МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ (ФИЛИАЛ)

ФЕДЕРАЛЬНОГО ГОСУДАРСТВЕННОГО БЮДЖЕТНОГО

ОБРАЗОВАТЕЛЬНОГО УЧРЕЖДЕНИЯ ВЫСШЕГО ОБРАЗОВАНИЯ

«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

В Г. ТАГАНРОГЕ РОСТОВСКОЙ ОБЛАСТИ

ПИ (филиал) ДГТУ в г. Таганроге

ЦМК «Прикладная информатика»

Практикум

По выполнению практических работ

по дисциплине

«Проектирование и дизайн информационных систем»

Таганрог
2023

Составители: А.В. Ганциевский

Практикум по выполнению практических работ по дисциплине 09.02.07 Информационные системы и программирование. ПИ (филиала) ДГТУ в г. Таганроге, 2023г.

В практикуме кратко изложены теоретические вопросы, необходимые для успешного выполнения практической работы, рабочее задание и контрольные вопросы для самопроверки.

Предназначено для обучающихся по специальности 09.02.07 Информационные системы и программирование.

Ответственный за выпуск:

Председатель ЦМК: _____ О.В. Андриян

Введение

В учебно-методическом пособии к практикуму по курсу «Проектирование и дизан информационных систем» изложены сведения, необходимые для успешного выполнения практических занятий по данному курсу. Описан процесс работы с инструментарием, применяемым на практических занятиях, представлен ряд типичных задач и подходы к их решению. Практические занятия посвящены углубленному знакомству обучающихся с управлением процесса разработки приложений с использованием инструментальных средств; обеспечением сбора данных для анализа использования и функционирования информационной системы; программированием в соответствии с требованиями технического задания; использованием критериев оценки качества и надежности функционирования информационной системы; применением методики тестирования разрабатываемых приложений; определением состава оборудования и программных средств разработки информационной системы; разработкой документации по эксплуатации информационной системы; проведением оценки качества и экономической эффективности информационной системы в рамках своей компетенции; модификацией отдельных модулей информационной системы..

Цель настоящего пособия – помочь обучающимся при выполнении практических работ, выполняемых для закрепления знаний по теоретическим основам и получения практических навыков работы на компьютерах.

Обучающийся должен знать: основные виды и процедуры обработки информации, модели и методы решения задач обработки информации; основные платформы для создания, исполнения и управления информационной системой; основные процессы управления проектом разработки; основные модели построения информационных систем, их структуру, особенности и области применения; методы и средства проектирования, разработки и тестирования информационных систем; систему стандартизации, сертификации и систему обеспечения качества продукции.

Обучающийся должен уметь: осуществлять постановку задач по обработке информации; проводить анализ предметной области; осуществлять выбор модели и средства построения информационной системы и программных средств; использовать алгоритмы обработки информации для различных приложений; решать прикладные вопросы программирования и языка сценариев для создания программ; разрабатывать графический интерфейс приложения; создавать и управлять проектом по разработке приложения; проектировать и разрабатывать систему по заданным требованиям и спецификациям.

Данное учебно-методическое пособие предназначено для обучающихся 3 и 4 курсов.

Правила выполнения практических занятий

Практические занятия выполняются каждым обучающимся самостоятельно в полном объеме и согласно содержанию методических указаний.

Перед выполнением обучающийся должен отчитаться перед преподавателем за выполнение предыдущего занятия (сдать отчет).

Обучающийся должен на уровне понимания и воспроизведения предварительно усвоить необходимую для выполнения практических занятий теоретическую и информацию.

Обучающийся, получивший положительную оценку и сдавший отчет по предыдущему практическому занятию, допускается к выполнению следующему занятию.

Обучающийся, пропустивший практическое занятие по уважительной либо неуважительной причине, закрывает задолженность в процессе выполнения последующих практических занятий.

ПРАКТИЧЕСКАЯ РАБОТА №1

Анализ предметной области различными методами: контент-анализ, вебометрический анализ, анализ ситуаций, моделирование и др.

Цели: ознакомиться с процессом анализа предметной области и получить навыки по использованию методов анализа предметной области.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Определение предметной области.

Основные понятия системного и структурного анализа.

Задание № 1

Ознакомиться с предложенным вариантом описания предметной области (согласно заданию индивидуального проекта).

Вариант	Предметнаяобласть	Сущностьзадачи
1	Страховаяме дицинскаяко мпания	Страховая медицинская компания (СМК) заключает договоры добровольного медицинского страхования с населением и договоры с лечебными учреждениями на лечение застрахованных клиентов. При возникновении страхового случая клиент подает заявку на оказание медицинских услуг по условиям договора инспектору, который работает с данным клиентом. Инспектор направляет данного клиента в лечебное учреждение. Отчеты о своей деятельности инспектор предоставляет в бухгалтерию. Бухгалтерия проверяет оплату договоров, перечисляет денежные средства за оказанные услуги лечебным учреждениям, производит отчисления в налоговые органы и предоставляет отчетность в органы государственной статистики. СМК не только оплачивает лечение застрахованного лица при возникновении с ним страхового случая, но и, при возникновениикаких-либо осложнений после лечения, оплачивает лечение этих осложнений
2	Агентствонедв ижимости	Агентство недвижимости занимается покупкой, продажей, сдачей в аренду объектов недвижимости по договорам с их собственниками. Агентство управляет объектами недвижимости как физических, так и юридических лиц. Собственник может иметь несколько объектов. В случае покупки или аренды клиент может произвести осмотр объекта. В качестве одной из услуг, предлагаемых

		агентством, является проведение инспектирования текущего состояния объекта для адекватного определения его рыночной цены. По результатам своей деятельности агентство производит отчисления в налоговые органы и предоставляет отчетность в органы государственной статистики
3	Кадровоеагентство	Кадровое агентство способствует трудоустройству безработных граждан. Агентство ведет учет и классификацию данных о безработных на основании резюме от них. От предприятий города поступают данные о свободных вакансиях, на основании которых агентство предлагает различные варианты трудоустройства соискателям. В случае положительного исхода поиска
Вариант	Предметнаяобласть	Сущностьзадачи
		вакансия считается заполненной, а безработный становится трудоустроенным. По результатам своей деятельности кадровое агентство производит отчисления в налоговые органы и предоставляет отчетность в органы государственнойстатистики
4	Компания по разработке программных продуктов	Компания заключает договор с клиентом на разработку программного продукта согласно техническому заданию. После утверждения технического задания определяется состав и объем работ, составляется предварительная смета. На каждый проект назначается ответственный за его выполнение – куратор проекта, который распределяет нагрузку между программистами и следит за выполнением технического задания. Когда программный продукт готов, то его внедряют, производят обучение клиента и осуществляют дальнейшее сопровождение. По результатам своей деятельности компания производит отчисления в налоговые органы и предоставляет отчетность ворганы государственнойстатистики
5	Туроператор	Туроператор предоставляет возможность своим клиентам осуществить туристическую или деловую поездку в различные города России и мира. При разработке нового тура сначала анализируется текущая ситуация на рынке туризма и выбирается направление тура. После этого определяется статус тура, бронируются места в гостиницах и билеты на переезд к месту тура, разрабатывается культурная/деловая/развлекательная программа, утверждаются сроки тура. На каждый тур назначается ответственное лицо от туроператора, которое будет вести данный тур для улаживания проблем в случае возникновения каких-нибудь чрезвычайных или форс-мажорных ситуаций. Клиент приходит в офис туроператора, где вместе с менеджером выбирает уже разработанный тур и оформляет путевку. После возвращения из тура клиент может высказать свои замечания или пожелания, которые будут учтены при доработке существующих туров или при разработке новых. Также, для дальнейшего улучшения тура, туроператор проводит анализ отчетов от посредников (гостиница, гиды и т.д.). По результатам своей деятельности

		туроператор производит отчисления в налоговые органы и предоставляет отчетность в органыгосударственной статистики

Задание № 2

Проанализировать предметную область, уточнив и дополнив ее, руководствуясь собственным опытом, консультациями и любыми источниками (книгами, учебниками или интернет- источниками).

Задание № 3

Выполнить структурное разбиение предметной области на отдельные подразделения (подсистемы) согласно выполняемым имифункциям.

Задание № 4

Определить задачи и функции системы в целом и функции каждого подразделения (подсистемы).

Задание № 5

Продумать подробное описание работы каждого подразделения (подсистемы), алгоритмов и сценариев выполнения ими отдельных работ. Продумать виды входной и выходной информации для каждого подразделения (подсистемы).

Задание № 6

Описать схему работы будущей информационной системы, учитывая выделенные и описанные ранее подсистемы.

Задание № 7

Определить группу пользователей, для которой данная система будет более востребована.

Описать перечень функций системы, которые будут доступны данной группе пользователей.

Задание № 8

Расписать основные функциональные возможности администратора системы, как одного из пользователей системы.

Задание № 9

Оформить отчет

ПРАКТИЧЕСКАЯ РАБОТА №2

Изучение устройств автоматизированного сбора информации.

Цели: изучение устройств автоматизированного сбора информации.

Форма отчета:

−выполнить задание;

−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Организация и методы сбора информации. Устройства автоматизированного сбора информации.

Задание № 1

Изучить и описать технологии штрихового кодирования (BarCodeTechnologies) сбора информации.

Задание № 2

Изучить и описать технологии радиочастотной идентификации (RFID – RadioFrequencyIdentificationTechnologies) сбора информации.

Задание № 3

Изучить и описать карточные технологии (CardTechnologies) сбора информации.

Задание № 4

Изучить и описать технологии сбора данных (DataCommunicationsTechnologies).

Задание № 5

Изучить и описать технологии распознавания голоса, оптического и магнитного распознавания текста, биометрические технологии и некоторыедругие.

Задание № 6

В зависимости от целей, сферы деятельности и располагаемых технических средств можно выделить методы сбора данных, применяемые:

1) в экономических информационных системах (например,маркетинга);
2) в геоинформационныхсистемах;
3) в статистических информационныхсистемах;
4) в информационных системах управления производственнымипроцессами.

Задание № 7

Для заданной предметной области (см. практическая работа № 1) опишите устройства и методы автоматизированного сбора информации.

Задание № 8

Оформить отчет.

ПРАКТИЧЕСКАЯ РАБОТА № 3
ОЦЕНКА ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ

ИНФОРМАЦИОННОЙ СИСТЕМЫ

Цели: изучение методов оценки экономической эффективности информационных систем.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Понятие экономической эффективности информационных систем.

Методы оценки экономической эффективности информационных систем. Расчет экономической эффективности информационных систем.

Задание № 1

Охарактеризуйте затратные методы оценки экономической эффективности информационных систем.

Задание № 2

Охарактеризуйте методы оценки прямого результата информационных систем.

Задание № 3

Охарактеризуйте методы оценки экономической эффективности информационных систем, основанные на оценке идеальности процесса.

Задание № 4

Охарактеризуйте квалиметрические подходы к оценке экономической эффективности

информационных систем, основанные на оценке идеальности процесса.

Задание № 5

Проведите сравнительный анализ методов оценки экономической эффективности информационныхсистем.

Задание № 6

Рассчитайте экономическую эффективность заданной информационной системы (см. практическая работа № 1).

Задание № 7

Оформить отчет.

ПРАКТИЧЕСКАЯ РАБОТА № 4

РАЗРАБОТКА МОДЕЛИ АРХИТЕКТУРЫ ИНФОРМАЦИОННОЙ СИСТЕМЫ

Цели: получение навыков разработки модели архитектуры информационной системы.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Понятие архитектуры информационной системы.

Виды архитектур информационных систем. Достоинства и недостатки.

Задание № 1

Спроектировать информационную систему (см. практическая работа № 1) на основе архитектуры «файл-сервер».

Задание № 2

Спроектировать информационную систему (см. практическая работа № 1) на основе архитектуры «клиент-сервер».

Задание № 3

Спроектировать информационную систему (см. практическая работа № 1) на основе многозвенной архитектуры «клиент-сервер».

Задание № 4

Оформить отчет.

Практическая работа №5

Обоснование выбора средств проектирования информационной системы

Цели: изучение методов средств проектирования информационной системы и выбор подходящего средства проектирования.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Краткие теоретические сведения

Классификация CASE-средств

Все CASE-средства делятся на типы, категории и уровни. Классификация по типам отражает функциональную ориентацию CASE-средств в технологическом процессе.

1) **Анализ и проектирование**. Средства данной группы используютсядля создания спецификаций системы и ее проектирования; они поддерживают широко известные методологии проектирования. Ктакимсредствамотносятся: CASE, Аналитик (Эйтэкс), The Developer (ASYST Technologies), POSE (Computer Systems Advisers), ProKil*Workbench

(McDonnell Douglas), Excelerator (Index Technology), Design-Aid (Naslec), Design Machine (Optima), MicroStep (Meta Systems), vsDesigner (Visucil Sofhvctrr), Anuijsl'Designer (Yourdon), Design/lDEF (Meta Soft\vare), BPWin (Logic Works). SELECT (Select Sofhvare Tools), System Architect (Popkin Software & Systems), Wesfmounl I-C.iSE Youi-don (Westmount Technology B. V. & CADRE Technologies), CASE/4/0 (microTOOL GmbH). Их целью является определение системных требований и свойств, которыми система должна обладать, а также создание проекта системы, удовлетворяющей этим требованиям и обладающей соответствующими свойствами. На выходе продуцируются спецификации компонент системы и интерфейсов, связывающих эти компоненты, а также "калька" архитектуры системы и детальная "калька" проекта, включающая алгоритмы и определения структур данных.

2) **Проектирование баз данных и файлов**. Средства данной группыобеспечивают логическое моделирование данных, автоматическое преобразование моделей данных в Третью Нормальную Форму, автоматическую генерацию схем БД и описаний форматов файлов на уровне программного кода: ERWin (Logic Works), Chen Toolkit (Chen & Asssociales), S-Designor (SDP), Designer2000 (Oracle), Silverrun (Computer Systems Advisers).

3) Программирование. Средства этой группы поддерживают этапыпрограммирования и тестирования, а также автоматическую кодогенерацию из спецификаций, получая полностью документированную выполняемую программу: COBOL 2/Workhench (Mikro Focus), DECASE (DEC), NETRON/CAP (Netron), APS (Sage Softwtire). Помимо диаграммеров различного назначения и средств поддержки работы с репозиторием. в эту группу средств включены и традиционные генераторы кодов, анализаторы кодов (как в статике, так и в динамике), генераторы наборов тестов, анализаторы покрытия тестами, отладчики.

4) **Сопровождение и реинжиниринг**. К таким средствам относятсядокументаторы, анализаторы программ, средства реструктурирования и реинженериинга: Adpac CASE Tools (Adpac). Scan/COBOL, u Superstructure (Computer Data Systems), Inspector/Recoder (Language Technology). Их целью является корректировка, изменение, анализ, преобразование и реинжениринг существующей системы. Средства позволяют осуществлять поддержку всей системной документации, включая коды, спецификации, наборы тестов;контролировать покрытие тестами для оценки полноты тестируемости; управлять функционированием системы и т.п. Особый интерес представляют средства обеспечения мобильности (в CASE они получили название средств миграции) и реинжиниринга. К средствам миграции относятся трансляторы, конверторы, макрогенераторы и др., позволяющие обеспечить перенос существующей системы в новое операционное или аппаратурное окружение.

Средства реинжиниринга включают:

- статические анализаторы для продуцирования схем системы ПО из
е кодов, оценки влияния модификаций (например, "эффекта ряби" - внесение изменений с целью исправления ошибок порождает новые ошибки);
- динамические анализаторы (обычно, компиляторы и интерпретаторы с встроенными отладочными возможностями);
- документаторы, позволяющие автоматически получать обновленную документацию при изменении кода;
- редакторы кодов, автоматически изменяющие при редактировании и все предшествующие коду структуры (например, спецификации);
- средства доступа к спецификациям, их модификации и генерации нового (модифицированного) кода;
- средства реверсного инжиниринга, транслирующие коды в спецификации.

5) Окружение. Средства поддержки платформ для интеграции, создания и придания товарного вида CASE-средствам: Multi/Cam (AGS Management Systems), Design/OA (Meta Software).

6) Управление проектом. Средства, поддерживающие планирование, контроль. руководство, взаимодействие, т.е. функции, необходимые в процессе разработки и сопровождения проектов: Project Workbench (Applied Business Technology). Классификация по категориям определяет уровень интегрированности по выполняемым функциям и включает вспомогательные программы (tools), пакеты разработчика (toolkit) и инструментальные средства (workbench). Категория tools обозначает вспомогательный пакет, решающий небольшую автономную задачу, принадлежащую проблеме более широкого масштаба. Категория toolkit представляет совокупность интегрированных программных средств, обеспечивающих помощь для одного из классов программных задач; использует репозиторий для всей технической и управляющей информации о проекте, концентрируясь при этом на поддержке, как правило, одной фазы или одного этапа разработки ПО.

Категория workbench представляет собой интеграцию программных средств, которые поддерживают системный анализ, проектирование и разработку ПО; используют репозиторий, содержащий всю техническую и управляющую информацию о проекте; обеспечивают автоматическую передачу системной информации между разработчиками и этапами разработки; организуют поддержку практически полного ЖЦ (от анализатребований и проектирования ПО до получения документированной выполняемой программы). Workbench, по сравнению с toolkit, обладает более высокой степенью интеграции выполняемых функций, большей самостоятельностью и автономностью использования, а также наличием тесной связи с системными и техническими средствами аппаратно-вычислительной среды, на которой workbench функционирует. По существу, workbench может рассматриваться как автоматизированная рабочая станция, используемая как инструментарий для автоматизации всех или отдельных совокупностей работ по созданию ПО.

Классификация по уровням связана с областью действия CASE в пределах жизненного цикла ПО. Однако четкие критерии определения границ между уровнями не установлены, поэтому данная классификация имеет, вообще говоря, качественный характер.

Верхние (Upper) CASE часто называют средствами компьютерного планирования. Они призваны повышать эффективность деятельности руководителей фирмы и проекта путем сокращения затрат на определение политики фирмы и на создание общего плана проекта. Этот план включает цели и стратегии их достижения, основные действия в свете целей и задач фирмы, установление стандартов на различные виды взаимосвязей и т.д. Использование верхних CASE позволяет построить модель предметной области, отражающую всю существующую специфику. Она направлена на понимание общего и частного механизмов функционирования, имеющихся возможностей, ресурсов, целей проекта в соответствии с назначением фирмы. Эти средства позволяют проводить анализ различных сценариев (в том числе наилучших и наихудших), накапливая информацию для принятия оптимальных решений. Средние (Middle) CASE считаются средствами поддержки этапов анализа требований и проектирования спецификаций и структуры ПО. Их использование существенно сокращает цикл разработки проекта; при этом важную роль играет возможность накопления и хранения знаний, обычно имеющихся только в голове разработчика-аналитика, что позволит использовать накопленные решения при создании других проектов. Основная выгода от использования среднего CASE состоит в значительном облегчении проектирования систем, проектирование превращается в итеративный процесс, включающий следующие действия:

- пользователь обсуждает с аналитиком требования к проектируемой системе;
- аналитик документирует эти требования, используя диаграммы и словари входных данных;
- пользователь проверяет эти диаграммы и словари, при необходимости модифицируя их;

- аналитик отвечает на эти модификации, изменяя соответствующие спецификации.

Кроме того, средние CASE обеспечивают возможности быстрого документирования требований и быстрого прототипирования.

Нижние (Lower) CASE являются средствами разработки ПО (при этом может использоваться до 30% спецификаций, созданных средствами среднего CASE). Они содержат системные словари и графические средства, исключающие необходимость разработки физических спецификаций. Имеются системные спецификации, которые непосредственно переводятся в программные коды разрабатываемой системы (при этом автоматически генерируется до 80-90% кодов). На эти средства возложены также функции тестирования, управления конфигурацией, формирования документации. Главными преимуществами нижних CASE являются: значительное уменьшение времени на разработку, облегчение модификаций, поддержка возможностей прототипирования (совместно со средними CASE).

Примеры различных CASE-средств

1. Silverrun

CASE-средство Silverrun американской фирмы Computer Systems Advisers, Inc. (CSA) используется для анализа и проектирования ИС бизнес-класса и ориентировано в большей степени на спиральную модель ЖЦ. Оно применимо для поддержки любой методологии, основанной на раздельном построении функциональной и информационной моделей (диаграмм потоков данных и диаграмм "сущность-связь").

Настройка на конкретную методологию обеспечивается выбором требуемой графической нотации моделей и набора правил проверки проектных спецификаций. В системе имеются готовые настройки для наиболее распространенных методологий: DATARUN (основная методология, поддерживаемая Silverrun), Gane/Sarson, Yourdon/DeMarco, Merise, Ward/Mellor, Information Engineering. Для каждого понятия, введенного в проекте имеется возможность добавления собственных описателей. Архитектура Silverrun позволяет наращивать среду разработки по мере необходимости.

Структура и функции

Silverrun имеет модульную структуру и состоит из четырех модулей, каждый из которых является самостоятельным продуктом и может приобретаться и использоваться без связи с остальными модулями.

Модуль построения моделей бизнес-процессов в форме диаграмм потоков данных (BPM - Business Process Modeler) позволяет моделировать функционирование обследуемой организации или создаваемой ИС. В модуле BPM обеспечена возможность работы с моделями большой сложности:автоматическая перенумерация, работа с деревом процессов (включая визуальное перетаскивание ветвей), отсоединение и присоединение частей модели для коллективной разработки. Диаграммы могут изображаться в нескольких предопределенных нотациях, включая Yourdon/DeMarco и Gane/Sarson. Имеется также возможность создавать собственные нотации, в том числе добавлять в число изображаемых на схеме дескрипторов определенные пользователем поля.

Модуль концептуального моделирования данных (ERX – Entity-Relationship eXpert) обеспечивает построение моделей данных "сущность-связь", не привязанных к конкретной реализации. Этот модуль имеет встроенную экспертную систему, позволяющую создать корректную нормализованную модель данных посредством ответов на содержательные вопросы о взаимосвязи данных. Возможно автоматическое построение модели данных из описаний структур данных. Анализ функциональных зависимостей атрибутов дает

возможность проверить соответствие модели требованиям третьей нормальной формы и обеспечить их выполнение. Проверенная модель передается в модуль RDM.

Модуль реляционного моделирования (RDM – Relational Data Modeler) позволяет создавать детализированные модели "сущность -связь", предназначенные для реализации в реляционной базе данных. В этом модуле документируются все конструкции, связанные с построением базы данных: индексы, триггеры, хранимые процедуры и т.д. Гибкая изменяемая нотация и расширяемость репозитория позволяют работать по любой методологии. Возможность создавать подсхемы соответствует подходу ANSI SPARC к представлению схемы базы данных. На языке подсхем моделируются как узлы распределенной обработки, так и пользовательские представления. Этот модуль обеспечивает проектирование и полное документирование реляционных баз данных.

Менеджер репозитория рабочей группы (WRM – Workgroup Repository Manager) применяется как словарь данных для хранения общей для всех моделей информации, а также обеспечивает интеграцию модулей Silverrun в единую среду проектирования.

Платой за высокую гибкость и разнообразие изобразительных средств построения моделей является такой недостаток Silverrun, как отсутствие жесткого взаимного контроля между компонентами различных моделей (например, возможности автоматического распространения изменений между DFD различных уровней декомпозиции). Следует, однако, отметить, что этот недостаток может иметь существенное значение только в случае использования каскадной модели ЖЦ ПО.

Взаимодействие с другими средствами

Для автоматической генерации схем баз данных у Silverrun существуют мосты к наиболее распространенным СУБД: Oracle, Informix, DB2, Ingres, Progress, SQL Server, SQLBase, Sybase. Для передачи данных в средства разработки приложений имеются мосты к языкам 4GL: JAM, PowerBuilder, SQL Windows, Uniface, NewEra, Delphi. Все мосты позволяют загрузить в Silverrun RDM информацию из каталогов соответствующих СУБД или языков 4GL. Это позволяет документировать, перепроектировать или переносить на новые платформы уже находящиеся в эксплуатации базы данных и прикладные системы. При использовании моста Silverrun расширяет свой внутренний репозиторий специфичными для целевой системы атрибутами. После определения значений этих атрибутов генератор приложений переносит их во внутренний каталог среды разработки или использует при генерации кода на языке SQL. Таким образом, можно полностью определить ядро базы данных с использованием всех возможностей конкретной СУБД: триггеров, хранимых процедур, ограничений ссылочной целостности. При создании приложения на языке 4GL данные, перенесенные из репозитория Silverrun, используются либо для автоматической генерации интерфейсных объектов, либо для быстрого их создания вручную.

Для обмена данными с другими средствами автоматизации проектирования, создания специализированных процедур анализа и проверки проектных спецификаций, составления специализированных отчетов в соответствии с различными стандартами в системе Silverrun имеется три способа выдачи проектной информации во внешние файлы:

1. Система отчетов. Можно, определив содержимое отчета по репозиторию, выдать отчет в текстовый файл. Этот файл можно затем загрузить в текстовый редактор или включить в другой отчет;
2. Система экспорта/импорта. Для более полного контроля над структурой файлов в системе экспорта/импорта имеется возможность определять не только содержимое экспортного файла, но и разделители записей, полей в записях, маркеры начала и конца текстовых полей. Файлы с указанной структурой можно не только формировать, но и загружать в репозиторий. Это дает возможность обмениваться данными с различными системами: другими CASE-средствами, СУБД, текстовыми редакторами и электронными таблицами;
3. Хранение репозитория во внешних файлах через ODBC-драйверы. Для доступа к данным репозитория из наиболее распространенных систем управления базами данных

обеспечена возможность хранить всю проектную информацию непосредственно в формате этих СУБД.

Групповая работа

Групповая работа поддерживается в системе Silverrun двумя способами:

1. В стандартной однопользовательской версии имеется механизм контролируемого разделения и слияния моделей. Разделив модель на части, можно раздать их нескольким разработчикам. После детальной доработки модели объединяются в единые спецификации;
2. Сетевая версия Silverrun позволяет осуществлять одновременную групповую работу с моделями, хранящимися в сетевом репозитории на базе СУБД Oracle, Sybase или Informix. При этом несколько разработчиков могут работать с одной и той же моделью, так как блокировка объектов происходит на уровне отдельных элементов модели.

Среда функционирования

Имеются реализации Silverrun трех платформ - MS Windows, Macintosh и OS/2 Presentation Manager - с возможностью обмена проектными данными между ними.

Для функционирования в среде Windows необходимо иметь компьютер с процессором модели не ниже i486 и оперативную память объемом не менее 8 Мб (рекомендуется 16 Мб). На диске полная инсталляция Silverrun занимает 20 Мб.

2. Rational Rose

Rational Rose – CASE-средство фирмы Rational Software Corporation (США) – предназначено для автоматизации этапов анализа и проектирования ПО, а также для генерации кодов на различных языках и выпуска проектной документации. Rational Rose использует синтез-методологию объектно-ориентированного анализа и проектирования, основанную на подходах трех ведущих специалистов в данной области: Буча, Рамбо и Джекобсона. Разработанная ими универсальная нотация для моделирования объектов (UML - Unified Modeling Language) претендует на роль стандарта в области объектно-ориентированного анализа и проектирования. Конкретный вариант Rational Rose определяется языком, на котором генерируются коды программ (C++, Smalltalk, PowerBuilder, Ada, SQLWindows и ObjectPro) . Основной вариант – Rational Rose/C++ – позволяет разрабатывать проектную документацию в виде диаграмм и спецификаций, а также генерировать программные коды на C++. Кроме того, Rational Rose содержит средства реинжиниринга программ, обеспечивающие повторное использование программных компонент в новых проектах.

Структура и функции

В основе работы Rational Rose лежит построение различного рода диаграмм и спецификаций, определяющих логическую и физическуюструктуры модели, ее статические и динамические аспекты. В их число входят диаграммы классов, состояний, сценариев, модулей, процессов.

В составе Rational Rose можно выделить 6 основных структурных компонент: репозиторий, графический интерфейс пользователя, средства просмотра проекта (browser), средства контроля проекта, средства сбора статистики и генератор документов. К ним добавляются генератор кодов (индивидуальный для каждого языка) и анализатор для C++, обеспечивающий реинжиниринг – восстановление модели проекта по исходным текстам программ.

Репозиторий представляет собой объектно-ориентированную базу данных. Средства просмотра обеспечивают "навигацию" по проекту, в том числе, перемещение по иерархиям классов и подсистем, переключение от одного вида диаграмм к другому и т. д. Средства контроля и сбора статистики дают возможность находить и устранять ошибки по мере развития проекта, а не после завершения его описания. Генератор отчетов формирует тексты выходных документов на основе содержащейся в репозитории информации.

Средства автоматической генерации кодов программ на языке C++, используя информацию, содержащуюся в логической и физической моделях проекта, формируют файлы заголовков и файлы описаний классов и объектов. Создаваемый таким образом скелет программы может быть уточнен путем прямого программирования на языке C++. Анализатор кодов C++ реализован в виде отдельного программного модуля. Его назначение состоит в том, чтобы создавать модули проектов в форме Rational Rose на основе информации, содержащейся в определяемых пользователем исходных текстах на C++. В процессе работы анализатор осуществляет контроль правильности исходных текстов и диагностику ошибок. Модель, полученная
в результате его работы, может целиком или фрагментарно использоваться в различных проектах. Анализатор обладает широкими возможностями настройки по входу и выходу. Например, можно определить типы исходных файлов, базовый компилятор, задать, какая информация должна быть включена в формируемую модель и какие элементы выходной модели следует выводить на экран. Таким образом, Rational Rose/C++ обеспечивает возможность повторного использования программных компонент.

В результате разработки проекта с помощью CASE-средства Rational Rose формируются следующие документы:

1. диаграммы классов;
2. диаграммы состояний;
3. диаграммы сценариев;
4. диаграммы модулей;
5. диаграммы процессов;
6. спецификации классов, объектов, атрибутов и операций
7. заготовки текстов программ;
8. модель разрабатываемой программной системы.

Последний из перечисленных документов является текстовым файлом, содержащим всю необходимую информацию о проекте (в том числе необходимую для получения всех диаграмм и спецификаций).

Тексты программ являются заготовками для последующей работы программистов. Они формируются в рабочем каталоге в виде файлов типов .h (заголовки, содержащие описания классов) и .cpp (заготовки программ для методов). Система включает в программные файлы собственные комментарии, которые начинаются с последовательности символов //##. Состав информации, включаемой в программные файлы, определяется либо по умолчанию, либо по усмотрению пользователя. В дальнейшем эти исходные тексты развиваются программистами в полноценные программы.

Взаимодействие с другими средствами и организация групповой работы

Rational Rose интегрируется со средством PVCS для организации групповой работы и управления проектом и со средством SoDA – для документирования проектов. Интеграция Rational Rose и SoDA обеспечивается средствами SoDA.

Для организации групповой работы в Rational Rose возможно разбиение модели на управляемые подмодели. Каждая из них независимо сохраняется на диске или загружается в модель. В качестве подмодели может выступать категория классов или подсистема.

Для управляемой подмодели предусмотрены операции:

1. загрузка подмодели в память;

2. выгрузка подмодели из памяти;
3. сохранение подмодели на диске в виде отдельного файла;
4. установка защиты от модификации;
5. замена подмодели в памяти на новую.

Наиболее эффективно групповая работа организуется при интеграции Rational Rose со специальными средствами управления конфигурацией и контроля версий (PVCS). В этом случае защита от модификации устанавливается на все управляемые подмодели, кроме тех, которые выделены конкретному разработчику. В этом случае признак защиты от записи устанавливается для файлов, которые содержат подмодели, поэтому при считывании "чужих" подмоделей защита их от модификации сохраняется и случайные воздействия окажутся невозможными.

Среда функционирования

Rational Rose функционирует на различных платформах: IBM PC (в среде Windows), Sun SPARC stations (UNIX, Solaris, SunOS), Hewlett-Packard (HP UX), IBM RS/6000 (AIX).

Для работы системы необходимо выполнение следующих требований:

1. Платформа Windows - процессор 80386SX или выше (рекомендуется 80486), память8Мб (рекомендуется 12Мб), пространство на диске 8Мб + 1-3Мб для одной модели.
2. Платформа UNIX – память 32+(16*число пользователей)Мб, пространство на диске 30Мб + 20 при инсталляции + 1-3Мб для одной модели.

Совместимость по версиям обеспечивается на уровне моделей.

3. Vantage Team Builder (Westmount I-CASE)

Vantage Team Builder представляет собой интегрированный программный продукт, ориентированный на реализацию каскадной модели ЖЦ ПО и поддержку полного ЖЦ ПО.

Структура и функции

Vantage Team Builder обеспечивает выполнение следующих функций:

1. проектирование диаграмм потоков данных, "сущность-связь", структур данных, структурных схем программ и последовательностей экранных форм;
2. проектирование диаграмм архитектуры системы – SAD (проектирование состава и связи вычислительных средств, распределения задач системы между вычислительными средствами, моделирование отношений типа "клиент-сервер", анализ использования менеджеров транзакций и особенностей функционирования систем в реальном времени);
3. генерация кода программ на языке 4GL целевой СУБД с полным обеспечением программной среды и генерация SQL-кода для создания таблиц БД, индексов, ограничений целостности и хранимых процедур;
4. программирование на языке C со встроенным SQL;
5. управление версиями и конфигурацией проекта;
6. многопользовательский доступ к репозиторию проекта;
7. генерация проектной документации по стандартным и индивидуальным шаблонам;
8. экспорт и импорт данных проекта в формате CDIF (CASE Data Interchange Format).

Vantage Team Builder поставляется в различных конфигурациях в зависимости от используемых СУБД (ORACLE, Informix, Sybase или Ingres) или средств разработки приложений (Uniface). Конфигурация Vantage Team Builder for Uniface отличается от остальных некоторой степенью ориентации на спиральную модель ЖЦ ПО за счет возможностей быстрого прототипирования, предоставляемых Uniface. Для описания проекта ИС используется достаточно большой набор диаграмм, конкретные

вариантыкоторого для наиболее распространенных конфигураций приведены ниже в таблице 1

Таблица 1.

Варианты диаграмм для различных конфигураций

Тип диаграммы	Обозначение	Vantage Team Builder for ORACLE	Vantage Team Builder for Informix	Vantage Team Builder for Uniface
Сущность-связь	ERD	+	+	+
Потоков данных	DFD	+	+	+
Структур данных	DSD	+	+	+
Архитектуры системы	SAD	+	+	+
Потоков управления	CSD	+	+	+
Типов данных	DTD	+	+	+
Структуры меню	MSD	+		
Последовательности блоков	BSD	+		
Последовательности форм	FSD		+	+
Содержимого форм	FCD		+	+
Переходов состояний	STD	+	+	+
Структурных схем	SCD	+	+	+

При построении всех типов диаграмм обеспечивается контроль соответствия моделей синтаксису используемых методов, а также контроль соответствия одноименных элементов и их типов для различных типов диаграмм.

При построении DFD обеспечивается контроль соответствия диаграмм различных уровней декомпозиции. Контроль за правильностью верхнего уровня DFD осуществляется с помощью матрицы списков событий (ELM). Для контроля за декомпозицией составных потоков данных используется несколько вариантов их описания: в виде диаграмм структур данных (DSD) или в нотации БНФ (форма Бэкуса-Наура).

Для построения SAD используется расширенная нотация DFD, дающая возможность вводить понятия процессоров, задач и периферийных устройств, что обеспечивает наглядность проектных решений.

При построении модели данных в виде ERD выполняется ее нормализация и вводится определение физических имен элементов данных и таблиц, которые будут использоваться в процессе генерации физическойсхемы данных конкретной СУБД. Обеспечивается возможность определения альтернативных ключей сущностей и полей, составляющих дополнительные точки входа в таблицу (поля индексов), и мощности отношений между сущностями.

Наличие универсальной системы генерации кода, основанной на специфицированных средствах доступа к репозиторию проекта, позволяет поддерживать высокий уровень исполнения проектной дисциплины разработчиками: жесткий порядок формирования моделей; жесткая структура
и содержимое документации; автоматическая генерация исходных кодов программ и т.д. – все это обеспечивает повышение качества и надежности разрабатываемых ИС.

Для подготовки проектной документации могут использоваться издательские системы FrameMaker, Interleaf или Word Perfect. Структура и состав проектной документации могут быть настроены в соответствии с заданными стандартами. Настройка выполняется без изменения проектных решений.

При разработке достаточно крупной ИС вся система в целом соответствует одному проекту как категории Vantage Team Builder. Проект может быть декомпозирован на ряд систем, каждая из которых соответствует некоторой относительно автономной подсистеме ИС и разрабатывается независимо от других. В дальнейшем системы проекта могут быть интегрированы.

Процесс проектирования ИС с использованием Vantage Team Builder реализуется в виде 4-х последовательных фаз (стадий) – анализа, архитектуры, проектирования и реализации, при этом законченные результаты каждой стадии полностью или частично переносятся (импортируются) в следующую фазу. Все диаграммы, кроме ERD, преобразуются в другой тип или изменяют вид в соответствии с особенностями текущей фазы. Так, DFD преобразуются в фазе архитектуры в SAD, DSD – в DTD. После завершения импорта логическая связь с предыдущей фазой разрывается, т.е. в диаграммы могут вноситься все необходимые изменения.

Взаимодействие с другими средствами

Конфигурация Vantage Team Builder for Uniface обеспечивает совместное использование двух систем в рамках единой технологической среды проектирования, при этом схемы БД (SQL-модели) переносятся в репозиторий Uniface, и, наоборот, прикладные модели, сформированные средствами Uniface, могут быть перенесены в репозиторий Vantage Team Builder. Возможные рассогласования между репозиториями двух систем устраняются с помощью специальной утилиты. Разработка экранных форм в среде Uniface выполняется на базе диаграмм последовательностей форм

(FSD) после импорта SQL- модели. Технология разработки ИС на базе данной конфигурации показана на рисунке 1.

Структура репозитория (хранящегося непосредственно в целевой СУБД) и интерфейсы Vantage Team Builder являются открытыми, что в принципе позволяет интеграцию с любыми другими средствами.

Среда функционирования

Vantage Team Builder функционирует на всех основных UNIX-платформах (Solaris, SCO UNIX, AIX, HP-UX) и VMS.

Vantage Team Builder можно использовать в конфигурации "клиент-сервер", при этом база проектных данных может располагаться на сервере, а рабочие места разработчиков могут быть клиентами.

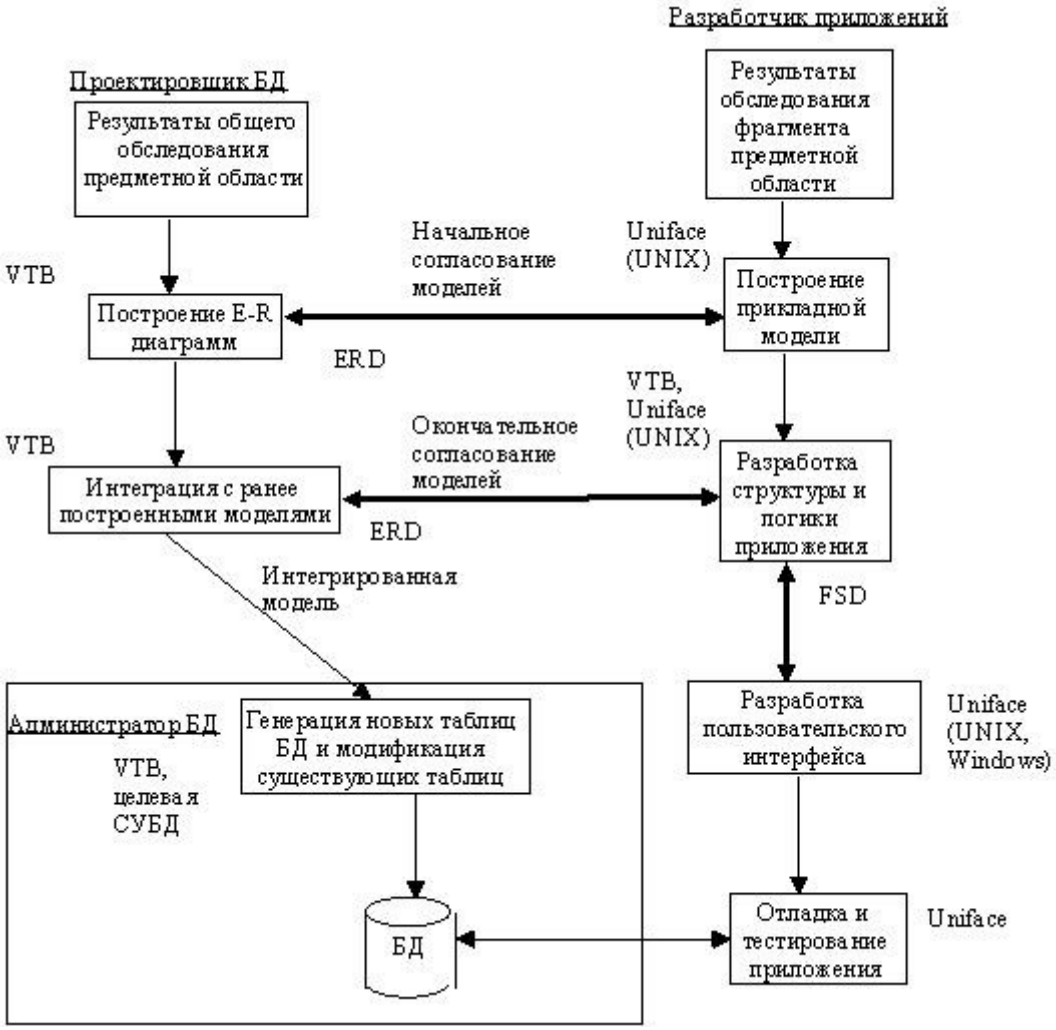


Рисунок1 - Взаимодействие Vantage Team Builder и Uniface

Вспомогательные средства поддержки жизненного цикла ПО

Средства конфигурационного управления

Цель конфигурационного управления (КУ) – обеспечить управляемость и контролируемость процессов разработки и сопровождения ПО. Для этого необходима точная и достоверная информация о состоянии ПО и его компонент в каждый момент времени, а также о всех предполагаемых и выполненных изменениях.

Для решения задач КУ применяются методы и средства обеспечивающие идентификацию состояния компонент, учет номенклатуры всех компонент и модификаций системы в целом, контроль за вносимыми изменениями в компоненты, структуру системы и ее функции, а также координированное управление развитием функций и улучшением характеристик системы.

Наиболее распространенным средством КУ является PVCS фирмы Intersolv (США), включающее ряд самостоятельных продуктов: PVCS Version Manager, PVCS Tracker, PVCS Configuration Builder и PVCS Notify.

PVCS Version Manager предназначен для управления всеми компонентами проекта и ведения планомерной многоверсионной и многоплатформенной разработки силами команды разработчиков в условиях одной или нескольких локальных сетей. Понятие "проект" трактуется как совокупность файлов. В процессе работы над проектом промежуточное состояние файлов периодически сохраняется в архиве проекта, ведутся записи о времени сохранения, соответствии друг другу нескольких вариантов разных файлов проекта. Кроме этого, фиксируются имена разработчиков, ответственных за тот или иной файл, состав файлов промежуточных версий проекта и др. Это позволяет вернуться при необходимости к какому-либо из предыдущих состояний файла (например, при обнаружении ошибки, которую в данный момент трудно исправить).

PVCS Version Manager предназначен для использования в рабочих группах. Система блокировок, реализованная в PVCS Version Manager позволяет предотвратить одновременное внесение изменений в один и тот же файл. В то же время, PVCS Version Manager позволяет разработчикам работать с собственными версиями общего файла с полуавтоматическим разрешением конфликтов между ними.

Доступ к архивам PVCS Version Manager возможен не только через сам Version Manager, но и из более чем 50 инструментальных средств, в том числе MS Visual C и MS Visual Basic, Uniface, PowerBuilder, SQL Windows, JAM, Delphi, Paradox и др.

Результатом работы PVCS Version Manager является созданный средствами файловой системы репозиторий, хранящий в компактной формевсе рабочие версии программного продукта вместе с необходимыми комментариями и метками.

PVCS Version Manager функционирует в среде MS Windows, Windows 95, Windows NT, OS/2, SunOS, Solaris, HP-UX, AIX и SCO UNIX и может исполняться на любом персональном компьютере с процессором 80386 или выше, рабочих станциях Sun, HP и IBM (RS-6000).

Другим средством конфигурационного управления является PVCS Tracker – специализированная надстройка над офисной электронной почтой, предназначенная для обработки сообщений об ошибках в продукте, доставке их исполнителям и контроля за исполнением. Интеграция с PVCS Version Manager дает возможность связывать с сообщениями те или иные компоненты проекта. Отчетные возможности PVCS Tracker включают множество разновидностей графиков и диаграмм, отражающих состояние проекта и процесса его отладки, срезы по различным компонентам проекта, разработчикам и тестировщикам. С их помощью можно наглядно показать текущее состояние работы над проектом и ее временные тенденции.

Персонал, работающий с PVCS Tracker делится на пять групп в зависимости от их обязанностей: пользователи, разработчики, группа тестирования и контроля качества, группа технической поддержки и сопровождения, управленческий персонал. Этим пяти группам персонала соответствуют пять предопределенных групп PVCS Tracker:

1. пользователи (Submitters) - имеют ограниченные права на внесение замечаний и сообщений об ошибках в базу данных PVCS Tracker;
2. разработчики (Development Engineers) - имеют право производить основные операции с требованиями и замечаниями в базе данных PVCS Tracker. Если разработчики делятся на подгруппы, то для каждой подгруппы могут быть заданы отдельные списки прав доступа;

3. тестировщики (Quality Engineers) - имеют право производить основные операции с требованиями и замечаниями;
4. сопровождение (Support Engineers) - имеют право вносить любые замечания, требования и рекомендации в базу данных, но не имеют прав по распределению работ и изменению их приоритетности и сроков исполнения;
5. руководители (Managers) - имеют право распределять работы между исполнителями и принимать решения об их надлежащем исполнении.

Руководителям разных групп могут заданы различные права доступа к базе данных PVCS Tracker.

В дополнение к этим пяти предопределенным группам, существует группа администратора базы данных и 11 дополнительных групп, которые могут быть настроены в соответствии со специфическими должностными обязанностями сотрудников, использующих PVCS Tracker.

Требование или замечание поступающее в PVCS Tracker проходит четыре этапа обработки:
1. регистрация - внесение замечания в базу данных;распределение – назначение ответственного исполнителя и сроков исполнения;
2. исполнение – устранение замечания, которое в свою очередь может вызвать дополнительные замечания или требования на дополнительные работы;
3. приемка – приемка работ и снятие их с контроля или направление на доработку.

Требования и замечания, поступающие в базу данных PVCS Tracker оформляются в виде специальной формы, которая может содержать до 18 полей выбора стандартных значений и до 12 произвольных текстовых строк. При разработке формы следует определить оптимальный набор информации, характерный для всех записей в базе данных.

Для получения содержательной информации о ходе разработки PVCS Tracker позволяет получать три типа статистических отчетов: частотные, тренды и диаграммы распределения.

Частотные отчеты содержат информацию о частоте поступающих замечаний за один час тестирования программного продукта. Однако универсального частотного отчета не существует, т.к. на оценку качества влияют тип методов тестирования, серьезность выявленных ошибок и значение дефектных модулей для функционирования всей системы. Малое число фатальных ошибок, приводящих к полной остановке разработки, хуже большого числа замечаний к внешнему виду интерфейса пользователя. Следовательно, частотные отчеты должны быть настроены на выявление какого-либо конкретного аспекта качества для того, чтобы их можно было использовать для прогнозирования окончания работ над проектом.

Тренды содержат информацию об изменениях того или иного показателя во времени и характеризуют стабильность и непрерывность процесса разработки. Они позволяют ответить на вопросы:
1. успевает ли группа разработчиков справляться с поступающими замечаниями;
2. улучшается ли качество программного продукта и какова динамика этого процесса;
3. как повлияло то или иное решение (увеличение числа разработчиков, введение скользящего графика, внедрение нового метода тестирования) на работу группы и т.п.

Диаграммы распределения – наиболее разнообразные и полезные для осуществления оперативного руководства формы отчетов. Они позволяют ответить на вопросы: какой метод тестирования более эффективен, какие модули вызывают наибольшее число нареканий, кто из разработчиков лучше справляется с конкретным типом заданий, нет ли перекоса в распределении работ между исполнителями, нет ли модулей, тестированию которых было уделено недостаточно внимания и т.д.PVCS Tracker предназначен для использования в рабочих группах, объединенных в общую сеть. В этом случае центральная база или проект PVCS Tracker находится на общедоступном сервере сети, доступ к которому реализуется посредством ODBC-драйверов, входящих в состав PVCS Tracker. Главной особенностью PVCS Tracker по сравнению с обычным приложением СУБД является его способность автоматически уведомлять пользователя о поступлении интересующей его или относящейся к его компетенции информации и гибкая система распределения полномочий внутри рабочей группы. При необходимости PVCS Tracker может использовать для уведомления удаленных членов группы электронную почту.

PVCS Tracker поддерживает групповую работу в локальных сетях и взаимодействует с СУБД dBase, ORACLE, SQL Server и SYBASE посредством ODBC.

PVCS Tracker может быть интегрирован с любой системой электронной почты, поддерживающей стандарты VIM, MAPI или SMTP.

PVCS Version Manager и PVCS Tracker окружены вспомогательными компонентами: PVCS Configuration Builder и PVCS Notify.

PVCS Configuration Builder предназначен для сборки окончательного продукта из компонент проекта. PVCS Configuration Builder позволяет описывать процесс сборки как на стандартном языке MAKE, так и на собственном внутреннем языке, имеющем существенно большие возможности. PVCS Configuration Builder позволяет осуществлять сборку программного продукта на основании файлов, хранящихся в репозитории PVCS Version Manager.

Обычная процедура сборки программного продукта с помощью PVCS Configuration Builder состоитизтрехшагов:

1. строится файл зависимостей между исходными модулями;
2. в полученный файл вносятся изменения с целью его настройки и оптимизации;
3. осуществляется сборка программного продукта из исходных модулей.

Результатом работы PVCS Configuration Builder является специальный файл, описывающий оптимальный алгоритм сборки программного продукта, построенный на основе анализа дерева зависимостей между исходными модулями.

PVCS Notify обеспечивает автоматическую рассылку сообщений об ошибках из базы данных пакета PVCS Tracker по рабочим станциям назначения. При этом используется офисная система электронной почты cc:Mail или Microsoft Mail. PVCS Notify расширяет возможности PVCS Tracker и используется только совместно с ним.

PVCS Notify настраивается из среды PVCS Tracker. Настройка включает
в себя определение интервала времени, через который PVCS Notify проверяет содержимое базы данных, определение критериев отбора записей длярассылки уведомлений, определение списков адресов для рассылки. После настройки PVCS Notify начинает работу в автономном режиме, автоматически рассылая уведомления об изменениях в базе данных PVCS Tracker.

PVCS Notify предназначен для использования в больших рабочих группах, часть членов которых хотя и доступна только через средства электронной почты, однако должна иметь оперативную информацию о требованиях на изменение программного продукта, замечаниях, ошибках, ходе и результатах его тестирования.

Результатом работы PVCS Notify являются оформленные в соответствии с одним из стандартов почтовые сообщения, готовые для рассылки посредством системы электронной почты.

Порядок выполнения практической работы:

1. Выполнить сравнительный анализ рассмотренных ранее методов проектирования применительно к обследуемому предприятию.
2. Обосновать выбор того или иного средства проектирования.

Практическая работа №6

Описание бизнес-процессов заданной предметной области

Цели: состоит в изучении основных определений процессного подхода к описанию деятельности предприятия, методов классификации его бизнес-процессов и принципов их выделения. В результате выполнения лабораторной работы студент **должен знать** основные определения процессного подхода и методы классификации бизнес – процессов предприятия, **должен уметь:**

1) разбить процесс на основные операции,
2) описать их, установить взаимосвязи, исполнителей;
3) выделить основные элементы процесса (входы, выходы, ресурсы и т.д.) и описать их;
4) выделить основных участников процесса и описать их полномочия по участию в процессе;
5) провести идентификацию процесса в различных видах классификаций и обосновать свои выводы;
6) построить тривиальную модель выбранного процесса и приложить к рисунку поясняющую спецификацию в виде таблицы.

Задание к выполнению работы

1. Изучить теоретическую часть, обращая внимание на основные определения и методы классификации процессов.
2. По указанию преподавателя из предложенного перечня процессов предприятия выбрать один.
3. Разбить процесс на основные операции, описать их, установить взаимосвязи, исполнителей
4. Выделить основные элементы процесса (входы, выходы, ресурсы и т.д.) и описать их.
5. Выделить основных участников процесса и описать их полномочия по участию в процессе
6. Провести идентификацию процесса в различных видах классификаций и обосновать свои выводы.
7. Построить тривиальную (простую, в произвольных символах) модель выбранного процесса и приложить к рисунку поясняющую спецификацию в виде таблицы.

Перечень процессов предприятия

1. Изучение рынков и потребителей.
2. Разработка видения и стратегии.
3. Разработка продуктов.
4. Маркетинг. Мониторинг рынка продукции.
5. Снабжение сырьем и оборудованием.
6. Производство.
7. Сбыт продукции.
8. Управление финансовыми ресурсами
9. Управление материальными ресурсами.
10. Развитие и управление персоналом.
11. Управление информационными ресурсами и технологиями.
12. Управление программами охраны внешней среды, здоровья**,** безопасности.
13. Управление внешними связями.
14. Управление улучшениями и изменениями.

Примечание. При декомпозиции процесса необходимо выделить не менее 8-10 операций.

ПРАКТИЧЕСКАЯ РАБОТА № 7

ПОСТРОЕНИЕ МОДЕЛИ УПРАВЛЕНИЯ КАЧЕСТВОМ ПРОЦЕССА ИЗУЧЕНИЯ МОДУЛЯ "ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ИНФОРМАЦИОННЫХ СИСТЕМ"

Цели: получение навыков построения модели управления качеством.

Форма отчета:

−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.

Время выполнения: 2 ч

Теоретические вопросы

Основные понятия качества информационной системы. Национальный стандарт обеспечения качества автоматизированных информационных систем.

Международная система стандартизации и сертификации качества продукции. Стандарты группы ISO.

Методы контроля качества в информационных системах. Особенности контроля в различных видах систем.

Автоматизация систем управления качеством разработки.

Задание № 1

Привести национальные стандарты обеспечения качества автоматизированных информационныхсистем.

Задание № 2

Охарактеризовать международную систему стандартизации и сертификации качества продукции.

Задание № 3

Описать стандарты группы ISO.

Задание № 4

Привести методы контроля качества в информационных системах.

Задание № 5

Постройте модель управления качеством процесса изучения модуля «Проектирование и разработка информационных систем».

Задание № 6

Оформить отчет.

Практическая работа № 8

РЕИНЖИНИРИНГ МЕТОДОМ ИНТЕГРАЦИИ

Цели: получение навыков реинжиниринга бизнес-процессов методом интеграции.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Порядок выполнения работы

1. Выделить основные, вспомогательные и процессы управления предметной области, согласно выбранного варианта (Приложение А) и внести результаты в форму для представления результатов (Приложение Б). Для выделения и анализа процессов используйте таблицу 1.

Типыпроцессов	Характерныепризнаки	Клиенты
Основные процессы	1.Назначениепроцессов—созданиеосновныхпродуктов 2.Результат—основнойпродукти/ илиполуфабрикатдляегоизготовленияПроцессылежатнапутисозданияосновныхпродуктов 3.Процессыдобавляюткпродуктуценностьдляпотребителя	1. Внешниеклиенты 2. Конечныепотребители 3. Внутренниеклиенты—другиепроцессы

		организации
Вспомогательные процессы	1. Назначениепроцессов—обеспечениедеятельностиосновныхпроцессов 2. Результат—ресурсыдляосновныхпроцессов 3. Деятельностьпроцессовнекасаетсяосновныхпродуктов 4. Процессыдобавляютпродуктустоимость	1. Внутренниеклиенты—другиепроцессыорганизации
Процессы управления	1. Назначениепроцесса—управлениедеятельностьювсейорганизации 2. Результат—деятельностьвсейорганизации	1. Собственники (инвесторы) 2. Потребители (клиенты) 3. Персонал (сотрудники) 4. Поставщикиисубподрядчики 5. Общество (внешняясреда)

2. Провести анализ выделенных процессов и заполнить таблицу 2.
3. Оформить отчет по лабораторной работе.
4. Представить отчет по лабораторной работе для защиты.

Приложение А
Тематика лабораторных работ

Вариант	Тема
1	Туристическая фирма
2	Молокозавод
3	Мебельная фабрика
4	Кредитование в банке
5	Проведение выставок
6	Нефтеснабжение
7	Страхование
8	Рекламное агентство
9	Строительная фирма
10	Фирма по подбору кадров

Приложение Б
Форма для представления результатов

Наименование предметной области_____

Основные процессы (перечислить, выделить не менее трех процессов)

_____

_____

Вспомогательные процессы(перечислить, выделить не менее трех процессов)

_____

_____

Процессы управления (перечислить)

_____

_____

Таблица1 - Анализ бизнес-процессов

Название	Описание и характерные	Клиенты	Результат

бизнес-процесс	признаки процесса		
Основные			
1			
2			
…			
Вспомогательные			
1			
…			
Управления			
…			

ПРАКТИЧЕСКАЯ РАБОТА № 9
РАЗРАБОТКА ТРЕБОВАНИЙ БЕЗОПАСНОСТИ ИНФОРМАЦИОННОЙ СИСТЕМЫ

Цели: получение навыков разработки требований безопасности информационной системы.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Угрозы тбезопасности информационных систем.

Обеспечение безопасности функционирования информационных систем.

Методы и средства обеспечения безопасности информационных систем.

Задание № 1.

Определите цели и задачи системы защиты информации.

Задание № 2

Перечислите факторы, влияющие на организацию системы защиты информации.

Задание № 3

Определите дестабилизирующие воздействия на информационную систему и способы их нейтрализации.

Задание № 4

Напишите программу по подсчету общей вероятности нарушения безопасности объекта, подсчитываемой по формуле

$$P = \sum_{j=1}^{k} \sum_{i=1}^{n} P_i p(j/i) q_{H1} (q_{H2} + [1 - \exp(-\alpha t_{oti})](1 - q_{H2}))$$

где к – число угроз; n – число нарушителей; P_i – вероятность появления субъекта i-го типа; p(j/i) – условная вероятность того, что субъект i-го типа выберет для реализации угрозу j-го типа; $q_{н1}$– вероятность несрабатывания средств обнаружения; $q_{н2}$ – вероятность несрабатывания средств

отражения; a – постоянная величина, характеризующая "скорость" реализации угрозы, tот – время, которым располагает субъект угрозы, если tот = 0 – угроза не реализуется.

Задание № 5

Разработайте требования безопасности информационной системы (см. практическая работа № 1).

Задание № 6

Выберите методы и средства защиты информации для исследуемой информационной системы.

Практическая работа № 10

РЕИНЖИНИРИНГ БИЗНЕС-ПРОЦЕССОВ МЕТОДОМ ГОРИЗОНТАЛЬНОГО И/ИЛИ ВЕРТИКАЛЬНОГО СЖАТИЯ

1. Определение целей перепроектирования

На основе результатов анализа окружения, логического анализа процесса и анализа по метрикам стоимости и времени следует составить список проблем существующего бизнес-процесса. Список можно проранжировать по важности. Исходя из выявленных проблем, выдвигаются цели совершенствования процесса. Цели необходимо структурировать в виде иерархии (дерева целей). Корнем дерева является одна глобальная цель, например, «Оптимизировать процесс», «Повысить эффективность процесса», «Увеличить доход». На втором уровне располагаются основные цели. Формулировки основных целей, по возможности, должны содержать значения метрик, например: «Сократить среднее время обработки заявки на 50%», «Увеличить количество обрабатываемых запросов в 5 раз». Для выделенных целей выявляются подцели, являющиеся средствами их достижения. В свою очередь, для каждой из подцелей также могут быть выдвинуты подцели и т.д. Подцели нижнего уровня представляют собой сценарии. Для их формирования необходимо проанализировать возможность применения эвристических правил реконструкции бизнеса (горизонтального сжатия, вертикального сжатия, делинеаризации, введения версий, минимизации согласований и др.). Важно не только указать на возможность использования какого-либо правила, но и указать, каким образом оно может быть применено. При этом нужно учитывать возможность использования новых информационных технологий. Например, Вы решаете, что принцип горизонтального сжатия можно применить к исследуемому процессу, объединив определенные шаги процесса за счет использования централизованной базы данных. При формировании сценариев используйте результаты логического анализа процесса и результаты оценки шагов (как УПЦ-, УОЦ- и НУЦ-действия), а также учитывайте выводы, сделанные на этапе анализа метрик стоимости и времени, о том, какие из шагов процесса являются наименее эффективными. Необходимо составить описание сценариев, например, в виде списка планируемых изменений.

2. Оценка приоритетов сценариев

Осуществляется оценка приоритетов сценариев методом анализа иерархий (МАИ). Сначала определяются локальные приоритеты целей (подцелей, сценариев). Для каждой группы подцелей, подчиненных одной цели, выставляются веса (числа в интервале от 0 до 1) так, чтобы их сумма равнялась единице. Для этого можно использовать метод парных сравнений или метод

непосредственной оценки. Затем определяются глобальные приоритеты для подцелей и сценариев. Глобальные приоритеты определяются, начиная со второго уровня вниз. Для любой подцели (сценария) локальный приоритет взвешивается, т.е. умножается на глобальный приоритет материнской подцели. Если материнских подцелей несколько, то находится сумма взвешенных приоритетов. Сделайте вывод о том, какие сценарии являются приоритетными.

3 Разработка модели нового бизнес-процесса (модели "Как должно быть")

Строится модель нового бизнес-процесса или той его части, которая изменяется. Виды используемых моделей зависят от планируемых изменений. Так, если предполагается, что изменится окружение процесса и/или интерфейс с окружением, то строится внешняя модель. Если изменяется последовательность шагов процесса, строится функциональная (событийная) модель. Если изменяются объекты, участвующие в выполнении процесса, или их взаимодействие, строится объектная модель. Диаграммы должны наглядно отобразить изменения. Можно выделить элементы модели, отражающие изменение процесса, цветом, штриховкой, шрифтом и т.д.

4. Анализ модели нового бизнес-процесса

Необходимо показать, каким образом изменятся значения метрик для нового бизнес-процесса. Для измерения стоимостных характеристик можно провести функционально-стоимостной анализ. Измерение по метрикам времени может проводиться с использованием диаграмм Ганта или имитационных моделей. По метрикам качества проводится экспертное оценивание. Следует сравнить значения метрик для существующего бизнес-процесса, для нового процесса и для целей перепроектирования. Необходимо сделать вывод, насколько улучшились характеристики процесса и достигаются ли поставленные цели. Если цели не достигаются, нужно проанализировать, в чем причина.

5. Разработка функциональных требований к информационной системе поддержки нового бизнеса

Если изменение бизнес-процесса предполагает использование новой информационной системы, то необходимо определить требования к системе на основе модели нового бизнеса. Сначала необходимо выделить акторов (пользователей) информационной системы и обязательства акторов, выполняемые с помощью ИС. Для этого осуществляется анализ модели нового бизнеса и каждому из объектов или акторов бизнеса, использующему информационную систему, сопоставляется актор модели ИС. Проверяются все обязательства выделенных акторов, выполняемые ими в новом бизнес-процессе, и выписываются те из них, которые предполагают взаимодействие с ИС. Проанализировав все выписанные обязательства, следует определить прецеденты информационной системы, с помощью которых они будут реализованы. Взаимодействие выделенных прецедентов ИС с акторами необходимо представить в виде диаграммы вариантов использования языка UML. Затем составляется высокоуровневое описание прецедентов, отражающее функции ИС. Описание может содержать не только внешние функции по взаимодействию с пользователями, но и внутренние функции системы.

Варианты индивидуального задания

1. Продажа туристического продукта

2. Выпуск газеты

3. Кредитование владельцев частных предприятий

4. Дипломирование студентов вузов

5. Предоставление доступа к местной телефонной сети

6. Ремонт квартиры

7. Аттестация муниципальных служащих

8. Организация выставки-ярмарки

9. Изготовление шкафа-купе на заказ

10.Страхование квартиры и домашнего имущества

11. Ремонт автомобилей

12.Проведение праздничных мероприятий (свадеб, юбилеев и т.д.)

13.Пошив верхней одежды

14.Проведение рекламных компаний

15.Оказание услуг по операциям с недвижимостью

16.Гостиничное обслуживание

17.Издание печатной продукции

18.Продажа и ремонт компьютеров

19.Производство и продажа мебели на заказ

20.Трудоустройство

21.Организация обучения и консультирования

22.Оказание жилищно-коммунальных услуг

23.Оказание услуг по автоперевозкам (пассажирским и/или грузовым)

24.Организация спортивных мероприятий (турниров, игр и т.д.)

25.Изготовление кондитерских изделий (тортов, пирожных)

26.Оказание медицинских услуг

27.Оказание маркетинговых услуг

28.Организация выборных компаний

29.Производство, продажа и сопровождение программной продукции

30.Строительство гаражей, садовых домиков и т.д.

ПРАКТИЧЕСКАЯ РАБОТА № 11

ПРОЕКТИРОВАНИЕ СПЕЦИФИКАЦИИ ИНФОРМАЦИОННОЙ СИСТЕМЫ ИНДИВИДУАЛЬНОМУ ЗАДАНИЮ

Цели: получение навыков проектирования спецификации информационной системы.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Требования к информационной системе.

Методы анализа и спецификации требований.

Концептуальные требования.

Функциональные требования.

Технические требования.

Технологии и методологии управления требованиями.

Задание № 1

Сформулировать цели и задачи создания информационной системы (см. практическая работа № 1). Охарактеризовать вид информационной системы, её назначение, используемые в работе системы данные. Сформулировать концептуальные требования к информационной системе.

Задание № 2

Дать характеристику типового объекта автоматизации (организации, предприятия) для которого создаётся и на котором будет внедрена информационная система. Описать автоматизируемые бизнес-процессы.

Задание № 3

Сформулировать требования к системе в целом. Описать структуру информационной системы. Перечислить функциональныеподсистемы.

Задание № 4

Сформулировать функциональные требования. Описать требования к функциям и задачам, выполняемым системой. Описать назначение и состав функций каждой из подсистем.

Задание № 5

Описать предметную область. Разработать концептуальную модель данных предметной области. Сформулировать требования к информационному обеспечению системы.

Задание № 6

Сформулировать требования к программному обеспечению системы. Описать требования к пользовательскому интерфейсу. Сформулировать технические требования к реализации и режимам работы информационнойсистемы.

Задание № 7

Используя полученные результаты, подготовить документ «Техническое задание на создание информационной системы», включающий в себя полное описание концептуальных, функциональных и технических требований к создаваемой системе.

ПРАКТИЧЕСКАЯ РАБОТА № 12

РАЗРАБОТКА ОБЩЕГО ФУНКЦИОНАЛЬНОГО ОПИСАНИЯ ПРОГРАММНОГО СРЕДСТВА ПО ИНДИВИДУАЛЬНОМУ ЗАДАНИЮ

Цели: получение навыков разработки общего функционального описания программного средства.

Форма отчета:

−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Виды информационных систем, их назначение и состав.

Технологии разработки информационных систем.

Методологии разработки программного обеспечения.

Процесс разработки программного обеспечения.

Управление разработкой программного обеспечения.

Проектирование информационных систем.

Этапы проектирования.

Задачи и результаты проектирования.

Задание № 1

Подготовить исходные данные для разработки информационной системы (см. практическая работа № 1). Исходными данными для планированияявляются:

- общее описание некоторой информационной системы (назначение, область применения, решаемые задачи, технологические особенности реализации ивнедрения);
- ограничения и условия разработки (требования заказчика, возможности команды разработчиков, сроки разработки, бюджет проекта ит.д.).

Задание № 2

Составить эскизный план разработки информационной системы.

Задание № 3

Составить документ «Технический проект» с описанием проектных решений (архитектура системы, логическая структура базы данных, решения по реализации пользовательского интерфейса и т.д.).

Задание № 4

Составить документ «План тестирования» с описанием методики тестирования и контрольныхтестов.

Задание № 5

Составить документ «План ввода информационной системы в эксплуатацию».

ПРАКТИЧЕСКАЯ РАБОТА № 13

РАЗРАБОТКА РУКОВОДСТВА ПО ИНСТАЛЛЯЦИИ ПРОГРАММНОГО СРЕДСТВА ПО ИНДИВИДУАЛЬНОМУ ЗАДАНИЮ

Цели: получение навыков разработки руководства по инсталляции программного средства.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Понятие дистрибутива. Виды дистрибутивов.

Типы инсталляции программного обеспечения.

Руководство по инсталляции программного средства.

Задание № 1

Разработать руководство по инсталляции программного средства для заданной информационной системы (см. практическая работа №1).

ПРАКТИЧЕСКАЯ РАБОТА № 14
РАЗРАБОТКА РУКОВОДСТВА ПОЛЬЗОВАТЕЛЯ ПРОГРАММНОГО СРЕДСТВА ПО ИНДИВИДУАЛЬНОМУ ЗАДАНИЮ

Цели: получение навыков разработки руководства пользователя программного средства.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Перечень и комплектность документов на информационные системы согласно ЕСПД и ЕСКД. Задачи документирования.

Проектная документация. Техническая документация. Отчетная документация.
Пользовательская документация. Маркетинговая документация.

Задание № 1

Разработать руководство пользователя программного средства (см. практическая работа № 1).

ПРАКТИЧЕСКАЯ РАБОТА№15
ИЗУЧЕНИЕ СРЕДСТВ АВТОМАТИЗИРОВАННОГО ДОКУМЕНТИРОВАНИЯ

Цели: получение навыков разработки документации по ИС и использования средств автоматизированного документирвоания.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

«Техническое задание на создание (развитие или модернизацию) системы» (далее – ТЗ на АС) является основным документом, определяющим требования и порядок создания (развития или модернизации – далее создания) автоматизированной системы, в соответствии

с которым проводится разработка АС и ее приемка при вводе в действие.

Включаемые в ТЗ на АС требования должны соответствовать современному уровню развития науки и техники и не уступать аналогичным требованиям, предъявляемым к лучшим современным отечественным и зарубежным аналогам. Задаваемые в ТЗ на АС требования не должны ограничивать разработчика системы в поиске и реализации наиболее эффективных технических, технико-экономических и других решений.

ТЗ на АС содержит следующие разделы, которые могут быть разделены на подразделы:

1) общие сведения;
2) назначение и цели создания (развития) системы;
3) характеристика объектов автоматизации;
4) требования к системе;
5) состав и содержание работ по созданию системы;
6) порядок контроля и приемки системы;
7) требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие;
8) требования к документированию;
9) источники разработки.

Требования к содержанию разделов – в ГОСТ 34.602–89 «Информационная технология. Техническое задание на создание автоматизированных систем»

В зависимости от вида, назначения, специфических особенностей объекта автоматизации и условий функционирования системы допускается оформлять разделы ТЗ в виде приложений, вводить дополнительные, исключать или объединять подразделы ТЗ. В ТЗ на части системы не включают разделы, дублирующие содержание разделов ТЗ на АС в целом.

ТЗ на АС оформляют на листах формата А4 без рамки, основной надписи и дополнительных граф к ней. Номера листов (страниц) проставляют, начиная с первого листа, следующего за титульным листом, в верхней части листа (над текстом, посередине) после обозначения кода ТЗ на АС.

На титульном листе (форма – в ГОСТ 34.602–89) помещают подписи заказчика, разработчика и согласующих организаций, которые скрепляют гербовой печатью. При необходимости титульный лист оформляют на нескольких страницах. Подписи разработчиков ТЗ на АС и должностных лиц, участвующих в согласовании и рассмотрении проекта ТЗ на АС, помещают на последнем листе.

Задание

Разработать техническое задание на автоматизацию управления деятельностью предприятия согласно ГОСТ 34.602–89 «Техническое задание на создание (развитие или модернизацию) системы» на примере гипотетического предприятия по выбранной теме.

Описать логическую модель БД ИС средствами BPWin.

В создаваемой ИС реализовать одну из задач:

- Управление документацией отдела системы качества.
- Управление процессом подготовки и переподготовки специалистов.
- Управление процессом закупки материалов, комплектующих.
- Управление процессом реализации готовой продукции.

Предметная область выбирается из списка

1. Ж/Д вокзал. Учет продажи билетов.
2. Поликлиника. Учет больных.
3. Информация в отделе кадров.
4. Учет движения товаров на складе.
5. Гостиница. Размещение клиентов.
6. Банк. Работа с клиентами.
7. Составление расписания занятий.
8. Налоговая инспекция. Учет уплаты налогов.
9. Страховая компания. Заключение договоров.
10. Ведение библиотечного фонда.
11. Городская телефонная сеть. Учет междугородных переговоров.
12. Театр. Продажа билетов.
13. Кадровое агенство.
14. Компьютерный сервисный центр.
15. Риэлторская фирма. Учет движения квартир.
16. Туристическое агентство.
17. Салон красоты. Оказание услуг.
18. Ресторан. Обслуживание посетителей.
19. Ателье пошива одежды. Учет заказов.
20. Химчистка. Учет заказов.
21. Прокат видеокассет. Работа с клиентами.
22. Поступление и продажа товаров в магазине вычислительной техники.
23. Библиотека. Выдача книг.
24. Мебельный салон. Учет заказов.
25. Аптека. Поступление и продажа лекарств.
26. Работа с клиентами на торговой фирме, занимающейся реализацией автомобилей.
27. Оптовый склад. Заключение договоров с поставщиками.
28. Деканат. Учет успеваемости в период сессии.
29. Продажа авиабилетов.
30. Фитнес-клуб.
31. Приемная комиссия ВУЗа. Учет сдачи приемных экзаменов.
32. Бухгалтерия. Учет основных фондов.

Порядок выполнения лабораторной работы

1. Выбрать предметную область и реализуемую задачу. Определить основные требования к ИС. Построить логическую модель БД ИС средствами BPWin.
2. Составить техническое задание на разработку ИС в соответствии с ГОСТ 34.602–89 «Информационная технология. Техническое задание на создание автоматизированных систем».

ПРАКТИЧЕСКАЯ РАБОТА № 16
ПОСТРОЕНИЕ ДИАГРАММЫ ВАРИАНТОВ ИСПОЛЬЗОВАНИЯ И ДИАГРАММЫ ПОСЛЕДОВАТЕЛЬНОСТИ И ГЕНЕРАЦИЯ КОДА

Цель: ознакомиться с методологией моделирования информационных систем на основе языка UML.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Универсальный язык моделирования UML.

Понятие диаграммы.

Виды диаграмм.

Основные элементы диаграммы вариантов использования.

Основные элементы диаграммы последовательности.

Задание № 1. Ознакомиться с методологией построения диаграммы вариантов использования основе языка UML.

Задание № 2. Проанализируйте пример построения диаграммы вариантов использования.

Пример. Магазин видеопродукции

Магазин продает видеокассеты, DVD-диски, аудиокассеты, CD-диски и т.д., а также предлагает широкой публике прокат видеокассет и DVD-дисков.

Товары поставляются несколькими поставщиками. Каждая партия товара предварительно заказывается магазином у некоторого поставщика и доставляется после оплаты счета. Вновь поступивший товар маркируется, заносится в базу данных и затем распределяется в торговый зал или прокат.

Видеоносители выдаются в прокат на срок от 1 до 7 дней. При прокате с клиента взимается залоговая стоимость видеоносителя. При возврате видеоносителя возвращается залоговая стоимость минус сумма за прокат. Если возврат задержан менее чем на 2 дня, взимается штраф в размере суммы за прокат за 1 день* кол-во дней задержки. При задержке возврата более чем на 2 дня – залоговая сумма не возвращается. Клиент может взять одновременно до 4 видеоносителей (прокат-заказ). На каждый видеоноситель оформляетсяквитанция.

Клиенты могут стать членами видео-клуба и получить пластиковые карточки. С членов клуба не берется залог (за исключением случая описанного ниже), устанавливается скидка на ставку проката и покупку товаров. Члены клуба могут делать предварительные заказы на подбор видеоматериалов для проката или покупки.

Каждый член клуба имеет некоторый стутус. Первоначально – "новичок". При возврате всрок 5 прокат-заказов, статус меняется на "надежный". При задержке хотя бы одного видеоносителя более чем на 2 дня, статус "новичок" или "надежный" меняется на "ненадежный" и клиенту высылается предупреждение. При повторном нарушении правил статус меняется на "нарушитель". Члены клуба со статусом "надежный" могут брать до 8 видеоносителей единовременно, все остальные – 4. С членов клуба со статусом "нарушитель" берется залоговая сумма.

Клиенты при покупке товара или получении видеоносителя в прокат могут расплачиваться наличными или кредитной картой.

Прокатные видеоносители через определенное количество дней проката списываются и утилизируются по акту. Списываются также товары и прокатные видеоносители, у которых обнаружился брак.

На рисунке 1 приведена диаграмма прецедентов для рассматриваемого примера. В этом примере можно выделить следующие субъекты и соответствующие им прецеденты:

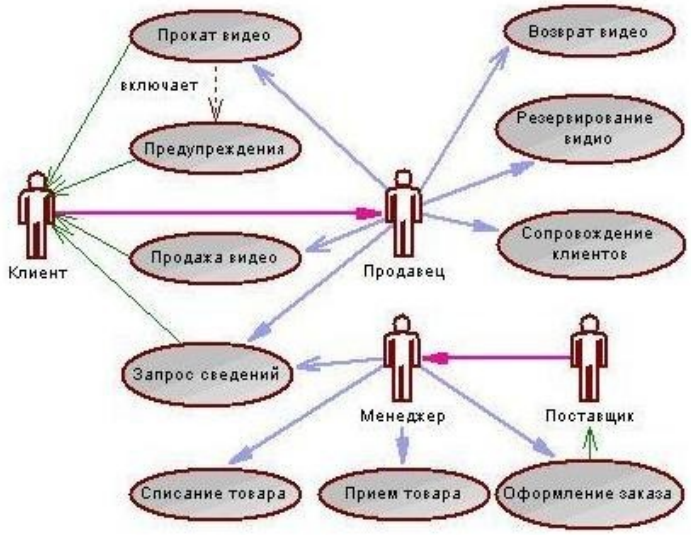


Рисунок 1

Менеджер изучает рынок видеопродукции, анализирует продажи (прецедент "Запрос сведений"), работает с поставщиками: составляет заявки на поставки товара (прецедент "Оформление заказа"), оплачивает и принимает товар (прецедент "Прием товара"), списывает товар (прецедент "Списаниетовара").

Продавец – работает с клиентами: продает товар (прецедент "Продажа видео"), оформляет членство в клубе (прецедент "Сопровождение клиентов"), резервирует (прецедент "Резервирование видио"), выдает в прокат (прецедент "Прокат видео") и принимает назад видеоносители (прецедент "Возврат видео"), отвечает на вопросы клиента (прецедент "Запрос сведений").

Поставщик – оформляет документы для оплаты товара (прецедент "Оформление заказа"), поставляет товар (прецедент "Прием товара"))

Клиент – покупает(прецедент "Продажа видео"), берет на прокат и возвращает видеоносители (прецеденты "Прокат видео" и "Возврат видео"), вступает в клуб (прецедент "Сопровождение клиентов"), задает вопросы (прецедент "Запрос сведений").

Последние два субъекта Поставщик и Клиент не будут иметь непосредственного доступа к разрабатываемой системе (второстепенные субъекты), однако именно они являются основным источником событий, инициализирующих прецеденты, и получателями результата работы прецедентов

От прецедента "Прокат видео" к прецеденту "Предупреждения" установлено отношение включения на том основании, что каждый выданный видеоноситель должен быть проверен на своевременный возврат и, в случае необходимости, выдано предупреждение клиенту.

Дальнейшее развитие модели поведения системы предполагает спецификацию прецедентов. Для этого традиционно используют два способа. Первый – описание с помощью текстового документа. Такой документ описывает, что должна делать система, когда субъект инициировал прецедент. Типичное описание содержит следующие разделы:

– краткоеописание;
– участвующиесубъекты;
– предусловия, необходимые для инициированияпрецедента;
– поток событий (основной и, возможно, подпотоки,альтернативный);
– постусловия, определяющие состояние системы, по достижении которого прецедент завершается.

Описательная спецификация прецедента "Прокат видео"

Раздел	Описание
Краткоеописание	Клиент желает взять на прокат видеокассету или диск, которые снимаются с полки магазина или были предварительно зарезервированы клиентом. При условии, что у клиента нет невозвращенных в срок видеоносителей, сразу после внесения платы фильм выдается напрокат. Если невозвращенные в срок видеоносители есть, клиенту выдается напоминание о просроченном возврате
Субъекты	Продавец, Клиент
Предусловия	В наличие имеются видеокассеты или диски, которые можно взять напрокат. У клиентов есть клубные карточки. Устройство сканирования работает правильно. Работники за прилавком знают, как обращаться с системой
Основнойпоток	Клиент может назвать номер заказа или взять видеоноситель с полки. Видеоноситель и членская карточка сканируются, и продавцу не сообщается никаких сведений о задержках, так, что он не задает клиенту соответствующих вопросов. Если клиент имеет статус <надежный>, он может взять до 8 видеоносителей, во всех остальных случаях – до 4-х. Если статус клиента определен как <нарушитель>, его просят внести задаток. Клиент расплачивается наличными или кредитной картой. После получения суммы, информация о наличии фильмов обновляется и видеоносители передаются клиенту вместе с квитанциями на прокат. О прокате каждого видеоносителя делается отдельная запись с указанием идентификационного номера клиента, даты проката, даты возврата, идентификационного номера продавца, полученнойсуммы. Прецедент генерирует предупреждения о просроченном возврате клиенту, если видеофильм не был возвращен в течение двух дней по истечении даты возврата и изменяет статус клиента на <ненадежный> (первое нарушение) или <нарушитель> (повторное нарушение)

Альтернативный поток	У клиента нет членской карточки. В этом случае прецедент <Сопровождение клиента> может быть активизирован для выдачи новой карточки. Видеофильмы не выдаются, поскольку у клиента есть невозвращенные в срок видеоносители. Попытка взять напрокат слишком много видеоносителей. Видеоноситель или кредитная карта не могут быть отсканированы из-за их повреждения У клиента не хватило наличных или платеж по кредитной карте отклонен
Постусловия	Видеофильмы сданы напрокат, и база данных соответствующим образом обновлена

Задание № 3. Постройте диаграмму вариантов использования для выбранной информационной системы (практическая работа №11).

Задание № 4. Ознакомиться с методологией построения диаграммы последовательности основе языка UML.

Задание № 5. Проанализируйте пример построения диаграммы последовательности (рисунок 2).

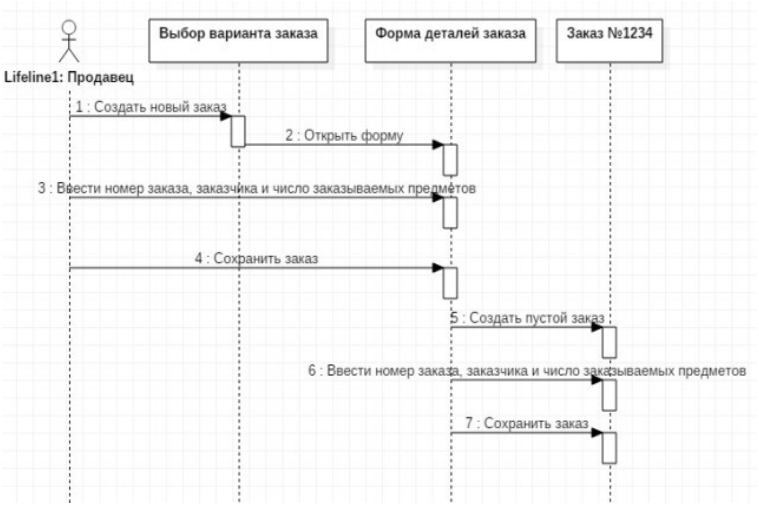


Рисунок 2

Пример

Вводзаказа.

Действующее лицо«Продавец».

Сообщения:

– создать новыйзаказ;

– открытьформу;

– ввести номер заказа, заказчика и число заказываемыхпредметов;

– сохранитьзаказ;

– создать пустойзаказ;

– ввести номер заказа, заказчика и число заказываемыхпредметов;

– сохранитьзаказ.

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 другойобъект.

Окончательный вид диаграммы последовательности представлен на рисунке 3.

Задание № 6. Постройте диаграмму последовательности для выбранной информационной системы (практическая работа № 11).

Задание № 7. Оформите отчет.

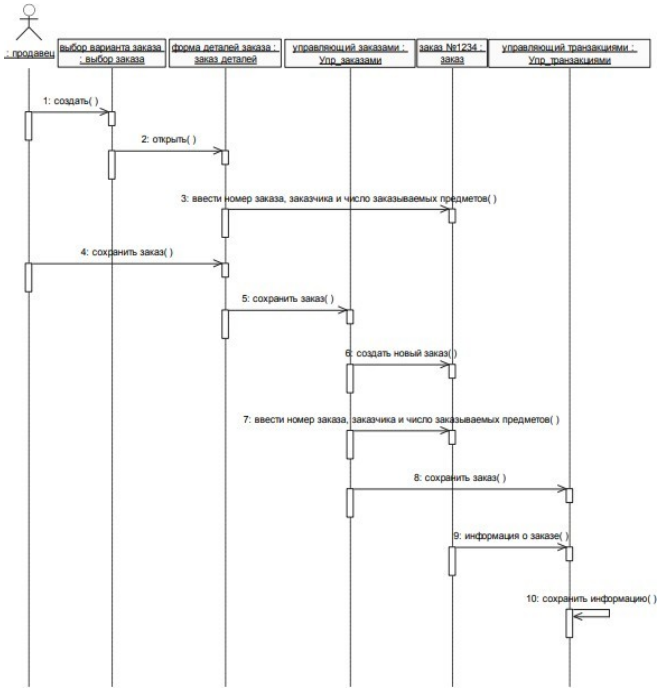


Рисунок 3

ПРАКТИЧЕСКАЯ РАБОТА № 17

ПОСТРОЕНИЕ ДИАГРАММЫ КООПЕРАЦИИ И ДИАГРАММЫ РАЗВЕРТЫВАНИЯ И ГЕНЕРАЦИЯКОДА

Цель: ознакомиться с методологией моделирования информационных систем на основе языка UML.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Универсальный язык моделирования UML.

Понятие диаграммы.

Виды диаграмм.

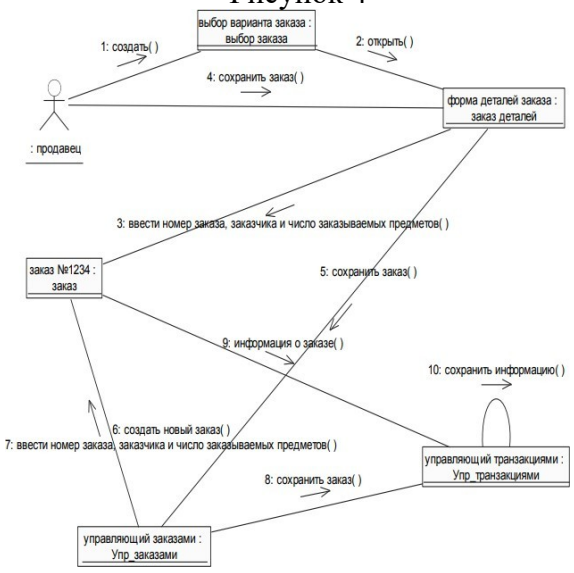
Основные элементы диаграммы кооперации.

Основные элементы диаграммы развертывания.

Задание № 1. Ознакомиться с методологией построения диаграммы кооперации основе языка UML.

Задание № 2. Проанализируйте пример построения диаграммы кооперации (рисунок 4).

Рисунок 4



Задание № 3. Постройте диаграмму кооперации для выбранной информационной системы (практическая работа № 11).

Задание № 4. Ознакомиться с методологией построения диаграммы развертывания основе языка UML.

Задание № 5. Проанализируйте пример построения диаграммы развертывания.

Примеры построения диаграмм развертывания

Фрагмент диаграммы развертывания с соединениями между узлами показан на рисунке 5.

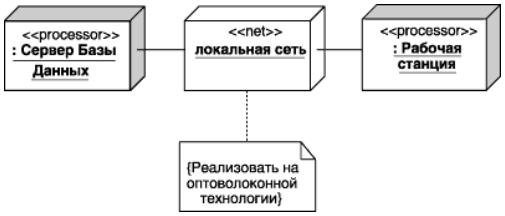


Рисунок 5

Диаграмма развертывания с отношением зависимости между узлом и развернутыми на нем компонентами приведена на рисунке 6.

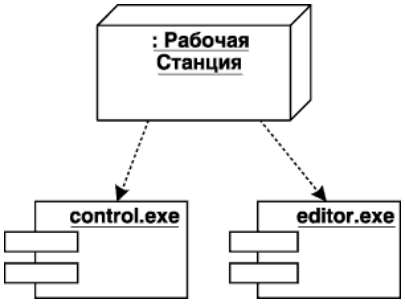


Рисунок 6

Диаграмма развертывания для системы мобильного доступа к корпоративной базе данных изображена на рисунке 7.

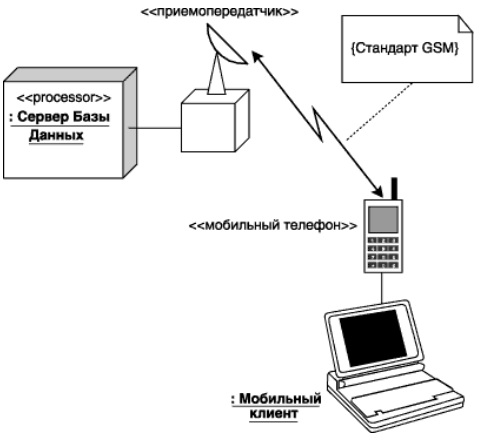


Рисунок 7

Задание № 6. Постройте диаграмму развертывания для выбранной информационной системы (практическая работа №11).

Задание № 7. Оформите отчет.

ПРАКТИЧЕСКАЯ РАБОТА №18

Построение диаграммы Деятельности, диаграммы Состояний и диаграммы Классов и генерация кода

Цель работы: научиться строить диаграмму вариантов использования.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретическая справка:

Визуальное моделирование в UML можно представить как некоторый процесс поуровневого спуска от наиболее обшей и абстрактной концептуальной модели исходной системы к логической, а затем и к физической модели соответствующей программной системы. Для достижения этих целей вначале строится модель в форме так называемой диаграммы вариантов использования (usecasediagram), которая описывает функциональное назначение системы или, другими словами, то, что система будет делать в процессе своего функционирования. Диаграмма вариантов использования является исходным концептуальным представлением или концептуальной моделью системы в процессе ее проектирования и разработки.

Разработка диаграммы вариантов использования преследует цели:

- Определить общие границы и контекст моделируемой предметной области на начальных этапах проектирования системы.
- Сформулировать общие требования к функциональному поведению проектируемой системы.
- Разработать исходную концептуальную модель системы для ее последующей детализации в форме логических и физических моделей.
- Подготовить исходную документацию для взаимодействия разработчиков системы с ее заказчиками и пользователями.

Суть данной диаграммы состоит в следующем: проектируемая система представляется в виде множества сущностей или актеров, взаимодействующих с системой с помощью так называемых вариантов использования. При этом актером (actor) или действующим лицом называется любая сущность, взаимодействующая с системой извне. Это может быть человек, техническое устройство, программа или любая другая система, которая может служить источником воздействия на моделируемую систему так, как определит сам разработчик. В свою очередь, вариант использования (usecase) служит для описания сервисов, которые система предоставляет актеру. Другими словами, каждый вариант использования определяет некоторый набор действий, совершаемый системой при диалоге с актером. При этом ничего не говорится о том, каким образом будет реализовано взаимодействие актеров с системой.

Состав диаграммы UseCase

Диаграмма вариантов использования состоит из актеров, для которых система производит действие и собственно действия UseCase, которое описывает то, что актер хочет получить от системы. Актер обозначается значком человечка, а UseCase - овалом. Дополнительно в диаграммы могут быть добавлены комментарии.

Виды взаимодействий

Между актерами и вариантами использования могут быть различные виды взаимодействия. Основные виды взаимодействия следующие:

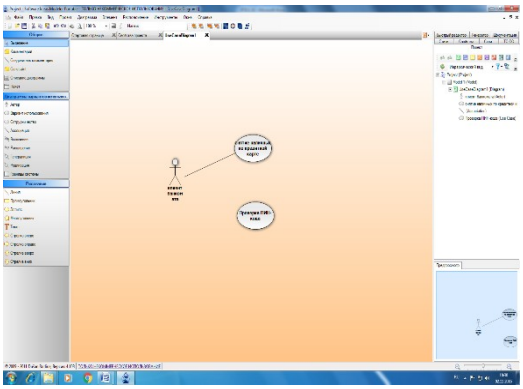
- **Простая ассоциация** - отражается линией между актером и вариантом использования (без стрелки). Отражает связь актера и варианта использования. На рисунке между актером администратор и вариантом использованияпросматривать заказ.
- **Направленная ассоциация** - то же что и простая ассоциация, но показывает, что вариант использования инициализируется актером. Обозначается стрелкой.
- **Наследование** - показывает, что потомок наследует атрибуты и поведение своего прямого предка. Может применяться как для актеров, так для вариантов использования.
- **Расширение** (extend) - показывает, что вариант использования расширяет базовую последовательность действий и вставляет собственную последовательность. При этом в отличие от типа отношений "включение" расширенная последовательность может осуществляться в зависимости от определенных условий.
- **Включение** (include) - показывает, что вариант использования включается в базовую последовательность и выполняется всегда (на рисунке не показан).

Существуют и другие виды взаимодействия, но они менее важны и реже применяются.

Задание 1. Построить диаграмму вариантов использования модели вариантов использования банкомата.

Выполните следующие действия:

1. Добавить актера с именем Клиент банкомата.
2. Добавить вариант использования Снятие наличных по кредитной карте
3. Добавить направленную ассоциацию от бизнес-актера Клиент Банкомата к варианту использования Снятие наличных по кредитной карте
4. Добавить вариант использования Проверка ПИН-кода.



5. Добавить актера с именем Банк.
6. Добавить вариант использования Получение справки о состоянии счета.
7. Добавить вариант использования Блокирование кредитной карточки.
8. Добавить направленную ассоциацию от бизнес-актера Клиент Банкомата к варианту использования Получение справки о состоянии счета.
9. Добавить направленную ассоциацию от варианта использования Снятие наличных по кредитной карточке к сервису Банк.
10. Добавить направленную ассоциацию от варианта использования Получение справки о состоянии счета к сервису Банк.
11. Добавить отношение зависимости со стереотипом «include», направленное от варианта использования Снятие наличных по кредитной карте к варианту использования Проверка Пин-кода.
12. Добавить отношение зависимости со стереотипом «include», направленное от варианта использования Получение справки о состоянии счета к варианту использования Проверка Пин-кода.

13. Добавить отношение зависимости со стереотипом «extend», направленное от варианта использования Блокирование кредитной карточки к варианту использования Проверка Пин-кода.

Задание 2. Построить диаграмму вариантов использования.
Имеются следующие данные:

- четыре действующих лица: Клиента банка, Банк, Кассира и Оператора,
- пять вариантов использования: Снять наличные, Перевести деньги со счета, Положить деньги на счет, Пополнить запас денег и Подтвердить пользователя,
- три зависимости, и отношения между действующими лицами и вариантами использования.

Варианты использования: Снять наличные, Перевести деньги со счета, Положить деньги на счет - требуют включения идентификации клиента в системе. Это поведение может быть выделено в новый вариант использования включения, называемый Подтвердить пользователя. Базовые варианты использования не зависимы от метода, используемого для идентификации. Поэтому он инкапсулируется (скрывается) в варианте использования включения. С точки зрения базовых вариантов использования не имеет значение производится ли идентификация с помощью магнитной карты или сканированием сетчатки глаза. Они только зависят от результата выполнения варианта использования Подтвердить клиента.

Задание для самостоятельной работы: Построить диаграмму вариантов использования на основе вербальной модели информационной системы «Компьютерный клуб»

Контрольные вопросы:

1. Какие цели преследует разработка диаграммы использования?
2. Для чего нужна диаграмма вариантов использования?
3. Из чего состоит диаграмма вариантов использования?
4. Виды взаимодействия используемые в диаграмме вариантов использования?
5. Из чего состоит созданная вами диаграмма?

ПРАКТИЧЕСКАЯ РАБОТА № 19

ПОСТРОЕНИЕ ДИАГРАММЫ КОМПОНЕНТОВ И ГЕНЕРАЦИЯ КОДА

Цель: ознакомиться с методологией моделирования информационных систем на основе языка UML.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Универсальный язык моделирования UML. Понятие диаграммы.

Виды диаграмм.

Основные элементы диаграммы компонентов. Основные элементы диаграммы развертывания.

Задание № 1. Ознакомиться с методологией построения диаграммы компонентов основе языкаUML.

Задание № 2. Проанализируйте пример построения диаграммы компонентов. Выделяем компоненты, отображаем зависимости между ними.

Фрагмент диаграммы компонентов с отношениями зависимости и реализации показан на рисунке8.

Графическое изображение отношения зависимости между компонентами приведено на рисунке9.

На рисунке 10 показано графическое изображение зависимости между компонентом и классами.

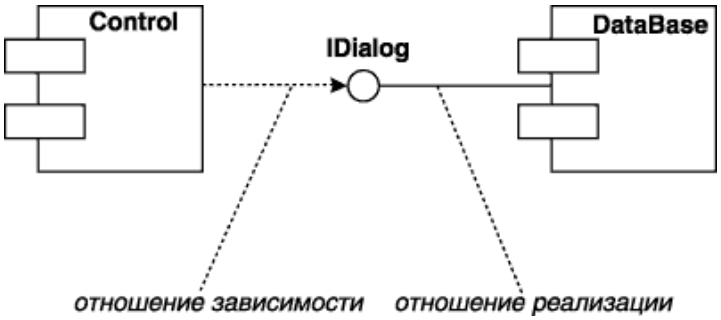


Рисунок8

Рисунок9

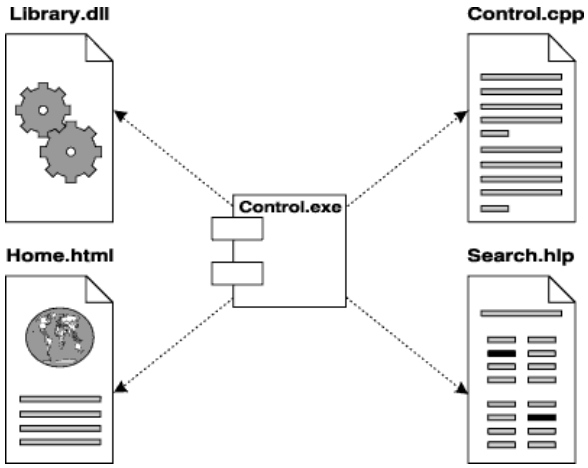


Рисунок 10

Задание № 3. Постройте диаграмму компонентов для выбранной информационной системы (практическая работа № 11).

Задание № 4. Оформите отчет.

Практическая работа №20
ОБОСНОВАНИЕ ВЫБОРА ТЕХНИЧЕСКИХ СРЕДСТВ

Цель:научиться определять необходимые технические средства.
Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

На выбор комплекса технических средств разработанной системы влияет её функциональность и методы хранения информации.

На рисунке 1 приведен пример диаграммы модулей (фрагмент) для системы составления линейного кроссворда. В таблице 1 приведено описание модулей.

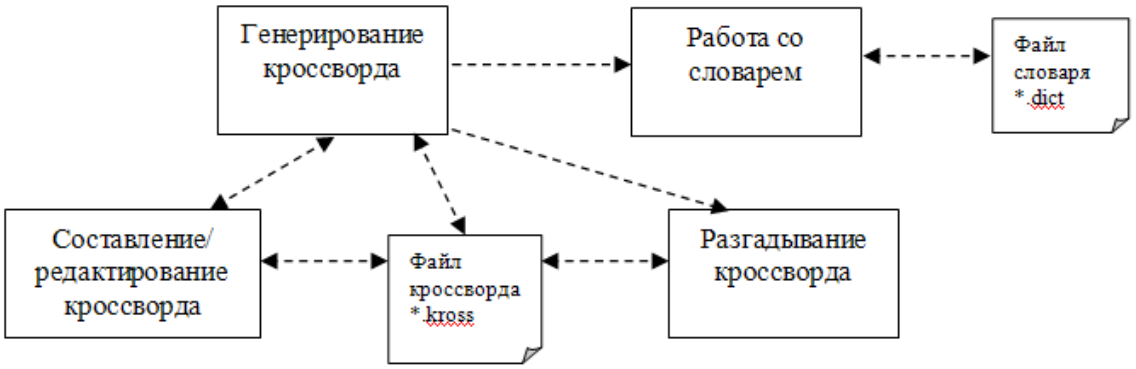


Рисунок 1 – Диаграмма модулей системы (фрагмент)

Таблица 1 – Описание модулей системы (фрагмент)

Название модуля	Название файла	Назначение	Подсистема
Генерирование кроссворда	Generation.c	Отвечает за автоматическое составление кроссворда при заданных параметрах	Подсистема генерирования кроссворда
Работа со словарем	Dict.c	Отвечает за редактирование словаря с понятиями и их определениями	Подсистема работы со словарем
Составление/ редактирование кроссворда	Rewrite_Cross.c	Отвечает за редактирование кроссворда	Подсистема ручного составления
Разгадывание кроссворда	Unravel.c	Отвечает за разгадывание кроссворда	Подсистемаразгадывания кроссворда

Периферийная техника необходимая для решения задач, поставленных перед системой, должна выполнять следующие функции:

- ввод данных с клавиатуры;
- управление курсором манипулятором типа «мышь»;
- вывод информации на дисплей и на печать;
- передача информации в БД.

Согласно вышеперечисленным требованиям и результатам расчетов быстродействия и ресурсов для нормального функционирования системы составить перечень необходимых технических средств.

Практическая работа №21
СТОИМОСТНАЯ ОЦЕНКА ПРОЕКТА

Цель:научиться определять стоимостную оценку проекта и определить сроки окупаемости внедряемой ИС при указанных затратах на проект внедрения.

Форма отчета:

−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Описание проведение стоимостной оценки:

Разработанная система предназначается для использования заместителем директора школы по АХЧ муниципального образовательного учреждения «Северодвинская общеобразовательная школа №23».

Основными задачами стоимостной оценки являются расчет периода окупаемости затрат на внедрение, разработку автоматизированной системы и оценка экономического эффекта, ожидаемого в результате внедрения разработанного проекта.

Результатом внедрения автоматизированной системы может быть прямой или косвенный экономический эффект. Под прямым экономическим эффектом понимается положительный результат от внедрения, выраженный в получении прибыли (реальных денег), чаще достигаемый за счет фактического сокращения персонала, т. к. комплекс работ, выполняемый на ЭВМ, позволяет отказаться от дополнительного увеличения штата сотрудников.

Косвенный (или условный) экономический эффект - это эффект, получаемый за счет внедрения программных средств, которые изменяют содержание работы персонала в сторону оперативности и надежности, но не приводят к обязательному фактическому сокращению штата сотрудников.

Может также наблюдаться социальный эффект (т.е. как скажется внедрение автоматизированной системы на работу заместителя директора школы по АХЧ с персоналом, с организациями – подрядчиками и т.д.) и характерный для автоматизированных систем технологический эффект (т.е. снижение трудоемкости за счет ускорения процедуры поиска необходимой информации, повышение качества принимаемых решений за счет автоматизации расчетов, производимых ранее вручную).

В данной методике оценки представлены следующие расчеты:

5) Расчет трудоемкости реализации функций до и после внедрения ИС;
6) Расчет затрат на реализацию функций ИС;
7) Расчет затрат на проектирование и внедрение системы
8) Расчет периода окупаемости и затрат на внедрение и разработку ИС.

Внедрение данного проекта даст возможность значительно снизить трудозатраты и существенно повысить качество работы.

Показателем экономической эффективности при внедрении автоматизированного рабочего места заместителя директора школы по административно-хозяйственной части является годовой экономический эффект и срок окупаемости (обратный показатель).

Годовой экономический эффект от использования адаптированной системы определяется по формуле: $Э = (З1-З2)- \Delta К*ЕН$, (5.1)

З1, З2 – текущие затраты на обработку экономической информации по базовому и новому вариантам, руб.;

ЕН – нормативный коэффициент эффективности капитальных вложений.

е К – капитальные дополнительные вложения предприятия, связанные с внедрением результатов дипломного проекта, руб.

Сначала рассчитываем трудоемкость реализации функций до внедрения и после внедрения информационной системы. Расчет трудоемкости реализации функций до внедрения информационной системы сведен в таблице 5.1

Таблица 5.1 - «Стоимость работ до эксплуатации ИС»

Шаг процесса (функция)	Трудоемкость (чел-мин)	Повторяемость в год	Трудоемкость в год (чел-мин)	Трудоемкость в год (чел-дней)

Внесение данных о поступивших/ выданных материальных ценностях в книгу складского учета (1 строка)	0,5	630	315	0,7875
Регистрация товарных накладных	0,5	65	32,5	0,08125
Внесение данных о поступивших материальных ценностях (спецодежда)вжурналучета спецодежды	0,5	10	5	0,0125
Составление ежемесячной ведомости выдачи МЦ на нужды учреждения	60	12	720	1,8
Заполнение раздела инвентарной карточки ОС "Индивидуальная характеристика"	25	15	375	0,9375
Заполнение раздела инвентарной карточки ОС "Сведения о перемещениях"	20	20	400	1
Заполнение раздела инвентарной карточки ОС "Ремонт/модернизация"	20	140	2800	7
Внесение данных материальных ценностей в паспорт помещения	5	30	150	0,375
Составление общей заявки на закупку материальных ценностей от сотрудников	40	1	40	0,1
Составление акта на списание мягкого и хозяйственного инвентаря	20	24	480	1,2
Составление актов на списание основных средств	40	10	400	1
Составление акта о бое, ломе посуды	20	12	240	0,6
Составление списка фактических остатков по категории МЦ	90	4	360	0,9
Составление списка фактических остатков по всем МЦ, находящихся в учреждении	120	1	120	0,3
Внесение записи о прохождении инструктажа сотрудником школы	0,2	120	24	0,06
Выписка сотрудников, не прошедших инструктаж (по различным причинам)	30	2	60	0,15
Заполнение/корректировка карты аттестации рабочего места	30	5	150	0,375
Составление списка сотрудников для прохождения медосмотра	0,5	1	0,5	0,00125
Составление графика осмотра помещений	30	6	180	0,45
Запись в журнале работ о задании для сотрудника в период каникул	5	60	300	0,75
Отслеживание выполнения работы персоналом (с приведением испол. МЦ)	20	60	1200	3
Отслеживание договоров,	15	20	300	0,75

заключенных с подрядчиками и актов выполненных работ				
Итого в человеко-днях				21,63

Примечание: При расчете трудоемкости выполнения функций в год в человеко-часах следует исходить из нормативной продолжительности рабочего дня 480 минут

♣ перерывов в работе по 10 минут после каждых 50-ти минут работы на ЭВМ (указанные нормативы перерывов предусмотрены действующими санитарно-гигиеническими нормами). Таким образом, время реальной работы человека в день составляет 400 минут.

Все функции приведенные в таблице 5.1 выполняет заместитель директора школы по АХЧ, что составляет 21,63 ч/д в год. Эти функции являются только небольшой частью его должностных обязанностей – они связаны, в основном, с материально-техническим обеспечение учебного процесса. Кроме этих функций в обязанности заместителя директора школы по АХЧ входит: текущий контроль за санитарно-гигиеническим состоянием здания, газовой котельной, сооружений, классов, учебных кабинетов, мастерских, спортивной площадки и школьного стадиона, иного имущества школы, а также буфета в соответствии с требованиями

норм и правил безопасности жизнедеятельности; поиск и заключениепредварительных договоров с организациями, которые могут поставлять МЦ или предоставлять услуги более качественно или дешевле, контролирует выполнение обязательств со стороны организаций коммунального хозяйства, руководит работами по благоустройству, озеленению и уборке территории школы (летом); ведет учет рабочего времени технического и обслуживающего персонала школы и т.д.

Расчет трудоемкости реализации функций усовершенствованной информационной системы приведен в таблице 5.2.

Таблица 5.2 - Расчет трудоемкости реализации функций усовершенствованной ИС

Шаг процесса (функция)	Трудоемкость (чел-мин)	Повторяемость в год	Трудоемкость в год (чел-мин)	Трудоемкость в год (чел-дней)
Запись справочной информации в БД	0,1	30	3	0,0075
Внесение данных о поступивших/ выданных материальных ценностях в базу данных	0,1	630	63	0,1575
Регистрация товарных накладных	0,1	65	6,5	0,01625
Внесение данных о поступивших материальных ценностях (спецодежда) в базу данных	0,1	10	1	0,0025
Составление ежемесячной ведомости выдачи МЦ на нужды учреждения	0,2	12	2,4	0,006
Заполнение раздела инвентарной карточки ОС "Индивидуальная характеристика" в БД	5	15	75	0,1875
Заполнение раздела инвентарной карточки ОС "Сведения о перемещениях" БД	2	20	40	0,1
Заполнение раздела инвентарной карточки ОС	2	140	280	0,7

"Ремонт/модернизация" БД				
Внесение данных материальных ценностей в паспорт помещения и в БД	10	30	300	0,75
Ввод данных заявок от сотрудников в БД	0,2	20	4	0,01
Составление общей заявки на закупку материальных ценностей от сотрудников	0,2	1	0,2	0,0005
Составление акта на списание мягкого и хозяйственного инвентаря	20	24	480	1,2
Составление актов на списание основных средств	5	10	50	0,125
Составление акта о бое, ломе посуды	5	12	60	0,15
Составление списка фактических остатков по категории МЦ	0,2	4	0,8	0,002
Составление списка фактических остатков по всем МЦ, находящихся в учреждении	0,2	1	0,2	0,0005
Внесение записи о прохождении инструктажа сотрудником школы	0,1	120	12	0,03
Выписка сотрудников, не прошедших инструктаж (по различным причинам)	0,2	2	0,4	0,001
Заполнение/корректировка карты аттестации рабочего места	30	5	150	0,375
Составление списка сотрудников для прохождения медосмотра	0,1	1	0,1	0,00025
Составление графика осмотра помещений	15	6	90	0,225
Запись в журнале работ о задании для сотрудника в период каникул ивБД	5,5	60	330	0,825
Отслеживание выполнения работы персоналом	3	60	180	0,45
Отслеживание договоров, заключенных с подрядчиками и актов выполненных работ	0,5	20	10	0,025
Итого в человеко-днях				5,339

Проанализировав таблицы 5.1 и 5.2, уже можно сделать вывод о том, что годовая трудоемкость выполнения функций уменьшится за счет снижения времени, затрачиваемого на составление производных отчетов (в основном, за счет автоматизации расчетов остатков, но и за счет других отчетов: список сотрудников, не прошедших инструктаж, на медосмотр, общей заявки на МЦ) и таких документов, ведомость выдачи материальных ценностей на нужды учреждения, актов списания (разного типа), т.к. эти документы формируются на основании ранее внесенных данных практически автоматически, а ведомость полностью автоматизирована.

Таким образом, автоматизация некоторых процессов позволит заместителю директора школа по АХЧ более быстро решать поставленные перед ним задачи. Например, решения о закупках материальных ценностей, остатки которых приближаются к критическим, или которые необходимы сейчас для обеспечения непрерывности учебного процесса, получения мгновенной информации о запрашиваемом бухгалтерией основном средстве, решения о списании основных средств вследствие разных причин и т.д.

При расчете трудоемкости реализации функций после внедрения автоматизированной системы учитывается ввод справочной информации в систему, которой не будет хватать после первоначальной загрузки данных.

На следующем этапе необходимо произвести расчет затрат на реализацию функций спроектированной системы, исходя из трудозатрат до и после внедрения системы, а так же из среднего количества рабочих дней в месяце (22) и среднего оклада заместителя директора школы по АХЧ - 8250**.**

Расчет затрат на реализацию функций системы представлен в таблице 5.3.

Таблица 5.3 - «Расчет затрат на реализацию функций ИС»

Статья затрат	Затраты до внедрения (руб.)	Затраты после внедрения (руб.)
Расходы по оплате труда, в т.ч.	19061,44	4704,99
-Основная з/пл	8111,25	2002,13
-Дополнительная з/пл, в т.ч.	10950,19	2702,87
Льготы крайнего севера 80%	6489,00	1601,70
Районный коэффициент 40%	3244,50	800,85
Вознаграждение за работу (стимулирующие надбавки)15%	1216,69	300,32
Начисления на заработную плату 30% к сумме расходов на оплату труда (статья 1)	5718,43	1411,50
ИТОГО текущих затрат ($З_1$ и $З_2$ соответственно):	24779,87	6116,49

Начисления на заработную плату в 2013 году составляют 30%, и, по сути, являются страховыми взносами во внебюджетные фонды. Эта сумма распределяется между фондами, согласно установленного процентного соотношения:

4. в Пенсионный фонд Российской Федерации уплачивается 22 процента;
5. в Фонд социального страхования Российской Федерации - 2,9 процента;
6. в Федеральный фонд обязательного медицинского страхования - 3,1 процента;
7. в территориальные фонды обязательного медицинского страхования - 2,0 процента.

Согласно ч. 1 ст. 58.2 Закона N 212-ФЗ данный тариф применяется в течение 2012 - 2013 гг. В 2012 г. взносы в указанном размере перечисляются за каждого работника на суммы выплат, не превышающие 512 000 руб. (Постановление Правительства РФ от 24.11.2011 N 974), которая исчисляется нарастающим итогом с начала календарного года (ч. 4 ст. 8 Закона N 212-ФЗ).

Путем калькуляции затрат, направляемых на внедрение системы, рассчитываем 3.таблице 5.4 дополнительные капитальные вложения учреждения (ΔК).

Таблица 5.4 - «Расчет дополнительных затрат предприятия»

№ п/п	Наименование статьи	Сумма (руб)
1	Затраты на приобретение и монтаж оборудования, всего в т.ч.	0
2	Затраты на программное обеспечение	0
3	Затраты на оплату времени работы вычислительных и коммуникационных ресурсов, всего в т.ч.	7 400
	машинное время	6400
	время работы в Internet	1 000
4	Затраты на обучение персонала, всего в т.ч.	3750
5	Расходы по оплате труда проектировщика в т.ч.	30 500
	Основная заработная плата	12 200
	Дополнительная заработная плата (120%), в т.ч.	
	- с учетом льгот Крайнего севера(80%)	
	- с учетом районного коэффициента (40%)	14 640
	Начисления на заработную плату (30%)	3660
	ИТОГО:	41 650

Примечание:

и Затраты на приобретение и монтаж оборудования, на программное обеспечение нулевые, так как в сентябре 2012 года в кабинете директора школа по АХЧ был установлен новый компьютер (стоимость составила 29850 руб.) с достаточно высокими эксплуатационными характеристиками и со всем необходимым программным обеспечением согласно приобретенной лицензии.

и Затраты на оплату времени вычислительных ресурсов складываются из фактически затраченного времени на разработку проекта: 2 месяца (4 недели в каждом) * 5 дней в неделю * 8 часов = 320 часов и стоимости часа машинного времени 20 рублей, а так же времени работы в сети Internet – 40 часов по 25 рублей (Интернет использовался, в основном, для разработки аналитического раздела проекта, используется расценки Интернет кафе).

и Затраты на обучение персонала определяются исходя из того, что стоимость часа обучения на компьютере составляет 250 руб. Время, требуемое для обучения заместителя директора школы по АХЧ для работы с информационной системой составляет пять дней по 3 часа.

4, Расходы на оплату труда проектировщика. Основная заработная плата проектировщика за 2 месяца составит 30500 руб. (по окладу 6100 руб. плюс все начисления).

После произведения расчетов:

$З_1$ = 24779,87 $З_2$ = 6116,49, ΔК = 41650, ЕН равный 0,15, т.е. 15% (данные получены с помощью справочной системы «Консультант Плюс»).

Подставив все полученные значения в формулу 5.1, получим сумму годового экономического эффекта от внедрения адаптированной системы:

Э = (24779,87 -6116,49) –41650* 0,15 = 12415,88 (руб.)

Так как не планируется дополнительно закупка технического и программного обеспечения, то сумма ежегодной амортизации не рассчитывается.

Для расчета периода окупаемости нужно вычислить показатель - индекс доходности, который рассчитывается по формуле:

индекс доходности = ДП / ИС (5.2),

где ДП – сумма денежного потока (в настоящей стоимости) за весь период эксплуатации программного продукта (до начала инвестиций);

ИС - сумма инвестиционных средств, направляемых на реализацию проекта;

В качестве сумм денежного потока может служить показатель годового экономического эффекта, приведенного к настоящей стоимости путем дисконтирования.

Если значение индекса доходности меньше или равно 1, проект не принесет дополнительного дохода инвестору в текущем году и только когда индекс доходности становится больше 1, с этого периода начинает поступать доход от инвестиций.

Доход от внедрения АС это накопленный годовой экономический эффект за минусом остаточной суммы затрат, это абсолютное выражение сумм дохода или убытков, получаемых предприятием в i –м году эксплуатации АС.

Расчет дохода от внедрения системы представлен в таблице 5.5.

Таблица 5.5 - «Расчет дохода от внедрения системы»

Год эксплуатации ИС	**1**	**2**	**3**	**4**	**5**
Остаточная сумма затрат (тыс. руб.)	41,65	41,65	41,65	41,65	41,65
Сумма ежегодной амортизации на оборудование и ПО	0,00	0,00	0,00	0,00	0,00
Накопленный годовой экономический эффект	12,42	46,82	81,22	115,62	150,02
Индекс доходности	29,81%	112,40%	195,00%	277,59%	360,18%
Доход от внедрения ИТУ	-29,23	5,17	39,57	73,97	108,37

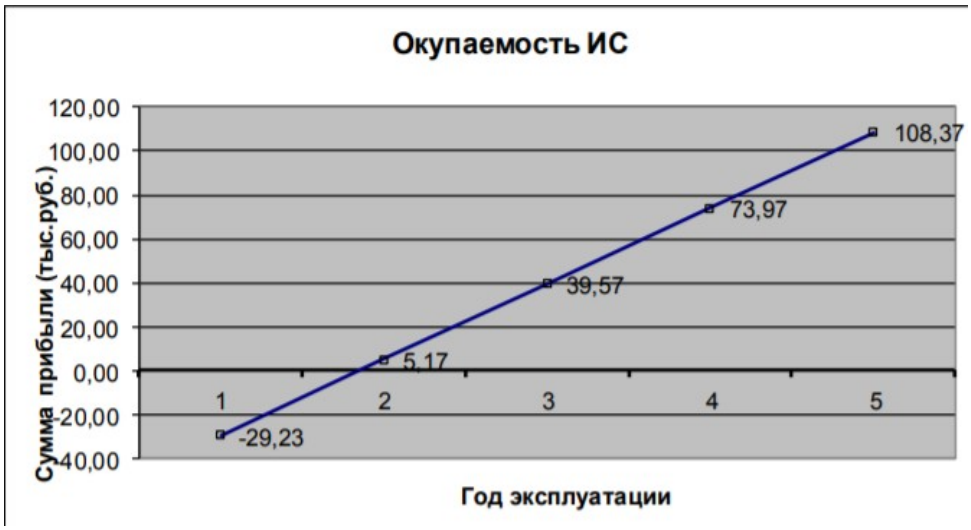


Рис.5.1 - График окупаемости затрат

Исходя из расчетов, система окупится на второй год эксплуатации. При финансово-экономических расчетах, связанных с инвестированием средств, процессы наращения и дисконтирования стоимости могут осуществляться как по формуле простых, так и по формуле сложных процентов. Поскольку инвестирование данного проекта менее трех лет, для расчета суммы дисконта применяется формула простых процентов.

Расчет дисконтированных сумм по простым процентам осуществляется по формуле:

D=Э/(1+n*i) (5.3)

где Э – годовой экономический эффект;

в – продолжительность инвестирования (год эксплуатации);

i – фиксированная ставка рефинансирования, равная 8,25% (величина ставки получена на сайте Центра экономического анализа и экспертизы. Сайт - http://www.assessor.ru/forum/index.php?t=1599).

Расчет суммы дисконта по простым процентам представлен в таблице 5.6.

Таблица 5.6 - «Сумма дисконта по простым процентам»

Год эксплуатации ИС	1	2	3	4	5	6
Остаточная сумма затрат	41,65	41,65	41,65	41,65	41,65	35,90
Сумма ежегодной амортизации	0,00	0,00	0,00	0,00	0,00	0,00
Дисконтированная сумма ежегодной амортизации	0,00	0,00	0,00	0,00	0,00	0,00
Сумма денежного потока	12,42	12,42	12,42	12,42	12,42	12,42
Дисконтированная сумма денежного потока	11,47	12,41	12,42	12,42	12,42	12,42
Накопленная дисконтированная сумма денежного потока	11,47	23,88	36,30	48,72	61,14	73,56
Индекс доходности	27,54%	57,33%	87,15%	116,97%	146,79%	204,90%
Доход от внедрения ИС	-30,18	-17,77	-5,35	7,07	19,49	37,66

График окупаемости затрат представлен на рисунке 2.

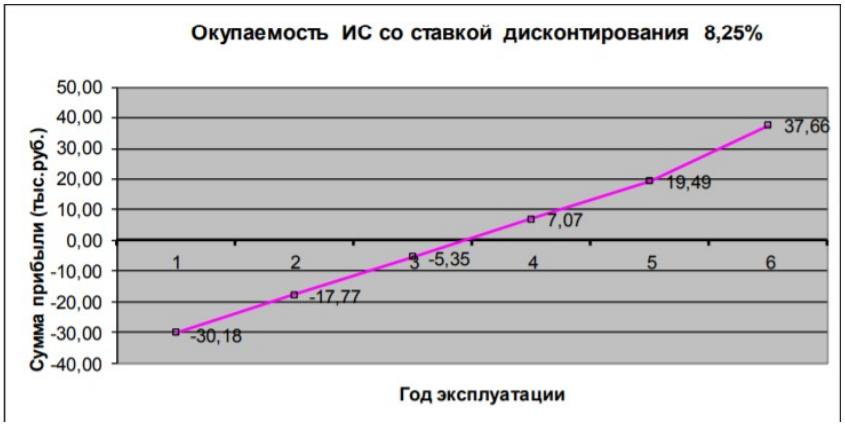


Рисунок 2 - График окупаемости ИС с учетом дисконтирования простым процентом

Таким образом, годовой экономический эффект от внедрения системы (Э), исходя из расчетов составил 12415,88 рублей. Автоматизированная система окупается на второй год

эксплуатации (без приведения денежного потока к настоящей стоимости) и на третий год при расчете настоящей стоимости по формуле простых процентов.

Полученный экономический эффект является косвенным, так как увольнение персонала в школе не предполагается, и эффект выражается не конкретной суммой прибыли, а снижением трудоемкости выполнения рабочих функций.

Технологический эффект от внедрения разрабатываемой системы состоит в заметном сокращении времени на формирование выходных документов: ведомости выдачи МЦ на нужды учреждения, актов списания, инвентарной карточки, списков. Заместителю директора школы по АХЧ не придется затрачивать значительное время на формирования списков остатков для разных категорий МЦ и общего списка остатков МЦ в учреждении – всё это результативно сделает автоматизированная система с отсутствием ошибок на основе заранее введенных данных. Данные вводятся в систему через экранные формы (приложение Е). для быстроты и удобства пользователя применяются маски ввода для дат на всех таких полях. При формировании текущих документов даты их составления или ввода записей в табличные части этих документов проставляются автоматически, применяя функцию текущей даты (листы книги складского учета, журнала учета спецодежды, акты на списание). Для устранения ошибок также в экранных формах применяются поля со списками (фиксированными – например, поле вид помещения в экранной форме «Список помещений», и на основе справочных данных – например, поле «Материальная ценность» в экранной форме «Список материальных ценностей к учету»).

Так же обеспечивается защита информации. Если раньше хранящиеся только на бумажном носителе данные могли быть утеряны или по прошествии времени (из-за внесения изменений, небрежного обращения) стать нечитабельными, то хранение информации в электронном виде само по себе устранит такого рода проблемы.

При внедрении проекта будет получен и социальный эффект (в данном случае для заместителя директора школы в виде повышения производительности труда), состоящий в том, что ответственное за материально-техническое обеспечение учебного процесса лицо, затратив меньшее количество времени на оформление документов, сможет более качественно выполнить свои обязанности. Высвободившееся время может быть перераспределено на выполнение других функций по должностным инструкциям работника.

Практическая работа №22
ПОСТРОЕНИЕ И ОБОСНОВАНИЕ МОДЕЛИ ПРОЕКТА

Метод IDEF1, разработанный Т.Рэмей (T.Ramey), также основан на подходе П.Чена и позволяет построить модель данных, эквивалентную реляционной модели в третьей нормальной форме. В настоящее время на основе совершенствования методологии IDEF1 создана ее новая версия - методология IDEF1X. IDEF1X разработаная с учетом таких требований, как простота изучения и возможность автоматизации. IDEF1X-диаграммы используются рядом распространенных CASE-средств (в частности, ERwin, Design/IDEF).

Сущность в методологии IDEF1X является независимой от идентификаторов или просто независимой, если каждый экземпляр сущности может быть однозначно идентифицирован без определения его отношений с другими сущностями. Сущность называется зависимой от идентификаторов или просто зависимой, если однозначная идентификация экземпляра сущности зависит от его отношения к другой сущности (рис. 3.1).

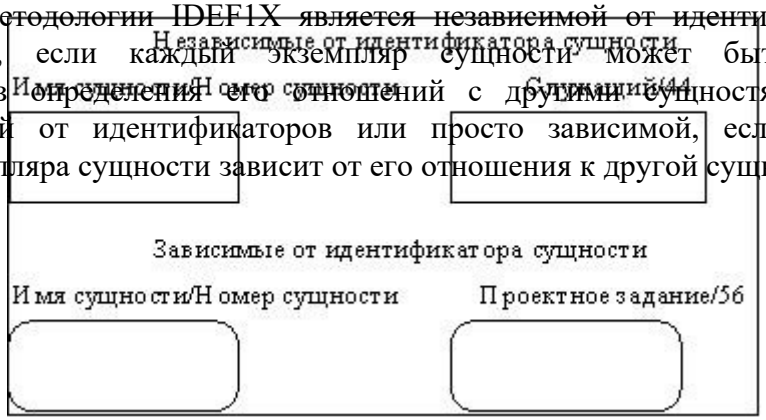


Рис. 3.1

Каждой сущности присваивается уникальное имя и номер, разделяемые косой чертой "/" и помещаемые над блоком.

вязь может дополнительно определяться с помощью указания степени или мощности (количества экземпляров сущности-потомка, которое может существовать для каждого экземпляра сущности-родителя). В IDEF1X могут быть выражены следующие мощности связей:

- В каждый экземпляр сущности-родителя может иметь ноль, один или более связанных с ним экземпляров сущности-потомка;
- В каждый экземпляр сущности-родителя должен иметь не менее одного связанного с ним экземпляра сущности-потомка;
- В каждый экземпляр сущности-родителя должен иметь не более одного связанного с ним экземпляра сущности-потомка;
- В каждый экземпляр сущности-родителя связан с некоторым фиксированным числом экземпляров сущности-потомка.

Если экземпляр сущности-потомка однозначно определяется своей связью с сущностью-родителем, то связь называется идентифицирующей, в противном случае - неидентифицирующей.

Связь изображается линией, проводимой между сущностью-родителем и сущностью-потомком с точкой на конце линии у сущности-потомка. Мощность связи обозначается как показано на рис. 3.2 (мощность по умолчанию - N).

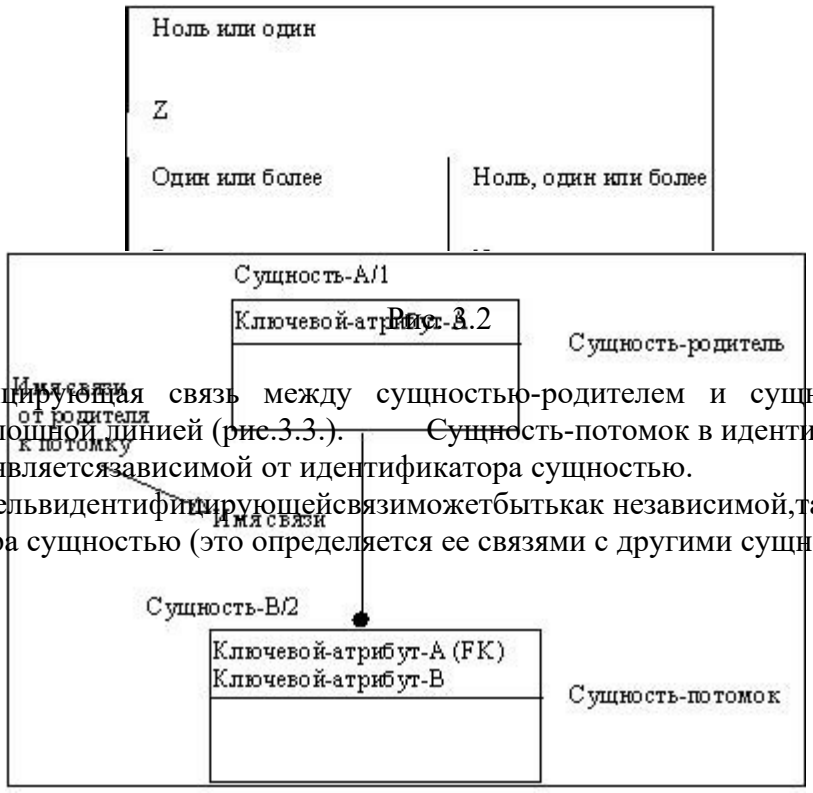


Рис. 3.2

Идентифицирующая связь между сущностью-родителем и сущностью-потомком изображается сплошной линией (рис.3.3.). Сущность-потомок в идентифицирующей связи является зависимой от идентификатора сущностью. Сущность-родитель в идентифицирующей связи может быть как независимой, так и зависимой от идентификатора сущностью (это определяется ее связями с другими сущностями).

Рис. 3.3

Пунктирная линия изображает неидентифицирующую связь (рис. 3.4.). Сущность-потомок в неидентифицирующей связи будет независимой от идентификатора, если она не является также сущностью-потомком в какой-либо идентифицирующей связи.

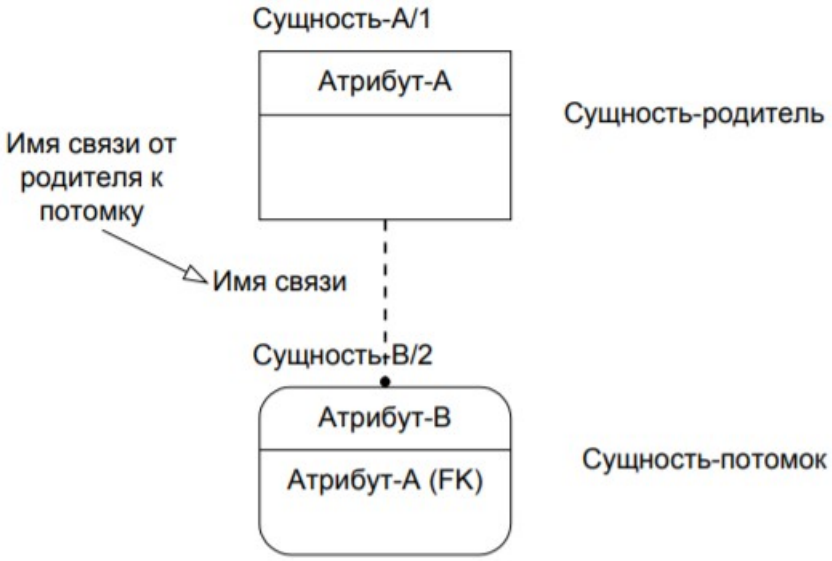


Рис. 3.4

Атрибуты изображаются в виде списка имен внутри блока сущности. Атрибуты, определяющие первичный ключ, размещаются наверху списка и отделяются от других атрибутов горизонтальной чертой (рис. 3.5).

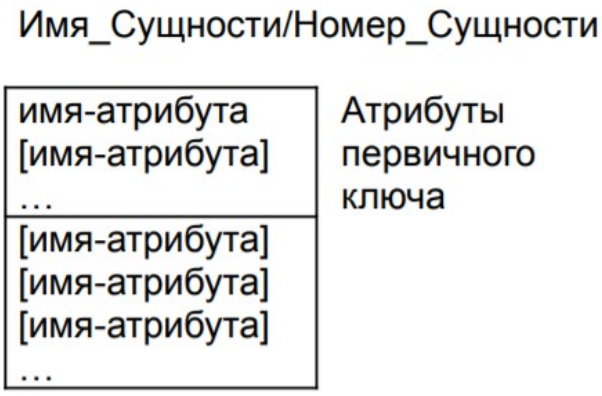


Рис. 3.5

Сущности могут иметь также внешние ключи (Foreign Key), которые могут использоваться в качестве части или целого первичного ключа или неключевого атрибута. Внешний ключ изображается с помощью помещения внутрь блока сущности имен атрибутов, после которых следуют буквы FK в скобках (рис. 3.6.).

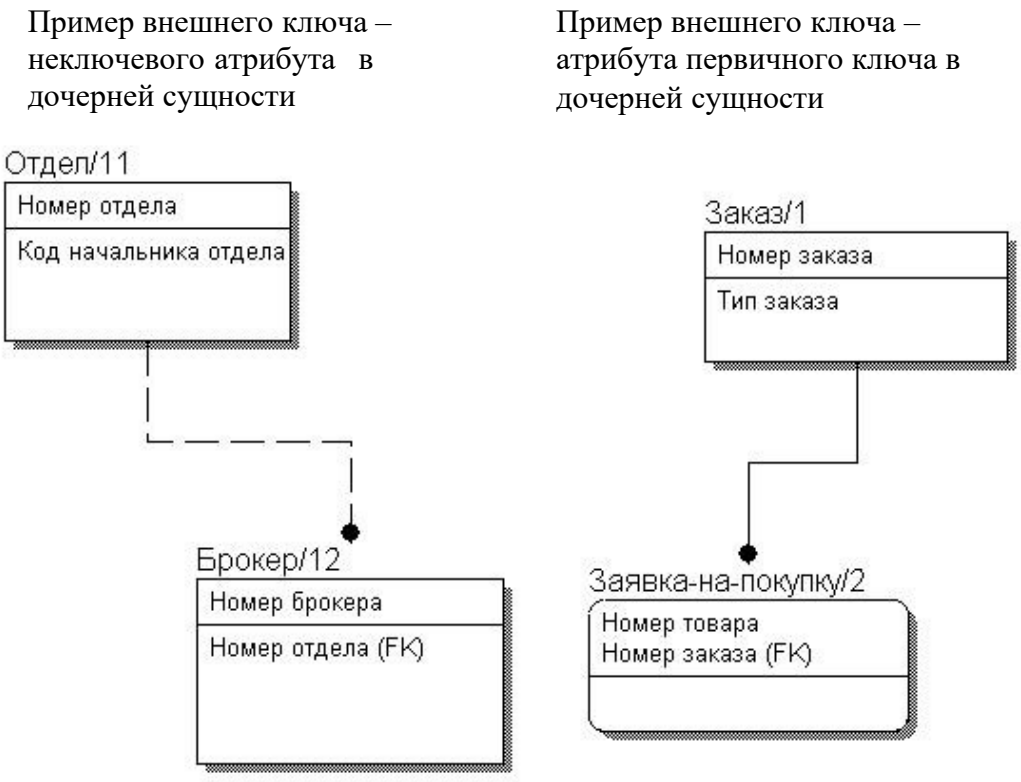


Рис. 3.6

3. Работаспакетом Computer Associates ERWin

Запуск программы осуществляется через меню Пуск -> Программы -> Computer Associates ERWin4.0. После загрузки программы появится главное окно программы (рис. 3.7.).

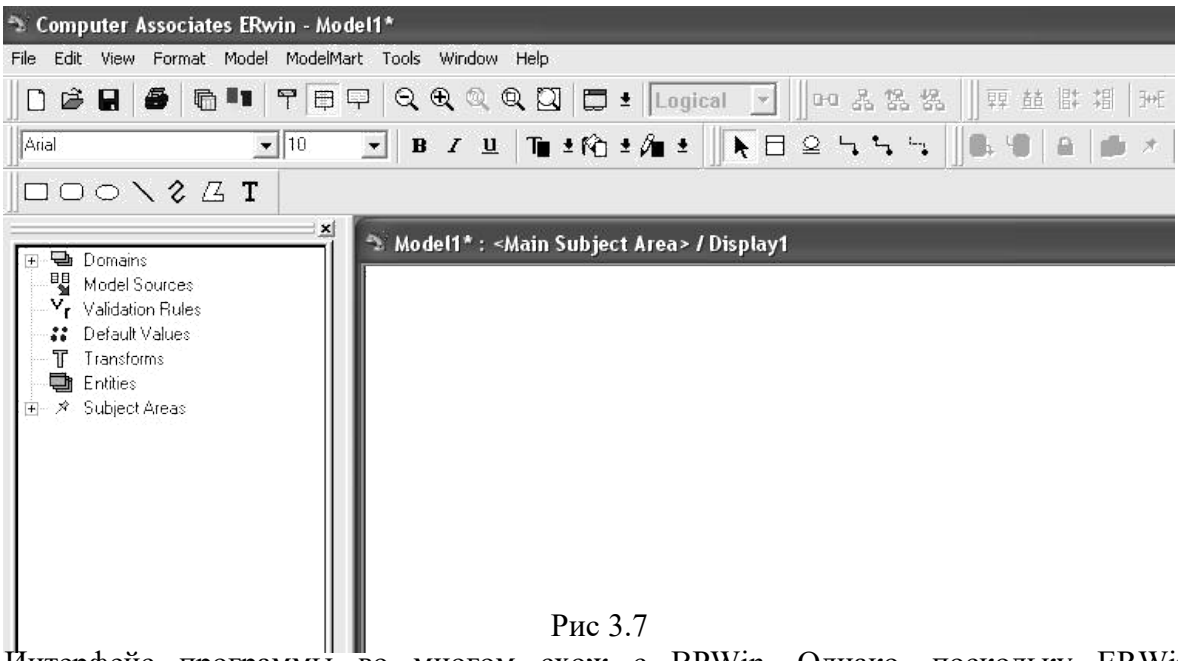


Рис 3.7

Интерфейс программы во многом схож с BPWin. Однако, поскольку ERWin воплощает другую методологию, то, соответственно, его инструменты будут отличаться.

Внимание! Пакет ERWin также использует по умолчанию шрифты без поддержки кириллицы. Для исправления этого выполните следующее:

1. На свободном участке поля модели щёлкните правой кнопкой мыши. Выберите пункт Default Fonts&Colors.
1. В появившемся окне (рис. 3.8) смените все упоминания шрифта Arial на Arial CYR во всех закладках.
2. Нажмите кнопку ОК.

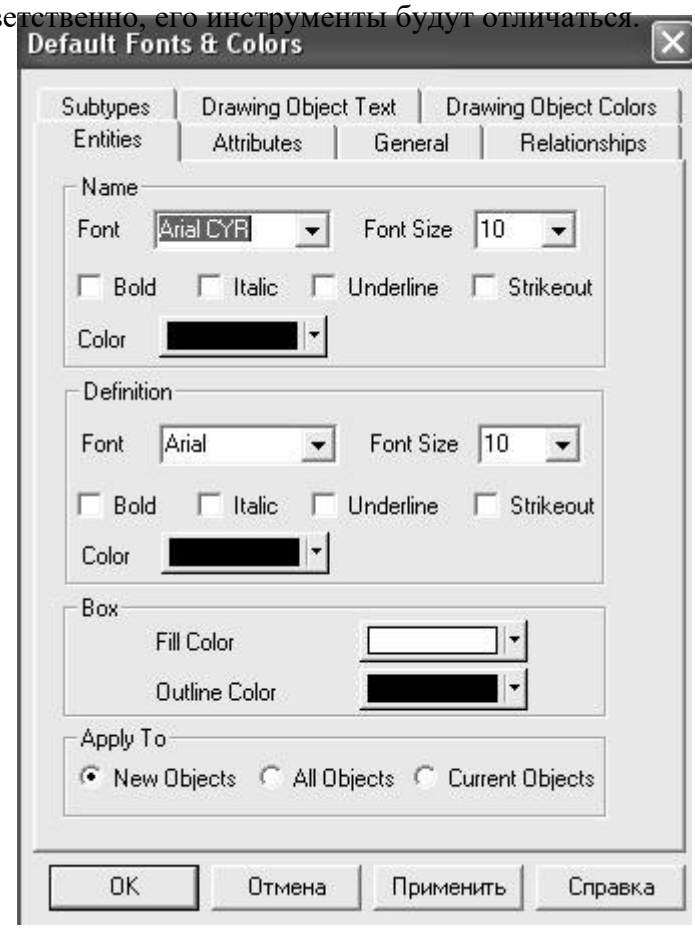


Рис. 3.8

3.1. Панель инструментов

Панель инструментов ERWin приведена на рисунке 3.9:

Рис. 3.9

Здесь:

 - режим выделения. Аналогичен BPWin.

 - создание новой сущности.

 - обозначение полного включения.

 - идентифицирующая связь (один ко многим)

 - Связь «многие ко многим».

 - неидентифицирующая связь.

3.2. Импорт сущностей из BPWin.

Для того чтобы импортировать файл с сущностями, созданный вами в BPWin, создайте новую модель в ERWIN (File->New), а затем дайте команду на импорт файла (File->Import->BPWin…). Появится стандартное окно Windows открытия файла. Найдите свой файл, и нажмите «Открыть». Сущности будут импортированы в новую модель ERWin.

3.3. Создание новой сущности

Для того, чтобы создать новую сущность, войдите в режим создания новой сущности (нажав левую кнопку мыши на

пиктограмме ), и щёлкните на свободном участке рабочей области экрана. На этом месте появится новая сущность.

3.4. Редактирование сущности.

Для того, чтобы отредактировать существующую сущность, выделите её левой кнопкой мыши, затем нажмите

правую кнопку мыши, и в появившемся окне выберите Attributes (Атрибуты). Появится окно атрибутов (рис. 3.10).

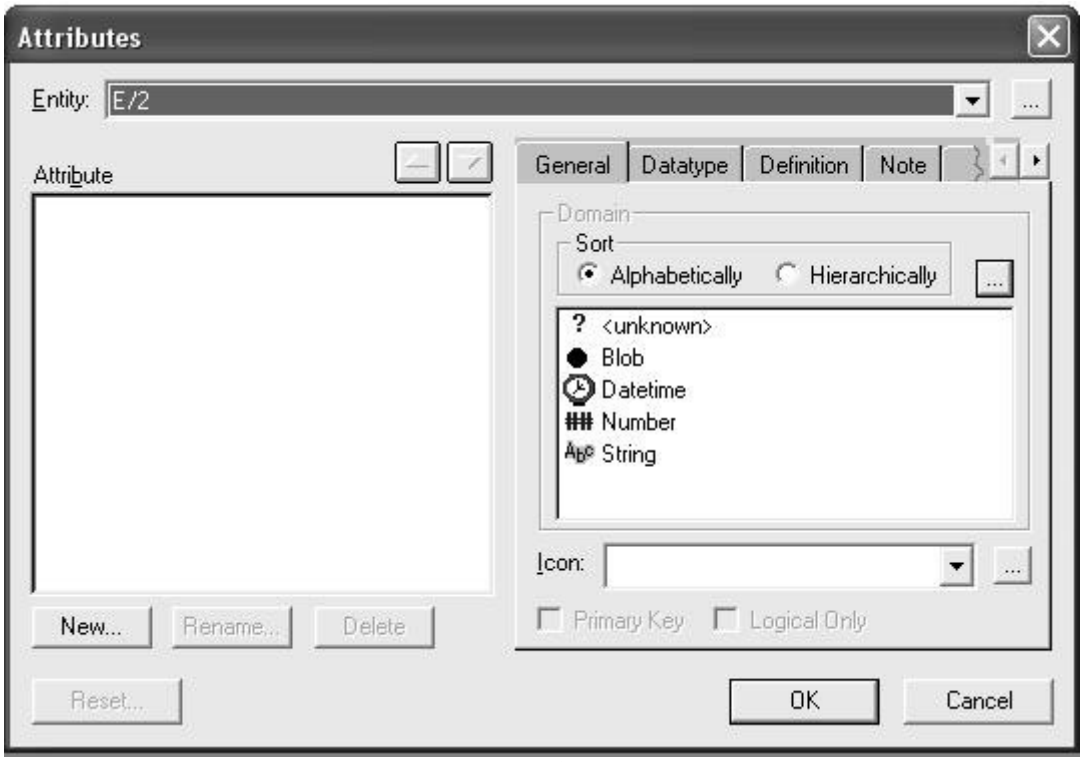


Рис. 3.10

В этом окне вы можете создавать/редактировать/удалять атрибуты для данной сущности.

Для того, чтобы создать новый атрибут, нажмите кнопку New. Появится окно создания атрибута (рис. 3.11).

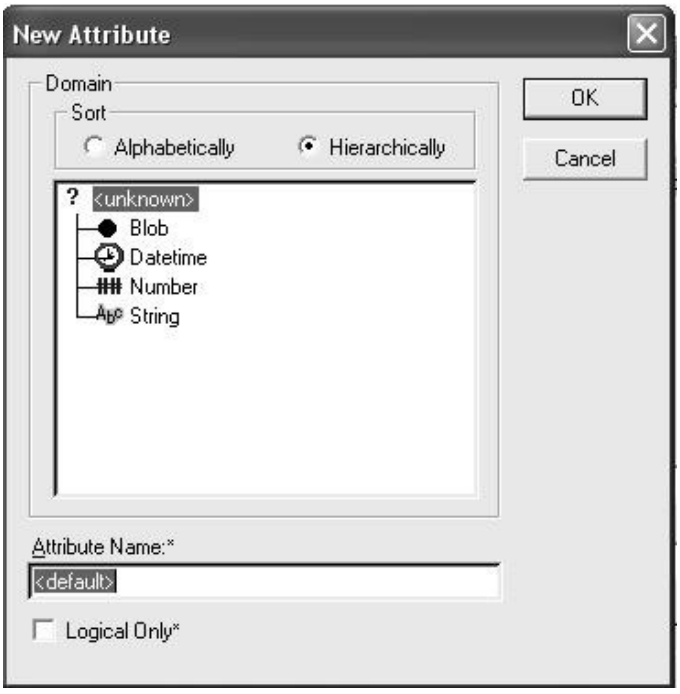


Рис 3.11

В этом окне вы даёте атрибуту имя (поле Attribute Name), и определяете его тип (поле по центру). Определив имя и тип атрибута, нажмите кнопку ОК.

В окне атрибутов появится новый атрибут.

Для того чтобы редактировать атрибут, воспользуйтесь закладками на правой части окна (рис. 3.10) Здесь вы можете определить, является ли атрибут первичным (Primary Key), уточнить тип данных атрибута (вкладка Datatype) и т.д.

3.5. Пример информационной модели

На рис. 3.11 представлена информационная модель физического уровня.

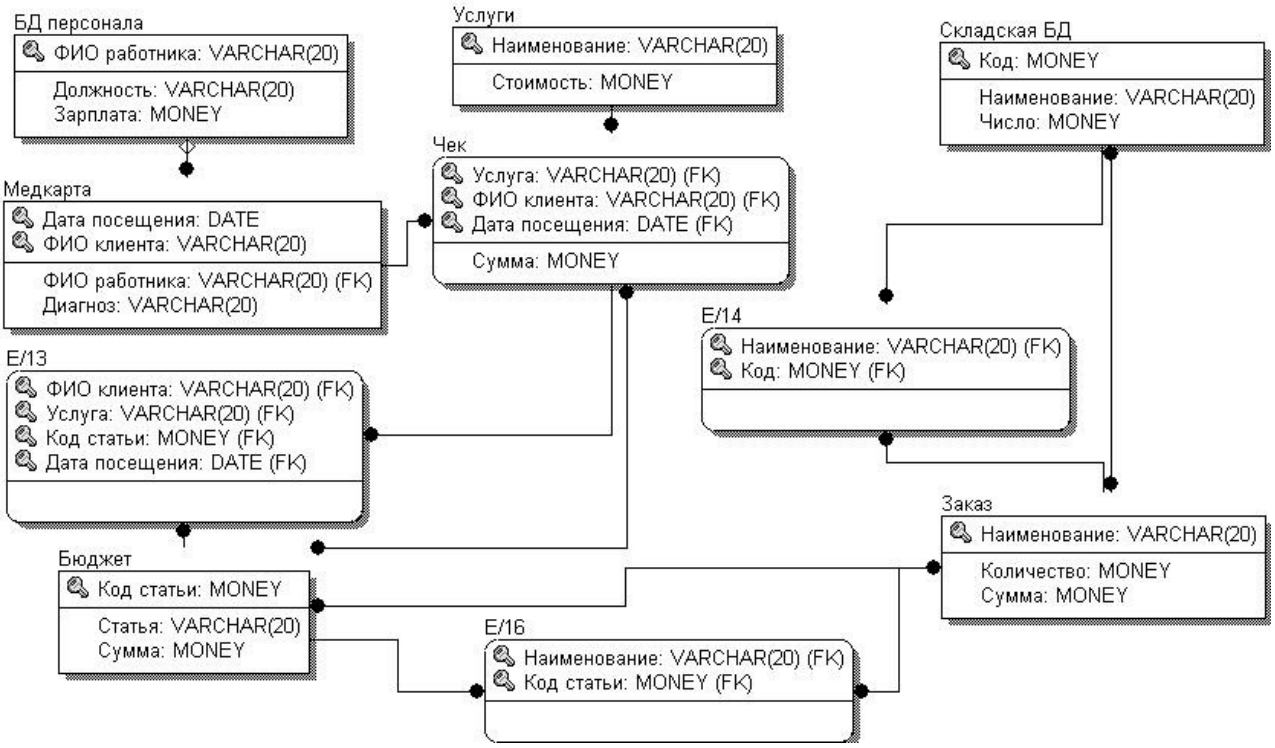


Рис. 3.11

4. Задания на лабораторную работу

3.1. Изучить представление методологии IDEF1X в пакете ERWin.

3.2. Построить информационную модель выбранной предметной области.

ПРАКТИЧЕСКАЯ РАБОТА № 23
ПОСТРОЕНИЕ ДИАГРАММ ПОТОКОВ ДАННЫХ И ГЕНЕРАЦИЯ КОДА

Цель: получение навыков построения диаграмм потоков данных.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Понятие диаграммы потоков данных.

Элементы диаграммы потоков данных.

Хранилищаданных.

Потокиуправления.

Задание № 1. Ознакомиться с методологией построения диаграмм потоков данных.

Диаграммы потоков данных (DataFlowDiagrams – DFD) используются для описания движения документов и обработки информации как дополнение к IDEF0. В отличие от IDEF0, где система рассматривается как взаимосвязанные работы, стрелки в DFD показывают лишь то, как объекты (включая данные) движутся от одной работы к другой. DFD отражает функциональные зависимости значений, вычисляемых в системе, включая входные значения, выходные значения и внутренние хранилища данных. DFD – это граф, на котором показано движение значений данных от их источников через преобразующие их процессы к их потребителям в других объектах.

DFD содержит процессы, которые преобразуют данные, потоки данных, которые переносят данные, активные объекты, которые производят и потребляют данные, и хранилища данных, которые пассивно хранят данные.

Диаграмма потоков данных содержит:

• процессы, которые преобразуютданные;
• потоки данных, переносящиеданные;
• активные объекты, которые производят и потребляютданные;
• хранилища данных, которые пассивно хранятданные.

Процесс DFD преобразует значения данных и изображается в виде эллипса, внутри которого помещается имя процесса (рисунок 11).

Рисунок 11

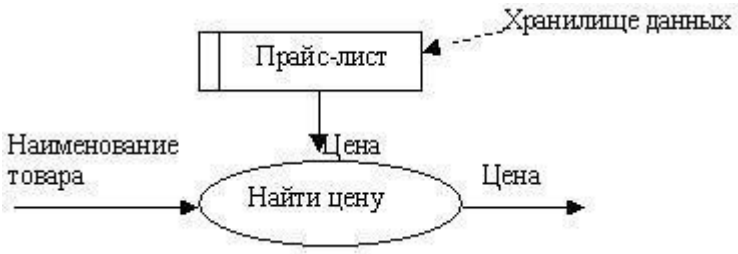
Поток данных соединяет выход объекта (или процесса) с входом другого объекта (или процесса) и представляет собой промежуточные данные вычислений. Поток данных изображается в виде стрелки между производителем и потребителем данных, помеченной именами соответствующих данных. Дуги могут разветвляться или сливаться, что означает соответственно разделение потока данных на части либо слияние объектов.

Активным объектом является объект, который обеспечивает движение данных, поставляя или потребляя их. Хранилище данных – это пассивный объект в составе DFD, в котором данные сохраняются для последующего доступа (рисунок 12).

Рисунок 12

Хранилища данных. Хранилище данных – это пассивный объект в составе DFD, в котором данные сохраняются для последующего доступа. Хранилище данных допускает доступ к хранимым в нем данным в порядке, отличном от того, в котором они были туда помещены. Агрегатные хранилища данных, как, например, списки и таблицы, обеспечивают доступ к данным в порядке их поступления, либо по ключам (рисунок 13).

Потоки управления. DFD показывает все пути вычисления значений, но не показывает, в каком порядке значения вычисляются. Решения о порядке вычислений связаны с управлением программой, которое отражается в динамической модели. Эти решения, вырабатываемые специальными функциями, или предикатами, определяют, будет ли выполнен тот или иной процесс, но при этом не передают процессу никаких данных, так что их включение вфункциональную модель необязательно. Тем не менее, иногда бывает полезно включать указанные предикаты в функциональную модель, чтобы в ней были отражены условия выполнения соответствующего процесса. Функция, принимающая решение о запуске процесса, будучи включенной в DFD, порождает в диаграмме поток управления и изображается пунктирной



стрелкой (рисунок 14).

Рисунок13

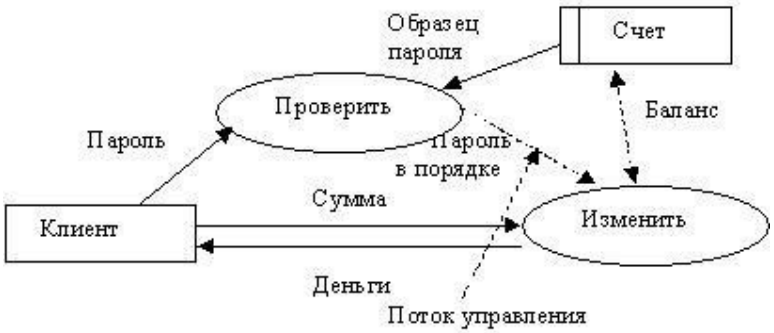


Рисунок14

Первым шагом при построении иерархии DFD является построение контекстных диаграмм. Обычно при проектировании относительно простых информационных систем строится единственная контекстная диаграмма со звездообразной топологией, в центре которой находится так называемый главный процесс, соединенный с приемниками и источниками информации, посредством которых с системой взаимодействуют пользователи и другие внешние системы.

Если же для сложной системы ограничиться единственной контекстной диаграммой, то она будет содержать слишком большое количество источников и приемников информации, которые трудно расположить на листе бумаги нормального формата, и, кроме того, главный единственный процесс не раскрывает структуры распределенной системы.

Для сложных информационных систем строится иерархия контекстных диаграмм. При этом контекстная диаграмма верхнего уровня содержит не главный единственный процесс, а набор подсистем, соединенных потоками данных. Контекстные диаграммы следующего уровня детализируют контекст и структуру подсистем.

При построении иерархии DFD переходить к детализации процессов следует только после определения содержания всех потоков и накопителей данных, которое описывается при помощи структур данных. Структуры данных конструируются из элементов данных и могут содержатьальтернативы, условные вхождения и итерации. Условное вхождение означает, что данный компонент может отсутствовать в структуре. Альтернатива означает, что в структуру может входить один из перечисленных элементов. Итерация означает вхождение любого числа элементов в указанном диапазоне. Для каждого элемента данных может указываться его тип (непрерывные или дискретные данные). Для непрерывных данных может указываться единица измерения (кг, см и т.п.), диапазон значений, точность представления и форма физического кодирования. Для дискретных данных может указываться таблица допустимыхзначений.

Задание № 2. Проанализируйте пример построения диаграммы потоков данных (рисунок 15).

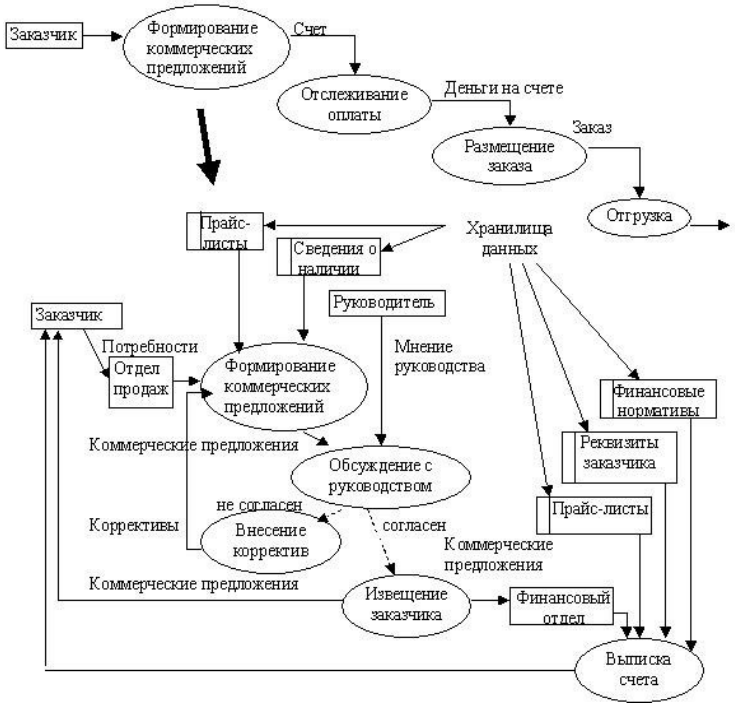


Рисунок 15

Задание № 3. Постройте диаграмму потоков данных для выбранной информационной системы (практическая работа №11).

Задание № 4. Оформите отчет.

ПРАКТИЧЕСКАЯ РАБОТА № 24
УСТАНОВКА И НАСТРОЙКА СИСТЕМЫ КОНТРОЛЯ ВЕРСИЙ С РАЗГРАНИЧЕНИЕМ РОЛЕЙ

Цель: получение навыков построения диаграмм потоков данных.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Понятие системы контроля версий (СКВ), решаемые задачи.

Основные понятия СКВ и их отношения: хранилище, commit, история, рабочая копия.

Отличия централизованных и децентрализованных СКВ. Примеры СКВ каждого вида.

Действия с СКВ при единоличной работе с хранилищем.

Порядок работы с общим хранилищем в централизованной СКВ.

Задание № 1. Изучите систему контроля версий, установленную на компьютере (например, TortoiseSVN). При необходимости установите систему контроля версий TortoiseSVN. Опишите основные возможности системы контроля версий.

Задание № 2. Создайте новый проект. Создайте локальный репозиторий для своего проекта. Удалите созданный проект на своем компьютере и обновите проект из репозитория.

Задание № 3. Внесите изменения в файлах с исходными кодами и сохраните изменения в репозитории. Обновите файлы с исходными кодами из репозитория. Внесите изменения в файлах с исходными кодами таким образом, чтобы у двух участников проекта изменения были в одном и том же файле. Попытайтесь сохранить изменения в репозитории. Устраните обнаруженные конфликты версий. Повторно сохраните изменения в репозитории. Создайте отдельную ветку проекта. Внесите изменения в файлы с исходнымикодами.

Задание № 4. Объедините созданную на предыдущем шаге ветку с основной веткой проекта. Выведите на экран данные изменений файла, в котором было наибольшее количество изменений. Отобразите на экране сравнение фала до и после внесения одного из изменений.

Задание № 5. Создайте репозитарий в сети Интернет. Удалите созданный проект на своем компьютере и обновите проект из репозитория. Внесите изменения в файлах с исходными кодами и сохраните изменения в репозитории. Обновите файлы с исходными кодами из репозитория. Внесите изменения в файлах с исходными кодами таким образом, чтобы у двух участников проектаизменениябыливодномитомжефайле.Попытайтесьсохранитьизмененияврепозитории.
Устраните обнаруженные конфликты версий. Повторно сохраните изменения в репозитории. Создайте отдельную ветку проекта. Внесите изменения в файлы с исходными кодами.

Задание № 6. Оформите отчет.

ПРАКТИЧЕСКАЯ РАБОТА № 25
ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ИНТЕРФЕЙСАПОЛЬЗОВАТЕЛЯ

Цели: получение навыков проектирования и разработки интерфейса пользователя.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Понятие пользовательского интерфейса.

Виды пользовательских интерфейсов.

Основные элементы пользовательского интерфейса.

Требования к разработке пользовательского интерфейса.

Задание № 1. Настроить среду разработки VisualStudio. Создать приложение для Windows, которое имитирует игровой автомат со «счастливыми» числами. Программа должна иметь следующий интерфейс (рисунок 16).

При нажатии на кнопку «Крутить» должны генерироваться три случайных числа от 0 до 9. Если хотя бы одно из них равно семи, на форме должны появляться надпись «Счастливая семерка» и картинка с изображением человека, платящего игроку деньги при выигрыше.Принажатии на кнопке «Выход» программа должна завершать работу. Решение сохранить под именем

«Игра». Создать исполняемый файл приложения.

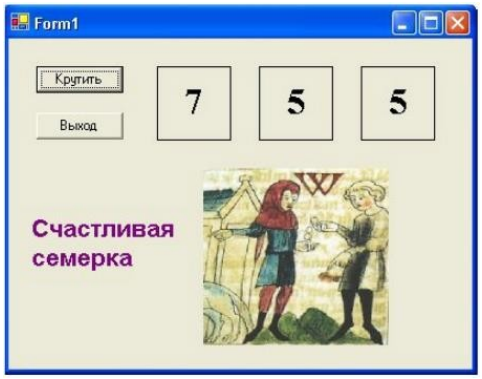


Рисунок 16

Задание № 2. Добавить в созданную форму метку и организовать отображение на ней процента выигрышей по отношению к общему числу нажатий на кнопку «Крутить».

Задание № 3. Добавить в программу оператор Randomize для того, чтобы программа при каждом запуске выдавала новую последовательность случайных чисел.

Задание № 4. Создать приложение для Windows «Продажи он-лайн», которое позволяет выбрать для заказа компьютер, офисную технику и периферийные устройства с выводом в форму изображения выбранного оборудования, указать способ оплаты и желаемую дату поставки. Возможные способы оплаты: рубли, доллары США, английские фунты. При выборе способа оплаты должно появляться его символическое изображение. Пользователь, выбрав товары для заказа, вводит название фирмы. Рекомендуемый интерфейс приложения приведен на рисунке 17.

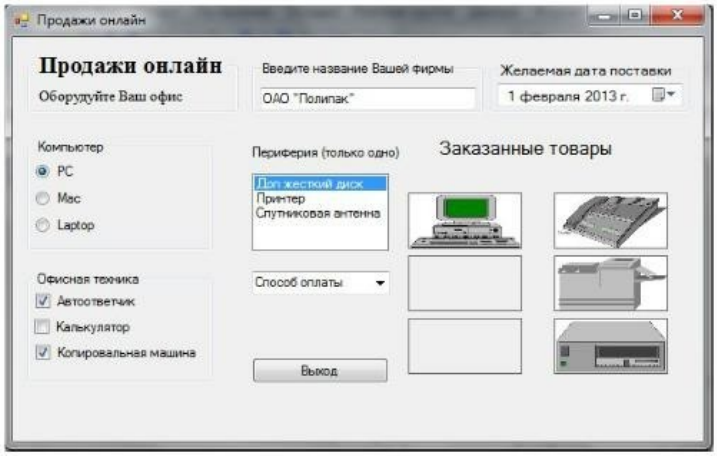


Рисунок 17

Решение сохранить под именем «Продажи». Создать исполняемый файл приложения.

Задание № 5. Добавить в список офисной техники «МФУ» и добавить еще один объект PictureBox для отображения рисунка МФУ. Соответствующим образом изменить программный код.

Задание № 6. Добавить в способы оплаты «Чек».

ПРАКТИЧЕСКАЯ РАБОТА № 26
РЕАЛИЗАЦИЯ АЛГОРИТМОВ ОБРАБОТКИ ЧИСЛОВЫХ ДАННЫХ. ОТЛАДКА ПРИЛОЖЕНИЯ

Цели: получение навыков реализации алгоритмов обработки числовых данных, отладки приложений.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Элементы управления, используемые для обработки числовыхданных.

Задание № 1. Разработать приложение Windows, которое по заданным значениям: цены покупки, суммы первоначального платежа, годовой процентной ставки и срока кредита рассчитывает размер ежемесячных выплат по кредиту, а также строит схему платежей за каждый период (месяц) с разделением на основные платежи и платежи по процентам. Рассчитать также сумму всех основных платежей (для контроля) и сумму платежей по процентам (размер переплаты). Рекомендуемый интерфейс приложения показан на рисунке18.

Решение сохранить под именем «Платежи по кредиту».

Задание № 2. Внесите изменения в программный код так, чтобы в схеме платежей в 4-ом столбце отображалась общая сумма платежа за каждый период.

Задание № 3. Внесите изменения в форму и программный код так, чтобы платежи по кредиту осуществлялись не ежемесячно, а ежеквартально.

Задание № 4. Предусмотрите возможность пересмотра схемы платежей на оставшиеся периоды, если в некоторый период внесен платеж больше требуемой суммы. Рассмотреть такую схему погашения, при которой не уменьшается срок погашения кредита, а уменьшается сумма периодического платежа в последующих периодах.

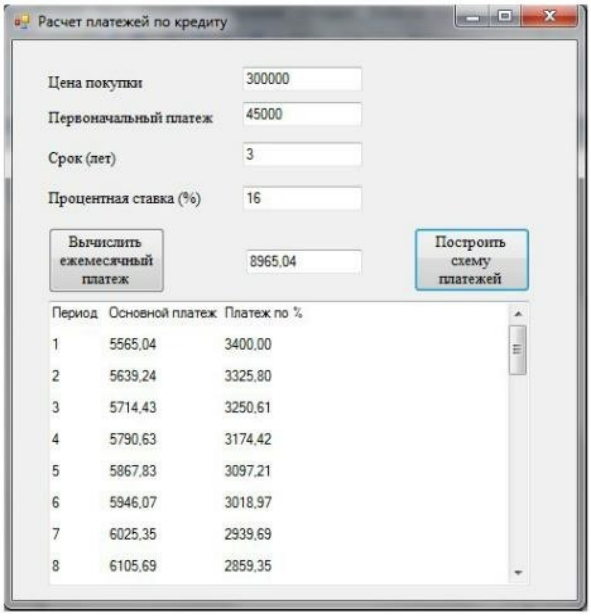


Рисунок 18

ПРАКТИЧЕСКАЯ РАБОТА № 27
РЕАЛИЗАЦИЯ АЛГОРИТМОВ ПОИСКА. ОТЛАДКА ПРИЛОЖЕНИЯ

Цели: получение навыков реализации алгоритмов поиска данных, отладки приложений.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Алгоритмы поиска в тексте.

Алгоритмы поиска в массивах.

Задание № 1. Написать программу «Результаты сессии», которая для выбранной из списка группы запрашивает ввод:− списка группы;− количества и названий предметов, покоторым

данная группа сдавала экзамены в последнюю сессию; − оценок студентов по предметам.
Программа должна также:

− отображать результаты сессии по даннойгруппе;

− вычислять качество знаний (процент студентов, успевающих на «хорошо» и«отлично»);

− вычислять процент успеваемости в группе (процент студентов, сдавшихсессию);

− определять количество студентов, успевающих на«отлично».

Вычисление качества знаний, процента успеваемости и количества отличников оформить в виде соответствующих процедур – функций. По итогам сессии должна быть рассчитана стипендия. Размеры минимальной и повышенной стипендии вводятся с клавиатуры. Минимальную стипендию получают студенты, сдавшие сессию на «хорошо» и«отлично».

В программе должны быть созданы 3 формы: главная форма «Результаты сессии и расчет стипендии», форма для отображения результатов сессии и форма «Размер стипендии» (рисунки 19, 20, 21, 22, 23, 24, 25, 26).

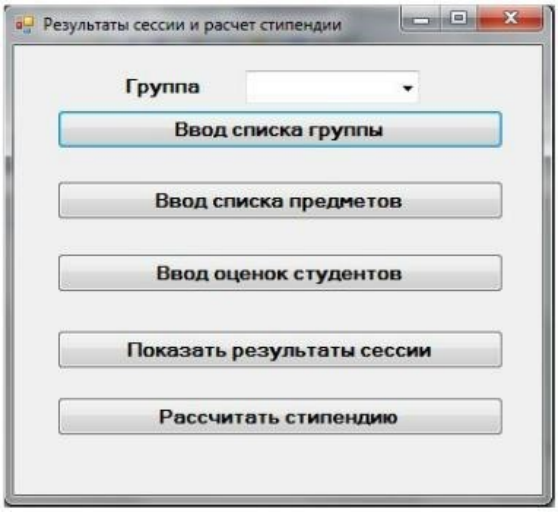
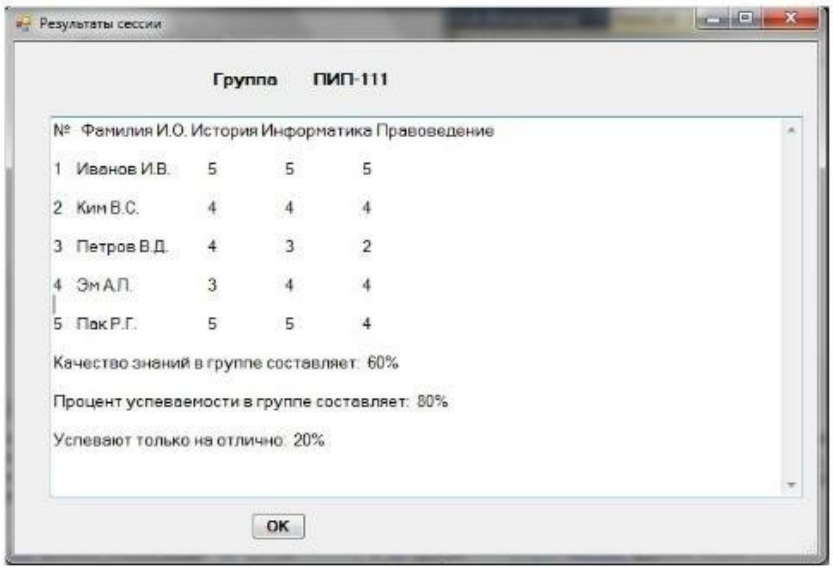


Рисунок 19

Рисунок 20



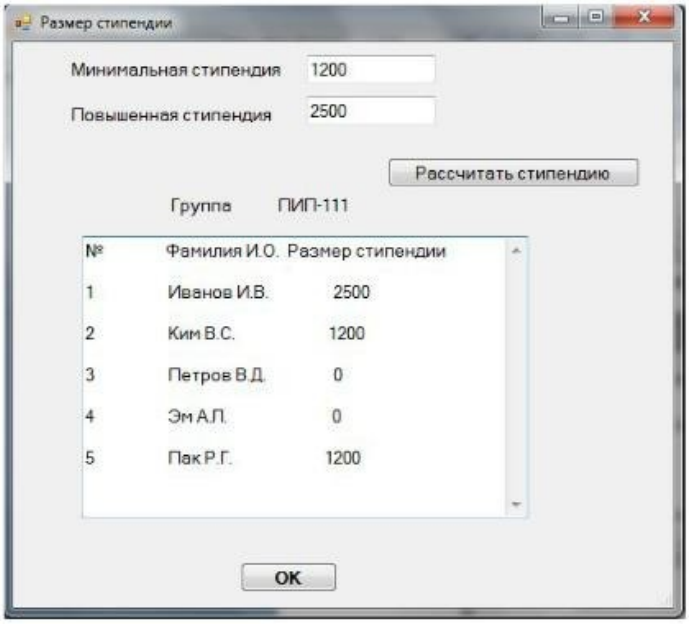


Рисунок21

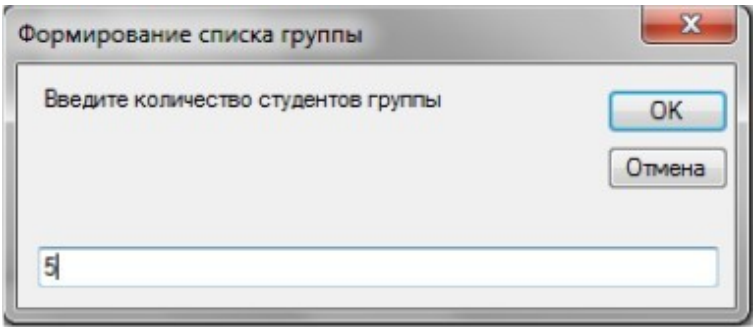


Рисунок22

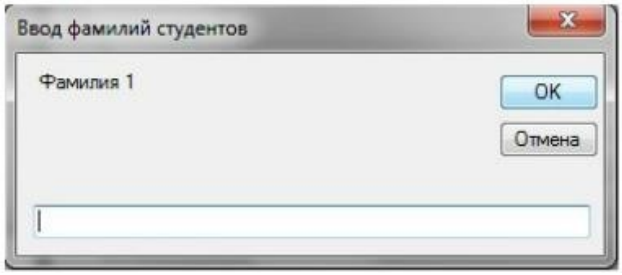


Рисунок2

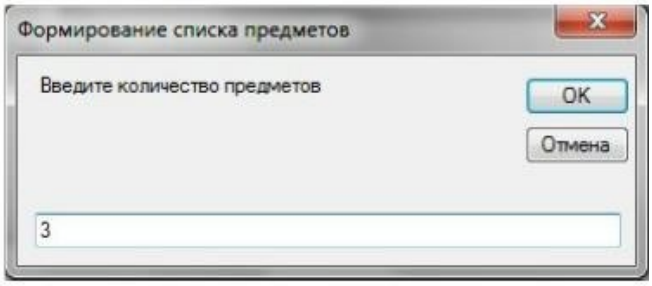


Рисунок 24

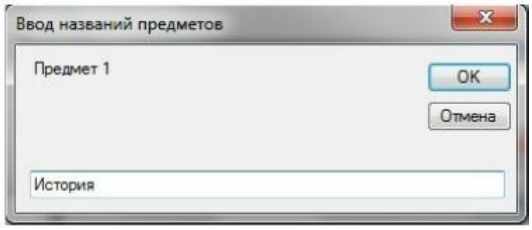


Рисунок25

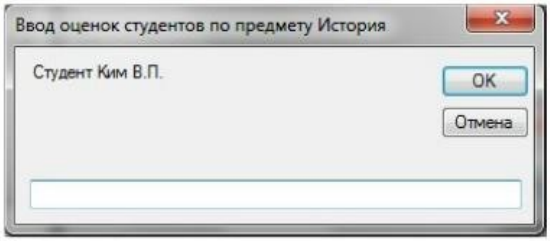


Рисунок26

Задание № 2. Написать программы, иллюстрирующие применение методов линейного поиска, поиска делением пополам, а также различные методы сортировки массивов.

ПРАКТИЧЕСКАЯ РАБОТА № 28
РЕАЛИЗАЦИЯ ОБРАБОТКИ ТАБЛИЧНЫХ ДАННЫХ. ОТЛАДКА ПРИЛОЖЕНИЯ

Цель: получение навыков обработки табличных данных, отладки приложений.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Обработка табличных данных в приложениях.

Задание № 1. Организовать работу с базой данных Студенты, которая храниться в текстовом файле. При выборе в списке ComboBox определенной группы на форме Списки групп отобразит в сетке данных DataGridView только фамилии студентов данной группы. Рекомендуемый интерфейс приложения изображен на рисунке 27.

Задание № 2. Создать запрос, который будет отбирать из базы данных Студенты фамилии студентов заданного курса, записывать их вместе с названием группы во временный файл СтудентыВрем и отображать на форме с помощью элемента DataGridView.

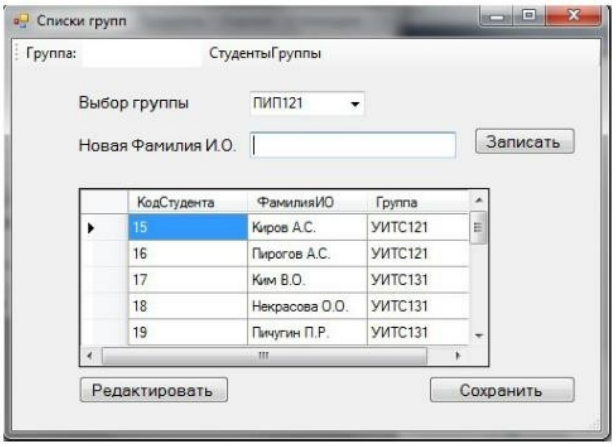


Рисунок 27

ПРАКТИЧЕСКАЯ РАБОТА № 29
РАЗРАБОТКА И ОТЛАДКА ГЕНЕРАТОРА СЛУЧАЙНЫХ СИМВОЛОВ

Цель: получение навыков разработки и отладки генератора случайных символов.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Понятие генератора случайных символов.

Управление генератором случайных символов.

Задание № 1. Разработать генератор случайных чисел.

Случайные числа в языке программирования C++ могут быть сгенерированы функцией rand() из стандартной библиотеки C++. Функцияrand() генерируетчисла в диапазоне от 0 до

RAND_MAX. RAND_MAX – это константа, определённая в библиотеке <cstdlib>. Для MVS RAND_MAX = 32767, но оно может быть и больше, в зависимости от компилятора. Ниже показана простая программка, использующая генератор случайных чиселrand():

```
#include "stdafx.h"
#include<iostream>u
singnamespacestd;

int main(intargc, char* argv[])

{

 cout<< "RAND_MAX = " << RAND_MAX <<endl; // константа, хранящая максимальный предел из интервала случайных чисел

 cout<< "random number = " << rand() <<endl; // запускгенератораслучайныхчисел
 system("pause");

 return 0;

}
```

Максимальное случайное число в примере – это 32767. Зачастую, нам не нужен такой большой диапазон чисел от 0 до RAND_MAX. Например, в игре «Наперстки» необходимо отгадать, под каким из трёх напёрстков спрятан шарик, то есть генерация чисел должна выполняться в пределе от 1 до 3-х. Бросая монету, может возникнуть только два случая, когда монета упадёт «орлом» или «решкой» вверх, нужный интервал – от 1 до 2. Возникает потребность в масштабировании интервала генерации случайных чисел. Для того чтобы масштабировать интервал генерации чисел нужно воспользоваться, операцией нахождения остатка от деления«%»:

```
// пример масштабирования диапазона генерации случайных чисел
rand() % 3 +1 // диапазон равен от 1 до 3 включительно
```

Число 3 является масштабируемым коэффициентом. То есть, какое бы не выдал число генератор случайных чисел rand() запись rand() % 3 в итоге выдаст число из диапазона от 0 до 2. Для того чтобы сместить диапазон, мы прибавляем единицу, тогда диапазон изменится на такой – от 1 до 3 включительно.

Задание № 2. Разработать программу, использующую масштабируемый генератор случайных чисел. Ниже показан код программы, которая несколько раз запускает функцию rand().

```
// rand_ost.cpp: определяет точку входа для консольного приложения.

#include "stdafx.h"
#include <iostream>
using namespace std;

int main(intargc, char* argv[])

{

cout<< "1-random number = " << 1 + rand() % 3 <<endl; // первый запуск генератора случайных чисел

cout<< "2-random number = " << 1 + rand() % 3 <<endl; // второй запуск генератора случайных чисел

cout<< "3-random number = " << 1 + rand() % 3 <<endl; // третий запуск генератора случайных чисел
```

```
cout<< "4-random number = " << 1 + rand() % 3 <<endl; // четвёртый запуск генератора случайных чисел
cout<< "5-random number = " << 1 + rand() % 3 <<endl; // пятый запуск генератора случайных чисел
cout<< "6-random number = " << 1 + rand() % 3 <<endl; // шестой запуск генератора случайных чисел
 cout<< "7-random number = " << 1 + rand() % 3 <<endl; // седьмой запуск генератора случайных чисел
 cout<< "8-random number = " << 1 + rand() % 3 <<endl; // восьмой запуск генератора случайных чисел
 system("pause");
 return 0;
}
```

При повторном запуске программы, печатаются те же самые числа. Суть в том, что функция rand() один раз генерирует случайные числа, а при последующих запусках программы всего лишь отображает сгенерированные первый раз числа. Такая особенность функции rand() нужна для того, чтобы можно было правильно отладить разрабатываемую программу. При отладке программы, внеся какие-то изменения, необходимо удостовериться, что программа срабатывает правильно, а это возможно, если входные данные остались те же, то есть сгенерированные числа. Когда программа успешно отлажена, нужно, чтобы при каждом выполнении программы генерировались случайные числа. Для этого нужно воспользоваться функцией srand() из стандартной библиотеки C++. Функция srand() получив целый положительный аргумент типа unsigned или unsignedint (без знаковое целое) выполняет рандомизацию, таким образом, чтобы при каждом запуске программы функция srand() генерировала случайные числа. Программа, использующая функцию srand() для рандомизации генератора случайных чисел rand():

```
// srand.cpp: определяет точку входа для консольного приложения.
#include "stdafx.h"
#include <iostream>
using namespace std;
int main(intargc, char* argv[])
{
 unsigned rand_value = 11;
 srand(rand_value); //
 рандомизациягенератораслучайныхчиселcout<< "rand_value = "
 <<rand_value<<endl;
 cout<< "1-random number = " << 1 + rand() % 10 <<endl; // первый запускгенератора
случайных чисел
cout<< "2-random number = " << 1 + rand() % 10 <<endl; // второй запуск генератора случайных
чисел
 system("pause");
 return 0;
}
```

Задание № 3. Разработать обобщённый пример использования автоматического генератора случайных чисел с масштабированием. Пример работы программы:

```
// srand_time.cpp: определяет точку входа для консольного приложения.

#include "stdafx.h"
#include <iostream>
#include <ctime>
using namespace std;
int main(intargc, char* argv[])
{
 srand( time( 0 ) ); //
 автоматическаярандомизациясout<< "rand_value
 = " << 1 + rand() % 10 <<endl; system("pause");
 return 0;
}
```

Теперь при каждом срабатывании программы будут генерироваться совершенно случайные числа в интервале от 1 до 10, включительно.

Задание № 4. Разработать генератор случайных символов. Сформировать случайную символьную последовательность.

ПРАКТИЧЕСКАЯ РАБОТА № 30
ИНТЕГРАЦИЯ МОДУЛЯ В ИНФОРМАЦИОННУЮ СИСТЕМУ

Цель: получение навыков интеграции модулей в информационную систему.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретическиевопросы

Понятие модуля.

Управление модулями.

Задание № 1. Создать файл, содержащий сведения о сдаче студентами сессии. Структура записи: индекс группы, фамилия студента с его инициалами, оценки по четырем экзаменам и пяти зачетам («з» означает зачет, «н» – незачет). Экзамены и зачеты нумеровать цифрами. Количество записей в файле не менее двадцати.

При разработке приложения использовать стандартные модули.

Задание № 2. Разработать программу, интегрирующую модули из приложения, разработанного в рамках задания №1, выводящую следующуюинформацию:

– фамилии неуспевающих студентов с указанием индексов групп и видазадолженности;
– фамилии студентов, сдавших все зачеты и получившие на экзаменах четверки ипятерки;

– средний бал, полученный каждымстудентом.

ПРАКТИЧЕСКАЯ РАБОТА №31

РАЗРАБОТКА ПРИЛОЖЕНИЙ ДЛЯ МОДЕЛИРОВАНИЯ ПРОЦЕССОВ И ЯВЛЕНИЙ. ОТЛАДКА ПРИЛОЖЕНИЯ

Цель: получение навыков разработки и отладки приложений для моделирования процессов и явлений.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретическиевопросы

Понятие модели.

Моделирование процессов и явлений.

Технологии моделирования процессов и явлений в приложениях.

Задание № 1. Разработать физико-математическую модель системы при сободном падении физического тела, брошенного с высоты h и падающего свободно в течение t времени. При построении модели принять следующие гипотезы:

1) падение происходит в вакууме (то есть коэффициент сопротивления воздуха равеннулю);

2) ветранет;

3) масса теланеизменна;

4) тело движется с одинаковым постоянным ускорением g в любойточке.

Слово "модель" (лат. modelium) означает "мера", "способ", "сходство с какой-то вещью".

Проблема моделирования состоит из трех взаимосвязанных задач: построение новой (адаптация известной) модели; исследование модели (разработка метода исследования или адаптация, применение известного); использование (на практике или теоретически) модели.

Схема построения модели M системы S с входными сигналами X и выходными сигналами Y изображена на рисунке 28.

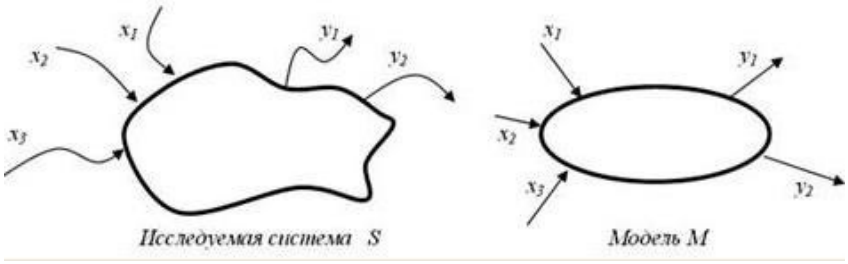


Рисунок 28

Если на вход M поступают сигналы из X и на входе появляются сигналы из Y, то задан закон, правило f функционирования модели, системы.

Классификацию моделей проводят по различным критериям.

Модель – статическая, если среди параметров описания модели нет (явно) временного параметра.

Модель – динамическая, если среди параметров модели явно выделен временной параметр.

Модель – дискретная, если описывает поведение оригинала лишь дискретно, например, в дискретные моменты времени (для динамической модели).

Модель – непрерывная, если описывает поведение оригинала на всем промежутке времени.

Модель – детерминированная, если для каждой допустимой совокупности входных параметров она позволяет определять однозначно набор выходных параметров; в противном случае – модель недетерминированная, стохастическая (вероятностная).

Модель – функциональная, если представима системой функциональных соотношений (например, уравнений).

Модель – теоретико-множественная, если представима некоторыми множествами и отношениями их и ихэлементов.

Модель – логическая, если представима предикатами, логическими функциями и отношениями.

Модель – информационно-логическая, если она представима информацией о составных элементах, подмоделях, а также логическими отношениями между ними.

Модель – игровая, если она описывает, реализует некоторую игровую ситуацию между элементами (объектами и субъектами игры).

Модель – алгоритмическая, если она описана некоторым алгоритмом или комплексом алгоритмов, определяющим ее функционирование, развитие. Введение такого, на первый взгляд, непривычного типа моделей (действительно, кажется, что любая модель может быть представлена алгоритмом ее исследования), на наш взгляд, вполне обосновано, так как не все модели могут быть исследованы или реализованыалгоритмически.

Модель – графовая, если она представима графом (отношениями вершин и соединяющих их ребер) или графами и отношениями между ними.

Модель – иерархическая (древовидная), если она представима иерахической структурой (деревом).

Модель – языковая, лингвистическая, если она представлена некоторым лингвистическим объектом, формализованной языковой системой или структурой. Иногда такие модели называют вербальными, синтаксическими и т.п.

Модель – визуальная, если она позволяет визуализировать отношения и связи моделируемой системы, особенно в динамике.

Модель – натурная, если она есть материальная копия оригинала.

Модель – геометрическая, если она представима геометрическими образами и отношениями между ними.

Модель – имитационная, если она построена для испытания или изучения, проигрывания возможных путей развития и поведения объекта путем варьирования некоторых или всех параметров модели.

Задание № 2. Разработать статическая модель движения тела по наклонной плоскости F = am. Динамическая модель типа закона Ньютона: F(t) = a(t)m(t) или, еще более точно и лучше, F(t)=s"(t)m(t). Если рассматривать только t = 0.1, 0.2, …, 1 (с), то модель St = gt2/2 или числовая последовательность S0 = 0, S1 = 0.01g/2, S2 = 0.04g, …, S10 = g/2 может служить дискретной моделью движения свободно падающего тела. Модель S = gt2/2, 0 < t < 10 непрерывна на *промежутке времени (0;10).*

Задание № 3. Разработать модель популяции рыб, из которой в текущий момент времени изымается некоторое количество особей (идет лов рыбы). Динамика такой системы определяется моделью вида: xi + 1 = xi + axi – kxi, x0 = c , где k – коэффициент вылова (скорость изъятия особей). Стоимость одной пойманной

рыбы равна b руб. Цель моделирования – прогноз прибыли при заданной квоте вылова. Для этой модели можно проводить имитационные вычислительные эксперименты и далее модифицировать модель, например следующим образом.

Эксперимент 1. Для заданных параметров a, c изменяя параметр k, определить его наибольшее значение, при котором популяция не вымирает.

Эксперимент 2. Для заданных параметров c, k изменяя параметр a, определить его наибольшее значение, при котором популяция вымирает.

Модификация 1. Учитываем естественную гибель популяции (за счет нехватки пищи, например) с коэффициентом смертности, равным, b: xi + 1 = xi + axi – (k + b)xi, x0 = c .

Модификация 2. Учитываем зависимость коэффициента k o от x (например, k = dx): $x_{i+1} = x_i + ax_i - dx_i^2, x_0 = c$.

ПРАКТИЧЕСКАЯ РАБОТА № 32
ПРОГРАММИРОВАНИЕ ОБМЕНА СООБЩЕНИЯМИ МЕЖДУ МОДУЛЯМИ

Цели: получение навыков программирования обмена сообщениями между модулями.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретическиевопросы

Понятие и структура сообщения. Обмен
сообщениями между модулями.

Задание № 1. Составить программу, помогающую сотрудникам Государственной инспекции безопасности дорожного движения (ГИБДД) обработать следующие данные: регистрационный номер автомобиля, марка автомобиля, цвет автомобиля, год выпуска, адрес владельца. Программа должна по требованию пользователя выдавать следующие сведения:

• адреса владельцев автомобилей заданной марки, определенногоцвета;
• все данные об автомобиле с заданным регистрационнымномером;
• *все данные об автомобилях с известной цифровой частью регистрационногономера.*

Задание № 2. Программу, разработанную в задании №1, разбить на модули. Например, создать такие модули, как главный (содержащий функцию main()), чтения из файла в массив структур, вывод на экран содержимого массива структур, сортировка данных (при необходимости), меню, формирование документов ит.д.

Задание № 3. Разработать схему межмодульных вызовов.

Задание № 4. Проанализировать способы передачи аргументов между функциями и целесообразность использования глобальных данных

ПРАКТИЧЕСКАЯ РАБОТА № 33
ОРГАНИЗАЦИЯ ФАЙЛОВОГО ВВОДА-ВЫВОДА ДАННЫХ

Цели: получить практические навыки программирования задач ввода-вывода с использованием файлов

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические сведения

Организация ввода и вывода. Файловая система

Операции ввода/вывода в языке С осуществляются через потоки. *Поток* - это логическое уст выдающее и принимающее информацию.

С потоком связано понятие внутреннего указателя, который определяет позицию, с которой на следующая операция чтения или записи. При каждой операции чтения или записи происходит автома перемещение указателя.

В языке С (С++) формат стандартных файлов ввода/вывода описан в заголовочном файле stdio.h стандартных файлов ввода/вывода для языка С (С++) представлены в табл. 7.1. В момент начала вып программы на языке С (С++) автоматически открываются три потока: stdin, stdout, stderr.

Таблица 7.1

Потоки, определяемые в языке С и С++

Имя стандартного файла	Описание
stdaux	Последовательный ввод/вывод
stderr	Выходной поток ошибок
stdin	Стандартный ввод
stdout	Стандартный вывод
stdprn	Вывод на принтер

С++ поддерживает всю систему ввода/вывода С и добавляет к ней дополнительные возможности, свя основном с вводом/выводом объектов. Описание средств для создания потоков в С++ представлено в загол файле iostream.h. Когда начинает работать программа на С++, открываются потоки, приведенные в табл. 7.2

Таб

Потоки, определяемые в языке С++

Имя стандартного файла	Описание

cin	Стандартный ввод - клавиатура
cout	Стандартный вывод - экран
cerr	Стандартная ошибка - экран
clog	Буферизованная версия cerr - экран

Файловая система языков C и C++ состоит как бы из двух уровней: логических файлов и физических фай которыми логические файлы всегда связаны.

Логический файл описывается как указатель на открываемый поток FILE * и служит средством взаимо с физическим файлом. Имя физического файла появляется в программе всего один раз, в тот момен происходит открытие файла, осуществляемое функцией fopen() и одновременно его связывание с лог файлом.

Основными действиями, производимыми над файлами, являются их открытие, обработка и з Обработка файлов может заключаться в считывании блока данных из потока в оперативную память, запи данных из оперативной памяти в поток, считывание определенной записи данных из потока, з определенной записи данных в поток. При этом необходимо помнить, что понятие файла в памяти определено, и приобретает смысл только после его связи с внешним физическим файлом.

Текстовые файлы

Тип FILE определяется в заголовочном файле stdio.h и обычно представляет собой структуру, соде параметры реализации потока, такие как адреса буферов, указатели позиций потока, маркеры ошибок поток

При работе с дисковыми файлами в момент их открытия следует задать режим доступа, чтобы опред какому файлу осуществляется доступ: к текстовому или двоичному, а также способ доступа: чтение или за это выполняется функцией fopen(), имеющей синтаксис:

fopen("**имя_файла**", "**режим_доступа**")

Режимы доступа к файлам для функции fopen() приведены в табл. 7.3.

Таблица 7.3

Режимы доступа к файлам

Режим	Описание
	Открыть файл только для чтения, модификации файла запрещены.
	Создать новый файл только для записи. При попытке открыть таким способом существующий файл происходит перезапись файла. Чтение данных из файла запрещено.
	Открыть файл для дозаписи. Если файла с указанным именем не существует, он будет создан.
	Открыть существующий файл для чтения и записи.
	Создать новый файл для чтения и записи.
	Открыть существующий файл для дозаписи и чтения.

Таким образом, чтобы открыть текстовый файл, например, для чтения, нужно произвести следующие действия:

```
FILE *ft;                    // объявили указатель на файловый поток
ft = fopen("inp_f.txt","r");        // открыли файл inp_f.txt
```

При попытке открыть существующий файл можно допустить ошибку в его имени или в указании нужному файлу. Это вызывает ошибку исполнения программы. Следует предвидеть подобные ситу проводить проверку возможности открытия файла. Такую проверку осуществить довольно легко, так как fopen() возвращает значение указателя в случае успешного его открытия и значение NULL, если доступ невозможен. Поэтому достаточно написать:

```
if (ft = fopen("inp_f.txt","r") != NULL)
{  // обработка файла
   }
```

Текстовой файл состоит из последовательности символов, разбитой на строки путем исполь управляющего символа \n. На диске текстовые файлы хранятся в виде сплошной последовательности сим их деление на строки становится заметным лишь в момент вывода на экран или печать, так как именно при управляющие символы начинают выполнять свои функции. Текстовые файлы легко переносятся с одно компьютера на другой лишь в случаях, когда они содержат только символы, принадлежащие стандартному символов.

При работе с текстовыми файлами возможна их посимвольная или построчная обработка.

Основные методы обработки текстовых файлов

Файловые функции ввода/вывода fprintf() и fscanf() работают аналогично функциям printf() и scanf() дополнительный аргумент, являющийся указателем на файловый поток.

Пример 7.1.Чтение одного элемента из файла, обработка и запись результата в текстовый файл.

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
void main()
{  FILE *f;
int dig;
   if (f = (fopen("inp_f","r")) == NULL)      // открытьфайлдлячтения
{ printf("Невозможно открыть файл!\n");
exit(0);
        }
fscanf(f, "%d", &dig);                    // считать значение dig из файла
```

```
fclose(f);                    // закрыть файл
f = fopen("out_f","w");                    // открыть файл для записи
fprintf(f, "Мы прочитали число %d", dig);
fclose(f);                    // закрыть файл
}
```

Мы использовали один и тот же указатель на файловый поток дважды, так как прежде, чем обращать вторично, закрыли связанный с ним файл и освободили, таким образом, указатель.

В приведенном примере имя файла было записано непосредственно в операторе открытия файла. сообщить имя открываемого файла, введя его с клавиатуры или пользуясь аргументами командной строки.

Пример 7.2.Написать программу, которая сжимает содержимое файла, записывая в выходной ф каждый третий символ из входного файла.

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>
void main(intargc, char *argv[])
{ FILE *f_in, *f_out;
intch;
  char *name;                 // имявходногофайла
int count = 0;                // счетчикэлементов
  if (argc<2)                 // в командной строке нет имени
{  printf("Введите имя входного файла");
gets(name);
        }
elsename = argv[1];           // взять имя из командной строки
if ((f_in = fopen(name, "r")) != NULL)
{ strcat(name, ".out");                  // добавляет расширение .out
                    // к имени файла
f_out = fopen(name, "w");     // открывает файл для записи
while((ch = fgetc(f_in)) != EOF)
if (count++ % 3 == 0)
fputc(ch, f_out);         // выводит каждый третий символ
```

```
fclose(f_in);

fclose(f_out);

}

elseprintf("Невозможно открыть файл\n ");

}
```

При работе с текстовыми файлами возможна не только поэлементная обработка файлов, но и построчная.

Пример 7.3.Построчное чтение информации из входного файла и вывод ее на экран как на с устройство вывода.

```
#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

#include <string.h>

void main(intargc, char *argv[])

{ FILE *f_in;

charbuffer[256];        // максимальная длина строки - 255 символов

char *name;    // имя входного файла

if (argc<2)                // в командной строке нет имени

{  printf("Введите имя входного файла");

gets(name);

        }

elsename = argv[1];              // взять имя из командной строки

if ((f_in = fopen(name, "r")) != NULL)

{ while (fgets(buffer,255,f_in)) != NULL)

{ fputs(buffer, stdout);

fputc('\n', stdout);

                }

fclose(f_in);

         }

     else printf("Невозможнооткрытьфайл\n ");

}
```

В цикле while присутствуют две файловые функции работы со строками: fgets() для чтения строки с буфер и fputs() - для записи содержимого буфера в файл.

Закрыть файл не менее важно, чем открыть его, так как в этот момент происходит заполнение ячейки размещения файлов значением, которое является признаком завершения файла и установка атрибутов файла

Кроме того, вывод текстового файла буферизован. Это значит, что в тот момент, когда работает операт в файл, фактическая запись может и не происходить, поскольку реально сначала происходит заполнение текстового буфера, а потом его содержимое записывается на диск. Запись буфера происходит как только он полностью заполненным или при выполнении специальных команд принудительной записи на диск. Проце недозаполненного буфера на диск называется *флэшированием* и обычно выполняется с помощью fflush(f_out). При необходимости завершить работу сразу со всеми открытыми файлами пользуются flushall().

Закрытие файла посредством функции fclose(f_out) также включает процесс флэширования, то ест информации из буфера на диск.

Доступ к элементам текстовых файлов возможен только в последовательном режиме как при записи фа при его чтении.

Задание

Напишите программу согласно Вашему варианту задания.

Варианты заданий

	Задание
	Случайным образом создать таблицу пар целочисленных значений и записать её в текстовый файл в виде: XY 5 1 2 8 12 3 - - - - Считать из файла пары значений и в тех из них, где X>Y , поменять значения местами. Результат записать в другой текстовый файл такого же формата.
	Ввести с клавиатуры попарно значения вещественного типа и записать их в текстовый файл в виде таблицы следующего формата: X :Y 2.1 :3.7 6.2 : 5.4 --- - --- Считать из файла полученные пары значений и создать из них другой файл вида:

	sin(x) :cos(y) значение sin(2.1) : значение cos(3.7) ---------- - ----------
	Задание
	Случайным образом создать таблицу пар значений и записать её в текстовый файл в виде: n * c 5 * m 7 * a 3 * q ----------
	Считать из файла пары значений и создать из них другой текстовый файл вида mmmmm aaaaaaa qqq
	Случайным образом создать таблицу пар целочисленных значений и записать её в текстовый файл в виде: XY 51 2 8 12 3 - - - -
	Считать из файла пары значений и в тех из них, где X кратен Y , пометить строку таблицы: XY 51 *** 2 8 12 3 *** - - в том же файле.
	Случайным образом создать таблицу пар значений и записать её в текстовый файл в виде: abc

	5.2 4.6 2.5 можно 1.2 8.9 2.3 ---------- Считать из файла записанные данные и определить, можно ли построить треугольник с такими сторонами. Пометить соответствующие строки таблицы (в том же файле).
	Задание
	Создать текстовый файл, содержащий целочисленные значения, следующего формата 5 21 4 37 52 9 . . .Определить, являются ли значения, находящиеся в файле, упорядоченными по возрастанию.
	Создать текстовый файл, содержащий вещественные значения, следующего формата 5.3 21.4 37.4 52.6 9.2 . . . Считать из файла записанные данные и определить максимальное значение. Если оно находится в первой половине файла, заменить его суммой последующих элементов, если во второй – суммой предыдущих элементов.
	Случайным образом создать таблицу пар целочисленных значений и записать её в текстовый файл в виде: XY 5 25 1 3 49 7 - - Считать из файла пары значений и в тех из них, где X является точным квадратом Y или наоборот, найти сумму значений X иY. Результат записать в другой текстовый файл в виде X Y sum 5 25 30 1 3 49 7 56
	Случайным образом создать таблицу пар целочисленных значений и записать её в текстовый файл в виде: XY 5 1 2 8 12 3 - -

	- - Считать из файла пары значений и в тех из них, где Y кратен X, а X – четное, пометить строку таблицы: XY . 510 2 8 *** 12 3 - - в том же файле.
	Случайным образом создать таблицу пар значений и записать её в текстовый файл в виде: abc 5.2 4.6 2.5 можно 1.2 8.9 2.3 --- Считать из файла записанные данные и определить, можно ли построить треугольник с такими сторонами. В соответствующих строках (где можно), указать площадь полученного треугольника (в другом файле).
	Создать текстовый файл, содержащий целые значения, следующего формата 5 3 21 4 37 52 9 2 . . . Считать из файла записанные данные и определить минимальное значение. Если оно кратно трем, заменить каждое третье значение файла нулем, если кратно пяти – заменить его суммой первого и последнего элементов.
	Случайным образом создать таблицу пар значений и записать её в текстовый файл в виде: n * c 5 * m 7 * a 3 * q --- Преобразовать эту таблицу по следующему образцу (преобразования производить в исходном файле) n * c # 5 * m mmmmm 7 * a aaaaaaa 3 * qqqq ---

	Если первое значение не является числом, то в третьем столбце стоит один символ #
	Задание
	Ввести с клавиатуры значения вещественного типа и записать их в текстовый файл в виде таблицы следующего формата: X :Y : Z 2.1 : 3.7 : 0.9 6.2 : 5.4 : 4.2 --- - --- : --- Считать из файла полученные значения и создать из них другой файл вида: sin(max{X,Y,Z}) : cos(min{X,Y,Z}) значение sin(3.7) : значение cos(0.9) ---------- - ----------

ПРАКТИЧЕСКАЯ РАБОТА №34
РАЗРАБОТКА МОДУЛЕЙ ЭКСПЕРТНОЙ СИСТЕМЫ

Цели: освоение технологии и методики построения экспертных систем на примере разработанной учебной экспертной системы Форма отчета:

−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.

Время выполнения: 2 ч

Экспертная оболочка EsWin

ОБЩИЕ ПОЛОЖЕНИЯ

ESWinv. 1.0 - программная оболочка для работы с продукционно-фреймовыми экспертными системами с возможностью использования лингвистических переменных. Описываемая программная оболочка предназначена для решения задач методом обратного логического вывода на основе интерпретации правил-продукций с использованием фреймов как структур данных, включающих в себя в частности лингвистические переменные.

БАЗА ЗНАНИЙ

База знаний состоит из набора фреймов и правил-продукций. Формат внешнего представления базы знаний (в текстовом файле) выглядит следующим образом:

TITLE = <название экспертной системы>
COMPANY = <название предприятия>
FRAME // фрейм
<описание фрейма>
ENDF
.
.
.
FRAME // фрейм
<описание фрейма>
ENDF
RULE // правило-продукция
<описание условий правила>
DO

<описание заключений правила>
ENDR
.
.
.
RULE // правило-продукция
<описание условий правила>
DO
<описание заключений правила>
ENDR

База знаний состоит из двух частей: постоянной и переменной. Переменная часть базы знаний называется базой данных и состоит из фактов, полученных в результате логического вывода. Факты в базе данных не являются постоянными. Их количество и значение зависит от процесса и результатов логического вывода.

До начала работы с экспертной оболочкой база знаний находится в текстовом файле. В файле с расширением ***.klb** (**KnowLedgeBase**) хранятся фреймы и правила-продукции (база знаний). При начале работы с программной оболочкой наличие данного файла обязательно. Этот файл создается пользователем с помощью специального редактора или вручную. В файле с расширением ***.dtb** (**DaTaBase**) хранятся факты, полученные в процессе логического вывода (база данных). При начале работы с программной оболочкой наличие данного файла необязательно. Файл с базой данных создается программной оболочкой в процессе логического вывода. Первые части имен этих двух файлов совпадают.

При работе с программной оболочкой (после загрузки в оперативную память баз) фреймы и правила-продукции, находившиеся в файле с расширением ***.klb**, остаются неизменными. Факты, находившиеся в файле с расширением ***.dtb**, могут изменяться в процессе логического вывода (появляться, удаляться или менять свое значение в результате срабатывания правил-продукций или диалога с пользователем).

Пример базы знаний:

```
TITLE = для выбора метода представления знаний
FRAME = Цель
  Метод представления знаний: ()
ENDF
FRAME = Тип
  Решаемые задачи: (диагностика; проектирование)
ENDF
FRAME = Область
  Применение [Какова область применения?]: (медицина;
  вычислительная техника)
ENDF
FRAME = Действие
Сообщение: ()
ENDF

RULE 1
  = (Область.Применение; медицина)
  = (Тип.Решаемые задачи; диагностика)
DO
  = (Метод представления знаний; Правила-продукции с
    представлением нечетких знаний) 90
ENDR

RULE 2
  = (Область.Применение; вычислительная техника)
  = (Тип.Решаемые задачи; проектирование)
DO
  = (Метод представления знаний; Фреймы) 100
  = (Метод представления знаний; Правила-продукции с
```

представлением нечетких знаний) 70
= (Метод представления знаний; Семантические сети) 70
MS (Действие.Сообщение; Доказано правило 4)
ENDR

ФРЕЙМЫ

Фреймы используются в базе знаний для описания объектов, событий, ситуаций, прочих понятий и взаимосвязей между ними. Фрейм - это структура данных, состоящая из слотов (полей). Формат внешнего представления фреймов (в текстовом файле) выглядит следующим образом:

FRAME (<тип фрейма>) = <имя фрейма>
PARENT: <имя фрейма-родителя>
OWNER: <имя фрейма-владельца>
<имя слота 1> (<тип слота>) [<вопрос слота>?]: (<значение 1>;
<значение 2>; ... ;
<значение k>)
<имя слота 2> (<тип слота>) [<вопрос слота>?]: (<значение 1>;
<значение 2>; ... ;
<значение l>)
.
.
.
<имя слота n> (<тип слота>) [<вопрос слота>?]: (<значение 1>;
<значение 2>; ... ;
<значение m>)
ENDF

Фрейм может принадлежать к одному из трех типов фреймов: фрейм-класс (тип описывается зарезервированным словом "класс"), фрейм-шаблон (тип описывается зарезервированным словом "шаблон"), фрейм-экземпляр (тип описывается зарезервированным словом "экземпляр"). В базе знаний содержатся фреймы-классы и фреймы-шаблоны. При создании базы знаний тип фрейма-класса можно не описывать, этот тип фрейма понимается по умолчанию. Явно следует описывать только тип фрейма-шаблона.

В базе данных хранятся только фреймы-экземпляры. Так как для хранения фреймов-экземпляров используется специальный файл с расширением *.dtb, явно их тип в этом файле также можно не описывать. (Описание типов фреймов-классов и фреймов-экземпляров используется по преимуществу во внутреннем представлении базы знаний и базы данных).

ИМЯ ФРЕЙМА, ФРЕЙМА-РОДИТЕЛЯ, ФРЕЙМА-ВЛАДЕЛЬЦА, СЛОТА

Имена фрейма, фрейма-родителя, фрейма-владельца, слота - это последовательность символов (русские и/или латинские буквы, цифры, пробелы, знаки подчеркивания).

Тип слота

Тип слота может принадлежать к одному из трех типов: символьный, численный, лингвистический. Описание типа слота определяет тип возможных значений слота. Обязательным является описание типов слотов численного (описывается зарезервированным словом "численный") и лингвистического (описывается зарезервированным словом "лп"). Слот без описания типа понимается как символьный по умолчанию.

Вопрос слота

Вопрос слота - любая последовательность символов. Вопрос слота не является обязательным. В таком случае, в процессе логического вывода, при возникновении необходимости задать вопрос пользователю, касающийся определения значения данного слота, пользователю будет предложена формулировка: "Выберите значение" или "Введите значение".

Значение слота

Значение слота - любая последовательность символов. Значения слота разделяются точками с запятыми. Список значений слота не обязателен, он может отсутствовать, в таком случае пустые круглые скобки необязательны. Во фрейме-экземпляре у каждого слота может быть только единственное значение, во фреймах-классах и фреймах-шаблонах число значений слотов не ограничено.

С помощью специальных слотов parent и owner фреймы могут объединяться в деревья. Кроме того,

между фреймами могут существовать и произвольные связи через обычные слоты (значением слота в этом случае является имя другого фрейма).
Примеры фреймов:
FRAME = Цель
Метод представления знаний: ()
ENDF
FRAME = Тип
Решаемые задачи: (диагностика; проектирование)
ENDF
FRAME = Область
Применение [Какова область применения?]: (медицина; вычислительная техника)
ENDF
FRAME = Количество
Число правил в базе знаний (численный): ()
Число объектов в базе знаний (численный): ()
ENDF
FRAME = Действие
Сообщение: ()
ENDF

ПРАВИЛА-ПРОДУКЦИИ (ПРАВИЛА)

Правила используются в базе знаний для описания отношений между объектами, событиями, ситуациями и прочими понятиями. На основе отношений, задаваемых в правилах, выполняется логический вывод. В условиях и заключениях правил присутствуют ссылки на фреймы и их слоты.
Формат внешнего представления правил (в текстовом файле) выглядит следующим образом:
RULE <номер правила>
<условие 1>
<условие 2>
.
.
.
<условие m>
DO
<заключение 1>
<заключение 2>
.
.
.
<заключение n>
ENDR

Номер правила

Номер правила - целое число. Начало нумерации и порядок нумерации может быть произвольным, но из соображений целесообразности лучше нумеровать правила по порядку и начинать нумерацию с единицы.

Условие и заключение

Формат записи условий и заключений одинаков и имеет следующий вид:
<отношение> (<имя слота>; <значение слота>) <коэффициент достоверности>

Отношение

Отношения в условиях и заключениях могут быть EQ/= (равно), LT/< (меньше), GT/> (больше), EX (запуск внешней программы), MS (выдача сообщения), FR (вывод фрейма-экземпляра). В заключениях правил используются только отношения EQ/= (равно), EX (запуск внешней программы), MS (выдача сообщения) и FR (вывод фрейма-экземпляра). Для строковых значений слотов могут использоваться только отношения EQ/= (равно), EX (запуск внешней программы), MS (выдача сообщения), FR (вывод фрейма-экземпляра). Для слотов лингвистического типа допустимы все отношения, так как с ними связаны как строковые, так и численные значения.

Имя слота

Имя слота может быть локальным или глобальным. Локальное имя слота - имя, соответствующее имени слота в некотором фрейме. Глобальное имя слота - имя фрейма, которому принадлежит слот и собственно имя слота, разделенные точкой.

Пример локального имени слота: *Применение*

Пример глобального имени слота: *Область.Применение*

Значение слота

Значение слота - строка или число, в зависимости от типа слота. Если в качестве значения слота используется имя фрейма-шаблона, то в процессе логического вывода выполняется одновременное определение значений для всех слотов данного фрейма.

Коэффициент достоверности

Коэффициент достоверности - число от 0 до 100. Коэффициент достоверности в заключении используется при формировании значения слота фрейма-экземпляра при срабатывании правила.

Примеры правил:

```
RULE 1
= (Область.Применение; медицина)
= (Тип.Решаемые задачи; диагностика)
DO
= (Метод представления знаний; Правила-продукции с представлением нечетких знаний) 90
ENDR
RULE 2
= (Область.Применение; вычислительная техника)
= (Тип.Решаемые задачи; проектирование)
DO
= (Метод представления знаний; Фреймы) 100
= (Метод представления знаний; Правила-продукции с представлением нечетких знаний) 70
= (Метод представления знаний; Семантические сети) 70
MS (Действие.Сообщение; Доказано правило 4)
ENDR
```

ИНТЕРПРЕТАЦИЯ ПРАВИЛ-ПРОДУКЦИЙ

Интерпретация правил начинается с выбора цели логического вывода. В качестве цели логического вывода используются целевые слоты, содержащиеся во фрейме-классе со специальным именем "Цель".

Далее определяется правило, в заключении которого присутствует выбранный целевой слот.

После определения правила начинается его интерпретация (перебор и проверка условий). При проверке условия ищется соответствующий слот. Первоначальный поиск выполняется в базе данных. Если слот имеет значение, то оно используется при проверке условия. Если значения нет, то значение слота запрашивается у пользователя, с использованием меню выбора символьных значений, или окна для ввода численного значения, или того и другого в случае слота лингвистического типа. Слот в условии может указываться своим локальным именем или глобальным (с указанием имени фреймов). При локальном имени слота поиск начинается с фрейма, использованного последним при логическом выводе. Такой фрейм считается текущим. Имя текущего фрейма хранится в качестве значения слота специального фрейма, описывающего контекст диалога. Этот фрейм всегда доступен для проверки условия в правилах.

При вводе пользователем значения слота лингвистического типа, формируется численное значение с коэффициентом достоверности равным 100, если пользователь ввел число, если пользователь выбрал символьное значение, формируется символьное значение с коэффициентом достоверности равным 100. Если значение слота в правиле было символьным, а пользователем было введено численное значение, то коэффициент достоверности формируется как значение функции принадлежности лингвистической переменной (введенное пользователем число используется в качестве аргумента функции принадлежности).

Коэффициент достоверности набора условий вычисляется как коэффициент достоверности конъюнкции (минимальное значение из значений коэффициентов достоверности условий).

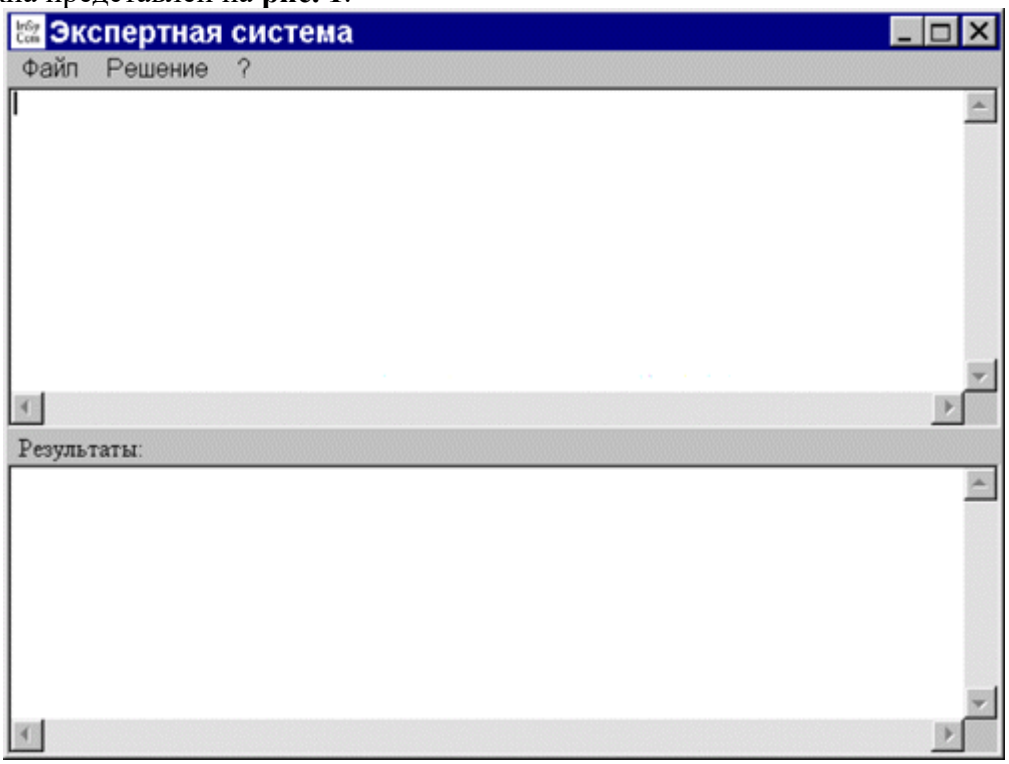
Коэффициент достоверности слота фрейма-экземпляра, формируемого на основе заключения, вычисляется как произведение коэффициент достоверности набора условий и коэффициента достоверности заключения. Если такой слот во фрейме-экземпляре уже есть, то его коэффициент достоверности меняется на новое значение, вычисляемое по формуле:

$$КД_{результирующий} = КД_{исходного\ слота} + КД_{набора\ условий} * (1-КД_{исходного\ слота})$$

ПОРЯДОК РАБОТЫ С ПРОГРАММНОЙ ОБОЛОЧКОЙ

Исполняемый модуль программной оболочки находится в файле **ESWin.exe**.

Общий вид окна представлен на **рис. 1**.



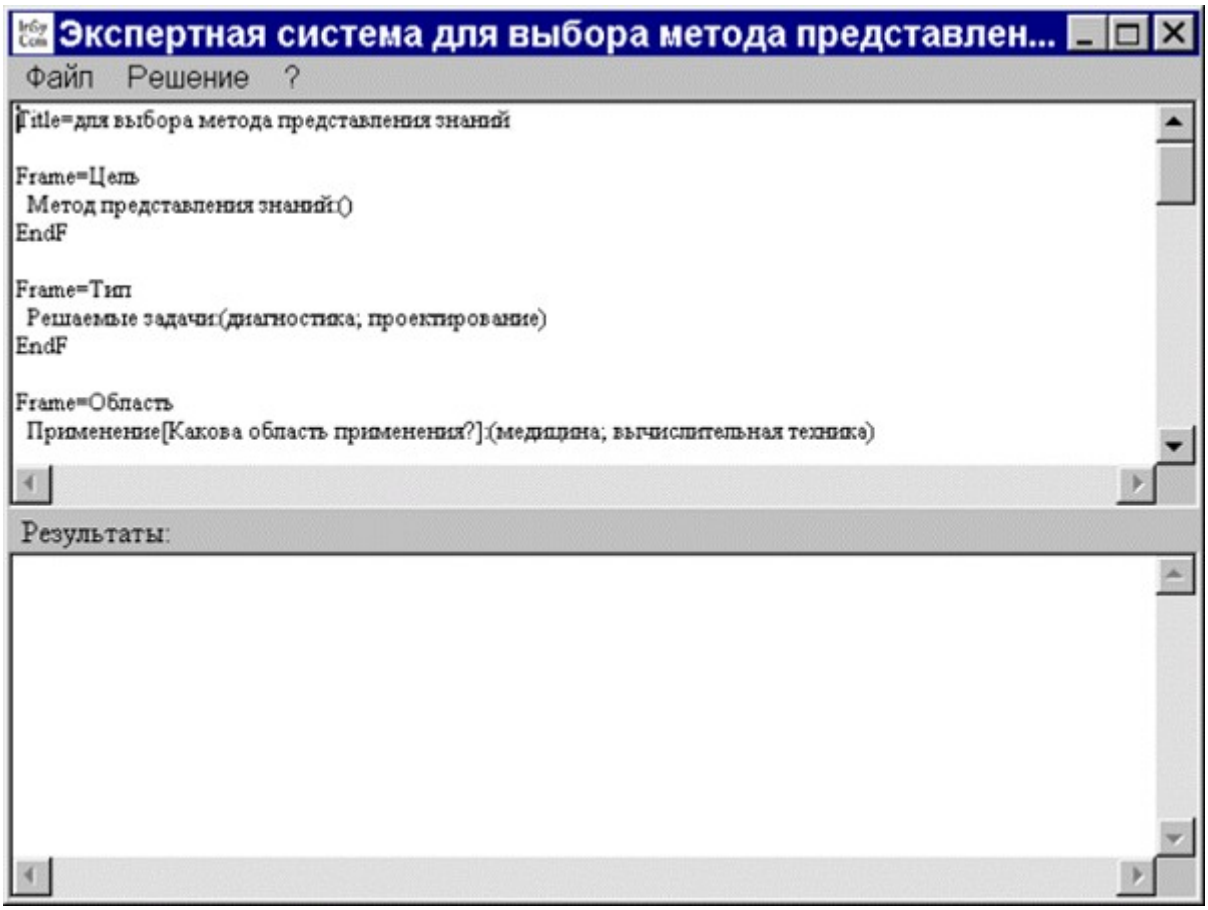
В строке заголовка окна при работе с конкретной базой знаний (экспертной системой) выводится название экспертной системы (строка, задаваемая в базе знаний зарезервированным словом **TITLE**).

Строка меню состоит из пунктов: "**Файл**", "**Решение**" и "**?**".

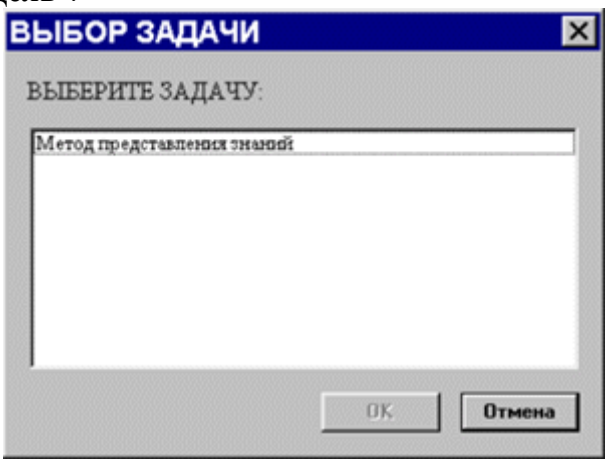
Работа с конкретной базой знаний начинается с ее загрузки. Для этого используется пункт меню "**Файл**"/"**Загрузить базу знаний**". База знаний находится в файле расширением ***.klb**. Если в загруженной базе знаний во фреймах-классах используются слоты лингвистического типа, то файл с описанием лингвистических переменных загружается автоматически.

При необходимости можно загрузить базу данных из одноименного файла с расширением ***.dtb**. Для этого используется пункт меню "**Файл**"/"**Загрузить базу данных**".

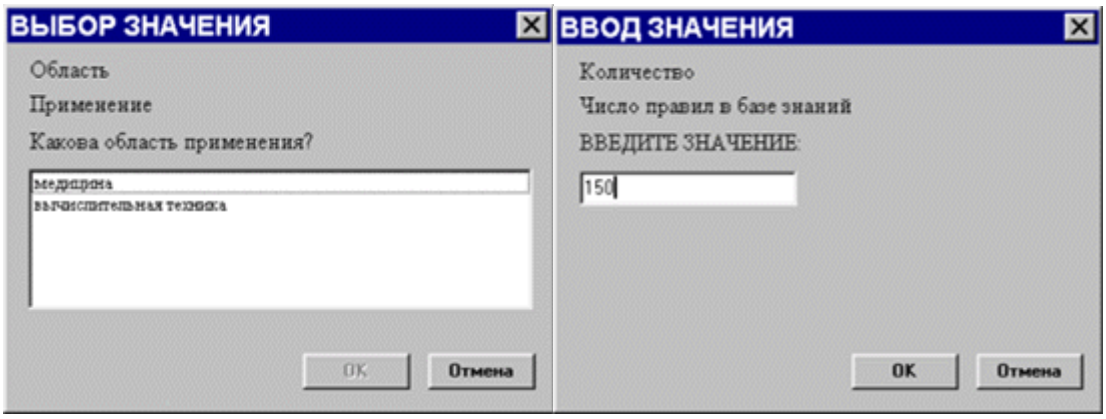
После загрузки фреймы и правила базы знаний отображаются в верхней части основного окна (**рис. 2**)



Для начала логического вывода используется пункт меню "**Решение**"/"**Поиск решения**". После выбора этого пункта меню на экране появляется окно "**Выбор задачи**" с перечнем целей логического вывода, одну из которых требуется выбрать (**рис. 3**). Перечень целей логического вывода описывается во фрейме-классе с именем "**Цель**".

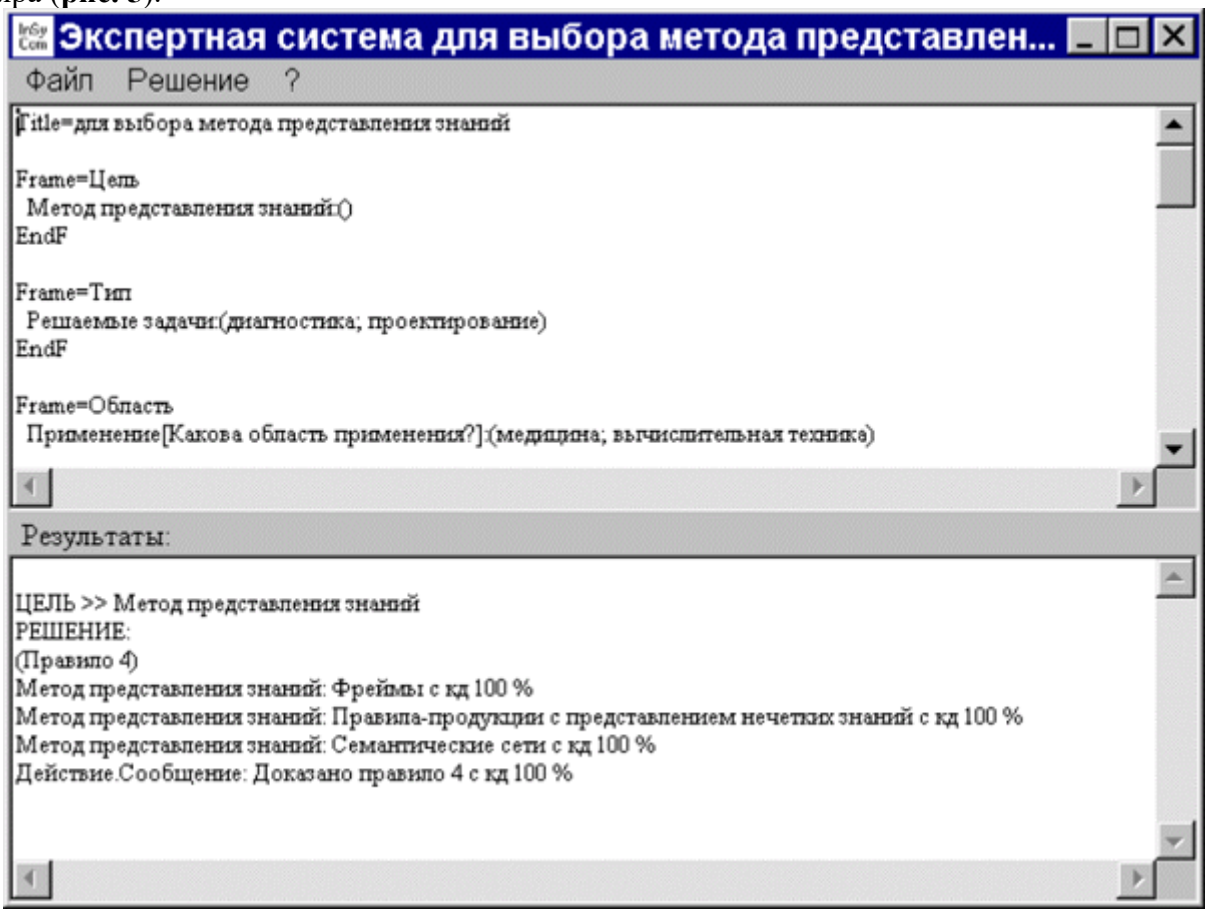


В процессе логического вывода, в качестве ответов на вопросы, задаваемые программной оболочкой, пользователю предлагается выбирать одно из символьных значений или вводить численное значение (**рис. 4**).

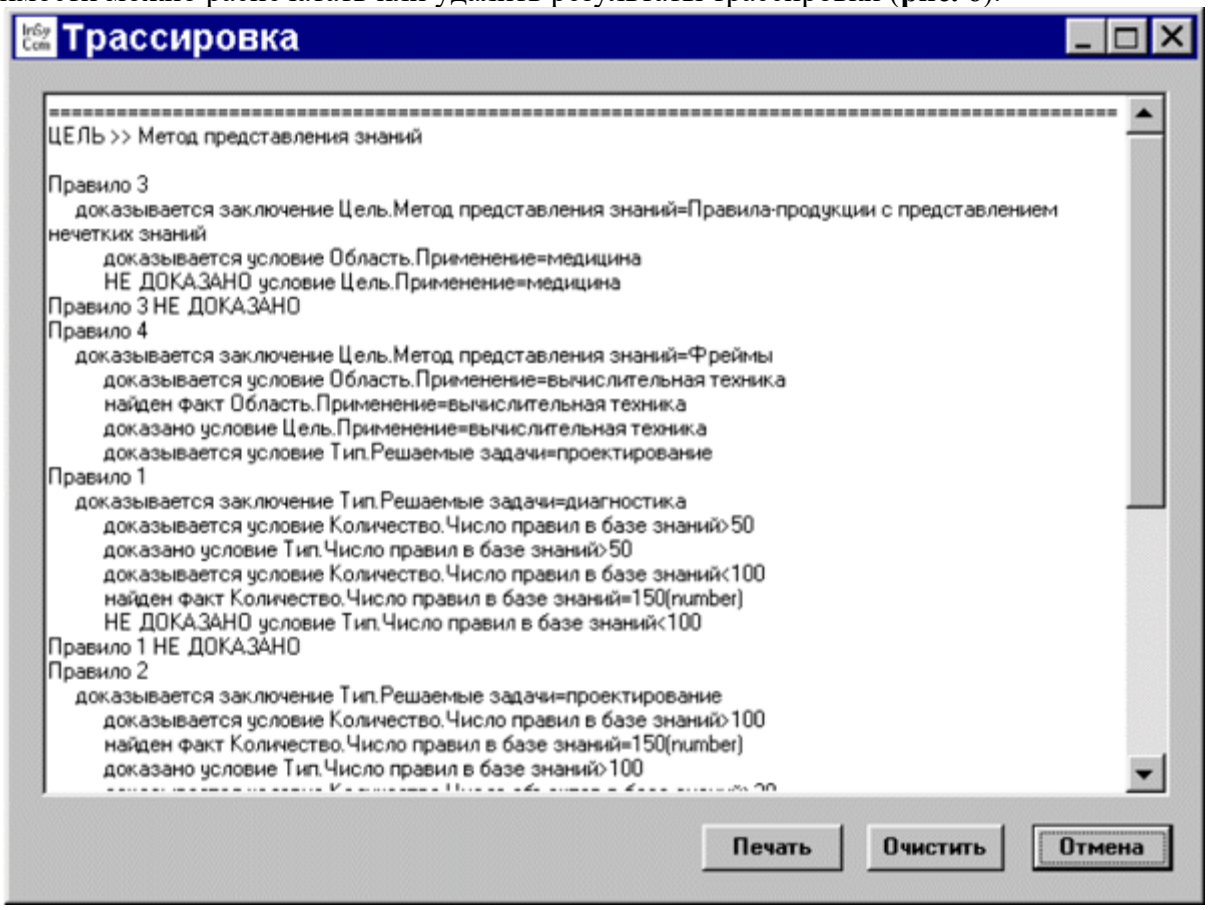


В случае с лингвистической переменной в одном окне будет предложено выбрать одно из символьных значений или ввести численное значение.

Результаты логического вывода отображаются в нижней части основного окна с комментариями, каким образом было получено решение: в результате доказательства какого-либо правила или фрейма-экземпляра (**рис. 5**).

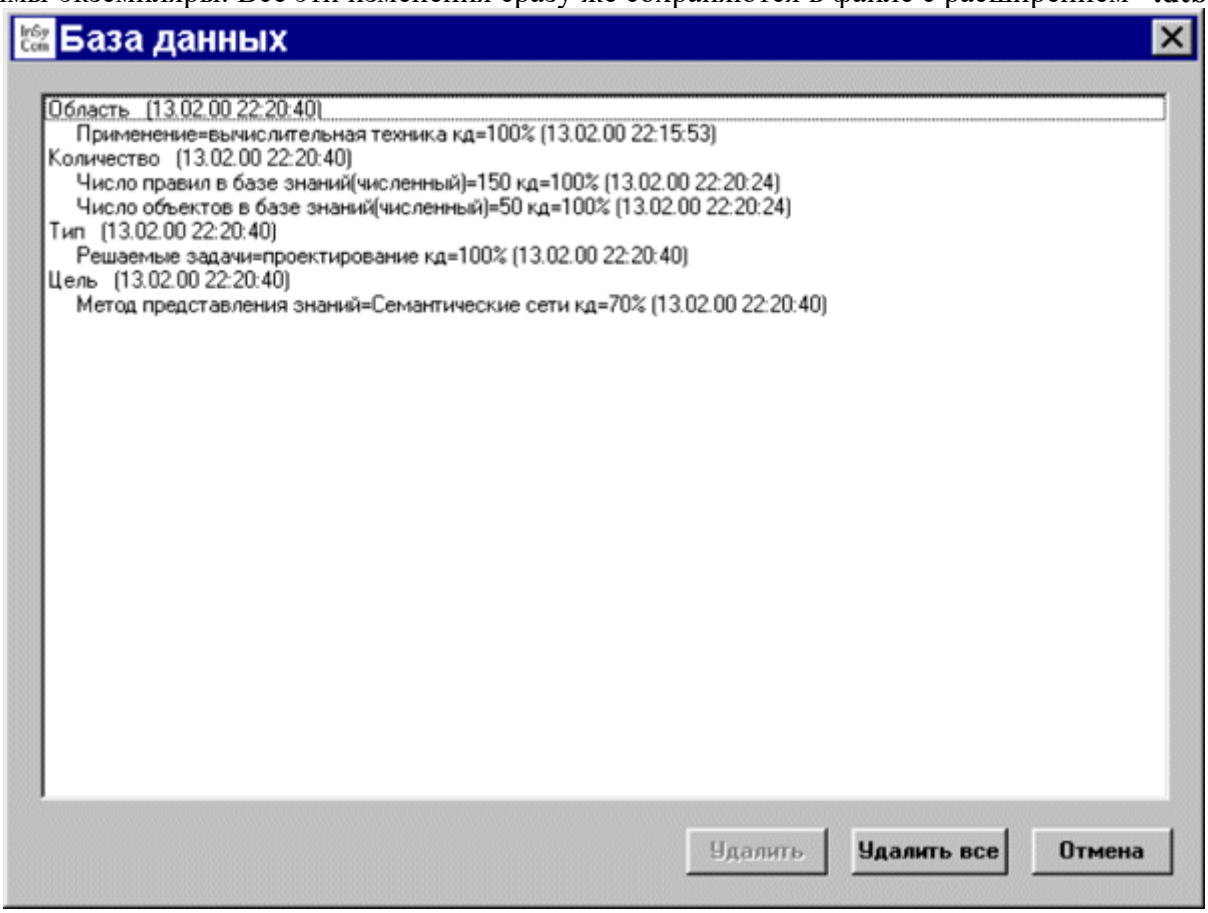


Для просмотра последовательности шагов, выполненных программной оболочкой в процессе логического вывода, можно воспользоваться пунктом меню "**Решение**"/"**Трассировка**". При необходимости можно распечатать или удалить результаты трассировки (**рис. 6**).



Для просмотра фреймов-экземпляров, полученных в результате вывода можно воспользоваться пунктом меню "**Решение**"/"**Просмотр базы данных**" (**рис. 7**) или просмотреть содержимое файла с

расширением ***.dtb** (этот файл постоянно обновляется в процессе логического вывода). При необходимости можно удалить отдельный слот во фрейме-экземпляре, полностью фрейм-экземпляр, все фреймы-экземпляры. Все эти изменения сразу же сохраняются в файле с расширением ***.dtb**.



Пункт меню "**Решение**"/"**Очистка базы данных**" используется для удаления всех фреймов-экземпляров из загруженной базы данных. То же действие можно проделать и с помощью пункта меню "**Решение**"/"**Просмотр базы данных**" (кнопка "**Удалить все**").

Пункты меню "**?**"/"**Вызов справки**" и "**?**"/"**О программе**" используются для получения справочной информации и сведений о программе.

Для завершения работы с программной оболочкой **ESWin** используется пункт меню "**Файл**"/"**Выход**".

Семантические сети

Термин семантическая означает смысловая, а сама семантика - это наука, устанавливающая отношения между символами и объектами, которые они обозначают, т.е. наука, определяющая смысл знаков,

Семантическая сеть- это ориентированный граф, вершины которого - понятия, а дуги - отношения между ними.

Понятиями обычно выступают абстрактные или конкретные объекты, а отношения - это связи типа: "это" ("is"), "имеет частью" ("haspart"), "принадлежит", "любит". Характерной особенностью семантических сетей является обязательное наличие трех типов отношений:

класс - элемент класса;

свойство - значение;

пример элемента класса.

Можно ввести несколько классификаций семантических сетей. Например, по **количеству типов отношений**:

однородные (с единственным типом отношений);

неоднородные (с различными типами отношений).

По типам отношений:

бинарные (в которых отношения связывают два объекта);

парные (в которых есть специальные отношения, связывающие более двух понятий).

Наиболее часто в семантических сетях используются следующие отношения:

связи типа "часть-целое" ("класс-подкласс", "элемент-множество" и т.п.);

функциональные связи (определяемые обычно глаголами "производит", "влияет"...);

количественные (больше, меньше, равно...);

пространственные (далеко от, близко от, за, под, над...);

временные (раньше, позже, в течение...);

атрибутивные связи (иметь свойство, иметь значение...);

логические связи (и, или, не) и др.

Проблема поиска решения в базе знаний типа семантической сети сводится к задаче поиска фрагмента сети, соответствующего некоторой подсети, соответствующей поставленному вопросу.

Пример. На рисунке изображена семантическая сеть. В качестве вершин понятия: Человек, Иванов, Волга. Автомобиль, Вид транспорта. Двигатель.

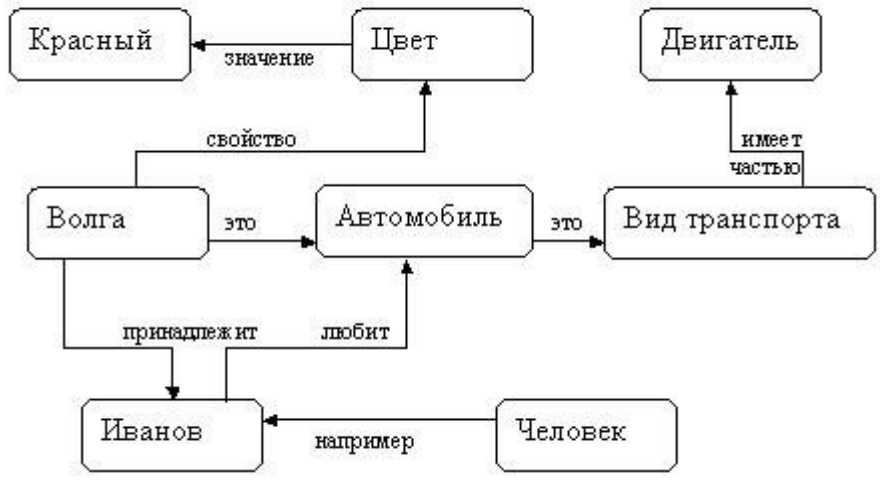


Рис. Семантическая сеть.

Основное преимущество этой модели - в соответствии современным представлениям об организации долговременной памяти человека.

Недостаток модели - сложность поиска вывода на семантической сети.

Для реализации семантических сетей существуют специальные сетевые языки, например NET и др. Широко известны экспертные системы, использующие семантические сети в качестве языка представления знаний - PROSPECTOR, CASNBT, TORUS.

При формализации созданной семантической сети как правило используется ее матричное (или табличное) отображение.

ПРАКТИЧЕСКАЯ РАБОТА № 35
СОЗДАНИЕ СЕТЕВОГО СЕРВЕРА И СЕТЕВОГО КЛИЕНТА

Цель работы: Изучение протоколов семейства TCP/IP, функций и методов стандарта WINSOCK, определяющего сетевой интерфейс для программирования сокетов в ОС MicrosoftWindows. Разработка программы-клиента в архитектуре взаимодействия “клиент-сервер” с использованием семейства протоколов TCP/IP.Разработка программы-сервера в архитектуре взаимодействия “клиент-сервер” с использованием семейства протоколов TCP/IP и библиотеки WinSock.

Форма отчета:

−выполнить задание;

−показать преподавателю;

−ответить на вопросы преподавателя.

Время выполнения: 2 ч

Задание 1: Разработать программу клиента, которая должна:

- запрашивать у пользователя адрес программы-сервера;
- устанавливать соединение с сервером;
- передавать на сервер данные;
- принимать ответ от сервера и выводить его на экран;
- закрывать соединение с сервером.

Основные сведения:

Для того чтобы установить связь с процессом, выполняющимся на другой ЭВМ, клиентская программа должна создать сокет. Сокет в любой современной системе представляет собой особый вид файла, из которого можно читать и в которой можно записывать двоичные данные. При операциях обмена с сокетом нет никакого контроля типов, эта задача возлагается на приложения. На схеме алгоритма (см. Error: Reference source not found) блок “Создание сокета” подразумевает вызов функции socket, который в случае успеха создает сокет – потоковое или дейтаграммное – и семейство протоколов. В данном случае создается потоковый сокет и используется семейство протоколов TCP/IP.

Установка соединения с другим процессом заключается в обмене специальными пакетами и возможно только тогда, когда тот процесс ожидает приема соединений. В противном случае результат операции будет неудачным и будет получено сообщение о том, что-либо не удалось установить соединение, либо оно было разорвано (зависит от реализации). Для установки соединения требуется указать ЭВМ по IP-адресу или по доменному имени, которое обязательно должно быть преобразовано в IP-адрес, и процесс на этой ЭВМ (по целочисленному идентификатору, называемому портом). Все это реализуется при помощи функции connect. Если соединение успешно установлено, то сразу после вызова этой функции можно вести обмен с гарантированной доставкой пакетов. В противном случае работа невозможна.

Передача и прием данных, то есть обмен с сокетом, производится всеми доступными в системе средствами обмена с файлами, например, системные вызовы read и write в UNIX и Windows, библиотечные функции fprintf, fgets и т. д. Перед осуществлением передачи данные, если это необходимо, шифруются каким-либо алгоритмом. На принимающей стороне полученные данные дешифруются, и определяется (или опровергается) их подлинность, от результата чего зависит дальнейшая работа с этим клиентом.

Разрыв соединения означает обмен специальными пакетами и может производиться при помощи системного вызова close. Функция close уничтожает сокет, делая его непригодным к использованию (любая операция с ним будет заканчиваться неудачей).

Задание 2: Разработать программу сервера, которая должна:

- ожидать запросов от программ клиентов на соединение;
- устанавливать соединение с клиентами; принимать данные от клиентов и выполнять их обработку;
- пересылать результат обработки клиенту.

Основные сведения:

Основной задачей серверной части является обработка. Обмен данными с клиентскими процессами есть важная составляющая часть этой задачи.

Для того чтобы процессы-клиенты могли связаться с сервером, сервер создает сокет для обмена данными. На схеме алгоритма (см. Error: Reference source not found) это представлено блоком “Создание сокета”.

Производится так же, как и в клиентской программе.

Следующий блок – “Получение локального адреса” – принципиально важен. Он служит для того, чтобы все запросы на соединения, приходящие на данную ЭВМ и обращающиеся к указанному порту, операционная система направляла данному процессу. Операция производится посредством системного вызова bind, в котором указывается созданный ранее сокет, IP–адрес ЭВМ (как правило, это константа 0) и идентификатор процесса, т. е. порт. После этого, в случае успеха, программа сервера вызывает функцию listen, которая говорит операционной системе о том, что процесс ожидает поступления запросов на соединение к данному сокету и, что эти запросы нужно ставить в очередь указанной длины (в штуках).

Получение запроса на соединение происходит тогда, когда клиентский процесс вошел в блок “Установка соединения”, т. е. вызвал функцию connect. ОС сервера при этом создает копию сокета, чтобы программа могла на первом экземпляре продолжить работу, а на другом – вести обмен с подключившимся клиентом. Следующий блок – “Создание нового потока” – подразумевает порождения новой параллельной ветки программы, которая будет вести обработку данных. В системах Windows это обычно нить (thread), создаваемая при помощи функции _beginthread, в UNIX-системах это новый процесс, создаваемый при помощи вызова fork.

Передача и прием данных, т. е. обмен с сокетом, производится всеми доступными в системе средствами обмена с файлами, например: системные вызовы read и write в UNIX и Windows, библиотечные функции fprintf, fgets и т. д.После приема данных они дешифруются с целью, во-первых, получить открытый текст и, во-вторых, чтобы определить подлинность данных. Затем если установлена подлинность данных и получен корректный открытый текст, производится обработка данных. Перед отправкой клиенту результатов работы данные снова шифруются.

ПРАКТИЧЕСКАЯ РАБОТА № 36
РАЗРАБОТКА ТЕСТОВОГО СЦЕНАРИЯ ПРОЕКТА

Цель: получить навыки разработки тестовых сценариев.

Форма отчета:

−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Оценка стоимости и причины ошибок в программном обеспечении.

Виды и методы тестирования.

Понятие теста.

Требования к разработке тестовых сценариев.

Правила разработки тестовых сценариев.

Задание № 1. Написать программу решения квадратного уравнения $ax^2 + bx + c = 0$.

Задание № 2. Найти минимальный набор тестов для программы нахождения веще-ственных корней квадратного уравнения $ax^2 + bx + c = 0$. Решение представлено в таблице.

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

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

Заповеди по отладки программного средства, предложенные Г. Майерсом.

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

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

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

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

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

Задание № 6. Разработайте набор тестовых сценариев (как позитивных, так и негативных) для следующей программы:

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

Строка 1	Строка 2	Вывод
абвгабвг	аб	2
стстсап	стс	2

Набор тестовых сценариев запишите в виде таблицы, приведенной выше.

Задание № 3. Оформите отчет.

ПРАКТИЧЕСКАЯ РАБОТА № 37

РАЗРАБОТКА ТЕСТОВЫХ ПАКЕТОВ

Цель: получить навыки разработки тестовых пакетов.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Системные основы разработки требований к сложным комплексам программ.

Формализация эталонов требований и характеристик комплекса программ.

Формирование требований компонентов и модулей путем декомпозиции функций комплексов программ.

Тестирование по принципу «белого ящика».

Задание № 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;

Тест	Ожидаемый результат	Фактический результат	Результат тестирования
…	…	…	…

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:

Задание № 3. Проверить все виды тестов и сделать выводы об их эффективности.

Задание № 4. Оформить отчет.

ПРАКТИЧЕСКАЯ РАБОТА № 38
ИСПОЛЬЗОВАНИЕ ИНСТРУМЕНТАРИЯ АНАЛИЗА КАЧЕСТВА

Цель: получить навыки использования инструментария анализа качества.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Общие требования к качеству функционирования сложных программных комплексов.

Требования к характеристикам качества сложных программных комплексов.

Требования к эффективности использования ресурсов ЭВМ программным комплексом в реальном времени.

Проверка корректности функциональных требований к сложным комплексам программ.

Задание № 1. Написать программу, генерирующую массив вещественных чисел в диапазоне от –10 до 10 и определяющую все минимальные положительныеэлементы.

Задание № 2. Оценить эффективность разработанной программы:

	Исходнаяпрограмма		Улучшеннаяпрограмма	
	Недостатки	Количественная оценка	Улучшения	Количественная оценка
Время выполнения				
Оперативная память				
Внешняя память				

Задание № 3. Оценить качество разработанной программы:

	Правильность	Универсальность	Проверяемость	Точность результатов
Недостатки				
Оценка				

Задание № 4. Оформить отчет.

ПРАКТИЧЕСКАЯ РАБОТА № 39

АНАЛИЗ И ОБЕСПЕЧЕНИЕ ОБРАБОТКИ ИСКЛЮЧИТЕЛЬНЫХ СИТУАЦИЙ

Цели: получение навыков анализа и обеспечения обработки исключительных ситуаций.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Исключения в C++.

Установленные исключения.

Спецификация исключения.

Задание № 1. Написать программу, в которой обрабатываются следующие исключительные ситуации: "отрицательное значение возраста" и "год рождения, больше текущего":

```
#include <iostream>
#include <ctime>
using namespace std;

int AgeCalc(int year)
{
    if (year <= 0)
    {
        throw "ERROR: negative value of the birth year!!!";
    }


    struct tm *CDate;
    time_t tt = time(NULL);
    CDate = localtime(&tt);

    if (year > (1900 + CDate->tm_year) )
    {
        throw "ERROR: The birth year value is greater than current year value!!!";
    }

    return 1900 + CDate->tm_year - year;
}

          int main()
          {
              int BYear = 1980;
              int PAge = 0;

              try{
                  PAge = AgeCalc(BYear);
              }
              catch (const char * s)
              {
                  cout << s << endl<<endl;
              }
              catch(...)
              {
                  cout << "Unknown exception" << endl<<endl;
              }


              cout << "For birth year " << BYear << " the age is " << PAge << endl;

              return 0;
          }
```

Задание № 2. Составить программу циклического вычисления значений функций, определенных из таблицы вариантов заданий. Значения R должны вводиться с клавиатуры. R1 и R2 –вещественные, R3 –комплексное. Предусмотреть вывод подсказок в виде(например):

Funkciyasin(x)

Q –

VyihodizprogrammyiVve

ditechisloili Q:

Для вычисления значений функции написать функцию, вычисляющую требуемые по заданию значения. При разработке функции разрешается использовать функции модуля math.h.

Предусмотреть анализ всей введенной информации на ошибки, обработку ошибок реализовать с использованием с использованием обработчиков try… в зависимости от варианта задания. Предусмотреть вывод имени функции, в которой произошла ошибка. Вывод на экран и

чтение с клавиатуры организовать при помощи стандартных потоков ввода/вывода/ошибки. Вывести исходные данные и результат ввиде(например):

Sin(R) = rez;

Где rez –результаты вычисления (вещественный).

Вариант	Функция	Обработчики
1	Sin(R1)*(pi)/R2-R3	Потеря разряда Деление на 0
2	Sin(R2)/pi*R1+R3	Потеря разряда Переполнение
3	Tan(R1)/R3+Cmod(R3)	Потеря разряда Прерывание
4	Arctan(R1)*R2+ R3	Потеря разряда Переполнение
5	Ln(R1-R2)*R2-R3	Обл.опр.арг. Исчезновение порядка

Задание № 3. Реализуйте класс «очередь» из строк. Реализуйте методы для вставки в очередь и удаления. Породите и обработайте ошибки динамического выделения памяти, переполнения очереди.

Задание № 4. Оформите отчет.

ПРАКТИЧЕСКАЯ РАБОТА № 40

ФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ

Цель: получение навыков проведения функционального тестирования.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Особенности функционального тестирования программного обеспечения(тестирования «черного ящика»).

Ошибки, выявляемые при функциональном тестировании.

Задачи, решаемые при функциональном тестировании.

Задание № 1. Опишите методы формирования тестовых наборов при использовании стратегии "черного ящика":

Эквивалентноеразбиение	
Анализграничныхзначений	
Анализпричинно-следственныхсвязей	
Предположениеобошибке	

Задание № 2. Пусть необходимо выполнить тестирование программы, определяющей точку пересечения двух прямых на плоскости. Попутно, она должна определять параллельность прямой одной их осей координат.

В основе программы лежит решение системы линейных уравнений

$$Ax + By = C \text{ и} Dx + Ey = F.$$

1. Используя метод эквивалентных разбиений, получаем для всех коэффициентов один правильный класс эквивалентности (коэффициент – вещественное число) и один неправильный (коэффициент – не вещественное число). Откуда можно предложить 7тестов:

1) все коэффициенты – вещественные числа;

2)– 7) поочередно каждый из коэффициентов – не вещественное число.

2. По методу граничных условий можно считать, что для исходных данных граничные условия отсутствуют (коэффициенты – "любые" вещественные числа); для результатов – получаем, что возможны варианты: единственное решение, прямые сливаются (множество решений), прямые параллельны (отсутствиерешений).

Следовательно, можно предложить тесты, с результатами внутри области и с результатами на границе.

3. По методу анализа причинно-следственных связей определяем множество условий.

а) для определения типапрямой;

б) для определения точки пересечения.

Выделяем три группы причинно-следственных связей (определение типа и существования первой линии, определение типа и существования второй линии, определение точки пересечения) и строим таблицы истинности.

К уже имеющимся тестам добавляются:

а) проверки всех случаев расположения обеих прямых – 6 тестов по первой прямой вкладываются в 6 тестов по второй прямой так, чтобы варианты не совпадали, – 6 тестов;

б) выполняется отдельная проверка несовпадения условия $x\delta = 0$ или $y = 0$ (в зависимости от того, какой тест был выбран по методу граничных условий) – тест также можно совместить с предыдущими 6 тестами;

4. По методу предположения об ошибке добавим тест, при которомвсе коэффициенты –нули.

Всего получили 20 тестов по всем четырем методикам. Если еще попробовать вложить независимые проверки, то возможно число тестов можно еще сократить.

Задание № 3. Разработать программу определения вида треугольника, заданного длинами его сторон: равносторонний, равнобедренный, прямоугольный, разносторонний.

Номер теста	Назначение теста	Значения исходных данных	Ожидаемый результат	Реакцияп рограммы	Вывод

Предлагаемые тесты свести в таблицу.

Задание № 4. Разработать программу решения уравнения $ax^2 + bx + c = 0$, где a, b, c – любые вещестенные числа.

Предлагаемые тесты свести в таблицу.

Номер теста	Назначение теста	Значения исходных данных	Ожидаемый результат	Реакцияп рограммы	Вывод

Задание № 5. Оформить отчет.

ПРАКТИЧЕСКАЯ РАБОТА № 41

ТЕСТИРОВАНИЕ БЕЗОПАСНОСТИ

Цель: получение навыков тестирования безопасности информационной системы.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Тестирование восстановления.

Тестирование безопасности.

Технологии тестирования безопасности.

Тестирование безопасности – оценка уязвимости программного обеспечения к различным атакам.

Компьютерные системы очень часто являются мишенью незаконного проникновения. Под проникновением понимается широкий диапазон действий: попытки хакеров проникнуть в систему из спортивного интереса, месть рассерженных служащих, взлом мошенниками для незаконной наживы. Тестирование безопасности проверяет фактическую реакцию защитных механизмов,

встроенных в систему, на проникновение. В ходе тестирования безопасности испытатель играет роль взломщика. Ему разрешено все:

– попытки узнать пароль с помощью внешних средств;
– атака системы с помощью специальных утилит, анализирующихзащиты;
– подавление, ошеломление системы (в надежде, что она откажется обслуживать других клиентов);
– целенаправленное введение ошибок в надежде проникнуть в систему в ходевосстановления;
– просмотр несекретных данных в надежде найти ключ для входа всистему.

При неограниченном времени и ресурсах хорошее тестирование безопасности взломает любую систему. Задача проектировщика системы – сделать цену проникновения более высокой, чем цена получаемой в результатеинформации.

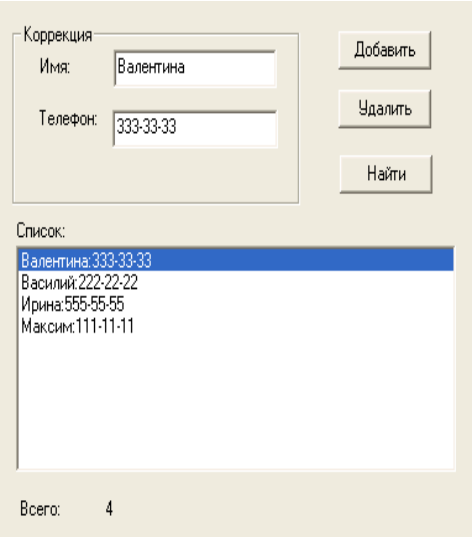
Задание № 1. Изучите и опишите одно из средств выявления уязвимостей: Таблица 1. Обзор средств выявления уязвимостей, работающих на уровне кода

Наименовани есредства	Назначение	Поддерживаем ыеязыки программирования	Примечание
Иностранныесредствавыявленияуязвимостей			
Its4	Статически просматривает исходный код для обнаружения потенциальных уязвимостей защиты	C/c++	Отмечает вызовы потенциально опасных функций, таких, как strcpy/memcpy, и выполняет поверхностный семантический анализ, пытаясь оценить, насколько опасен такой код, а также дает советы по его улучшению
Rats(rough auditing tool for security)	Просматривает исходный текст, находя потенциально опасные обращения к функциям	C/c++, php, perl, python	Использует сочетание проверок надежности защиты от семантических проверок в its4 до глубокого семантического анализа в поисках дефектов, способных привести к переполнению буфера, полученных из mops
Flawfinder	Просматривает исходный текст, находя потенциально опасные обращения к функциям	C/c++	Выполняет поиск функций, которые чаще всего используются некорректно, присваивает им коэффициенты риска (опираясь на такую информацию, как передаваемые параметры) и составляет списокпотенциально уязвимых мест, упорядочивая их по степени риска
Flexelint (pc-lint)	Производит семантический анализ исходного кода, анализ потоков данных и управления	C/c++	В конце работы выдаются сообщения нескольких основных типов: – возможен нулевой указатель – проблемы с выделением памяти (например, нет free() после malloc()) – проблемный поток управления (например, недостижимыйкод); – возможнопереполнение буфера, арифметическоепереполнение;

Наименованиесредства	Назначение	Поддерживаемыеязыкипрограммирования	Примечание
			– предупреждения о плохом и потенциальноопасномстилекода
Parasoftc++ test	Формированиетестов анализауязвимостейна уровне метода, класса, файла и проекта	C++	Генерирует тестовый код, вызывая для его подготовки компилятор visualc++
Coverity	Используется для выявления и исправления дефектов безопасности и качества в приложениях критическогоназначения	C/c++, java	Способен с минимальной положительной погрешностью обрабатывать десятки миллионов строк кода, обеспечивая 100- процентное покрытие трассы
Klocwork k7	Предназначен для автоматизированного статического анализа кода, выявления и предотвращения дефектов программного обеспечения и проблембезопасности	C/c++, java	Выявляет коренные причины недостатков качества и безопасности программного обеспечения
Codesurfer	Может применяться для поиска ошибок в исходном коде, для улучшения понимания исходногокода	C/c++	Позволяет проводить анализ указателей, использовать и определять переменные, зависимости данных, строить графывызовов
Fxcop	Способен обнаружить более 200 недочетов (или ошибок) в следующих областях: – архитектур абиблиотеки ; – правилаименования; – производительность; – безопасность	C/c++	Откомпилированный код проверяется с помощью механизмов рефлексии, парсингamsil и анализа графа вызовов
Qaudit	Быстрый анализ исходных файлов на наличие переполнения буфера, ошибок форматной строки, запросов исполняемых вызовов, переменных среды,	C/c++	Написать на интерпретируемом языке perl, прост в использовании

	и функций, имеющих проблемы защиты		
Российскиесредствавыявленияуязвимостей			
Ак-вс	Автоматизированный анализ исходных текстов, с целью выявления потенциально опасныхсигнатур	C/c++, java, pascal, c#, php, assembler	Позволяет проводить статический анализ исходных текстов, динамический анализ, имеет базы сигнатур для каждого из поддерживаемых языков программирования
Аист-с	Автоматизированный анализисходныхтекстов	C/c++	Позволяет проводить статический анализ исходных текстов
Ксаит	Автоматизированный анализисходныхтекстов	C/c++	Позволяет проводить статический анализ исходных текстов
Uca	Предназначено для выявления потенциально опасных сигнатур	C/c++, pascal, perl, plm	Имеет базы сигнатур для каждого из поддерживаемых языков программирования
Viva64	Помогает отслеживать в исходном коде потенциально опасные фрагменты, связанные с переходом от 32-битных систем к 64-битным	C/c++	Помогает писать корректный и оптимизированный код для 64- битных систем

Задание № 2. Разработать приложение, интерфейс которого представлен на рисунке.



Задание № 3. Добавить в программу форму авторизации по имени и паролю.

ПРАКТИЧЕСКАЯ РАБОТА № 42

НАГРУЗОЧНОЕ ТЕСТИРОВАНИЕ, СТРЕССОВОЕ ТЕСТИРОВАНИЕ

Цель: получение навыков проведения нагрузочного и стрессового тестирования.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Особенности нагрузочного тестирования.

Особенности стрессового тестирования.

Задание № 1. Разработать Компилятор простых арифметических выражений, например 2+ +(-5)*(7-8). Вход и выход осуществляются в виде строк.

Задание № 2. Разработать тестовый сценарий нагрузочного тестирования. Ответить на вопрос – сколько запросов в секунду может обработать приложение при условии, что они идут последовательно. Построить график зависимости времени ответа от количества параллельных запросов (рассматривать логарифмическую шкалу по основанию два, т.е. 1, 2,4,8,16,32 и т.д. запроса) Ответить на вопрос – какое максимальное количество параллельных запросов может обработать приложение без сбоев.

Задание № 3. Оформить отчет.

ПРАКТИЧЕСКАЯ РАБОТА № 43

ТЕСТИРОВАНИЕ ИНТЕГРАЦИИ

Цель: получение навыков тестирования интеграции.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Особенности тестирования интеграции.

Методы интеграционного тестирования.

Нисходящее тестирование интеграции.

Восходящее тестирование интеграции.

Сравнение нисходящего и восходящего тестирования интеграции

Задание № 1. Разработать приложение, состоящее из трех модулей:

1) главный модуль, считывающий из текстового файла координаты точек наплоскости;
2) модуль, содержащий функции расчета расстояния между двумяточками;
3) модуль, содержащий функцию, определяющую треугольник с максимальной площадью. **Задание № 2.** Описать этапы нисходящего проектирования разработанного приложения. **Задание № 3.** Описать этапы восходящего проектирования, разработанного приложений. **Задание № 4.** Оформитьотчет.

ПРАКТИЧЕСКАЯ РАБОТА № 44

КОНФИГУРАЦИОННОЕ ТЕСТИРОВАНИЕ

Цели: получение навыков проведения конфигурационного тестирования.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Особенности конфигурационного тестирования.

Конфигурационное тестирование (Configurationtesting). Проверяется работоспособность при различных конфигурациях, предполагает тестирование работы системы на различных платформах: различных вариантах аппаратной конфигурации, версиях операционной системы и окружения.

Задание № 1. Дана структура с именем ZNAK, состоящая из полей:

– фамилия,имя;
– знакЗодиака;
– дата рождения (массив из трех чисел).

Написать программу, которая выполняет следующие действия:

– ввод с клавиатуры данных в массив, состоящий из 8 элементов типа ZNAK, и занесение их в файлданных;

– чтение данных из файла и вывод их наэкран;

– вывод на экран информации о людях, родившихся в месяц, значе-ние которого введено с клавиатуры (если таких нет – вывести об этомсо-общение);

– список должен быть упорядочен по знакам Зодиака.

Задание № 2. Описать и обосновать итоги тестирования работы разработанного приложения на различных платформах: различных вариантах аппаратной конфигурации, версиях операционной системы иокружения.

ПРАКТИЧЕСКАЯ РАБОТА № 45
ТЕСТИРОВАНИЕ УСТАНОВКИ

Цель: получение навыков тестирования установки.

Форма отчета:
−выполнить задание;
−показать преподавателю;
−ответить на вопросы преподавателя.
Время выполнения: 2 ч

Теоретические вопросы

Комплексное тестирование приложения.

Задание № 1. Разработать приложение, интерфейс которого представлен на рисунке 1.

Задание № 2. Провести комплексное тестирование разработанного приложения.

Задание № 3. Оформить отчет.

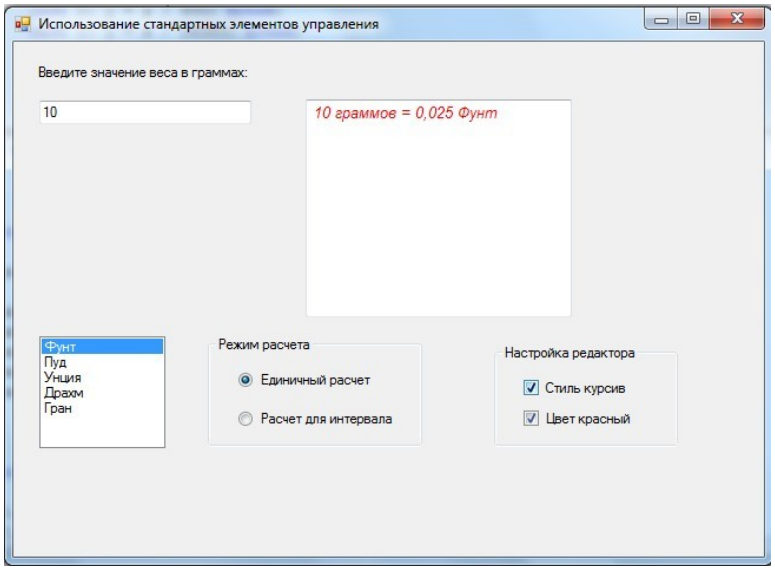


Рисунок 1

Список литературы

1. *Волкова, В. Н.* Теория информационных процессов и систем : учебник и практикум для академического бакалавриата / В. Н. Волкова. — 2-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2019. — 432 с. — (Бакалавр. Академический курс). — ISBN 978-5-534-05621-1. — Текст : электронный // ЭБС Юрайт [сайт]. — URL: https://biblio-online.ru/bcode/432843
2. Проектирование информационных систем : учебник и практикум для среднего профессионального образования / Д. В. Чистов, П. П. Мельников, А. В. Золотарюк, Н. Б. Ничепорук ; под общей редакцией Д. В. Чистова. — Москва : Издательство Юрайт, 2019. — 258 с. — (Профессиональное образование). — ISBN 978-5-534-03173-7. — Текст : электронный // ЭБС Юрайт [сайт]. — URL: https://biblio-online.ru/bcode/437463
3. *Зараменских, Е. П.* Информационные системы: управление жизненным циклом : учебник и практикум для среднего профессионального образования / Е. П. Зараменских. — Москва : Издательство Юрайт, 2019. — 431 с. — (Профессиональное образование). — ISBN 978-5-534-11624-3. — Текст : электронный // ЭБС Юрайт [сайт]. — URL: https://biblio-online.ru/bcode/445765
4. *Черткова, Е. А.* Компьютерные технологии обучения : учебник для вузов / Е. А. Черткова. — 2-е изд., испр. и доп. — Москва : Издательство Юрайт, 2019. — 250 с. — (Университеты России). — ISBN 978-5-534-07491-8. — Текст : электронный // ЭБС Юрайт [сайт]. — URL: https://biblio-online.ru/bcode/437244
5. *Маркин, А. В.* Программирование на SQL в 2 ч. Часть 2 : учебник и практикум для вузов / А. В. Маркин. — 2-е изд., испр. и доп. — Москва : Издательство Юрайт, 2019. — 340 с. — (Высшее образование). — ISBN 978-5-534-12258-9. — Текст : электронный // ЭБС Юрайт [сайт]. — URL: https://biblio-online.ru/bcode/448191