Федеральное государственное образовательное бюджетное
учреждение высшего образования
«Финансовый университет при Правительстве Российской Федерации»
(Финуниверситет)

Самарский финансово-экономический колледж
(Самарский филиал Финуниверситета)

УТВЕРЖДАЮ
Заместитель директора по учебно-методической работе
_____Л.А Косенкова
«21» декабря 20 22 г.

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ОРГАНИЗАЦИИ И ВЫПОЛНЕНИЮ ПРАКТИЧЕСКИХ ЗАНЯТИЙ ПО ПРОФЕССИОНАЛЬНОМУ МОДУЛЮ «ПМ.02 ОСУЩЕСТВЛЕНИЕ ИНТЕГРАЦИИ ПРОГРАММНЫХ МОДУЛЕЙ»

СПЕЦИАЛЬНОСТЬ: 09.02.07 ИНФОРМАЦИОННЫЕ СИСТЕМЫ И ПРОГРАММИРОВАНИЕ

Самара – 2022

Методические указания по организации и выполнению практических занятий разработаны на основе рабочей программы по профессиональному модулю «Осуществление интеграции программных модулей», в соответствии с федеральным государственным образовательным стандартом среднего профессионального образования по специальности 38.02.06 Финансы, утвержденного приказом Министерства образования науки Российской Федерации от 09.12.2016 года № 1547, с учетом Профессионального стандарта, утвержденного приказом Министерства труда и социальной защиты Российской Федерации от 11 февраля 2014 г. № 647н «Об утверждении профессионального стандарта 06.011 Администратор баз данных» (зарегистрирован Министерством юстиции Российской Федерации 24 ноября 2014 г., регистрационный № 34846)
Присваиваемая квалификация: администратор баз данных

Разработчики:

Платковская Е.А. _____ Преподаватель Самарского филиала Финуниверситета

Методические указания по организации и выполнению практических занятий рассмотрены и рекомендованы к утверждению на заседании предметной (цикловой) комиссии естественно-математических дисциплин

Протокол от «24» января 20 22 г. № 5

Председатель ПЦК _____ М.В. Писцова

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Методические указания по выполнению практических работ по профессиональному модулю ПМ.02 Осуществление интеграции программных модулей разработаны с целью оказания помощи студентам специальности 09.02.07 Информационные системы и программирование и преподавателям по организации практических занятий по изучаемой дисциплине, в соответствии с требованиями федерального государственного стандарта среднего профессионального образования.

Методические разработка включает в себя краткие теоретические сведения, указания по выполнению практических работ, контрольные вопросы, формы контроля.

В соответствии с учебным планом на практические занятия для студентов отводится **56** часов.

В результате изучения профессионального модуля студент **должен освоить основной вид деятельности:** осуществление интеграции программных модулей и соответствующие ему общие компетенции и профессиональные компетенции:

Перечень общих компетенций

Код	Наименование общих компетенций
ОК 01	Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам.
ОК 02	Осуществлять поиск, анализ и интерпретацию информации, необходимой для выполнения задач профессиональной деятельности.
ОК 03	Планировать и реализовывать собственное профессиональное и личностное развитие.
ОК 04	Работать в коллективе и команде, эффективно взаимодействовать с коллегами, руководством, клиентами.
ОК 05	Осуществлять устную и письменную коммуникацию на государственном языке с учетом особенностей социального и культурного контекста.
ОК 06	Проявлять гражданско-патриотическую позицию, демонстрировать осознанное поведение на основе традиционных общечеловеческих ценностей.
ОК 07	Содействовать сохранению окружающей среды, ресурсосбережению, эффективно действовать в чрезвычайных ситуациях.
ОК 08	Использовать средства физической культуры для сохранения и укрепления здоровья в процессе профессиональной деятельности и поддержания необходимого уровня физической подготовленности.
ОК 09	Использовать информационные технологии в профессиональной деятельности.
ОК 10	Пользоваться профессиональной документацией на государственном и иностранном языке.
ОК 11	Планировать предпринимательскую деятельность в профессиональной сфере.

Перечень профессиональных компетенций

Код	Наименование видов деятельности и профессиональных компетенций
ВД 2	**Осуществление интеграции программных модулей**

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.

В результате освоения учебной дисциплины обучающийся **должен**
иметь практический опыт: в проектировании модели процесса разработки программного обеспечения; в применении основных принципов процесса разработки программного обеспечения; в использовании основных подходов к интегрированию программных модулей; в применении основ верификации и аттестации программного обеспечения.
уметь:
– использовать выбранную систему контроля версий;
– использовать методы для получения кода с заданной функциональностью и степенью качеств.
знать:
– модели процесса разработки программного обеспечения;
– основные принципы процесса разработки программного обеспечения;
– основные подходы к интегрированию программных модулей;
– основы верификации и аттестации программного обеспечения.

Количество часов, отводимое на освоение профессионального модуля

Всего – 272 часа
Из них на освоение
МДК.02.01 технология разработки программного обеспечения – 53 часа
объём работы обучающегося во взаимодействии с преподавателем – 42 часа;
самостоятельная работа – 10 часа.
МДК.02.02 Инструментальные средства разработки программного обеспечения – 63 часа
объём работы обучающегося во взаимодействии с преподавателем – 52 часа;
самостоятельная работа – 10 часов.
МДК.02.03 Математическое моделирование – 38 часов
объём работы обучающегося во взаимодействии с преподавателем – 32 часа;
самостоятельная работа – 6 часов;
на учебную практику – 72 часа;
на практику производственную (по профилю специальности) – 36 часов;
экзамен по модулю – 10 часов.

Количество практических работ по разделам

Раздел 1. Разработка программного обеспечения	18
Раздел 2. Средства разработки программного обеспечения	24

Раздел 3. Моделирование в программных системах	14

Характерная черта практических занятий – индивидуальное выполнение заданий, самостоятельное приобретение знаний. В связи с этим предусмотрены работы по всем основным разделам курса. Перед выполнением практической работы обучающийся получает опережающее теоретическое домашнее задание. На занятии объясняются вопросы, уточняются определения, которые помогают выполнению заданий. Обучающийся может просмотреть запись объяснения любой примерной работы по всем темам. И только после этого обучающийся приступает к выполнению практической работы.

При выполнении работы обучающийся должен самостоятельно изучить методические рекомендации по проведению практической работы, подготовить ответы на контрольные вопросы. Все практические задания выполняются за компьютером, теоретические вопросы сдаются устно или письменно.

После выполнения работы обучающийся должен представить отчет о проделанной работе с полученными результатами и в устной форме защитить.

При отсутствии по неуважительной причине обучающийся выполняет работу самостоятельно во внеурочное время и защищает на консультации по расписанию.

Структура практических работ:

1. Тема.
2. Цель.
3. Теоретическое обоснование.
4. Ход работы.
5. Контрольные вопросы.
6. Содержание отчета.
7. Литература.

При изучении дисциплины необходимо постоянно обращать внимание студентов на ее прикладной характер, показывать, где и когда изучаемые теоретические положения, и практические навыки могут быть использованы в будущей профессиональной деятельности.

ПЕРЕЧЕНЬ ПРАКТИЧЕСКИХ ЗАНЯТИЙ И ЛАБОРАТОРНЫХ РАБОТ

Практическая занятие №1. Анализ предметной области. Разработка и оформление технического задания.

Практическая занятие № 2. Построение архитектуры программного средства. Изучение работы в системе контроля версий.

Лабораторная работа №1. Построение диаграммы Вариантов использования и диаграммы. Последовательности. Построение диаграммы Кооперации и диаграммы Развертывания.

Лабораторная работа №2. Построение диаграммы Деятельности, диаграммы Состояний и диаграммы Классов. Изучение работы в системе контроля версий.

Лабораторная работа №3. Построение диаграммы компонентов. Построение диаграмм потоков данных.

Лабораторная работа №4. Разработка тестового сценария. Оценка необходимого количества тестов.

Лабораторная работа №5. Разработка тестовых пакетов.

Лабораторные работа №6. Оценка программных средств с помощью метрик.

Лабораторные работа №7. Инспекция программного кода на предмет соответствия стандартам кодирования.

Лабораторная работа №8. Разработка структуры проекта. Разработка модульной структуры проекта (диаграммы модулей).

Лабораторная работа №9. Разработка перечня артефактов и протоколов проекта.

Лабораторная работа №10. Настройка работы системы контроля версий (типов импортируемых файлов, путей, фильтров и др. параметров импорта в репозиторий).

Лабораторная работа №11. Разработка и интеграция модулей проекта (командная работа).

Лабораторная работа №12. Отладка отдельных модулей программного проекта. Организация обработки исключений.

Лабораторная работа №13. Применение отладочных классов в проекте. Отладка проекта.

Лабораторная работа №14. Инспекция кода модулей проекта

Лабораторная работа №15. Тестирование интерфейса пользователя средствами инструментальной среды разработки.

Лабораторная работа №16. Разработка тестовых модулей проекта для тестирования отдельных модулей.

Лабораторная работа №17. Выполнение функционального тестирования.

Лабораторная работа №18. Тестирование интеграции.

Лабораторная работа №19. Документирование результатов тестирования.

Лабораторная работа №20. Построение простейших математических моделей. Построение простейших статистических моделей. Решение простейших однокритериальных задач.

Лабораторная работа №21. Задача Коши для уравнения теплопроводности. Сведение произвольной задачи линейного программирования к основной задаче линейного программирования.

Лабораторная работа №22. Решение задач линейного программирования симплекс- методом. Нахождение начального решения транспортной задачи. Решение транспортной задачи методом потенциалов.

Лабораторная работа №23. Применение метода стрельбы для решения линейной краевой задачи. Задача о распределении средств между предприятиями. Задача о замене обо-

рудования. Нахождение кратчайших путей в графе. Решение задачи о максимальном потоке.

Практическая работа №3. Составление систем уравнений Колмогорова. Нахождение финальных вероятностей. Нахождение характеристик простейших систем массового обслуживания. Решение задач массового обслуживания методами имитационного моделирования.

Практическая работа №4. Построение прогнозов. Решение матричной игры методом итераций.

Лабораторная работа №24. Моделирование прогноза. Выбор оптимального решения с помощью дерева решений.

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ СТУДЕНТОВ ПО ВЫПОЛНЕНИЮ ПРАКТИЧЕСКИХ И ЛАБОРАТОРНЫХ РАБОТ

МДК. 02.01 ТЕХНОЛОГИЯ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Практическая работа №1. Анализ предметной области. Разработка и оформление технического задания.

Цель занятия:

- освоение технологии документирования программных средств на начальных стадиях проектирования ИС в соответствии с ЕСПД.

Краткие теоретические сведения

Основу отечественной нормативной базы в области документирования ПС составляет комплекс стандартов Единой системы программной документации (ЕСПД).

Основная и большая часть комплекса ЕСПД была разработана в 70-е и 80-е годы 20 века. Сейчас этот комплекс представляет собой систему межгосударственных стандартов стран СНГ (ГОСТ), действующих на территории Российской Федерации на основе межгосударственного соглашения по стандартизации.

Единая система программной документации - это комплекс государственных стандартов, устанавливающих взаимоувязанные правила разработки, оформления и обращения программ и программной документации.

Стандарты ЕСПД в основном охватывают ту часть документации, которая создается в процессе разработки программных средств, и связаны, по большей части, с документированием функциональных характеристик программных средств.

Следует отметить, что стандарты ЕСПД (ГОСТ 19) носят рекомендательный характер. Впрочем, это относится и ко всем другим стандартам в области ПС (ГОСТ 34, международному стандарту ISO/IEC и др.). Дело в том, что в соответствии с Законом РФ «О стандартизации» эти стандарты становятся обязательными на контрактной основе, т.е. при ссылке на них в договоре на разработку (поставку) программного средства.

Говоря о состоянии ЕСПД в целом, можно констатировать, что большая часть стандартов ЕСПД морально устарела. Тем не менее до пересмотра всего комплекса многие стандарты могут с пользой применяться в практике документирования программных средств.

К числу программных ЕСПД относит документы, содержащие сведения, необходимые для разработки, изготовления, сопровождения и эксплуатации программ.

Как известно, грамотно составленный пакет программной документации позволяет избежать при проектировании многих неприятностей. В частности, избавиться от назойливых вопросов и необоснованных претензий заказчика можно, просто отослав пользователя к документации. Это касается прежде всего важнейшего документа — Технического задания.

Техническое задание (ТЗ) содержит совокупность требований к программному средству и может использоваться как критерий проверки и приемки разработанной программы. Поэтому достаточно полно составленное (с учетом возможности внесения дополнительных разделов) и принятое заказчиком и разработчиком ТЗ является одним из основополагающих документов проекта программного средства.

ГОСТ 19.201-78, входящий в ЕСПД, устанавливает порядок построения и оформления технического задания на разработку программы или программного изделия для вычислительных машин, комплексов и систем независимо от их назначения и области применения.

Практические задания

Задание: разработать техническое задание на проектирование информационной системы, предназначенной для решения задач автоматизации деятельности организации.

Исходными данными для проектирования информационной системы являются описание предметной области и виды запросов в информационной системе (приложение 1).

Алгоритм выполнения работы

1) В соответствии с назначенным преподавателем вариантом определить наименование информационной системы (табл. 1), подлежащей проектированию в ходе лабораторного практикума, для удовлетворения основных требований к ней с применением системы управления базами данных Microsoft Access 2007 и/или инструментального средства Borland Turbo Delphi.

Таблица 1

№ варианта	Наименование информационной системы
1	Информационная система медицинских организаций города
2	Информационная система автопредприятия города
3	Информационная система проектной организации
4	Информационная система ГИБДД
5	Информационная система строительной организации
6	Информационная система библиотечного фонда города
7	Информационная система спортивных организаций города
8	Информационная система аэропорта
9	Информационная система гостиничного комплекса
10	Информационная система торговой организации
11	Информационная система ВУЗа
12	Информационная система железнодорожной пассажирской станции
13	Информационная система зоопарка
14	Информационная система театра
15	Информационная система фотоцентра

2) Изучить описание предметной области информационной системы (приложение 1).

3) На основании анализа описания предметной области и запросов к будущей информационной системе (приложение 1) сформулировать основные требования к ее функциям.

4) Выполнить поиск прототипа проектируемой информационной системы с применением Интернет.

5) Используя сформулированные требования к информационной системе, а также документацию пользователя на прототип найденного программного средства, разработать техническое задание в соответствии с ГОСТ 19.201-78 (приложение 2).

6) Ответить на контрольные вопросы.

Контрольные вопросы:

1) Как можно охарактеризовать понятие «программная документация»?

2) Что представляет собой внешняя и внутренняя программная документация?

3) Дайте определение понятию «единая система программной документации».

4) В чем заключаются основные недостатки единой системы программной документации?

5) Дайте определение понятию «техническое задание».

6) Объясните смысл понятия «документация пользователя».

7) Какими свойствами должна обладать документация пользователя? Дайте краткую характеристику.

Практическая работа № 2. Построение архитектуры программного средства. Изучение работы в системе контроля версий.

Цель занятия:

- реализация начальных этапов процесса разработки программного средства в соответствии с ГОСТ Р ИСО/МЭК 12207

Краткие теоретические сведения

При возникновении потребностей в заказе, приобретении, разработке, эксплуатации и сопровождении программ перед всеми сторонами, вовлеченными в жизненный цикл программного средства (ПС), возникает целый ряд вопросов, связанных с определением и детальным структурированием жизненного цикла (ЖЦ) ПС, с организационными и техническими правами и обязанностями сторон, с управлением ЖЦ и контролем за его реализацией. Одним из действенных инструментов для решения данных вопросов является использование унифицированных подходов, закрепленных в современных международных и российских стандартах.

Понятия «жизненный цикл системы» или «жизненный цикл программного средства» часто появляются в статьях и звучат в разговорах разработчиков, по крайней мере руководителей проектов и подразделений. Всем понятно, что относятся они к тому, что и в какой последовательности должно делаться при создании и эксплуатации систем. Но прежде чем две организации или два специалиста договорятся о том, что конкретно входит или не входит в ЖЦ, проходит значительное время. А позже вполне может обнаружиться, что эти двое (две «стороны») все-таки по-разному понимают, какие работы будут входить в ЖЦ, а какие - нет, какие проверки будут планироваться, когда и т. д. Естественно, общие принципы организации работ описаны давно, но что делать сторонам в конкретном проекте — это каждый раз приходится решать заново.

В стандартах, регламентирующих жизненный цикл программных средств, обобщаются опыт и результаты исследований множества специалистов и рекомендуются наиболее эффективные современные методы и процессы создания и развития комплексов программ. В результате таких обобщений оттачиваются технологические процессы и приемы разработки, а также методическая база для их автоматизации.

ЖЦ ПС в стандартах представляет собой набор этапов, частных работ и операций в последовательности их выполнения и взаимосвязи, регламентирующих ведение работ от подготовки технического задания до завершения испытаний ряда версий и окончания эксплуатации ПС или информационной системы (ИС).

Стандарты включают правила описания исходной информации, способов и методов выполнения операций, устанавливают правила контроля технологических процессов, требования к оформлению их результатов, а также регламентируют содержание технологических и эксплуатационных документов на комплексы программ. Они определяют организационную структуру коллектива, обеспечивают распределение и планирование заданий, а также контроль за ходом создания ПС.

Кроме вопросов выбора типа общего устройства ЖЦ есть проблемы с решением частных вопросов о включении или невключении в ЖЦ отдельных работ, очень важных для качества ПС и системы: что документировать при создании системы и ПС, какие работы должны будут гарантировать качество продукта, с какой степенью организационной независимости должны выполняться проверочные процедуры разных типов, чем будет обеспечиваться соответствие разрабатываемого ПС требованиям ко всей системе и соответствие ПС потребностям в системе.

Для того чтобы привнести порядок и понимание, общие для любых сторон, участвующих в ЖЦ систем и ПС, давно разрабатывались стандарты различных уровней утверждения - национальные и международные.

В России основы построения и использования профилей стандартов ЖЦ ПС заложены принятием в качестве базового стандарта ГОСТ Р ИСО/МЭК 12207. Данный документ введен в действие с 1 июля 2000 г., тесно взаимоувязан с рядом стандартов, принятых ранее, и с некоторыми стандартами, разрабатываемыми в данное время на основе прямого применения стандартов ИСО.

Актуальность стандарта ГОСТ Р ИСО/МЭК 12207 для современных условий настолько высока, что принятие в ISO его исходного, международного варианта вскоре вызвало самую положительную оценку российских экспертов. Был дан ряд рекомендаций, но его использованию в реальных условиях.

В данном стандарте программное обеспечение (ПО) или программный продукт определяется как набор компьютерных программ, процедур и, возможно, связанной с ними документации и данных.

Процесс определяется как совокупность взаимосвязанных действий, преобразующих некоторые входные данные в выходные. Каждый процесс характеризуется определенными задачами и методами их решения, исходными данными, полученными от других процессов, и результатами.

В соответствии с ГОСТ Р ИСО/МЭК 12207 все процессы ЖЦ ПО разделены на три группы:

1) Основные процессы:
- приобретение;
- поставка;
- разработка;
- эксплуатация;
- сопровождение.

2) Вспомогательные процессы:
- документирование;
- управление конфигурацией;
- обеспечение качества;
- верификация;
- аттестация;
- совместная оценка;
- аудит;
- разрешение проблем.

3) Организационные процессы:
- управление;
- усовершенствование;
- создание инфраструктуры;
- обучение.

Процесс разработки предусматривает действия и задачи, выполняемые разработчиком, и включает следующие действия:

А) Подготовительная работа начинается с выбора модели ЖЦ ПО, соответствующей масштабу, значимости и сложности проекта. Действия и задачи процесса должны соответствовать выбранной модели. Разработчик должен выбрать, адаптировать к условиям проекта и использовать согласованные с заказчиком стандарты, методы и средства разработки, а также составить план выполнения работ.

Б) Анализ требований к системе подразумевает определение ее функциональных возможностей, пользовательских требований, требований к надежности и безопасности, требований к внешним интерфейсам и т.д. Требования к системе оцениваются исходя из критериев реализуемости и возможности проверки при тестировании.

Анализ требований к ПО предполагает определение следующих характеристик для каждого компонента ПО:

- функциональных возможностей, включая характеристики производительности и среды функционирования компонента;
- внешних интерфейсов;
- спецификаций надежности и безопасности;
- эргономических требований;
- требований к используемым данным;
- требований к установке и приемке;
- требований к пользовательской документации;
- требований к эксплуатации и сопровождению.

Требования к ПО оцениваются исходя из критериев соответствия требованиям к системе, реализуемости и возможности проверки при тестировании.

В) Проектирование архитектуры системы на высоком уровне заключается в определении компонентов ее оборудования, ПО и операций, выполняемых эксплуатирующим систему персоналом. Архитектура системы должна соответствовать требованиям, предъявляемым к системе, а также принятым проектным стандартам и методам.

Проектирование архитектуры ПО включает задачи (для каждого компонента ПО):

- трансформацию требований к ПО в архитектуру, определяющую на высоком уровне структуру ПО и состав ее компонентов;
- разработку и документирование программных интерфейсов ПО и баз данных;
- разработку предварительной версии пользовательской документации;
- разработку и документирование предварительных требований к тестам и планам интеграции ПО.

Архитектура компонентов ПО должна соответствовать требованиям, предъявляемым к ним, а также принятым проектным стандартам и методам.

Г) Детальное проектирование ПО включает следующие задачи:

- описание компонентов и интерфейсов между ними на более низком уровне, достаточном для их последующего самостоятельного кодирования и тестирования;
- разработку и документирование детального проекта базы данных;
- обновление (при необходимости) пользовательской документации;
- разработку и документирование требований к тестам и плана тестирования компонентов ПО;
- обновление плана интеграции ПО.

Д) Кодирование и тестирование ПО охватывает задачи:

- разработку и документирование каждого компонента ПО и базы данных а также совокупности тестовых процедур и данных для их тестирования;
- тестирование каждого компонента ПО и базы данных на соответствие предъявляемых к ним требованиям. Результаты тестирования компонентов должны быть документированы;
- обновление (при необходимости) пользовательской документации;
- обновление плана интеграции ПО.

Е) Интеграция ПО предусматривает сборку разработанных компонентов ПО в соответствии с планом интеграции и тестирование агрегированных компонентов. Для каждого из агрегированных компонентов разрабатываются наборы тестов и тестовые процедуры, предназначенные для проверки каждого из квалификационных требований при последующем квалификационном тестировании.

Интеграция системы заключается в сборке всех ее компонентов, включая ПО и оборудование. После интеграции система, в свою очередь, подвергается квалификационному тестированию на соответствие совокупности требований к ней. При этом также производится оформление и проверка полного комплекта документации на систему.

Ж) Квалификационное тестирование - это набор критериев и условий, которые необходимо выполнить, чтобы квалифицировать программный продукт как соответствующий своим спецификациям и готовый к использованию в условиях эксплуатации.

Квалификационное тестирование ПО проводится разработчиком в присутствии заказчика (по возможности) для демонстрации того, что ПО удовлетворяет своим спецификациям и готово к использованию в условиях эксплуатации. Квалификационное тестирование выполняется для каждого компонента ПО по всем разделам требований при широком варьировании тестов. При этом также проверяются полнота технической и пользовательской документации и ее адекватность самим компонентам ПО.

З) Установка ПО осуществляется разработчиком в соответствии с планом в той среде и на том оборудовании, которые предусмотрены договором. В процессе установки проверяется работоспособность ПО и баз данных. Если устанавливаемое программное обеспечение заменяет существующую систему, разработчик должен обеспечить их параллельное функционирование в соответствии с договором.

И) Приемка ПО предусматривает оценку результатов квалификационного тестирования ПО и системы и документирование результатов оценки, которые проводятся заказчиком с помощью разработчика. Разработчик выполняет окончательную передачу ПО заказчику в соответствии с договором, обеспечивая при этом необходимое обучение и поддержку.

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, еѐ 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, расчѐтов и моделирования. Типовые требования к составу и содержанию технического задания приведены в таблице 1.

Таблица 1. Состав и содержание технического задания (ГОСТ 34.602- 89)

№ 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».

Практические задания

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.
3. Сдать и защитить работу

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

Контрольные вопросы:

1. Какие текущие проблемы существуют в организации и как новая система поможет их решить?
2. Каким образом (и будет ли) ИС способствовать целям бизнеса?
3. Требует ли разработка ИС технологии, которая до этого раньше не использовалась в организации?
4. Где будет размещена ИС? Кто является пользователем ИС?

5. Комплекс технических средств ИТ

Лабораторная работа №1. Построение диаграммы Вариантов использования и диаграммы. Последовательности. Построение диаграммы Кооперации и диаграммы Развертывания.

Цель занятия:

- исследование процесса построения диаграмм прецедентов и диаграмм взаимодействий в заданной предметной области с помощью пакета Rational Rose

Краткие теоретические сведения

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».

Практические задания

Задание 1. Создание диаграммы вариантов использования и действующих лиц. Окончательный вид диаграммы показан на рис. 1.

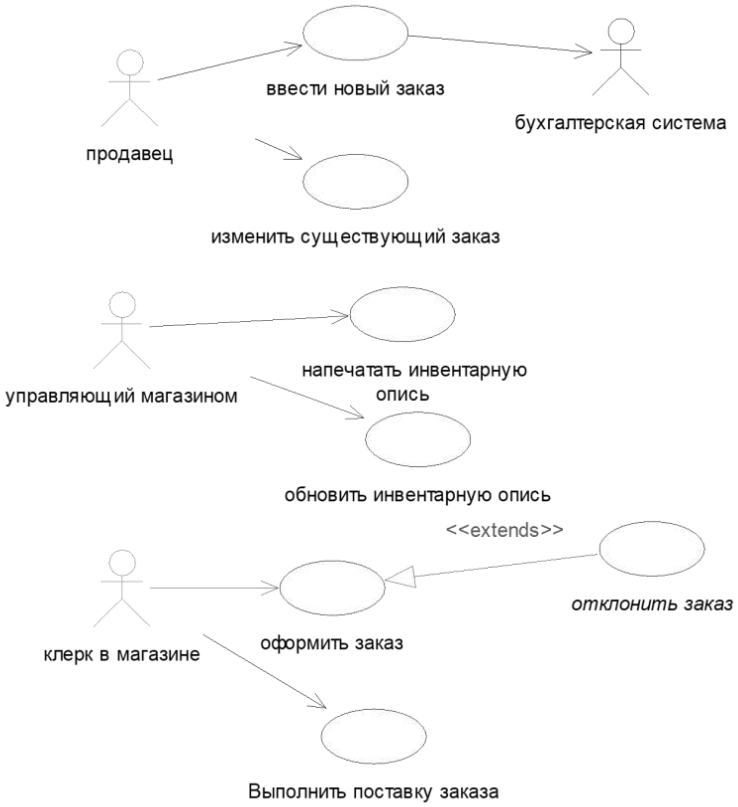


Рис. 1 Диаграмма вариантов использования задачи о заказе товара.

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. Создание диаграммы Последовательности.

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.

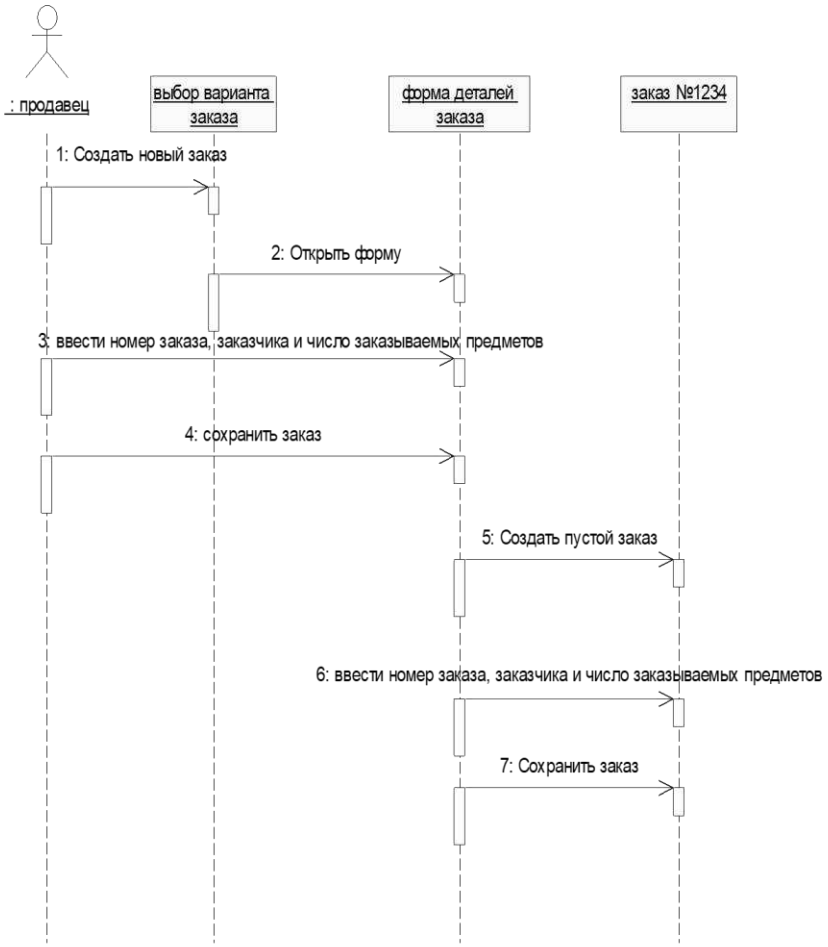


Рис. 2. Диаграмма последовательности без управляющих элементов.

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 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

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 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 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* — 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.

Диаграмма должна выглядеть, как на рис. 3.

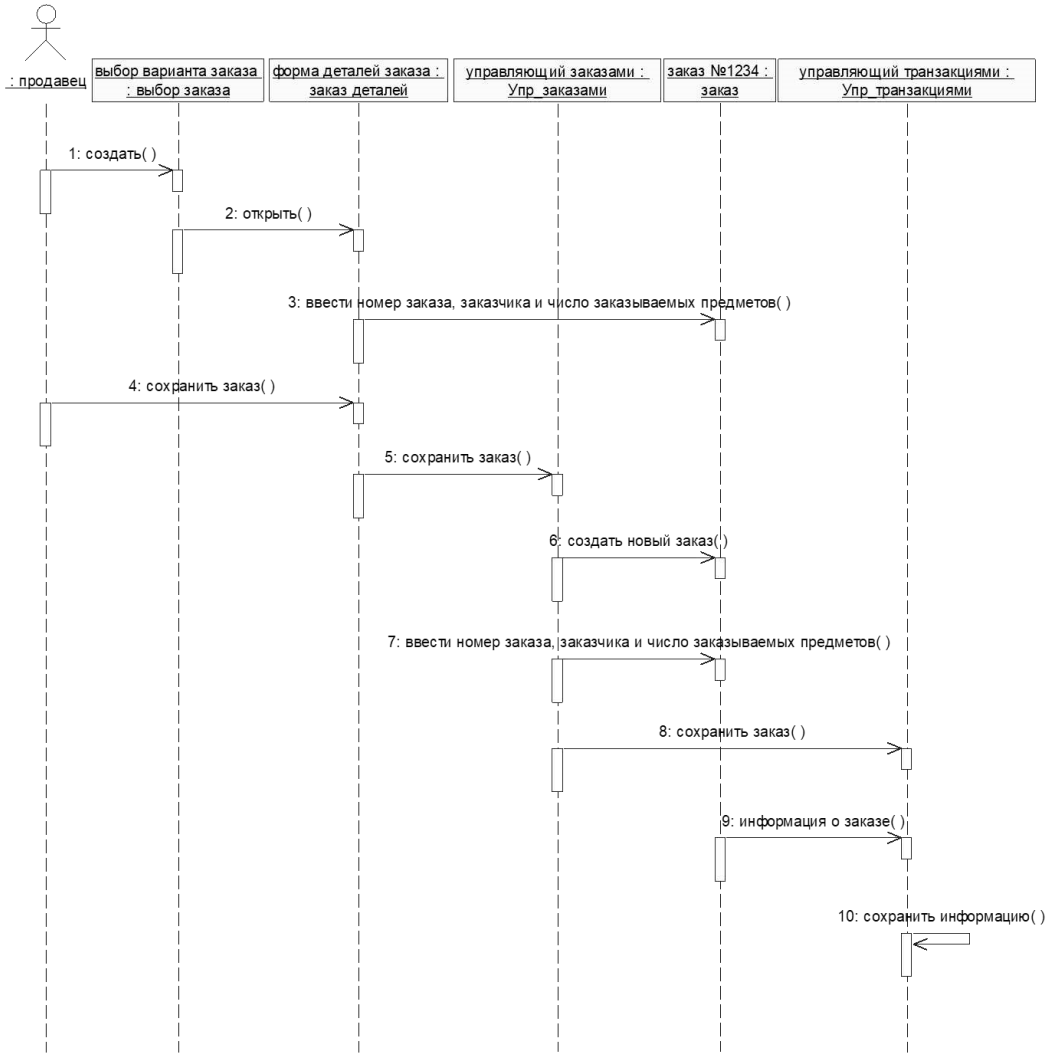


Рис. 3. Окончательный вид диаграммы последовательности

Контрольные вопросы:

1. Какие средства компьютерной техники необходимы для ИС?
2. Какие средства коммуникационной техники необходимы для ИС?
3. Какие средства организационной техники необходимы для ИС?
4. Какие средства оперативной полиграфии необходимы для ИС?

Лабораторная работа №2. Построение диаграммы Деятельности, диаграммы Состояний и диаграммы Классов. Изучение работы в системе контроля версий.

Цель занятия:

- исследование процесса построения диаграммы состояний и диаграммы классов в заданной предметной области с помощью пакета Rational Rose 2000.

Практические задания

Задание 1. Постройте диаграмму Состояний

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

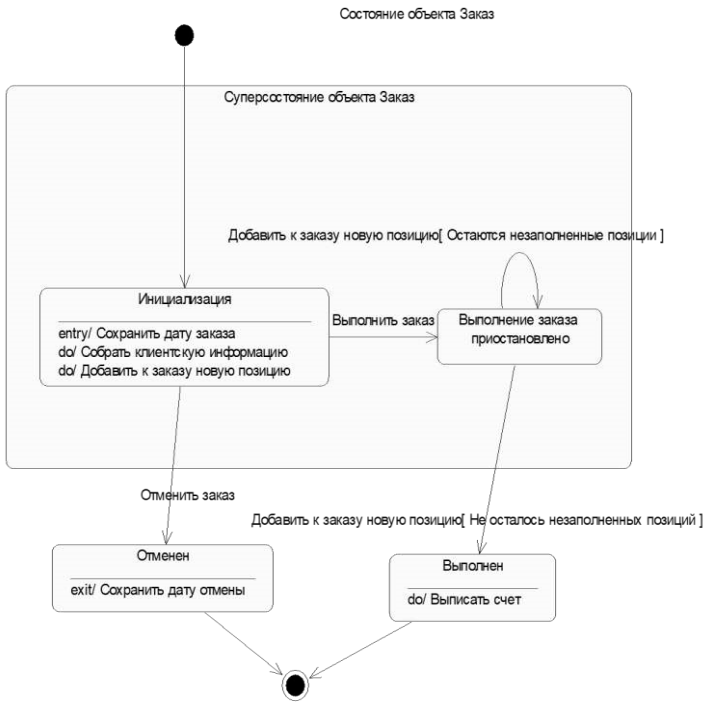


Рис 5. Диаграмма состояний для класса *Заказ*

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 ОК, 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 ОК, 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 ОК, 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 ОК, 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 ОК, 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 ОК, 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.

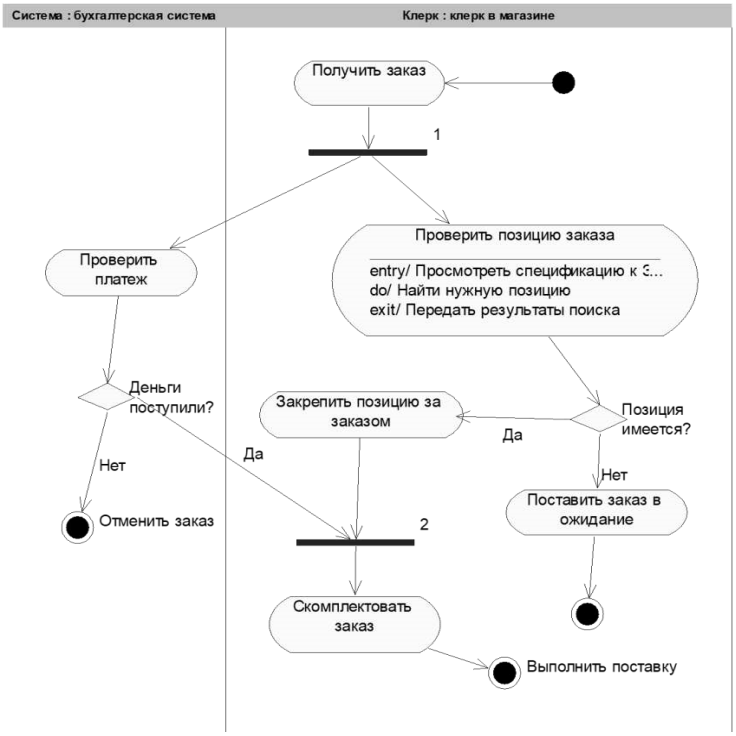


Рис. 6 Диаграмма активности для варианта использования «Выполнить поставку заказа»

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

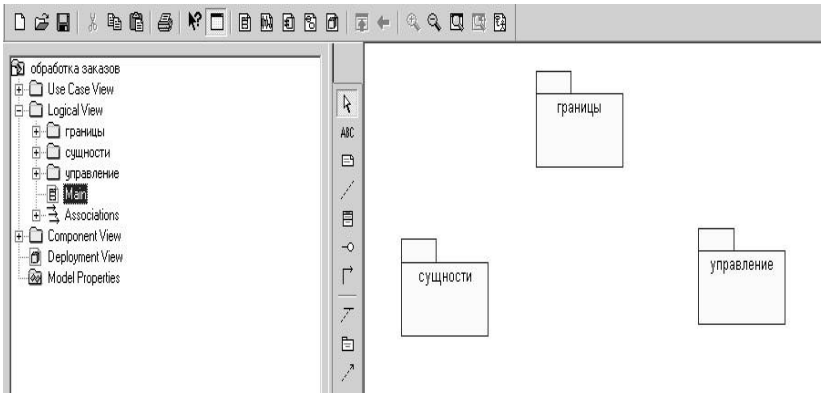


Рис. 7 Главная диаграмма классов в логическом представлении браузера.

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.
3. Перетащите классы *Упр_заказами* и *Упр-транзакциями* на пакет Управление.
4. Перетащите класс *Заказ* на пакет Сущности.

Классы и пакеты в браузере показаны на рис. 9

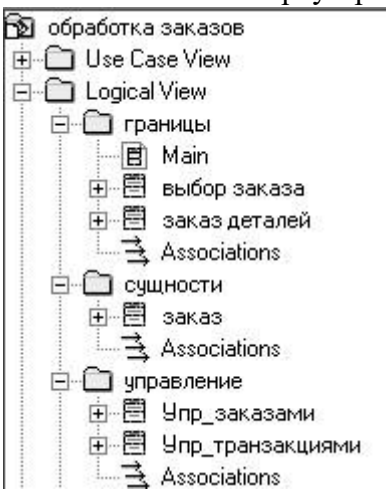


Рис. 8 Представление пакетов и классов

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 C1ass 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()

6. Нажмите клавишу Enter.
7. Введите операцию:

Дать_информацию()

Подробное описание операций с помощью диаграммы Классов 1. Щелкнув мышью на классе *Заказ*, выделите его.

2. Щелкните на этом классе еще раз, чтобы переместить курсор внутрь.
3. Отредактируйте операцию *Создать()*, чтобы она выглядела следующим образом: Создать() : Boo1ean
4. Отредактируйте операцию *Взять_информацию*:

Взять_информацию (OrderNum : Integer, Customer : String, OrderDate : Date, FillDate : Date):

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 Boo1ean.

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 Boo1ean.

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() : Boo1ean

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

Сохранить заказ(OrderID : Integer) : Boo1ean

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

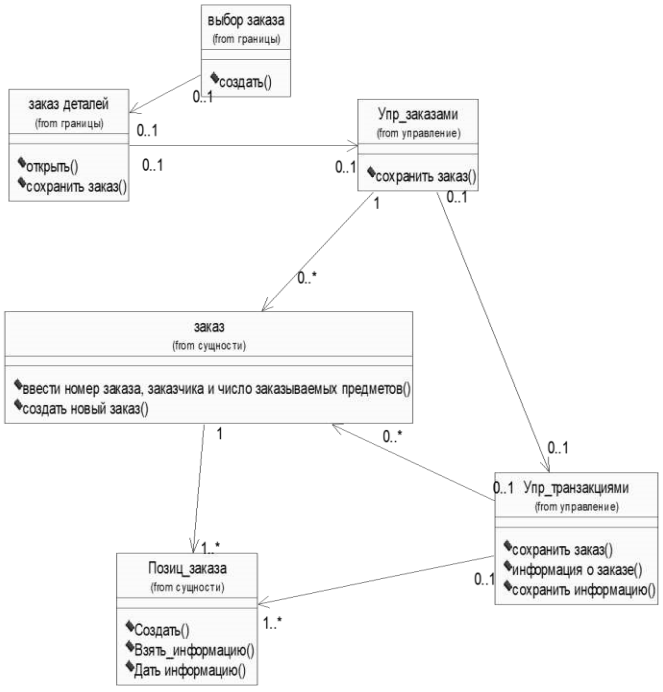


Рис. 10 Ассоциации сценария "Ввести новый заказ"

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.

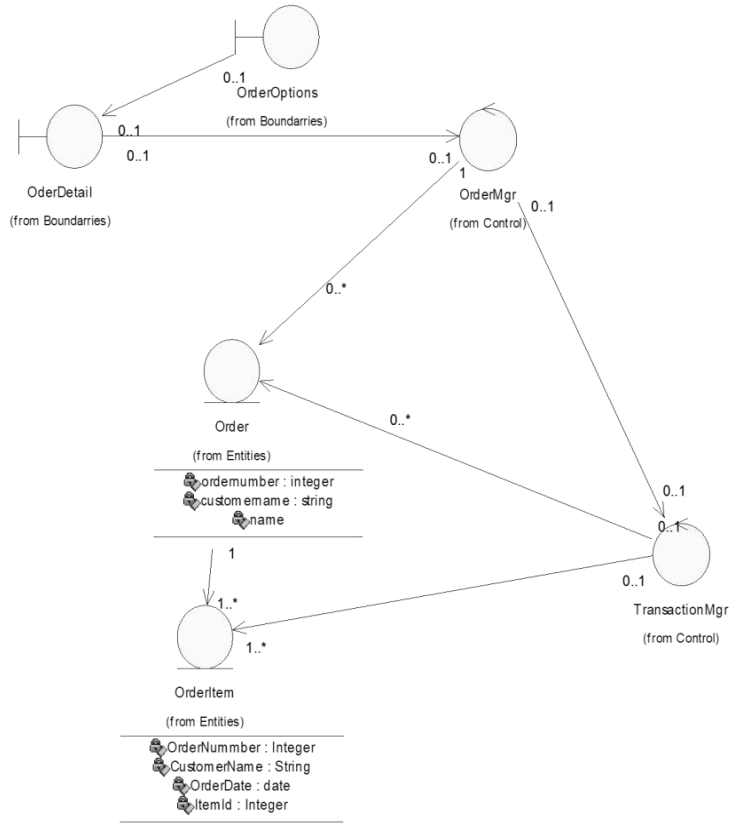


Рис. 11 Основная диаграмма классов

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)

Контрольные вопросы:

1. Перечислите критерии качества.
2. Дайте определению понятию «метрики»
3. Как построить модель управления качеством?
4. Перечислите основные пункты национального стандарта обеспечения качества АИС.
5. Для чего предназначена международная система стандартизации и сертификации качества продукции
6. Каким образом можно автоматизировать системы управления качеством разработки?

Лабораторная работа №3. Построение диаграммы компонентов. Построение диаграмм потоков данных.

Цель занятия:

- исследование процесса построения диаграммы потоков данных и диаграммы размещения в заданной предметной области с помощью пакета Rational Rose и Delphi.

Практические задания

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)

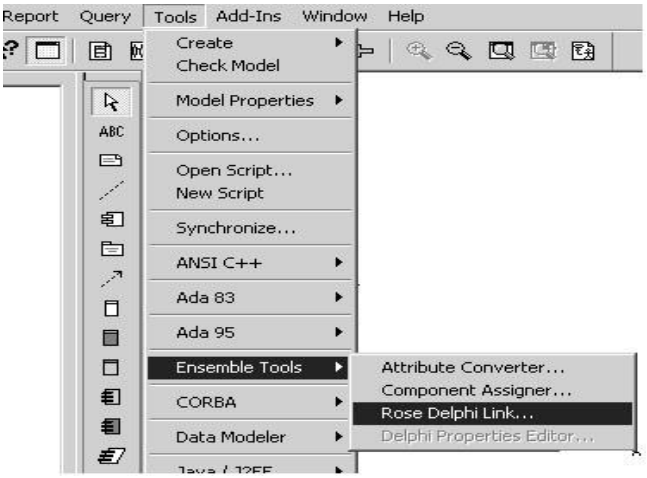


Рис. 14. Меню для выбора процесса кодогенерации

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)

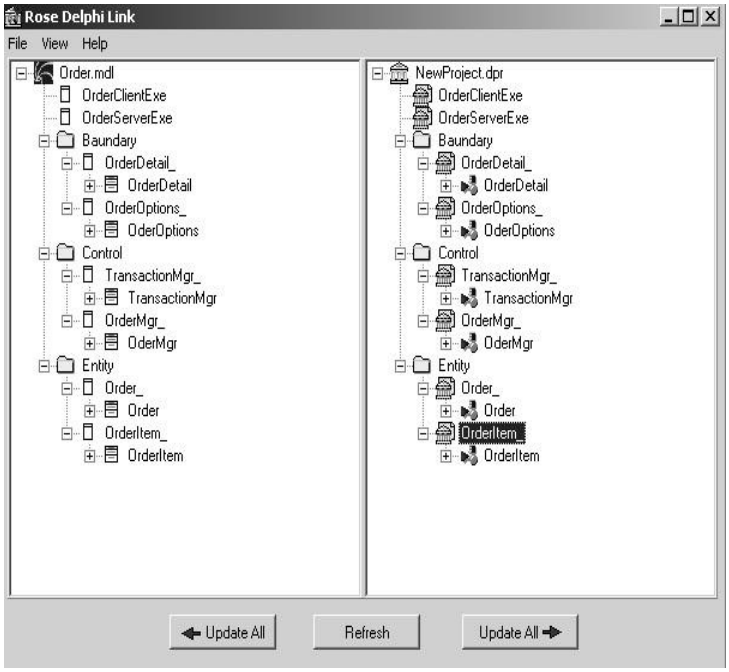


Рис. 15. Представление результатов кодогенерации в окне Rose Delphi Link

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):

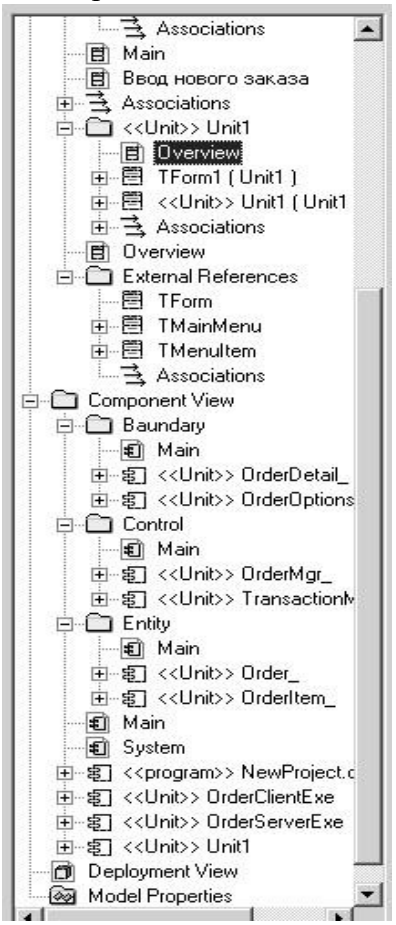


Рис. 16. Окно Rose Delphi Link после кодогенерации

- 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

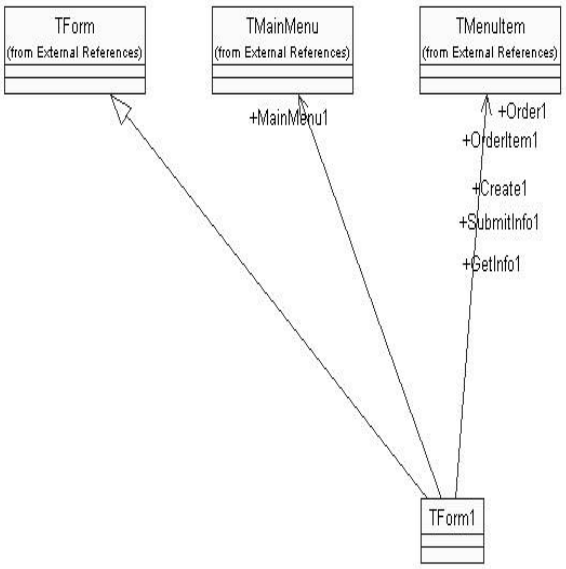


Рис. 17. Результаты реинжиниринга проекта Delphi в Rose

Задание 3. Построение диаграммы размещения

В этом задании создается диаграмма Размещения для системы обработки заказов.

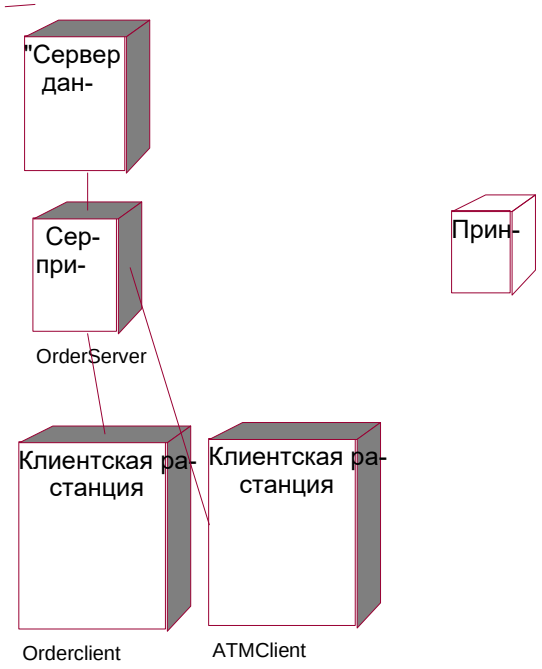


Рис. 18 Диаграмма размещения для модельной задачи

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

Контрольные вопросы:

1. Что такое DFD? Какие новые типы объектов может содержать DFD? Какие физические характеристики системы отражаются в DFD моделях? Область применения DFD.
2. Привести пример контекстной DFD-диаграммы.
3. Назначение функциональных блоков, внешних сущностей и стрелок (потоков данных) в DFD.
4. Какова функция хранилищ данных в производственных и информационных системах?
5. Привести пример разветвления и объединения стрелок.
6. Каковы основные подходы к построению DFD-моделей?
7. Принципы нумерации объектов в DFD.

Лабораторная работа №4. Разработка тестового сценария. Оценка необходимого количества тестов.

Цель занятия:

- усвоить знание о видах тестирования; освоить способы обнаружения и фиксирования ошибок.

Краткие теоретические сведения

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.

Практические задания

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

Контрольные вопросы:

1. Что такое тестирование? Что является объектами тестирования?
2. Опишите виды тестирования.
3. Поясните понятия «тест», «тестовые данные», «тестовый эксперимент».

Лабораторная работа №5. Разработка тестовых пакетов.

Цель занятия:

- получить навыки разработки тестовых пакетов.

Практические задания

- 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 № 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**
Оформить отчет.

Контрольные вопросы:

1. Что такое тестирование?
2. Что такое тестовый сценарий?
3. Что используется в качестве основы для разработки тестового сценария?
4. Что такое тест кейс? Из чего состоит тест-кейс? Принципы написания тест-кейсов.

Лабораторные работа №6. Оценка программных средств с помощью метрик.

Цель занятия:

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

Краткие теоретические сведения

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.

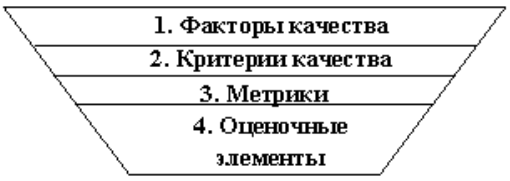


Рис. 1 – Уровни системы показателей качества

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 спецификирования 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.

Практические задания

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

Контрольные вопросы:

1. Что такое метрика качества кода?
2. Как определить качество кода, используя метрики?
3. Приведите порядок действия для оценки качества кода

Лабораторные работа №7. Инспекция программного кода на предмет соответствия стандартам кодирования.

Цель занятия:

- научиться выполнять реорганизацию программного кода на основании шаблонов рефакторинга.

Практические задания

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 ста-
тика елизавета елизавета; статический VendingMachine vending-
Machine; статическое учебное пособие;
частные статические CutScenes cutScene; публичная ста-
тика 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.up-
date (); booker.update ();
if (! energetic.getName (). equals
("")) energetic.update (); if (level-
Number> 0) elizabeth.update ();
если (молния! = ноль) { light-
ning.update ();
if (lightning.getDelete ())
молния = ноль;
} menu.update
(); hud.update
(); weapon.up-
date ();
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.cre-
ateLevels (levelNumber);
level.changeImageView (levelNumber);
vendingMachine = new VendingMachine
(); букер = новый персонаж ();
booker.setMoney (dataInputStream.readInt
()); booker.setSalt (dataIn-
putStream.readByte ());
booker.setCountLives (2); оружие = новое
оружие ();
weapon.setWeaponClip (dataInputStream.readInt
()); weapon.setBullets (dataInputStream.readInt
()); hud = новый HUD ();
```

```
Елизавета = новая Елизавета (); энергич-
ный = новый Энергетический ();
} catch (IOException e) { level-
Number = 0;
level.createLevels (levelNumber); vend-
ingMachine = 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 Enemy-
Comstock (Level_data.BLOCK_SIZE * 161, 200)); враги.адд (новый En-
emyComstock (Level_data.BLOCK_SIZE * 171, 200)); враги.адд (новый
EnemyComstock (Level_data.BLOCK_SIZE * 185, 200)); враги.адд (но-
вый 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 Enemy-
Comstock (Level_data.BLOCK_SIZE * 252, 200)); враги.адд (новый En-
emyComstock (Level_data.BLOCK_SIZE * 262, 200)); враги.адд (новый
EnemyComstock (Level_data.BLOCK_SIZE * 277, 200)); враги.адд (но-
вый EnemyComstock (Level_data.BLOCK_SIZE * 280, 200)); враги.адд
(новый EnemyComstock (Level_data.BLOCK_SIZE * 286, 200)); пере-
мена; 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 Enemy-
Comstock (Level_data.BLOCK_SIZE * 87, 200)); враги.адд (новый Ene-
myRedEye (Level_data.BLOCK_SIZE * 104, 150)); враги.адд (новый
EnemyComstock (Level_data.BLOCK_SIZE * 117, 200)); враги.адд (но-
вый 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 Ene-
myRedEye (Level_data.BLOCK_SIZE * 224, 200)); враги.адд (новый
EnemyComstock (Level_data.BLOCK_SIZE * 246, 200)); враги.адд (но-
вый 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) { gam-
eRoot.setLayoutX (- (смещение - 600)); level.getBackground
(). setLayoutX ((смещение - 600) / 1,5);
}
если (смещение <= 100) level.getBack-
ground () 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. Персо-
нажи. *; импорт игры. Персонажи.
Персонажи; импорт игры. Энерге-
тика. 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.ani-
mation.AnimationTimer; импорт ja-
vafx.application.Application; import
javafx.scene.Scene;
import javafx.scene.image.Im-
age; import javafx.scene.in-
```

```
put.KeyCode; import ja-
vafx.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 Ar-
rayList <Block> blocks = new ArrayList <> (); public static Ar-
rayList <Bullet> bullets = new ArrayList <> (); public static Ar-
rayList <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 (DataIn-
putStream dataInputStream = new DataInputStream (новый FileInputStream ("C:
/DeadShock/saves/data.dat"))) { levelNumber =
dataInputStream.readInt (); level.cre-
ateLevels (levelNumber);
level.changeImageView (levelNumber);
vendingMachine = new VendingMachine
(); букер = новый персонаж ();
booker.setMoney (dataInputStream.readInt
()); booker.setSalt (dataIn-
putStream.readByte ());
booker.setCountLives (2); оружие = новое
оружие ();
weapon.setWeaponClip (dataInputStream.readInt
()); weapon.setBullets (dataInputStream.readInt
()); hud = новый HUD ();
Елизавета = новая Елизавета (); энергич-
ный = новый Энергетический ();
} catch (IOException e) {
levelNumber = 0; level.createLevels
(levelNumber); vendingMachine = new
```

```
VendingMachine (); букер = новый пер-
сонаж (); оружие = новое оружие ();
hud = новый HUD (); энергичный = но-
вый Энергетический (); tutorial = new
Tutorial (levelNumber);
} createEnemies (); 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.setLay-
outX (- (смещение - 600));
level.getBackground (). setLayoutX ((смещение - 600) / 1,5);
}
если (смещение <= 100) level.getBack-
ground () 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 Enemy-
Comstock (Level_data.BLOCK_SIZE * 161, 200)); враги.адд (новый En-
emyComstock (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 Enemy-
Comstock (Level_data.BLOCK_SIZE * 243, 200)); враги.адд (новый En-
emyComstock (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.GetDe-
lete()) { enemies.remove(enemy);
break;
}
}
Bullet.update(); Controller.up-
date(); booker.update();
If (!energetic.Get-
Name().Equals("")) energetic.up-
date(); if (levelNumber > 0) eliza-
beth.update(); if (lightning !=
null) { lightning.update(); If
(lightning.GetDelete()) lightning
= null;
} menu.up-
date(); hud.up-
date();
weapon.up-
date();
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/im-
ages/icon.jpg")); stage.setTitle("DeadShock"); stage.setResiza-
ble(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)

Контрольные вопросы:

1. Для чего нужны стандарты кодирования.
2. Какие существуют соглашения по именованию переменных, функций.
3. В каком формате следует записывать константы в тексте программы.
4. Какие стили расстановки скобок существуют? В чем их различия?

МДК.02.02 ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Лабораторная работа №8. Разработка структуры проекта. Разработка модульной структуры проекта (диаграммы модулей).

Цель занятия:

- формирование навыков постановки задачи и разработки технического задания на программный продукт.

Практические задания

Задание

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?

Лабораторная работа №9. Разработка перечня артефактов и протоколов проекта.

Цель занятия:

- научиться составлять артефакты и протоколы проекта.

Краткие теоретические сведения

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.

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 доку-

ментом 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.

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.

Практические задания

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?

Лабораторная работа №10. Настройка работы системы контроля версий (типов импортируемых файлов, путей, фильтров и др. параметров импорта в репозиторий).

Цель занятия:

- научиться устанавливать, настраивать и работать с репозиторием GitHub.

Краткие теоретические сведения

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 ведѐтся работа с большим количеством непрерывно изменяющихся электронных документов, в частности, они всѐ чаще применяются в САПР, обычно, в составе систем управления данными об изделии (PDM). Управление версиями используется в инструментах конфигурационного управления (Software Configuration Management Tools).

Распространѐнные 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

пакета Microsoft Visual Studio и интегрирован с продуктами этого пакета. Доступен только для платформы Windows. Версию для Unix поддерживает компания MainSoft. В ноябре 2005 года вышла обновлѐнная версия продукта — Visual SourceSafe 2005, обещающая повышенную стабильность и производительность, улучшенный механизм слияния для XML-файлов и файлов в Юникоде, а также работу через HTTP. Visual SourceSafe нацелен на индивидуальных разработчиков либо небольшие команды разработчиков. Там где VSS недостаточно, ему на замену предлагается новый продукт Майкрософт — Team Foundation Server, входящий в состав Visual Studio Team System.

Rational Clear Case

ClearCase поддерживает следующие возможности, разительно отличающие его в лучшую сторону от других средств контроля:

- Общий контроль версий не только файлов, но и директорий/поддиректорий;
- Бесконечное число ответвлений от определенной версии;
- Автоматическая компрессия файлов и их кеширование (CC позволяет хранить большое количество данных, при всем при этом база данных остается компактной и быстрой);
- Позволяет легко конвертировать базы данных других средств контроля, например: PVCS, SourceSafe, RCS, CVS и SCCS;
- Поддерживает параллельную разработку и мультикомандные подразделения, расположенные в географически удаленных друг от друга местах;
- Мультиплатформенность (способен объединить единой средой участников, работающих на разных операционных системах);
- Имеет интеграцию со средствами разработки;
- Имеет Web-интерфейс для удаленного контроля. CVS

CVS (Concurrent Versions System, "Система Конкурирующих Версий"). Хранит историю изменений определѐнного набора файлов, как правило исходного кода программного обеспечения, и облегчает совместную работу группы людей (часто — программистов) над одним проектом. CVS популярна в мире открытого ПО. Система распространяется на условиях лицензии GNU GPL.

Subversion

Subversion — централизованная система (в отличие от распределѐнных систем, таких, как Git или Mercurial), то есть данные хранятся в едином хранилище. Хранилище может располагаться на локальном диске или на сетевом сервере. Работа в Subversion мало отличается от работы в других централизованных системах управления версиями. Для совместной работы над файлами в Subversion преимущественно используется модель Копирование-Изменение-Слияние. Кроме того, для файлов, не допускающих слияние (различные бинарные форматы файлов), можно использовать модель Блокирование-Изменение-Разблокирование.

Практические задания

Задание.

1. Настроить подключение к репозиторию
2. Скачать проект
3. Добавить свой класс к проекту
4. Внести изменения к класс
5. Обновить класс в репозитории
6. Удалить все локальные файлы и скачать проект из репозитория
7. Добавить "лишний" файл в репозиторий и затем удалить его из репозитория.
8. Изучить журнал изменений файлов, посмотреть какие изменения внесены другими разработчиками.

Примечание: опробовать Git, Subversion, Mercurial (локально) Ссылки на учебные репозитории:

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

Контрольные вопросы:

1. В чем заключается экономия времени при использовании системы контроля версий?
2. В чем преимущества использования системы контроля версий?
3. Что такое Git?
4. Как начать использовать git?
5. Как начать использовать GitHub?
6. Основные (наиболее часто используемые) команды Git.
7. Какие сервисы существуют для Git?
8. Как работать с локальным репозиторием?
9. Как работать с распределенным репозиторием?

Лабораторная работа №11. Разработка и интеграция модулей проекта (командная работа).

Цель занятия:

– освоить процесс проектирования модулей программного обеспечения.

Практические задания

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 10х10 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 10х10 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 10х10 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 10х10 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 10х10 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.

Задание 2.

1. Разработать модули программы, спроектированные во время практического занятия
2. Отладить программу с использованием тестов, составленных во время практического занятия
3. Составить отчет по практической работе.

Контрольные вопросы:

1. Какие ошибки в программах существуют?
2. Что понимают под отладкой программы?
3. Чем отладка отличается от тестирования?

Лабораторная работа №12. Отладка отдельных модулей программного проекта. Организация обработки исключений.

Цель занятия:

- изучить основные подходы к проектированию тестов.

Краткие теоретические сведения

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 107 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 1018 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!

Практические задания

Задание:

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).

5. Оформить отчет по лабораторной работе.

Отчет по лабораторной работе должен включать:

1. Внешнюю спецификацию.
2. Алгоритм решения задачи.
3. Текст программы на языке программирования.
4. Набор тестов для отладки программы, соответствующий конкретным методам «белого ящика».

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 X0 и шаг изменения аргумента h задаются пользователем.

Точка (точки), в которой функция $\left\|\frac{2x^3-3}{x^2+1}\right\|$ равна 10.
Локальный минимум функции $3(x+4)^2-\sin(x+15)$
Точка (точки), в которой функция $\frac{2x^2}{\sin(x-1)}-1$ равна -500.
Локальный максимум функции $\frac{2x+15}{3+x^2}$
Пересечение графиков функций $(x^2+1)\sin(3+x)$ и $\frac{x+5}{x^2+1}$.
Локальный минимум функции $x^2-3x+15$
Все нули функции $\sqrt{x^2+0.5}-\sin(3x)$.
Локальный максимум функции $200x-e^x$
Все нули функции $x-e^x\cos(x)$.

Контрольные вопросы:

1. Что такое тестирование программы?
2. Что такое отладка программы? Какие ошибки можно выявить в ходе отладки?
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?

Лабораторная работа №13. Применение отладочных классов в проекте. Отладка проекта.

Цель занятия:

- научиться составлять техническое задание (ТЗ) на разработку программного продукта, применять отладочные классы в проекте.

Краткие теоретические сведения

Теоретические сведения:

Большинство разработчиков предпочитают работать с техническим заданием на разработку программного обеспечения, так как этот документ обычно содержит следующее:

- Полное описание целей и функциональности программного обеспечения;
- Детали того, как программа будет работать с точки зрения скорости, времени отклика, доступности, мобильности, надёжности, скорости восстановления и т.д.;
- Варианты того, как пользователи будут использовать программное обеспечение;
- Определение того, как приложение будет взаимодействовать с оборудованием или другими программами;
- Нефункциональные требования (например: требования к обеспечению эффективности, стандарты качества, или проектные ограничения).

Рассмотрим образец технического задания на разработку программы. 1 Введение

1.1. Наименование программы

1.2. Назначение и область применения программы 2 Требования к программе

2.1. Требования к функциональным характеристикам программы

2.2. Требования к надежности программы

2.2.1. Требования к обеспечению надежного функционирования программы

2.2.2. Время восстановления программы после отказа

2.2.3. Отказы программы из-за некоректных действий оператора 3 Условия эксплуатации программы 3.1. Климатические условия эксплуатации программы

3.2. Требования к квалификации и численности персонала

3.3. Требования к составу и параметрам технических средств

3.4. Требования к информационной совместимости

3.4.1. Требования к информационным структурам и методам решения

3.4.2. Требования к исходным кодам и языкам программирования

3.4.3. Требования к программным средствам, используемым программой

3.4.4. Требования к защите информации и программ

3.5. Специальные требования

4 Требования к программной документации

4.1. Предварительный состав программной документации 5 Технико-экономические показатели

5.1. Экономические преимущества разработки программы 6 Стадии и этапы разработки программы

6.1. Стадии разработки программы

6.2. Этапы разработки программы

6.3. Содержание работ по этапам 7 Порядок контроля и приемки

7.1. Виды испытаний

7.2. Общие требования к приемке работы 1 Введение

1.1. Наименование программы Наименование программы: "Тестовая программа"

1.2. Назначение и область применения Программа предназначена для...

2 Требования к программе

2.1. Требования к функциональным характеристикам

Программа должна обеспечивать возможность выполнения перечисленных ниже функций:

2.2. Требования к надежности

2.2.1 Требования к обеспечению надежного функционирования программы

Надежное (устойчивое) функционирование программы должно быть обеспечено выполнением Заказчиком совокупности организационно-технических мероприятий, перечень которых приведен ниже:

а) организацией бесперебойного питания технических средств; б) использованием лицензионного программного обеспечения;

в) регулярным выполнением рекомендаций Министерства труда и социального развития РФ, изложенных в Постановлении от 23 июля 1998 г. Об утверждении межотраслевых типовых норм времени на работы по сервисному обслуживанию ПЭВМ и оргтехники и сопровождению программных средств»;

г) регулярным выполнением требований ГОСТ 51188-98. Защита информации. Испытания программных средств на наличие компьютерных вирусов

2.2.2. Время восстановления после отказа

Время восстановления после отказа, вызванного сбоем электропитания технических средств (иными внешними факторами), не фатальным сбоем (не крахом) операционной системы, не должно превышать 30-ти минут при условии соблюдения условий эксплуатации технических и программных средств.

Время восстановления после отказа, вызванного неисправностью технических средств, фатальным сбоем (крахом) операционной системы, не должно превышать времени, требуемого на устранение неисправностей технических средств и переустановки программных средств.

2.2.3. Отказы из-за некоректных действий оператора

Отказы программы возможны вследствие некорректных действий оператора (пользователя) при взаимодействии с операционной системой.

Во избежание возникновения отказов программы по указанной выше причине следует обеспечить работу конечного пользователя без предоставления ему административных привилегий

3 Условия эксплуатации

3.1. Климатические условия эксплуатации

Климатические условия эксплутатации, при которых должны обеспечиваться заданные характеристики, должны удовлетворять требованиям, предъявляемым к техническим средствам в части условий их эксплуатации

3.2. Требования к квалификации и численности персонала

Минимальное количество персонала, требуемого для работы программы, должно составлять не менее 2 штатных единиц — системный администратор и конечный пользователь программы — оператор.

Системный администратор должен иметь высшее профильное образование и сертификаты компании- производителя операционной системы. В перечень задач, выполняемых системным администратором, должны входить:

а) задача поддержания работоспособности технических средств;

б) задачи установки (инсталляции) и поддержания работоспособности системных программных средств — операционной системы;

в) задача установки (инсталляции) программы.

г) задача создания резервных копий базы данных.

3.3. Требования к составу и параметрам технических средств

3.3.1. В состав технических средств должен входить IBM-совместимый персональный компьютер (ПЭВМ), выполняющий роль сервера, включающий в себя:

3.3.1.1. процессор Pentium-2.0Hz, не менее;

3.3.1.2. оперативную память объемом, 1Гигабайт, не менее;

3.3.1.3. оперативную память объемом, 1Гигабайт, не менее;

3.3.1.4. операционную систему Windows 2000 Server или Windows 2003;

3.3.1.5. операционную систему Windows 2000 Server или Windows 2003;

3.3.1.6. Microsoft SQL Server 2000

3.4. Требования к информационной и программной совместимости

3.4.1. Требования к информационным структурам и методам решения

База данных работает под управлением Microsoft SQL Server. Используется много поточный доступ к базе данных. Необходимо обеспечить одновременную работу с программой с той же базой данной модулей экспорта внешних данных.

3.4.2. Требования к исходным кодам и языкам программирования Дополнительные требования не предъявляются

3.4.3. Требования к программным средствам, используемым программой

Системные программные средства, используемые программой, должны быть представлены лицензионной локализованной версией операционной системы Windows 2000 Server или Windows 2003 и Microsoft SQL Server 2000

3.4.4. Требования к защите информации и программ Требования к защите информации и программ не предъявляются

3.5. Специальные требования

Специальные требования к данной программе не предьявляются 4 Требования к программной документации

4.1. Предварительный состав программной документации Состав программной документации должен включать в себя:

4.1.1. техническое задание;

4.1.2. программу и методики испытаний;

4.1.3. руководство оператора; 5 Технико-экономические показатели

5.1. Экономические преимущества разработки

Ориентировочная экономическая эффективность не рассчитываются. Аналогия не проводится ввиду уникальности предъявляемых требований к разработке.

6 Стадии и этапы разработки

6.1. Стадии разработки

Разработка должна быть проведена в три стадии: 1 разработка технического задания;

2 рабочее проектирование;

3 внедрение.

6.2. Этапы разработки

На стадии разработки технического задания должен быть выполнен этап разработки, согласования и утверждения настоящего технического задания.

На стадии рабочего проектирования должны быть выполнены перечисленные ниже этапы работ: 1 разработка программы;

2 разработка программной документации; 3 испытания программы.

На стадии внедрения должен быть выполнен этап разработки подготовка и передача программы

На этапе разработки технического задания должны быть выполнены перечисленные ниже работы:

1 постановка задачи;

2 определение и уточнение требований к техническим средствам; 3 определение требований к программе;

4 определение стадий, этапов и сроков разработки программы и документации на 5 согласование и утверждение технического задания.

На этапе разработки программы должна быть выполнена работа по программированию (кодированию) и отладке программы.

На этапе разработки программной документации должна быть выполнена разработка программных документов в соответствии с требованиями к составу документации.

На этапе испытаний программы должны быть выполнены перечисленные ниже виды работ: 1 разработка, согласование и утверждение и методики испытаний;

2 проведение приемо-сдаточных испытаний;

3 корректировка программы и программной документации по результатам испытаний.

На этапе подготовки и передачи программы должна быть выполнена работа по подготовке и передаче программы и программной документации в эксплуатацию на объектах Заказчика.

7 Порядок контроля и приемки

7.1. Виды испытаний

Приемо-сдаточные испытания должны проводиться на объекте Заказчика в оговоренные сроки.

Приемо-сдаточные испытания программы должны проводиться согласно разработанной Исполнителем и согласованной Заказчиком Программы и методик испытаний.

Ход проведения приемо-сдаточных испытаний Заказчик и Исполнитель документируют в Протоколе проведения испытаний

7.2. Общие требования к приемке работы

На основании Протокола проведения испытаний Исполнитель совместно с Заказчиком подписывает Акт приемки-сдачи программы в эксплуатацию.

ВВЕДЕНИЕ

Полное наименование программной разработки: "Программа К", в дальнейшем именуемая как "программа". Краткое название программы – «ПК».

На данный момент аналогичных программных продуктов не существует.

Разрабатываемая программа применяется на любом предприятии, где имеется рабочий персонал. Разработчик данного программного продукта - студент группы А19 Иванов А.В. в дальнейшем именуемый как "разработчик ".

Заказчик программного продукта – ОАО «РТС», в лице директора А.М. Гутенко. 1 ОСНОВАНИЕ ДЛЯ РАЗРАБОТКИ

1.1 Документ, на основании которого ведётся разработка

Работа ведётся на основании задания по дисциплине «Теоретические основы автоматизированного управления»

1.2 Организация, утвердившая этот документ, и дата его утверждения

Задание утверждено и выдано начальником технического отдела ОАО «РТС» Козаковым А.В. Козаков А.В.

1.3 Наименование темы разработки

Наименование темы разработки – «Учёт рабочего времени». 2 НАЗНАЧЕНИЕ РАЗРАБОТКИ

Данная разработка является семестровой работой по дисциплине «Теоретические основы автоматизированного управления»

2.1 Критерии эффективности и качества программы

Социальный фактор. Данная программная разработка очень проста в освоении и рассчитана не только на профессионалов, но и на рядовых пользователей, работающих в ОС Windows. Удобный, интуитивно понятный интерфейс в сочетании с мощной системой вспомогательных рисунков и всплывающих подсказок позволяют работать с программой без предварительной подготовки.

Соответствие текущему состоянию на рынке ПО данного профиля. В отличие от дорогих и сложных программ «ПК» идеально подходит для представителей бизнеса, так как содержит все, что им необходимо, но не перегружена бесполезными и ненужными возможностями. Технология создания программы в визуальных средах программирования делает ее интерфейс универсальным и совместимым с операционными системами Windows 7/8/10.

Экономические факторы. Программа представляет наилучшее соотношение цены и

предоставляемых ей возможностей и несомненно займет свою нишу на рынке дешевых программ. Основными пользователями станут представители бизнеса, которые просто не могут заплатить за дорогие программы фирмы 1С и ей подобных.

2.2 Цели разработки программы

Создание данной программы преследует ряд технико-экономических целей: Создание программного продукта, необходимого для учёта рабочего времени. Создание дешевой альтернативы существующим в настоящее время дорогим программам.

Создание интуитивно понятной программы с удобным и универсальным Windows.

Техническое задание (ТЗ) - исходный документ, который является основанием для разработки и испытания программы или автоматизированной системы. Техническое задание на программу и программное обеспечение разрабатывается в соответствии с требованиями. Основанием для разработки ТЗ чаще всего является договор.

ТЗ на программу разрабатывается, прежде всего, для тех людей, которые в последствии будут разрабатывать программный продукт. Как и любое другое ТЗ на программу должно быть предельно ясно и не содержать двусмысленные формулировки и должно максимально полно описывать все требования и пожелания Заказчика к создаваемой программе, но при этом не стоит забывать, что программисты люди

творческие и освоить 150 листов технического текста им не всегда под силу.

Кому поручить написание ТЗ на программу Хочется акцентировать внимание на часто совершаемой ошибке – поручить написание технического задание на программный продукт программисту, обосновывая тем, что программисту будет проще потом реализовывать собственное техзадание.

Техническое задание на программу должно разрабатываться техническим писателем! Во-первых, помимо знания ГОСТ 19.201-78, необходимо знание и других стандартов (например, ГОСТ 19.106-78 , ГОСТ 19.104 – 78 и др.), не многие программисты знают эти ГОСТы, а ещё меньше согласятся их изучить. Во-вторых, необходимы знания и опыт владения техническим письменным языком (не путать с написанием кода программного обеспечения). В-третьих, только совместно работающая команда (технический писатель, программист, менеджер проекта) смогут вместе разработать полноценное техническое задание на программу и программное обеспечение.

Структура технического задания

Любая работа начинается с задания, а работа технического писателя должна начинаться с технического задания. Осталось только разобраться, что это такое и зачем оно нам нужно. Прочитайте статью Кимберли Чан, чтобы не попасть в такую же ситуацию, как разработчик из уже любимой нами серии комиксов.

Что такое техническое задание на разработку программного обеспечения?

Большинство разработчиков предпочитают работать с техническим заданием на разработку программного обеспечения, так как этот документ обычно содержит следующее:

Полное описание целей и функциональности программного обеспечения;

Детали того, как программа будет работать с точки зрения скорости, времени отклика, доступности, мобильности, надёжности, скорости восстановления и т.д.;

Варианты того, как пользователи будут использовать программное обеспечение;

Определение того, как приложение будет взаимодействовать с оборудованием или другими программами;

Нефункциональные требования (например: требования к обеспечению эффективности, стандарты качества, или проектные ограничения)

Выполнение работы:

Содержание ТЗ должно включать следующие разделы:

Введение

1 Основание для разработки 2 Назначение разработки

3 Требования к программе

4 Требования к программной документации 5 Технико-экономические показатели

6 Стадии и этапы разработки

7 Порядок контроля и приемки

В разделе “Введение” указывают наименование, краткую характеристику области применения программы или программного изделия и объекта, в котором будут использовать программу.

1 В разделе “Основания для разработки” указывается документ, на основании которого ведется разработка (приказ по университету, задание на лаб. работу и т.п.);

организация, утвердившая документ и дата его утверждения; наименование темы разработки.

2 В разделе “Назначение разработки” указывают функциональное эксплуатационное назначение программы.

3 Раздел “Требования к программе” должен содержать следующие подразделы:

3.1. “Требования к функциональным характеристикам” –состав функций, которые будет выполнять программа, организация входных и выходных данных (синтаксис и семантика входных данных, форматы выходных сообщений и соответствующие им ситуации), временные характеристики. В этом подразделе должно быть описано поведение системы с точки зрения соотношения входа и выхода без конкретизации внутренней структуры и реакция программы на непредусмотренные данные на входе.

3.2. “Требования к надежности” –контроль входных и выходных данных, последствия возможных отказов, время восстановления, защита от несанкционированного доступа и др.

3.3. “Условия эксплуатации” – характеристики операционной среды, вид обслуживания, количество и квалификация персонала, затрачиваемое время процессора и каналов связи, число пользователей и др., а также допустимые параметры окружающей

3.4. “Требования к составу и параметрам технических средств” –конфигурация системы, основные характеристики требуемых устройств.

3.5. “Требования к информационной и программной совместимости” – требования к методам решения, языкам программирования, программным средствам, используемых системой, протоколам обмена, к СУБД и операционным системам.

3.6. “Требования к маркировке и упаковке” –варианты и способы упаковки (обычно специальных требований не предъявляется).

3.7. “Требования к транспортированию и хранению” –места хранения, условия и сроки, способы создания и хранения резервных копий (обычно специальных требований не предъявляется).

Отдельные разделы и подразделы по согласованию с заказчиком могут быть опущены.

4 В разделе “Требования к программной документации” – состав документации и специальные требования к ней. Виды программных документов: спецификация, текст программы, описание программы, пояснительная записка, ТЗ, программа и методика испытаний, руководство программиста, руководство системного программиста, руководство оператора, руководство по техническому обслуживанию и т.д.

5 В разделе “Технико-экономические преимущества, предполагаемая экономическая эффективность и годовая потребность, экономические преимущества разработки, предельный объем программы, время реакции программы.

6 В разделе “Стадии и этапы разработки” -перечень стадий, разбивка на этапы, содержание, сроки разработки (дни, недели и т.д.) и исполнители.

7 В разделе “Порядок контроля и приемки” –виды испытаний, требования к приемке работ, способы проверки важнейших характеристик.

Цель испытаний –установление степени соответствия готового продукта и характеристикам технического задания.

Формы представления результатов: программная документация, конструкторская документация на изделие, программное изделие.

В приложении приводят перечень проведенных научных и исследовательских работ, схемы алгоритмов, таблицы, описания и т.д.

1 Разработайте проект в соответствии с вариантом.

2 перейдите в режим отладки. При остановке выполнения программы добавьте в окно просмотра наименования нескольких интересующих Вас переменных.

Требования к отчету: Текст должен быть написан шрифтом Times New Roman,

12 Интервал между строками и абзацами – 1,5. Отступ слева 1,5. Ориентация текста – по ширине страницы. Скриншоты необходимо подписать. Название практической работы, цель работы, ход работы, вывод, ответы на контрольные вопросы, должны быть выделены жирным шрифтом, так же как в методичке.

Практические задания

Задание:

Для выбранного по индивидуальному заданию программного продукта разработать техническое задание в соответствии с ГОСТ 19.201-78, предполагая, что сначала разрабатывается ТЗ, а затем будет написана программа для ТЗ. Отчет по лабораторной работе должен содержать разделы технического задания.

Контрольные вопросы:

1. Назначение технического задания?
2. Кто составляет и утверждает ТЗ?
3. На каком этапе разработки программного изделия составляется ТЗ?
4. Какими документами регламентируется написание ТЗ?

Лабораторная работа №14. Инспекция кода модулей проекта.

Цель занятия:

– получить практические навыки разработки модулей программной системы и интеграции этих модулей.

Краткие 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.

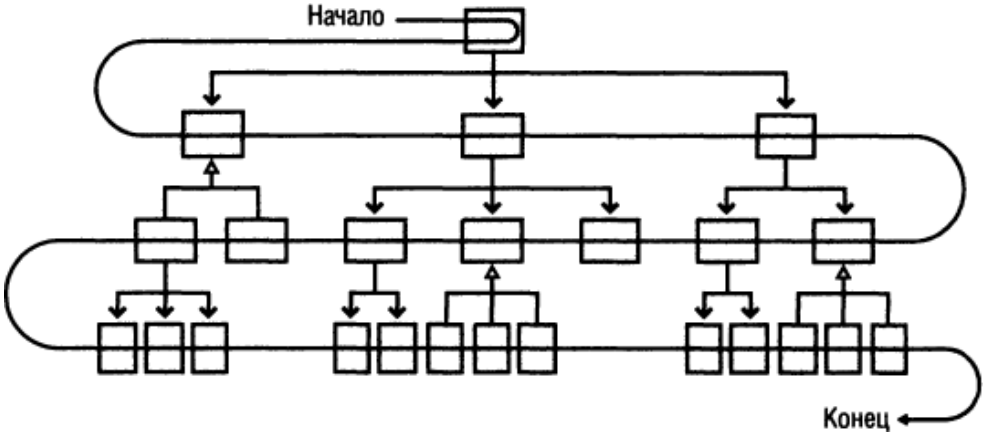


Рис.3 Нисходящая интеграция

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.

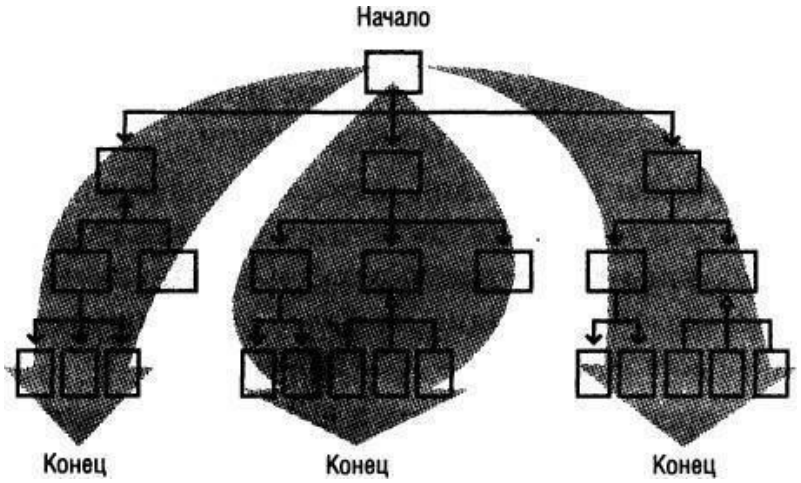


Рис. 4 Вертикальное секционирование

При этом систему реализуют сверху вниз по частям, возможно, по очереди выделяя функциональные области и переходя от одной к другой.

Восходящая интеграция

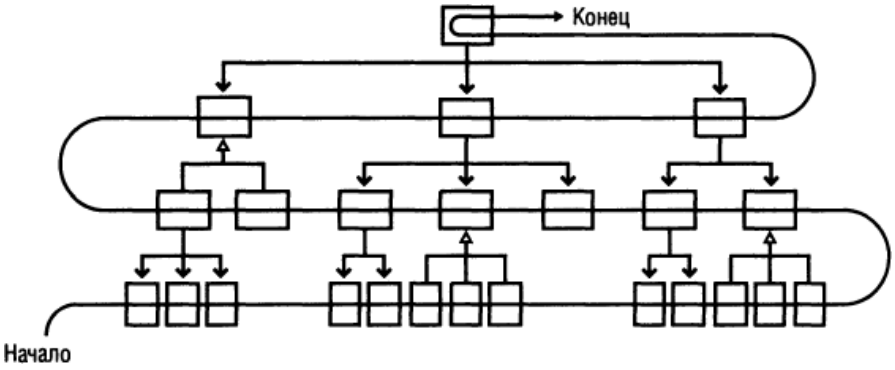


Рис.5 Восходящая интеграция

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.

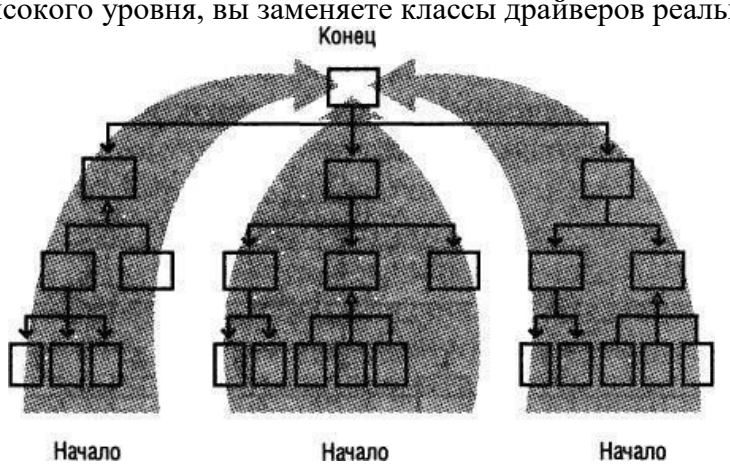


Рис. 6 Гибридный подход при восходящей интеграции

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.

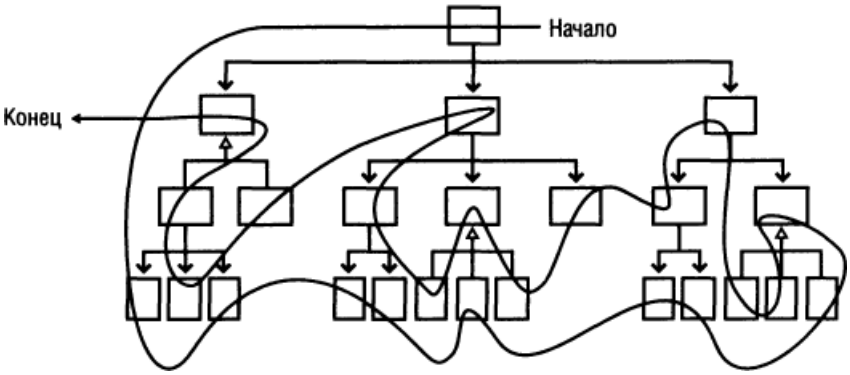


Рис. 7 Сэндвич-интеграция

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.

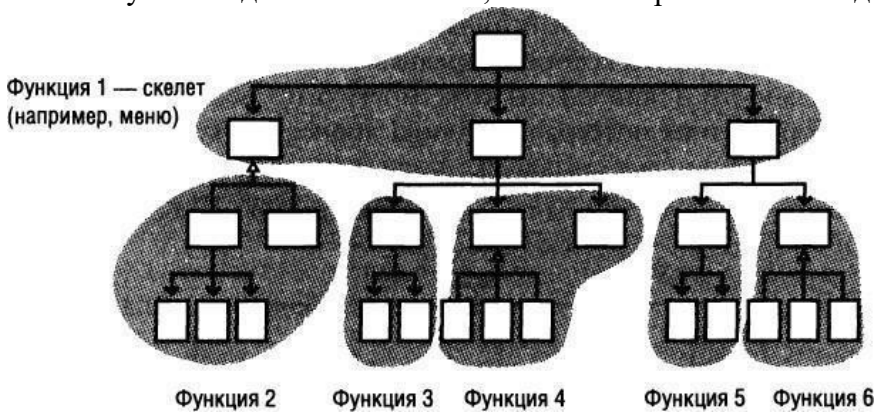


Рис. 8 Функционально-ориентированная интеграция

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.

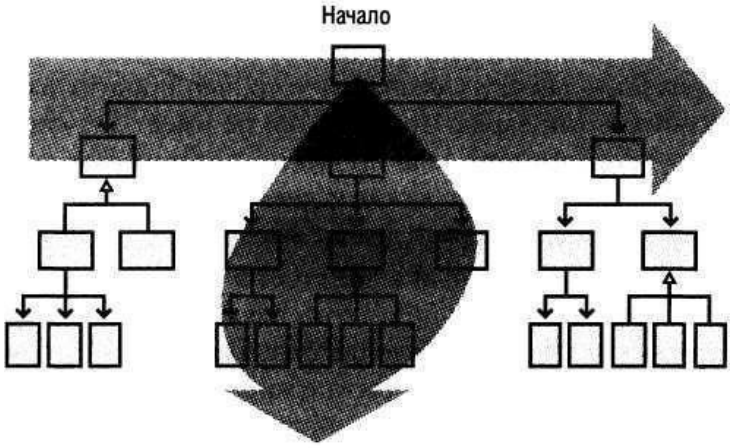


Рис. 9 Т-образная интеграция

Практические задания

Задание.

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.
7. Оформить отчет по лабораторной работе.

Отчет по лабораторной работе должен включать:

1. Внешнюю спецификацию.
2. Алгоритм решения задачи.
3. Текст программы на языке программирования.
4. Набор тестов для отладки модулей программы.
5. Описание процесса интеграции модулей.

Задача. Задан двумерный массив размерности nxm. 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.
3. Эффективность и оптимизация программ.

Лабораторная работа №15. Тестирование интерфейса пользователя средствами инструментальной среды разработки.

Цель занятия:

- получение практических навыков автоматической генерации тестов на основе формального описания.

Краткие теоретические сведения

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 трудоѐмка 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 еѐ 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. Распространѐнным 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 трудоѐмкое занятие, так как число состояний системы очень велико. С ростом функциональности системы спецификация становится всѐ 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.

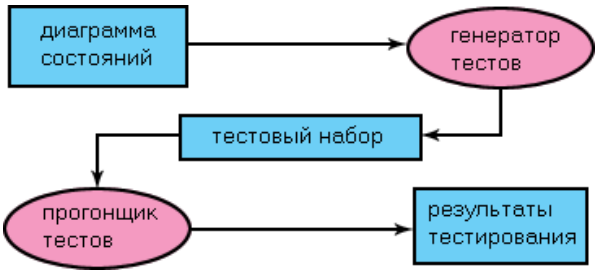


Рис. 2 Общая схема генерации и прогона тестов

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 рѐбрам 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 даѐтся 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.

Практические задания

Задание.

1. Сформировать диаграмму вариантов использования для задачи лабораторной

работы № 1.

2. Сгенерировать набор тестов.

3. Составить отчет по лабораторной работе.

Отчет по лабораторной работе должен включать:

1. Диаграмму вариантов использования. Файл с тестовым набором.

Контрольные вопросы:

1. Дайте определение модели ИС.

2. Что такое гипотетическая модель ИС?

3. Каковы этапы автоматизированного проектирования ИС?

4. По каким параметрам можно произвести анализ систем автоматизированного проектирования ЭИС?

5. Что является основой CASE – технологии?

6. Что такое методология в CASE – технологии?

7. Что такое метод в CASE – технологии?

8. Что такое нотация в CASE – технологии?

9. Что такое средства в CASE – технологии?

10. Какова классификация диаграмм в CASE – технологиях?

11. На какие стадии делится проектирование ИС с использованием CASE – технологии?

12. Перечислите основные факторы эффективности CASE – технологии.

Лабораторная работа №16. Разработка тестовых модулей проекта для тестирования отдельных модулей.

Цель занятия:

– получение практических навыков использования средств автоматизации тестирования.

Краткие теоретические сведения

Для того чтобы продолжать тестирование, когда один тест не прошѐл, 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 прошѐл. В противном случае, если на каком-то этапе выполнения теста, поведение системы не соответствует требованиям, Robot прекращает его выполнение, вынося соответствующий вердикт – тест не прошѐл.

Практические задания

Задание.

1. Выполнить тестовый набор лабораторной работы № 2.
2. Проанализировать отчѐт о прохождении тестов.
3. Составить отчет по лабораторной работе.

Отчет по лабораторной работе должен включать:

1. Отчѐт о прохождении тестов.
2. Анализ отчѐта о прохождении тестов.

Контрольные вопросы:

1. Что такое модульное (Unit) тестирование?
2. Зачем оно нужно?
3. Как его провести?
4. Методы модульного тестирования
5. Разработка через тестирование (TDD)
6. Преимущества модульного тестирования
7. Недостатки модульного тестирования
8. Рекомендации по модульному тестированию

Лабораторная работа №17. Выполнение функционального тестирования.

Цель занятия:

– получить практические навыки разработки тестов на основе внешней спецификации программы.

Краткие теоретические сведения

Программа в случае тестирования с управлением по данным рассматривается как "черный ящик", 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.

Практические задания

Задание.

1. На основе внешней спецификации задачи Практического занятия №5 составить набор тестов на основе подхода «чѐрного ящика».
2. Провести тестирование программы.
3. Оформить отчет по лабораторной работе.

Отчет по лабораторной работе должен включать:

1. Внешнюю спецификацию.
2. Алгоритм решения задачи.
3. Текст программы на языке программирования.
4. Набор тестов на основе подхода «чѐрного ящика» для отладки программы.

Контрольные вопросы:

1. Функциональное (ручное) тестирование.
2. Цели функционального тестирования.

3. Виды функционального тестирования.
4. Как выполнить функциональное тестирование?
5. Мифы о ручном тестировании.
6. Ручное тестирование против автоматизированного тестирования.
7. Инструменты для автоматизации ручного тестирования.
8. Заключение.

Лабораторная работа №18. Тестирование интеграции.

Цель занятия:

- получить практические навыки отладки программ с помощью отладчика среды программирования.

Краткие теоретические сведения

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, приводящих 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.

Практические задания

Задание.

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.
4. Составить отчет по лабораторной работе.

Отчет по лабораторной работе должен включать:

1. Алгоритм решения задачи.
2. Текст программы на языке программирования.
3. Набор тестов для отладки программы.

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 чѐм суть метода эквивалентных разбиений?
10. В чѐм суть метода анализа граничных значений?
11. В чѐм суть метода анализа причинно-следственных связей?

Лабораторная работа №19. Документирование результатов тестирования.

Цель занятия:

– получение практических навыков оформления протоколов тестирования и отладки программы.

Краткие теоретические сведения

Тестирование – процесс выполнения программы на наборе тестов с целью выявления ошибок.

Обеспечить повторяемость процесса тестирования недостаточно – вы должны оценивать и проект, чтобы можно было точно сказать, улучшается он в результате изменений или ухудшается. Вот некоторые категории данных, которые можно собирать с целью оценки проекта:

- административное описание дефекта (дата обнаружения, сотрудник, сообщивший о дефекте, номер сборки программы, дата исправления);
- полное описание проблемы;
- действия, предпринятые для воспроизведения проблемы;
- предложенные способы решения проблемы;
- родственные дефекты;
- тяжесть проблемы (например, критическая проблема, «неприятная» или косметическая);
- источник дефекта: выработка требований, проектирование, кодирование или тестирование;
- вид дефекта кодирования: ошибка занижения или завышения на 1, ошибка присваивания, недопустимый индекс массива, неправильный вызов метода и т. д.;
- классы и методы, измененные при исправлении дефекта;
- число строк, затронутых дефектом;
- время, ушедшее на нахождение дефекта;
- время, ушедшее на исправление дефекта.

Собирая эти данные, вы сможете подсчитывать некоторые показатели, позволяющие сделать вывод об изменении качества проекта:

- число дефектов в каждом классе; все числа целесообразно отсортировать в порядке от худшего класса к лучшему и, возможно, нормализовать по размеру класса;
- число дефектов в каждом методе, все числа целесообразно отсортировать в порядке от худшего метода к лучшему и, возможно, нормализовать по размеру метода;
- среднее время тестирования в расчете на один обнаруженный дефект;
- среднее число обнаруженных дефектов в расчете на один тест;
- среднее время программирования в расчете на один исправленный дефект;
- процент кода, покрытого тестами;
- число дефектов, относящихся к каждой категории тяжести.

Кроме протоколов тестирования уровня проекта, вы можете хранить и личные протоколы тестирования. Можете включать в них контрольные списки ошибок, которые вы допускаете чаще всего, и указывать время, затрачиваемое вами на написание кода, его тестирование и исправление ошибок.

Практические задания

Задание.

1. Выполнить тестирование программы, разработанной в предыдущей работе.
2. Оформить протоколы тестирования.
3. Оформить отчет по лабораторной работе.

Отчет по лабораторной работе должен включать:

1. Внешнюю спецификацию.
2. Набор тестов.

3. Текст программы на языке программирования.
4. Протоколы тестирования программы.

Контрольные вопросы:

1. Назовите формальные методы проверки правильности программ.
2. Какие процессы проверки зафиксированы в стандарте?
3. Какие функции у процесса верификации программ?
4. Сравните задачи процессов верификации и валидации программ.
5. В чем отличие верификации и валидации?
6. Назовите методы тестирования.
7. Объясните значения терминов "черный ящик", "белый ящик".

МДК.02.03 МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ

Лабораторная работа №20. Построение простейших математических моделей. Построение простейших статистических моделей. Решение простейших однокритериальных задач.

Цель занятия:

- научиться выполнять построение простейших математических моделей, статических моделей; научиться решать простейшие однокритериальные задачи

Краткие теоретические сведения

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, заменяющий оригинал, наделенный основными характеристиками (чертами) оригинала и предназначенный для проведения некоторых действий над ним с целью получения новых сведений об оригинале.

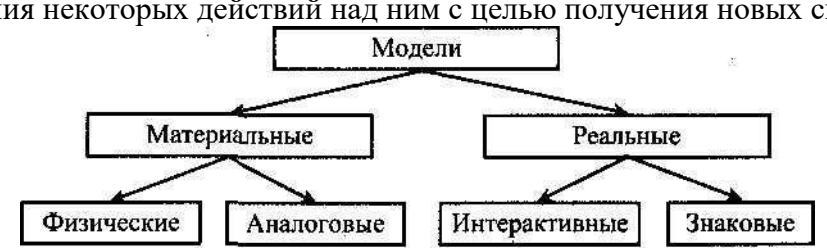


Рис. 1. Классификация моделей

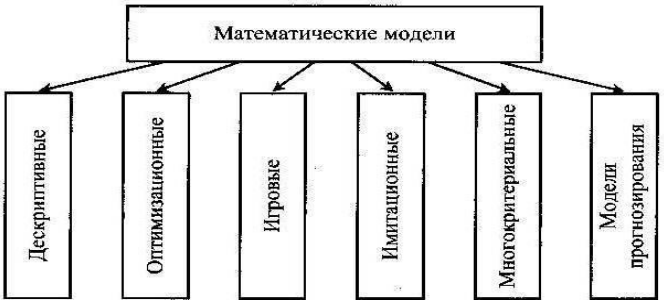


Рис. 2. Классификация математических моделей

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 функ- циональной 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. 3).

Входной параметр 1		Выходной параметр 1
Входной параметр 2		Выходной параметр *2*
Входной параметр 3	Модель системы	Выходной параметр 3
Входной параметр *n*-1	(объекта или процесса)	Выходной параметр *m* - 1
Входной параметр *n*		Выходной параметр *m*

Рис. 3. 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, рас- пределения производственных, финансовых и людских ресурсов.

5. Принятие решений и элементы планирования.

В этом случае и качестве целевой функции выступает набор ста- тистических данных или некоторые данные прогноза. Решением задачи являются рекомендации о спосо-

бах 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.

Решение системы неравенств с двумя переменными графическим методом включает следующие этапы.

1. На плоскости X1OX2 строят прямые, уравнения которых получаются в результате замены в ограничениях знаков неравенств на знаки точных равенств.
2. Находят полуплоскости, определяемые каждым из неравенств.
3. Строят многоугольник решений.

Пример:

Решить систему неравенств графическим способом.

$$P=\left\{\begin{array}{ll} X_1+2\cdot X_2\le 6 & (а)\\ 2\cdot X_1+X_2\le 8 & (б)\\ X_1+0.8\cdot X_2\le 5 & (в)\\ -X_1+X_2\le 1 & (г)\\ X_2\le 2 & (д)\\ X_1\ge 0,\ X_2\ge 0 & (е) \end{array}\right.$$

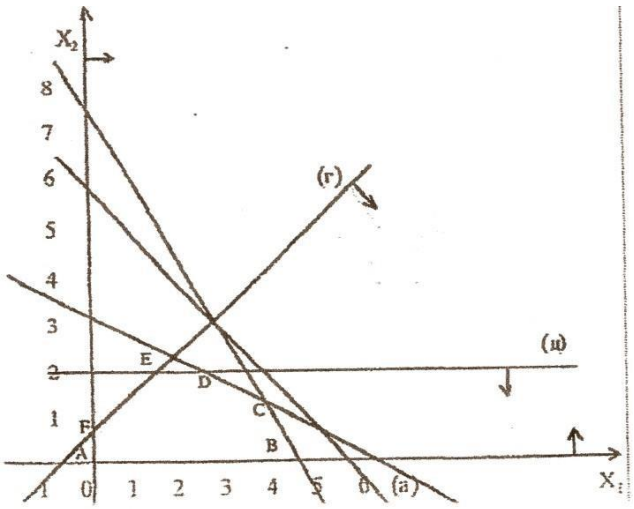
Решение:

Шаг 1. Строим область допустимых решений - область Р, т.е. геометрическое место точек, в котором одновременно удовлетворяются все ограничения ЗЛП. Каждое из неравенств (а)-(д) системы ограничений задачи геометрически определяет полуплоскость соответственно с граничными прямыми:

$$\begin{array}{ll} X_1+2\cdot X_2=6 & (а)\\ 2\cdot X_1+X_2=8 & (б)\\ X_1+0.8\cdot X_2=5 & (в)\\ -X_1+X_2=1 & (г)\\ X_2=2 & (д) \end{array}$$

Условия неотрицательности переменных (е) ограничивают область допустимых решений первым квадратом. Области, в которых выполняются соответствующие ограничения в виде неравенств, указываются стрелками, направленными в сторону допустимых значений переменных

Решение системы – многоугольник ABCDEF.



Практические задания

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 x1 – единица готовой продукции вида А, x2 - единица готовой продукции вида В,

Цель фабрики получить максимальную прибыль от реализации всей продукции видов А и В, тогда:

$$F = 7 \cdot x_1 + 3 \cdot x_2 \rightarrow \max$$

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:

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.

Построение математической модели – неизвестными в этой задачи является объем перевозок . Пусть x_{ij} - объем перевозок сі-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 сі-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:

$$F = \sum_{i=1}^{4} \sum_{j=1}^{5} c_{ij} x_{ij} \to \min,$$

Найти минимум функционала:

При ограничениях:

$$\begin{cases} \sum_{i=1}^{4} x_{ij} = 100, \\ \sum_{i=1}^{4} x_{ij} = 125, \\ \sum_{i=1}^{4} x_{ij} = 325, \quad \sum_{j=1}^{5} x_{ij} = 200, \\ \sum_{i=1}^{4} x_{ij} = 250, \quad \sum_{j=1}^{5} x_{ij} = 450, \\ \sum_{i=1}^{4} x_{ij} = 100, \quad \sum_{j=1}^{5} x_{ij} = 250, \end{cases} \qquad x_{ij} \geq 0, i \in [1,3], j \in [1,5].$$

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 а1 кг первого сорта, а2 кг второго сорта и а3 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.*	$D = \begin{bmatrix} 4 & 8 & 13 & 2 & 7 \\ 9 & 4 & 11 & 9 & 17 \\ 3 & 16 & 10 & 1 & 4 \end{bmatrix}$

Задача 3.

1. Решить графически систему неравенств: а) x1 + x2 ≤ 5
3x1 - x2 ≤ 3 x1 ≥0, x2 ≥0

б) x1 + x2 ≤ 4 6x1 + 2x2 ≥ 6 x1 + 5x2 ≥ 5 x1 ≥0, x2 ≥0

2. Составить математическую модель задачи и найти решение системы ограничений:

Чулочно-носочная фирма производит и продает два вида товаров: мужские носки и женские чулки. Фирма получает прибыль в размере 10 руб. от производства и продажи одной пары чулок и в размере 4 руб. от производства и продажи одной пары носков. Производство каждого изделия осуществляется на трех участках. Затраты труда (в часах) на производство одной пары указаны в следующей таблице для каждого участка:

Участок производства Чулки
Носки 1
0,02
0,01
2
0,03
0,01
3

0,03
0,02

Руководство рассчитало, что в следующем месяце фирма ежедневно будет располагать следующими ресурсами рабочего времени на каждом из участков: 60 ч на участке 1; 70 ч на участке 2 и 100 ч на участке 3. Сколько пар носков и чулок следует производить ежедневно, если фирма хочет максимизировать прибыль?

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.*	$D = \begin{bmatrix} 22 & 14 & 16 & 28 & 30 \\ 19 & 17 & 26 & 36 & 36 \\ 37 & 30 & 31 & 39 & 41 \end{bmatrix}$

Задача 3.

1. Решить графически систему неравенств:

а) x1 + x2 ≤ 5 3x1 - x2 ≤ 3 x1 ≥0, x2 ≥0

б) x1 - x2 ≤ 3 x1 + x2 ≤ 9
-x1 + x2 ≥ 3 x1 + x2 ≥ 3/2 x1 ≥0, x2 ≥0

2. Составить математическую модель задачи и найти решение системы ограничений:

После предпринятой рекламной компании фирма «Отдых» испытывает рост спроса на два типа мангалов для приготовления шашлыков на открытом воздухе – газовые и угольные. Фирма заключила контракт на ежемесячную поставку в магазины 300 угольных и 300

газовых мангалов. Производство мангалов ограничивается мощностью следующих трех участков: производства деталей, сборки и упаковки. В таблице показано, сколько человеко-часов затрачивается на каждом участке на каждую единицу продукции, а также приведен допустимый ежемесячный объем трудозатрат:

Участок
Трудозатраты на производство одного мангала, ч Фонд времени, человекочасы
угольного газового Производство
5
8
2600
Сборка 0,8
1,2
400
Упаковка 0,5
0,5
200

Контрольные вопросы:

1. Что такое модель?
2. Приведите классификацию моделей.
3. Какие вы знаете виды математических моделей?
4. Дайте определение целевой функции.
5. Что такое область допустимых решений?
6. Что называется, допустимым решением, оптимальным решением?
7. Какие способы реализации математических моделей вы знаете?

Лабораторная работа №21. Задача Коши для уравнения теплопроводности. Сведение произвольной задачи линейного программирования к основной задаче линейного программирования.

Цели занятия:

- закрепить практические навыки по построению простейших физических моделей;
- научиться сводить произвольную задачу линейного программирования к основной задаче линейного программирования;
- решать задачи линейного программирования симплекс-методом.

Краткие теоретические сведения

Уравнение теплопроводности — дифференциальное уравнение в частных производных второго порядка, которое описывает распределение температуры в заданной области пространства и ее изменение во времени.

Для получения однозначного решения системы дифференциальных уравнений должны быть заданы дополнительные условия. Их должно быть задано столько, каков порядок решаемой системы. Если все эти условия задаются в одной точке, т.е. при одном значении $X=X_0$, то такая задача называется задачей Коши. Эти дополнительные условия называются начальными условиями, а X_0 - называется начальной точкой.

Покажем применение метода Рунге-Кутта четвертого порядка для решения задачи Коши системы двух уравнений вида

$dY/dX = f_1(X,Y,Z)$

$dZ/dX = f_2(X,Y,Z)$.

Начальные условия зададим в виде $Y(X_0)=Y_0$ и $Z(X_0)=Z_0$.

Запишем формулы Рунге-Кутта для приближенного решения этой системы

$Y_{i+1}=Y_i+h\Phi_1= (K_1+2K_2+2K_3+K_4)/6$,

$Z_{i+1} = Z_i+h\Phi_2=(L_1+2L_2+2L_3+L_4)/6$,

где

$K_1= f_1(X_i,Y_i,Z_i)$ и $L_1= f_2(X_i,Y_i,Z_i)$,

$K_2= f_1(X_i +h/2, Y_i+hK_1/2, Z_i+hL_1/2)$ и $L_2= f_2(X_i +h/2, Y_i+hK_1/2, Z_i+hL_1/2)$,

$K_3= f_1(X_i +h/2, Y_i+hK_2/2, Z_i+hL_2/2)$ и $L_3= f_2(X_i +h/2, Y_i+hK_2/2, Z_i+hL_2/2)$,

$K_4= f_1(X_i +h,Y_i+hK_3, Z_i+hL_3)$ и $L_4= f_2(X_i +h,Y_i+hK_3, Zi+hL_3)$.

В этих формулах i = 0,1,2,...,n-1. Для оценки погрешности используется правило двойного пересчета точно так же, как и при решении одного уравнения.

К решению подобной системы уравнений можно свести решение задачи Коши для уравнения второго порядка

$d^2Y/dX^2 = f(X,Y,dY/dX)$

с начальными условиями $Y(X_0)=Y_0$ и $Y'(X_0)=Z_0$.

Введем новую переменную $Z(X) = dY/dX$. Тогда исследуемое уравнение заменяется следующей системой из двух уравнений

$dZ/dX= f(X,Y,Z)$

$dY/dX=Z$,

с начальными условиями $Y(X_0)=Y_0$ и $Z(X_0)=Z_0$.

Пример 1.

Применяя метод Рунге-Кутта, вычислить на отрезке [1;1,5] таблицу значений решения уравнения

$Y'' +Y'/X+Y=0$

с начальными условиями $Y(1)=0,77$ и $Y'(1)=-0,5$, выбрав шаг 0,1 с погрешностью 0,0001.

С помощью подстановки новой переменной Z, перейдем к решению системы уравнений

Y’=Z

Z’=-Z/X-Y,

с начальными условиями Y(1)=0,77 и Z(1)=-0,5.

Откроем новый рабочий лист EXCEL и выделим в нем столбцы A, B и C под переменные X, Y и Z. В последующие столбцы будем вычислять последовательно значения коэффициентов K_i, $Ф_1$, L_i, $Ф_2$. Пусть значения X помещаются в блок A3:A8. Тогда в ячейки B3 и C3 занесем начальные значения Y(1) и Z(1). Формулы в остальных ячейках строки 3 и 4 приведены в таблице

Ячейка	Формула
D3	=C3
E3	=-C3/A3-B3
F3	=C3+0,1*E3/2
G3	=-F3/(A3+0,1/2)-(B3+0,1*D3/2)
H3	=C3+0,1*G3/2
I3	=-H3/(A3+0,1/2)-(B3+0,1*F3/2)
J3	=C3+0,1*I3
K3	=-J3/(A3+0,1)-(B3+0,1*H3)
L3	=D3+2*F3+2*H3+J3
M3	= E3+2*G3+2*I3+K3
B4	=B3+0,1*L3/6
C4	=C3+0,1*M3/6

Формулы в остальных строках решения получаются путем копирования.

Результаты решения задачи с шагом 0,1 приведены в таблице.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Решение системы дифференциальных уравнений												
2	x	y	z	K1	L1	K2	L2	K3	L3	K4	L4	Ф1	Ф2
3	1	0,77	-0,5	-0,5	-0,27	-0,5135	-0,255952	-0,512798	-0,255946	-0,525595	-0,240907	-3,07819	-1,534704
4	1,1	0,7187	-0,525578	-0,525578	-0,240898	-0,537623	-0,224919	-0,536824	-0,225012	-0,54808	-0,208281	-3,222553	-1,349042
5	1,2	0,66499	-0,548062	-0,548062	-0,208269	-0,558476	-0,190804	-0,557603	-0,190982	-0,567161	-0,17295	-3,34738	-1,14479
6	1,3	0,6092	-0,567142	-0,567142	-0,172935	-0,575789	-0,15433	-0,574859	-0,154587	-0,582601	-0,135569	-3,451039	-0,926338
7	1,4	0,55168	-0,582581	-0,582581	-0,135551	-0,589359	-0,116097	-0,588386	-0,116429	-0,594224	-0,096693	-3,532295	-0,697297
8	1,5	0,49281	-0,594203	-0,594203	-0,096674	-0,599037	-0,076624	-0,598034	-0,077029	-0,601906	-0,056815	-3,59025	-0,460793

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 переменным 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:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \{\leq, \geq, =\} b_1, \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \{\leq, \geq, =\} b_2, \\ \ldots\ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \{\leq, \geq, =\} b_m. \end{cases} \quad (1)$$

$$x_1 \geq 0,\ x_2 \geq 0,\ \ldots,\ x_n \geq 0 \quad (2)$$

$$F = c_1x_1 + c_2x_2 + \ldots + c_nx_n \rightarrow \max(\min) \quad (3)$$

Коэффициенты $a_{i,j}$, 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:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1, \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2, \\ \ldots\ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n = b_m. \end{cases} \quad (4)$$

$$x_1 \geq 0,\ x_2 \geq 0,\ \ldots,\ x_n \geq 0 \quad (5)$$

$$F = c_1x_1 + c_2x_2 + \ldots + c_nx_n \rightarrow \max(\min) \quad (6)$$

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 «-»

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \leq b_1 \quad (7)$$

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:

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n + x_{n+1} = b_1 \quad (8)$$

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

$$\begin{aligned} x_k &= x_k - x_l, \\ x_k &\geq 0,\ x_l \geq 0 \end{aligned} \quad , l \text{ - свободный индекс}$$

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.

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \le b_1, \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \le b_2, \\ \ldots\ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \le b_m. \end{cases} \qquad (9)$$

$$F = c_1x_1 + c_2x_2 + \ldots + c_nx_n \to \max(\min)$$

$$x_1 \ge 0,\ x_2 \ge 0,\ \ldots,\ x_n \ge 0$$

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:

$$a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n = b_i \Leftrightarrow$$

$$\Leftrightarrow \begin{cases} a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n \le b_i, \\ -a_{i1}x_1 - a_{i2}x_2 - \ldots - a_{in} \le b_i, \end{cases}$$

Существует и другие способы преобразования системы равенств в систему неравенств, т.е. ***всякую задачу линейного программирования можно сформулировать в стандартной форме.***

Практические задания

Задание 1.

1. Изучите краткие теоретические основания выполнения работы.
2. Выполните решение примера 1. Сравните полученные значения.
3. Для оценки погрешности полученного решения проведите решение вновь с шагом, равным половине начального, т.е. 0,05.
4. Постройте графики по результатам, полученных в пунктах 2 и3.

Задание 2.

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:

$$\begin{cases} 2x_1 - x_2 + 3x_3 + x_4 = 5 \\ x_1 + 2x_3 = 8 \\ -x_1 - 2x_2 - x_5 = 1 \end{cases}$$
$$x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, x_4 \geq 0, x_5 \geq 0$$

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:

$$\begin{cases} 2x_1 - x_2 + 3x_3 \leq 5 \\ x_1 + 2x_3 \leq 8 \\ -x_1 - 2x_3 \leq 8 \\ -x_1 - 2x_2 \geq 1 \end{cases}$$
$$x_1 \geq 0, x_2 \geq 0, x_3 \geq 0$$
$$F = -x_1 + x_2 - 3x_3 \to \max$$

Задания для самостоятельной работы

1 вариант.

Задача 1. а) Привести к канонической форме задачу линейного программирования.

$$\begin{cases} x_1 - 2x_2 + x_3 \geq 4, \\ x_1 + x_2 - 3x_3 \leq 9 \\ x_1 + 3x_2 + 2x_3 = 10 \end{cases}$$

$$x_1 \geq 0, x_2 \geq 0, x_3 \geq 0$$

$$F = 2x_1 + x_2 - x_3 \to \max$$

б) Напишите задачу в стандартной форме.

2 вариант.

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.

3 вариант.

Задача 1. а) Привести к канонической форме задачу линейного программирования.

$$\begin{cases} 2x_1 - x_2 + 6x_3 \le 12, \\ 3x_1 + 5x_2 - 12x_3 = 14 \\ -3x_1 + 6x_2 + 4x_3 \le 18 \end{cases}$$

$$x_1 \ge 0, x_2 \ge 0, x_3 \ge 0$$

$$F = -2x_1 - x_2 + x_3 \to \min$$

б) Напишите задачу в стандартной форме.

4 вариант.

Задача 1. а) Привести к канонической форме задачу линейного программирования.

$$\begin{cases} -x_1 + x_2 + x_3 \ge 4, \\ 2x_1 - x_2 + x_3 \le 16 \\ 3x_1 + x_2 + x_3 \ge 18 \end{cases}$$

$$x_1 \ge 0, x_2 \ge 0, x_3 \ge 0$$

$$F = 2x_1 - 5x_2 - 3x_3 \to \min$$

б) Напишите задачу в стандартной форме.

5 вариант.

Задача 1. а) Привести к канонической форме задачу линейного программирования.

$$\begin{cases} -4x_1 + 3x_2 + 8x_3 \ge 15, \\ 2x_1 + 5x_2 - 7x_3 \le 12 \\ 3x_1 - 2x_2 + 10x_3 \le 17 \end{cases}$$

$$x_1 \ge 0, x_2 \ge 0, x_3 \ge 0$$

$$F = -3x_1 - 5x_2 - 6x_3 \to \min$$

б) Напишите задачу в стандартной форме.

Контрольные вопросы:

1. Что такое модель?
2. Что называется уравнением теплопроводности?
3. Какая задача называется задачей Коши?
4. Какие задачи можно отнести к задачам линейного программирования?
5. Какова основная идея линейного программирования?
6. Что образует систем ограничений?
7. Что называется допустимым планом?
8. Что называется целевой функцией?
9. Как записывается общая форма задачи линейного программирования?
10. Как строится каноническая форма ЗЛП?
11. Как перевести ЗЛП в стандартную форму?

Лабораторная работа №22. Решение задач линейного программирования симплекс-методом. Нахождение начального решения транспортной задачи. Решение транспортной задачи методом потенциалов.

Цель занятия:

- научиться применять симплекс-метод при решении задач линейного программирования.

Краткие теоретические сведения:

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:

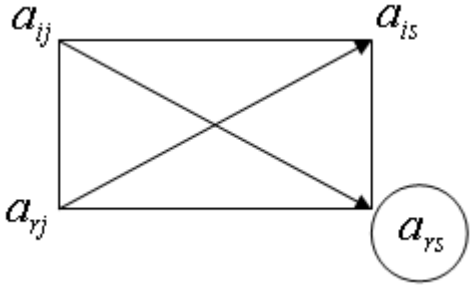
разрешающий элемент $b_{rs} = \frac{1}{a_{rs}}$,

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:

$$b_{ij} = \frac{a_{ij} \cdot a_{rs} - a_{rj} \cdot a_{is}}{a_{rs}}$$



Вновь полученный план проверяем на оптимальность.

Практические задания

Задание. 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: $X_0 = (0;\ \ 0;\ \ 100;\ \ 60;\ \ 180)$.

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	$\approx 37{,}14$
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.

Задания для самостоятельной работы

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.

1 вариант.

a1= 19, a2= 16, a3= 19, b1= 31, b2= 9, b3= 1, c1= 1121, c2= 706, c3= 1066, α=16, β=19.

2 вариант.

a1= 14, a2= 15, a3= 20, b1= 40, b2= 27, b3= 4, c1= 1200, c2= 993, c3= 1097, α=5, β=13.

3 вариант.

a1= 14, a2= 15, a3= 20, b1= 40, b2= 27, b3= 4, c1= 1200, c2= 993, c3= 1097, α=5, β=13.

4 вариант.

a1= 9, a2= 15, a3= 15, b1= 27, b2= 15, b3= 3, c1= 606, c2= 802, c3= 840, α=11, β=6.

5 вариант.

a1= 13, a2= 13, a3= 11, b1= 23, b2= 11, b3= 1, c1= 608, c2= 614, c3= 575, α=5, β=7.

Контрольные вопросы:

1. Как строится каноническая форма ЗЛП?
2. Как перевести ЗЛП в стандартную форму?
3. Какова идея симплекс-метода?
4. В чем суть условия оптимальности плана?
5. Из каких пунктов состоит алгоритм решения ЗЛП симплекс-методом?
6. Что такое симплекс-отношение?

Лабораторная работа №23. Применение метода стрельбы для решения линейной краевой задачи. Задача о распределении средств между предприятиями. Задача о замене оборудования. Нахождение кратчайших путей в графе. Решение задачи о максимальном потоке.

Цели занятия:

- научиться применять симплекс-метод при решении задач линейного программирования;
- приобрести навык применения метода стрельбы для решения линейной краевой задачи;
- решить простейшие задачи методом динамического программирования;
- изучить процесс построения модели решения задач замены оборудования, а также получить практические навыки решения задач замены оборудования методом динамического программирования (ДП).

Краткие теоретические сведения

ПРИМЕНЕНИЕ МЕТОДА СТРЕЛЬБЫ ДЛЯ РЕШЕНИЯ ЛИНЕЙНОЙ КРАЕВОЙ ЗАДАЧИ.

Если при решении дифференциального уравнения высокого порядка дополнительные условия, определяющие однозначное решение, заданы при разных значениях независимой переменной, обычно в двух точках, являющихся границами области решения уравнения, то такая задача называется краевой. При этом сами дополнительные условия называются граничными или краевыми.

Рассмотрим решение краевой задачи для уравнения второго порядка

d2Y/dX2 = f(X,Y,dY/dX)

с граничными условиями Y(X0)=Y0 и Y(Xn)=Yn.

Сущность метода стрельбы заключается в сведении краевой задачи к многократному решению задачи Коши для того же уравнения с подбором недостающего начального условия Y’(X0)=Z0 так, чтобы решение задачи Коши в точке Xn совпадало бы с заданным граничным условием Yn с заданной точностью E.

Алгоритм метода стрельбы таков

1) выбирается любое значение Z_0,
2) исходное уравнение второго порядка приводится к системе из двух уравнений первого порядка введением дополнительных переменных,
3) полученная система решается численным методом, например, методом Рунге-Кутта при некотором начальном значении шага h и запоминается полученное решение в точке Xn,
4) шаг h уменьшается в два раза и вновь находится решение в точке X_n,
5) если новое решение отличается от старого меньше, чем на заданную точность E, то переходят к следующему этапу; иначе - возвращаются к предыдущему этапу и снова уменьшают шаг h в два раза и т.д. до тех пор, пока не будет достигнута заданная точность,
6) при выбранном значении шага, обеспечивающего необходимую точность решения, многократно решают задачу Коши, подбирая начальное условие Y’(X0)=Z0 так, чтобы получаемое в точке Y(Xn) решение отличалось от заданного краевого условия Yn меньше, чем на заданную величину E.

Краевая задача: метод прогонки.

Рассмотрим метод прогонки на примере решения уравнения второго порядка вида

Y”+p(X)Y’+q(X)Y=f(X)

на отрезке [X0, Xn] с заданными граничными условиями Y(X0)=Y0=A и Y(Xn)=Yn=B. Смысл решения заключается в расчете таблицы приближенных значений искомой функции Y(X) в узлах Xi= X0+ih, где h=(Xn - X0)/n и i=1,2,...,n-1.

Заменим приближенно в каждом внутреннем узле производные Y" и Y' конечными центрально-разностными отношениями

Yi"=(Yi+1 -2Yi + Yi-1)/h2 и Yi'=(Yi+1 - Yi-1)/(2h).

Обозначим pi=p(Xi), qi=q(Xi), fi=f(Xi).

Используя эти формулы, приближенно заменим исходное дифференциальное уравнение второго порядка на систему линейных алгебраических уравнений

(Yi+1 -2Yi + Yi-1)/h2 + pi (Yi+1 - Yi-1)/(2h) + qiYi = fi,

где i=1,2,...,n-1 , Y0=A и Yn=B. Решив эту систему, получим таблицу приближенных значений искомой функции. При большом n непосредственное решение такой системы, например, методом Гаусса становится громоздким. Учитывая специфический вид полученной системы алгебраических уравнений, а именно- что матрица коэффициентов ее трехдиагональна, применим специальный метод решения, называемый методом прогонки.

Запишем систему в виде

ai Yi-1+bi Yi +ci Yi+1 =di ,

где ai =1-hpi /2, bi = h2qi -2, ci =1+hpi /2, di = h2fi.

Введем дополнительные переменные

vi = - ci /(bi + ai vi-1) и ui = (di - ai ui-1)/(bi + ai vi-1).

Чтобы сделать схему вычислений однородной, положим a0=0 и cn=0. Тогда vn=0 и Yn=un. Кроме того, v0 = - c0/b0 и u0 = d0/b0.

Тогда решение системы определяется формулой

Yi = ui + vi Yi+1 .

Пример 2.

Применяя метод Рунге-Кутта, вычислить на отрезке [1;1,5] таблицу значений решения уравнения Y" +Y'/X+Y=0. С граничными условиями Y(1)=0,77 и Y(1,5)=0,49281, выбрав шаг 0,1 с погрешностью 0,001.

Как видно из условий задачи, h = 0,1, q=1, f=0, d=0, p=1/X для всех X.

Откроем новый рабочий лист и выделим блок A4:A9 под значения X от 1 до 1,5. В блок B5:B8 занесем значения 1/X, отведем столбцы C,D,E для текущих значений ai, bi, ci, столбцы F и G - для значений vi и ui , а столбец H - для значений Y.

Занесем числа: в ячейку F4 - ноль, в ячейки G4 и H4 - число 0,77. Соответственно, в ячейку F9 - тоже ноль, в ячейки G9 и H9 - число 0,49281.

Формулы ячеек в строке 5 представлены в таблице.

ячейка	формула
B5	=1/A5
C5	=1-B5*$B$2/2
D5	=$B$2^2-2
E5	=1+B5*$B$2/2
F5	=-E5/(D5+C5*F4)
G5	=-C5*G4/(D5+C5*F4)
H5	=G5+F5*H6

Формулы в строках 6,7,8 должны быть скопированы из строки 5. Результаты вычислений приведены ниже.

	A	B	C	D	E	F	G	H
1	Метод прогонки							
2	h=	0,1	q=	1	f=	0	d=	0
3	x	p=1/x	a	b	c	v	u	y
4	1					0	0,77	0,77
5	1,1	0,909091	0,954545	-1,99	1,045455	0,525354	0,369347	0,718712
6	1,2	0,833333	0,958333	-1,99	1,041667	0,700734	0,238109	0,665009
7	1,3	0,769231	0,961538	-1,99	1,038462	0,788974	0,173946	0,609219
8	1,4	0,714286	0,964286	-1,99	1,035714	0,84259	0,136457	0,551694
9	1,5					0	0,49281	0,49281

ЗАДАЧА О РАСПРЕДЕЛЕНИИ СРЕДСТВ МЕЖДУ ПРЕДПРИЯТИЯМИ.

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.

ЗАДАЧА О ЗАМЕНЕ ОБОРУДОВАНИЯ.

Старое оборудование имеет физический и моральный износ, в результате чего растут производственные затраты по выпуску продукции на старом оборудовании, увеличиваются затраты на его ремонт и обслуживание, а вместе с тем снижаются его производительность и ликвидная стоимость.

Наступает момент, когда старое оборудование более выгодно продать (заменить новым), чем эксплуатировать ценой больших затрат.

Оптимальная стратегия замены оборудования состоит в определении оптимальных сроков замены. Критериями оптимальности при определении сроков замены могут служить либо прибыль от эксплуатации, которую следует максимизировать, либо суммарные затраты на эксплуатацию в течение рассматриваемого промежутка времени, которые подлежат минимизации.

Условимся считать, что решение о замене оборудования принимается периодически в начале каждого промежутка, на который разбит плановый период.

Основными функциональными характеристиками оборудования являются:

- t -возраст оборудования (t =0, 1, 2, 3,..., n), где $t=0$ - использование нового оборудования, t = 1 -использование оборудования возраста одного периода и т.д.;
- $f(t)$ -стоимость продукции (для автомобиля -выручка за транспортные услуги), произведенной за период на оборудовании возраста t;
- $r(t)$ -эксплуатационные затраты за период на оборудования возраста t;
- $\varphi(t)$ -остаточная стоимость оборудования возраста t;
- P –цена нового оборудования;
- t_0 -начальный возраст оборудования;
- n - продолжительность планового периода.

Схема возможных состояний оборудования может выглядеть так (рисунок 1), где U^c -сохранность и продолжительность использования оборудования; $U^з$ - заменяемость оборудования новым; S_k^t -состояние оборудования, соответствующее возрасту t.

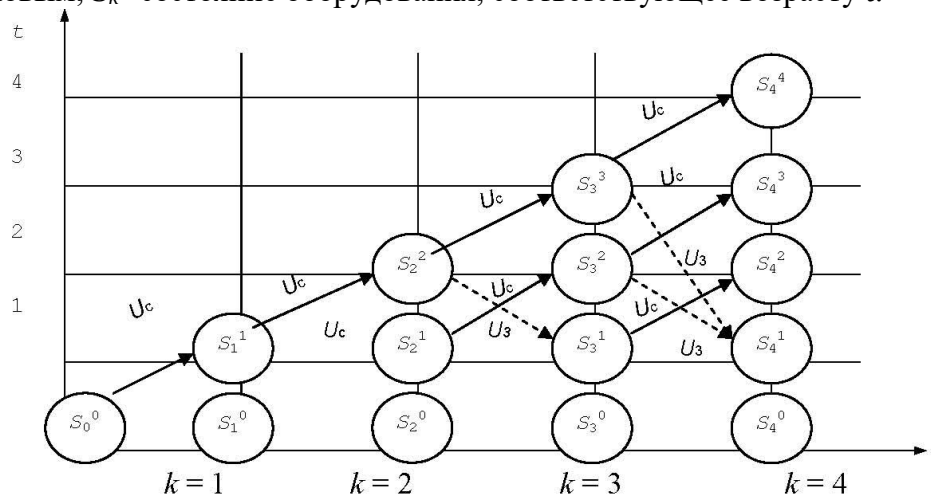


Рисунок 1 - Схема возможных состояний оборудования

Для определения условных оптимальных решений необходимо составить функциональное уравнение Беллмана.

Поставленную задачу можно рассматривать как задачу динамического программирования, в которой в качестве системы S выступает оборудование. Состояния этой системы определяется фактическим временем использования оборудования (его возраста) t, т.е. описывается единым параметром t.

Алгоритм решения задачи методом ДП реализуется в два этапа.

Первый этап. При движении от начала n-го периода к началу 1-го периода для каждого допустимого состояния оборудования находится условное оптимальное управление (решение) -$u(t)$.

Второй этап. При движении от начала 1-го периода к началу *n*-го периода из условных оптимальных решений составляется оптимальный план замены оборудования -$u^*(t)$.

Рассмотрим *n*-шаговый процесс (см. рисунок 1), считая *k*-м шагом номер *k*-го периода от начала эксплуатации *(k* = 1, 2, 3, ..., *n*). Уравнение на *k*-м шаге выбирается из двух возможных решений: u^c -сохранить и продолжить использование старого оборудования или $u^з$ -заменить оборудование новым.

Состояние S_{k-1} системы в начале *k*-го шага характеризуется параметром *t*

-возраст оборудования, который может принимать значения 0, 1, 2, .., *k* - 1, т.е. $t \le k - 1$.

Если к началу *k*-го шага система находится в состоянии S_{k-1} и возраст ее равен *t* периодам ($S_{k-1} = t$), то под влиянием уравнения U^c в конце *k*-го шага она перейдет в состояние S_k с возрастом оборудования *t*+ 1 ($S_k = t + 1$) (рисунок 1), т.е. возраст оборудования увеличится на один период. Под влиянием уравнения $U^з$, принятого на *k*-м шаге, система перейдет в состояние с возрастом оборудования, равным одному периоду. Замену произвели в начале *k*-го шага(S_k = 1).

Определим прибыль на *k*-м шаге (показатель эффективности *k*-го шага) соответствующую каждому из альтернативных управлений U^c *и* $U^з$.

Выбирая на *k*-м шаге управление U^c, мы сможем произвести продукцию стоимостью $f(t)$ на старом оборудовании, что потребует затрат $r(t)$, поэтому прибыль равна $f(t)$ -$r(t)$. Обозначим ее через

$$W_k^c = f(t) - r(t). \quad (1)$$

При управлении $U^з$ получим доход $\varphi(t)$ от продажи старого оборудования (ликвидную стоимость) и $f(0)$ от произведенной на новом оборудовании продукции, затратив *P* рублей на приобретение нового оборудования, и $r(0)$ -на содержание нового оборудования. В этом случае прибыль составит

$$W^з_k = \varphi(t) + f(0) - P - r(0). \quad (2)$$

Так как на последнем этапе процесса планирования мы можем действовать без учета предыдущих этапов и считать, что оптимальное управление на последнем этапе должно обеспечить максимальный доход за последний период, то функциональное уравнение, отражающее возможные решения, будет следующим:

$$W_n^*(t) = \max\begin{cases} f(t) - r(t) & \text{при } U_n = U^c, \\ \varphi(t) + f(0) - P - r(0) & \text{при } U_n = U^з \end{cases} \quad (3)$$

Сравнив эти две величины для всех возможных $i < n$ получим значение $W_n^*(t)$ и соответствующее значение оптимального управления $U^*_n(t)$.

Предположим, что для всех значений *t* S_k -о состояния системы известка максимальная прибыль, полученная за *n* - *k* шагов с *k* + 1 -го по *n*-й включительно. Поэтому основные рекуррентные соотношения можно записать в виде

$$W_k^*(t) = max\begin{cases} f(t) - r(t) + W^*_{k+1}(t+1) & при U_k = U^c, \\ \varphi(t) + f(0) - P - r(0) + W^*_{k+1}(1) & при U_k = U^з. \end{cases} \quad (4)$$

В уравнении (4) величина $W^*_{k+1}(1)$ -условная максимальная прибыль, полученная за *n* - *k* шагов, если к началу (*k* + 1)-го шага системы находились в состоянии S_k и *t* = 1 (возраст оборудования составлял 1 период).

НАХОЖДЕНИЕ КРАТЧАЙШИХ ПУТЕЙ В ГРАФЕ. РЕШЕНИЕ ЗАДАЧИ О МАКСИМАЛЬНОМ ПОТОКЕ

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.

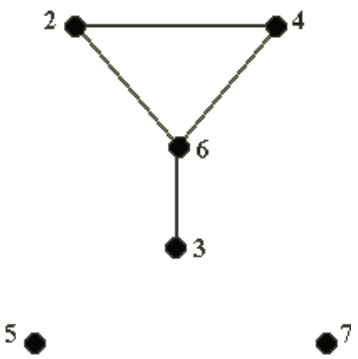


Рис. 1

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 дуг 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

- **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.

- **Матрица смежности**. Это матрица ***n×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 ***Г(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.

Практические задания

ЗАДАНИЕ 1.

Применяя метод Рунге-Кутта, вычислить на отрезке [1;1,5] таблицу значений решения уравнения Y” +Y’/X+Y=0. С граничными условиями Y(1)=0,77 и Y(1,5)=0,49281, выбрав шаг 0,1 с погрешностью 0,001.

ЗАДАНИЕ 2. Решить задачу из задания 1 методом прогонки и оцените полученную погрешность.

ЗАДАЧА 3. Двум предприятиям А и В на 4 квартала выделено $S_0 = 1000$ единиц средств. Каждый квартал предприятие А получает x средств, предприятие В - y средств. При этом от выделенных средств предприятие А получает 5x единиц и остаток средств 0,3x единиц, а предприятие В - доход 4y единиц и остаток выделенных средств 0,5y 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 і-ый квартал, S_i - остаток средств на конец і-ого квартала, і – 1,2,3,4.

№	Состояние	Вклад		Доход	Остаток
		А	В		
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
А				$0{,}3x_1$			$0{,}3x_2$			$0{,}3x_3$	
В				$0{,}5y_1$			$0{,}5y_2$			$0{,}5y_3$	
	$S_0=x_1+y_1$	$W_1=5x_1+4y_1$	$S_1=0{,}3x_1+0{,}5y_1$	$S_1=x_2+y_2$	$W_2=5x_2+4y_2$	$S_2=0{,}3x_2+0{,}5y_2$	$S_2=x_3+y_3$	$W_3=5x_3+4y_3$	$S_3=0{,}3x_3+0{,}5y_3$	$S_3=x_4+y_4$	$W_4=5x_4+4y_4$

Отыскание оптимального управления начнем с 4 квартала.

$$W_4^* = \max W_4 = \max\left(5x_4 + 4y_4\right) = \left\{\begin{array}{c} x_4 + y_4 = S_3, \\ x_4 = S_3 - y_4, S_3 = const \end{array}\right\} = \max\left(5\left(S_3 - y_4\right) + 4y_4\right) =$$

$$= \max_{0 \le y_4 \le S_3}\left(5S_3 - y_4\right) = \left(5S_3 - y_4\right)\Big|_{y_4=0} = 5S_3.$$

3 квартал.

$$W_{3-4}^* = \max(W_3 + W_4^*) = \max\left(5x_3 + 4y_3 + 5S_3\right) = \left\{\begin{array}{c} x_3 + y_3 = S_2, \quad x_3 = S_2 - y_3, \quad S_3 = 0{,}3x_3 + \\ + 0{,}5y_3 = 0{,}3S_2 - 0{,}3y_3 + 0{,}5y_3 = 0{,}3S_2 + 0{,}2y_3, \\ S_2 = const \end{array}\right\} =$$

$$= \max_{0 \le y_3 \le S_2}\left(5S_2 - 5y_3 + 4y_3 + 1{,}5S_2 + y_3\right) = \max_{0 \le y_3 \le S_2}\left(6{,}5S_2\right) = 6{,}5S_2.$$

Так как максимум дохода за 3-4 кварталы постоянен при любом распределении средств, то пусть $x_3 = \dfrac{S_2}{2}, \quad y_3 = \dfrac{S_2}{2}$.

2 квартал.

$$W_{2-4}^* = \max(W_2 + W_{3-4}^*) = \max\left(5x_2 + 4y_2 + 6{,}5S_2\right) = \left\{\begin{array}{c} x_2 + y_2 = S_1, \quad x_2 = S_1 - y_2, \quad S_2 = 0{,}3x_2 + \\ + 0{,}5y_2 = 0{,}3S_1 - 0{,}3y_2 + 0{,}5y_2 = 0{,}3S_1 + 0{,}2y_2, \\ S_1 = const \end{array}\right\} =$$

$$= \max_{0 \le y_2 \le S_1}\left(5S_1 - 5y_2 + 4y_2 + 1{,}95S_1 + 1{,}3y_2\right) = \max_{0 \le y_2 \le S_1}\left(6{,}95S_1 + 0{,}3y_2\right) = \left(6{,}95S_1 + 0{,}3y_2\right)\Big|_{y_2 = S_1} = 7{,}25S_1$$

1 квартал.

$$W^*_{1-4} = \max(W_1 + W^*_{2-4}) = \max\left(5x_1 + 4y_1 + 7{,}25S_1\right) = \left\{\begin{array}{c} x_1 + y_1 = S_0, \quad x_1 = S_0 - y_1, \quad S_1 = 0{,}3x_1 + \\ + 0{,}5y_1 = 0{,}3S_0 - 0{,}3y_1 + 0{,}5y_1 = 0{,}3S_0 + 0{,}2y_1, \\ S_0 = const \end{array}\right\} =$$

$$= \max_{0 \le y_1 \le S_0} \left(5S_0 - 5y_1 + 4y_1 + 2{,}175S_0 + 1{,}45y_1\right) = \max_{0 \le y_1 \le S_0} \left(7{,}175S_0 + 0{,}45y_1\right) =$$

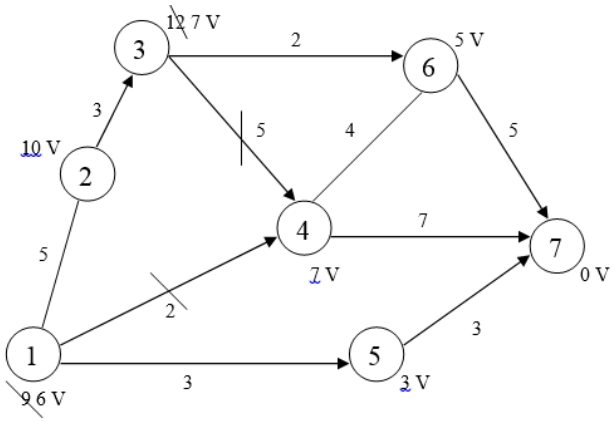
$$\left(7{,}175S_0 + 0{,}45y_1\right)\Big|_{y_1 = S_0} = 7{,}625S_0$$

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 4. 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>б, 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.

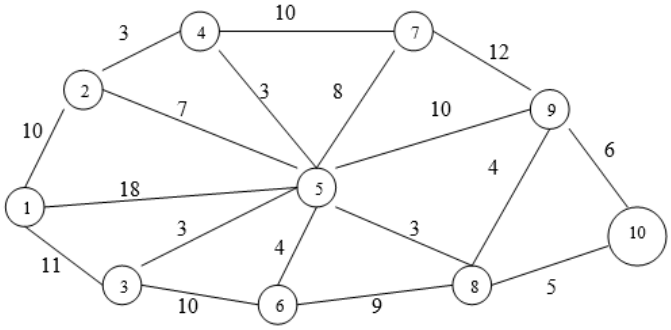
Номера точек, между которыми рассчитывается расстояние	Кратчайшее расстояние	Маршрут, по которому проходит кратчайшее расстояние
1-7	6	1-5-7
2-7	10	2-3-6-7
3-7	7	3-6-7
4-7	7	4-7
5-7	3	5-7
6-7	5	6-7
7-7	0	

Задания для самостоятельной работы

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.

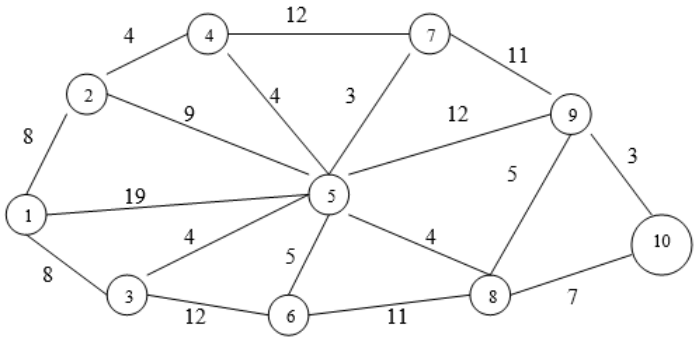
Задача 2. По заданной схеме, соединяющей 10 точек, найти кратчайшее расстояние от 1 точки до 10.



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.

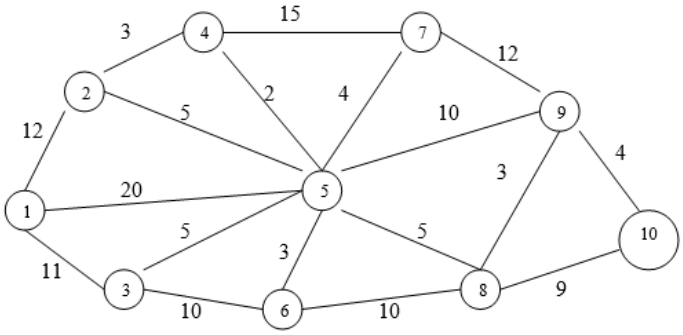
Задача 2. По заданной схеме, соединяющей 10 точек, найти кратчайшее расстояние от 1 точки до 10.



3 вариант.

Задача 1. Планируется работа двух отраслей производства А и В на 4 года. Количество x средств, вложенных в отрасль А, позволяет получить доход 5x и уменьшается до 0,1x. Количество y средств, вложенных в отрасль В, позволяет получить доход 3y и уменьшается до 0,5y. 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.



ЗАДАНИЕ 5. Составить план замены оборудования по исходным данным представленным таблицей, где значения *f(t)* и *r(t)* даны в условных единицах. Первоначальная стоимость оборудования равна 15 условным единицам (у.е.).

Таблица 1 - Исходные данные для задачи оптимизации

Возраст оборудования t , год						
Стоимость продукции *f(t),* у.е.	4	3	3	3	0	0
Эксплуатационные затраты *r(t),* у.е.	5	5	6	7	8	0

1. Нахождение решения исходной задачи начинаем с определения условного оптимального управления (решения) для последнего 5-го года, в связи с чем находим множество допустимых состояний оборудования к началу данного года. Так как в начальный момент имеется новое оборудование ($t^{(1)}=0$), то возраст оборудования к началу 5-го года может составлять 1,2,3,4 года. Поэтому допустимые состояния системы на данный период времени таковы: ($t_1^{(5)}=1$), ($t_2^{(5)}=2$), ($t_3^{(5)}=3$), ($t_4^{(5)}=4$). Для каждого из этих состояний найдем условное оптимальное решение и соответствующее значение функции $W_5(t^{(5)})$.

Используя приведенные ранее уравнения и соотношение $W_6(t^{(i+1)}) = 0$ (так как рассматривается последний год расчетного периода), получаем:

$$W_5(t^{(5)}) = \max\left\{\frac{f(t^{(5)}) - r(t^{(5)}),}{f(t^{(5)}=0) - r(t^{(5)}=0) - P.}\right.$$

Подставляя теперь в эту формулу вместо $t^{(5)}$ его значение, равное 1, и учитывая данные таблицы 1, находим:

$$W_5(t_1^{(5)}) = \max\left\{\begin{array}{l} f(t^{(5)}=1) - r(t^{(5)}=1), \\ f(t^{(5)}=0) - r(t^{(5)}=0) - P. \end{array}\right\} = \max\left\{\begin{array}{l} 23-15, \\ 24-15-15. \end{array}\right\} = 8, \quad U = U^c.$$

Значит, условное оптимальное решение в данном случае - сохранить оборудование.

Проведем аналогичные вычисления для других допустимых состояний оборудования к началу 5-го года.

$$W_5(t_2^{(5)}) = \max\left\{\begin{array}{l} 22-16, \\ 24-15-15. \end{array}\right\} = 6, \quad U = U^c,$$

$$W_5(t_3^{(5)}) = \max\left\{\begin{array}{l} 22-17, \\ 24-15-15. \end{array}\right\} = 5, \quad U = U^c,$$

$$W_5(t_4^{(5)}) = \max\left\{\begin{array}{l} 20-18, \\ 24-15-15. \end{array}\right\} = 2, \quad U = U^c,$$

Полученные данные сводим в таблицу 2.

Таблица 2 - Варианты условных оптимальных решений

Возраст оборудования $t^{(5)}$, год	**Значение функции $W5(t^{(5)})$, у.е.**	**Условное оптимальное решение, U**
1	8	U^C
2	6	U^C
3	5	U^C
4	2	U^C

2. Рассмотрим теперь возможные состояния оборудования к началу 4-го года. Очевидно, что допустимыми состояниями являются ($t_1^{(4)}$=1), ($t_2^{(4)}$=2), ($t_3^{(4)}$=3). Для каждого из этих состояний найдем условное оптимальное решение и соответствующее значение функции $W_4(t^{(4)})$. Для этого используем уравнение и данные таблиц 1, 2. Так, в частности, для $t_1^{(4)}$=1 имеем

$$W_4(t_1^{(4)}) = \max\left\{\begin{array}{l} f(t^{(4)}=1) - r(t^{(4)}=1) + W_5(t^{(5)}=2), \\ f(t^{(4)}=0) - r(t^{(4)}=0) - P + W_5(t^{(5)}=1). \end{array}\right\} =$$

$$= \max\left\{\begin{array}{l} 23-15+6, \\ 24-15-15+8. \end{array}\right\} = 14, \quad U = U^c.$$

Значит, условное оптимальное решение в данном случае - сохранить оборудование.

Проведем аналогичные вычисления для других допустимых состояний оборудования к началу 4-го года.

$$W_4(t_2^{(4)}) = \max\left\{\begin{array}{l} 22-16+5, \\ 24-15-15+8. \end{array}\right\} = 11, \quad U = U^c,$$

$$W_4(t_3^{(4)}) = \max\left\{\begin{array}{l} 22-17+2, \\ 24-15-15+8. \end{array}\right\} = 7, \quad U = U^c.$$

Полученные данные сводим в таблицу 3.

Таблица 3- Варианты условных оптимальных | решений

Возраст оборудования $t^{(4)}$, год	**Значение функции $W^{(4)}$,у.е.**	**Условное оптимальное решение, U**
1	14	U^C
2	11	U^C
3	7	U^C

3. Определим теперь условное оптимальное решение для каждого из допустимых состояний оборудования к началу 3-го года. Очевидно, такими состояниями являются ($t_1^{(3)}$=1), ($t_2^{(3)}$=2). В соответствии с уравнением и табл. 1,2,3 имеем

$$W_3(t_1^{(3)}) = \max\left\{\begin{array}{l} f(t^{(3)}=1) - r(t^{(3)}=1) + W_4(t^{(4)}=2), \\ f(t^{(3)}=0) - r(t^{(4)}=0) - P + W_4(t^{(4)}=1). \end{array}\right\} =$$

$$= \max\left\{\begin{array}{l} 23-15+11, \\ 24-15-15+14. \end{array}\right\} = 19, \quad U = U^c.$$

$$W_3(t_2^{(3)}) = \max\left\{\begin{array}{l} f(t^{(3)}=2) - r(t^{(3)}=2) + W_4(t^{(4)}=3), \\ f(t^{(3)}=0) - r(t^{(4)}=0) - P + W_4(t^{(4)}=1). \end{array}\right\} =$$

$$= \max\left\{\begin{array}{l} 22-16+7, \\ 24-15-15+14. \end{array}\right\} = 13, \quad U = U^c.$$

Полученные данные сводим в таблицу 4.

Таблица 4 - Варианты условных оптимальных решений

Возраст оборудования t^4, год	**Значение функции $W3(t^{(3)})$,у.е.**	**Условное оптимальное решение, U**
1	19	U^C
2	13	U^C

4. Наконец, рассмотрим допустимые состояния оборудования к началу 2-го года. Очевидно, что на данный момент времени возраст оборудования может быть равен только одному году. Тогда имеем

$$W_2(t_1^{(2)}) = \max\left\{\begin{array}{l} f(t^{(2)}=1) - r(t^{(2)}=1) + W_3(t^{(3)}=2), \\ f(t^{(2)}=0) - r(t^{(2)}=0) - P + W_3(t^{(3)}=1) \end{array}\right\} =$$

$$= \max\left\{\begin{array}{l} 23-15+13, \\ 24-15-15+19. \end{array}\right\} = 21, \quad U = U^c.$$

Полученные данные сводим в таблицу 5.

Таблица 5 - Варианты условных оптимальных решений

Возраст оборудования t^4 , год	**Значение функции $W2(t^a)$, у.е.**	**Условное оптимальное решение, U**
1	21	U^C

5. Согласно условию в начальный момент установлено новое оборудование ($t_1^{(1)}$=0). Поэтому проблемы выбора между сохранением и заменой оборудования не существует:

оборудование следует сохранить. Значит, условным оптимальным решением является U^c, и значение функции

$$W_1(t_1^{(1)}) = f(t_1^{(1)} = 0) - r(t_1^{(1)} = 0) + W_2(t^{(1)} = 1) = 24 - 15 + 21 = 30.$$

Таким образом, получилось так, что оборудование на всем протяжении периода эксплуатации менять ненужно.

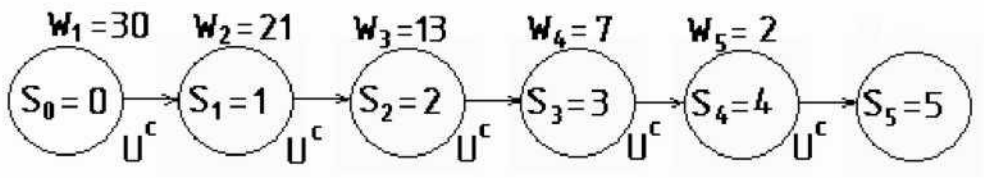


Рисунок 2 - Оптимальное решение задачи замены оборудования

Задания для самостоятельной работы

Вариант №1

Составить оптимальный план замены оборудования по исходным данным, представленным таблицей 6, где f(t) и r(t) даны в условных единицах. Первоначальная стоимость оборудования равна 15 условным единицам.

Таблица 6 - Исходные данные для задачи оптимизации

Возраст оборудования t, год	0	1	2	3	4	5
Стоимость продукции f(t)	24	23	22	22	20	19
Эксплуатационные затраты r(t)	15	15	16	17	18	19

Вариант №2

Составить план замены оборудования в течение пяти лет, при котором общая прибыль в данный период времени максимальна, если затраты, связанные с приобретением и установкой нового оборудования, составляют 35 тыс. руб.

Другие исходные данные представлены в таблице 7.

Таблица 7 - Исходные данные для задачи оптимизации

Возраст оборудования t, год	0	1	2	3	4	5
Годовой выпуск продукции f(t), тыс. руб.	70	65	55	50	50	45
Эксплуатационные затраты r(t), тыс. руб.	10	15	20	25	35	45

Вариант №3

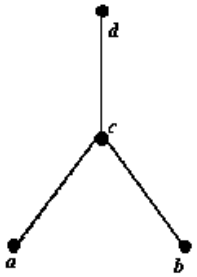
Составить оптимальный план замены оборудования по исходным данным, представленным таблицей 8, где f(t) и r(t) даны в условных единицах. Первоначальная стоимость оборудования равна 15 условным единицам.

Таблица 8 - Исходные данные для задачи оптимизации

Возраст оборудования t, год	0	1	2	3	4	5
Стоимость продукции f(t)	35	30	28	27	26	25

ЗАДАНИЕ 6.

Задача 1. Составить матрицы инцидентности и смежности для графа:



Решение.

Матрица инцидентности

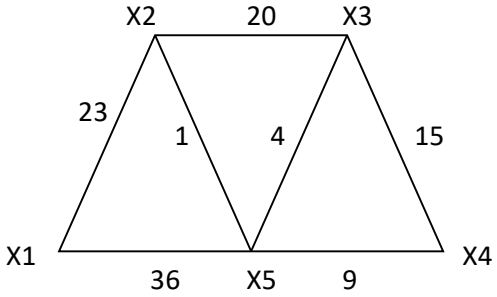
	u	***v***	***w***
a	1	0	0
b	0	0	1
c	1	1	1
d	0	1	0

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

Где ***u, v, w*** – ребра данного графика

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			

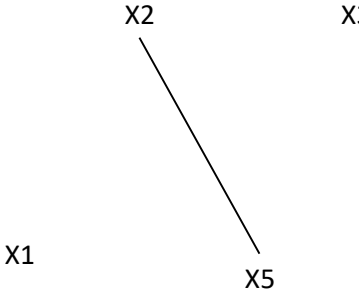
X2 X

X1 X5

X4			15		
X5	36	1	4	9	

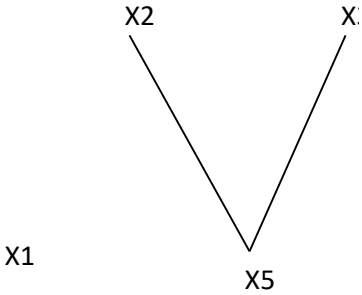
Из элементов матрицы выбираем минимальный - ***(X2,X5) = 1***. Обводим выбранный элемент кружком и указываем на рисунке соответствующее ребро.

	X1	X2	X3	X4	X5
X1					
X2	23				
X3		20			
X4			15		
X5	36	1	4	9	



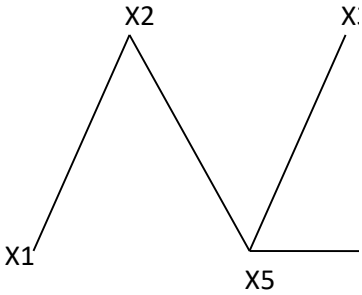
Из оставшихся элементов выбираем минимальный - ***(X3,X5) = 4***. Элемент обводим кружком. Чтобы выполнялось условие **2** пункты **X2** и **X3** не должны соединяться, поэтому элемент ***(X2,X3)*** зачёркивается. И т.д.

	X1	X2	X3	X4	X5
X1					
X2	23				
X3		20			
X4			15		
X5	36	1	4	9	



В итоге получаем:

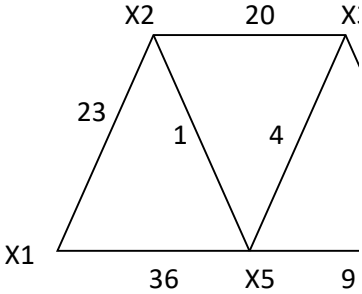
	X1	X2	X3	X4	X5
X1					
X2	23				
X3		20			
X4			15		
X5	36	1	4	9	



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,

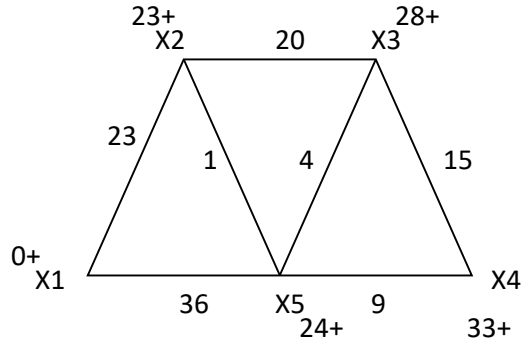
X3: I(X3)=28*, p=28, рядом с точкой X3 поставим расстояние 28.

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.



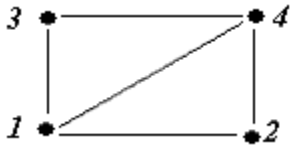
Запишем ответ в виде таблицы кратчайших расстояний от точки X1 до всех остальных точек графа.

Кратчайший путь	значение
X1-X2	23
X1-X2-X5-X3	28
X1-X2-X5-X4	33
X1-X2-X5	24

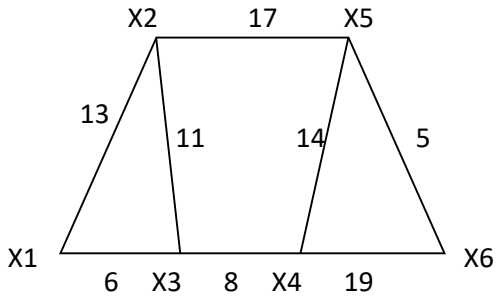
Задания для самостоятельной работы

1 вариант

Задача 1. Составить матрицы инцидентности и смежности для графа:

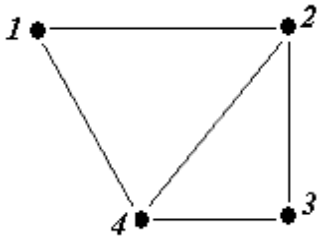


Задача 2. На представленном графе найдите: а) минимальный остов дерева, б) найдите кратчайший путь от начальной точки X1 до всех остальных точек.

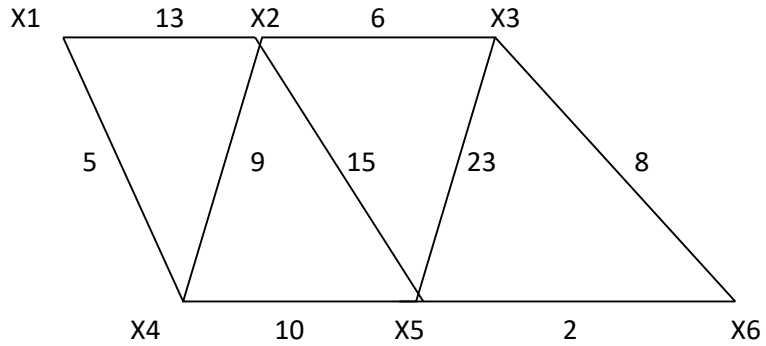


2 вариант

Задача 1. Составить матрицы инцидентности и смежности для графа:

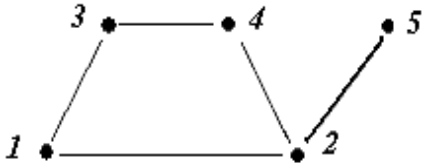


Задача 2. На представленном графе найдите: а) минимальный остов дерева, б) найдите кратчайший путь от начальной точки X1 до всех остальных точек.

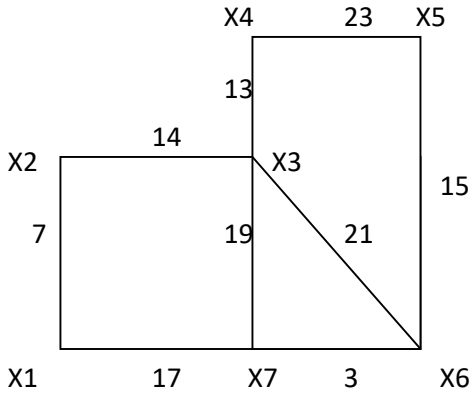


3 вариант

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.



Контрольные вопросы:

1. Что называется, дифференциальным уравнением.
2. Какое дифференциальное уравнение называется дифференциальным уравнением второго порядка?
3. Какая задача называется краевой задачей?
4. Что называется, граничными или краевыми условиями?
5. В чем заключается сущность метода стрельбы?
6. Что называется, динамическим программированием?
7. Какие характерные особенности задач динамического программирования вы знаете?
8. Что называется, управлением?
9. В чем состоит метод динамического программирования?
10. Сформулируйте принцип оптимальности Беллмана?
11. Что называется, сетью, звеньями?
12. Что такое характеристика точки?
13. К какому классу задач относится задача о замене оборудования?
14. В чем состоит оптимальная стратегия замены оборудования?
15. Назовите основные функциональные характеристики оборудования.
16. Для чего нужно функциональное уравнение Беллмана?
17. На какие этапы делится решение задачи динамического программирования?
18. Дайте определение граф.
19. В чем состоит отличие ориентированного графа от неориентированного графа?

20. В чем отличие пустого графа от простого графа?
21. Как определить степень вершины?
22. Чем отличается цепь в графе от цикла?
23. Дайте понятие подграф графа.
24. В чем суть связанного графа?
25. Как находятся матрицы инцидентности и матрицы смежности?
26. Как найти минимальный остов дерева?
27. Как найти кратчайшее расстояние в графе?

Практическая работа №3. Составление систем уравнений Колмогорова. Нахождение финальных вероятностей. Нахождение характеристик простейших систем массового обслуживания. Решение задач массового обслуживания методами имитационного моделирования.

Цели занятия:

- научиться составлять систему уравнений Колмогорова, решать ее относительно финальных вероятностей состояний;
- построить графы состояний и найти параметры для простейших систем массового обслуживания.

Краткие теоретические сведения

Построение математических моделей в условиях неопределенности — очень сложная или невыполнимая задача. Лишь для некоторых упрощенных случаев можно построить математическую модель.

Следует различать два вида неопределенности:

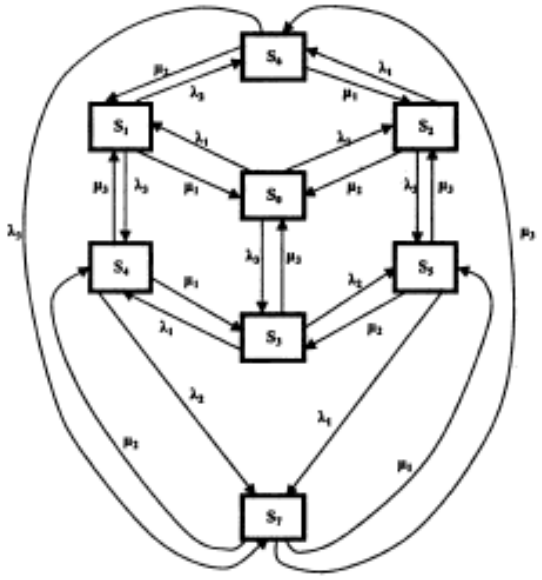
- вероятностные характеристики либо известны, либо могут быть получены в результате эксперимента. Такая неопределенность называется стохастической, и для большинства объектов, содержащих такую неопределенность, можно построить математическую модель, например, выход из строя оборудования, приход нового клиента и т.д.
- вероятностные характеристики определить невозможно. В этом случае задачу можно попытаться решить с помощью экспертных оценок, но результат будет весьма приблизительным, например, каковы будут модели женской одежды через пять лет?

Строгую математическую модель с аналитическим вычислением всех интересующих величин можно построить только в том случае, если случайный процесс носит марковский характер.

Случайный процесс будет **марковским**, если вероятностные характеристики процесса в момент времени t зависят только от текущего (настоящего) состояния процесса в этот момент времени t и не зависят от того, как (каким способом и когда) рассматриваемый процесс перешел в текущее состояние.

Размеченным графом будем считать такой граф, у которого стрелками указаны переходы из одного состояния в другое, а рядом со стрелкой указана интенсивность перехода. Будем различать две интенсивности — прямую λ и обратную μ. Тогда λ_1, λ_2, λ_3 — интенсивности потоков отказов соответственно первого, второго и третьего узлов, а μ_1, μ_2, μ_3 — соответственно интенсивности потоков возвратов (ремонтов) узлов.

$$\begin{cases} \frac{dp_1(t)}{dt} = \lambda_1 p_0 + \mu_3 p_4 + \mu_2 p_6 - (\mu_1 + \lambda_2 + \lambda_3) p_1; \\ \frac{dp_2(t)}{dt} = \lambda_2 p_0 + \mu_3 p_5 + \mu_1 p_6 - (\mu_2 + \lambda_1 + \lambda_3) p_2; \\ \frac{dp_3(t)}{dt} = \lambda_3 p_0 + \mu_2 p_5 + \mu_1 p_4 - (\mu_3 + \lambda_1 + \lambda_2) p_3; \\ \frac{dp_4(t)}{dt} = \lambda_1 p_3 + \lambda_3 p_1 + \mu_2 p_7 - (\mu_1 + \mu_3 + \lambda_2) p_4; \\ \frac{dp_5(t)}{dt} = \mu_1 p_7 + \lambda_2 p_3 + \lambda_3 p_2 - (\lambda_1 + \mu_2 + \mu_3) p_5; \\ \frac{dp_6(t)}{dt} = \lambda_1 p_2 + \lambda_2 p_1 + \mu_3 p_7 - (\mu_1 + \mu_2 + \lambda_3) p_6; \\ \frac{dp_7(t)}{dt} = \lambda_1 p_5 + \lambda_2 p_4 + \lambda_3 p_6 - (\mu_1 + \mu_2 + \mu_3) p_7; \\ \frac{dp_0(t)}{dt} = \mu_1 p_1 + \mu_2 p_2 + \mu_3 p_3 - (\lambda_1 + \lambda_2 + \lambda_3) p_0. \end{cases}$$



Эта система дифференциальных уравнений называется системой уравнений Колмогорова. Имеем систему из линейных дифференциальных уравнений. Известно, что сумма всех вероятностей равна единице, т. е. $P_0+P_1+P_2+P_3+P_4+P_5+P_6+P_7=1$.

Эту систему можно составить непосредственно по размеченному графу состояний, если руководствоваться правилом, согласно которому слева в уравнениях стоит предельная вероятность данного состояния p_i, умноженная на суммарную интенсивность всех потоков, ведущих из данного состояния, а справа — сумма произведений интенсивностей всех потоков, входящих в i-е состояние, на вероятности тех состояний, из которых эти потоки исходят.

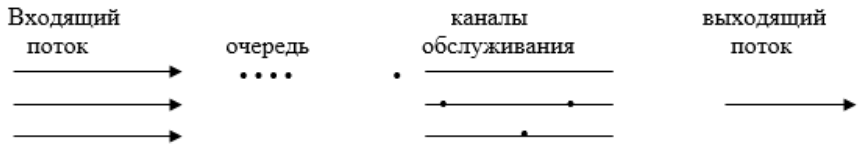
Характеристики систем массового обслуживания

Часто приходится сталкиваться с такими ситуациями: очередь покупателей в кассах магазинов; колонна автомобилей, движение которых остановлено светофором; ряд станков, вышедших из строя и ожидающих ремонта, и т.д. Все эти ситуации объединяет то, что системам необходимо пребывать в состоянии ожидания. Ожидание является следствием вероятностного характера возникновения потребностей в обслуживании и разброса показателей обслуживающих систем, которые называют *системами массового обслуживания (СМО).*

Цель изучения СМО состоит в том, чтобы взять под контроль некоторые характеристики системы, установить зависимость между числом обслуживаемых единиц и качеством обслуживания. Качество обслуживания тем выше, чем больше число обслуживаемых единиц. Но экономически невыгодно иметь лишние обслуживающие единицы.

В промышленности СМО применяются при поступлении сырья, материалов, комплектующих изделий на склад и выдаче их со склада; обработке широкой номенклатуры деталей на одном и том же оборудовании; организации наладки и ремонта оборудования; определении оптимальной численности обслуживающих отделов и служб предприятий и т.д.

Основными элементами СМО являются источники заявок; их входящий поток; каналы обслуживания и выходящий поток.



В зависимости от характера формирования очереди СМО различают:

1. системы с отказами, в которых при занятости всех каналов обслуживания заявка не встает в очередь и покидает систему необслуженной;
2. системы с неограниченными ожиданиями, в которых заявка встает в очередь, если в момент ее поступления все каналы были заняты.
3. системы смешанного типа с ожиданием и ограниченной длиной очереди: заявка получает отказ, если приходит в момент, когда все места в очереди заняты. Заявка, попавшая в очередь, обслуживается обязательно.

По числу каналов обслуживания СМО делятся на одноканальные и многоканальные.

В зависимости от расположения источника требований, системы могут быть разомкнутыми (источник заявок находится вне системы) и замкнутыми (источник находится в самой системе).

Рассмотрим в отдельности элементы СМО.

Входящий поток: на практике наиболее распространенным является простейший поток заявок, обладающий свойствами стационарности ординарности и отсутствия последействия.

Стационарность характеризуется тем, что вероятность поступления определенного количества требований (заявок) в течение некоторого промежутка времени зависит только от длины этого промежутка.

Ординарность потока определяется невозможностью одновременного появления двух или более заявок.

Отсутствие последействия характеризуется тем, что поступление заявки не зависит от того, когда и сколько заявок поступило до этого момента. В этом случае вероятность того, что число заявок, поступивших на обслуживание за промежуток времени *t*, равно *k*, определяется по закону Пуассона.

$P_k(t)=((\lambda t)^k / k!) e^{-\lambda t}$

где λ – интенсивность потока заявок, т.е. среднее число заявок в единицу времени: $\lambda = 1/\bar{\tau}$ (чел/мин, р/ч, автом/дн, кВт/ч), где $\lambda = \bar{\tau}$ – среднее значение интервала времени между двумя соседними заявками;

k – число заявок, поступивших на обслуживание за промежуток времени *t*.

Для такого потока время между двумя соседними заявками распределено экспоненциально с плотностью вероятности:

$f(t)= \lambda e^{-\lambda t}$

Случайное время ожидания в очереди начала обслуживания считают распределенным экспоненциально:

$f(t)=\upsilon e^{-\upsilon t}$,

где υ – интенсивность движения очереди, т.е. среднее число заявок, приходящихся на обслуживание в единицу времени: $\upsilon = 1/\overline{t_{оч}}$, где $\overline{t_{оч}}$ – среднее значение времени ожидания в очереди.

Выходящий поток заявок связан с потоками обслуживания в канале, где длительность обслуживания $\overline{t_{обс}}$ является случайной величиной и часто подчиняется показательному закону распределения с плотностью

$f(t_{обс})= \mu e^{-\mu t}$,

где μ – интенсивность потока обслуживания, т.е. среднее число заявок, обслуживаемых в ед. времени: $\mu=1/ t_{обс}$(чел/мин, р/дн, кг/ч, докум/дн), *t* – среднее время обслуживания.

Важной характеристикой СМО, объединяющей λ и μ, является интенсивность нагрузки

$\rho=\lambda/ \mu$

СМО с отказами

Заявка, поступившая в систему с отказами и нашедшая все каналы занятыми, получает отказ и покидает систему необслуженной. Показателем качества обслуживания выступает вероятность получения отказа. Предполагается, что все каналы доступны в равной степени всем заявкам, входящий поток является простейшим, длительность (время) обслуживания одной заявки ($t_{обс}$) распределена по показательному закону.

Формулы для расчета установившегося режима

1. Вероятность простая каналов обслуживания, когда нет заявок (*k* = 0): $P_0 = \dfrac{1}{\sum_{k=0}^{n}(\rho^k / k!)}$

2. Вероятность отказа в обслуживании, когда поступившая на обслуживание заявка найдет все каналы занятыми (*k* = *n*): $P_{отк} = P_n = \dfrac{P_0 \cdot \rho^n}{n!}$

3. Вероятность обслуживания: $P_{обс}= 1- P_{отк}$

4. Среднее число занятых обслуживанием каналов: $\overline{n_з} = \rho \cdot P_{обс}$

5. Доля каналов, занятых обслуживанием: $k_з = \dfrac{\overline{n_з}}{n}$

6. Абсолютная пропускная способность СМО: $A=\lambda P_{обс}$

СМО с неограниченным ожиданием

Заявка, поступившая в систему с неограниченным ожиданием и нашедшая все каналы занятыми, становится в очередь, ожидая освобождения одного из каналов.

Основной характеристикой качества обслуживания является время ожидания (время пребывания заявки в очереди).

Для таких систем характерно отсутствие отказа в обслуживании, т.е. $P_{отк}=0$ и $P_{обс}=1$.

Для систем с ожиданием существует дисциплина очереди:

1. обслуживание в порядке очереди по принципу «первым пришел – первым обслужен»;
2. случайное неорганизованное обслуживание по принципу «последний пришел - первым обслужен»;
3. обслуживание с приоритетами по принципу «генералы и полковники вне очереди».

Формулы для расчета установившегося режима

1. Вероятность простоя каналов, когда нет заявок (k=0): $P_0 = \dfrac{1}{\sum_{k=0}^{n}(\rho^k / k!)} + \dfrac{\rho^{n+1}}{n!(n-\rho)}$

Предполагается, что $\rho/n<1$, т.е. интенсивность нагрузки меньше числа каналов.

2. Вероятность занятости обслуживанием k заявок: $P_k = \dfrac{P_0 \cdot \rho^k}{k!}$, $1 \le k \le n$

3. Вероятность занятости обслуживанием всех каналов: $P_n = \dfrac{P_0 \cdot \rho^n}{n!}$

4. Вероятность того, что заявка ожидается в очереди: $P_{оч} = \dfrac{\rho^{n+1}}{n!(n-\rho)} \cdot P_0$

5. Среднее число заявок в очереди: $\overline{L_{оч}} = \dfrac{\rho^{n+1}}{(n+\lambda)!(n-\rho)^2} \cdot P_0$

6. Среднее время ожидания заявки в очереди: $\overline{t_{оч}} = \dfrac{\overline{L_{оч}}}{\lambda}$

7. Среднее время ожидания заявки в СМО: $\overline{t_{смо}} = \overline{t_{оч}} + t_{обс}$

8. Среднее число занятых обслуживанием каналов: $\overline{n_з} = \rho$

9. Среднее число свободных каналов: $\overline{n_{св}} = n - \overline{n_з}$

10. Коэффициент занятости каналов обслуживания: $k_з = \dfrac{\overline{n_з}}{n}$

11. Среднее число заявок в СМО: $\bar{z} = L_{оч} + \overline{n_з}$

СМО с ожиданием и с ограниченной длиной очереди

Заявка, поступившая в систему с ожиданием с ограниченной длиной очереди и нашедшая все каналы и ограниченную очередь занятыми, покидает систему необслуженной.

Основной характеристикой качества системы является отказ заявке в обслуживании. Ограничения на длину очереди могут быть из-за:

1. ограничения сверх времени пребывания заявки в очереди;

2. ограничения сверх длины очереди;
3. ограничения общего времени пребывания заявки в системе.

Формулы для установившегося режима

1. Вероятность простоя каналов, когда нет заявок (k=0):

$$P_0 = \frac{1}{\sum_{k=0}^{n} \frac{\rho^k}{k!} + \frac{\rho^{n+1}}{n!\cdot(n-\rho)} \cdot \left(1 - \left(\frac{\rho}{n}\right)^m\right)}$$

n – число каналов;

m – длина накопителя;

ρ – интенсивность нагрузки;

K – число заявок, поступивших на обслуживание за промежуток времени t.

2. Вероятность отказа в обслуживании: $P_{отк} = \frac{P_0 \cdot \rho^{n+m}}{n! \cdot n^m}$
3. Вероятность обслуживания: $P_{обс} = 1 - P_{отк}$
4. Абсолютная пропускная способность: $A = \lambda P_{обс}$
5. Среднее число занятых каналов: $\overline{n_з} = \frac{A}{\mu} = \frac{\lambda \cdot P_{обс}}{\mu} = \rho \cdot P_{обс}$, где $\rho = \lambda / \mu$
6. Среднее число заявок в очереди: $\overline{L_{оч}} = \frac{\rho^{n+1}}{n \cdot n!} \cdot \frac{1 - (\rho/n)^m (m + 1 - m\rho/n)}{(1 - \rho/n)^2} \cdot P_0$
7. Среднее время ожидания обслуживания: $\overline{t_{оч}} = \frac{\overline{L_{оч}}}{\lambda}$
8. Среднее число заявок в системе: $\bar{z} = L_{оч} + \overline{n_з}$
9. Среднее время пребывания в системе: $t_{смо} = z/\lambda$ $\overline{t_{смо}} = \frac{\bar{z}}{\lambda}$

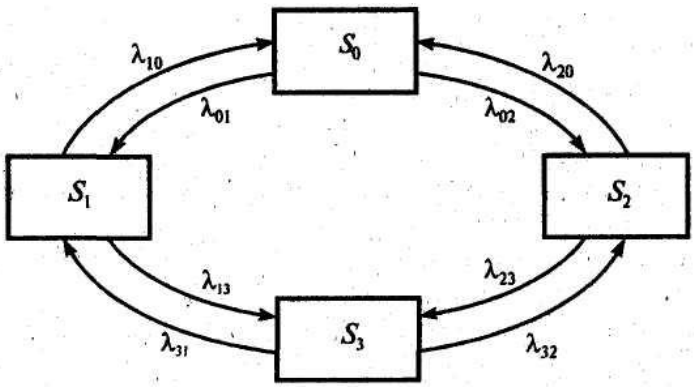
Практические задания

Пример 1. Найти предельные вероятности для системы S, граф состояний которой приведен на рисунке, при $\lambda_{01}=1$, $\lambda_{02}=2$, $\lambda_{10}=2$, $\lambda_{13}=2$, $\lambda_{20}=3$, $\lambda_{23}=1$, $\lambda_{31}=3$, $\lambda_{32}=2$.

Решение

Система алгебраических уравнений, описывающих стационарный режим для данной системы, имеет вид

$$\begin{cases} (\lambda_{01} + \lambda_{02}) \cdot p_0 = \lambda_{10} \cdot p_1 + \lambda_{20} \cdot p_2, \\ (\lambda_{10} + \lambda_{13}) \cdot p_1 = \lambda_{01} \cdot p_0 + \lambda_{31} \cdot p_3, \\ (\lambda_{20} + \lambda_{23}) \cdot p_2 = \lambda_{02} \cdot p_0 + \lambda_{32} \cdot p_3, \\ (\lambda_{31} + \lambda_{32}) \cdot p_3 = \lambda_{13} \cdot p_1 + \lambda_{23} \cdot p_2. \end{cases} \quad (1)$$



или

$$\begin{cases} 3 \cdot p_0 = 2 \cdot p_1 + 3 \cdot p_2 \\ 4 \cdot p_1 = p_0 + 3 \cdot p_3 \\ 4 \cdot p_2 = 2 \cdot p_0 + 2 \cdot p_3 \\ p_0 + p_1 + p_2 + p_3 = 1. \end{cases}$$

Здесь мы вместо одного "лишнего" уравнения системы (1) записали нормировочное условие.

Решив систему, получим $p_0 = 0{,}4$, $p_1 = 0{,}2$, $p_2 = 0{,}27$, $p_3 = 0{,}13$, т.е. в предельном, стационарном режиме система S в среднем 40% времени будет находиться в состоянии S_0 (оба узла исправны), 20% — в состоянии S_1 (первый узел ремонтируется, второй работает), 27% — в состоянии S_2 (второй узел ремонтируется, первый работает) и 13% времени — в состоянии S_3 (оба узла ремонтируются).

Пример 2. Дежурный администратор города имеет 5 телефонов. Звонки поступают с интенсивностью 90 звонков/час. Средняя продолжительность разговора составляет 2 мин.

Определить характеристики дежурного администратора как системы массового обслуживания.

Решение

1. Классифицируем СМО:

- с отказами (нет накопителя);
- многоканальная (5 телефонов = 5 каналов).

2. Обозначения:

- λ – интенсивность потока заявок (λ=90зв/60мин=3зв/2мин)
- n – число каналов (n=5);
- μ – интенсивность потока обслуживания, т.е. среднее число заявок, обслуживаемых в единицу времени ($\mu=1/ t_{обс}$)
- $t_{обс}$ – среднее время обслуживания ($t_{обс}$=2мин)
- ρ – интенсивность нагрузки;
- k – номер заявки (число заявок), k=n=5;
- P_0 – вероятность простоя каналов обслуживания, когда нет заявок;
- $P_{отк}$ – вероятность отказа в обслуживании, когда поступившая на обслуживание заявка найдет все каналы занятыми;
- $P_{обс}$ – вероятность обслуживания.

- $n_з = \rho * P_{обс}$ - среднее число занятых обслуживанием каналов.
- $к_з = n_з / n$ - для каналов, занятых обслуживанием.
- $A = \lambda P_{обс}$ - абсолютная пропускная способность СМО.

3. Определяем характеристики данной СМО:

а) $\rho = \lambda/\mu = \lambda/(1/t_{обс}) = \lambda t_{обс} = 3/2 * 2 = 3$

б) $P_0 = \frac{1}{\sum_{k=0}^{n}(\rho^k / k!)} =$ $1/ (\rho^0/0!)+(\rho^1/1!)+(\rho^2/2!)+(\rho^3/3!)+(\rho^4/4!)+(\rho^5/5!) =$

$=1/ (1+3/1)+(3*3/1*2)+(3*3*3/1*2*3)+(3*3*3*3/1*2*3*4)+(3*3*3*3*3/1*2*3*4*5) =$
$=1/ 1+3+(9/2)+(27/6)+(81/24)+(243/120)=0,054$

в) $P_{отк} = \frac{P_0 \cdot \rho^n}{n!} =$ $(3^5/ 1*2*3*4*5)*0,054=$

$= (3*3*3*3*3/1*2*3*4*5)*0,054=(243/120)*0,054=0,12$

г) $P_{обс} = 1- P_{отк}= 1-0,12=0,88$

д) $n_з = \rho*P_{обс}= 3*0,88=2,6$

е) $к_з = n_з / n = 2,6/5=0,52$

ж) $A = \lambda P_{обс} = (3/2)*0,88 = 1,31.$

Пример 3. На автомобильной стоянке возле магазина имеется 2 места. Рядом находится площадка на 2 а/м. На стоянку прибывает 1 машина в 3 мин. Среднее время нахождения водителя в магазине 2 мин. Определить характеристики этой СМО.

Решение

1. Классифицируем СМО:

- с ограниченной длиной очереди
- с накопителем
- многоканальная
- с ограничением общего времени пребывания заявки в системе СМО с ожиданием и с ограниченной длиной очереди.

2. Обозначения:

- m=2 - длина накопителя
- n=2 - число каналов

Остальные обозначения - как в Примере 2.

3. Определяем характеристики данной СМО:

а) $\lambda = 1/3$;

б) $t_{обс} = 2$ мин;

в) $\rho = \lambda/\mu = \lambda/(1/t_{обс}) = \lambda\, t_{обс} = (1/3)*2=2/3$.

г) Вероятность простоя каналов: $P_0 = \dfrac{1}{\sum_{k=0}^{n} \dfrac{\rho^k}{k!} + \dfrac{\rho^{n+1}}{n!\cdot(n-\rho)} \cdot \left(1-\left(\dfrac{\rho}{n}\right)^m\right)} =$

$=1/\ ((\rho^0/0!)+(\rho^1/1!)+(\rho^2/2!)+(\rho^{2+1}/1*2(2-\rho))*[1-(\rho/2)^2]=1/\ (\ (2/3)/0!\ )+2/3+(\ (2/3)^2/(1*2)\ )+$
$+(\ (2/3)^3/\ 2(2-2/3)\ )\ [1-\ (\ (2/3)/2\ )]= 1/\ 1+2/3+2/9+1/9[1-1/9]=0,52$

д) Вероятность отказа в обслуживании: $P_{отк} = \dfrac{P_0 \cdot \rho^{n+m}}{n!\cdot n^m} =$ $((2/3)^4/1*2*2^2)*0,52$
$=(16/81)/8*0,52=0,013$

е) Вероятность обслуживания: $P_{обс} = 1- P_{отк}= 0,987$

ж) Абсолютная пропускная способность: $A = \lambda\, P_{обс}= 0,987*1/3=0,33$

з) Среднее число занятых каналов: $n_з = \rho*P_{обс}= 2/3*0,987=0,658$ Для каналов, занятых обслуживанием: $к_з = 0,658/2=0,329$.

и) Среднее число заявок в очереди: $\overline{L_{оч}} = \dfrac{\rho^{n+1}}{n\cdot n!} \cdot \dfrac{1-(\rho/n)^m(m+1-m\rho/n)}{(1-\rho/n)^2} \cdot P_0$

$\overline{L_{оч}} =(\ (2/3)^3/(2*2)\ )*\ 1-(\ (2/3)/2)^2\ )*(\ 2+1-2*((2/3)/2)\ )/\ (1-(2/3)/2)^2)\ *0.52$
$=(8/27)/4*\ *\ (1-1/9*7/3)\ /(4/9)= 2/27*((20/27)/(4/9))*0.52=2/27*5/3*0.52=0.14$

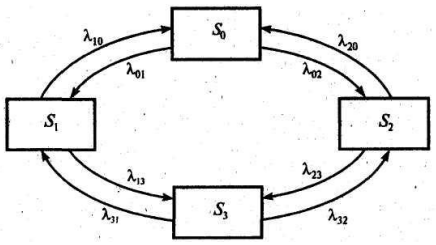
к) Среднее время ожидания обслуживания: $\overline{t_{оч}} = \dfrac{\overline{L_{оч}}}{\lambda} = 0.14/0.33=0.42$

л) Среднее число заявок в системе: $\bar{z} = L_{оч} + \overline{n_з} =0,14+0,66=0,8$

м) Среднее время пребывания в системе: $t_{смо}= Z/\ \lambda\ = 0,8/0,33=2,42$ или $t_{смо}= t_{ог}+ t_{обс}= 0,42+2=2,42$ мин

Задания для самостоятельной работы

1 вариант



Задача 1. Найти предельные вероятности для системы S, граф состояний которой приведен на рисунке, при $\lambda_{01}=1$, $\lambda_{02}=2$, $\lambda_{10}=3$, $\lambda_{13}=4$, $\lambda_{20}=1$, $\lambda_{23}=2$, $\lambda_{31}=3$, $\lambda_{32}=4$.

Задача 2. Дежурный по администрации города имеет 8 телефонов. Телефонные звонки поступают с интенсивностью 120 заявок в час. Средняя продолжительность разговора составляет 2мин. Определить показатели дежурного администратора как объекта СМО.

2 вариант

Задача 1. Найти предельные вероятности для системы S, граф состояний которой приведен на рисунке, при $\lambda_{01}=4$, $\lambda_{02}=3$, $\lambda_{10}=2$, $\lambda_{13}=1$, $\lambda_{20}=4$, $\lambda_{23}=3$, $\lambda_{31}=2$, $\lambda_{32}=1$.

Задача 2. АТС предприятия обеспечивает не более 5 переговоров, одновременно. Средняя продолжительность разговоров составляет 1 мин. На станцию поступает в среднем 10 вызовов в секунду. Определить характеристики АТС как объекта СМО.

3 вариант

Задача 1. Найти предельные вероятности для системы S, граф состояний которой приведен на рисунке, при $\lambda_{01}=1$, $\lambda_{02}=2$, $\lambda_{10}=1$, $\lambda_{13}=2$, $\lambda_{20}=1$, $\lambda_{23}=2$, $\lambda_{31}=3$, $\lambda_{32}=2$.

Задача 2. В морской порт поступает в среднем 6 сухогрузов в сутки. В порту имеются 3 крана, каждый из которых обслуживает 1 сухогруз в среднем за 8 часов. Краны работают круглосуточно. Определить характеристики работы порта как объекта СМО.

Контрольные вопросы:

1. Какой процесс называется марковским процессом?
2. Что такое «система уравнений Колмогорова», как она формируется?
3. Что понимается под системами массового обслуживания (СМО) и для чего они предназначены?
4. Что понимается под «потоком обслуживания заявок»?
5. Понятия входного и выходного потоков СМО.

Практическая работа №4. Построение прогнозов. Решение матричной игры методом итераций.

Цель занятия:

- приобретение навыков прогнозирования и планирования временных рядов.

Краткие теоретические сведения

ПОСТРОЕНИЕ ПРОГНОЗОВ

Регрессионный и корреляционный анализ позволяет установить и оценить зависимость изучаемой случайной величины Y от одной или нескольких других величин X, и делать прогнозы значений Y. Параметр Y, значение которого нужно предсказывать, является зависимой переменной. Параметр X, значения которого нам известны заранее и который влияет на значения Y, называется независимой переменной. Например, X – количество внесенных удобрений, Y – снимаемый урожай; X – величина затрат компании на рекламу своего товара, Y – объем продаж этого товара и т.д.

Корреляционная зависимость Y от X – это функциональная зависимость:

$$\bar{y}_x = f(x), \qquad (1)$$

где $\bar{y}_x$ – среднее арифметическое (условное среднее) всех возможных значений параметра Y, которое соответствует значение X=x.

Уравнение (1) называется уравнением регрессии Y на X, функция f(x) – регрессией Y на X, а ее график – линией регрессии Y на X.

Основная задача **регрессионного анализа** – установление формы корреляционной связи, то есть вида функции регрессии (линейная, квадратичная, показательная и т.д.).

Метод наименьших квадратов позволяет определить коэффициенты уравнения регрессии таким образом, чтобы точки, построенные по исходным данным (x_i, y_i), лежали как можно ближе к точкам линии регрессии. Формально это записывается как минимизация суммы квадратов отклонений (ошибок) функции регрессии и исходных точек.

$$S = \sum_{i=1}^{n} (y_i^p - y_i)^2 \to \min, \qquad (2)$$

где y_i^p – значение, вычисленное по уравнению регрессии;

$(y_i^p - y_i)$ - отклонение ε (ошибка, остаток) (рис. 1.);

n – количество пар исходных данных.

В регрессионном анализе предполагается, что математическое ожидание случайной величины ε равно нулю и ее дисперсия одинакова для всех наблюдаемых значений Y. Отсюда следует, что рассеяние данных возле линии регрессии должно быть одинаково при всех значениях параметра X.

В случае, показанном на рисунке 2 данные распределяются вдоль линии регрессии неравномерно, поэтому метод наименьших квадратов в этом случае неприменим.

Основная задача корреляционного анализа – оценка тесноты (силы) корреляционной связи. Теснота корреляционной зависимости Y от X оценивается по величине рассеяния значений параметра Y вокруг условного среднего $\bar{y}_x$. Большое рассеяние говорит о слабой зависимости Y от X, либо об ее отсутствии и, наоборот, малое рассеяние указывает на наличие достаточно сильной зависимости.

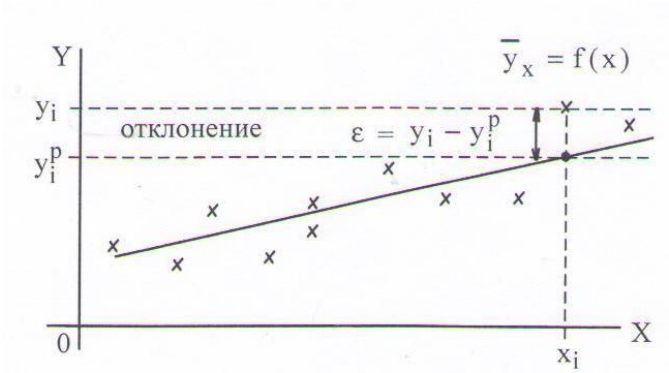


Рисунок 1 – Понятие отклонения ε для случая линейной регрессии

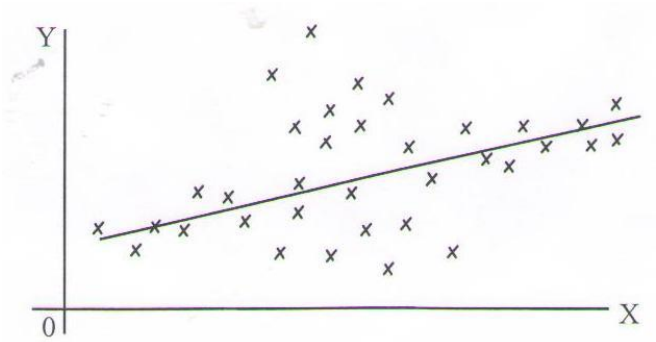


Рисунок 2 – Неравномерное распределение исходных точек вдоль линии регрессии

Коэффициент детерминации r^2 показывает, на сколько процентов (r^2*100%) найденная функция регрессии описывает связь между исходными значениями параметров Y и X:

$$r^2 = \frac{\sum_{i=1}^{n}(y_i^p - \bar{y})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}, \qquad (3)$$

где $(y_i^p - \bar{y})^2$ – объясненная вариация;

$(y_i - \bar{y})^2$ - общая вариация (рисунок 3).

Рисунок 3 – Графическая интерпретация коэффициента детерминации для случая линейной регрессии

Соответственно, величина (1- r^2)*100% показывает, сколько процентов вариации параметра Y обусловлены факторами, не включенными в регрессионную модель. При высоком ($r^2 \geq 75\%$) значение коэффициента детерминации можно делать прогноз y*=f(x*) для конкретного значения x*.

Для проведения регрессионного анализа и прогнозирования необходимо:

1) ***построить график*** исходных данных и попытаться зрительно, приближенно определить характер зависимости;

2) ***выбрать вид функции*** регрессии, которая может описывать связь исходных данных;

3) ***определить численные коэффициенты*** функции регрессии;

4) ***оценить силу*** найденной регрессионной зависимости на основе коэффициента детерминации r^2;

5) ***сделать прогноз*** (при $r^2 \geq 75\%$) или сделать вывод о невозможности прогнозирования с помощью найденной регрессионной зависимости. При этом не рекомендуется использовать модель регрессии для тех значений независимого параметра X, которые не принадлежат интервалу, заданному в исходных данных.

Линейная регрессия. Коэффициенты линейной регрессии $y=a_0+a_1x$ вычисляются по следующим формулам (все суммы берутся по n парам исходных данных):

$$a_1 = \frac{n(\sum y_i x_i) - \sum y_i \sum x_i}{n(\sum x_i^2) - (\sum x_i)^2}; \qquad (4)$$

$$a_0 = \frac{1}{n}(\sum y_i - a_1 \sum x_i).$$

Для удобства вычислений используют вспомогательную таблицу (таблица 1.), в которой рассчитываются необходимые суммы.

Таблица 1. Вспомогательная таблица для линейной функции

Заголовки данных	x_i	y_i	x_i^2	$x_i y_i$	y_i^p	$(y_i^p - \bar{y})^2$	$(y_i - \bar{y})^2$
Промежуточные значения							
	…	…	…	…	…	…	…
Сумма ($\sum_{i=1}^{n}$) по столбцу							

Пример 1. Некоторая фирма занимается поставками различных грузов на короткие расстояния внутри города. Перед менеджером стоит задача оценить стоимость таких услуг, зависящую от затраченного на поставку времени. В качестве наиболее важного фактора, влияющего на время поставки, менеджер выбрал пройденное расстояние. Были собраны исходные данные о десяти поставках (таблица 1.15).

Таблица 2. Исходные данные для примера 1

Расстояние, миль	3,5	2,4	4,9	4,2	3,0	1,3	1,0	3,0	1,5	4,1
Время, мин	16	13	19	18	12	11	8	14	9	16

Необходимо построить график исходных данных, определить по нему характер зависимости между расстоянием и затраченным временем, проанализировать применимость метода наименьших квадратов, построить уравнение регрессии, проанализировать силу регрессионной связи и сделать прогноз времени поездки на 2 мили.

Решение. На рисунке 4. построены исходные данные по десяти поездкам.

Помимо расстояния на время поставки влияют пробки на дорогах, время суток, дорожные работы, квалификация водителя, вид транспорта. Построенные точки не находятся точно на линии, что обусловлено описанными выше факторами. Но эти точки собраны вокруг прямой линии, поэтому можно предположить линейную связь между параметрами. Все исходные точки равномерно распределены вдоль предполагаемой прямой линии, что позволяет применить метод наименьших квадратов.

Вычислим суммы, необходимые для расчета коэффициентов линейной регрессии, коэффициента детерминации с помощью таблицы 3.

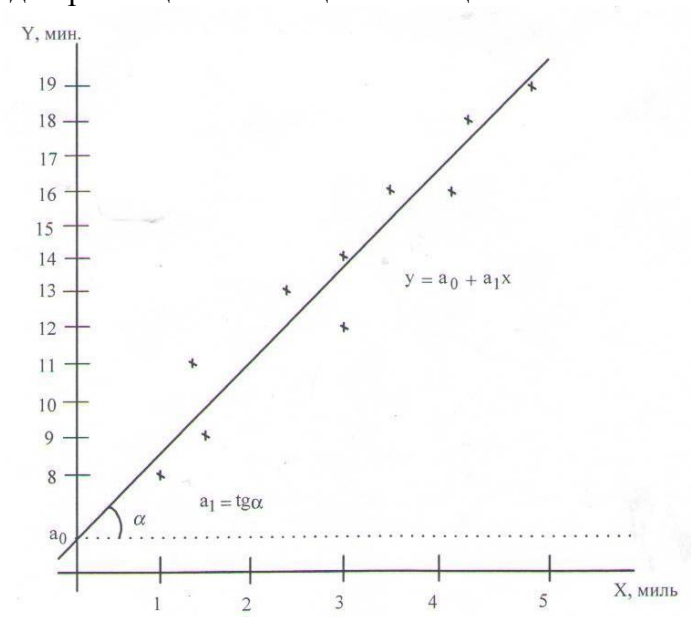


Рисунок 4 – График исходных данных для примера 1

Таблица 3. Вспомогательная таблица для примера 1

x_i	y_i	x_i^2	$x_i y_i$	y_i^p	$(y_i^p - \bar{y})^2$	$(y_i - \bar{y})^2$
3,5	16	12,25	56,00	15,223	2,634129	5,76
2,4	13	5,76	31,2	12,297	1,697809	0,36
4,9	19	24,01	93,1	18,947	28,59041	29,16
4,2	18	17,64	75,60	17,085	12,14523	19,36
3,0	12	9,00	36,00	13,893	0,085849	2,56
1,3	11	1,69	14,30	9,371	17,88444	6,76
1,0	8	1,00	8,00	8,573	25,27073	31,36
3,0	14	9,00	42,00	13,893	0,085849	0,16
1,5	9	2,25	13,50	9,903	13,66781	21,16
4,1	16	16,81	65,60	16,819	10,36169	5,76
∑=28,9	∑=136	∑=99,41	∑=435,30	-	112,4242	122,4

$$\bar{y} = \frac{\sum y_i}{n} = \frac{16+13+19+18+12+11+8+14+9+16}{10} = 13.6.$$

По формулам (4) вычислим коэффициенты линейной регрессии:

$$a_1 = \frac{10 \cdot 435{,}30 - 136 \cdot 28{,}9}{10 \cdot 99{,}41 - 835{,}21} = 2{,}660;$$

$$a_0 = 0{,}1 \cdot (136 - 2{,}660 \cdot 28{,}9) = 5{,}913.$$

Таким образом, искомая регрессионная зависимость имеет вид:

$$y^p = 5.913 + 2.660\,x \qquad (5)$$

Наклон линии регрессии a_1=2,66 минут на милю – это количество минут, приходящееся на одну милю расстояния. Координата точки пересечения прямой с осью Y a_0=5,913 минут – это время, которое не зависит от пройденного расстояния, а обуславливается всеми остальными возможными факторами, явно не учтенными при анализе.

По формуле (28) вычислим коэффициент детерминации:

$$r^2 = \frac{112{,}424}{122{,}400} = 0{,}918,$$

или 91,8%.

Таким образом, линейная модель объясняет 91,8% вариации времени доставки. Не объясняется 100% - 91,8% = 8,2% вариации времени поездки, которые обусловлены остальными факторами, влияющими на время поставки, но не включенными в линейную модель регрессии.

Поскольку коэффициент детерминации имеет достаточно высокое значение и расстояние 2 мили, для которого надо сделать прогноз, находится в пределах диапазона исходных данных (см. таблицу 2), то мы можем использовать полученное уравнение линейной регрессии (5) для прогнозирования:

y* (2 мили) = 5,913+2,660*2 = 11,2 минут.

При прогнозах на расстояния, не входивших в диапазон исходных данных, нельзя гарантировать справедливость модели (5). Это объясняется тем, что связь между временем и расстоянием может изменяться по мере увеличения расстояния. На время дальних перевозок могут влиять новые факторы такие, как использование скоростных шоссе, остановки на отдых, обед и т.п.

Приблизительным, но самым простым и наглядным способом проверки удовлетворительности регрессионной модели является графическое представление отклонений (рисунок 5).

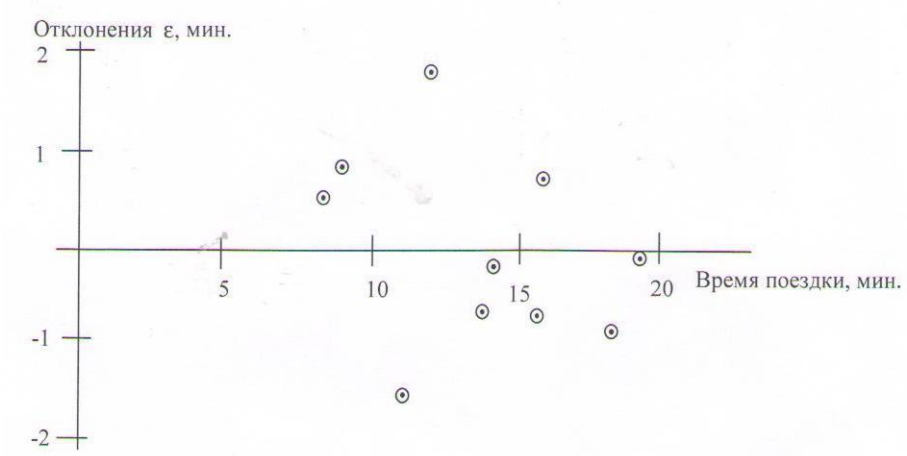


Рисунок 5 – График отклонений в примере 1.5

Отложим отклонений $(y_i^p - \overline{y})^2$ по оси Y, для каждого значения y_i. Если регрессионная модель близка к реальной зависимости, то отклонения будут носить случайный характер и их сумма будет близка к нулю. В рассмотренном примере $\sum_{i=1}^{n}(y_i^p - y_i) = 0.004$.

Нелинейная регрессия.

Рассмотрим наиболее простые случаи нелинейной регрессии: гиперболу, экспоненту и параболу. При нахождении коэффициентов гиперболы и экспоненты используют прием приведения нелинейной регрессионной зависимости к линейному виду. Это позволяет использовать для вычисления коэффициентов функции регрессии формулы (4).

Гипербола. При нахождении гиперболы $y = a_0 + \frac{a_1}{x}$ вводят новую переменную $z = \frac{1}{x}$, тогда уравнение гиперболы принимает линейный вид $y = a_0 + a_1 z$. После этого используют формулы (4) для нахождений линейной функции, но вместо значений x_i используются значения $z_i = \frac{1}{x_i}$

$$a_1 = \frac{n(\sum y_i z_i) - \sum y_i \sum z_i}{n(\sum z_i^2) - (\sum z_i)^2}; \qquad a_0 = \frac{1}{n}(\sum y_i - a_1 \sum z_i).$$

При проведении вычислений во вспомогательную таблицу вносятся соответствующие колонки.

Экспонента. Для приведения к линейному виду экспоненты $y = a_0 e^{a_1 x}$ проведем логарифмирование

$$\ln y = \ln(a_0 e^{a_1 x});$$

$$\ln y = \ln a_0 + \ln(e^{a_1 x});$$

$$\ln y = \ln a_0 + a_1 x.$$

Введем переменные $b_0 = \ln a_0$ и $b_1 = a_1$, тогда $\ln y = b_0 + b_1 x$, откуда следует, что можно применять формулы (29), в которых вместо значений y_i надо использовать $\ln y_i$

$$b_1 = \frac{n(\sum [\ln y_i] x_i) - \sum \ln y_i \sum x_i}{n(\sum x_i^2) - (\sum x_i)^2}; \qquad b_0 = \frac{1}{n}(\sum \ln y_i - b_1 \sum x_i).$$

При этом мы получим численные значения коэффициентов b_0 и b_1, от которых надо перейти к a_0 и a_1, используемых в модели экспоненты. Исходя из введенных обозначений и определения логарифма, получаем

$$a_0 = e^{b_0}, \quad a_1 = b_1.$$

Парабола. Для нахождения коэффициентов параболы $y = a_0 + a_1 x + a_2 x^2$ необходимо решить линейную систему из трех уравнений

$$\begin{cases} n \cdot a_0 + (\sum x_i) a_1 + (\sum x_i^2) a_2 = \sum y_i, \\ (\sum x_i) a_0 + (\sum x_i^2) a_1 + (\sum x_i^3) a_2 = \sum (y_i x_i), \\ (\sum x_i^2) a_0 + (\sum x_i^3) a_1 + (\sum x_i^4) a_2 = \sum (y_i x_i^2) \end{cases}$$

Оценка силы нелинейной регрессионной связи. Силы регрессионной связи для гиперболы и параболы определяется непосредственно по формуле (28). При вычислении коэффициента детерминации экспоненты все значения параметра Y (исходные, регрессионные, среднее) необходимо заменить на их логарифмы, например, y_i^p - на $\ln(y_i^p)$ и т.д.

Порядок выполнения задания

Пример 1. По данным о средних доходах на конечное потребление за десять лет, которые представлены в табл. 1, оцените наличие тренда и в случае положительного ответа постройте трендовую модель.

Таблица 1. Расходы на конечное потребление, тыс. у.е.

Год (t)	**Расходы (y_t)**
1-й	7
2-й	8

3-й	8
4-й	10
5-й	11
6-й	12
7-й	14
8-й	16
9-й	17
10-й	19

Решение

Для формального определения структуры временного ряда проводится автокорреляционный анализ уровней временного ряда. С этой целью рассчитываются коэффициенты автокорреляции уровней временного ряда. Расчет автокорреляционной функции можно осуществлять на компьютере средствами Excel: **Анализ** данных / **Корреляция.**

	A	B	C	D	E	F	G
1	t	y_t	y_{t-1}		t	y_t	y_{t-2}
2	1	7			1	7	
3	2	8	7		2	8	
4	3	8	8		3	8	7
5	4	10	8		4	10	8
6	5	11	10		5	11	8
7	6	12	11		6	12	10
8	7	14	12		7	14	11
9	8	16	14		8	16	12
10	9	17	16		9	17	14
11	10	19	17		10	19	16
12			19				17
13							19
14							

На рис. 2 представлены диалоговые окна расчета коэффициентов автокорреляции первого и второго порядков временного ряда y_t.

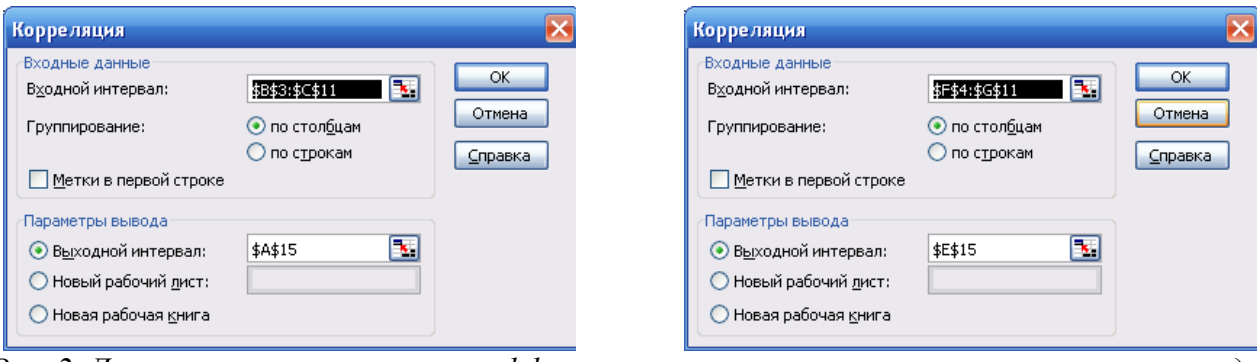


Рис. 2. Диалоговые окна расчета коэффициента автокорреляции первого и второго порядков ряда y_t

На рис. 3 представлены результаты расчетов.

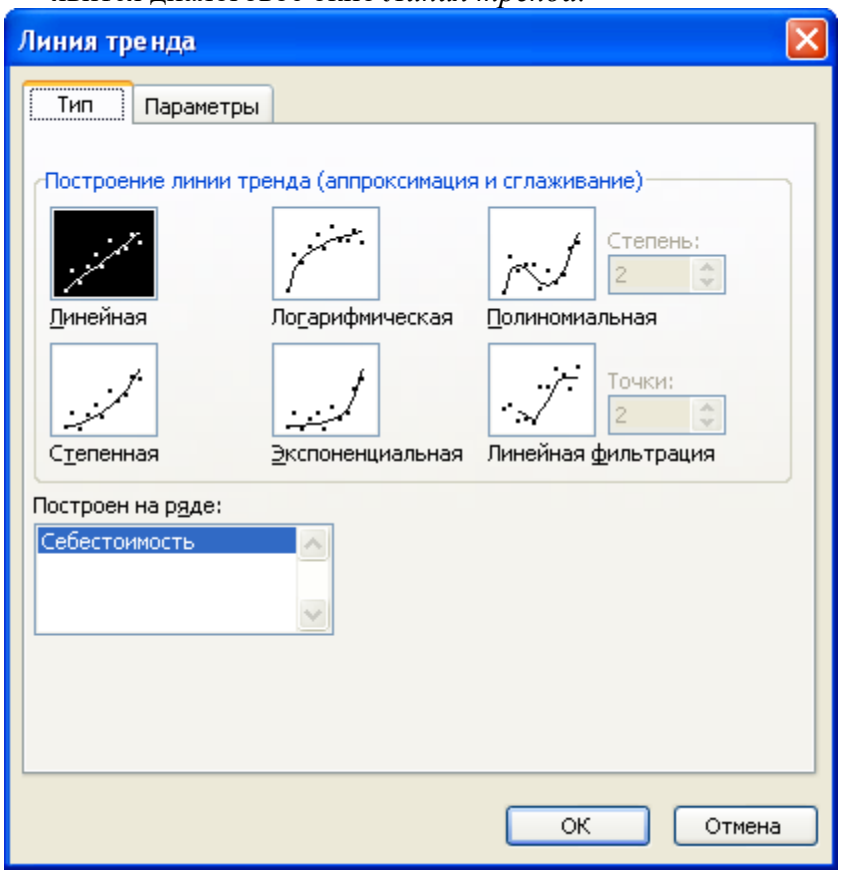
	Столбец1	***Столбец2***			***Столбец1***	***Столбец2***
Столбец 1	1			Столбец 1	1	
Столбец 2	0,986854	1		Столбец 2	0,981221	1

Рис. 3. Результаты расчетов коэффициентов автокорреляции.

Поскольку коэффициенты автокорреляции первых порядков (r_1, r_2) являются высокими, можно предположить наличие линейного тренда $T = a + bt$.

Определите уравнение линейного тренда. Для этого:

1. Постройте диаграмму по значениям временного ряда y_t. (тип *Точечная).*
2. Нажмите правой кнопкой мыши на одной из точек данных на диаграмме. В открывшемся меню необходимо выбрать команду *Добавить линию тренда.* На экране появится диалоговое окно *Линия тренда.*



3. Выберете тип регрессии. Например, *линейная.*
4. Переключитесь на вкладку *Параметры.* В разделе *Название аппроксимирующей (сглаженной) кривой* установите переключатель *автоматическое*, установите *отображение на диаграмме уравнения* и *величины достоверности аппроксимации.*

Аналогично постройте линию тренда, в качестве функции выбрать степенную.

Замечание: Уравнение линейного тренда можно получить, используя инструмент Регрессия Пакета анализа.

Пример 2. Провести сглаживание данных задачи 1 и выполнить прогноз на период t=11.

Скопируйте условие предыдущего примера на Лист 2. (Диапазон A1:B11)

С помощью пакета анализа рассчитайте значения скользящего среднего (инструмент **«скользящее среднее»).**

Заполните диалог следующим образом:

- входной интервал - **$B$2:$B$11,**
- интервал - 3,
- выходной интервал -**$C$3,**
- установите флажок **Вывод графика.**

Удалите значения равные **#Н/Д.** Результаты оформите в таблицу с тремя столбцами: **t, y_t, Прогноз (скользящ.)**

Скорректируйте построенный график таким образом, чтобы по оси X были значения t (от 1 до 11), по оси У – значения скользящего среднего. График фактических значений y_t должен быть построен для дней, начиная с 1-го по 10-ый, график прогнозируемых значений должен быть построен для дней начиная с 4-го по 11-ый.

С помощью пакета анализа рассчитайте значения экспоненциального сглаживания (инструмент **«экспоненциальное сглаживание»).**

Заполните диалог следующим образом:

- входной интервал - **$B$2:$B$11,**
- фактор затухания - **0,25,**
- выходной интервал - **$D$2,**
- установите флажок **Вывод графика.**

Удалите значения равные **#Н/Д.**

Продлите значения рассчитанного столбца для получения прогноза на 11-й день. Назовите столбец **Прогноз (экспоненц.).** Скорректируйте построенный график таким образом, чтобы по оси Х были значения дней (от 1 до 11), по оси У – спрогнозированные значения. График фактических значений y_t должен быть построен для дней, начиная с 1-го по 10-ый, график прогнозируемых значений должен быть построен для дней начиная с 2-го по 11-ый.

Сформулируйте экономический смысл полученных моделей. Объясните механизм прогнозирования в каждой их них.

РЕШЕНИЕ МАТРИЧНОЙ ИГРЫ МЕТОДОМ ИТЕРАЦИЙ

Заинтересованность игроков в тех или иных ситуациях проявляется в том, что каждому игроку P_i в каждой ситуации $\bar{S}$ приписывается число, выражающее степень удовлетворения его интересов в данной ситуации. Это число называется *выигрышем* игрока P_i и обозначается через $H_i(\bar{S})$, а само соответствие между множеством ситуаций и выигрышем игрока P_i называется *функцией выигрыша* (*платежной функцией*) этого игрока.

Таким образом, формальное определение игры сводится к заданию трех классов множеств:

а) множества игроков;

б) совокупности множеств стратегий каждого из игроков $\{S_i\}_{i \in I}$;

в) совокупности функций выигрыша каждого из игроков $\{H_i\}_{i \in I}$.

При этом предполагается, что функции выигрыша и множества стратегий игроков общеизвестны. В соответствии с этой информацией каждый из участников игры и организует свое поведение, стремясь обеспечить себе максимально возможный выигрыш при любых действиях партнеров.

Содержательный анализ игры в такой обобщенной постановке весьма затруднителен. Методы анализа игр значительно различаются в зависимости от числа игроков, от количества стратегий, от свойств платежных функций, а также от характера предварительной договоренности между игроками. Поэтому в дальнейшем ограничимся рассмотрением лишь одного, наиболее изученного класса игр, а именно класса *матричных игр*.

Матричная игра описывается следующим образом.

- В игре участвуют 2 игрока: допустим, игроки А и В.
- Каждый из игроков располагает конечным набором стратегий: $A_1,\ldots,A_m$ и $B_1,\ldots,B_n$ - возможные стратегии игроков А и В (в этом случае говорят, что игра имеет размерность mxn).
- Значения функций выигрыша H_A и H_B игроков в каждой ситуации (A_i,B_j) равны по величине и противоположны по знаку, то есть

$H_A(A_i,B_j)=-H_B(A_i,B_j)= a_{ij}$

для всех $i=1,\ldots,m$, $j=1,\ldots,n$ (выигрыш одного из игроков равен проигрышу другого).

Очевидно, задание такой игры эквивалентно заданию всех значений функции выигрыша одного из игроков (например, игрока А) в виде так называемой *платежной матрицы* или матрицы игры:

$$H = \begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \ldots & \ldots & \ldots & \ldots \\ a_{m1} & a_{m2} & \ldots & a_{mn} \end{pmatrix} \quad (2.1)$$

Строки этой матрицы соответствуют стратегиям игрока А, а столбцы - стратегиям игрока В; элементы a_{ij} задают выигрыш игрока А в ситуации, когда А выбирает стратегию A_i, а В – стратегию B_j.

Итерационный метод Брауна – Робинсон.

Основная идея метода состоит в следующем.

Разыгрывается «мысленный» эксперимент, в котором игроки А и В поочередно применяют друг против друга свои стратегии, стремясь выиграть побольше. При этом каждый игрок при выборе очередной стратегии ориентируется не на оптимальный выигрыш относительно последней стратегии противника, а на оптимальный «накопленный» выигрыш за все предыдущие ходы. Приближенные оптимальные стратегии игроков определяются относительными частотами применения ими чистых стратегий.

Рассмотрим реализацию этого метода на примере.

Пример 1. Найти приближенное решение игры, заданной матрицей

$$H = \begin{pmatrix} 3 & 6 & 8 \\ 9 & 4 & 2 \\ 7 & 5 & 4 \end{pmatrix}.$$

Игра не имеет доминируемых стратегий и поэтому не может быть сведена к игре меньшей размерности. Нижняя цена игры α=4, A_3 - соответствующая максиминная стратегия игрока А; верхняя цена игры β=6, B_2 – соответствующая минимаксная стратегия игрока В. Оформим расчеты методом Брауна в виде таблицы.

k	$\boldsymbol{A_i}$	B_1	B_2	B_3	$\boldsymbol{B_j}$	A_1	A_2	A_3	v_*	v^*	v_S
									0	*1*	*2*
1	A_3	7	5	4*	B_3	8*	2	4	.00	.00	.00
2	A_1	10*	11	12	B_1	11*	11	11	.00	.50	.25
3	A_1	13*	17	20	B_1	14	20*	18	.33	.67	.55
4	A_2	22	21*	22	B_2	20	24*	23	.25	.00	.63
5	A_2	31	25	24*	B_3	28*	26	27	.80	.60	.20
6	A_1	34	31*	32	B_2	34*	30	32	.17	.67	.32
7	A_1	37*	37	40	B_1	37	39*	39	.29	.86	.58
8	A_2	46	41*	42	B_2	43	43	44*	.13	.50	.31
9	A_3	53	46*	46	B_2	49*	47	49	.11	.37	.24
10	A_1	56	52*	54	B_2	55*	51	54	.20	.50	.35
…	…	…	…	…	…	…	…	…	…	…	…

Здесь:

- k – номер партии (пары выборов игроками своих стратегий);
- A_i – стратегия, выбранная игроком А в этой партии;
- в следующих трех столбцах – «накопленный выигрыш» за первые k партий при тех стратегиях, которые применяли игроки в предыдущих партиях и при стратегиях B_1, B_2, B_3 в данной партии (получается прибавлением элементов соответствующей строки к тому, что было строкой выше);

- из этих накопленных выигрышей выделяется минимальный (если их несколько, то – любой из них), выделенное число определяет ответный выбор игрока В в данной партии – он выбирает ту стратегию, которая соответствует выделенному числу; таким образом, определяется оптимальная в данной партии стратегия B_j игрока В;
- в следующих трех столбцах дается накопленный выигрыш за k партий соответственно при стратегиях A_1, A_2, A_3 игрока А (получается прибавлением столбца B_j к тому, что было строкой выше); из этих значений выделяется максимальное; оно определяет выбор стратегии игрока А в следующей партии;
- v_* - нижняя оценка цены, равная минимальному накопленному выигрышу, деленному на k;
- v^* - верхняя оценка цены игры, равная максимальному накопленному выигрышу, деленному на k;
- v_S – среднее арифметическое v_* и v^*.

Рассмотрим подробно несколько шагов методом Брауна в данной игре.

В *1-й партии* игрок А может выбрать любую из своих чистых стратегий, но лучше, если это будет максиминная стратегия A_3 (вносим это выражение во 2-й столбец). Этой стратегии соответствует 3-я строка матрицы выигрышей (7 5 4), соответствующих стратегиям B_1, B_2, B_3 игрока В (заносим их в 3-й, 4-й и 5-й столбцы). Среди этих чисел выделяем значком "$*$" минимальное. Оно соответствует наиболее выгодной для игрока В стратегии B_3 в этой партии. Этой стратегии соответствует 3-й столбец платежной матрицы $(8\ 2\ 4)^T$. Заносим эти значения в 7-й, 8-й и 9-й столбцы, выделяя среди них значком * максимальное, соответствующее наибольшему выигрышу игрока А.

Поэтому в начале *2-й партии* игрок А выбирает стратегию A_1, которой соответствует 1-я строка (3 6 8) матрицы H. «Накопленный выигрыш» при этой и предыдущей стратегиях равен (3 6 8) + (7 5 4) = (10 11 12). Именно эти значения и заносим в 3-й, 4-й и 5-й столбцы. Минимальному из них значению соответствует стратегия B_1, т. е. 1-й столбец $(3\ 9\ 7)^T$. С учетом предыстории «накопленный выигрыш» игрока А равен $(3\ 9\ 7)^T + (8\ 2\ 4)^T = (11\ 11\ 11)^T$. Заполняем этими значениями 7-й, 8-й и 9-й столбцы таблицы и т. д.

В таблице приведены первые 10 шагов методом Брауна-Робинсон. В результате игрок А применял 5 раз стратегию A_1, 3 раза - стратегию A_2, 2 раза – стратегию A_3; игрок В – 3 раза стратегию B_1, 5 раз – стратегию B_2, 2 - раза стратегию B_3. Поэтому оптимальные стратегии игроков, приближенно вычисленные по относительным частотам использования своих чистых стратегий, имеют вид: S_A^{π}=(0.5, 0.3,0.2), S_B^{π}=(0.3,0.5,0.2).

Нижняя и верхняя оценки цены игры равны соответственно v_*=5.2 и v^*=5.5 (вычисляются делением соответственно минимального и максимального накопленных выигрышей (52 и 55) на количество сыгранных партий (10)). Приближенная цена игры v_S^{π}=(5.2+5.5)/2=5.35.

После 20-ти шагов методом Брауна аналогичные результаты выглядят следующим образом: приближенные оптимальные стратегии S_A^{π}=(0.4,0.1,0.5), S_B^{π}=(0.25,0.6,0.15), приближенная цена игры v_S^{π}=5.275. При этом точное решение игры, которое может быть получено методом сведения игры к задаче линейного программирования, имеет вид: S_A*=(0.4,0,0.6), S_B*=(0.2,0.8,0), v_S=5.4.

Исходя из рассмотренного примера и некоторых теоретических выкладок, которые мы опускаем, можно сделать два вывода:

1) Метод Брауна позволяет сравнительно просто находить приближенные решения матричных игр, причем трудоемкость метода с увеличением размерности игры возрастает незначительно (в отличие от метода сведения игры к задаче линейного программирования).

2) Сходимость приближенных решений, рассчитанных методом Брауна, к точному решению происходит довольно медленно.

Практические задания

Задание 1. По данным о выпуске продукции за десять лет, которые представлены в табл. 7, оцените наличие тренда и в случае положительного ответа постройте трендовую модель.

	Годы выпуска продукции (t)									
№ варианта	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**

1.	13,5	12,7	12	11,9	11,5	11,2	10,8	10,7	10,6	10,5
2.	251	249	248	246	242	239	235	230	228	225
3.	0,91	0,87	0,85	0,82	0,79	0,75	0,7	0,66	0,62	0,6

Задание 2. По данным задания 1 проведите сглаживание данных и выполните прогноз на период t=11.

Задание 3.

Порядок выполнения задания

Пример 1 Решение матричной игры 2*2 аналитическим методом.

Создайте в MS Excel модуль для решения матричных игр 2*2 аналитическим методом. Вы можете размещать ячейки и формулы удобным для вас способом, так чтобы в созданном модуле было легко ориентироваться человеку, умеющему решать матричные игры 2*2. Пример модуля изображен на рисунке ниже.

	A	B	C	D	E	F	G	H
1			Задание	6				
2								
3			0	-1	-1	бетта	0	нет седловой точки
4			-3	0	-3	альфа	-1	
5			0	0				
6			первый игрок					
7			p1	0,75		q1	0,250	
8			p2	0,25		q2	0,750	
9			v	-0,75				
10			Ответ: V = -0,75 P = (0,75; 0,25) Q = (0,25; 0,75)					

*Рисунок 1 – Модуль для решения матричных игр 2*2 аналитическим методом*

В данном модуле в область, отмеченную желтой заливкой, вводятся элементы матрицы, после чего автоматически формируется решение игры, заданное с помощью формул в MS Excel. Рассмотрим более подробно шаги по созданию данного модуля.

Шаг 1. Проверка наличия седловой точки.

- Ячейки E3 и E4 – это минимальные элементы по строкам матрицы. Здесь можно воспользоваться функцией =МИН(…).
- Ячейки C5 и D5 – это максимальные элементы по столбцам матрицы. Соответственно следует воспользоваться функцией = МАКС(…).
- Ячейка G4 – это α (альфа) – нижняя цена игры или максимин. Его можно найти двумя способами: как максимальный элемент из E3 и E4, как максимальный из минимальных элементов по каждой из строк матрицы =МАКС(МИН(C3:D3);МИН(C4:D4)).
- Ячейка G3 – это β – верхняя цена игры. Находится аналогичным образом.
- Ячейка H3 – решение о наличие седловой точки. Здесь следует использовать функцию = ЕСЛИ(). По следующей схеме: в случае равенства верхней и нижней цены игры «седловая точка есть», а иначе «нет седловой точки».

Шаг 2. Получение решения игры.

Ячейка D7 – p1 – вероятность с которой первый игрок выбирают свою первую чистую стратегию. Способ расчета p1 зависит от того есть или нет седловая точка. Если седловой точки нет, то данная вероятность рассчитывается по формуле. Если седловая точка есть, то эта p1 принимает значение 0 или 1 в зависимости от того является ли A1 чистой стратегией первого игрока (p1 = 1) или нет (p1 = 0).

Формулу в ячейке D7 можно создать по следующей схеме с использованием двух функций =ЕСЛИ(…).

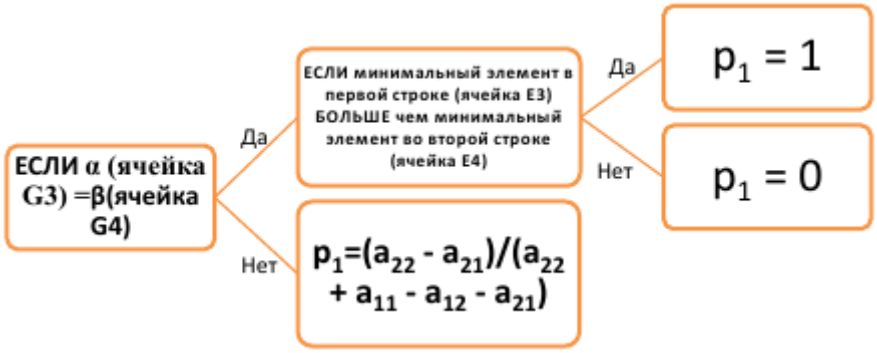


Рисунок 2 – Схема для формулы в ячейке D7

Ячейка D8 – вероятность с которой первый игрок выберет свою вторую чистую стратегия (p2) всегда равна (1 – p1), так как p1 и p2 – это вероятности и в сумме дают единицу.

Ячейки G7 и G8 вероятности, с которыми второй игрок выбирает свои чистые стратегии, находятся аналогичным образом.

Ячейка D9 – цена игры V также зависит от того есть седловая точка или нет. Если седловая точка есть (α=β), то V=α. Если седловой точки нет, то цена игры находится аналитически, по формуле:

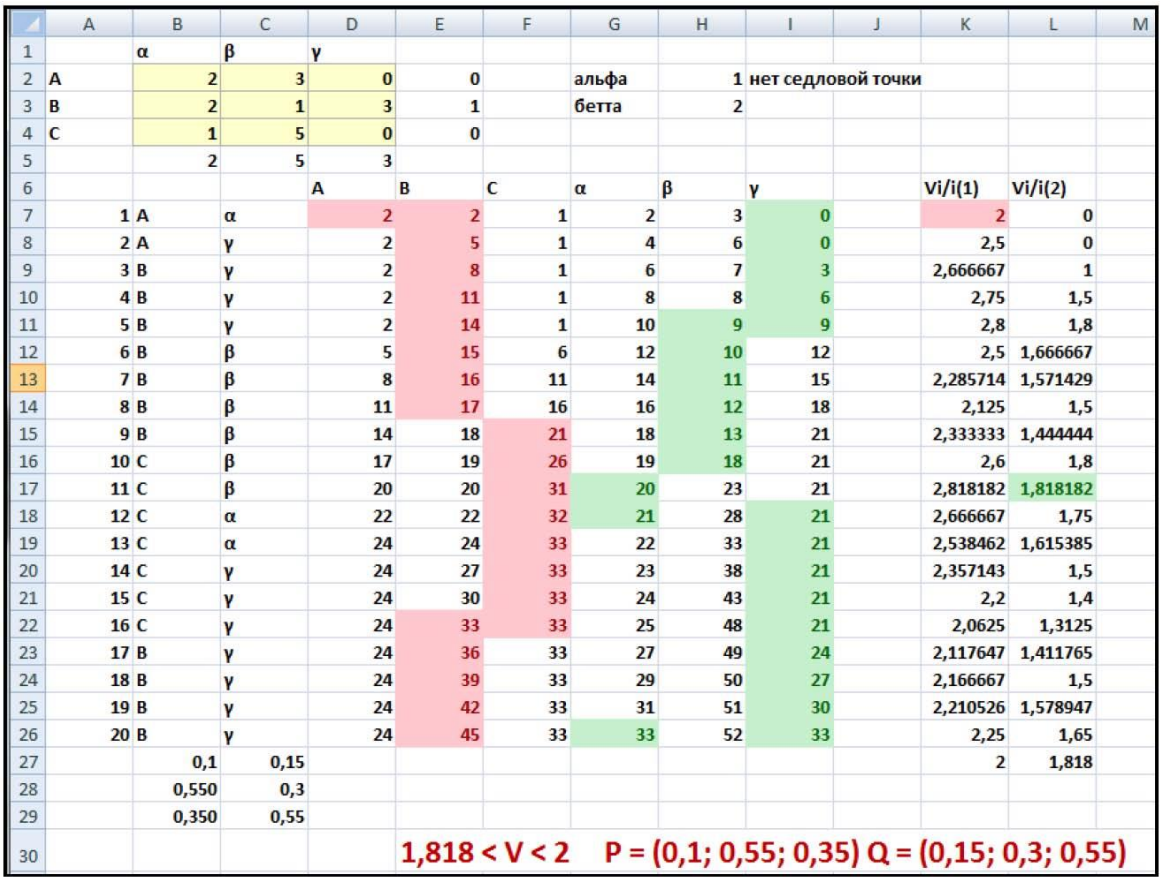
$$\nu = \frac{a_{11}a_{22} - a_{12}a_{21}}{a_{11} + a_{22} - a_{21} - a_{12}}$$

Шаг 3. Формирование ответа.

Чтобы записать ответ в одно строку нужно воспользоваться функцией = СЦЕПИТЬ(…). Поскольку значения вероятностей и цены игры могут содержать сколько угодно знаков после запятой и ответ может в этом случае может получиться слишком длинным лучше округлить значение вероятностей и цены игры. Для это нужно воспользоваться функцией =ОКРУГЛ(Ячейка с числом; количество знаков после замятой). Достаточно оставить три знака после запятой для каждого числа в ответе.

Пример 2. Решение матричной игры 3*3 методом Робинсона-Брауна.

Создайте в MS Excel модуль для решения матричных игр 3*3 методом Робисона-Брауна. Вы можете размещать ячейки и формулы удобным для вас способом, так чтобы в созданном модуле было легко ориентироваться человеку, умеющему решать матричные игры 3*3. Пример модуля изображен на рисунке ниже.



	A	B	C	D	E	F	G	H	I	J	K	L	M
1		α	β	γ									
2	A	2	3	0	0		альфа	1	нет седловой точки				
3	B	2	1	3	1		бетта	2					
4	C	1	5	0	0								
5		2	5	3									
6				A	B	C	α	β	γ		Vi/i(1)	Vi/i(2)	
7	1	A	α	2	2	1	2	3	0		2	0	
8	2	A	γ	2	5	1	4	6	0		2,5	0	
9	3	B	γ	2	8	1	6	7	3		2,666667	1	
10	4	B	γ	2	11	1	8	8	6		2,75	1,5	
11	5	B	γ	2	14	1	10	9	9		2,8	1,8	
12	6	B	β	5	15	6	12	10	12		2,5	1,666667	
13	7	B	β	8	16	11	14	11	15		2,285714	1,571429	
14	8	B	β	11	17	16	16	12	18		2,125	1,5	
15	9	B	β	14	18	21	18	13	21		2,333333	1,444444	
16	10	C	β	17	19	26	19	18	21		2,6	1,8	
17	11	C	β	20	20	31	20	23	21		2,818182	1,818182	
18	12	C	α	22	22	32	21	28	21		2,666667	1,75	
19	13	C	α	24	24	33	22	33	21		2,538462	1,615385	
20	14	C	γ	24	27	33	23	38	21		2,357143	1,5	
21	15	C	γ	24	30	33	24	43	21		2,2	1,4	
22	16	C	γ	24	33	33	25	48	21		2,0625	1,3125	
23	17	B	γ	24	36	33	27	49	24		2,117647	1,411765	
24	18	B	γ	24	39	33	29	50	27		2,166667	1,5	
25	19	B	γ	24	42	33	31	51	30		2,210526	1,578947	
26	20	B	γ	24	45	33	33	52	33		2,25	1,65	
27		0,1	0,15								2	1,818	
28		0,550	0,3										
29		0,350	0,55										
30					1,818 < V < 2		P = (0,1; 0,55; 0,35) Q = (0,15; 0,3; 0,55)						

В данном модуле в область, отмеченную желтой заливкой, вводятся элементы матрицы, после чего автоматически формируется решение игры, заданное с помощью формул в MS Excel.

Рассмотрим создание данного модуля более подробно.

Сначала необходимо организовать проверку наличия седловой точки. Это делается как в примере 1.

Далее перейдем к заполнению основной таблицы.

В ячейки A7:A26 необходимо введи числа от 1 до 20 – это номера партий игры.

В ячейку B7 ввести символ A, а в ячейку C7 – символ α – это выбор игроков в первой партии игры, который всегда одинаковый.

Выигрыши первого игрока в первой партии соответствуют стратегии α, которую выбрал второй игрок и равны числам стоящим соответственно в ячейках B2, B3 и B4.

Обратите внимание, что в ячейки B7:F7 необходимо ввести не числа, а ссылки на ячейки в которых находятся эти числа (элементы первого столбца матрицы), чтобы модули работал для любых матриц.

Выигрыши второго игрока в первой партии задаются аналогичным образом – это ссылки на элементы первой строки матрицы.

Выбор стратегии первого игрока для второй и последующих партий (ячейки B8:B26) – это стратегия (A, B или C), которой соответствует максимальный выигрыш в предыдущей партии игры. Данное условие задается с помощью функции =ЕСЛИ(…):

=ЕСЛИ(МАКС(D7:F7)=D7;$A$2; ЕСЛИ(МАКС(D7:F7)=E7;$A$3;$A$4))

Обратите внимание, ссылки на ячейки A2, A3 и A4 являются абсолютными, это необходимо для того, чтобы при копировании формулы из ячейки B8 в ячейки B9:B26 ссылки на эти ячейки не изменялись. В самих ячейках A2:A4 находятся символы A, B и C. С одной стороны эти символы можно было просто ввести с клавиатуры, не создавая ссылки на ячейки, но в этом случае могут возникать ошибки из-за схожего написания латинских и русских букв.

Выбор стратегии второго игрока для второй и последующих партий осуществляется аналогично, но второй игрок выбирает не максимальный выигрыш, а минимальный проигрыш. Формулу для ячеек C8:C26 создайте самостоятельно.

Расчет выигрыша первого игрока для второй и последующих партий. Сумма выигрыша первого игрока во второй партии зависит от того какую стратегию выбрал второй игрок.

Если второй игрок выбрал на предыдущем шаге стратегию α (т.е. α находится в ячейке C8), то в ячейках D8, E8 и F8 будет соответственно сумма чисел содержащихся в ячейках D7, E7 и F7 (выигрыши первого игрока в предыдущей партии) и в ячейках B2, B3 и B4 (соответствующих стратегии второго игрока α).

Если второй игрок выбрал на предыдущем шаге стратегию β (т.е. β находится в ячейке C8), то в ячейках D8, E8 и F8 будет соответственно сумма чисел содержащихся в ячейках D7, E7 и F7(выигрыши первого игрока в предыдущей партии) и в ячейках C2, C3 и C4 (соответствующих стратегии второго игрока β).

И наконец, если второй игрок выбрал на предыдущем шаге стратегию ϒ (т.е. ϒ находится в ячейке C8), то в ячейках D8, E8 и F8 будет соответственно сумма чисел содержащихся в ячейках D7, E7 и F7(выигрыши первого игрока в предыдущей партии) и в ячейках D2, D3 и D4 (соответствующих стратегии второго игрока ϒ).

Это можно реализовать с помощью следующей формулы для ячейки D8: =ЕСЛИ(C8=$B$1;D7+$B$2;ЕСЛИ(C8=$C$1;D7+$C$2;D7+$D$2))

Обратите внимание на использование абсолютных ссылок на ячейки. Формулы для ячеек E8, F8, G8, H8 и I8 создайте самостоятельно аналогичным образом. Не забудьте, что выигрыш второго игрока зависит от того какую стратегию выбрал первый игрок.

Скопируйте полученные формулы до 26 строки.

Для наглядности можно также организовать подсветку ячеек с максимальными и минимальными выигрышами. Чтобы в каждой строке подствечивался максимальный элемент нужно:

Шаг 1. Выделить ячейки D7:F7

Шаг 2. Нажать кнопку «Условное форматирование» выбрать пункт «Правила отбора первых и последних значений», затем пункт «10 первых элементов» и исправить число 10 на 1. Таким образом, будет подсвечен максимальный выигрыш первого игрока в первой партии.

Шаг 3. Еще раз выделить ячейки D7:F7 и нажать дважды кнопку «формат по образцу». Она находится на вкладке «Главная» и выглядит как кисть для краски . При это кнопка станет оранжевой.

Шаг 4. Выделить ячейки D8:F8. При этом на них распространится заданное форматирование.

Шаг 5. Выделить ячейки D9:F9. При этом на них распространится заданное форматирование, т.к. кнопка «форматирование по образцу остается нажатой.

И так далее до 26 строки.

Подсветку ячеек второго игрока создайте самостоятельно.

Далее рассчитаем средние выигрыши для каждой из партий. Для первого игрока в ячейке K7 будет находиться число, которое является максимальным из D7, E7 и F7, деленное на номер партии (он находится в ячейке A7). Создайте формулу, реализующую данный алгоритм для первого и второго игрока (используйте функцию +ЕСЛИ(…)) и скопируйте ее до 26 строки.

В ячейках K27 и L27 соответственно рассчитываются верхняя и нижняя цена игры как минимальный (K27) и максимальный (L27) элементы в столбцах. Подсветку максимального и минимального элементов в столбцах организуйте самостоятельно с помощью условного форматирования.

Осталось найти смешанные стратегии игроков.

Вероятность, с которой первый игрок выбирает свою первую чистую стратегию, (ячейка B27) находится с помощью функции =СЧЕТЕСЛИ(…):

=СЧЁТЕСЛИ(B7:B26;D6)/20, которая считает количество ячеек, удовлетворяющих заданному условию. В данном случае тех, в которых находится символ A.

Аналогично найдите вероятность выбора первым игроков второй стратегии.

Вероятность выбора первым игроком третьей стратегии можно найти, если из единицы вычесть вероятность, с которыми он выбирает сои первую и втору. Стратегии. Для второго смешанная стратегия находится таким же образом в ячейках C27:C29.

Ответ можно записать в ячейке E30 с помощью функции =СЦЕПИТЬ.

Задания для самостоятельной работы

Задача 1. Решить матричную игру 2х2 аналитическим методом.

Задача 2. Решить матричную игру 3х3 итерационным методом Робинсона-Брауна.

1 вариант	3 2 1 5	1 0 0 0 3 2 0 7 1
2 вариант	0 5 5 2	1 8 6 9 5 8 1 0 7
3 вариант	4 3 2 6	4 1 1 7 0 5 4 8 8

Контрольные вопросы:

1. Для чего применяются методы регрессивного и корреляционного анализа?
2. Чем корреляционная зависимость отличается от функциональной?
3. В чем состоит задача регрессионного анализа?
4. В чем состоит задача корреляционного анализа?
5. Как провести регрессионный анализ?
6. Какие виды регрессии существуют?
7. Какие игры называются матричными играми?
8. Что называется платежной матрицей игры?
9. Что называется седловой точкой?
10. Как найти седловую точку?
11. Что называется игрой с нулевой суммой?

Лабораторная работа №24. Моделирование прогноза. Выбор оптимального решения с помощью дерева решений.

Цель занятия: приобретение навыков прогнозирования и планирования временных рядов.

Краткие теоретические сведения

Метод скользящей средней - метод изучения в рядах динамики основной тенденции развития явления.

Суть метода скользящей средней состоит в том, что вычисляется средний уровень из определенного числа первых по порядку уровней ряда, затем средний уровень из того же числа уровней, начиная со второго, далее начиная с третьего и т. д. Таким образом, при расчетах среднего уровня как бы «скользят» по ряду динамики от его начала к концу, каждый раз отбрасывая один уровень в начале и добавляя один следующий.

Средняя из нечетного числа уровней относится к середине интервала. Если интервал сглаживания четный, то отнесение средней к определенному времени невозможно, она относится к середине между датами. Для того чтобы правильно отнести среднюю из четного числа уровней, применяется центрирование, т. е. нахождение средней из средней, которую относят уже к определенной дате.

Метод аналитического выравнивания

Уравнение прямой при аналитическом выравнивании ряда динамики имеет следующий вид: $\bar{y}_t = a_0 + a_1 t,$

где $\bar{y}_t$ *- выровненный (средний) уровень динамического ряда;* a_0, a_1 *- параметры искомой прямой; t- обозначение времени.*

Способ наименьших квадратов дает систему двух нормальных уравнений для нахождения параметров a_0 и a_1:

$$\begin{cases} a_0 n + a_1 \sum t = \sum y; \\ a_0 \sum t + a_1 \sum t^2 = \sum ty, \end{cases}$$

где y - исходный уровень ряда динамики; n - число членов ряда.

Система уравнений упрощается, если значения t подобрать так, чтобы их сумма равнялась нулю, т. е. начало времени перенести в середину рассматриваемого периода.

Если $\sum t = 0,$ то $a_0 = \frac{\sum y}{n}, \quad a_1 = \frac{\sum ty}{\sum t^2}.$

Исследование динамики соц.-экон. явлений и установление основной тенденции развития дают основание для прогнозирования (экстраполяции) - определения будущих размеров уровня экономического явления. Используют следующие методы экстраполяции:

- средний абсолютный прирост с/показатель, исчисляемый для выражения средней скорости роста (снижения) соц.-эк. процесса. Определяется по формуле: $\bar{\Delta} y = \frac{\sum \Delta_i}{n-1},$ или $\bar{\Delta} y = \frac{y_n - y_0}{n-1}.$
- средний темп роста;
- экстраполяцию на основе выравнивания по какой-либо аналитической формуле. Метод аналитического выравнивания-метод исследования динамики соц.-экон. явлений, позволяющий установить основные тенденции их развития.

Расчет индекса сезонности

Индексы сезонности (*I*$_s$) - спец. показатели, используемые при изучении сезонных колебаний. Рассчитываются по формуле:

$$I_s = \frac{\bar{y}_i}{\bar{y}} \cdot 100\%.$$

где $\bar{y}_i$ — средняя для каждого месяца за изучаемый период;

$\bar{y}$ — общий средний месячный уровень за изучаемый **период.**

Покажем расчет индекса сезонности на примере. Имеются следующие данные по строительной фирме об объеме выполненных работ по месяцам 2001–2003 гг. по сметной стоимости.

(млн руб.)

Месяц	Год			В среднем за три года $\bar{y}_i$	Индекс сезонности $(\bar{y}_i/\bar{y})100\%$
	2001	2002	2003		
Январь	1,7	1,9	2,0	1,9	67,9
Февраль	1,8	2,1	2,3	2,1	75,0
Март	2,1	2,4	2,8	2,4	85,7
Апрель	2,4	2,6	2,9	2,6	92,9
Май	2,6	2,8	3,0	2,8	100,0
Июнь	2,8	3,0	3,2	3,0	107,1
Июль	3,2	3,3	3,4	3,3	117,9
Август	3,4	3,5	3,4	3,4	121,4
Сентябрь	3,2	3,3	3,0	3,2	114,3
Октябрь	2,9	3,1	3,2	3,1	110,7
Ноябрь	2,7	2,7	3,1	2,8	100,0
Декабрь	2,6	2,5	2,7	2,6	92,9
Средний уровень ряда $(\bar{y})$	2,6	2,8	2,9	2,8	100,0

Для получения $\bar{y}_i$ проведем осреднение уровней одноименных периодов по формуле простой средней арифметической:

январь — $\bar{y} = \frac{y_{\text{янв.2001}} + y_{\text{янв.2002}} + y_{\text{янв.2003}}}{3} = \frac{1,7+1,9+2,0}{3} = 1,9(\text{млн руб.});$ …декабрь — $\bar{y} = \frac{y_{\text{дек.2001}} + y_{\text{дек.2002}} + y_{\text{дек.2003}}}{3} = \frac{2,6+2,5+2,7}{3} = 2,6(\text{млн руб.})$

Осредненные значения уровней ряда $\bar{y}_i$ для каждого месяца годового цикла представлены в таблице данного примера.

Далее по исчисленным месячным средним уровням $\bar{y}_i$ определяем общий средний уровень $(\bar{y})$:

$$\bar{y} = \frac{\sum \bar{y}_i}{n} = \frac{33,2}{12} = 2,8(\text{млн руб.}),$$

где n — число месяцев.

Значение общего среднего уровня можно вычислить также по итоговым данным за отдельные годы:

$$\bar{y} = \frac{\sum(\bar{y}_i)}{n} = \frac{8,3}{3} = 2,8(\text{млн руб.}),$$

где n — число лет;

$\sum(\bar{y}_i)$ — сумма среднегодовых уровней ряда динамики.

В завершение определим индексы сезонности по месяцам года по формуле: $I_s = \frac{\bar{y}_i}{\bar{y}} \cdot 100\%:$

январь — $I_{s_1} = \frac{1,9}{2,8} \cdot 100\% = 67,9\%;$ февраль — $I_{s_2} = \frac{2,1}{2,8} \cdot 100\% = 75\%$ и т. д.

Рассчитанные индексы сезонности представлены в таблице примера.

Следовательно, мин. объем выполненных работ строительная фирма имела в январе, а максимальный — в августе.

Для ряда внутригодовой динамики, в которой основная тенденция роста незначительна, изучение сезонности основано на методе постоянной средней, являющейся средней из всех рассматриваемых уровней. Самый простой способ заключается в следующем: для каждого года рассчитывается средний уровень, а затем с ним сопоставляется (в процентах) уровень каждого месяца.

Однако помесячные данные одного года из-за элемента случайности могут быть ненадежными для выявления закономерности колебаний. Поэтому на практике используются помесячные данные за ряд лет (обычно не менее трех лет). Тогда для каждого месяца рассчитывается средняя

величина уровня за три года, затем определяются среднемесячный уровень для всего ряда и отношение средних для каждого месяца к общему среднемесячному уровню ряда (в процентах).

Порядок выполнения задания

Пример 1. По данным табл. 1 исследуйте структуру временного ряда по квартальным данным потребления электроэнергии за 2001 – 2004 гг. Оцените уровень и структуру потребления электроэнергии в 2005 г.

Таблица 2. Исходные данные

Период	**Потребление электроэнергии, млрд. кВт - ч**
I кв. 2001 г.	6,0
II кв. 2001 г.	4,4
III кв. 2001 г.	5,0
IV кв. 2001 г.	9,0
I кв. 2002 г.	7,2
II кв. 2002 г.	4,8
III кв. 2002 г.	6,0
IV кв. 2002 г.	10,0
I кв. 2003 г.	8,0
II кв. 2003 г.	5,6
III кв. 2003 г.	6,4
IV кв. 2003 г.	11,0
I кв. 2004 г.	9,0
II кв. 2004 г.	6,6
III кв. 2004 г.	7,0
IV кв. 2004 г.	10,8

Решение. В данной задаче в качестве зависимой переменной y выступает потребление электроэнергии, в качестве независимой переменной — время t ($t = \overline{1,16}$). Проверим наличие сезонности в ряде y_t.

Первоначально изобразите ряд графически. Постройте диаграмму по исходным данным задачи. Тип диаграммы – график с маркерами. Периоды от 1 до 16 использовать в качестве подписи по оси X.

Попробуйте пообобрать линию тренда на построенном графике.

Рис. 1. График потребления электроэнергии за I кв. 2001 г. - IV кв. 2004 г.

Из рис. 1 видно, что в IV кв. потребление электроэнергии каждый год возрастает, поэтому есть подозрение на наличие сезонной компоненты в ряде. Также видно, что амплитуда сезонных колебаний постоянна, что позволяет предположить аддитивную структуру временного ряда $y = T + S + E$.

Решение задачи (с расчетом сезонных компонент). Для расчета сезонных компонент воспользуемся методом скользящей средней. Просуммируем уровни ряда последовательно за каждые четыре квартала со сдвигом на один момент времени и определим условные годовые объемы потребления электроэнергии (гр. 3 табл. 2). Полученные суммы разделим на длину периода (в нашем случае на 4) и найдем скользящие средние, которые уже не зависят от сезонности (гр. 4 табл. 2). Чтобы привести эти значения в соответствие с фактическими моментами времени, найдем средние значения из каждых двух соседних скользящих средних (гр. 5 табл. 2). Оценку сезонной компоненты найдем, вычитая из фактического значения уровня ряда *y,* центрированную скользящую среднюю (гр. 6 табл. 2).

Таблица 2. Исходные данные

Номер периода (t)	Потребление электроэнергии (y_t)	Итого за четыре квартала	Скользящая средняя за четыре квартала	Центрированная скользящая средняя	Оценка сезонной компоненты
1	2	3	4	5	6
1	6	-	-	-	-
2	4,4	24,4	6,1	-	-
3	5	25,6	6,4	6,25	-1,25
4	9	26	6,5	6,45	2,55
5	7,2	27	6,75	6,625	0,575
6	4,8	28	7	6,875	-2,075
7	6	28,8	7,2	7,1	-1,1
8	10	29,6	7,4	7,3	2,7
9	8	30	7,5	7,45	0,55
10	5,6	31	7,75	7,625	-2,025
11	6,4	32	8	7,875	-1,475
12	11	33	8,25	8,125	2,875
13	9	33,6	8,4	8,325	0,675
14	6,6	33,4	8,35	8,375	-1,775
15	7	24,4	-	-	-
16	10,8	-	-	-	-

На следующем этапе подготовим вторую вспомогательную табл. 3. Занесем в нее оценки сезонных компонент, распределив их по кварталам. За каждый квартал найдем среднюю оценку сезонной компоненты. Например, для I кв. $\overline{S_1}$ = (0,575 + 0,55 + 0,675) / 3 = 0,6.

Сезонные воздействия за период должны взаимопогашаться. В аддитивной модели это выражается в том, что сумма всех сезонных компонент за период должна быть равна нулю. Рассчитаем корректирующий коэффициент по формуле $k = \sum_{i=1}^{n} \overline{S_i} / n$, где n — длина периода.

Для нашего примера k = (0,6 - 1,958 - 1,275 + 2,708) / 4 = 0,01875.

Скорректированные значения сезонной компоненты рассчитываем как разность между средним значением сезонной компоненты и корректирующим коэффициентом $S_i = \overline{S_i} - k$, $i = \overline{1,n}$ (табл. 3).

Таблица 3.

Квартал	Год				Средняя оценка сезонной компоненты для i-го квартала ($\overline{S_i}$)	Скорректированная сезонная компонента (S_i)
	2001	2002	2003	2004		
I	-	0,575	0,55	0,675	0,6	0,6

II	-	-2,075	-2,025	-1,775	-1,95833	-2,0
III	-1,25	-1,1	-1,475	-	-1,275	-1,3
IV	2,55	2,7	2,875	-	2,708333	2,7
Корректирующий коэффициент					0,01875	

Элиминируем сезонную компоненту из исходного ряда, т.е. рассчитаем *y - S.* С этой целью заполним рабочую табл. 4.:

Таблица 4

Номер периода (t)	Исходный ряд *(y)*	Сезонная компонента *(S)*	Преобразованный ряд *(y -S)*
1-й	6,0	0,6	5,4
2-й	4,4	-2,0	6,4
3-й	5,0	-1,3	6,3
4-й	9,0	2,7	6,3
5-й	7,2	0,6	6,6
6-й	4,8	-2,0	6,8
7-й	6,0	1,3	7,3
8-й	10,0	2,7	7,3
9-й	8,0	0,6	7,4
10-й	5,6	-2,0	7,6
11-й	6,4	-1,3	7,7
12-й	11,0	2,7	8,3
13-й	9,0	0,6	8,4
14-й	6,6	-2,0	8,6
15-й	7,0	-1,3	8,3
16-й	10,8	2,7	8,1

Далее в преобразованном ряду y — S можно выделить линейный тренд.

Зная значения сезонных компонент

$$S = \begin{cases} 0,6;\ t = 1,5,9,13 \\ -2;\ t = 2,6,10,14 \\ -1,3;\ t = 3,7,11,15 \\ 2,7;\ t = 4,8,12,16 \end{cases}$$

и тренд y=a+bt, можно прогнозировать потребление электроэнергии в каждом квартале с использованием модели y = a+bt + S_t:

Вычислите y_{17}; y_{18} ; y_{19} и y_{20}.

Практические задания

Задание 1. В табл. 5 имеются данные об объеме экспорта по кварталам за 2000 — 2005 гг. Постройте аддитивную модель временного ряда и спрогнозируйте экспорт по кварталам на 2006 г.

Таблица 5.

Номер периода	Экспорт, млрд дол. США	Номер периода*(t)*	Экспорт, млрд. дол. США
1-й	4087	13-й	6975
2-й	4737	14-й	6891
3-й	5768	15-й	7527
4-й	6005	16-й	7971
5-й	5639	17-й	5875
6-й	6745	18-й	6140

7-й	6311	19-й	6248
8-й	7107	20-й	6041
9-й	5741	21-й	4626
10-й	7087	22-й	6501
11-й	7310	23-й	6284
12-й	8600	24-й	6707

Контрольные вопросы:

1. В чем заключается метод скользящей средней?
2. Для чего применяется метод скользящей средней?
3. В чем заключается метод аналитического выравнивания?
4. Для чего применяется метод аналитического выравнивания?
5. Что такое индекс сезонности?
6. Для чего применяется индекс сезонности?

ПРИЛОЖЕНИЕ 1

ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ И ЗАПРОСОВ ИНФОРМАЦИОННОЙ СИСТЕМЫ

Вариант 1: Информационная система медицинских организаций города

Каждая больница города состоит из одного или нескольких корпусов, в каждом из которых размещается одно или несколько отделений, специализирующихся на лечении определенной группы болезней; каждое отделение и имеет некоторое количество палат на определенное число коек. Поликлиники могут административно быть прикрепленными к больницам, а могут быть и нет. Как больницы, так и поликлиники обслуживаются врачебным (хирурги, терапевты, невропатологи, окулисты, стоматологи, рентгенологи, гинекологи и пр.) и обслуживающим персоналом (мед. сестры, санитары, уборщицы и пр.). Каждая категория врачебного персонала обладает характеристиками, присущими только специалистам этого профиля и по-разному участвует в связях: хирурги, стоматологи и гинекологи могут проводить операции, они же имеют такие характеристики, как число проведенных операций, число операций с летальным исходом; рентгенологи и стоматологи имеют коэффициент к зарплате за вредные условия труда, у рентгенологов и невропатологов более длительный отпуск. Врачи любого профиля могут иметь степень кандидата или доктора медицинских наук. Степень доктора медицинских наук дает право на присвоение звания профессора, а степень кандидата медицинских наук на присвоение звания доцента. Разрешено совместительство, так что каждый врач может работать либо в больнице, либо в поликлинике, либо и в одной больнице и в одной поликлинике. Врачи со званием доцента или профессора могут консультировать в нескольких больницах или поликлиниках.

Лаборатории, выполняющие те или иные медицинские анализы, могут обслуживать различные больницы и поликлиники, при условии наличия договора на обслуживание с соответствующим лечебным заведением. При этом каждая лаборатория имеет один или несколько профилей: биохимические, физиологические, химические исследования.

Пациенты амбулаторно лечатся в одной из поликлиник, и по направлению из них могут стационарно лечиться либо в больнице, к которой относится поликлиника, либо в любой другой, если специализация больницы, к которой приписана поликлиника не позволяет провести требуемое лечение. Как в больнице, так и в поликлинике ведется персонифицированный учет пациентов, полная история их болезней, все назначения, операции и т.д. В больнице пациент имеет в каждый данный момент одного лечащего врача, в поликлинике - несколько.

Виды запросов в информационной системе:

1) Получить перечень и общее число врачей указанного профиля для конкретного медицинского учреждения, больницы, либо поликлиники, либо всех медицинских учреждений города.

2) Получить перечень и общее число обслуживающего персонала указанной специальности для конкретного медицинского учреждения, больницы, либо поликлиники, либо всех медицинских учреждений города.
3) Получить перечень и общее число врачей указанного профиля, сделавших число операций не менее заданного для конкретного медицинского учреждения, больницы, либо поликлиники, либо всех медицинских учреждений города.
4) Получить перечень и общее число врачей указанного профиля, стаж работы которых не менее заданного для конкретного медицинского учреждения, больницы, либо поликлиники, либо всех медицинских учреждений города.
5) Получить перечень и общее число врачей указанного профиля со степенью кандидата или доктора медицинских наук, со званием доцента или профессора для конкретного медицинского учреждения, либо больницы, либо поликлиники, либо всех медицинских учреждений города.
6) Получить перечень пациентов указанной больницы, отделения, либо конкретной палаты указанного отделения, с указанием даты поступления, состояния, температуры, лечащего врача.
7) Получить перечень пациентов, прошедших стационарное лечение в указанной больнице, либо у конкретного врача за некоторый промежуток времени.
8) Получить перечень пациентов, наблюдающихся в врача указанного профиля в конкретной поликлинике.
9) Получить общее число палат, коек указанной больницы в общем и по каждому отделению, а также число свободных коек по каждому отделению и число полностью свободных палат.
10) Получить общее число кабинетов указанной поликлиники, число посещений каждого кабинета за определенный период.
11) Получить данные о выработке (среднее число принятых пациентов в день) за указанный период для конкретного врача, либо всех врачей поликлиники, либо для всех врачей названного профиля.
12) Получить данные о загрузке (число пациентов, у которых врач в настоящее время является лечащим врачом) для указанного врача, либо всех врачей больницы, либо для всех врачей названного профиля.
13) Получить перечень пациентов, перенесших операции в указанной больнице, либо поликлинике, либо у конкретного врача за некоторый промежуток времени.
14) Получить данные о выработке лаборатории (среднее число проведенных обследований в день) за указанный период для данного медицинского учреждения, либо всех медицинских учреждений города.

Вариант 2: Информационная система автопредприятия города

Автопредприятие города занимается организацией пассажирских и грузовых перевозок внутри города. В ведении предприятия находится автотранспорт различного назначения: автобусы, такси, маршрутные такси, прочий легковой транспорт, грузовой транспорт, транспорт вспомогательного характера, представленный различными марками. Каждая из перечисленных категорий транспорта имеет характеристики, свойственные только этой категории: например, к характеристикам только грузового транспорта относится грузоподъемность, пассажирский транспорт характеризуется вместимостью и т.д. С течением времени, с одной стороны, транспорт стареет и списывается (возможно, продается), а с другой, - предприятие пополняется новым автотранспортом.

Предприятие имеет штат водителей, закрепленных за автомобилями (за одним автомобилем может быть закреплено более одного водителя). Обслуживающий персонал (техники, сварщики, слесари, сборщики и др.) занимается техническим обслуживанием автомобильной техники, при этом различные вышеперечисленные категории также могут иметь уникальные для данной категории атрибуты. Обслуживающий персонал и водители объединяется в бригады, которыми руководят бригадиры, далее следуют мастера, затем начальники участков и цехов. В ведении предприятия находятся объекты гаражного хозяйства (цеха, гаражи, боксы и пр.), где содержится и ремонтируется автомобильная техника.

Пассажирский автотранспорт (автобусы, маршрутные такси) перевозит пассажиров по определенным маршрутам, за каждым из них закреплены отдельные единицы автотранспорта. Ведется учет числа перевозимых пассажиров, на основании чего производится перераспределением транспорта с одного маршрута на другой. Учитывается также пробег, число ремонтов и затраты на ремонт по всему автотранспорту, объем грузоперевозок для грузового транспорта, интенсивность использования транспорта вспомогательного назначения. Учитывается интенсивность работы бригад по ремонту (число ремонтов, объем выполненных работ), число замененных и отремонтированных узлов и агрегатов (двигателей, КП, мосты, шасси и т.д.) по каждой автомашине, и суммарно по участку, цеху, предприятию.

Виды запросов в информационной системе:

1) Получить данные об автопарке предприятия.
2) Получить перечень и общее число водителей по предприятию, по указанной автомашине.
3) Получить распределение водителей по автомобилям.
4) Получить данные о распределении пассажирского автотранспорта по маршрутам.
5) Получить сведения о пробеге автотранспорта определенной категории или конкретной автомашины за указанный день, месяц, год.

6) Получить данные о числе ремонтов и их стоимости для автотранспорта определенной категории, отдельной марки автотранспорта или указанной автомашины за указанный период.
7) Получить данные о подчиненности персонала: рабочие - бригадиры - мастера - начальники участков и цехов.
8) Получить сведения о наличии гаражного хозяйства в целом и по каждой категории транспорта.
9) Получить данные о распределении автотранспорта на предприятии.
10) Получить сведения о грузоперевозках, выполненных указанной автомашиной за обозначенный период.
11) Получить данные о числе использованных для ремонта указанных узлов и агрегатов для транспорта определенной категории, отдельной марки автотранспорта или конкретной автомашины за указанный период.
12) Получить сведения о полученной и списанной автотехники за указанный период.
13) Получить состав подчиненных указанного бригадира, мастера и пр.
14) Получить данные о работах, выполненных указанным специалистом (сварщиком, слесарем и т.д.) за обозначенный период в целом и по конкретной автомашине.

Вариант 3: Информационная система проектной организации

Проектная организация представлена следующими категориями сотрудников: конструкторы, инженеры, техники, лаборанты, прочий обслуживающий персонал, каждая из которых может иметь свойственные только ей атрибуты. Например, конструктор характеризуется числом авторских свидетельств, техники - оборудованием, которое они могут обслуживать, инженер или конструктор может руководить договором или проектом и т.д. Сотрудники разделены на отделы, руководимые начальником так, что каждый сотрудник числится только в одном отделе.

В рамках заключаемых проектной организацией договоров с заказчиками выполняются различного рода проекты, причем по одному договору может выполняться более одного проекта, и один проект может выполняться для нескольких договоров. Суммарная стоимость договора определяется стоимостью всех проектных работ, выполняемых для этого договора. Каждый договор и проект имеет руководителя и группу сотрудников, выполняющих этот договор или проект, причем это могут быть сотрудники не только одного отдела. Проекты выполняются с использованием различного оборудования, часть которого приписано отдельным отделам, а часть является коллективной собственностью проектной организации, при этом в процессе работы оборудование может передаваться из отдела в отдел. Для выполнения проекта оборудование придается группе, работающей над проектом, если это оборудование не используется в другом проекте.

Для выполнения ряда проектов подрядная организация может привлекать субподрядные организации, передавая им объемы работ.

Ведется учет кадров, учет выполнения договоров и проектов, стоимостной учет всех выполненных работ.

Виды запросов в информационной системе:

1) Получить данные о составе указанного отдела или всей организации полностью, по указанной категории сотрудников, по возрастному составу.
2) Получить перечень руководителей отделов.
3) Получить перечень договоров или проектов, выполняемых в данный момент или в период указанного интервала времени.
4) Получить информацию о том, какие проекты выполняются (выполнялись) в рамках указанного договора и какие договора поддерживаются указанными проектами.
5) Получить данные о стоимости выполненных договоров (проектов) в течение указанного периода времени.
6) Получить данные о распределении оборудования на данный момент или на некоторую указанную дату.
7) Получить сведения об использовании оборудования указанными проектами (договорами).
8) Получить сведения об участии указанного сотрудника или категории сотрудников в проектах (договорах) за определенный период времени.
9) Получить перечень и стоимость работ, выполненных субподрядными организациями.
10) Получить данные о численности и составе сотрудников в целом и по отдельным категориям, участвующих в указанном проекте.
11) Получить данные об эффективности использования оборудования (объемы проектных работ, выполненных с использованием того или иного оборудования).
12) Получить сведения об эффективности договоров (стоимость договоров соотнесенная с затраченным временем или стоимость с учетом привлеченных людских ресурсов).
13) Получить данные о численности и составе сотрудников в целом и по отдельным категориям, участвующих в проектах за указанный период времени.
14) Получить сведения об эффективности проектов (стоимость договоров соотнесенная с затраченным временем или стоимость с учетом привлеченных людских ресурсов).

Вариант 4: Информационная система ГИБДД

У ГИБДД есть три наиболее важные функциональные задачи:

• регистрация автотранспортных средств при совершении сделки купли-продажи;

• разработка мер, повышающих безопасность дорожного движения и выполнение всех мер при совершении ДТП (дорожно-транспортное происшествие) на улицах города (регистрация, разбор, выявление виновных, автоэкспертиза и т.п.);

• борьба с угоном автотранспортных средств, оперативный поиск угнанных машин и задержание преступников.

ГИБДД занимается выделением и учетом номерных знаков на автотранспорт. К автотранспортным средствам относятся легковые, грузовые автомобили, прицепы, полуприцепы, мотоциклы, тракторы, автобусы, микроавтобусы. На разные виды транспорта выдаются разные виды номеров и в базу данных заносятся разные характеристики. Номера могут выделяться как частным владельцам, так и организациям. В справочнике номеров, выданных частным владельцам, фиксируется: номер, ФИО владельца, его адрес, марка автомобиля, дата выпуска, объем двигателя, номера двигателя, шасси и кузова, цвет и т.п. В справочнике номеров, выданных организации, дополнительно фиксируется: название организации, район, адрес, руководитель. Существует справочник свободных номеров (серия, диапазон номеров). ГИБДД периодически проводит технический осмотр (ТО) машин. Для прохождения техосмотра необходима квитанция об оплате налогов, сумма оплаты зависит от объема двигателя. Периодичность прохождения зависит от года выпуска и вида транспортного средства. Технические характеристики, проверяемые на ТО и допуски также зависят от вида транспортного средства.

ГИБДД занимается учетом и анализом ДТП (дорожно-транспортное происшествие). При регистрации ДТП фиксируется: дата, тип происшествия (наезд на пешехода, наезд на ограждение либо столб, лобовое столкновение, наезд на впереди стоящий транспорт, боковое столкновение на перекрестке и т.п.), место происшествия, марки пострадавших автомобилей, государственный номер, тип машины (легковая, грузовая, специальная), краткое содержание, число пострадавших, сумма ущерба, причина, дорожные условия и т.п. Анализ накопленной по ДТП статистике поможет правильно расставить запрещающие и предупреждающие знаки на улицах города, а так же спланировать местонахождение постов патрульных.

Угон либо исчезновение виновника ДТП с места происшествия требует оперативного вмешательства всех постов ГИБДД и патрульных машин. Для информирования о разыскиваемой машине ее данные (включая номера двигателя и кузова) извлекаются из базы зарегистрированных номеров и передаются по рации всем постам. Ведение статистики угонов, ее анализ и опубликование результатов в СМИ поможет снизить количество угонов, а хозяевам машин принять необходимые

меры (самые угоняемые марки, самый популярный способ вскрытия, самые надежные сигнализации и т. п.).

Виды запросов в информационной системе:

1) Получить перечень и общее число организаций, которым выделены номера либо с указанной серией, либо за указанный период.
2) Получить сведения о владельце автотранспортного средства по государственному номеру автомашины.
3) Получить "досье" на автомобиль по государственному номеру - номера двигателя, кузова и шасси, участвовал ли в ДТП, прошел ли техосмотр.
4) Получить перечень и общее число владельцев машин, не прошедших вовремя техосмотр.
5) Получить статистику по любому типу ДТП за указанный период.
6) Получить результаты анализа ДТП: самые опасные места в городе, самая частая причина ДТП.
7) Получить данные о количестве ДТП, совершаемых водителями в нетрезвом виде и доля таких происшествий в общем количестве ДТП.
8) Получить список машин, отданных в розыск, будь то скрывшиеся с места ДТП или угнанные.
9) Получить данные об эффективности розыскной работы: количество найденных машин в процентном отношении.
10) Получить перечень и общее число угонов за указанный период.
11) Получить статистику по угонам: самые угоняемые марки машин, самые надежные сигнализации и т. п.

Вариант 5: Информационная система строительной организации

Строительная организация занимается строительством различного рода объектов: жилых домов, больниц, школ, мостов, дорог и т.д. по договорам с заказчиками (городская администрация, ведомства, частные фирмы и т.д.). Каждая из перечисленных категорий объектов имеет характеристики, свойственные только этой или нескольким категориям: например, к характеристикам жилых домов относится этажность, тип строительного материала, число квартир, для мостов уникальными характеристиками являются тип пролетного строения, ширина, количество полос для движения.

Структурно строительная организация состоит из строительных управлений, каждое строительное управление ведет работы на одном или нескольких участках, возглавляемых начальниками участков, которым подчиняется группа прорабов, мастеров и техников. Каждой категории инженерно-технического персонала (инженеры, технологи, техники) и рабочих (каменщики, бетонщики, отделочники, сварщики, электрики, шофера, слесари, и пр.) также свойственны характерные только для этой группы атрибуты. Рабочие объединяется в бригады, которыми руководят бригадиры. Бригадиры выбираются из числа рабочих, мастера, прорабы,

начальники участков и управлений назначаются из числа инженерно-технического персонала.

На каждом участке возводится один или несколько объектов, на каждом объекте работу ведут одна или несколько бригад. Закончив работу, бригада переходит к другому объекту на этом или другом участке. Строительному управлению придается строительная техника (подъемные краны, экскаваторы, бульдозеры и т.д.), которая распределяется по объектам.

Технология строительства того или иного объекта предполагает выполнение определенного набора видов работ, необходимых для сооружения данного типа объекта. Например, для жилого дома - это возведение фундамента, кирпичные работы, прокладка водоснабжения и т.д. Каждый вид работ на объекте выполняется одной бригадой. Для организации работ на объекте составляется графики работ, указывающие в каком порядке и в какие сроки выполняются те или иные работы, а также смета, определяющая какие строительные материалы и в каких количествах необходимы для сооружения объекта. По результатам выполнения работ составляется отчет с указанием сроков выполнения работ и фактических расходов материалов.

Виды запросов в информационной системе:

1) Получить перечень строительных управлений и/или участков и их руководителей.
2) Получить список специалистов инженерно-технического состава обозначенного участка или строительного управления с указанием их должностей.
3) Получить перечень объектов, возводимых указанным строительным управлением и/или участком, и графики их возведения.
4) Получить состав бригад, работавших (работающих) на строительстве указанного объекта.
5) Получить перечень строительной техники, приданной указанному строительному управлению.
6) Получить перечень строительной техники, выделенной на указанный объект либо работавшей там в течение указанного периода времени.
7) Получить график и смету на строительство указанного объекта.
8) Получить отчет о сооружении указанного объекта.
9) Получить перечень объектов, возводимых в некотором строительном управлении или в целом по организации, на которых в обозначенный период времени выполнялся указанный вид строительных работ.
10) Получить перечень видов строительных работ, по которым имело место превышение сроков выполнения на указанном участке, строительном управлении или в целом по организации.
11) Получить перечень строительных материалов, по которым имело место превышение по смете на указанном участке, строительном управлении или в целом по организации.

12) Получить перечень видов строительных работ, выполненных указанной бригадой в течение обозначенного периода времени с указанием объектов, где эти работы выполнялись.
13) Получить перечень бригад, выполненных указанный вид строительных работ в течение обозначенного периода времени с указанием объектов, где эти работы выполнялись.

Вариант 6: Информационная система библиотечного фонда города

Библиотечный фонд города составляют библиотеки, расположенные на территории города. Каждая библиотека включает в себя абонементы и читальные залы. Пользователями библиотек являются различные категории читателей: студенты, научные работники, преподаватели, школьники, рабочие, пенсионеры и другие жители города. Каждая категория читателей может обладать непересекающимися характеристиками-атрибутами: для студентов это название учебного заведения, факультет, курс, номер группы, для научного работника - название организации, научная тема и т. д. Каждый читатель, будучи зарегистрированным в одной из библиотек, имеет доступ ко всему библиотечному фонду города.

Библиотечный фонд (книги, журналы, газеты, сборники статей, сборники стихов, диссертации, рефераты, сборники докладов и тезисов докладов и пр.) размещен в залах-хранилищах различных библиотек на определенных местах хранения (номер зала, стеллажа, полки) и идентифицируется номенклатурными номерами. При этом существуют различные правила относительно тех или иных изданий: какие-то подлежат только чтению в читальных залах библиотек, для тех, что выдаются, может быть установлен различный срок выдачи и т.д. С одной стороны, библиотечный фонд может пополняться, с другой, - с течением времени происходит его списание.

Произведения авторов, составляющие библиотечный фонд, также можно разделить на различные категории, характеризующиеся собственным набором атрибутов: учебники, повести, романы, статьи, стихи, диссертации, рефераты, тезисы докладов и т.д.

Сотрудники библиотеки, работающие в различных залах различных библиотек, ведут учет читателей, а также учет размещения и выдачи литературы

Виды запросов в информационной системе:

1) Получить список читателей с заданными характеристиками: студентов указанного учебного заведения, факультета, научных работников по определенной тематике и т.д.
2) Выдать перечень читателей, на руках у которых находится указанное произведение.
3) Получить список читателей, на руках у которых находится указанное издание (книга, журнал и т.д).

4) Получить перечень читателей, которые в течение указанного промежутка времени получали издание с некоторым произведением, и название этого издания.
5) Выдать список изданий, которые в течение некоторого времени получал указанный читатель из фонда библиотеки, где он зарегистрирован.
6) Получить перечень изданий, которыми в течение некоторого времени пользовался указанный читатель из фонда библиотеки, где он не зарегистрирован.
7) Получить список литературы, которая в настоящий момент выдана с определенной полки некоторой библиотеки.
8) Выдать список читателей, которые в течение обозначенного периода были обслужены указанным библиотекарем.
9) Получить данные о выработке библиотекарей (число обслуженных читателей в указанный период времени).
10) Получить список читателей с просроченным сроком литературы.
11) Получить перечень указанной литературы, которая поступила (была списана) в течение некоторого периода.
12) Выдать список библиотекарей, работающих в указанном читальном зале некоторой библиотеки.
13) Получить список читателей, не посещавших библиотеку в течение указанного времени.
14) Получить список инвентарных номеров и названий из библиотечного фонда, в которых содержится указанное произведение.
15) Выдать список инвентарных номеров и названий из библиотечного фонда, в которых содержатся произведения указанного автора.
16) Получить список самых популярных произведений.

Вариант 7: Информационная система спортивных организаций города

Спортивная инфраструктура города представлена спортивными сооружениями различного типа: спортивные залы, манежи, стадионы, корты и т.д. Каждая из категорий спортивных сооружений обладает атрибутами, специфичными только для нее: стадион характеризуется вместимостью, корт - типом покрытия.

Спортсмены под руководством тренеров занимаются отдельными видами спорта, при этом один и тот же спортсмен может заниматься несколькими видами спорта, и в рамках одного и того же вида спорта может тренироваться у нескольких тренеров. Все спортсмены объединяются в спортивные клубы, при этом каждый из них может выступать только за один клуб.

Организаторы соревнований проводят состязания по отдельным видам спорта на спортивных сооружениях города. По результатам участия спортсменов в соревнованиях производится награждение.

Виды запросов в информационной системе:

1) Получить перечень спортивных сооружений указанного типа в целом или удовлетворяющих заданным характеристикам (например, стадионы, вмещающие не менее указанного числа зрителей).
2) Получить список спортсменов, занимающихся указанным видом спорта в целом либо не ниже определенного разряда.
3) Получить список спортсменов, тренирующихся у некого тренера в целом либо не ниже определенного разряда.
4) Получить список спортсменов, занимающихся более чем одним видом спорта с указанием этих видов спорта.
5) Получить список тренеров указанного спортсмена.
6) Получить перечень соревнований, проведенных в течение заданного периода времени в целом либо указанным организатором.
7) Получить список призеров указанного соревнования.
8) Получить перечень соревнований, проведенных в указанном спортивном сооружении в целом либо по определенному виду спорта.
9) Получить перечень спортивных клубов и число спортсменов этих клубов, участвовавших в спортивных соревнованиях в течение заданного интервала времени.
10) Получить список тренеров по определенному виду спорта.
11) Получить список спортсменов, не участвовавших ни в каких соревнованиях в течение определенного периода времени.
12) Получить список организаторов соревнований и число проведенных ими соревнований в течение определенного периода времени.
13) Получить перечень спортивных сооружений и даты проведения на них соревнований в течение определенного периода времени.

Вариант 8: Информационная система аэропорта

Работников аэропорта можно подразделить на пилотов, диспетчеров, техников, кассиров, работников службы безопасности, сплавочной службы и других, которые административно относятся каждый к своему отделу. Каждая из перечисленных категорий работников имеет уникальные атрибуты-характеристики, определяемые профессиональной направленностью. В отделах существует разбиение работников на бригады. Отделы возглавляются начальниками, которые представляют собой администрацию аэропорта. В функции администрации входит планирование рейсов, составление расписаний, формирование кадрового состава аэропорта. За каждым самолетом закрепляется бригада пилотов, техников и обслуживающего персонала. Пилоты обязаны проходить каждый год медосмотр, не прошедших медосмотр необходимо перевести на другую работу. Самолет должен своевременно осматриваться техниками и при необходимости ремонтироваться. Подготовка к рейсу включает в себя техническую часть (техосмотр, заправка

необходимого количества топлива) и обслуживающую часть (уборка салона, запас продуктов питания и т.п.).

В расписании указывается тип самолета, рейс, дни вылета, время вылета и прилета, маршрут (начальный и конечный пункты назначения, пункт пересадки), стоимость билета. Билеты на авиарейсы можно приобрести заранее или забронировать в авиакассах. Цена билета зависит не только от маршрута, но и от времени вылета (в неудобное время - ночь, раннее утро - цена билета ниже). До отправления рейса, если в этом есть необходимость, билет можно вернуть. Авиарейсы могут быть задержаны из-за погодных условий, технических неполадок, а также могут быть отменены, если не продано меньше установленного минимума билетов.

Авиарейсы можно разделить на следующие категории: внутренние, международные, чартерные, грузоперевозки, специальные рейсы. Пассажир при посадке в самолет должен предъявить билет, паспорт, а для международного рейса обязан также предъявить заграничный паспорт и пройти таможенный досмотр. Пассажиры могут сдавать свои вещи в багажное отделение. На рейсы грузоперевозок и специальные рейсы билеты не продаются. Для спец. рейсов не существует расписания. Билеты на чартерные рейсы распространяет то агентство, которое его организовало.

Виды запросов в информационной системе:

1) Получить список и общее число всех работников аэропорта, начальников отделов, работников указанного отдела, по стажу работы в аэропорту, половому признаку, возрасту, признаку наличия и количеству детей, по размеру заработной платы.
2) Получить перечень и общее число работников в бригаде, по всем отделам, в указанном отделе, обслуживающих конкретный рейс, по возрасту, суммарной (средней) зарплате в бригаде.
3) Получить перечень и общее число пилотов, прошедших медосмотр либо не прошедших его в указанный год, по половому признаку, возрасту, размеру заработной платы.
4) Получить перечень и общее число самолетов приписанных к аэропорту, находящихся в нем в указанное время, по времени поступления в аэропорт, по количеству совершенных рейсов.
5) Получить перечень и общее число самолетов, прошедших техосмотр за определенный период времени, отправленных в ремонт в указанное время, ремонтированных заданное число раз, по количеству совершенных рейсов до ремонта, по возрасту самолета.
6) Получить перечень и общее число рейсов по указанному маршруту, по длительности перелета, по цене билета и по всем этим критериям сразу.
7) Получить перечень и общее число отмененных рейсов полностью, в указанном направлении, по указанному маршруту, по количеству невостребованных мест, по процентному соотношению невостребованных мест.

8) Получить перечень и общее число задержанных рейсов полностью, по указанной причине, по указанному маршруту, и количество сданных билетов за время задержки.
9) Получить перечень и общее число рейсов, по которым летают самолеты заданного типа и среднее количество проданных билетов на определенные маршруты, по длительности перелета, по цене билета, времени вылета.
10) Получить перечень и общее число авиарейсов указанной категории, в определенном направлении, с указанным типом самолета.
11) Получить перечень и общее число пассажиров на данном рейсе, улетевших в указанный день, улетевших за границу в указанный день, по признаку сдачи вещей в багажное отделение, по половому признаку, по возрасту.
12) Получить перечень и общее число свободных и забронированных мест на указанном рейсе, на определенный день, по указанному маршруту, по цене, по времени вылета.
13) Получить общее число сданных билетов на некоторый рейс, в указанный день, по определенному маршруту, по цене билета, по возрасту, полу.

Вариант 9: Информационная система гостиничного комплекса

Гостиничный комплекс состоит из нескольких зданий-гостиниц (корпусов). Каждый корпус имеет ряд характеристик, таких, как класс отеля (двух-, пятизвездочные), количество этажей в здании, общее количество комнат, комнат на этаже, местность номеров (одно-, двух-, трехместные и т.д.), наличие служб быта: ежедневная уборка номера, прачечная, химчистка, питание (рестораны, бары) и развлечения (бассейн, сауна, бильярд и пр.). От типа корпуса и местности номера зависит сумма оплаты за него. Химчистка, стирка, дополнительное питание, все развлечения производятся за отдельную плату.

С крупными организациями (туристические фирмы, организации, занимающиеся проведением международных симпозиумов, конгрессов, семинаров, карнавалов и т.д.) заключаются договора, позволяющие организациям бронировать номера с большими скидками на определенное время вперед не для одного человека, а для группы людей. Каждая из перечисленных групп организаций обладает характеристиками, свойственными только этой группе. Желательно группы людей от одной организации не расселять по разным этажам. В брони указывается класс отеля, этаж, количество комнат и общее количество людей. Броня может быть отменена за неделю до заселения. На основе маркетинговых работ расширяется рынок гостиничных услуг, в результате чего заключаются договора с новыми фирмами. Также исследуется мнение жильцов о ценах и сервисе. Жалобы фиксируются и исследуются. Изучается статистика популярности номеров. Ведется учет долгов постояльца гостинице за все дополнительные услуги.

Новые жильцы пополняют перечень клиентов гостиницы. Ведется учет свободных номеров, дополнительных затрат постояльцев гостиницы и учет расходов и доходов гостиничного комплекса.

Виды запросов в информационной системе:

1) Получить перечень и общее число фирм, забронировавших места в объеме, не менее указанного, за весь период сотрудничества, либо за некоторый период.
2) Получить перечень и общее число постояльцев, заселявшихся в номера с указанными характеристиками за некоторый период.
3) Получить количество свободных номеров на данный момент.
4) Получить сведения о количестве свободных номеров с указанными характеристиками.
5) Получить сведения о конкретном свободном номере: в течение какого времени он будет пустовать и о его характеристиках.
6) Получить список занятых сейчас номеров, которые освобождаются к указанному сроку.
7) Получить данные об объеме бронирования номеров данной фирмой за указанный период, и каким номерам отдавались предпочтения.
8) Получить список недовольных клиентов и их жалобы.
9) Получить данные о рентабельности номеров с определенными характеристиками: соотношение об объеме продаж номеров к накладным расходам за указанный период.
10) Получить сведения о постояльце из заданного номера: его счет гостинице за дополнительные услуги, поступавшие от него жалобы, виды дополнительных услуг, которыми он пользовался.
11) Получить сведения о фирмах, с которыми заключены договора о брони на указанный период.
12) Получить сведения о наиболее часто посещающих гостиницу постояльцах по всем корпусам гостиниц, по определенному зданию.
13) Получить сведения о новых клиентах за указанный период.
14) Получить сведения о конкретном человеке, сколько раз он посещал гостиницу, в каких номерах и в какой период останавливался, какие счета оплачивал.
15) Получить сведения о конкретном номере: кем он был занят в определенный период.
16) Получить процентное отношение всех номеров к номерам, бронируемым партнерами.

Вариант 10: Информационная система торговой организации

Торговая организация ведет торговлю в торговых точках разных типов: универмаги, магазины, киоски, лотки и т.д.), в штате которых работают продавцы. Универмаги разделены на отдельные секции, руководимые управляющими секций и расположенные, возможно, на разных этажах здания. Как универмаги, так и магазины могут иметь несколько залов, в которых работает определенное число продавцов, универмаги, магазины, киоски могут иметь такие характеристики, как размер торговой точки, платежи за аренду, коммунальные услуги, количество прилавков и т.д. Кроме того, в универмагах и магазинах учет проданных товаров ведется персонифицировано с фиксацией имен и характеристик покупателя, чего в киосках и на лотках сделать не представляется возможным.

Заказы поставщику составляются на основе заявок, поступающих из торговых точек. На основе заявок менеджеры торговой организации выбирают поставщика, формируют заказы, в которых перечисляются наименования товаров и заказываемое их количество, которое может отличаться от запроса из торговой точки. Если указанное наименование товара ранее не поставлялось, оно пополняет справочник номенклатуры товаров. На основе маркетинговых работ постоянно изучается рынок поставщиков, в результате чего могут появляться новые поставщики и исчезать старые. При этом одни и те же товары торговая организация может получать от разных поставщиков и, естественно, по различным ценам.

Поступившие товары распределяются по торговым точкам и в любой момент можно получить такое распределение.

Продавцы торговых точек ведут продажу товаров, учитывая все сделанные продажи, фиксируя номенклатуру и количество проданного товара, а продавцы универмагов и магазинов дополнительно фиксируют имена и характеристики покупателей, что позволяет вести учет покупателей и сделанных ими покупок. В процессе торговли торговые точки вправе менять цены на товары в зависимости от спроса и предложения товаров, а также по согласованию передавать товары в другую торговую точку.

Виды запросов в информационной системе:

1) Получить перечень и общее число поставщиков, поставляющих указанный вид товара, либо некоторый товар в объеме, не менее заданного, за весь период сотрудничества, либо за указанный период.
2) Получить перечень и общее число покупателей, купивших указанный вид товара за некоторый период, либо сделавших покупку товара в объеме, не менее заданного.
3) Получить номенклатуру и объем товаров в указанной торговой точке.
4) Получить сведения об объеме и ценах на указанный товар по всем торговым точкам, по торговым точкам заданного типа, по конкретной торговой точке.
5) Получить данные о выработке на одного продавца за указанный период по всем торговым точкам, по торговым точкам заданного типа.

6) Получить данные о выработке отдельно взятого продавца отдельно взятой торговой точки за указанный период.
7) Получить данные об объеме продаж указанного товара за некоторый период по всем торговым точкам, по торговым точкам заданного типа, по конкретной торговой точке.
8) Получить данные о заработной плате продавцов по всем торговым точкам, по торговым точкам заданного типа, по конкретной торговой точке.
9) Получить сведения о поставках определенного товара указанным поставщиком за все время поставок, либо за некоторый период.
10) Получить данные об отношении объема продаж к объему торговых площадей, либо к числу торговых залов, либо к числу прилавков по торговым точкам указанного типа, о выработке отдельно взятого продавца торговой точки, по заданной торговой точке.
11) Получить данные о рентабельности торговой точки: соотношение объема продаж к накладным расходам (суммарная заработная плата продавцов + платежи за аренду, коммунальные услуги) за указанный период.
12) Получить сведения о поставках товаров по указанному номеру заказа.
13) Получить сведения о покупателях указанного товара за обозначенный, либо за весь период, по всем торговым точкам, по торговым точкам указанного типа, по данной торговой точке.
14) Получить сведения о наиболее активных покупателях по всем торговым точкам, по торговым точкам указанного типа, по данной торговой точке.
15) Получить данные о товарообороте торговой точки, либо всех торговых определенной группы за указанный период.

Вариант 11: Информационная система ВУЗа

Студенты, организованные в группы, учатся на одном из факультетов, возглавляемом деканатом, в функции которого входит контроль за учебным процессом. В учебном процессе участвуют преподаватели кафедр, административно относящиеся к одному из факультетов. Преподаватели подразделяются на следующие категории: ассистенты, преподаватели, старшие преподаватели, доценты, профессора. Ассистенты и преподаватели могут обучаться в аспирантуре, ст. преподаватели, доценты, могут возглавлять научные темы, профессора - научные направления. Преподаватели любой из категории в свое время могли защитить кандидатскую, а доценты и профессора и докторскую диссертацию, при этом преподаватели могут занимать должности доцента и профессора только, если они имеют соответственно звания доцента и профессора.

Учебный процесс регламентируется учебным планом, в котором указывается, какие учебные дисциплины на каких курсах и в каких семестрах читаются для студентов каждого года набора, с указанием количества часов на каждый вид занятий по дисциплине (виды занятий: лекции, семинары, лабораторные работы,

консультации, курсовые работы, ИР и т.д.) и формы контроля (зачет, экзамен). Перед началом учебного семестра деканаты раздают на кафедры учебные поручения, в которых указываются какие кафедры (не обязательно относящиеся к данному факультету), какие дисциплины и для каких групп должны вести в очередном семестре. Руководствуясь ими, на кафедрах осуществляется распределение нагрузки, при этом по одной дисциплине в одной группе разные виды занятий могут вести один или несколько разных преподавателей кафедры (с учетом категории преподавателей, например, ассистент не может читать лекции, а профессор никогда не будет проводить лабораторные работы). Преподаватель может вести занятия по одной или нескольким дисциплинам для студентов как своего, так и других факультетов. Сведения о проведенных экзаменах и зачетах собираются деканатом.

По окончании обучения студент выполняет дипломную работу, руководителем которой является преподаватель с кафедры, относящейся к тому же факультету, где обучается студент, при этом преподаватель может руководить несколькими студентами.

Виды запросов в информационной системе:

1) Получить перечень и общее число студентов указанных групп либо указанного курса (курсов) факультета полностью, по половому признаку, году рождения, возрасту, признаку наличия детей, по признаку получения и размеру стипендии.
2) Получить список и общее число преподавателей указанных кафедр либо указанного факультета полностью, либо указанных категорий (ассистенты, доценты, профессора и т.д.) по половому признаку, году рождения, возрасту, признаку наличия и количеству детей, размеру заработной платы, являющихся аспирантами, защитивших кандидатские, докторские диссертации в указанный период.
3) Получить перечень и общее число тем кандидатских и докторских диссертаций, защитивших сотрудниками указанной кафедры либо указанного факультета.
4) Получить перечень кафедр, проводящих занятия в указанной группе либо на указанном курсе указанного факультета в указанном семестре, либо за указанный период.
5) Получить список и общее число преподавателей, проводивших (проводящих) занятия по указанной дисциплине в указанной группе либо на указанном курсе указанного факультета.
6) Получить перечень и общее число преподавателей проводивших (проводящих) лекционные, семинарские и другие виды занятий в указанной группе либо на указанном курсе указанного факультета в указанном семестре, либо за указанный период.
7) Получить список и общее число студентов указанных групп, сдавших зачет либо экзамен по указанной дисциплине с указанной оценкой.

8) Получить список и общее число студентов указанных групп или указанного курса указанного факультета, сдавших указанную сессию на отлично, без троек, без двоек.
9) Получить перечень преподавателей, принимающих (принимавших) экзамены в указанных группах, по указанным дисциплинам, в указанном семестре.
10) Получить список студентов указанных групп, либо которым заданный преподаватель поставил некоторую оценку за экзамен по определенным дисциплинам, в указанных семестрах, за некоторый период.
11) Получить список студентов и тем дипломных работ, выполняемых ими на указанной кафедре либо у указанного преподавателя.
12) Получить список руководителей дипломных работ с указанной кафедры, либо факультета полностью и раздельно по некоторым категориям преподавателей.
13) Получить нагрузку преподавателей (название дисциплины, количество часов), ее объем по отдельным видам занятий и общую нагрузку в указанном семестре для конкретного преподавателя либо для преподавателей указанной кафедры.

Вариант 12: Информационная система железнодорожной пассажирской станции

Работников железнодорожной станции можно подразделить на водителей подвижного состава, диспетчеров, ремонтников подвижного состава, путей, кассиров, работников службы подготовки составов, справочной службы и других, которые административно относятся каждый к своему отделу. Каждая из перечисленных категорий работников имеет уникальные атрибуты-характеристики, определяемые профессиональной направленностью. В отделах существует разбиение работников на бригады. Отделы возглавляются начальниками, которые представляют собой администрацию железнодорожной станции. В функции администрации входит планирование маршрутов, составление расписаний, формирование кадрового состава железнодорожной станции. За каждым локомотивом закрепляется локомотивная бригада. За несколькими локомотивами закрепляется бригада техников-ремонтников, выполняющая рейсовый и плановый техосмотр (по определенному графику), ремонт, техническое обслуживание. Водители локомотивов обязаны проходить каждый год медосмотр, не прошедших медосмотр необходимо перевести на другую работу. Локомотив должен своевременно осматриваться техниками-ремонтиками и при необходимости ремонтироваться. Подготовка к рейсу включает в себя техническую часть (рейсовый техосмотр, мелкий ремонт) и обслуживающую часть (уборка вагонов, запас продуктов питания и т.п.).

В расписании указывается тип поезда (скорый, пассажирский . . .), номер поезда, дни и время отправления и прибытия, маршрут (начальный и конечный пункты назначения, основные узловые станции), стоимость билета. Билеты на поезд

можно приобрести заранее или забронировать в железнодорожных кассах. До отправления поезда, если есть необходимость, билет можно вернуть. Отправление поездов может быть задержано из-за опозданий поездов, погодных условий, технических неполадок.

Железнодорожные маршруты можно разделить на следующие категории: внутренние, международные, туристические, специальные маршруты. Пассажиры могут сдавать свои вещи в багажное отделение.

Виды запросов в информационной системе:

1) Получить перечень и общее число всех работников железнодорожной станции, начальников отделов, работников указанного отдела, по стажу работы на станции, половому признаку, возрасту, признаку наличия и количества детей, размеру заработной платы.
2) Получить перечень и общее число работников в бригаде, по всем отделам, в указанном отделе, обслуживающих некоторый локомотив, по возрасту, суммарной (средней) зарплате в бригаде.
3) Получить перечень и общее число водителей локомотивов, прошедших медосмотр либо не прошедших медосмотр в указанный год, по половому признаку, возрасту, размеру заработной платы.
4) Получить перечень и общее число локомотивов, приписанных к железнодорожной станции, находящихся на ней в указанное время, по времени прибытия на станции, по количеству совершенных маршрутов.
5) Получить перечень и общее число локомотивов, прошедших плановый техосмотр за определенный период времени, отправленных в ремонт в обозначенное время, ремонтированных указанное число раз, по количеству совершенных рейсов до ремонта, по возрасту локомотива.
6) Получить перечень и общее число поездов на указанном маршруте, по длительности маршрута, по цене билета и по всем этим критериям сразу.
7) Получить перечень и общее число отмененных рейсов полностью, в указанном направлении, по указанному маршруту.
8) Получить перечень и общее число задержанных рейсов полностью, по указанной причине, по указанному маршруту, и количество сданных билетов за время задержки.
9) Получить перечень и среднее количество проданных билетов за указанный интервал времени на определенные маршруты, по длительности маршрута, по цене билета.
10) Получить перечень и общее число маршрутов указанной категории, следующих в определенном направлении.
11) Получить перечень и общее число пассажиров на указанном рейсе, уехавших в указанный день, уехавших за границу в указанный день, по признаку сдачи вещей в багажное отделение, по половому признаку, по возрасту.
12) Получить перечень и общее число невыкупленных билетов на указанном рейс, день, некоторый маршрут.
13) Получить общее число сданных билетов на указанный рейс, день, маршрут.

Вариант 13: Информационная система зоопарка

Служащих зоопарка можно подразделить на несколько категорий: ветеринары, уборщики, дрессировщики, строители-ремонтники, работники администрации. Каждая из перечисленных категорий работников имеет уникальные атрибуты-характеристики, определяемые профессиональной направленностью. За каждым животным ухаживает определенный круг служащих, причем только ветеринарам, уборщикам и дрессировщикам разрешен доступ в клетки к животным.

В зоопарке обитают животные различных климатических зон, поэтому часть животных на зиму необходимо переводить в отапливаемые помещения. Животных можно подразделить на хищников и травоядных. При расселении животных по клеткам необходимо учитывать не только потребности данного вида, но и их совместимость с животными в соседних клетках (нельзя рядом селить, например, волков и их добычу - различных копытных).

Для кормления животных необходимы различные типы кормов: растительный, живой, мясо и различные комбикорма. Растительный корм это фрукты и овощи, зерно и сено. Живой корм - мыши, птицы, корм для рыб. Для каждого вида животных рассчитывается свой рацион, который в свою очередь варьируется в зависимости от возраста, физического состояния животного и сезона. Таким образом, у каждого животного в зоопарке имеется меню на каждый день, в котором указывается количество и время кормлений в день, количество и вид пищи (обезьянам необходимы фрукты и овощи, мелким хищникам - хорькам, ласкам, совам, некоторым кошачьим, змеям - надо давать мышей). У зоопарка имеются поставщики кормов для животных. Каждый поставщик специализируются на каких-то конкретных видах кормов. Часть кормов зоопарк может производить сам: запасать сено, разводить мышей и т.д.

Ветеринары должны проводить медосмотры, следить за весом, ростом, развитием животного, ставить своевременно прививки и заносить все эти данные в карточку, которая заводится на каждую особь при ее появлении в зоопарке. Больным животным назначается лечение и при необходимости их можно изолировать в стационаре.

При определенных условиях (наличие пары особей, подходящих по возрасту, физическому состоянию) можно ожидать появления потомства. Потомство от данной пары животных при достижении ими положенного возраста можно либо оставить в зоопарке, создав для них подходящие условия содержания, либо обменяться с другими зоопарками или просто раздать в другие зоопарки - по решению администрации.

Виды запросов в информационной системе:

1) Получить список и общее число служащих зоопарка, либо служащих данной категории полностью, по продолжительсти работы в зоопарке, по половому признаку, возрасту, размеру заработной платы.
2) Получить перечень и общее число служащих зоопарка, ответственных за указанный вид животных либо за конкретную особь за все время пребывания животного в зоопарке, за указанный период времени.

3) Получить список и общее число служащих зоопаркав, имеющих доступ к указанному виду животных либо к конкретной особи.
4) Получить перечень и общее число всех животных в зоопарке либо животных указанного вида, живших в указанной клетке все время пребывания в зоопарке, по половому признаку, возрасту, весу, росту.
5) Получить перечень и общее число нуждающихся в теплом помещении на зиму, полностью животных только указанного вида или указанного возраста.
6) Получить перечень и общее число животных, которым поставлена указанная прививка, либо переболевших некоторой болезнью, по длительности пребывания в зоопарке, половому признаку, возрасту, признаку наличия и количеству потомства.
7) Получить перечень всех животных, совместимых с указанным видом, либо только тех животных, которых необходимо переселить, или тех, которые нуждаются в теплом помещении.
8) Получить перечень и общее число поставщиков кормов полностью, либо поставляющих только определенный корм, поставлявших в указанный период, по количеству поставляемого корма, цене, датам поставок.
9) Получить перечень и объем кормов, производимых зоопарком полностью, либо только тех кормов, в поставках которых зоопарк не нуждается (обеспечивает себя сам).
10) Получить перечень и общее число животных полностью, либо указанного вида, которым необходим определенный тип кормов, в указанном сезоне, возрасте или круглый год.
11) Получить полную информацию (рост, вес, прививки, болезни, дата поступления в зоопарк или дата рождения, возраст, количество потомства) обо всех животных, или о животных только данного вида, о конкретном животном, об особи, живущей в указанной клетке.
12) Получить перечень животных, от которых можно ожидать потомство в перспективе, в указанный период.
13) Получить перечень и общее число зоопарков, с которыми был произведен обмен животными в целом или животными только указанного вида.

Вариант 14: Информационная система театра

Работников театра можно подразделить на актеров, музыкантов, постановщиков и служащих. Каждая из перечисленных категорий имеет уникальные атрибуты-характеристики и может подразделяться (например, постановщики) на более мелкие категории. Театр возглавляет директор, в функции которого входят контроль за постановками спектаклей, утверждение репертуара, принятие на работу новых служащих, приглашение актеров и постановщиков. Актеры, музыканты и постановщики, работающие в театре, могут уезжать на гастроли. Актеры театра могут иметь звания заслуженных и народных артистов, могут быть лауреатами конкурсов. Также актерами театра могут быть и студенты театральных училищ.

Каждый актер имеет свои вокальные и внешние данные (пол, возраст, голос, рост и т.п.), которые могут подходить для каких-то ролей, а для каких-то нет (не всегда женщина может сыграть мужчину и наоборот).

Для постановки любого спектакля необходимо подобрать актеров на роли и дублеров на каждую главную роль. Естественно, что один и тот же актер не может играть более одной роли в спектакле, но может играть несколько ролей в различных спектаклях. У спектакля также имеется режиссер-постановщик, художник-постановщик, дирижер-постановщик, автор. Спектакли можно подразделить по жанрам: музыкальная комедия, трагедия, оперетта и пр. С другой стороны, спектакли можно подразделить на детские, молодежные и пр. В репертуаре театра указывается какие спектакли, в какие дни и в какое время будут проходить, а также даты премьер. В кассах театра можно заранее приобрести билеты или абонемент на любые спектакли. Абонемент обычно включает в себя билеты на спектакли либо конкретного автора, либо конкретного жанра. Цена билетов зависит от места, и спектакля. На премьеры билеты дороже. Администрацией театра фиксируется количество проданных билетов на каждый спектакль.

Виды запросов в информационной системе:

1) Получить список и общее число все работников театра, актеров, музыкантов, по стажу работы в театре, по половому признаку, году рождения, возрасту, признаку наличия и количества детей, размеру заработной платы.
2) Получить перечень и общее число спектаклей, указанных в репертуаре на данный сезон, уже сыгранных спектаклей, спектаклей указанного жанра, когда-либо сыгранных в этом театре, за указанный период.
3) Получить перечень и общее число всех поставленных спектаклей, спектаклей указанного жанра, когда-либо поставленных в этом театре, поставленных за указанный период.
4) Получить список авторов поставленных спектаклей, авторов, живших в указанном веке, авторов указанной страны, авторов спектаклей указанного жанра когда-либо поставленных в этом театре, поставленных за указанный период времени.
5) Получить перечень спектаклей указанного жанра, некоторого автора, авторов обозначенной страны, спектаклей, написанных в определенном веке, впервые поставленных на сцене указанного театра в обозначенный период времени.
6) Получить список актеров, подходящих по своим данным на указанную роль.
7) Получить общее число и список актеров театра, имеющих звания, получивших их за некоторый период, на указанных конкурсах, по половому признаку, по возрасту.
8) Получить список актеров и постановщиков, приезжавших когда-либо на гастроли в театр за указанный период, перечень уезжавших на гастроли в определенное время с данным спектаклем.

9) Получить список для указанного спектакля: актеров, их дублеров, имена режиссера-постановщика, художника-постановщика, дирижера-постановщика, авторов, дату премъеры.
10) Получить перечень и общее число ролей, сыгранных указанным актером всего, за некоторый период времени, в спектаклях определенного жанра, в спектаклях указанного режиссера-постановщика, в детских спектаклях.
11) Получить сведения о числе проданных билетов на все спектакли, на конкретный спектакль, на премьеры, за указанный период, в том числе проданных предварительно.
12) Получить общую сумму вырученных денег за указанный спектакль, за некоторый период времени.
13) Получить перечень и общее число свободных мест на все спектакли, на конкретный спектакль, на премьеры.

Вариант 15: Информационная система фотоцентра

Фотоцентр имеет главный офис и сеть филиалов и киосков приема заказов, расположенных по определенным адресам. Филиалы и киоски различаются количество рабочих мест. В киосках осуществляется только прием заказов, поэтому каждый киоск прикреплен к определенному филиалу, в котором эти заказы выполняются. В филиалах имеется необходимое оборудование для проявки пленок и печати фотографий. Филиалы и киоски принимают заказы на проявку пленок, печать фотографий и проявку, и печать вместе. В заказе на печать указывается количество фотографий с каждого кадра, общее количество фотографий, формат, тип бумаги и срочность выполнения заказа. При заказе большого количества фотографий предоставляются скидки. Срочные заказы принимаются только в филиалах, и они имеют цену в два раза больше, чем обычный заказ. При приобретении дисконтной карты клиент получает значительные скидки на печать фотографий. Пленка, приобретенная в том же филиале, куда она принесена на проявку, проявляется бесплатно.

Клиентов можно разделить на профессионалов и любителей. Профессионалам, приносящим заказы в один и тот же филиал, могут быть предложены персональные скидки. Фотомагазины и киоски предлагают к продаже различные фототовары: фотопленки, фотоаппараты, альбомы и другие фотопринадлежности. Фотомагазины также предлагают дополнительные виды услуг: фотографии на документы, реставрация фотографий, прокат фотоаппаратов, художественное фото, предоставление услуг профессионального фотографа.

Сведения о выполненных заказах и продаже различных фототоваров собираются и обрабатываются, и на основе этой информации делается общий заказ на поставку расходных материалов (фотобумага, фотопленка, химические реактивы), фототоваров и оборудования. Полученные товары и материалы распределяются в соответствии с запросами по киоскам и магазинам. У фотоцентра может быть несколько поставщиков, которые специализируются на различных поставках, либо на поставках фототоваров различных фирм.

Виды запросов в информационной системе:

1) Получить перечень и общее число пунктов приема заказов на фотоработы по филиалам, по киоскам приема заказов, в целом по фотоцентру.
2) Получить перечень и общее число заказов на фотоработы по филиалам, киоскам приема заказов, в целом по фотоцентру, поступивших в течение некоторого периода времени.
3) Получить перечень и общее число заказов (отдельно простых и срочных) на отдельные виды фоторабот по указанному филиалу, киоску приема заказов, поступивших в течение некоторого периода времени.
4) Получить сумму выручки с заказов (отдельно простых и срочных) на отдельные виды фоторабот по указанному филиалу, киоску приема заказов, поступивших в течение некоторого периода времени.
5) Получить количество отпечатанных фотографий в рамках простых и срочных заказов по указанному филиалу, киоску приема заказов, фотоцентру в целом за некоторый период времени.
6) Получить количество проявленных фотопленок в рамках простых и срочных заказов по указанному филиалу, киоску приема заказов, фотоцентру в целом за некоторый период времени.
7) Получить перечень поставщиков в целом по фотоцентру, поставщиков отдельных видов фототоваров, сделавших поставки в некоторый период, поставки определенного объема.
8) Получить список клиентов в целом по фотоцентру, клиентов указанного филиала, имеющих скидки, сделавших заказы определенного объема.
9) Получить сумму выручки от реализации фототоваров в целом по фотоцентру, по указанному филиалу, проданных в течение некоторого периода времени.
10) Получить перечень фототоваров и фирм, их производящих, которые пользуются наибольшим спросом в целом по фотоцентру, в указанном филиале.
11) Получить перечень реализованных фототоваров и объемы их реализации в целом по фотоцентру, по указанному филиалу, проданных в течение некоторого периода времени.
12) Получить перечень рабочих мест фотоцентру в целом и указанного профиля.

ПРИЛОЖЕНИЕ 2

ГОСУДАРСТВЕННЫЙ СТАНДАРТ СОЮЗА ССР

Единая система программной документации

ТЕХНИЧЕСКОЕ ЗАДАНИЕ, ТРЕБОВАНИЯ К СОДЕРЖАНИЮ И ОФОРМЛЕНИЮ

ГОСТ 19.201-78

Настоящий стандарт устанавливает порядок построения и оформления технического задания на разработку программы или программного изделия для вычислительных машин, комплексов и систем независимо от их назначения и области применения.

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 в документ не включать.

1.3. Для внесения изменений и дополнений в техническое задние 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:

- 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 приёмки;
- приложения.

В зависимости от особенностей программы или программного изделия допускается уточнять содержание разделов, вводить новые разделы или объединять отдельные из них.

2. Содержание разделов.

2.1. 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.

2.2. В разделе "Основание для разработки" должны быть указаны:

- документ (документы), на основании которых ведется разработка;
- организация, утвердившая этот документ, и дата его утверждения;
- наименование и (или) условное обозначение темы разработки.

2.3. В разделе " Назначение разработки" должно быть указано функциональное и эксплуатационное назначение программы или программного изделия.

2.4. Раздел "Технические требования к программе или программному изделию" должен содержать следующие подразделы:

- требования к функциональным характеристикам;
- требования к надёжности;
- 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.

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

2.4.2. В подразделе "Требования к надёжности" должны быть указаны требования к обеспечению надёжного функционирования (обеспечение устойчивого функционирования, контроль входной и выходной информации, время восстановления после отказа и т.п.)

2.4.3. 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.

2.4.4. В подразделе " Требования к составу и параметрам технических средств" указывают необходимый состав технических средств с указанием их технических характеристик.

2.4.5. 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.

2.4.6. В подразделе "Требования к маркировке и упаковке" в общем случае указывают требования к маркировке программного изделия, варианты и способы упаковки.

2.4.7. 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. 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 приёмки" должны быть указаны виды испытаний и общие требования к приёмке работы.

2.8. В приложениях к техническому заданию, при необходимости, приводят:

- перечень научно-исследовательских и других работ, обосновывающих разработку;
- схемы алгоритмов, таблицы, описания, обоснования, расчёты и другие документы, которые могут быть использованы при разработке;
- другие источники разработки.