЧЕСКИЕ УКАЗАНИЯ

№ изменения, дата внесения, № страницы с изменением	
Было	**Стало**
Основание: **Подпись лица, внесшего изменения**	