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 № 1 - 33
по дисциплине
МДК.05.02 Разработка кода информационных систем
для специальности 09.02.07Информационные системы и программирование,
квалификации
«Разработчик веб и мультимедийных приложений»

Таганрог

2023

Составители: Е.В. Михайлович

Практикум по выполнению практических работ по дисциплине «МДК.05.02 Разработка кода информационных систем». ПИ (филиала) ДГТУ в г.Таганроге, 2023г.

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

Предназначено для обучающихся по специальности 09.02.07 «Информационные системы и программирование». Квалификации выпускника: «Разработчик веб и мультимедийных приложений»

Ответственный за выпуск:

Председатель ЦМК: _____ О.В. Андриян

СОДЕРЖАНИЕ

1. Введение

В учебно-методическом пособии к практикуму по курсу «Проектирование и разработка 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.

2. 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.

3. Практические работы по дисциплине

МДК.05.02 Разработка кода информационных систем

Материально техническое обеспечение практических работ

Реализация рабочей программы предполагает наличие компьютерного класса.
Оборудование учебного кабинета и рабочих мест:
1) рабочее место для преподавателя;
2) столы, стулья на 25-30 обучающихся;
Технические средства обучения:
1) Проектор / BENQ MX506, экран для проектора / CACTUS Wallscreen CS-PSW-206x274,274x206 см,4:3, настенно-потолочный, белый
2) Персональные компьютеры с программным обеспечением:

- 7-Zip 1602
- Adobe PDF Reader 11.0
- Google Chrome
- Notepad++ 6.9.2
- OpenOffice
- Openproj 1.4
- VirtualBox 5.1.12
- Microsoft Office Pro 2016
- Windows 10

ПРАКТИЧЕСКАЯ РАБОТА № 1
ПОСТРОЕНИЕ ДИАГРАММЫ ВАРИАНТОВ
ИСПОЛЬЗОВАНИЯ И ДИАГРАММЫ ПОСЛЕДОВАТЕЛЬНОСТИ И ГЕНЕРАЦИЯ КОДА

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

Рабочее задание

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

Теоретическая часть

Универсальный язык моделирования UML. Понятие диаграммы.
Виды диаграмм.
Основные элементы диаграммы вариантов использования. Основные элементы диаграммы последовательности.

Порядок выполнения работы

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:

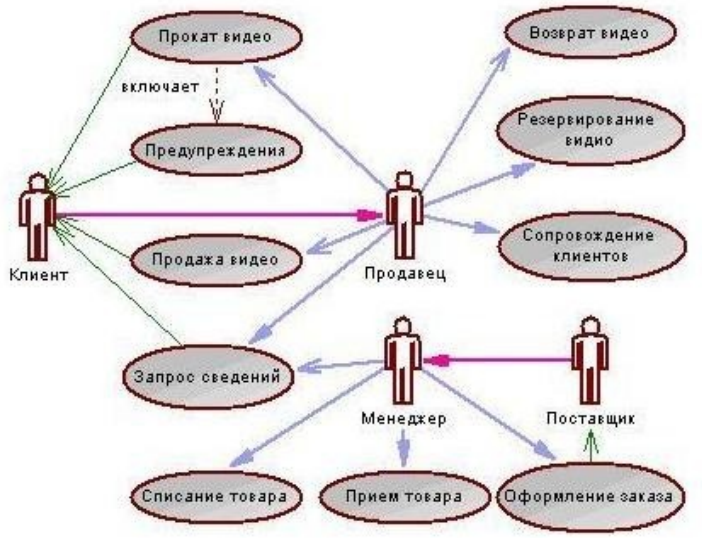


Рисунок 1

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)
Альтернативныйпоток	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).

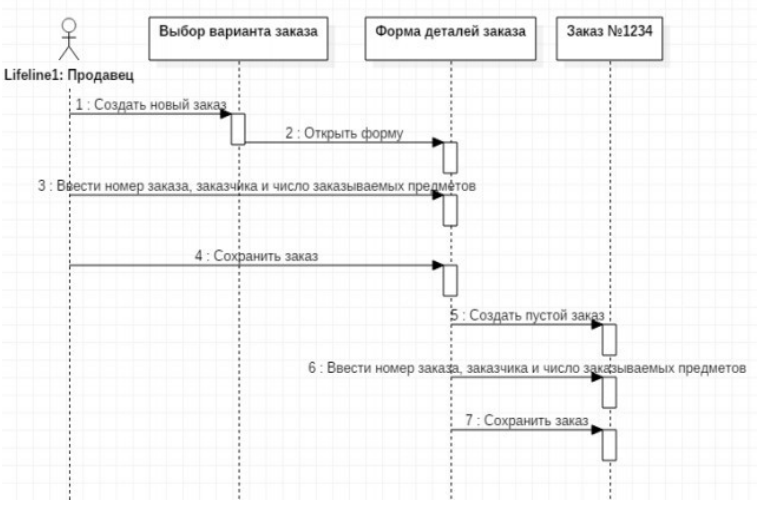


Рисунок 2

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.

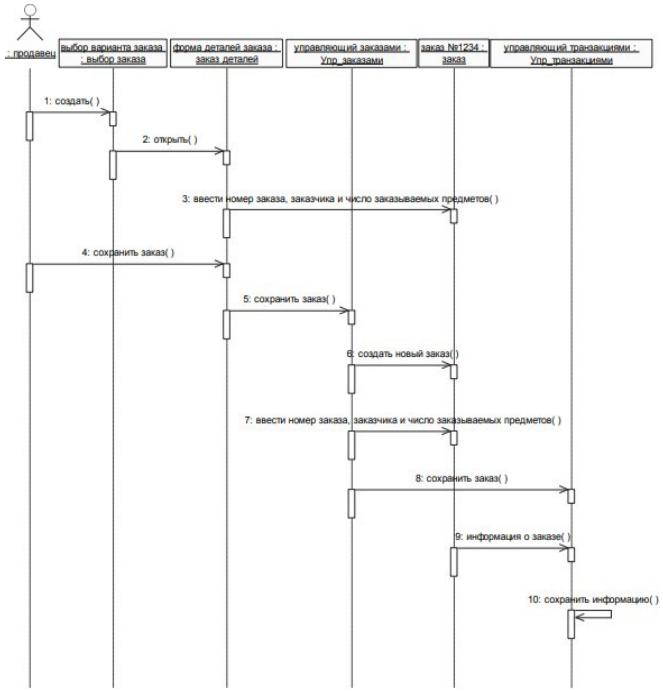


Рисунок 3

Контрольные вопросы

1. Какие основные диаграммы UML используются при моделировании информационных систем?
2. Каково назначение диаграммы классов в UML при моделировании информационных систем?
3. Какие сущности и связи описываются на диаграмме классов UML?
4. Что такое диаграмма вариантов использования UML и как она помогает при моделировании информационных систем?
5. В чем заключается роль диаграммы последовательности UML при моделировании информационных систем?
6. Какие диаграммы UML используются для описания поведения информационных систем?
7. Что такое диаграмма состояний UML и как она применяется при моделировании информационных систем?
8. Какие диаграммы UML помогают описать структуру базы данных в информационной системе?
9. Чем отличаются диаграмма компонентов и диаграмма развёртывания в UML и в чём состоит их роль при моделировании информационных систем?
10. Каким образом UML поддерживает моделирование информационных систем времени выполнения?

ПРАКТИЧЕСКАЯ РАБОТА № 2
ПОСТРОЕНИЕ ДИАГРАММЫ КООПЕРАЦИИ И ДИАГРАММЫ РАЗВЕРТЫВАНИЯ И ГЕНЕРАЦИЯКОДА

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

Рабочее задание

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

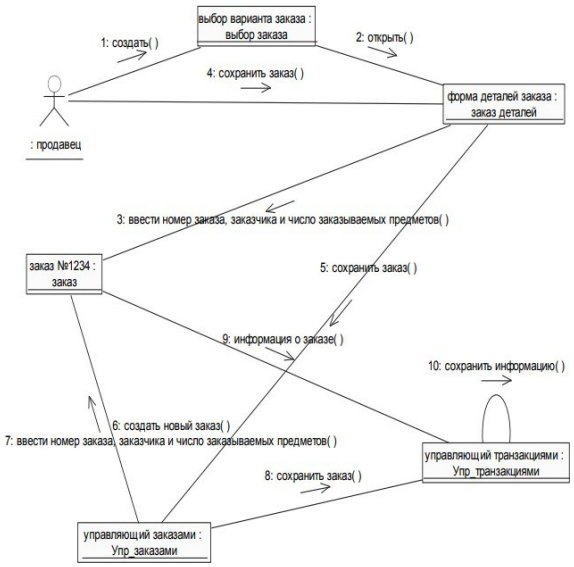
Теоретическая часть

Универсальный язык моделирования UML. Понятие диаграммы.
Виды диаграмм.
Основные элементы диаграммы кооперации. Основные элементы диаграммы развертывания.

Порядок выполнения работы

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.

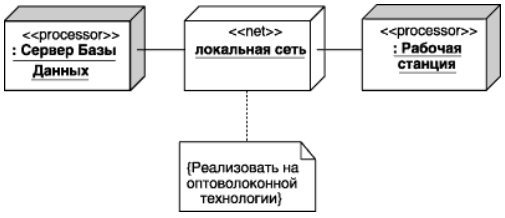


Рисунок 5
Диаграмма развертывания с отношением зависимости между узлом и развернутыми на нем компонентами приведена на рисунке 6.

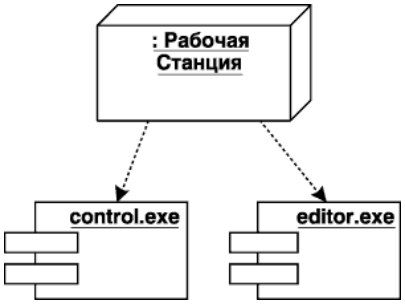


Рисунок 6

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

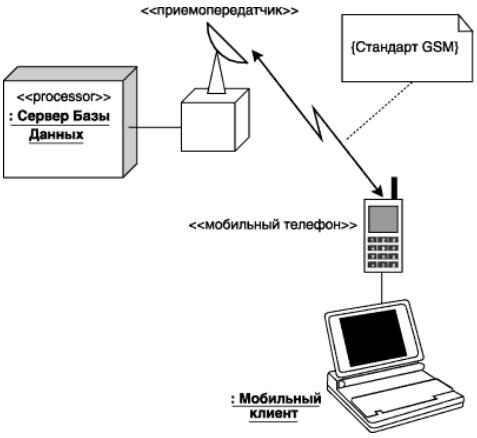


Рисунок 7

Задание № 6. Постройте диаграмму развертывания для выбранной информационной системы (практическая работа №11).
Задание № 7. Оформите отчет.

Контрольные вопросы

1. Какое место занимает UML в методологиях моделирования информационных систем?
2. Какие типы диаграмм UML наиболее часто применяются при моделировании информационных систем?
3. Каким образом диаграмма классов UML помогает описать структуру информационной системы?
4. Какие основные элементы и связи используются на диаграмме последовательности UML для моделирования работающей системы?
5. Как диаграмма случаев использования UML помогает описать функциональные требования информационной системы?
6. В чем заключается роль диаграммы активностей UML при моделировании бизнес-процессов в информационной системе?
7. Как диаграмма компонентов UML помогает описать архитектуру информационной системы?
8. Что такое диаграмма развертывания UML и как она применяется при моделировании информационных систем?
9. Каким образом диаграмма состояний UML помогает моделировать поведение объектов в информационной системе?
10. Какие инструменты и среды разработки поддерживают использование UML при моделировании информационных систем?

ПРАКТИЧЕСКАЯ РАБОТА №3
Построение диаграммы Деятельности, диаграммы Состояний и диаграммы Классов и генерация кода

Цель работы: научиться строить диаграмму вариантов использования.

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

Теоретическая часть:
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).

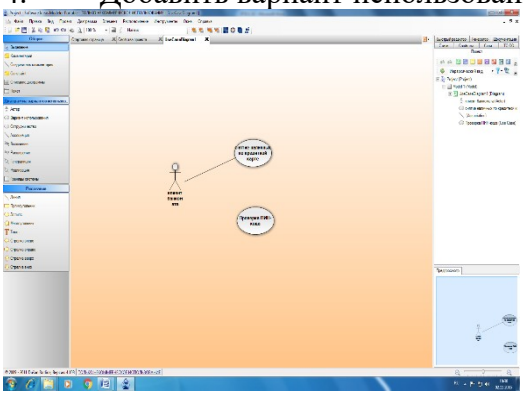
Существуют и другие виды взаимодействия, но они менее важны и реже применяются.

Порядок выполнения работы

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 № 4
ПОСТРОЕНИЕ ДИАГРАММЫ КОМПОНЕНТОВ И ГЕНЕРАЦИЯ КОДА

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

Рабочее задание:

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

Теоретическая часть

Универсальный язык моделирования UML. Понятие диаграммы.
Виды диаграмм.
Основные элементы диаграммы компонентов. Основные элементы диаграммы развертывания.

Порядок выполнения работы

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.

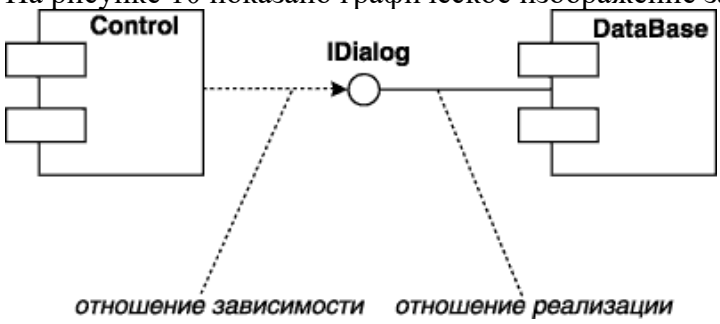


Рисунок8
Рисунок9

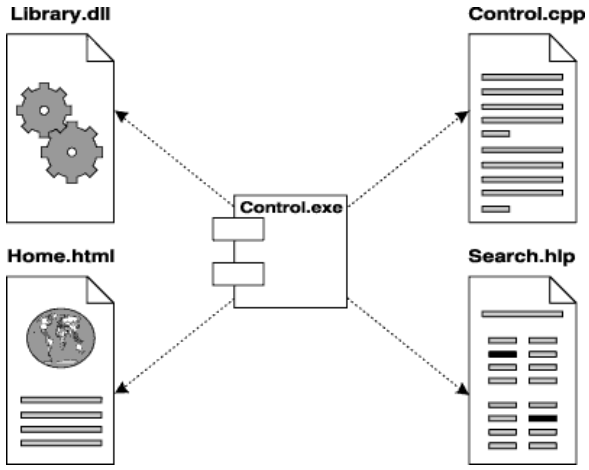


Рисунок 10
Задание № 3. Постройте диаграмму компонентов для выбранной информационной системы (практическая работа № 11).
Задание № 4. Оформите отчет.

Контрольные вопросы

1. Каким образом диаграмма компонентов UML помогает описать архитектуру информационной системы?
2. Какие элементы и связи используются на диаграмме компонентов UML?
3. Какие типы компонентов могут быть представлены на диаграмме компонентов UML?

4. Как диаграмма компонентов UML помогает визуализировать зависимости и взаимодействия между компонентами системы?
5. Какие дополнительные артефакты и свойства могут быть представлены на диаграмме компонентов UML?
6. В чем состоит роль артефактов на диаграмме компонентов UML и как они связаны с компонентами?
7. Каким образом можно представить зависимости и связи между компонентами на диаграмме компонентов UML?
8. Какие инструменты и среды разработки поддерживают создание и визуализацию диаграмм компонентов UML?
9. Какие преимущества и недостатки существуют при использовании диаграмм компонентов UML для описания архитектуры информационной системы?
10. Как включить информацию о интерфейсах и использовании компонентов в диаграмму компонентов UML?

Практическая работа №5
ОБОСНОВАНИЕ ВЫБОРА ТЕХНИЧЕСКИХ СРЕДСТВ

Цель работы:научиться определять необходимые технические средства.

Рабочее задание:

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

Теоретическая часть

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.

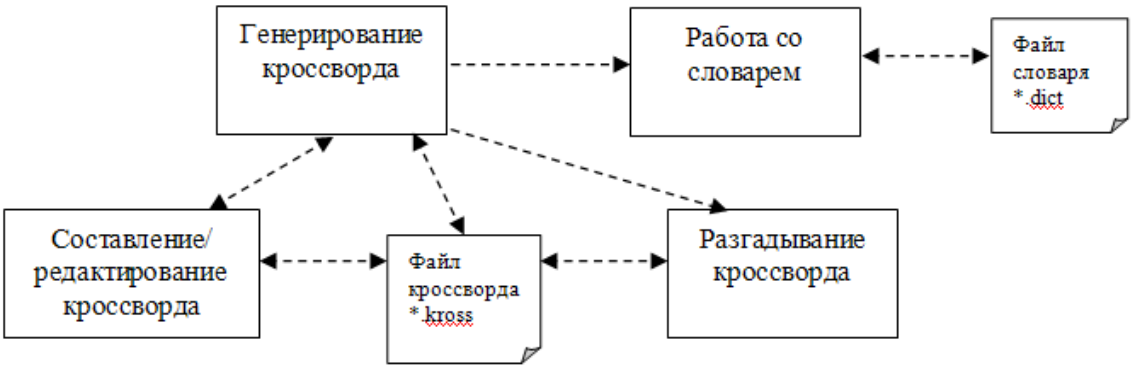


Рисунок 1 – Диаграмма модулей системы (фрагмент)

Таблица 1 – Описание модулей системы (фрагмент)

Название модуля	Название файла	Назначение	Подсистема
Генерирование кроссворда	Generation.c	Отвечает за автоматическое составление кроссворда	Подсистема генерирования кроссворда

		при заданных параметрах	
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.

Порядок выполнения работы

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

Контрольные вопросы

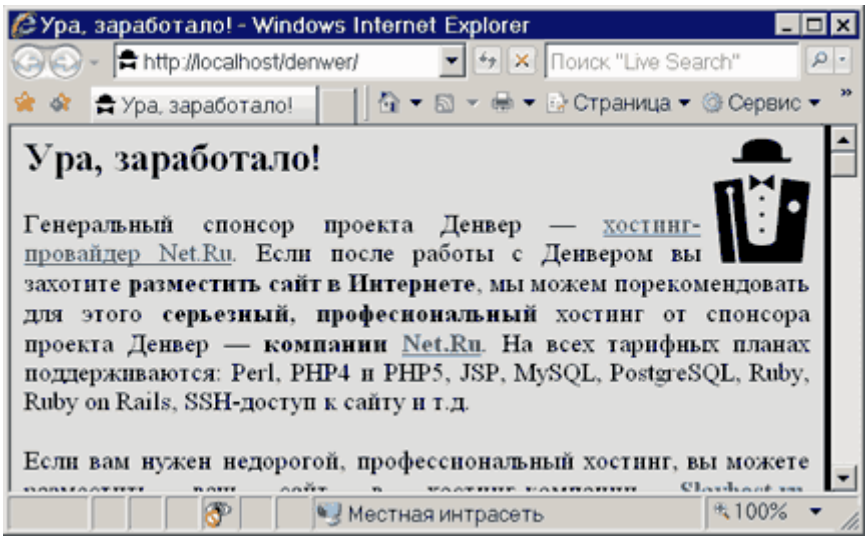
1. Какая роль играет методология обоснования выбора технических средств при создании информационной системы?
2. Какие основные этапы включает процесс обоснования выбора технических средств для информационной системы?
3. Какие критерии следует учитывать при выборе технических средств для информационной системы?
4. Как проводится сравнительный анализ различных технических средств на основе заданных критериев?
5. Каким образом оцениваются затраты на внедрение и поддержку выбранных технических средств?
6. Какие риски могут возникнуть при выборе и использовании определенных технических средств?
7. Как влияют требования к безопасности на выбор технических средств для информационной системы?
8. Какие факторы следует учитывать при оценке совместимости выбранных технических средств с существующей инфраструктурой?
9. Каким образом проводится оценка достижимости заданных целей и требований с использованием выбранных технических средств?
10. Каким образом документируется и обосновывается выбор технических средств для информационной системы?

Практическая работа №6
«Основной синтаксис языка PHP»

Теоретическая часть

Как запустить установленный на компьютере пакет Denwer.

Щелкайте по созданному инсталлятором ярлыку **Start Denwer** на Рабочем столе, а затем, дождавшись, когда все консольные окна исчезнут, открывайте браузер и набирайте в нем адрес: http://localhost/denwer/. Выходить из Интернета при этом не обязательно.



Некоторые ОС имеют обыкновение при первом запуске Internet Explorer-а вызывать **Мастер подключения**. Если это произошло на вашей машине, прикажите горе-мастеру «отвалить» — якобы, вы уже настроили подключение самостоятельно.

Если тестовая страница все же не загрузится, проверьте:

- Отключен ли у вас прокси-сервер в настройках браузера.
- Запущен ли Денвер? Если да, нет ли ошибок при щелчке на пиктограмме пера (справа внизу)?
- Не запущен ли у вас какой-то другой Web-сервер, который мешает Денверу (часто бывает в Windows XP)? Например, Microsoft IIS? Если да, отключите его.

.

Денвер прошел тестирование в следующих ОС:

- Windows 95/98/ME;
- Windows NT/2000/XP/2003;
- Windows Vista, Windows 7.

. **Тестовые проверки**

На странице с адресом **`http://localhost/denwer/`** `имеются ссылки позволяющие проверить работоспособность отдельных компонентов пакета «денвер»`
`Пожалуйста, сделайте такую проверку перед началом работы!!!`

Вид страницы на `http://localhost/denwer/`.

Ура, заработало!

Генеральный спонсор проекта Денвер — хостинг-провайдер Net.Ru. Если после работы с Денвером вы захотите **разместить сайт в Интернете**, мы можем порекомендовать для этого **серьезный, професиональный** хостинг от спонсора проекта Денвер — **компании** Net.Ru. На всех тарифных планах поддерживаются: Perl, PHP4 и PHP5, JSP, MySQL, PostgreSQL, Ruby, Ruby on Rails, SSH-доступ к сайту и т.д.

Поиск по документации в Интернете

В состав Денвера не входит документация к компонентам, т.к. она слишком быстро устаревает. Вместо этого вы можете воспользоваться адаптированной формой поиска. После нажатия на Enter она сама переадресует запросы на необходимые сайты.

PHP
MySQL
Apache
Perl
PostgreSQL

Тестирование Денвера

Настоятельно рекомендуем проверить работоспособность сервера при помощи следующих далее ссылок.

URL	Описание
https://subdomain.localhost/ssl.php	Проверка SSL
http://subdomain.localhost/	Проверка "не-Интернет" доменов второго уровня, а также SSI
http://test1.ru/	Проверка "Интернет"-доменов второго уровня: test1.ru (вначале отключите прокси-сервер!)
http://subdomain.test1.ru/	Проверка "Интернет"-доменов третьего уровня
http://localhost/Tests/phpnotice/index.php	Проверка перехвата PHP Notice в Денвере
http://localhost/Tests/PHP5/index.php5	PHP5 information
http://localhost/Tools/phpMyAdmin	Проверка MySQL и phpMyAdmin
http://custom-host:8648	Проверка хоста с другим IP-адресом и портом (127.0.0.2:8648) *В Windows XP SP2 имеется ошибка, из-за которой данный хост может не работать. Официальную "заплатку" от Microsoft качайте здесь.*
http://localhost/Tests/sendmail/index.php	Проверка отладочной заглушки для sendmail

Цель работы: освоить основные операторы языка сценариев PHP

Лабораторная работа рассчитана на 4 академических часа.

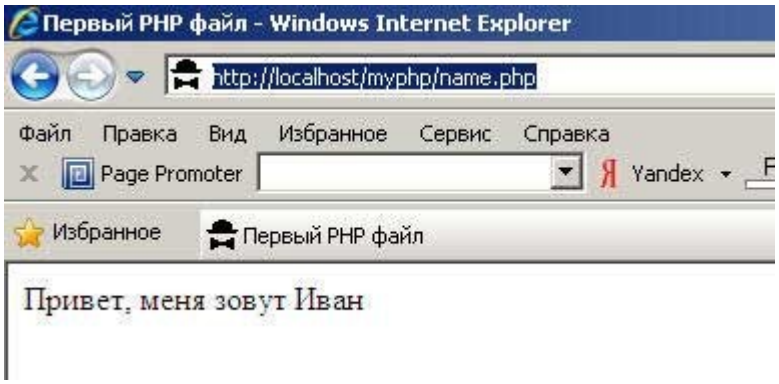
Порядок выполнения работы

Работа с Денвером

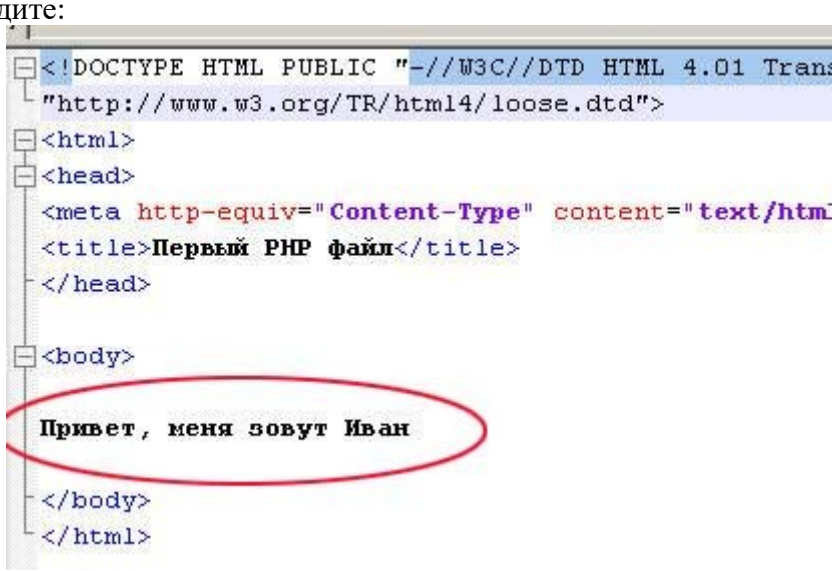
Итак, заходим на диск **Z**, здесь нас интересует папка **home**, открываем эту папку, далее открываем папку **localhost** и открываем следующую папку **www**.
В адресной строке у Вас должен быть вот такой адрес: **Z:\home\localhost\www** и уже **здесь создаем отдельную папку для нового сайта**.
Обращаю Ваше внимание! Все новые папки для разных проектов (читай сайтов) создаются именно в папке www - это важно!
Для определённости открываем папку **www** и создаем здесь для наших тренировок папку **myphp**. **Это и будет наш тестовый сайт на Локальном сервере для тренировки и работы с PHP. Папку myphp создаём до запуска сервера - это тоже важно.**
!!!!(эта папка у вас уже есть!!!!)

Мы уже создали php-файл с названием **name.php**и написали там простенький код. Скопируйте этот файл и положите в папку **myphp**. **!!!!(этот файл там уже есть!!!!)**

Теперь открываем браузер и в адресной строке набираем адрес нового сайта и полный путь до запускаемого файла (с которым Вы работаете в данный момент), в нашем случае этот путь будет выглядеть вот так: **http://localhost/myphp/name.php** и нажимаем **Enter**.
Если Вы все сделали правильно, то на мониторе (в браузере) Вы увидите вот такую картинку (фрагмент):



Поздравляю, Вы научились запускать файлы под Денвером, т.е. работать на Локальном сервере! Таким же образом будут запускаться и тестироваться все последующий php-файлы.
Теперь кликните правой кнопкой мыши и из выпавшего списка выберите пункт**Просмотр HTML-кода**, вот что Вы увидите:



Обращаю Ваше внимание, что Вы видите просто надпись: Привет, меня зовут Иван и никаких php-вставок. Это значит, что установленный на Вашем компьютере Локальный сервер, обработал весь PHP код и выдал уже готовый результат в виде простого html-кода. **Точно так же происходит обработка php-файлов и на реальном сервере в Интернете.**

Как еще можно прописать сайт в Денвере?

Иногда можно встретить рекомендации прописывать новые сайты в Денвере сразу в папке **home**. Здесь механизм выглядит так же, создаём папку **www**, а уже в ней создаём папку для нового сайта.
Но тогда уже при открытии нового сайта под Денвером путь будет выглядеть несколько по другому, вот так: **http://myphp/name.php** , т.е. как и в Интернете.

Рабочее задание

Упражнение 1: Первая программа на PHP

В этом упражнении Вы напишите программу “Hello, world!” на языке PHP

```
<BODY>
<?PHP
echo "Hello, world!";
?>
</BODY>
```

3. Сохраните сценарий в папку сайта
4. Проверьте результат в браузере

При просмотре результата выполнения файла hello.php в браузере
просмотрите html-код (например, в IE меню Вид – Просмотр HTML-кода).

Обратите внимание на то, что php-кода на странице нет – это значит, что php-сценарий был обработан сервером, после чего сервер передал в браузер результат обработанного php-сценария.

Упражнение 2: Простейшие программы на PHP

создайте новый .php файл и выполните предложенные далее задания.

1. Переменной $a необходимо присвоить значение 10, переменной $b присвоить значение 20. Выведите значения переменных на экран.
2. Затем переменной $c присвойте значение суммы этих переменных (переменной $a и переменной $b). Выведите значение переменной $c на экран.
3. Далее увеличьте значение переменной $c в три раза и выведите полученный результат на экран.
4. Разделите переменную $c на разность переменных $b и $a, выведите результат на экран.
5. Введите новые переменные $p и $b. Присвойте переменной $p значение «Программа», а переменной $b значение «работает».
6. Затем сложите переменные, содержащие эти слова («Программа» и «работает»), при этом слова должны быть разделены пробелом (‘ ‘). Результат необходимо присвоить переменной $result.
7. Далее с помощью оператора «.=» необходимо к строке «Программа работает» добавит слово «хорошо». Результат необходимо присвоить переменной $result.
8. Есть две переменные: $q = 5 и $w =7. Создайте скрипт, в результате выполнения которого эти две переменные «обмениваются» значениями – 10

переменная $q получает значение 7, переменная $w получает значение 5, при этом не создавая новых переменных (вариант $q = 7 и $w = 5 не рассматривается).

Упражнение 3: Простейшие операции

□□□Средствами PHP вывести строку “Hello Ваша Фамилия!”. Причём строка должна быть выведена наклонным и жирным шрифтом. Теги <b> и <i> указать **вне** PHP кода.

2. Средствами PHP вывести строку “Hello Ваша Фамилия!”. Причём строка должна быть выведена наклонным и жирным шрифтом. Теги <b> и <i> указать **с использованием** PHP кода.

3. Создать таблицу, каркас для которой дан ниже

```
<table border="3">
<tr>
<td>

_
</td>
<td>

_
</td>
</tr>
<tr>
<td>

_
</td>
<td>

_
</td>
</tr>
</table>
```

В каждой ячейке вывести средствами PHP Вашу фамилию и имя, на русском языке, при чём стиль либо цвет текста в каждой ячейке должен быть разным.

4. Закомментировать созданный выше код, используя различные типы комментариев.

```
echo "Это тест"; // Это однострочный комментарий в стиле c++
/* Это многострочный комментарий
еще одна строка комментария
*/
echo "Последний тест"; # Это комментарий в стиле оболочки Unix
```

5. Вызвать функцию phpinfo().

Упражнение 4. Использование основных операторов

приведение типов

1. Создать переменную $var и последовательно присвоить ей значения имеющие следующие типы:

• boolean (логический)

```
$var = false;
echo gettype($var); // функция gettype определяет тип переменной
```

• integer (целый)

```
$var = 11;
echo gettype($var);
```

• float (вещественный с плавающей точкой)

```
$var = 11.2;
echo gettype($var);
```

• string (строковый)

```
$var = “String value”;
echo gettype($var);
```

2. Использование функций приведения типов. Запустить и прокомментировать результат выполнения следующих функций:

• intval() (преобразование аргумента к целому типу)

o echo intval(42) ." <hr>";
o echo intval(4.2) ." <hr>";
o echo intval('42') ." <hr>";
o echo intval('+42') ." <hr>";
o echo intval('-42') ." <hr>";
o echo intval(042) ." <hr>";
o echo intval('042') ." <hr>";

• floatval() (преобразование аргумента к типу «с плавающей точкой»)

o echo floatval('122') ." <hr>";
o echo floatval('The') ." <hr>";
o echo floatval('122.34343') ." <hr>";
o echo floatval('122.34343The') ." <hr>";
o echo floatval('The122.34343') ." <hr>";
o echo floatval('122.The34343') ." <hr>";

• strval() (преобразование аргумента к строковому типу (string))

o echo strval(122) ." <hr>";
o echo strval(122.01) ." <hr>";
o echo strval(0x122) ." <hr>";
o echo strval(0122) ." <hr>";

• settype() (установка аргумента заданного типа)

```
$foo = "5bar"; // string
$bar = true; // boolean
settype($foo, "integer"); // $foo is now 5 (integer)
settype($bar, "string"); // $bar is now "1" (string)
```

Операции с логическими переменными

1. Запустить и прокомментировать следующий код

```
if(TRUE){
echo "TRUE is true<hr />";
}else{
echo "TRUE is flase<hr />";
}
```

```
if(1){
echo "1 is true<hr>";
}else{
echo "1 is flase<hr>";
}

if(0.0){
echo "0.0 is true<hr>";
}else{
echo "0.0 is flase<hr>";
}

if(""){
echo "\"\" is true<hr>";
}else{
echo "\"\" is flase<hr>";
}
```

Операции с целыми числами

1. Выполните следующие операции с целыми числами

```
$a = 3;
$b = 2;
echo '<br>$a='.$a;
echo '<br>$b='.$b;
$c = $a+$b;
echo '<br>$a+$b='.$c;
$c = $a-$b;
echo '<br>$a-$b='.$c;
$c = $a*$b;
echo '<br>$a*$b='.$c;
$c = $a/$b;
echo '<br>$a*$b='.$c;
```

Операции со строками

1. Запустить и прокомментировать результат выполнения

```
$expand = “EXPAND”;
$either = “EITHER”;
echo 'это простая строка';
echo 'Также вы можете вставлять в строки
символ новой строки таким образом,
поскольку это нормально';
echo 'Однажды Арнольд сказал: "I\'ll be back"';
echo 'Вы удалили C:\\*.*?';
echo 'Вы удалили C:\*.*?';
echo 'Это не вставит: \n новую строку';
echo 'Переменные $expand также $either не подставляются';
echo “Переменные $expand также $either подставляются. Почему?”;
echo $expand + $either;
```

Упражнение 5: Условные операторы

Приоритет операторов и управление им

1. Вывести результат выполнения следующих выражений:

1 + 5 * 3
(1 + 5) * 3
(12 + 13) – (4 + 6)
12 + 13 – 4 + 6

Пример желаемого вывода:
1 + 5 * 3 = 16

Операторы

1) if

а) Вначале задать значение переменной $name. Написать проверку её значения. Если в $name содержит имя “John” вывести сообщение «Привет Джон!»

б) Добавить переменную $num, присвоить ей значение 5.
Сделать следующую проверку: если переменная $name содержит имя “John” и переменная $num содержит 5 вывести сообщение: «всё правильно! Джону 5 лет»

в) Доработать предыдущее задание. Сделать следующую проверку: если хотя бы в одной из переменных ($num или $name) содержится имя “John” вывести сообщение: «Мы нашли Джона!!!»

2) **else**
а) Доработать предыдущее задание, так чтобы в случае, если в $name содержится не «John» вывести “Приветствую Незнакомец”.

б) Доработать предыдущее задание так чтобы в случае, если $name не содержит “John” или $num не содержит 5 вывести сообщение «Возможно ты не Джон! Или тебе не 5 лет!!»

в) Доработать предыдущее задание так чтобы в случае, если ни одна из имеющихся переменных не содержит значений “John”и «5» вывести сообщение «Я знаю! Ты точно не Джон и тебе не 5 лет!»

3) **elseif**
а)Завести массив $names, содержащий элементы “John”, “Bill”, “Sam”.
Выбрать один из элементов массива и записать его значение в переменную $name. Произвести проверку $name на имена “John”, “Bill”, “Sam”. Также, в случае отрицательного результата вывести “Приветствую Незнакомец”.

б)Проверить каждый элемент массива $names на имена “John”, “Bill”, “Sam”. В каждом случае выдать сообщение, содержащее информацию о том кто на каком месте находится в списке. Например: «Джон на первом месте, Билл на втором месте, Сэм на третьем месте»

Упражнение 6: «Циклы»

1. с помощью цикла **while** распечатать 10 раз строку «Здравствуй мир!»

2. Вывести последовательно числа от 1990 до 2007 используя цикл **while**.

3. Вывести все нечетные числа от 19 до 200 используя цикл **while**. Числа на странице должны располагаться в столбик.

4. Найти сумму первых 100 чисел с помощью цикла **do-while**

5. Вывести последовательно в строке числа от 0 до 18 и в другой строке числа последовательно от 18 до 0. Использовать цикл **for**

Справочная информация

while

```
$i=0;
whie($i<3){
$i++;
echo $i.”<br />”;
}
```

- **do-while**

```
$i = 5;
do {
echo $i.”<br />”;
$i--;
} while ($i > 0);
```

- **for**

```
for($i=0;$i<=5;$i++){
echo $i.”<br />”;
}
```

Упражнение 7: «Массивы»

1. Создать массив **$arr**, состоящий из 4-х произвольных элементов двумя способами:
а)перечислением элементов в квадратных скобках;
б)указывая в квадратных скобках элементы и их ключи.
Затем вывести содержимое массива вместе с ключами.

2. Создать массив **$arr**, состоящий из 4-х целых чисел. Произвести следующие действия:
добавить в массив два других целых числа, строку «Иван» и вещественное число 1,25
Распечатать полученный массив вместе с ключами.

3. Создать массив **$oll**, таким образом, что 0-й элемент равен строке «Привет», а 4-ый элемент равен строке «Мир!». Добавить в массив 1-ый, 2-ой и 3-ий элементы (любые).
Распечатать 2-ой и 3-ий элементы полученного массива.

4. В массиве из предыдущего задания удалить 1-ый, 2-ой и 3-ий элементы. Распечатать полученный массив двумя способами – а) вывести только содержимое массива;
б) вывести содержимое массива вместе с ключами.

5. Переиндексировать массив, из предыдущего задания. Затем распечатать полученный массив двумя способами – а) вывести только содержимое массива;
б) вывести содержимое массива вместе с ключами.

Упражнение 8: «условия, циклы, массивы»

1. С помощью цикла **for** распечатать на странице 10 раз надпись «**Hello world!!!**» следующими способами:
а) друг за другом в одной строке через пробел;
б) по одному разу в каждой строке;
в) по два раза в каждой строке.

2. С помощью цикла **for** и инструкции **if** распечатать на странице 10 раз надпись «**Hello world**!!!» следующим образом:
а) через строку (в одной строке печатаем, следующую пропускаем итд);
б)в каждой третьей строке;
в) в четной строке по два раза, в нечетной строке по одному разу.

3. С помощью цикла **for** и инструкции **if-else** распечатать на странице 10 раз надпись «**Hello world!!!**» следующим образом:
а) в четных строках жирным шрифтом, в нечетных строках курсивом
б) в четных строках красным цветом, в нечетных строках зеленым цветом.

4.Создать переменную **$day**, присвоить ей некоторое значение (от 1 до 7). С помощью инструкции **switch** распечатать переменную **$day** и название дня недели которое соответствует значению переменной **$day**.

5. Создать массив **$arr**, заполнить его строками, содержащими названия месяцев. С помощью цикла **while**, инструкции **if** и функции **echo** распечатать содержимое массива следующим образом: название каждого четного месяца распечатать жирным шрифтом, название нечетных месяцев распечатать курсивом.

Контрольные вопросы

1. Как объявить переменную в PHP и какой символ используется для обозначения строк?
2. Каким образом можно выводить текст и значения переменных на экран в PHP?
3. Какая конструкция используется для выполнения условных операций в PHP?
4. Как объявить и использовать массив в PHP?
5. Каким образом осуществляется циклическое выполнение определенного блока кода в PHP?
6. Как объявить и вызвать функцию в PHP?

Практическая работа №7
(функции для работы со строками)

Цель работы:
Научиться работать со строками в php

Теоретическая часть
Для того чтобы определить, входит ли данная подстрока в состав строки, используется функция strpos(). Синтаксис strpos() такой:

```
strpos (исходная строка,строка для поиска
[,с какого символа искать])
```

Она возвращает позицию появления искомой строки в исходной строке или возвращает логическое false, если вхождение не найдено. Дополнительный аргумент позволяет задавать символ, начиная с которого будет производиться поиск. Кроме логического false эта функция может возвращать и другие значения, которые приводятся к false (например, 0 или ""). Поэтому для того, чтобы проверить, найдена ли искомая строка, рекомендуют использовать оператор эквивалентности «===».

```
<?
$str = "Идея наносить данные на перфокарты
и затем считывать и обрабатывать их
автоматически принадлежала Джону Биллингсу,
а ее техническое решение осуществил Герман
Холлерит. Перфокарта Холлерита оказалась
настолько удачной, что без малейших изменений
просуществовала до наших дней.";
$pos = strpos($str,"Холлерит");
if ($pos !== false) echo "Искомая строка
встречена в позиции номер $pos ";
else echo "Искомая строка не найдена";
/* заметим, что мы проверяем значение
$pos на эквивалентность с false.
Иначе строка, находящаяся в первой позиции,
не была бы найдена, так как 0
интерпретируется как false. */
?>
```

Выделение подстроки
Функция strstr
Говоря о выделении подстроки из искомой строки в языке PHP, в первую очередь стоит отметить функцию strstr():

```
strstr (исходная строка, строка для поиска)
```

Она находит первое появление искомой строки и возвращает подстроку, начиная с этой искомой строки до конца исходной строки.
Если строка для поиска не найдена, то функция вернет false. Если строка для поиска не принадлежит строковому типу данных, то она переводится в целое число и рассматривается как код символа. Кроме того, эта функция чувствительна к регистру, т.е. если мы будем параллельно искать вхождения слов «Идея» и «идея», то результаты будут разными. Вместо strstr() можно использовать абсолютно идентичную ей функцию strchr().
Пример 8.4. Выделим из строки, содержащей название и автора исследования, подстроку, начинающуюся со слова «Название»:

```
<?
$str = "Автор: Иванов Иван (<a href=mailto: van@mail.ru>написать письмо</a>),
Название: 'Исследование языков
программирования' ";
echo "<b>Исходная строка: </b>",$str;
if (!strstr($str, "Название"))
echo "Строка не найдена<br>";
else echo "<p><b>Полученная подстрока: </b>",
```

```
strstr($str, "Название");
?>
```

Пример 8.4. Использование функции strstr()
В результате получим:

```
Исходная строка: Автор: Иванов Иван
(написать письмо),
Название: 'Исследование языков
программирования'
Полученная подстрока: Название:
'Исследование языков программирования'
```

Для реализации регистронезависимого поиска подстроки существует соответствующий аналог этой функции – функция stristr (исходная строка, искомая строка). Действует и используется она точно так же, как и strstr(), за исключением того, что регистр, в котором записаны символы искомой строки, не играет роли при поиске.

Удаление тегов
На самом деле решить такую задачу можно гораздо проще, с помощью функции strip_tags:

```
strip_tags (строка [, допустимые теги])
```

Эта функция возвращает строку, из которой удалены все html и php-теги. С помощью дополнительного аргумента можно задать теги, которые не будут удалены из строки. Список из нескольких тегов вводится без каких-либо знаков разделителей. Функция выдает предупреждение, если встречает неправильные или неполные теги.

```
<?php
$string = "<b>Bold text</b>
<i>Italic text</i>";
$str = strip_tags($string);
// удаляем все теги из строки
$str1 = strip_tags($string, '<i>');
// удаляем все теги кроме тега <i>
$str2 = strip_tags($string, '<i><b>');
// удаляем все теги кроме тегов <i> и <b>
echo $str,"<br>",$str1,"<br>", $str2;
?>
```

Пример 8.6. Использование функции strip_tags()
В результате получим:

```
Bold text Italic text
Bold text Italic text
Bold text Italic text
```

Если нам нужно получить один конкретный символ из строки, зная его порядковый номер, то не следует задействовать функции типа substr. Можно воспользоваться более простым синтаксисом – записывая номер символа в фигурных скобках после имени строковой переменной. В контексте предыдущего примера букву «р», расположенную второй по счету, можно получить так:

```
echo $text{1}; // выведет символ "p"
```

3. Определение длины строки

Функция **strlen** используется очень часто, поскольку значение длины строки требуется для многих ее преобразований. Использовать ее просто – передайте ей строку, а функция вернет количество символов, например:

```
$str="Hello";
echo strlen($str);//5
```

Замена вхождения подстроки

Функция str_replace

Для замены вхождения подстроки можно использовать функцию str_replace(). Это простая и удобная функция, позволяющая решать множество задач, не требующих особых тонкостей при выборе заменяемой подстроки. Для того чтобы производить замены с более сложными условиями, используют механизм регулярных выражений и соответствующие функции ereg_replace() и preg_replace(). Синтаксис функции str_replace() такой:

```
str_replace(искомое значение,
значение для замены, объект)
```

Функция str_replace() ищет в рассматриваемом объекте значение и заменяет его значением, предназначенным длязамены

explode()

Функция explode() делит исходную строку на подстроки, каждая из которых отделена от соседней с помощью указанного разделителя, и возвращает массив полученных строк.

При вызове **explode()** мы должны указать два параметра: разделитель и исходная строка.

```
$A= explode(";",$str)
```

$A-массив, и обращаться с ним как с массивом!

Функция count( )

Возвращает количество элементов в массиве. Например:

```
<?php
$mass=array(1,2,6,8,4,9);
echo count($mass);
?>
```

Пример 9. Просмотр ассоциативного массива

```
<?php
$names["Иванов"] = "Андрей";
$names["Петров"] = "Борис";
$names["Волков"] = "Сергей";
$names["Макаров"] = "Федор";
foreach ($names as $key => $value) {
echo "<b>$value $key</b><br>";
}
?>
```

Рабочее задание:

1. Пусть задана строка, содержащая фамилию, имя и отчество пользователя через пробел. Например: **Иванов Иван Иванович**.

Проанализировать содержимое введенной строки. Если строка содержит слово «**Иван**», то выдать сообщение**: Здравствуйте, Иван! Мы вас нашли!**

В противном случае (если слова «**Иван**» в строке нет) выдать стандартное приветствие: «**Здравствуйте фамилия имя отчество**»

Подсказка: воспользоваться функцией ***strpos (исходная строка, искомая строка)***

2. Пусть задан текст следующего содержания: **Иванов Иван Иванович это автор монографии, имеющей название: «Исследования плоскофигуральнолинейных гладкогиперболических псевдоструктур »**
Выделить из введенной строки название монографии и распечатать его.

```
Подсказка: воспользоваться функцией strstr (исходная строка, строка начиная с которой формируется результат)
```

3. Пусть задана строка, содержащая названия любимых цветов пользователя, например: «**Я люблю красные гвоздики, красные маки и зеленые розы**»
Заменить в строке все слова «**красные**» на «**вкусные**» и распечатать полученную строку.

```
Подсказка: воспользоваться функцией str_replace(искомое значение,
значение для замены, исходная строка)
```

4. Пусть задана строка, содержащая фамилию, имя и отчество пользователя через пробел. Например: **Иванов Иван Иванович**.
Распечатать приветствие следующего вида:
Здравствуйте уважаемый клиент!
Ваше имя – Иван. Оно содержит 4 буквы.
Ваша фамилия – Иванов. Она содержит 6 букв.
Ваше отчество – Иванович. Оно содержит 8 букв.
Подсказка: воспользоваться функцией explode("разделитель", “исходная строка”) которая разделяет исходную строку на элементы и функцией count( ) для подсчета числа букв в слове.

5. Пусть заданы два произвольных слова, разделенных пробелами. Например: «**красные розы**»
Распечатать сообщение следующего вида: **слово «красные» содержит на 3 буквы больше чем слово «розы»**
Подсказка: воспользоваться функцией explode("разделитель", “исходная строка”) которая разделяет исходную строку на элементы и функцией count( ) для подсчета числа букв в слове.

Контрольные вопросы

1. Как объединить две строки в PHP, используя строковые функции?
2. Как определить длину строки в PHP?
3. Как сделать все символы строки в верхнем регистре с помощью строковых функций в PHP?
4. Как найти позицию первого вхождения подстроки в строку в PHP?
5. Как заменить все вхождения определенного символа или подстроки в строке с помощью строковых функций в PHP?
6. Как проверить, содержит ли строка определенную подстроку в PHP?
7. Как разбить строку на массив подстрок в PHP с использованием строковых функций?
8. Как удалить пробелы в начале и конце строки в PHP с помощью строковых функций?
9. Как преобразовать строку в число в PHP с помощью строковых функций?
10. Как преобразовать регистр символов в строке из нижнего в верхний или наоборот с помощью строковых функций в PHP?

Практическая работа №8
Загрузка файлов на сервер в PHP

Цель работы:

Научиться загружать файлы на удаленный сервер средствами php

Теоретическая часть

Настройки в php.ini (WebServers\usr\local\php5)

Первая настройка называется file_uploads(on|off). Если она в off, то загрузка файлов не пройдет. Вторая настройка – это upload_tmp_dir. Это временная папка, куда загружаются файлы. И третья настройка называется upload_max_filesize(2Mb). Это максимальный размер загружаемого файла. По умолчанию равен два мегабайта.

Здесь есть один очень важный момент: у PHP есть такая настройка, как post_max_size и равна она 8 мегабайтам. Она означает, что объем посылаемых данных через форму не должен их превышать. Следовательно, объем upload_max_filesize не должен превышать или быть равен значению post_max_size. В противном случае, загрузка файлов на сервер в PHP не состоится.

Итак, как же происходит загрузка файла на сервер?
Конечно с помощью формы отправки данных. Однако, эта форма не совсем простая. В чем же ее отличие от обычной формы? Давайте вспомним, какие атрибуты имеет форма?
Во-первых атрибут action. Во-вторых, атрибут method.
Эти атрибуты имеет простая форма отправки данных. А вот форма, которая отправляет файлы на сервер имеет все те же атрибуты, что и обычная форма плюс третий атрибут, который называется*enctype*. По умолчанию этот атрибут равен «multipart/form-data».
Еще один важный момент – все файлы, которые отправляются на сервер, отправляются только методом POST.
Далее, у нас в форме идут текстовые поля. Вопрос – какой тип должно иметь текстовое поля для отправки файлов в PHP? Конечно – это тип «file»:

```
<?php
<form action="<?=$_SERVER['PHP_SELF']?>"
method="post" enctype="multipart/form-data">
<input type="file" name="file_name">
<input type="submit" value="Отправить файл">
</form>
?>
```

(**здесь скрипт обработчик находится на той же странице, что и форма отправки файлов**)
Либо так:

```
<?php
<form action="<адрес скрипта обработчика>" method="post"
enctype="multipart/form-data">
<input type="file" name="file_name">
<input type="submit" value="Отправить файл">
</form>
?>
```

Это стандартная форма отправки файлов. Смотрите, у нас, как мы уже говорили, в uploads_max_filesize объем передаваемых файлов стоит по умолчанию 2Mb. Допустим, мы хотим, чтобы этот объем был меньше, например 100Kb. Для чего это нужно? Например, для загрузки маленьких аватарок на форум. Мы можем сделать следующее: в форме отправки файлов поставить скрытое поле типа hidden, с именем MAX_FILE_SIZE и со значением value нужного нам размера:

```
<?php
<form action="<?=$_SERVER['PHP_SELF']?>" method="post" enctype="multipart/form-
data">
<input type="hidden" name="MAX_FILE_SIZE" value="0.1"
<input type="file" name="file_name">
<input type="submit" value="Отправить файл">
</form>
?>
```

(**здесь скрипт обработчик находится на той же странице, что и форма отправки файлов**)
Либо так:

```
<?php
<form action="кудапошлете" method="post"
enctype="multipart/form-data">
<input type="hidden" name="MAX_FILE_SIZE" value="0.1"
<input type="file" name="file_name">
<input type="submit" value="Отправить файл">
</form>
?>
```

.
Идем дальше. Когда пользователь загружает файл на сервер в PHP, то этот файл сначала попадает во временную директорию upload_tmp_dir.
Обратите внимание вот на что – два файла с одинаковым именем в папке upload_tmp_dir лежать не могут. Для этого они временно переименовываются.

Вся информация, которая приходит к нам, она попадает в массив **$_FILES[]**. Этот массив двумерный. Первая ячейка массива – это имя поля, через которое нам послали файл. В этой ячейке, в свою очередь, тоже находится массив, состоящий из пяти ячеек (реальное имя файла, временное имя файла, размер файла, тип файла, ошибка):

```
<?php
$_FILES['name_file']['name'];
$_FILES['name_file']['tmp_name'];
$_FILES['name_file']['size'];
$_FILES['name_file']['type'];
$_FILES['name_file']['error'];
?>
```

Посмотреть, какая информация приходит нам при отправке файла, можно так:

```

<?php
if($_SERVER['REQUEST_METHOD'] == "POST"){
echo "<pre>";
print_r($_FILES);
echo "</pre>";
}
?>
<form action="<?=$_SERVER['PHP_SELF']?>" method="post"
enctype="multipart/form-data">
<input type="hidden" name="MAX_FILE_SIZE" value="0.1"
<input type="file" name="file_name">
<input type="submit" value="Отправитьфайл">
</form>

```

Теперь все, что нам нужно сделать, это забрать файл, если он нас устраивает, из временной папки upload_tmp_dir. Делается это с помощью функции **move_uploaded_file($tmp, name)**. В первом параметре этой функции мы указываем – откуда мы забираем файл, а во втором параметре, соответственно, куда мы его забираем:

Пример

```


<?php
/*проверяем, если файл загружен и ошибок нет*/
if($_FILES['file_name']['error'] == 0){
/*выбираем путь временного хранилища файла*/
$temp = $_FILES['name_file']['tmp_name'];
/*выбираем путь, куда будем сохранять файл*/
$name_file = $_FILES['name_file']['name'];
/*перемещаем файл из временной папки к нам на сервер*/
move_uploaded_file($temp, $name_file);
}
?>
<form action="<?=$_SERVER['PHP_SELF']?>" method="post"
enctype="multipart/form-data">
<input type="file" name="name_file">
<input type="submit" value="Отправитьфайл">
</form>


```

Если хотите, чтобы файл загружался на сервер в PHP в определенную папку, тогда необходимо эту папку создать и в строке:

```
<?php
move_uploaded_file($temp, $name_file);
?>
```

Подставить путь к этой папке:

```
1 <?php
2 move_uploaded_file($temp, "uploads/".$name_file);
3 ?>
```

uploads – это название папки. Вы можете придумать любое свое название и указать к ней путь. Так как в массиве $_FILES указывается тип файла, можно создать несколько папок под каждый тип и при проверке файла на тип, загружать его в определенную папку.

Готовый пример

Форма отправки файлов – файл zagruz.php

```
<?php header('Content-Type: text/html; charset= utf-8')?>
<!DOCTYPE HTML>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Загрузка и удаление файлов</title>

<style type="text/css">

</style>

</head>
<body>
<div id="page">
<center>
<br><br><br><br>
 <p style='font-size:25px;  '>Хотите загрузить файл? </p>
<br>
<br>
 <p>Если хотите загрузить файл в основную папку - в поле ваша папка введите no</p>
 <br>
<form name="fila" action="upload.php" method="post" enctype="multipart/form-data">
Ваша папка<input type="text"  required name="name" id="name" />
<input type="file" name="file" id="file" required />

<input type="submit" name="go" id="go" value="Загрузить" />
</form>
 <br><br>

 <br><br><br><br> <br>
<p style='font-size:25px;  '>Хотите удалить файл? </p>

<form name='formdel' action='filedel.php' method='post'>

<p>Ваша папка<br><input type="text"  required name="namefold"  /></p>
<p>Имя файла<br /><input type='text' name='namefil' required  ></p>
 <p><input type='submit'value="Удалить"> </p>
</form>

<div id="footer">
<div class="top"></div>
<div id="bottom_menu">      </div>
<div id="bottom_addr"></div>
</div>
</div>
</center>
</body>
</html>
```

Скрипт для загрузки файлов - upload.php

```
<?php
  $error = $_FILES['file']['error'];
  switch($error) {
    case 0 :
      $error = 'нет';
      break;
    case 1 : case 2 :
      $error = 'слишком большой файл';
      break;
    case 3 :
      $error = 'файл загружен частично';
      break;
    case 4 :
      $error = 'файл не был загружен';
  }
?>

Ваша папка: <?=$_POST['name']; ?><br />
Имя файла: <?=$_FILES['file']['name']; ?><br />
Mime type: <?=$_FILES['file']['type']; ?><br />
Ошибки: <?=$error; ?><br />
Размер файла: <?=$_FILES['file']['size']; ?> байт <br />

<?
$namefold=trim($_POST['name']);
if( $namefold != "no" ){
if( is_dir($namefold) === false )
{
   mkdir($namefold);
}
}

// Проверяем загружен ли файл
   if(is_uploaded_file($_FILES["file"]["tmp_name"]))
   {
      //Если файл загружен успешно, перемещаем его
      //из временной директории в конечную
            if( $namefold != "no" ){
     $uploaddir = $namefold."/";
            $uploadfile = $uploaddir . basename($_FILES['file']['name']);
            } else {
            $uploadfile =  basename($_FILES['file']['name']);
            }
     move_uploaded_file($_FILES["file"]["tmp_name"],$uploadfile);
   } else {
      echo("Ошибка загрузки файла");
  }
?>
<a href=zagruz.php >Вернуться назад</a><br>
<? echo '<META HTTP-EQUIV=Refresh Content="5;URL=zagruz.php">';?>
```

Скрипт для удаления файлов - filedel.php

```
<? $namefold=trim($_POST['namefold']); ?>
<? $namefil=trim($_POST['namefil']); ?>
<?
if( $namefold != "no" ){
$filepath =$namefold."/".$namefil;
} else {
$filepath =$namefil;
}
```

```
unlink($filepath);
echo "Файл был удалён!";

?>
<a href=zagruz.php >Вернуться назад</a><br>
<? echo '<META HTTP-EQUIV=Refresh Content="2;URL=zagruz.php">';?>
```

Рабочее задание

1. Загрузить на сервер в свою папку файлы (**zagruz.php , upload.php , filedel.php**)

2. Открыть файл **zagruz.php,** в поле формы «ваша папка» ввести какое нибудь название папки (только латиница, без пробелов). В поле «выберите файл» выбрать какой нибудь файл для загрузки. Нажать кнопку «загрузить». Посмотреть содержимое вашей основной директории. Внутри должна появиться папка с именем которое вы выбрали, а внутри созданной папки – загруженный вами файл.

3. Открыть файл **zagruz.php,** в поле формы «ваша папка» ввести строку – no . В поле «выберите файл» выбрать какой нибудь файл для загрузки. Нажать кнопку «загрузить». Посмотреть содержимое вашей основной директории. Внутри основной директории должен появиться загруженный вами файл.

Контрольные вопросы

1. Каким образом можно загружать файлы на удаленный сервер с помощью PHP?
2. Какими функциями PHP можно использовать для загрузки файлов на удаленный сервер?
3. Каким образом можно ограничить типы файлов, которые могут быть загружены на удаленный сервер с помощью PHP?
4. Как определить размер загружаемого файла перед его сохранением на удаленном сервере с помощью PHP?
5. Каким образом можно получить информацию о загружаемом файле, такую как имя, тип и временное имя файла в PHP?
6. Как обработать ошибки, которые могут возникнуть при загрузке файла на удаленный сервер с помощью PHP?
7. Каким образом можно сохранять загруженные файлы на удаленном сервере в определенной папке с помощью PHP?
8. Как обеспечить безопасную загрузку файлов на удаленный сервер с помощью PHP, чтобы предотвратить возможные уязвимости?
9. Каким образом можно проверять наличие и уникальность загружаемого файла на удаленном сервере с помощью PHP?
10. Каким образом можно реализовать прогресс-бар или индикатор загрузки при загрузке файла на удаленный сервер с помощью PHP?

Практическая работа №9
Работа с архивными файлами на сервере средствами php

Цель работы:

Научиться создавать и раскрывать архивные папки на удаленном сервере средствами php

Теоретическая часть

Часто бывает необходимо загрузить на хостинг папку с файлами. Если у пользователя нет доступа к панели управления хостингом, и отсутствуют пароли для передачи данных по ftp, то задача становится трудно выполнимой, поскольку файлы в таком случае придется загружать по одному.
Выходом в данной ситуации является возможность загрузки на сервер архивных папок и последующее извлечение их из архива непосредственно на хостинге средствами php.
Рассмотрим алгоритм работы с архивными файлами средствами php.

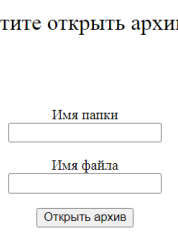
Итак, для работы с архивными файлами существует множество различных специальных библиотек. Мы воспользуемся библиотекой - **pclzip.lib.php**
(файл библиотеки находится в папке **extractarhive**)
Как пользоваться данной библиотекой для раскрытия архивных файлов?
Пример

```
<?
require_once('pclzip.lib.php');//подключаем библиотеку
$archive = new PclZip($filepath);// $filepath – переменная в которой находится путь к файлу
if ($archive->extract() == 0) {
die("Error : ".$archive->errorInfo(true));
}
?>
```

Таким образом, сначала нам нужно подключить библиотеку **pclzip.lib.php** с помощью функции **require_once**, а затем передать объекту путь **PclZip** к архивному файлу **$filepath**.
Файл с библиотекой **pclzip.lib.php** должен находиться в той же папке, что и срипт, приведенный выше.
Извлечение из архива производится командой **$archive->extract()**.
Название папки в которую будут помещены файлы из архива будет совпадать с названием самого архивного файла. Папка с извлеченными файлами будет создана в текущей директории (там где лежат скрипт примера и библиотека).
Важно отметить, что библиотека **pclzip.lib.php** умеет распаковывать только **zip**-архивы. Поэтому не стоит пытаться «подсунуть» ей **rar**, **7z** или что-нибудь еще, кроме **zip**.
В связи со всем вышесказанным… Для того, чтобы сделать достаточно полноценное приложение по распаковке архивов на сервере, приведенного выше кода явно не достаточно. Перед тем как раскрывать архив очень неплохо бы убедиться, что файл, который мы распаковываем существует, что файл этот имеет расширение **zip**, а не какое-нибудь другое. Так же желательно уметь извлекать архивы не только из текущей директории, но и из внутренних папок. При этом желательно проверить эти самые папки на существование.
Итак, начинаем делать приложение.
1. Создадим html-форму с текстовыми полями для передачи имени папки и архивного файла

```
<form name='formarhiv' action="<?=$_SERVER['PHP_SELF']?>" method='post'>
<p>Имя папки<br><input type="text"   name="namefold"  /></p>
<p>Имя файла<br /><input type='text' name='namefil' required  ></p>
 <p><input type='submit'value="Открыть архив"> </p>
</form>
```

Здесь в параметре **action** передается **<?=$_SERVER['PHP_SELF']?>**
Это означает, что php-скрипт будет находиться в том же файле, что и html-форма отправки данных.
Выглядеть в браузере это будет вот так



Работать с данной формой будем так: Если наш архив находится в текущей папке, то имя папки в поле мы не пишем (оставляем поле пустым), а если архив находится в какой нибудь внутренней папке, то в поле «имя папки» записываем имя данной папки.

2. Выведем на экран содержимое текущей папки на сервере (куда мы загрузили все наши скрипты по работе с архивами)

```
<?
echo " <br>сейчас в текущей папке   лежат следующие файлы:<br><br>";
$dir = opendir('.');

//List files in images directory
while (($file = readdir($dir)) !== false)
  {

  echo  $file . "<br />";
```

```
  }
 closedir($dir);
?>
```

Функция **opendir** возвращает дескриптор каталога для последующего использования с функциями **closedir()**, **readdir()**
Функция **readdir** -- получает элементы каталога(все что лежит в папке) по его дескриптору.

3. Получим из html-формы и запишем в переменные $namefil и $namefold – имя архивного файла и папки в которой он находится

```
<?
$namefold=trim($_POST['namefold']);
$namefil=trim($_POST['namefil']);
?>
```

4. Если архивный файл находится в какой то внутренней папке(то есть переменная $namefold не пуста), то проверим, существует ли эта папка. Если такой папки нет, то остановим скрипт командой exit(). Если папка найдена, то путь к архивному файлу запишем в переменную $filepath. Если архивный файл находится в текущей папке, то путь нему – есть имя этого файла.

```
<?
if( !empty($namefold)){
if( is_dir($namefold) === false )
{
    echo "<br>ТАКОЙ ПАПКИ ".$namefold." НЕ СУЩЕТВУЕТ!!!<br>";
    exit();
}
else
{
$filepath =$namefold."/".$namefil;
}
}
else
{
$filepath =$namefil;
}
?>
```

5. Проверим, существует ли файл по тому пути, который мы записали в переменную $filepath, и имеет ли этот файл расширение zip. В противном случае останавливаем скрипт.

```
<?
if (file_exists($filepath)) {
    $pinf=pathinfo($filepath, PATHINFO_EXTENSION);
     if($pinf !== "zip" ){
          echo "<br>Такого архивного файла $filepath не существует<br>";
          exit();
     }else {
     $pfold=pathinfo($filepath, PATHINFO_FILENAME);
     }
} else {
    echo "<br>Такого файла $filepath не существует<br>";
     exit();
}
?>
```

6. Подключаем библиотеку для работы с архивами

```
<? require_once('pclzip.lib.php'); ?>
```

7. И наконец распаковываем архив в текущую папку

```
<?
$archive = new PclZip($filepath);
if ($archive->extract() == 0) {
die("Error : ".$archive->errorInfo(true));
}
?>
```

Полный код приложения

```
<?php header('Content-Type: text/html; charset= utf-8')?>
<!DOCTYPE HTML>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Раскрытие архивов</title>
<style type="text/css">
</style>
</head>
<body>
<div id="page">
 <center>
  <br><br><br><br> <br>
<p style='font-size:25px;  '>Хотите открыть архив? </p>
 <form name='formarhiv' action="<?=$_SERVER['PHP_SELF']?>" method='post'>
 <br>
  <br>
<p>Имя папки<br><input type="text"   name="namefold"  /></p>
<p>Имя файла<br /><input type='text' name='namefil' required  ></p>
 <p><input type='submit'value="Открыть архив"> </p>
</form>
<?
echo " <br>сейчас в текущей папке   лежат следующие файлы:<br><br>";
$dir = opendir('.');

//List files in images directory
while (($file = readdir($dir)) !== false)
  {

  echo  $file . "<br />";
  }
 closedir($dir);
?>

<?
$namefold=trim($_POST['namefold']);
$namefil=trim($_POST['namefil']);

if( !empty($namefold)){
if( is_dir($namefold) === false )
{
    echo "<br>ТАКОЙ ПАПКИ ".$namefold." НЕ СУЩЕТВУЕТ!!!<br>";
    exit();
}
else
{
$filepath =$namefold."/".$namefil;
}
}
else
{
$filepath =$namefil;
```

```
}

if (file_exists($filepath)) {
    $pinf=pathinfo($filepath, PATHINFO_EXTENSION);
     if($pinf !== "zip" ){
          echo "<br>Такого архивного файла $filepath не существует<br>";
          exit();
     }else {
     $pfold=    pathinfo($filepath, PATHINFO_FILENAME);
     }
} else {
    echo "<br>Такого файла $filepath не существует<br>";
     exit();
}

echo "<br>текущая папка   $namefold  <br>";
echo "<br> распакуем файл    $filepath  <br>";
echo "<br>расширение файла    $pinf  <br>";


require_once('pclzip.lib.php');
$archive = new PclZip($filepath);
if ($archive->extract() == 0) {
die("Error : ".$archive->errorInfo(true));
}
?>
<div id="footer">
<div class="top"></div>
<div id="bottom_menu">      </div>
<div id="bottom_addr"></div>
</div>
</div>
 </center>
</body>
</html>
```

Рабочее задание

1. Из папки **extractarhive** загрузить на сервер в свою папку файлы **extractarhive.php** и **pclzip.lib.php**
2. Загрузить на сервер в свою папку какой нибудь архивный **zip**-файл. (Внимание!!!! В названии архивного файла и названиях внутренних файлов не должно быть русских букв и пробелов!!!)
3.открыть файл **extractarhive.php**, ввести в текстовое поле название архивного файла и распаковать архив в текущую директорию.
4. Создать в своей папке на сервере какую нибудь внутреннюю папку и записать в нее **zip**-файл. (Внимание!!!! В названиях папки, архивного файла и названиях внутренних файлов не должно быть русских букв и пробелов!!!)
5. открыть файл **extractarhive.php**, ввести в текстовые поля названия внутренней папки и архивного файла и распаковать архив в текущую директорию.

Контрольные вопросы

1. Какими функциями PHP можно работать с архивными файлами и папками на удаленном сервере?
2. Каким образом можно создать архивную папку на удаленном сервере с помощью PHP?
3. Каким образом можно добавить файлы и папки в существующий архив на удаленном сервере с помощью PHP?
4. Каким образом можно извлечь файлы и папки из архива на удаленном сервере с помощью PHP?
5. Каким образом можно просмотреть содержимое архива на удаленном сервере с помощью PHP?
6. Каким образом можно изменять и обновлять файлы в архиве на удаленном сервере с помощью PHP?
7. Каким образом можно удалить файлы или папки из архива на удаленном сервере с помощью PHP?
8. Каким образом можно проверять наличие и доступность архива на удаленном сервере с помощью PHP?

9. Каким образом можно архивировать файлы и папки с различными форматами на удаленном сервере с помощью PHP?
10. Как обработать ошибки, которые могут возникнуть при работе с архивными файлами и папками на удаленном сервере с помощью PHP?

Практическая работа №10
Обработка форм с использованием php

Цель работы:
Научиться обрабатывать контактные пользовательские html-формы средствами php

Теоретическая часть
Итак, рассмотрим html – документ с формой (**forma.html**)

```
<html>
<head>
<link rel='stylesheet' type='text/css' href='style.css' />
<title>обработка форм на PHP</title>
</head>
<body>

<form id='forma' action='script.php' method='post'>
<h1>Данные о клиентах</h1>
<p>Заполнив данные поля формы, вы сможете отправить нам своё сообщение,
</p>
<p> Имя пользователя <br/> <input type='text' name='name1'> </p>
<p> E-mail <br /> <input type='text' name='email'> </p>
<p> Сообщение  <br />
 <textarea  name='mail' cols='40' rows='3'> </textarea> </p>
<p> <input type='submit'> </p> </form>

</body>
</html>
```

На что следует обратить внимание?
1) обязательно должен присутствовать параметр **action**, указывающий на скрипт обработчика формы (**script.php)**
2) обязательно указывать метод передачи данных, в данном случае **method='post'**
3) обязательно указывать **name** для каждого поля формы. Имена полей обязательно в кавычках, только латиница, названия должны отражать суть.

Теперь рассмотрим обработчик этой формы (**script.php**)

```
<?php
//Передаём переменным данные форм
//Присваиваем каждой форме переменную
$name=$_POST['name1'];
$email=$_POST['email'];
$mail=$_POST['mail'];
//отправим данные из формы на e-mail админа
//Задаём переменные для письма админу о новом сообщении клиента
$address = 'здесь должен быть ваш email';
```

```
$sub = "Здесь должна быть ваша тема сообщения";
$mes = "Посетитель: $name \nУказал свой адрес: $email \n Содержание письма: $mail";
//Отправляем письмо админу о новом сообщении
$verify = mail ($address,$sub,$mes,"From: $email");
//проверка отправилось ли письмо
if ($verify == 'true')
{
echo "<p>Сообщение отправлено<br /><br />
<a href=forma.html >Вернуться назад </a> </p>";
echo '<META HTTP-EQUIV=Refresh Content="3;URL=forma.html">';
}
else
{
echo "<p>Сообщение не отправлено<br /><br />
<a href=forma.html >Вернуться назад </a> </p>";
echo '<META HTTP-EQUIV=Refresh Content="3;URL=forma.html">';
}
?>
```

Разъяснение по отправке почты
php-функция реализующая отправку сообщения электронной почты.

mail(получатель, тема, сообщение, дополнительные заголовки)

Дополнительный заголовок должен содержать слово **From**:
(по сути он указывает от кого пришло сообщение)
Как увидеть отправленное вами сообщение, если работаем в Денвер-е?

На локалхост (**диск z**) откройте папку **Z:\tmp\!sendmail** и просмотрите отправленное письмо с помощью любой доступной программы. К сожалению с помощью одного только **denwer**-а мы не можем отправить письмо на указанный нами **e-mail** адрес, поэтому все письма отправленные нами складываются в эту папку.

Дело в том, что функция **mail** сама по себе почту **не отправляет**, она просто вызывает программу **sendmail**, которая в дистрибутив web сервера и php интерпретатора **не входит** (и не должна).

Sendmail, в свою очередь, для отправки почты использует **SMTP сервер**.

Таким образом, чтобы php скрипт мог отправлять почту нужно **установить и настроить sendmail и SMTP сервер**.

Рабочее задание

1. Загрузить на сервер архивную папку «flower514»
2. Открыть архив, открыть файл index1.html
3. Убедиться в работоспособности обработчика формы (script1.php)
4. Разработать обработчик контактной формы для другого сайта.

Контрольные вопросы

1. Как обработать данные, отправленные из HTML-формы на сервер с помощью PHP?
2. Каким образом можно получить значения полей формы, отправленной из HTML, в PHP?
3. Как проверить, была ли форма отправлена на сервер с помощью PHP?
4. Каким образом можно проверить и фильтровать пользовательский ввод, полученный из формы, в PHP?
5. Как обработать и сохранить данные из HTML-формы в базе данных с помощью PHP?
6. Каким образом можно валидировать данные из HTML-формы на стороне сервера с помощью PHP?
7. Как реализовать отправку уведомлений или сообщений на электронную почту при отправке формы с помощью PHP?
8. Как обрабатывать и сохранять загруженные файлы, полученные из формы, с помощью PHP?
9. Как можно защитить обработку формы от атак типа SQL-инъекции и кросс-сайтового скриптинга с помощью PHP?
10. Каким образом можно реализовать обработчик формы с валидацией на стороне сервера и выводом ошибок пользователю с помощью PHP?

Практическая работа №11
_php «файлы»

Цель работы:
Научиться работать с файловой системой на сервере средствами php

Теоретическая часть

1. как создать файл и записать в него текст?

```
<?php
$h = fopen("file.html","w");
$text = "Этот текст запишем в файл.";
fwrite($h,$text);
fclose($h);
?>
```

Таблица 9.1. Значения принимаемые параметром тип доступа

Тип доступа	Описание
r	Открывает файл только для чтения; устанавливает указатель позиции в файле на начало файла.
r+	Открывает файл для чтения и записи; устанавливает указатель файла на его начало.
w	Открывает файл только для записи; устанавливает указатель файла на его начало и усекает файл до нулевой длины. Если файл не существует, то пытается создать его.
w+	Открывает файл для чтения и записи; устанавливает указатель файла на его начало и усекает файл до нулевой длины. Если файл не существует, то пытается создать его.
a	Открывает файл только для записи; устанавливает указатель файла в его конец. Если файл не существует, то пытается создать его.
a+	Открывает файл для чтения и записи; устанавливает указатель файла в его конец. Если файл не существует, то пытается создать его.
x	Создает и открывает файл только для записи; помещает указатель файла на его начало. Если файл уже существует, то fopen() возвращает false и генерируется предупреждение. Если файл не существует, то делается попытка создать его. Этот тип доступа поддерживается начиная с версии PHP 4.3.2 и работает только с локальными файлами.
x+	Создает и открывает файл для чтения и записи; помещает указатель файла на его начало. Если файл уже существует, то fopen() возвращает false и генерируется предупреждение. Если файл не существует, то делается попытка создать его. Этот тип доступа поддерживается, начиная с версии PHP 4.3.2, и работает только с локальными файлами.

2. Как в тот же файл добавить еще текст?

```
<?php
$h = fopen("file.html","a");
$text = "Этот текст добавим в файл.";
fwrite($h,$text);
fclose($h);
?>
```

3.Как записать данные в текстовый файл в несколько строк? (отобразится в тестовом редакторе)

```
<?php
$h = fopen("file.html","w");
$text = "Первая строка. \r\n  Вторая строка \r\n  Третья строка \r\n ";
fwrite($h,$text);
fclose($h);
?>
```

Создавая файл, нужно учитывать, под какой операционной системой вы работаете, и под какой ОС предположительно этот файл будет читаться. Дело в том, что разные операционные системы по-разному отмечают конец строки. В Unix-подобных ОС конец строки обозначается \n, в системах типа Windows - \r\n.

4.Как записать данные в текстовый файл в несколько строк? (отобразится в тестовом редакторе и браузере)

```
<?php
$h = fopen("file.html","w");
$text = "Первая строка. \r\n <br> Вторая строка \r\n<br> Третья строка \r\n <br>";
fwrite($h,$text);
fclose($h);
?>
```

Замечание: Если открывать файл с параметром **"w"** , то все что в нем было ранее, уничтожится и файл запишется по новой.
Если мы хотим оставить предыдущие записи в файле, то его надо открывать с параметром **"a"или "a+".**

5. Как прочитать из файла некоторое количество байтов?

```
<?php
$h = fopen("my_file.html","r+");  // отрываем файл на запись и чтение
$content = fread($h, 10);  // считываем 10 байт (символов)
echo $content;
fclose($h);
```

6. Как прочитать весь файл?

```
<?php
$h = fopen("my_file.html","r+");  // отрываем файл на запись и чтение
$content = fread($h, filesize("my_file.html"));  // считываем содержимое файла в строку
echo $content;
fclose($h);
?>
```

7. Как прочитать из файла несколько строк?

```
<?php
$h = fopen("file.html","r+");
$content = fgets($h); // считает первую строку файла file.html
$content1= fgets($h); // считает вторую строку файла file.html
fclose($h);
echo $content, $content1;
?>
```

8. как прочитать весь файл?

```
<?php
$h = fopen("file.html","r");
while (!feof ($h)) {
$content = fgets($h);
echo $content,"<br>";
}
fclose($h);
?>
```

9. Как прочитать из файла несколько символов?

```php
<?php
$h = fopen("file.html","r");
$content = fgetc($h); // читаем один символ
echo $content,"<br>";
fclose($h);
?>
```

10. Как прочитать файл целиком и вывести его содержание на экран?

```php
<?php
readfile ("file.html");
?>
```

11. Как считать содержимое файла в массив?

```php
<?php
$arr = file ("file.html");
print_r($arr);
?>
```

12. Как получить нужную строку файла?

```php
<?php
$arr = file ("file.html");
echo $arr[5];
?>
```

13. как считать весь файл в одну строку?

```php
<?php
$n=file_get_contents ("file.html");
echo $n;
?>
```

14. Как записать информацию в нужную строку файла?

```php
<?php
$h = fopen("file.html","r+");

for($i=0; i<4; i++)
 fgets($h);
fwrite($h,"чего то пишем в пятую строку файла");



fclose($h);
?>
```

15.Как начать читать файл сначала, если часть его уже была прочитана?

```php
<?php
 $h = fopen("a.txt", "rt");
 $content = fgets($h);
  echo $content."<br />";
  fseek($h, 0);
$content = fgets($h);
  echo $content"<br />";
?>
```

Порядок выполнения работы:

Из предыдущих заданий взять html - документ с формой отправки данных на сервер. (если такого файла нет, то создать). В параметре **action** формы указать название файла-обработчика формы, например: "**w_r_file.php**". (**Как альтернатива**: можно вообще убрать параметр **action** из формы, переименовать документ, дав ему расширение **.php** и все php- скрипты писать в этом же документе, разместив их ниже кнопки отправки формы)

Рабочее задание:

1. С помощью **php** открыть файл с расширением **html** и с помощью функции **fwrite()** записать туда все значения текстовых полей формы следующими способами:
а) все данные последовательно в одной строке;
б) значение каждого текстового поля в своей строке.

2. Дописать в файл приветствие пользователю, таким образом, чтобы при открытии файла в браузере оно отобразилось читабельно и красиво. Закрыть созданный файл.

3. Открыть созданный файл для чтения.
С помощью функций **fread** и **filesize** считать содержимое файла в одну строку и вывести результат на экран.
С помощью функции fgets() считать:
а) первую строку файла;
б) первые пять символов первой строки.
в) используя feof() и fgets() считать все данные из файла;
Результаты вывести на экран.
Подсказка: возврат в начало файла для повторного чтения осуществлять с помощью функции **fseek(дескриптор, позиция, SEEK_SET);**

4. С помощью функции **fgetc** считать первые 7 символов из файла, потом вернуться в начало файла и считать первые 5 символов. Результаты вывести на экран. Затем распечатать весь файл посимвольно.

5. С помощью функции readfile выдать на экран содержимое всего файла.

6. С помощью функции file() считать содержимое файла в массив. Вывести на экран (из массива) вторую и четвертую строки файла.

7. С помощью функции fgets() считать первые 3 строки файла, затем вернуть курсор в начало файла и прочитать первые 2 строки. Результаты вывести на экран.

8. Написать скрипт, подсчитывающий количество строк в файле

9. Записать текст: «А вот и оно!» в четвертую строку файла.

10. Создать три php-документа. В одном из них реализовать алгоритм чтения файла и выдачи его содержимого на экран. Во втором документе реализовать алгоритм чтения третьей строки файла и выдачи его на экран. В третьем документе с помощью функции include реализовать вызов скрипта 1-го и 2-го документов.

Контрольные вопросы

1. Каким образом можно создать новую папку на сервере с помощью PHP?
2. Каким образом можно проверить, существует ли папка на сервере с помощью PHP?
3. Каким образом можно создать новый файл на сервере с помощью PHP?
4. Каким образом можно проверить, существует ли файл на сервере с помощью PHP?
5. Каким образом можно переименовать папку или файл на сервере с помощью PHP?
6. Каким образом можно переместить папку или файл на сервере с помощью PHP?
7. Каким образом можно скопировать папку или файл на сервере с помощью PHP?
8. Каким образом можно удалить папку или файл на сервере с помощью PHP?
9. Каким образом можно получить список файлов и папок в определенной директории на сервере с помощью PHP?
10. Каким образом можно проверить, является ли объект на сервере файлом или папкой с помощью PHP?

Практическая работа №12
Сессии php. Регистрация и авторизация.

Цель работы:
Научиться использовать сессии php для создания личного кабинета пользователя на веб сайте

Теоретическая часть
При разработке сайтов практически всегда требуется ***авторизация***. Например, для входа в личный кабинет пользователя..
Форма авторизации обычно состоит из двух текстовых полей (Логин и пароль) и кнопки Войти.
PHP сессии
Сессии нужны для сохранения определенной информации на стороне браузера. Практически тоже самое что и переменные, отличается только тем, что переменные при переходе на другую страницу сайта, или при обновлении текущей страницы теряют свои значения, а данные записанные в сессии сохраняются, и доступны на любых страницах сайта.
Например, на странице **http://ox2.ru/index.php** мы записали в сессию **'session_test'** значение **'123**'. На странице **http://ox2.ru/shop.php** мы можем прочитать сессию **session_test**, и получить значение **123**.

Для работы с сессиями на php нужно на каждой странице где будет производиться работа с сессиями написать **session_start()**, в самом начале, до вывода любой информации на экран.
Для записи в сессии существует переменная **$_SESSION**.

Пример

//Открываем сессию в **index.php**
```
<?php
session_start();
```
// устанавливаем значение переменной сессии равное 1 (**'auth'** это просто ключ массива, он может быть любым – какой придумаете)
```
$_SESSION['auth']=1;
?>
```
// теперь на любой странице сайта можно выдать на экран значение переменной **$_SESSION['auth']** и оно будет одинаковым и равным 1
```
<?
echo$_SESSION['auth'];
?>
```
//А чтобы обнулить переменную сессии достаточно написать следующее:
```
<?
unset($_SESSION['auth']);
?>
```

Для чего это нам может понадобиться?
- для реализации алгоритма авторизации пользователя.
Пусть у нас имеется некоторая html-форма, в которую пользователь вводит свой логин и пароль. Данные формы передаются php-скрипту, который проверяет соответствие данного логина паролю и наличие пользователя в базе данных. Если пользователь найден в базе данных и пароль соответствует логину, то мы устанавливаем переменную сессии **$_SESSION['auth']=1;**
И всё! Пользователь вошел в систему. Теперь на любой странице сайта переменная сессии будет сохранять своё значение, и это позволит нам идентифицировать данного пользователя.
Если же пользователь захочет выйти из системы, то нажмет на кнопку «выход». Обработчик данной формы выполнит команду **unset($_SESSION['auth']);**После этого переменная сессии **$_SESSION['auth']**обнулится.

Отметим еще, что описанный здесь алгоритм не предполагает использование базы данных MySQL.Данные (логины и пароли) зарегистрированных пользователей хранятся в текстовом файле.

Внимание! кодировка всех php-файлов - utf8
наличие файла .htaccess в папке с сайтом обязательно!!!!

Рассмотрим код авторизации более подробно:

```
<?php
session_start();// запускаем сессию

if(empty($_SESSION['auth'])) //если переменная сессии пуста, то выводим на //экран форму авторизации

{
   ?>
<table>
<form method=post >
<tr>
<td>Имя:</td>
<td><input type=text name=name></td>
</tr>
<tr>
<td>Пароль:</td>
<td><input type=password name=pass></td>
</tr>
<tr>
<td> </td>
<td><input type=submit value='Войти'></td>
</tr>
</form>
</table>
```

```
<?php

if(isset($_POST['name'])&&isset($_POST['pass'])) // если переданы логин и //пароль, то входим в блок обработки данных пользователя

{

$filename = "text.txt";  // имя файла в котором хранятся данные пользователей

// Проверяем корректность введённого имени
// ипароля
   $arr = file($filename); // считываем содержимое файла в массив $arr
$i = 0;
   $temp = array();
foreach($arr as $line)
{
// Разбиваем строку по разделителю ::
     $data = explode("::",$line);
// В массив $temp помещаем имена и пароли
      // зарегистрированных посетителей

$temp['name'][$i]     = $data[0];
$temp['password'][$i] = $data[1];
     $temp['email'][$i]    = $data[2];
     $temp['url'][$i]      = trim($data[3]);
// Увеличиваем счётчик
     $i++;
   }
```

```
        // Если в массиве $temp['name'] нет введённого
           // логина - останавливаем работу скрипта
        if(!in_array($_POST['name'],$temp['name']))
           {
        echo("Пользователь с таким именем не зарегистрирован");
           }
        else
          {
        // Если пользователь с именем $_POST['name'] обнаружен
           // проверяем правильность введённого пароля
           $index = array_search($_POST['name'],$temp['name']);
        if($_POST['pass'] != $temp['password'][$index])
        {
        echo("Пароль не соответствует логину");
           }
        else
           {
        // если пароль соответствует логину, то устанавливаем значения //переменных сессии
        $_SESSION['auth']=1;

            $_SESSION['name']=$_POST['name'];
        $_POST['password']=$_POST['pass'];

        // перезагружаем страницу, чтобы сбросить значения $_POST
        echo "<HTML><HEAD>
        <META HTTP-EQUIV='Refresh' CONTENT='0; URL=$_SERVER[PHP_SELF]'>
        </HEAD></HTML>";
          }
        }
   }
}
  ?>
```

```
        // если  существует $_SESSION['auth'], то есть пользователь зарегистрирован, //то пишем приветствие и выводим на экран кнопку «выход»
        <?php
        if(isset($_SESSION['auth']))
        {
        echo "Здравствуйте  ".$_SESSION['name']."<br>";
        ?>
        <form method=post >
        <input type=submit name='exzit' value='Выход'>
        </form>

        //если пользователь нажал кнопку «выход», то обнуляем переменную сессии и //перезагружаем страницу
        <?php
        if(isset($_POST['exzit']))
        {
        unset($_SESSION['auth']);
        echo 'Выход успешен';
        echo "<HTML><HEAD>
        <META HTTP-EQUIV='Refresh' CONTENT='0; URL=$_SERVER[PHP_SELF]'>
        </HEAD></HTML>";
          }
        }
        ?>
```

Регистрация пользователя

Внимание! кодировка всех php-файлов - utf8

наличие файла .htaccess в папке с сайтом обязательно!!!!

Регистрация пользователя будет производится скриптом, находящемся в файле **registr1.php**.
А в **index.php** мы расположим ссылку на этот скрипт.

Рассмотрим более подробно код скрипта регистрации пользователя.
(файл **registr1.php**.)

```
<! Вывод формы для регистрации>
<table>
<form method=post>
<tr><td>Имя:</td><td><input type=text name=name></td></tr>
<tr><td>Пароль:</td><td><input type=password name=pass></td></tr>
<tr><td>Пароль:</td><td><input type=password name=pass_again></td></tr>
<tr><td>e-mail:</td><td><input type=text name=email></td></tr>
<tr><td>URL:</td><td><input type=text name=url></td></tr>
<tr><td></td><td><input type=submit value='Зарегистрировать'></td></tr>
</form>
</table>
<a href="index.php">Вернуться на главную</a>
```

```
<?php

// Обработчик HTML-формы
session_start();
if(empty($_SESSION['auth']))
{
//////////////////////////////////////////////////
  // 1. Блок проверки правильности данных
//////////////////////////////////////////////////
// Удаляем лишние пробелы
$_POST['name'] = trim($_POST['name']);
  $_POST['pass'] = trim($_POST['pass']);
  $_POST['pass_again'] = trim($_POST['pass_again']);
// Проверяем не пустой ли суперглобальный массив $_POST

  // Проверяем правильно ли заполнены обязательные поля
if(empty($_POST['name'])) exit('Поле "Имя" не заполнено');
if(empty($_POST['pass'])) exit('Одно из полей "Пароль" не заполнено');
if(empty($_POST['pass_again'])) exit('Одно из полей "Пароль" не
 заполнено');
if($_POST['pass'] != $_POST['pass_again']) exit('Пароли не совпадают');
// Если введён e-mail проверяем его на соответсвие
if(!empty($_POST['email']))
  {
if(!preg_match("|^[0-9a-z_]+@[0-9a-z_^\.]+\.[a-z]{2,6}$|i", $_POST

['email']))
    {
exit('Поле "E-mail" должно соответствовать формату

somebody@somewhere.ru');
    }
  }

//////////////////////////////////////////////////
  // 2. Блок проверки имени на уникальность
  //////////////////////////////////////////////////
// Имя файла данных
$filename = "text.txt";
  // Проверяем не было ли переданное имя
```

```
  // зарегистрировано ранее
$arr = file($filename);
foreach($arras $line)
  {
// Разбиваем строку по разделителю ::
$data = explode("::",$line);
// В массив $temp помещаем имена уже зарегистрированных
    // посетителей
$temp[] = $data[0];
  }
// Проверяем не содержится ли текущее имя
// вмассивеимён $temp
if(in_array($_POST['name'], $temp))
{
exit("Данное имя уже зарегистрировано, пожалуйста, выберите другое");
  }

//////////////////////////////////////////////////
  // 3. Блок регистрации пользователя
  //////////////////////////////////////////////////
  // Помещаем данные в текстовый файл
$fd = fopen($filename, "a");
if(!$fd) exit("Ошибка при открытии файла данных");
$str = $_POST['name']."::".
        $_POST['pass']."::".
        $_POST['email']."::".
        $_POST['url']."\r\n";
fwrite($fd,$str);
fclose($fd);
echo "Уважаемый  ".$_POST['name']."! вы успешно зарегистрировались на

сайте!";

}
else
{
echo "Уважаемый пользователь! Вы уже зарегестрированы";

}
?>
```

Рабочее задание

1. Загрузить на сервер архивную папку «auto_reg.zip», открыть архив

2. Отладить скрипты авторизации и регистрации

3. Создать свой сайт с личным кабинетом, использующий авторизацию и регистрацию «на файлах»

Внимание! кодировка всех php-файлов - utf8
наличие файла .htaccess в папке с сайтом обязательно!!!!

Контрольные вопросы

1. Как создать базу данных для системы авторизации и регистрации на сайте с помощью PHP?
2. Каким образом можно создать формы регистрации и авторизации пользователей на сайте с помощью PHP?
3. Как осуществить проверку введенных пользователем данных при регистрации (например, проверку уникальности email) с помощью PHP?

4. Как добавить нового пользователя в базу данных при регистрации с помощью PHP?
5. Каким образом можно реализовать аутентификацию пользователей при входе на сайт с помощью PHP?
6. Как провести проверку правильности введенного пароля при авторизации пользователя с помощью PHP?
7. Каким образом можно сохранить данные пользователя (например, его идентификатор) на протяжении сессии с помощью PHP?
8. Как создать страницу для "выхода" пользователя из системы с помощью PHP?
9. Как реализовать восстановление пароля пользователя с помощью PHP?
10. Как обеспечить безопасность системы авторизации и регистрации средствами PHP (например, хранение паролей в зашифрованном виде)?

Практическая работа №12

по теме: Работа с базой данных mysql в PhpMyAdmin

Создание баз данных в phpMyAdmin

Цель работы:

Научиться создавать БД, писать запросы

Порядок выполнения работы

Запустим Денвер нажав "Start Denwer", после, пройдем по ссылке localhost и нажмем на ссылку Заведение новых БД и пользователей MySQL для создания базы данных (Рисунок 1)

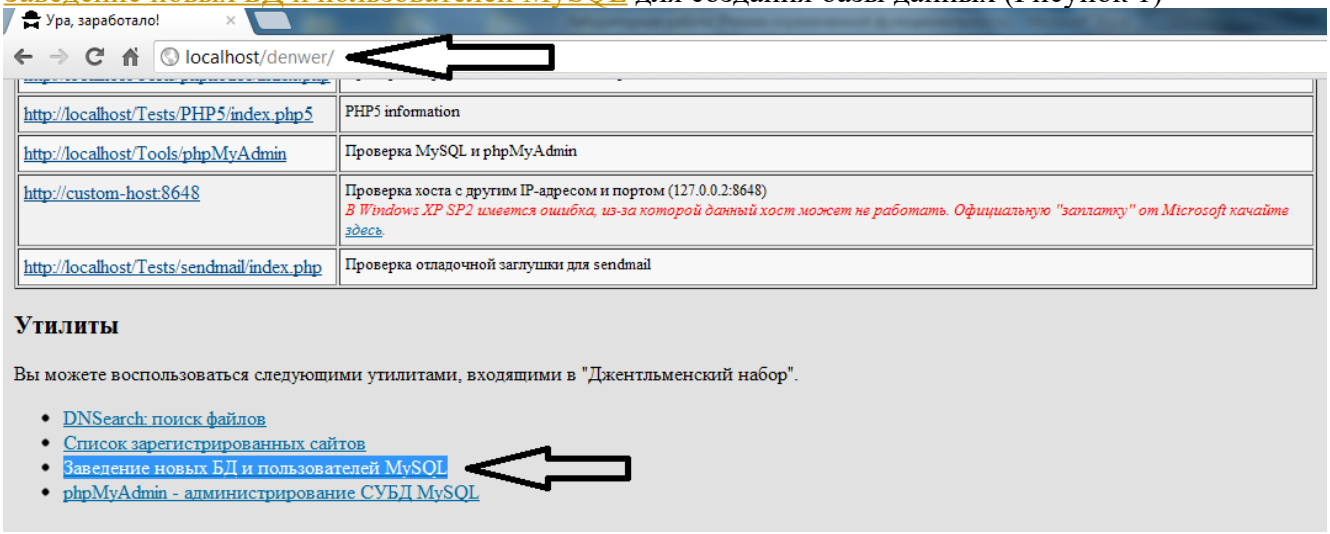


Рисунок 1 - Нажатие на необходимую ссылку

В окне введем имя базы данных, логин, пароль и нажимаем Создать БД пользователя (Рисунок 2)

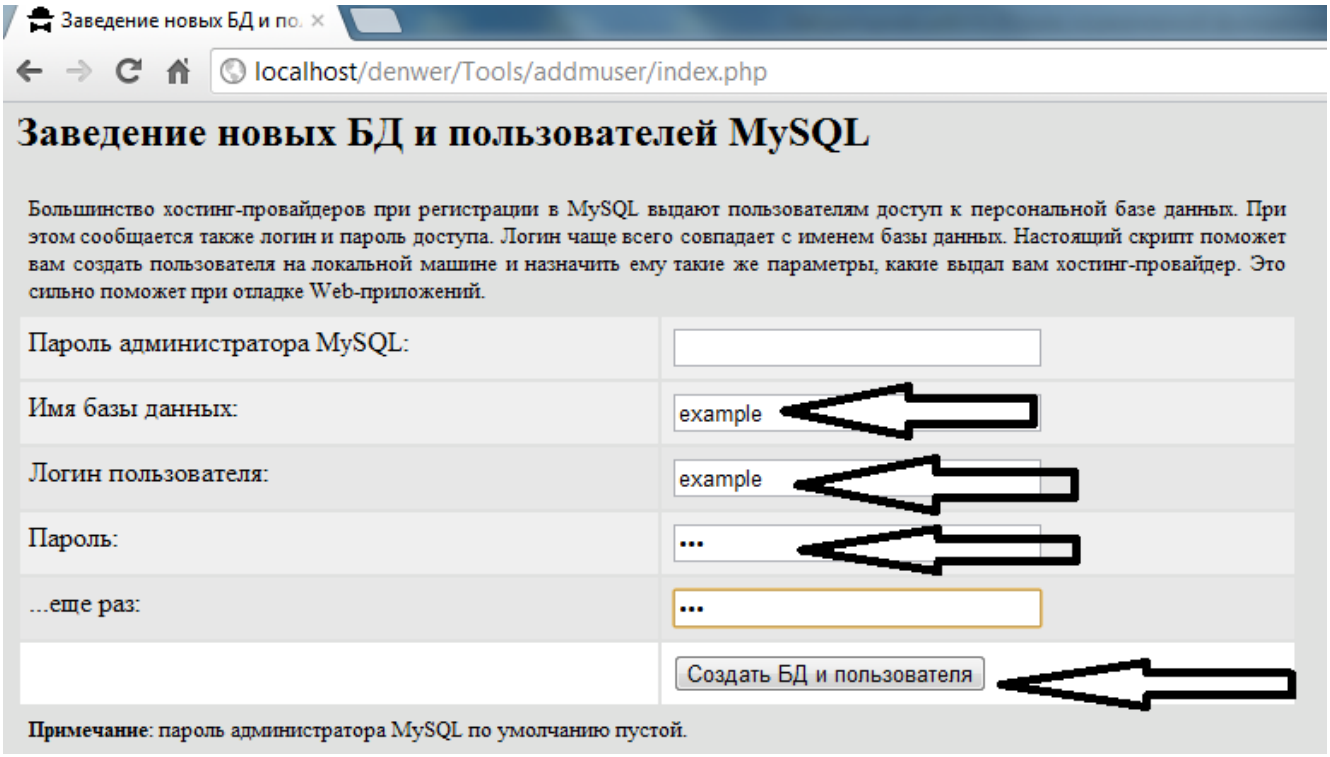


Рисунок.2 - Создание нового пользователя

После чего, возвращаемся на 2 шага назад и переходим на ссылку phpMyAdmin - администрирование СУБД MySQL. phpMyAdmin - это утилита которая помогает взаимодействовать с MySQL базами данных.И видим нашу пустую базу данных (Рисунок 3).

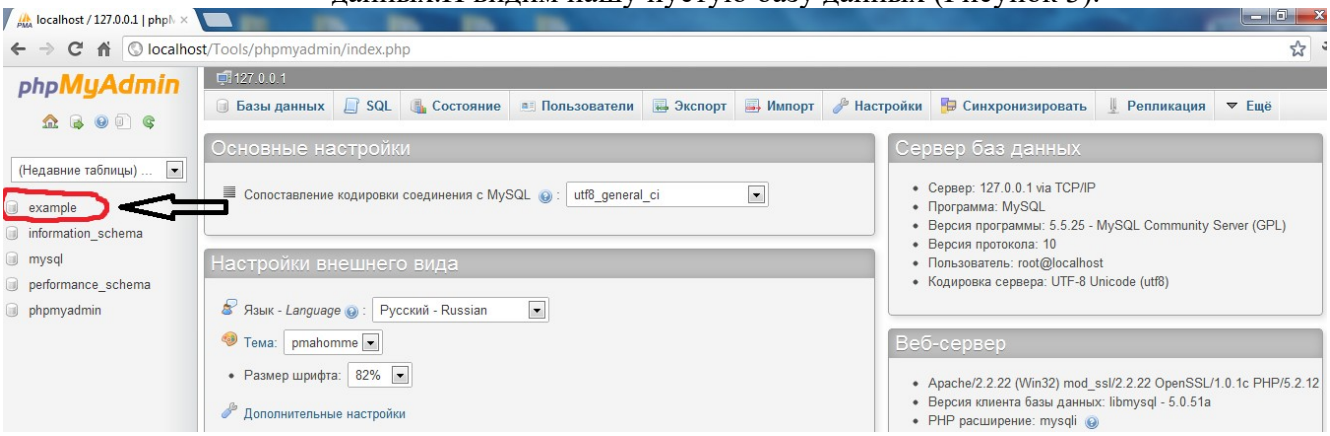


Рисунок 3 - Начало базы

Также создать базу данных можно непосредственно в самом phpMyAdmin. Переходим во вкладку Базы Данных -> Создать БД (пишем название бд, кодировку выбираем utf8_general_ci) -> Создать (Рисунок 4)

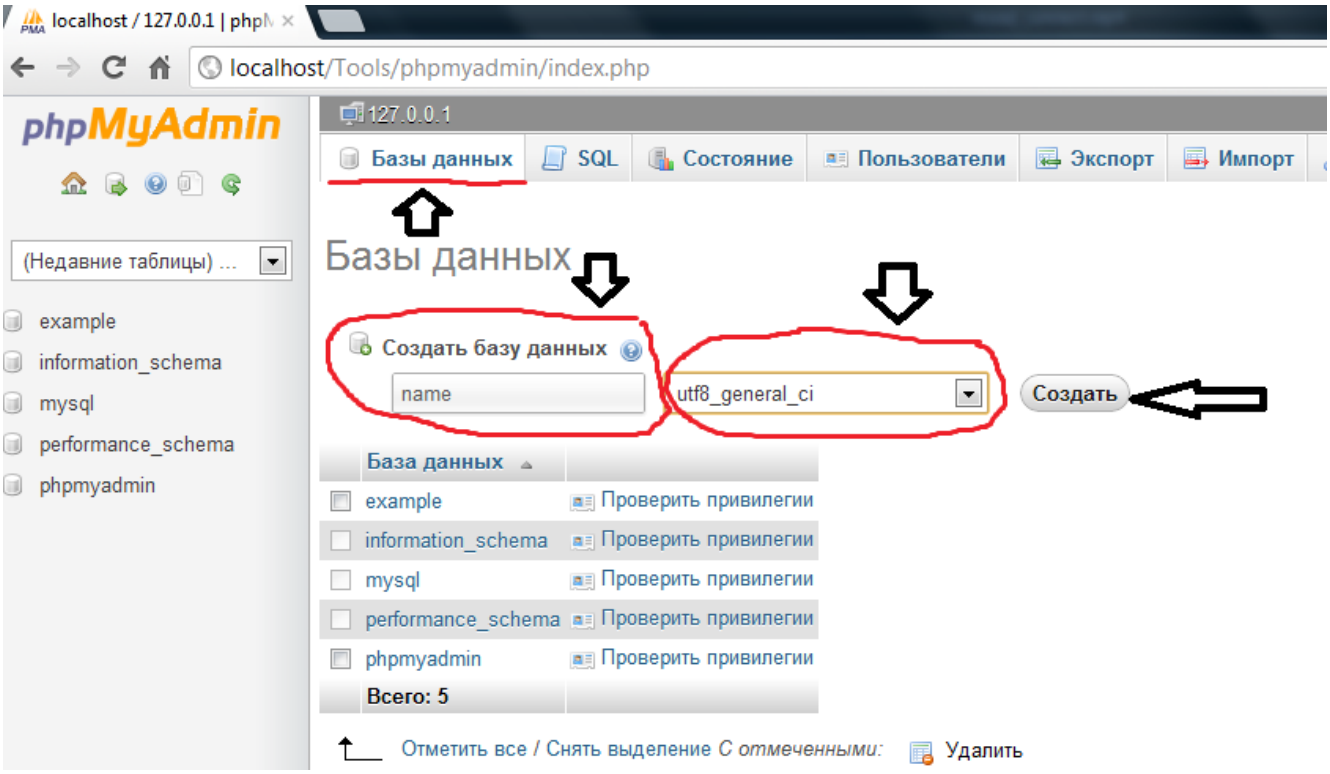


Рисунок 4 - Заполнение данных для создания базы

Каждая база данных состоит из таблиц (новости, таблица пользователей, таблица фотографий, галереи, видео), которые потом, на сайте выводятся в цикле.

Итак, создадим первую таблицу в нашей БД: Заходим в базу данных example, и сразу видим поля заполнения для создания таблицы (название таблицы, количество полей), заполняем их, после чего нажимаем ОК. В нашем примере создадим таблицу пользователей - Users (Рисунок 5)

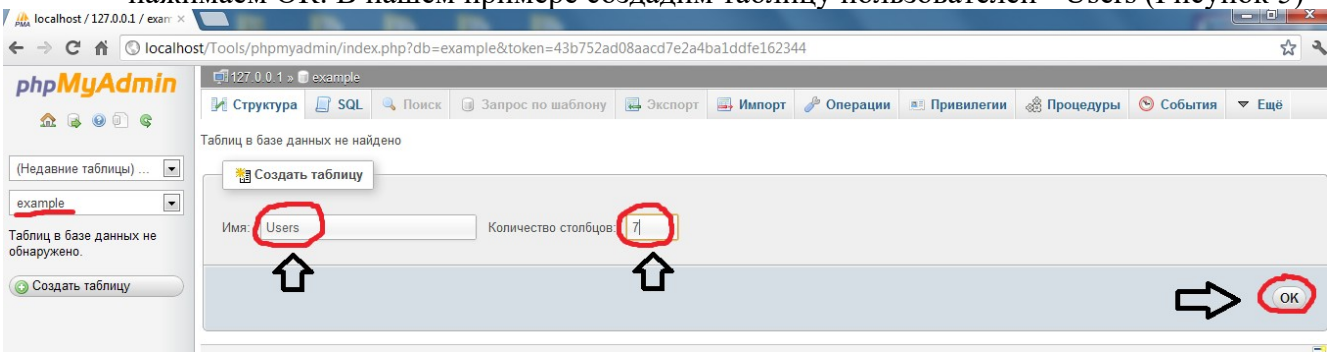


Рисунок 5 - Создание базы

После нажатия ОК мы видим поля для заполнения. В первом столбце мы пишем название нашего поля, во втором - тип данных, в 3м - длина / значение и тд. Заполняем первый поле: Название Поля - id, Тип данных - INT (целое число нормального размера), Длина / значения - 5 (это означает, что максимальное число будет с 5ти символов, это почти 99 999 максимальное число записей) в графе A_I ставим галочку (это означает, что поле Автоинкрементный, т.е. с добавлением новой записи идентификатор будет сам увеличиваться, нам не нужно будет вручную заполнять данное поле), Индекс - PRIMARY (это первичный ключ). Второе поле: Название Поля - name, Тип данных - VARCHAR, Длина / значение - 20 символов (максимальное количество символов, нам этого хватит). Третье поле: Название Поля - last_name, Тип данных - VARCHAR, Длина / значение - 20 символов. И так далее, постепенно, заполняем все поля ... Когда поля заполнены нажимаем Сохранить (Рисунок 6)

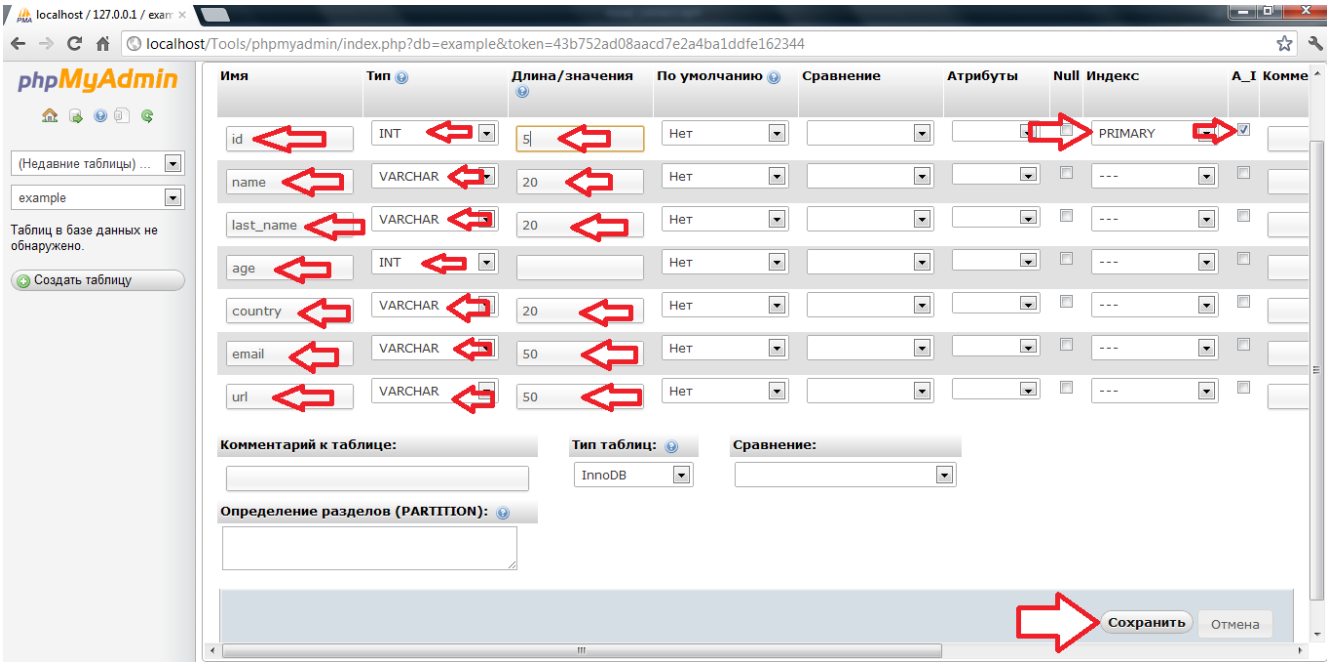


Рисунок 6 - Заполнение полей таблицы

После чего мы видим, что в нашей example появилась таблица users(Рисунок 7)

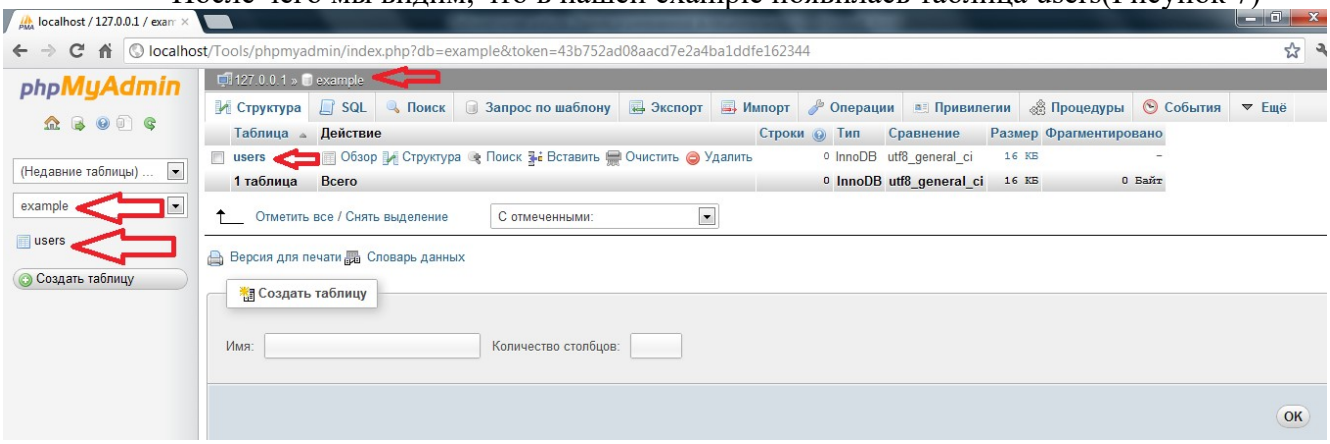


Рисунок 7 - Таблица создана

Нажимаем на таблицу users и мы видим ее структуру (все как мы и заполняли) (Рисунок 8)

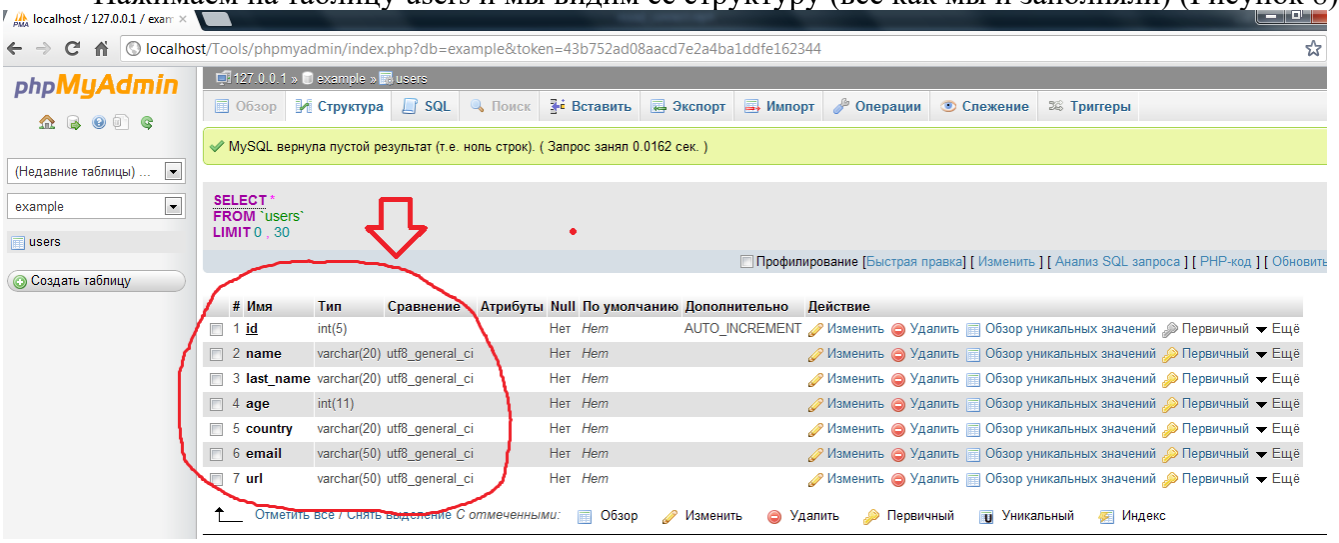


Рисунок 8 - Структура таблицы

Теперь, необходимо заполнить нашу таблицу данными: переходим во вкладку Вставить и заполняем: Поле id мы не заполняем, тк оно у нас автоматически увеличиваться, во все остальные поля заносим данные (Рисунок 9).

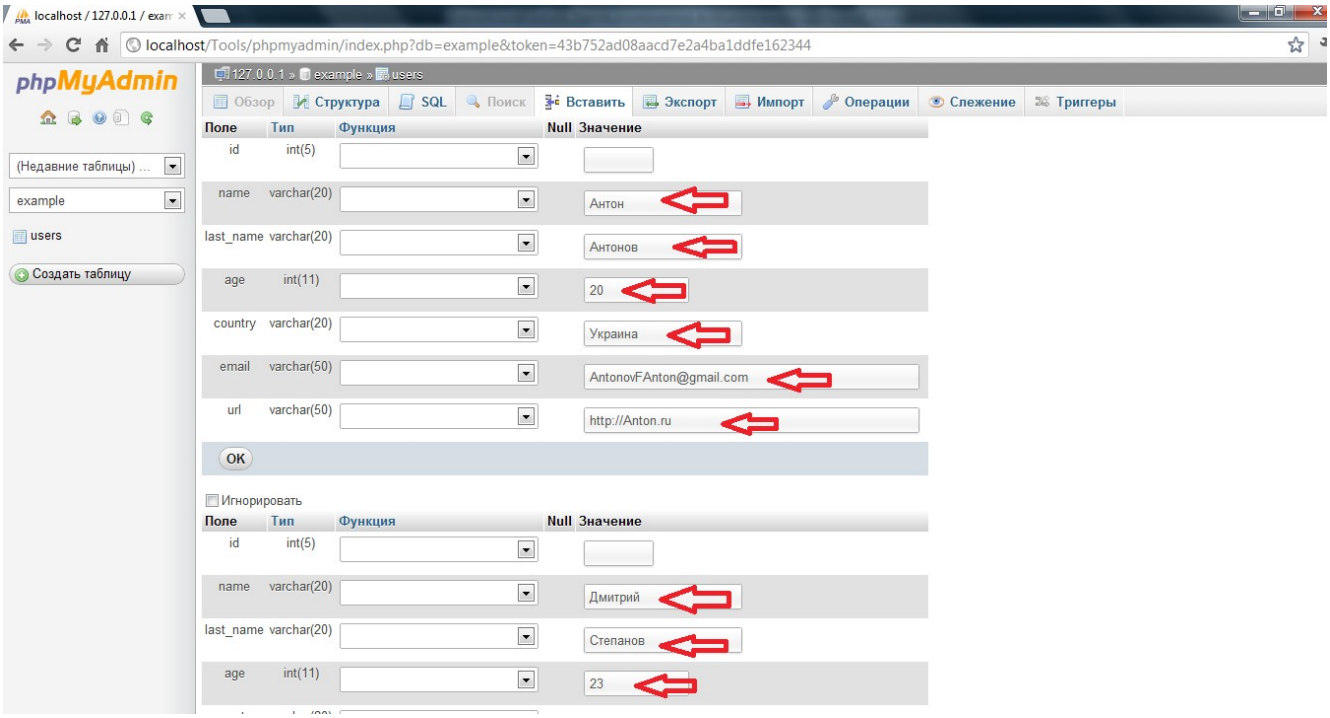


Рисунок 9 - Заполнение таблицы данными

нажимаем ОК, после чего вы попадете на страничку изображенную на Рисунке 10

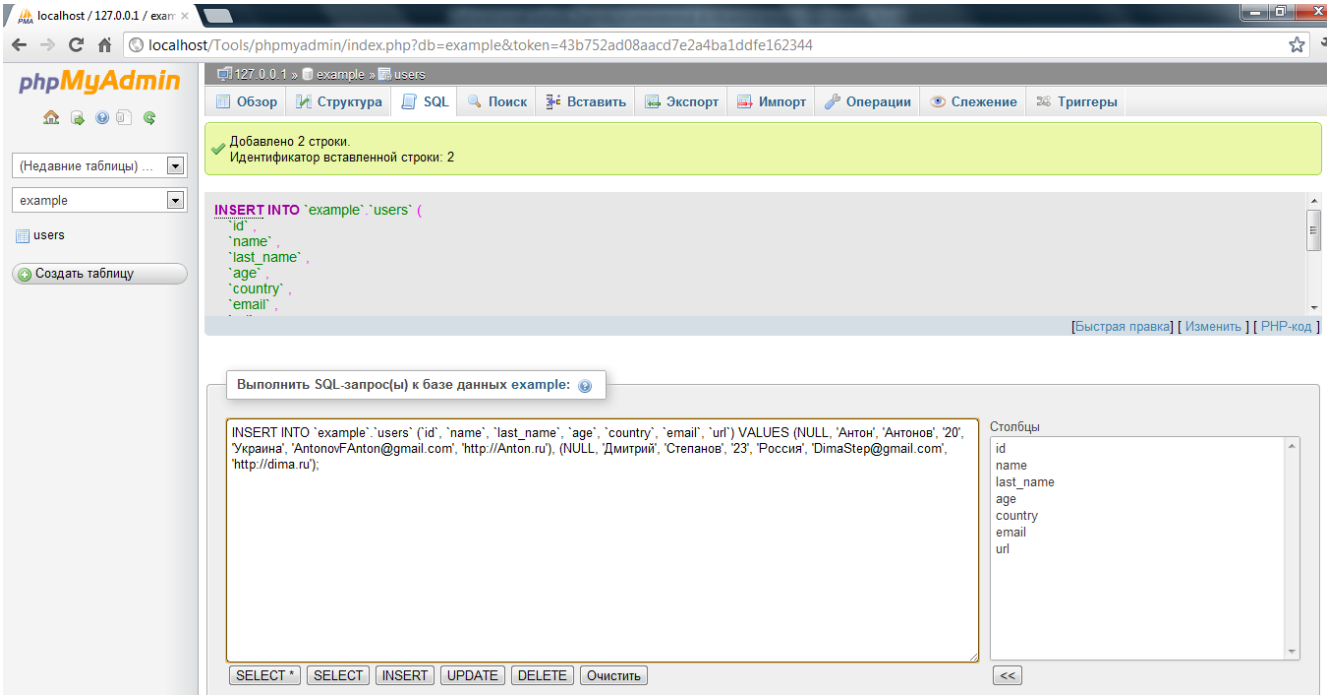


Рисунок 10 - Сформированный запрос после заполнения таблицы

Далее, переходим на вкладку обзор и видим, что в нашей таблице users теперь 10 записей (ровно столько, сколько вы решили добавить). Наблюдаем, что идентификатор (id) самостоятельно увеличивается (Рисунок 11)

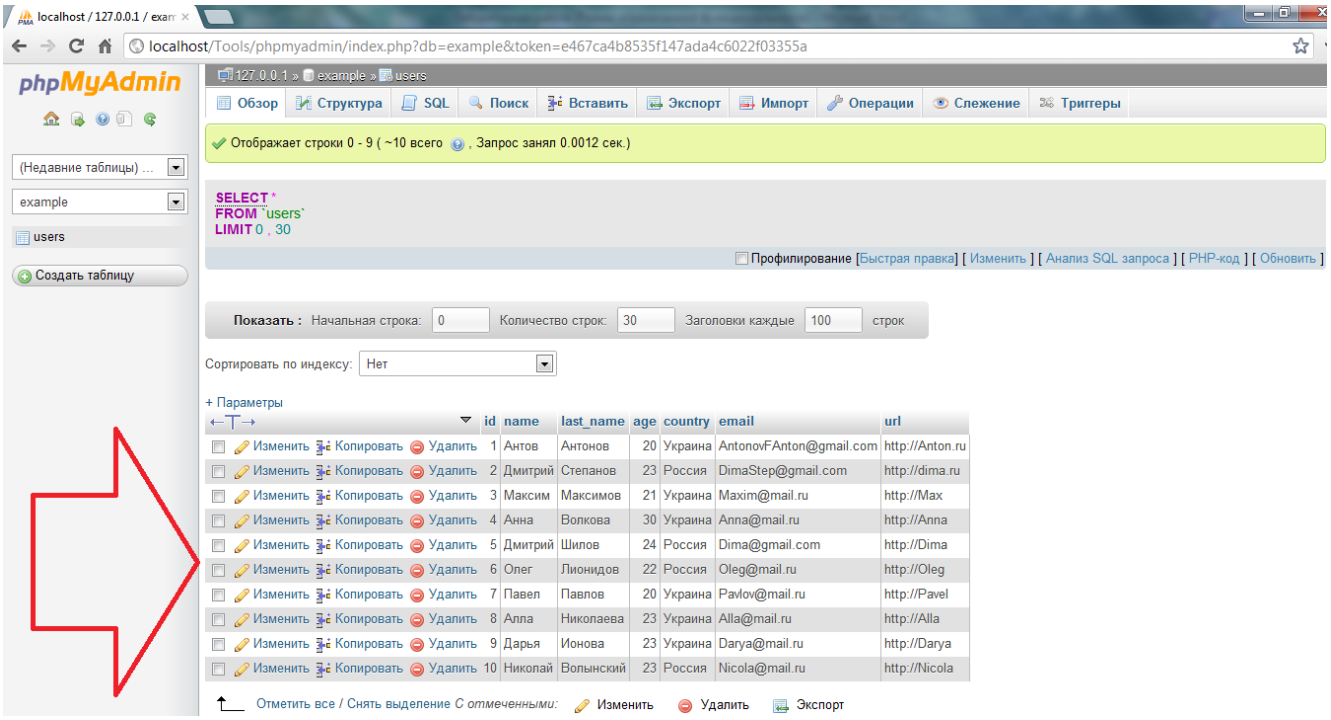


Рисунок 11 - самостоятельное увеличение идентификатору

Составление запросов в MySQL

Что такое SQL?

SQL (англ. Structured Query Language - язык структурированных запросов) - универсальный язык, применяемый для создания, модификации и управления данными в реляционных базах данных. Архитектура данных, к которой обращается SQL называется реляционной. В реляционных базах данных все данные представлены в виде простых таблиц, разбитых на строки и столбцы, на пересечении которых расположены данные. Запросы к таким таблицам возвращают таблицы, которые сами могут становиться предметом дальнейших запросов. Каждая база данных может включать несколько таблиц, которые, как правило, связаны друг с другом.

Изменение табличных данных с помощью phpMyAdmin

Мы изучим базовый синтаксис выражений INSERT, UPDATE, DELETE, и SELECT.

INSERT-добавление данных

После части INSERT INTO, следует имя таблицы. В MySQL мы можем заключать имена таблиц и имена столбцов в обратные галочки "` ", если в именах используются спецсимволы, зарезервированные слова. Затем мы открываем первый скобки, перечисляем столбцы которые будет осуществлена ?? вставка, разделяя их друг от друга запятыми. После перечисления списка названий столбцов, дужка закрывается, и указывается зарезервированное слово VALUES, после которого в скобках перечисляются значения, которые нужно вставить в таблицу, причем перечисляются в том же порядке, что и названия столбцов. Если значение имеют символьный тип данных, необходимо заключать их в кавычки

Напишем запрос, после выполнения которого добавится строка. Для этого переходим во вкладку SQL, мы видим окошко, в которое необходимо вписать наш запрос на добавление, после чего нажимаем ОК (Рисунок 12)

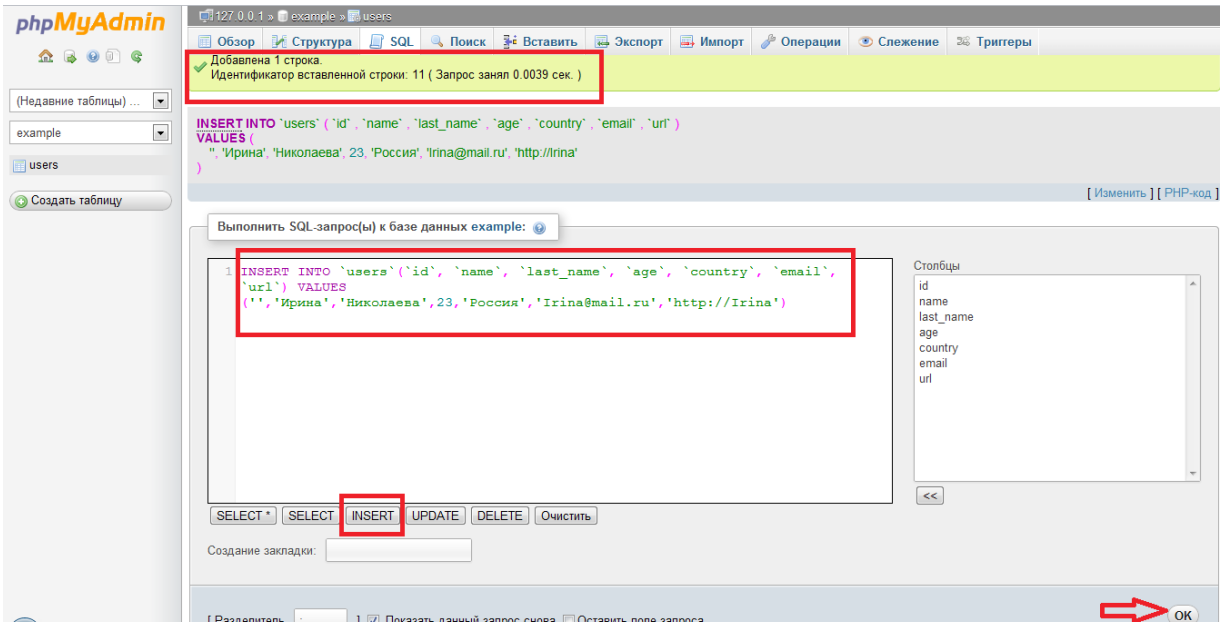


Рисунок 12 - Запрос на добавление данных

Текст запроса:

INSERT INTO `users` (`id`, `name`, `last_name`, `age`, `country`, `email`, `url`)
VALUES ('', 'Ирина', 'Николаева ", 23," Россия "," Irina@mail.ru', 'http://Irina')

После нажатия ОК переходим во вкладку Обзор и наблюдаем, что строка добавилась (Рисунок 13)

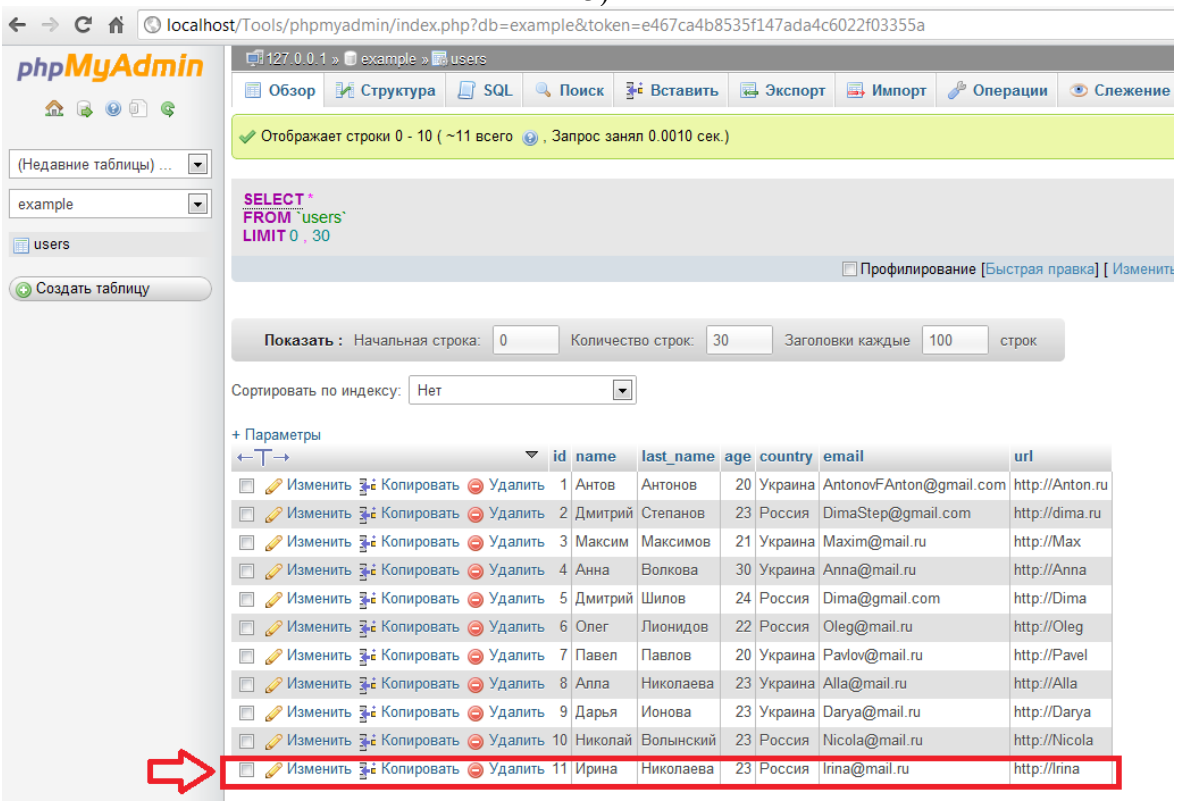


Рисунок 13 - Результат запроса

1. Обновление данных с помощью UPDATE

Оператор UPDATE обновляет столбцы в соответствии с их новых значений в строках существующей таблицы. В выражении SET указывается, какие именно столбцы следует модифицировать и какие величины должны быть в них установлены. В выражении WHERE, если оно присутствует, задается, какие строки подлежат восстановлению. В других случаях, обновляются все строки. Если для выражение ORDER BY, то строки будут обновляться в указанном в нем порядке.

Если указывается ключевое слово LOW_PRIORITY, то выполнение данной команды UPDATE задерживается до тех пор, пока другие клиенты не завершат чтение этой таблицы.

Если указывается ключевое слово IGNORE, то команда обновления не будет прервана, даже если при обновлении возникнет ошибка дублирования ключей. Строки, из-за которых возникают конфликтные ситуации, обновлены не будут.

Напишем запрос на обновление последнего поля, у которого id = 11, поменяем email и url пользователя (Irina@mail.ru на Nicolaeva@mail.ru и http://Irina на http://Nicolaeva). Написание запроса происходит аналогично предыдущему.

Текст запроса:

```
UPDATE `users`
SET `email` = 'Nicolaeva@mail.ru', `url` = 'http://Nicolaeva '
WHERE `id` = '11 '
```

Результат наблюдаем на рисунке 14

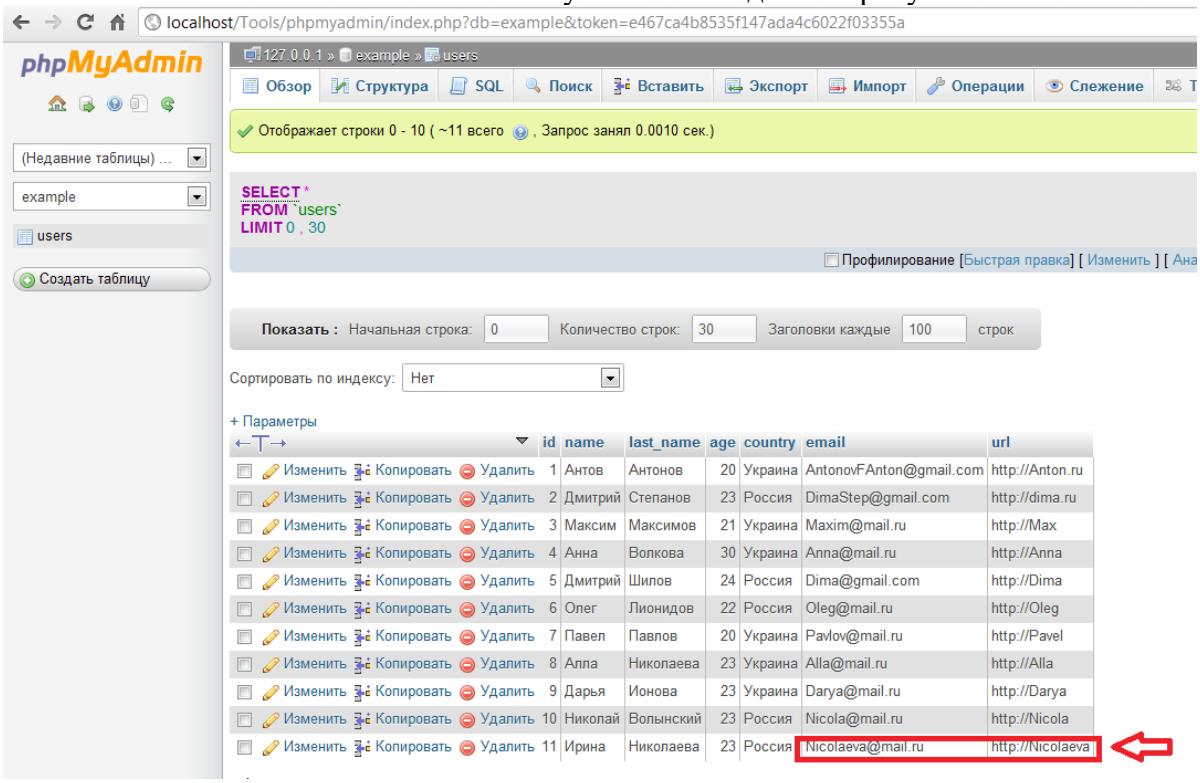


Рисунок 14 - Результат запроса на обновление

Удаление данных с помощью DELETE

Синтаксис здесь очень простой, и включает только название таблицы, и условие при котором будет выполнена операция удаления. Исключение условия WHERE из запросов UPDATE или DELETE вполне допустимо в SQL, но в таком случае действие выражение будет применен к каждой записи таблицы!

Удалим всех пользователей из России.

Текст запроса:

```
DELETE FROM `users`
WHERE `country` = 'Россия'
```

Результат выполненного запроса на рисунке 15

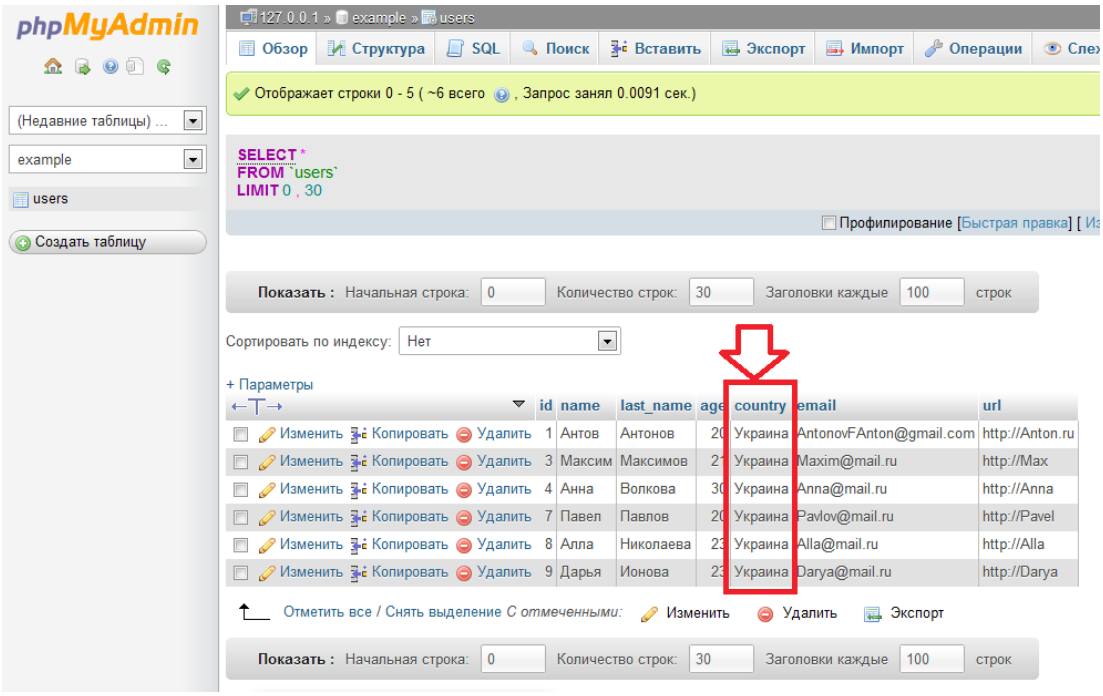


Рисунок 15. Результат запроса на удаление данных

Выборка данных с помощью SELECT

Извлечение информации из таблиц - вероятно наиболее часто используемый вид запроса. Например, выберем пользователей которым по 20 лет.

Текст запроса:

```
SELECT * FROM `users`
  WHERE `age` = '20 '
```

Звездочка здесь означает "все столбцы".

Результат на рисунке 16

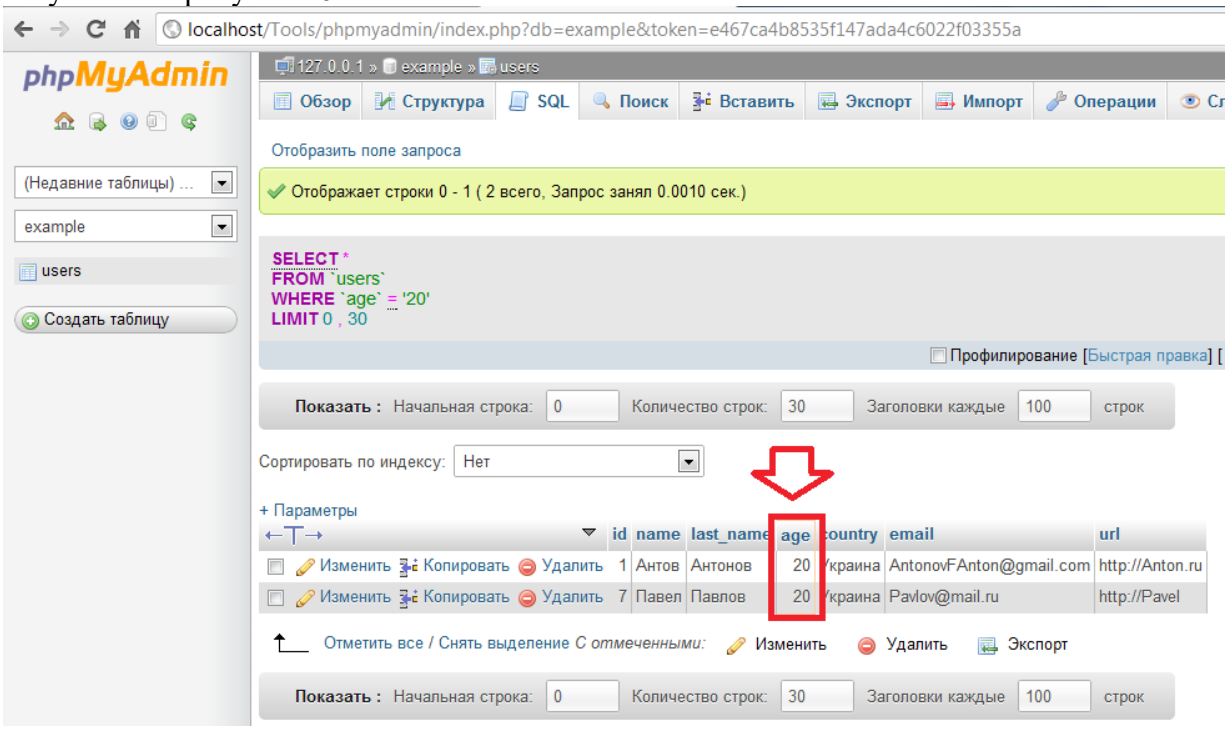


Рисунок 16 - Результат выборки данных

Контрольные вопросы

1. Каким образом можно создать новую базу данных в PhpMyAdmin?
2. Как создать новую таблицу в базе данных через интерфейс PhpMyAdmin?
3. Каким образом можно добавить новую колонку в существующей таблице в PhpMyAdmin?
4. Как изменить тип данных колонки в PhpMyAdmin?

5. Каким образом можно удалить таблицу из базы данных в PhpMyAdmin?
6. Как добавить новую запись в таблицу базы данных в PhpMyAdmin?
7. Каким образом можно обновить данные в определенной записи таблицы базы данных в PhpMyAdmin?
8. Как удалить определенную запись из таблицы базы данных в PhpMyAdmin?
9. Каким образом можно выполнить SQL-запросы в PhpMyAdmin?
10. Как экспортировать или импортировать базу данных в PhpMyAdmin?

Практическая работа №13
(MySQL в phpmyadmin)

Теоретическая часть
Теоретический материал, необходимый для выполнения практической работе можно найти в лекции, по ссылке https://disk.yandex.ru/i/9EVeyDKzmUforA

Цель работы:
Научиться создавать таблицы в СУБД MySQL, писать запросы к БД

Рабочее задание
Выполнить инструкции приведенные ниже

Порядок выполнения работы
Дана следующая схема базы данных.

1. **Реализовать запросы по созданию таблиц базы данных**.
В атрибуте ТипПубликации (таблица Источник) должно быть установлено значение по умолчанию "Статья".
2. **Выполнить вставку данных**. В каждую созданную таблицу вставить как минимум по 6 строк.
3. **Выполнить выборку данных**.
3.1. Вывессти ФИО всех исследователей, у которых фамилия начинается с буквы П.
3.2. Вывести описание исследования с названием «Теория упругости» (Описание - это атрибут таблицы Исследование, название - атрибут таблицы Источник, название "Теория упругости" можно заменить на любое другое название источника, имеющееся в базе данных)
3.3. Вывести фамилию и имя исследователей, которые опубликовали монографию (т.е. тип тубликации – "Монография")
4. **Выполнить обновление данных**.
Поменяйте фамилию исследователя Склодовская Мария Саломея на Склодовская-Кюри (если такой исследователь в базе данных отсутствует, то по аналогии взять любую другую преставительницу женского пола)

5. **Выполнить удаление данных**.
Удалите исследователей, у которых нет ни одного исследования

> В качестве решения принимается текстовый файл со всеми сохраненными в нем запросами, выполненными в рамках итоговой работы.

Контрольные вопросы

1. Каким образом можно создать новую таблицу в базе данных MySQL с помощью SQL-запроса?
2. Как добавить новую колонку в существующую таблицу в базе данных MySQL с помощью SQL-запроса?

3. Как обновить данные в определенной записи таблицы базы данных MySQL с помощью SQL-запроса?
4. Как удалить определенную запись из таблицы базы данных MySQL с помощью SQL-запроса?
5. Каким образом можно выбрать только определенные колонки из таблицы базы данных MySQL с помощью SQL-запроса?
6. Каким образом можно отфильтровать данные в таблице базы данных MySQL с помощью SQL-запроса?
7. Как сортировать данные в таблице базы данных MySQL по определенной колонке с помощью SQL-запроса?
8. Каким образом можно соединить данные из нескольких таблиц в базе данных MySQL с помощью SQL-запроса?
9. Как выполнить агрегатные функции, такие как SUM, AVG или COUNT, над данными в таблице базы данных MySQL с помощью SQL-запроса?
10. Каким образом можно создать условия для выполнения операций, таких как IF или CASE, в SQL-запросах к базе данных MySQL?

Практическая работа №14

(проектирование базы данных MySQL)
(4 часа)

Теоретическая часть

Пусть таблица **person** имеет такие поля: **number** (int(7) PRI auto_increment), **name** (varchar(25)), **last_name** (varchar(50)), **age** (int(3)),
Приведем несколько примеров запросов с коротким объяснением:

number	name	last_name	age
1	Anna	Kryku	20

Тип столбца (тип в выражении определение_столбца) может быть одним из следующих:

- целый: INT[(length)] [UNSIGNED] [ZEROFILL]
- действительный: REAL[(length,decimals)] [UNSIGNED] [ZEROFILL]
- символьный: CHAR(length) [BINARY] и VARCHAR(length) [BINARY]
- дата и время: DATE и TIME
- для работы с большими объектами: BLOB
- текстовый: TEXT
- перечислимое множество: ENUM(value1,value2,value3,...) и SET(value1,value2,value3,...)

Оператор выборки данных – SELECT
Синтаксис оператора SELECT

```
SELECT имена_столбцов from имя_таблицы [WHERE ...условия]
```

SELECT Ключевое слово, которое сообщает базе данных о том, что оператор является запросом. Все запросы начинаются с этого слова, за ним следует пробел.
`имена_столбцов` Список столбцов таблицы, которые выбираются запросом. Столбцы, не указанные в операторе, не будут включены в результат. Если необходимо вывести данные всех столбцов, можно использовать сокращенную запись. Звездочка (*) означает полный список столбцов.
FROM `имя_таблицы` Ключевое слово, которое должно присутствовать в каждом запросе. После него через пробел указывается имя таблицы, являющейся источником данных.
Код в скобках является не обязательным в операторе SELECT. Он необходим для более точного определения запроса.
Также необходимо сказать, что SQL код является регистронезависимым. Это означает, что запись SELECT можно написать как select. СУБД не отличит эти две записи, однако советуют все операторы SQL писать прописными буквами, чтобы его легко можно было отличить от другого кода

1. Узнать количество записей в таблице;
SELECT count(*) FROM имя_таблицы.

2. Выбрать все записи из таблицы;

SELECT * FROM имя_таблицы.

Выбрать записи, соответствующие данным столбцам
SELECT name, last_name, age from person

10. Выбрать все пары, которые встречаются, name - last_name из таблицы:
SELECT name, last_name FROM person.

Вывод подмножества данных

По мере увеличения таблиц возникает необходимость вывода только подмножества данных. Этого можно добиться с помощью предложения **LIMIT**.
Например, чтобы вывести из таблицы имена только первых пяти сотрудников, используется оператор LIMIT с аргументом равным 5.
SELECT name, last_name from person LIMIT 5

3. Выбрать несколько записей из таблицы, например 5. Это полезно, когда требуется узнать, как приблизительно выглядят данные в таблице;
SELECT * FROM имя_таблицы LIMIT 5.

Извлечение подмножеств

LIMIT можно использовать также для *извлечения подмножества* данных, используя дополнительные аргументы.
Общая форма оператора **LIMIT** имеет следующий вид:
SELECT (что-нибудь) **from** таблица **LIMIT** начальная строка, извлекаемое число записей

SELECT name, last_name from person LIMIT 6,3

Будут извлечены три строки, начиная с седьмой.

Запросы с условиями

7. Выбрать все записи с person, где поле name равняется Anna;
SELECT * FROM person WHERE name='Anna'.

Как быть, если надо вывести данные о сотрудниках, имя которых начинается с буквы В?
Язык *SQL* позволяет выполнить *поиск* строковых данных *по* шаблону. Для этого в предложении where используется оператор **LIKE** следующим образом.
select name, last_name from person where name LIKE "B%"

8. Выбрать все записи с person, где поле name начинается с An
SELECT * FROM person WHERE name LIKE 'An%';

Можно видеть, что здесь в условии вместо знака равенства используется LIKE и знак процента в шаблоне.
Знак % действует как символ-заместитель Он заменяет собой любую последовательность символов.
Таким образом "B%" обозначает все строки, которые начинаются с буквы В.
Аналогично "%B"выбирает строки, которые заканчиваются символом В, а "%B%" строки, которые содержат букву В.
Давайте выведем, например, всех сотрудников, которые имеют в фамилии строку "про".
select name, last_name from person where last_name like '%про%'

Выбрать все записи с person, где поле name заканчивается на na:
SELECT * FROM person WHERE name LIKE '%na'

Упорядочивание данных

Рассмотрим вопрос о том, как можно изменить *порядок вывода данных*, извлеченных из таблиц MySQL, используя предложение **ORDER BY** оператора **SELECT**.

Извлекаемые до сих пор данные всегда выводились в том порядке, в котором они были сохранены в таблице. В действительности *SQL* позволяет сортировать извлеченные данные с помощью предложения **ORDER BY**. *Это предложение требует имя столбца, на основе которого будут сортироваться данные.*

Давайте посмотрим, как можно вывести имена сотрудников с упорядоченными *по* алфавиту фамилиями сотрудников (в возрастающем порядке).

```
SELECT name, last_name  from person ORDER BY last_name;
```

А вот так сотрудников можно отсортировать *по* возрасту.

```
SELECT name, last_name, age from person ORDER BY age;
```

Предложение **ORDER BY** может сортировать в возрастающем порядке (**ASCENDING** или **ASC**) или в убывающем порядке (**DESCENDING** или **DESC**) в зависимости от указанного аргумента.

Чтобы вывести *список* сотрудников в убывающем порядке, можно использовать следующий оператор.

```
SELECT  name, last_name from person ORDER by last_name DESC
```

Примечание: Возрастающий порядок (ASC) используется *по* умолчанию.

9. Выбрать все записи с person, упорядоченные по number, где поле name
аканчивается на na:

```
SELECT * FROM person WHERE name LIKE '%na' ORDER BY number.
```

4. Выбрать все записи с person, отсортированные в порядке
возрастания номера number:

```
SELECT * FROM person ORDER BY number.
```

5. Выбрать все записи с person, отсортированные в порядке
убывания номера number;

```
SELECT * FROM person ORDER BY number DESC.
```

6. Выбрать немного (12) записей с person, отсортированные в
порядке возрастания номера number;

```
SELECT * FROM person ORDER BY number LIMIT 12.
```

Можно соединить оператор **LIMIT** с оператором **ORDER BY**. Таким образом, следующий оператор выведет четверых самых молодых сотрудников компании.

```
SELECT name, last_name, age  from person  ORDER BY age LIMIT 4
```

Аналогично можно вывести двух самых старых сотрудников

```
SELECT name, last_name, age from person ORDER BY age DESC LIMIT 2
```

Выбрать последнюю запись из таблицы

```
SELECT * FROM person ORDER BY number DESC LIMIT 1
```

Оператор вставки данных - INSERT

INSERT INTO имя_таблицы (поле1, поле2) VALUES(_”значение1”,”значение2”)

Пример

Добавление информации в таблицу **person**

```
INSERT INTO person(number,name, last_name, age) VALUES (12,'Ольга','Зайцева',25)
```

Оператор изменения данных – UPDATE

UPDATE таблица **SET** поле1=‘значение1‘, поле2=‘значение2‘ **WHERE** условие

Пример

UPDATE person SET last_name ='Егоров', **name**='Егор' **WHERE number** =3

Оператор удаления данных – DELETE

DELETE FROM таблица **WHERE** условие

Пример удаление строки с номером 1
DELETE FROM person WHERE number =1

создание базы данных

Оператор CREATE TABLE создает таблицу с заданным именем в текущей базе данных. Правила для допустимых имен таблицы приведены в документации. Если нет активной текущей базы данных или указанная таблица уже существует, то возникает ошибка выполнения команды

CREATE TABLE [IF NOT EXISTS] имя_таблицы [(определение_столбца,...)] [опции_таблицы] [select_выражение]

Пример 2

CREATE TABLE IF NOT EXISTS person (`number` int(10) unsigned NOT NULL auto_increment, `name` text, ` last_name` text, ` age` text, PRIMARY KEY (`number`))

number	name	last_name	age
1	Anna	Kryku	20

В выражении определение_столбца перечисляют, какие столбцы должны быть созданы в таблице. Каждый столбец таблицы может быть пустым (NULL), иметь значение по умолчанию, являться ключом или автоинкрементом. Кроме того, для каждого столбца обязательно указывается тип данных, которые будут в нем храниться. Если не указывается ни NULL, ни NOT NULL, то столбец интерпретируется так, как будто указано NULL. Если поле помечают как автоинкремент (AUTO_INCREMENT), то его значение автоматически увеличивается на единицу каждый раз, когда происходит добавление данных в таблицу и в это поле записывается пустое значение (NULL, т.е. ничего не записывается) или 0. Автоинкремент в таблице может быть только один, и при этом он обязательно должен быть проиндексирован.
Последовательность AUTO_INCREMENT начинается с 1. Наличие автоинкремента является одной из особенностей MySQL.

Тип столбца (тип в выражении определение_столбца) может быть одним из следующих:
- целый: INT[(length)] [UNSIGNED] [ZEROFILL]
- действительный: REAL[(length,decimals)] [UNSIGNED] [ZEROFILL]
- символьный: CHAR(length) [BINARY] и VARCHAR(length) [BINARY]
- дата и время: DATE и TIME
- для работы с большими объектами: BLOB
- текстовый: TEXT

перечислимое множество: ENUM(value1,value2,value3,...) и SET(value1,value2,value3,...)

Цель работы:

Научиться создавать таблицы в СУБД MySQL, писать запросы к БД средствами php

Рабочее задание

1. Нарисовать макет таблицы, соответствующей вашему варианту. Дать названия полям таблицы (используя латинские буквы, без пробелов!!!). Указать тип данных для каждого поля.

Пример:

number	name	last_name	age
1	Anna	Kryku	20

Название таблицы: **person**
Названия и тип полей:
number (int(7) PRI auto_increment), - номер пользователя
name (varchar(25)), - имя пользователя
last_name (varchar(50)), - фамилия пользователя
age (int(3)), - возраст пользователя

2. Заполнить таблицу, поместив в неё 10 записей. Данные в таблице могут быть любыми, **но придумывать их надо с условием, чтобы запросы, приведённые в ваших заданиях, могли быть выполнены**.

3. Написать запросы для данной таблицы, соответствующие своему варианту. Оформить текст запросов в word –документе.

4. Выполнить запросы к базе данных с помощью **PhpMyAdmin.**
Результаты ваших запросов в виде таблиц представить в word –документе.

5.Создать html-файл с формой для запросов к базе данных. Поля формы должны соответствовать полям таблицы.

6. Написать PHP-скрипт, обрабатывающий данную форму.
PHP-скрипт должен заполнять таблицу записями. С помощью формы и PHP-скрипта добавить 10 записей в таблицу.

7. Написать php-скрипт, извлекающий требуемую информацию из базы данных и формирующий ответы на поступившие запросы (на те запросы, которые указаны в вашем варианте)

8. С помощью PHP-скрипта выполнить все запросы к базе данных для своего варианта.

9. Оформить работу, поместив её результаты в word – документ. Показать преподавателю, получить свою отметку.

Приложение

Варианты заданий

#варианта	Задание
1	Создать БД с таблицей «Книги», в которой есть такие поля: Автор книги, название, год издания, цена, количество экземпляров, краткая аннотация. Выполнить запросы: 1. Узнать количество записей в таблице 2. Выбрать все записи, соответствующие столбцам «автор книги» и «количество экземпляров» 3. Вывести имена первых пяти авторов 4. Вывести четыре строки таблицы с названиями книг, начиная со второй строки
2	Создать БД с таблицей «Оптовая база» », в которой есть такие поля: Код товара, название товара, количество на складе, единица измерения, стоимость единицы товара, примечания - описание

	товара; Выполнить запросы: 1. вывести товары которых более 1000 и стоимость которых не превышает 100 2. Выбрать все записи, соответствующие столбцам «название товара» и «количество на складе» 3. Вывести три строки таблицы с названиями товаров, начиная со второй строки 4. Выбрать все записи таблицы, где названием товара является велосипед
3	Создать БД с таблицей «Производство» », в которой есть такие поля: Код изделия, название изделия, являются типичными (да / нет), примечание - для каких целей предназначен, годовой объем выпуска; Выполнить запросы: 1. Выбрать все записи, соответствующие столбцам «код изделия» и «название изделия» 2. Вывести две строки таблицы с названиями изделий, начиная со второй строки 3. Выбрать все записи таблицы, где изделия являются типичными 4. Вывести названия четырех «самых выпускаемых» изделий
4	Создать БД с таблицей «Сеть магазинов» », в которой есть такие поля: Номер, ФИО, адрес, телефон владельца магазина, размер вклада в магазин, номер регистрации, дата регистрации; Выполнить запросы: 1. Выбрать все записи, соответствующие столбцам «ФИО» и «адрес» 2. Вывести четыре строки таблицы с адресами, начиная со второй строки 3. Выбрать все записи таблицы, где владельцем магазина является Иванов И. И. 4. Вывести имена четверых «самых богатых» владельцев
5	Создать БД с таблицей «Деканат» », в которой есть такие поля: Код группы, название группы, курс, количество студентов, общий объем часов; Выполнить запросы: 1. Вывести четыре строки таблицы с названиями групп, начиная со второй строки 2. Выбрать все записи таблицы, где количество студентов равно 30 3. Выбрать все коды групп, названия которых начинается на букву А; 4. Вывести названия четверых «самых загруженных» групп студентов
6	Создать БД с таблицей «Поликлиника» », в которой есть такие поля: Номер, фамилия, имя, отчество, дата рождения пациента, социальный статус, текущее состояние; Выполнить запросы: 1. Выбрать все записи, соответствующие столбцам «фамилия» и «имя» 2. Вывести четыре строки таблицы с именами пациентов, начиная со второй строки 3. Выбрать имена пациентов, фамилия которых начинается на букву А; 4. Вывести имена четверых «самых молодых» пациентов
7	Создать БД с таблицей «Телефонная станция» », в которой есть такие поля: Номер абонента, фамилия абонента, адрес, код района (1,2,3 ...) наличие блокиратора (да / нет), примечание;

	Выполнить запросы: 1. Выбрать все записи, соответствующие столбцам «фамилия абонента» и «адрес» 2. Вывести четыре строки таблицы с фамилиями, начиная со второй строки 3. Вывести фамилии двух первых абонентов у которых есть блокиратор, упорядоченные по коду района 4. Вывести имена четверых абонентов, имеющих блокиратор
8	Создать БД с таблицей «Спорт» », в которой есть такие поля: ФИО спортсмена, год рождения, название команды, спортивный разряд; Выполнить запросы: 1. Выбрать все записи, соответствующие столбцам «ФИО» и «название команды» 2. Вывести ФИО первых пяти спортсменов 3. Вывести имена двух первых спортсменов, имеющих 2-й разряд, упорядоченные по году рождения 4. Вывести имена четверых «самых молодых» спортсменов
9	Создать БД с таблицей «Сельскохозяйственные работы» », в которой есть такие поля: Название сельскохозяйственного предприятия, дата регистрации, вид собственности, число работников, основной вид продукции, является передовым в освоении новой технологии, прибыль, примечание; Выполнить запросы: 1. Выбрать все записи, соответствующие столбцам «Название сельскохозяйственного предприятия» и «дата регистрации» 2. Вывести имена первых пяти предприятий 3. Вывести два первых предприятия, производящих картофель, упорядоченные по дате регистрации 4. Вывести названия четырех «самых молодых» предприятий
10	Создать БД с таблицей «География» », в которой есть такие поля: Название страны, регион, столица, площадь территории, является ли страна развитой в экономическом отношении (да / нет); Выполнить запросы: 1. Узнать количество записей в таблице 2. Выбрать все записи, соответствующие столбцам «название страны» и «столица» 3. Вывести две первые развитые страны, упорядоченные по площади территории 4. Вывести названия четырёх самых больших стран
11	Создать БД с таблицей «Оптовая база» », в которой есть такие поля: Код товара, название товара, количество на складе, единица измерения, стоимость единицы товара, примечания - описание товара; Выполнить запросы: • Добавить товар с кодом 11111; • поменять код в последней записи с 11111 на 00000; • удалить товары стоимость которых превышает 1000 р; • вывести товары которых более 1000 и стоимость которых не превышает 100
12	Создать БД с таблицей «Производство» », в которой есть такие поля: Код изделия, название изделия или являются типичными (да / нет),

	примечание - для каких целей предназначен, годовой объем выпуска; Выполнить запросы: • Добавить типовое изделие с кодом 123; • Поменять на последнем код с 123 на 124; • удалить все типичные устройства; • вывести изделия годовой объем выпуска которых превышает 10000 экземпляров;
13	Создать БД с таблицей «Сеть магазинов» », в которой есть такие поля: Номер, ФИО, адрес, телефон владельца магазина, размер вклада в магазин, номер регистрации, дата регистрации; Выполнить запросы: • Добавить новый магазин с номером регистрации 111; • Поменять телефон владельца магазина в последней записи; • удалить магазины с датой регистрации не текущего года; • вывести магазин с номером 5;
14	Создать БД с таблицей «Деканат» », в которой есть такие поля: Код группы, курс, количество студентов, общий объем часов; Выполнить запросы: • Добавить группу с кодом 333; • поменять в добавленной записи код 333 на 444; • удалить все группы третьего курса; • вывести группы общий объем часов которых превышает 1000;
15	Создать БД с таблицей «Поликлиника» », в которой есть такие поля: Номер, фамилия, имя, отчество, дата рождения пациента, социальный статус, текущее состояние; Выполнить запросы: • Добавить нового пациента 2003 года рождения; • Поменять в добавленной записи социальный статус; • удалить всех пациентов имеющих удовлетворительное состояние; • вывести пациентов родившихся после 2000го года;
16	Создать БД с таблицей «Телефонная станция» », в которой есть такие поля: Номер абонента, фамилия абонента, адрес, код района (1,2,3 ...) наличие блокиратора (да / нет), примечание; Выполнить запросы: • добавить нового абонента по фамилии Петров с кодом района 3; • Поменять на абонента Петрова код района на 1; • удалить всех абонентов у которых есть блокиратор; • вывести всех абонентов с кодом района 3;
17	Создать БД с таблицей «Спорт» », в которой есть такие поля: ФИО спортсмена, год рождения, название команды, спортивный разряд; Выполнить запросы: • Добавить спортсмена по плаванию Иванова из «Сборной России» • Изменить в добавленной записи название команды; • удалить всех спортсменов 1995 года рождения; • вывести спортсменов с высшим разрядом;
18	Создать БД с таблицей «Сельскохозяйственные работы» », в которой есть такие поля: Название сельскохозяйственного предприятия, дата регистрации, вид собственности, число работников, основной вид продукции, является передовым в освоении новой технологии, прибыль, примечание; Выполнить запросы: • добавить сельскохозяйственное предприятие, зарегистрировалось 1.03.2012; • изменить в добавленной записи дату с 1.03.2012 на 3.03.2012;

	• Удалить все предприятия, количество работников, в которых не превышает 20; • вывести предприятия зарегистрировались ранее 10.05.2012;
19	Создать БД с таблицей «Городской транспорт» », в которой есть такие поля: Вид транспорта, номер маршрута, количество остановок в пути, количество машин на маршруте, количество пассажиров в день, стоимость проезда; Выполнить запросы: ˅ добавить маршрут с номером 111; ˅ зменить в последней записи номер с 111 на 121; ˅ удалить все маршруты, количество пассажиров в которых не превышает 200чел/день; ˅ вывести маршруты в которых стоимость проезда 1.50;
20	Создать БД с таблицей «География» », в которой есть такие поля: Название страны, регион, столица, площадь территории, является ли страна развитой в экономическом отношении (да / нет); Выполнить запросы: • Добавить страну - Данию площадью 43000 км2; • поменять в добавленной записи площать с 43000км2 на 43093км2; • удалить все страны в которых площадь меньше 50000км2; • вывести список развитых стран;
21	Создать БД с таблицей «Домоуправление» », в которой есть такие поля: Номер квартиры, номер дома, число жителей, площадь; Выполнить запросы: • Добавить дом № 157 с квартирой № 1; • поменять в добавленной записи номер квартиры на 5; • удалить дом № 157; • вывести список квартир, число жителей в которых превышает 3 чел;
22	Создать БД с таблицей «Аэропорт» », в которой есть такие поля: Номер самолета, тип, число мест, скорость полета (число маха) Выполнить запросы: • Добавить самолет с номером 171; • в добавленной записи поменять номер 171 на 132; • удалить самолеты в которых число мест меньше 180; • вывести список самолетов число маха которых не более 0,7;
23	Создать БД с таблицей «Персональные ЭВМ» », в которой есть такие поля: Тип процессора, тактовая частота, объем ОЗУ, объем жесткого диска, дата выпуска ПЭВМ; Выполнить запросы: • Добавить процессор с датой выпуска 1.07.2010; • в добавленной записи поменять 2010 на 2011; • удалить процессоры серии Intel: • вывести ЭВМ в которых объем жесткого диска превышает 300Gb;
24	Создать БД с таблицей «Личные данные о студентах» », в которой есть такие поля: ФИО студента, курс, факультет, специальность, дата рождения студента, семейное положение, сведения о семье; Выполнить запросы: • добавить студента 2го курса; • изменить в добавленной записи 2й курс на 3й; • удалить всех студентов третьего курса; • вывести всех студентов, которые родились после 1990 года;
25	Создать БД с таблицей «Шахматы» », в которой есть такие поля: Фамилия спортсмена, дата рождения, страна, спортивный разряд, участвовал в борьбе за звание чемпиона мира (да / нет), рейтинг,

	примечание; Выполнить запросы: • добавить спортсмена из Украины; • поменять в добавленной записи страну Украину на Россию; • удалить всех спортсменов 1993 года рождения; • вывести всех спортсменов, которые принимали участие в борьбе за звание чемпиона мира;
26	Создать БД с таблицей «Программные продукты» », в которой есть такие поля: Название продукта, версия, тип, фирма, дата выпуска, прикладная область, стоимость лицензии; Выполнить запросы: • добавить программный продукт фирмы «Intel»; • поменять в добавленной записи фирму «Intel» на «Apple» • удалить все продукты, которые были выпущены ранее 2010г года; • вывести продукты, стоимость лицензии которых превышает 300 $;
27	Создать БД с таблицей «Изучение студентами дисциплин по выбору» », в которой есть такие поля: Фамилия студента, номер зачетки, наименование дисциплины, количество лекционных часов, семинарских и лабораторных занятий. Выполнить запросы: • добавить студента Карповича Ивана изучает дисциплину «ООП»; • поменять в добавленной записи имя Иван на Василий; • удалить всех студентов изучают «ООП»; • вывести дисциплины количество лекционных часов на которых превышает 100 часов;

Контрольные вопросы

1. Как подключиться к базе данных MySQL с помощью PHP?
2. Как выполнить SQL-запрос к базе данных MySQL с использованием PHP?
3. Как получить результаты SQL-запроса в виде массива ассоциативных массивов с помощью PHP?
4. Как выполнить вставку данных в таблицу MySQL с помощью PHP?
5. Как обновить данные в таблице MySQL с использованием PHP?
6. Как удалить данные из таблицы MySQL с помощью PHP?
7. Как обработать ошибки, возникшие при работе с базой данных MySQL, с использованием PHP?
8. Как обезопасить SQL-запросы в PHP с помощью подготовленных выражений?
9. Как установить соединение с базой данных MySQL с использованием SSL-шифрования в PHP?
10. Как создать резервную копию базы данных MySQL с помощью PHP?

Практическая работа №15

Разработка системы авторизации и регистрации на PHP

Теоретическая часть

Теоретический материал, необходимый для выполнения практической работе можно найти в лекции, по ссылке https://disk.yandex.ru/i/9EVeyDKzmUforA

Цель работы:

Научиться разрабатывать систему авторизации и регистрации на веб сайте средствами php

Рабочее задание

Выполнить инструкции приведенные ниже

Порядок выполнения работы

Настройка базы данных:

Для начала нам надо создать и настроить базу данных, для этого заходим в PhpMyAdmin.

Создание базы данных:

Создаём базу данных, называем её **user-login** и выбираем кодировку **utf8_general_ci**.

Нажимаем кнопку, создать БД

Настройка БД

Создание и настройка таблицы в БД:

Называем таблицу **users**, и настраиваем её, я не буду объяснять что каждая настройка значит, так как эта статья не о том как работать с БД, а просто покажу на скриншоте, после нажимаем сохранить.

База данных у нас готова, теперь надо подключится к ней.

Подключение БД к PHP:

Для этого создаём файлы connect.php, index.php и checkin.php. Сначала мы в connect.php подключаемся к самой БД, для этого пишем код ниже.

PHP

```
<?php
$server = 'localhost'; // Имя или адрес сервера
$user = 'root'; // Имя пользователя БД
$password = ''; // Пароль пользователя
$db = 'authorization-system'; // Название БД
$db = mysqli_connect($server, $user, $password, $db); // Подключение
// Проверка на подключение
if (!$db) {
    // Если проверку не прошло, то выводится надпись ошибки и заканчивается работа скрипта
    echo "Не удается подключиться к серверу базы данных!";
    exit;
}

```

Подключаем connect.php к index.php

Default

```
// Подключение БД
require_once 'connect.php';
```

Проверяем скрипт, для этого запускаем программу.

После того как проверили на работа способность, можете убрать вывод надписи «подключение к базе данных прошло успешно».

Создаём регистрацию на PHP:

Для этого пишем скрипт который будет ниже:

PHP

```php
// Проверяем нажата ли кнопка отправки формы
if (isset($_REQUEST['doGo'])) {

    // Все последующие проверки, проверяют форму и выводят ошибку
    // Проверка на совпадение паролей
    if ($_REQUEST['pass'] !== $_REQUEST['pass_rep']) {
        $error = 'Пароль не совпадает';
    }

    // Проверка есть ли вообще повторный пароль
    if (!$_REQUEST['pass_rep']) {
        $error = 'Введите повторный пароль';
    }

    // Проверка есть ли пароль
    if (!$_REQUEST['pass']) {
        $error = 'Введите пароль';
    }
    // Проверка есть ли email
    if (!$_REQUEST['email']) {
        $error = 'Введите email';
    }
    // Проверка есть ли логин
    if (!$_REQUEST['login']) {
        $error = 'Введите login';
    }
    // Если ошибок нет, то происходит регистрация
    if (!$error) {
        $login = $_REQUEST['login'];
        $email = $_REQUEST['email'];
        // Пароль хешируется
        $pass = password_hash($_REQUEST['pass'], PASSWORD_DEFAULT);
        // Если день рождения не был указан, то будет самый последний год из доступных
        $DOB = $_REQUEST['year_of_birth'];

        // Добавление пользователя
        mysqli_query($db, "INSERT INTO `users` (`login`, `email`, `password`, `DOB`) VALUES ('" . $login . "','" .
$email . "','" . $pass . "', '" . $DOB . "')");

        // Подтверждение что всё хорошо
        echo 'Регистрация прошла успешна';
    } else {
        // Если ошибка есть, то выводить её
        echo $error;
    }
}
?>
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Зарегистрироваться</title>
```

```
55
56
57 </head>
58 <body>
59    <form action="<?= $_SERVER['SCRIPT_NAME'] ?>">
60       <p>Логин: <input type="text" name="login" id=""> <samp style="color:red">*</samp></p>
61       <p>EMail: <input type="email" name="email" id=""><samp style="color:red">*</samp></p>
62       <p>Пароль: <input type="password" name="pass" id=""><samp style="color:red">*</samp></p>
63       <p>Повторите пароль: <input type="password" name="pass_rep" id=""><samp
64 style="color:red">*</samp></p>
65       <?php $year = date('Y'); ?>
66       Год рождения:
67       <select name="year_of_birth" id="">
68       <option value="">----</option>
69          <?php for ($i = $year - 14; $i > $year - 14 - 100; $i--) { ?>
70             <option value="<?= $i ?>"><?= $i ?></option>
71          <?php } ?>
72       </select>
73       <p><input type="submit" value="Зарегистрироваться" name="doGo"></p>
74    </form>
75 </body>
76 </html>
77
```

HTML я не стал рассказывать, так как, там всё понятно, да и вообще программист должен разберется в коде.

Создаём авторизацию на PHP:

Код будет очень сильно похож на файл регистрацию, но есть ряд не больших отличий.

PHP

```
1  // Проверка нажата ли кнопка отправки формы
2  if (isset($_REQUEST['doGo'])) {
3     // Последующий код проверяет вообще есть формы
4     // Проверяет логин
5     if (!$_REQUEST['login']) {
6        $error = 'Введите логин';
7     }
8     // Проверяет пароль
9     if (!$_REQUEST['pass']) {
10       $error = 'Введите пароль';
11    }
12    // Проверяет ошибки
13    if (!$error) {
14       $login = $_REQUEST['login'];
15       $pass = $_REQUEST['pass'];
16       // берёт из БД пароль и id пользователя
17       if ($result = mysqli_query($db, "SELECT `password`, `id` FROM `users` WHERE `login`='" . $login . "'")) {
18          while( $row = mysqli_fetch_assoc($result) ){
19             // Проверяет есть ли id
20             if ($row['id']) {
21                // если id есть, то он сравнивает пароли функцией password_verify
22                if (password_verify($pass, $row['password'])) {
23                   // Если функция возвращает true, то вы входите
24                   echo "Вы вошли";
25                   // скрипт больше не выполняется
26                   exit;
27                } else {
28                    // Если функция возвращает false, то выводит ошибку
29                    echo "Пароль не совпадает";
30                }
31             } else {
32                // Выводит ошибку если не нашли такой логин
33                echo "Ввели не верны логин";
```

```
34
35
36              }
37           }
38        }
39    } else {
40        // Выводит ошибки, если есть пустые поля формы
41        echo $error;
42    }
43 }
44 ?>
45 <!DOCTYPE html>
46 <html lang="ru">
47 <head>
48    <meta charset="UTF-8">
49    <meta name="viewport" content="width=device-width, initial-scale=1.0">
50    <meta http-equiv="X-UA-Compatible" content="ie=edge">
51    <title>Войти</title>
52 </head>
53 <body>
54    <form action="<?= $_SERVER['SCRIPT_NAME'] ?>">
55        <p>Логин: <input type="text" name="login" id=""></p>
56        <p>Пароль: <input type="password" name="pass" id="">
57        <p><input type="submit" value="Войти" name="doGo"></p>
58    </form>
59 </body>
60 </html>
61
62
```

Информацию о функции password_verify найдёте здесь, как видите код очень простой, но это ещё не всё, дальше вы сами напишите.

Что ещё надо сделать:

Ну, во-первых, авторизацию нельзя считать полноценной, пока данные о пользователи не будут сохранятся в браузере, для этого можно использовать куки, во-вторых, надо улучить условия принятия данных из формы, например, пропускать только определёныt символы в паролях и логинах

Контрольные вопросы

1. Как создать таблицу в базе данных MySQL для хранения информации о пользователях?
2. Как реализовать процесс регистрации пользователя на сайте с использованием PHP и MySQL?
3. Как проверить уникальность логина или электронной почты при регистрации пользователя?
4. Как создать процесс аутентификации пользователя на сайте с использованием PHP и MySQL?
5. Как хранить пароли пользователей в базе данных MySQL с безопасностью?
6. Как реализовать функцию восстановления пароля пользователя на сайте с помощью PHP и MySQL?
7. Как реализовать систему валидации данных при регистрации пользователя с помощью PHP и MySQL?
8. Как управлять сеансами пользователей для обеспечения безопасности с использованием PHP и MySQL?
9. Как создать страницы "Личный кабинет" и "Выход" после успешной авторизации пользователя?
10. Как добавить защиту от атаки подбора пароля (брутфорс) с помощью PHP и MySQL?

Практическая работа №16
Создание страницы отзывов на сайте

Теоретическая часть
Теоретический материал, необходимый для выполнения практической работе можно найти в лекции, по ссылке https://disk.yandex.ru/i/9EVeyDKzmUforA

Цель работы:
Научиться создавать страницу отзывов на веб сайте средствами php и MySQL

Рабочее задание
1.Отладить работу скрипта простой гостевой книги, расположенном в папке «скрипт гостевой книги».

2.Добавить готовый скрипт гостевой книги в «движок» Вашего сайта. Для этой цели взять динамический сайт «о цветах» .

3.Отформатировать отображение гостевой книги в соответствии с общим оформлением сайта.

4.Самостоятельно добавить проверку корректного ввода данных текстовых полей гостевой книги.

5. Файл **.htaccess** поместить в корневую директорию сайта.

6. На странице index.php внутри контейнера <head> прописать следующую строчку

```
<meta http-equiv="content-type" content="text/html; charset=utf-8" />
```

Контрольные вопросы
1. Как создать таблицу в базе данных MySQL для хранения информации об отзывах?
2. Как реализовать форму отправки отзыва на веб-странице с использованием PHP и MySQL?
3. Как сохранить данные отзыва в базе данных MySQL с помощью PHP?
4. Как вывести список всех отзывов из базы данных на веб-страницу с использованием PHP и MySQL?
5. Как реализовать функцию сортировки отзывов по дате или рейтингу с помощью PHP и MySQL?
6. Как добавить возможность комментирования отзывов на веб-странице с использованием PHP и MySQL?
7. Как реализовать функцию редактирования и удаления отзыва на веб-странице с помощью PHP и MySQL?
8. Как добавить возможность оценки отзывов (лайк или дизлайк) на веб-странице с использованием PHP и MySQL?
9. Как осуществить поиск отзывов по ключевым словам в базе данных MySQL с помощью PHP?
10. Как обезопасить работу с отзывами, предотвращая атаки XSS и SQL-инъекции с использованием PHP и MySQL?

Практическая работа №17

по теме «Технология Ajax»

Теоретическая часть

Теоретический материал, необходимый для выполнения практической работе можно найти в лекции, по ссылке https://yadi.sk/i/qodneyDjinWJmA

Цель работы:

научиться создавать ajax-запросы

Рабочее задание

Задание 1

Реализовать tml-страница, содержащая два div-блока. При загрузке страницы в первом из div-блоков должен отображаться текст –«отправить запрос», а во втором div-блоке – «здесь будет ответ сервера»
Написать javascript – функцию, которая при клике мышкой по первому div-блоку, отправляет ajax-запрос на сервер и выводит во втором div-блоке ответ полученный от сервера в виде текста.

Задание 2

Реализовать следующий сценарий:
Пусть имеется html-страница, содержащая текстовое поле и кнопку
Написать javascript – функцию, которая при клике мышкой по кнопке, отправляет ajax-запрос на сервер и выводит в текстовом поле ответ полученный от сервера в виде текста.

Задание 3

Реализовать следующий сценарий:
Пусть имеется html-страница, содержащая гиперссылку и два div-блока
Написать javascript – функцию, которая при клике мышкой по ссылке, отправляет ajax-запрос на сервер и выводит в обоих div-блоках ответы от сервера в виде текста. Это должно быть два разных сообщения.

Задание 4

Реализовать следующий сценарий:
Пусть имеется html-страница, содержащая два текстовых поля и кнопку
Написать javascript – функцию, которая при клике мышкой по кнопке, отправляет ajax-запрос на сервер и выводит в текстовых полях ответы полученные от сервера в виде текста. Это должно быть два разных ответа.

Задание 5

Реализовать следующий сценарий:
Пусть имеется html-страница, содержащая изображение
Написать javascript – функцию, которая при клике мышкой по изображению, отправляет ajax-запрос на сервер и выводит ниже изображения ответ, полученный от сервера в виде текста.

Задание 6

Реализовать следующий сценарий:
Пусть имеется html-страница, содержащая текстовое поле и кнопку
Написать javascript – функцию, которая при клике мышкой по кнопке, отправляет ajax-запрос на сервер методом GET. При этом передаются 4 произвольных числа. php – обработчик на сервере вычисляет сумму этих чисел. Ответ полученный от сервера выводится в текстовое поле.

Задание 7

Реализовать следующий сценарий:
Пусть имеется html-страница, содержащая текстовое поле и кнопку
Написать javascript – функцию, которая при клике мышкой по кнопке, отправляет ajax-запрос на сервер методом POST. При этом передаются 3 произвольных числа. php – обработчик на сервере вычисляет произведение этих чисел. Ответ полученный от сервера выводится в текстовое поле.

Задание 8

Реализовать следующий сценарий:
Пусть имеется html-страница, содержащая два текстовых поля и кнопку
В одно из полей пользователь вводит произвольное число.

Написать javascript – функцию, которая при клике мышкой по кнопке, отправляет ajax-запрос на сервер методом GET. При этом передается число, введенное пользователем в текстовое поле. php – обработчик на сервере вычисляет квадрат этого числа. Ответ полученный от сервера выводится во второе текстовое поле.

Задание 9
Реализовать следующий сценарий:
Пусть имеется html-страница, содержащая три текстовых поля и кнопку
В текстовые поля пользователь вводит произвольные числа.
Написать javascript – функцию, которая при клике мышкой по кнопке, отправляет ajax-запрос на сервер методом POST. При этом передаются числа, введенные пользователем в текстовые поля. php – обработчик на сервере вычисляет сумму этих чисел. Ответ, полученный от сервера выводится в документ ниже.

Задание 10
Реализовать следующий сценарий:
Пусть имеется html-страница, содержащая два кнопку с надписью «узнать точное время»
Написать javascript – функцию, которая при клике мышкой по кнопке, отправляет ajax-запрос на сервер методом POST. Результатом запроса должно быть время на сервере. Результат выводится в документ ниже кнопки.
Подсказка: время в php выводит функция: echo date('H:i:s');

Задание 11
Реализовать следующий сценарий:
Пусть имеется html-страница, содержащая два кнопку с надписью «запустить часы»
Написать javascript – функцию, которая при клике мышкой по кнопке, отправляет ajax-запрос на сервер методом POST. Результатом запроса должно быть время на сервере. Результат выводится в документ ниже кнопки с интервалом 1 секунда. Получаются ajax-часы.
Материалы для выполнения практических заданий находятся в папке «**AJAX материалы к практическим занятиям**».

Контрольные вопросы

1. Что такое AJAX и для чего его используют в веб-разработке?
2. Как отправить AJAX-запрос с помощью JavaScript?
3. Как указать цель (URL) для AJAX-запроса?
4. Как передать данные в AJAX-запросе с использованием метода POST или GET?
5. Как обрабатывать ответ от сервера после AJAX-запроса?
6. Как обработать ошибку или неудачный AJAX-запрос?
7. Как реализовать асинхронную загрузку данных на страницу с использованием AJAX-запросов?
8. Как отправить файлы через AJAX-запрос с помощью FormData?
9. Как использовать AJAX-запросы для получения данных из базы данных MySQL?
10. Как осуществить безопасность AJAX-запросов, предотвращая атаки CSRF и XSS?

Практическая работа №18 по теме «Технология Ajax» (AJAX-формы)

Теоретическая часть

Теоретический материал, необходимый для выполнения практической работе можно найти в лекции, по ссылке https://yadi.sk/i/y5vBFMsjs_LtCA

Цель работы:

научиться проводить обработку пользовательских форм с использованием технологии AJAX

Рабочее задание

1) создать html-страницу, содержащую форму с четырьмя текстовыми полями(имя, фамилия, адрес, email) и кнопкой «отправить». Написать функцию javascript, осуществляющую следующий сценарий:при клике по кнопке «отправить»javascript-функция выполняет следующее:
а) производит проверку всех полей формы на наличие там каких либо данных

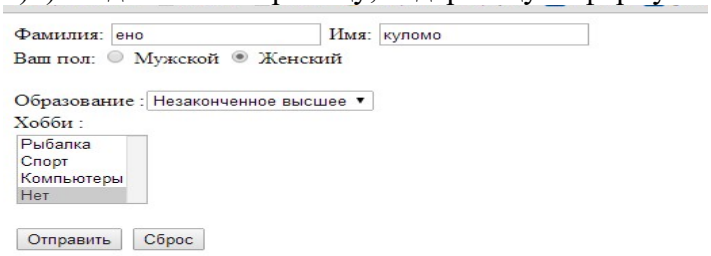
б)при наличии данных во всех полях формы - отправляет данные на сервер AJAX-ом, используя метод GET.
С сервера должно прийти сообщение, следующего вида:

Метод отправки данных: GET Вы отправили следующие данные: Фамилия: Иванов Имя: Иван Адрес: г. Москва ул. Ленина 20 Email: ivan@mail.ru

2)1) создать html-страницу, содержащую форму с тремя текстовыми полями(имя, email, сообщение) и кнопкой «отправить». Написать функцию javascript, осуществляющую следующий сценарий:при клике по кнопке «отправить» javascript-функция выполняет следующее:
а) производит проверку всех полей формы на наличие там каких либо данных
б)при наличии данных во всех полях формы - отправляет данные на сервер AJAX-ом, используя метод POST.
С сервера должно прийти сообщение, следующего вида:

Метод отправки данных: POST Вы отправили следующие данные: Имя: Иванов Иван Email: ivan@mail.ru Текст сообщения: Я пришел к тебе с приветом!

3)1) создать html-страницу, содержащую форму



Написать функцию javascript, осуществляющую следующий сценарий:при клике по кнопке «отправить» javascript-функция выполняет следующее:
а) производит проверку всех полей формы на наличие там каких либо данных
б)при наличии данных во всех полях формы - отправляет данные на сервер AJAX-ом, методомPOST**с использованием FormData**.
Скрипт обработчик на сервере принимает данные и отправляет письмо на адресadmin@mail.ru.
Содержание письма: Уважаемый admin! Пользователь «фамилия, имя» отправил вам свои данные «число.месяц. год»
С сервера должно прийти сообщение, следующего вида:

Данные получены! На адрес admin@mail.ru отправлено письмо от «фамилия имя»

4)1) создать html-страницу, содержащую форму как в задании № 3.
Написать функцию javascript, осуществляющую следующий сценарий:при клике по кнопке «отправить» javascript-функция выполняет следующее:
а) производит проверку всех полей формы на наличие там каких либо данных
б)при наличии данных во всех полях формы - отправляет данные на сервер AJAX-ом, методом POST**с использованием FormData** .
Скрипт обработчик на сервере принимает данные и записывает их в базу данных.
Примерный вид таблицы:

id	surname	name	sex	education	hobby

С сервера должно прийти сообщение, следующего вида:

Данные полученыи занесены в базу! id = surname= name = sex= education = hobby=

5) доработать php-скрипт из задания 4, таким образом, чтобы на страницу выводились последние 10 записей из базы данных.

Материалы для выполнения практических заданий находятся в папке «**AJAX материалы к практическим занятиям**».

Контрольные вопросы

1. Как отправить данные из формы с помощью AJAX-запроса?
2. Как обработать ответ от сервера после отправки формы с использованием AJAX?
3. Как осуществить валидацию данных формы перед отправкой с использованием AJAX?
4. Как реализовать динамическую проверку уникальности введенных данных формы с помощью AJAX?
5. Как использовать AJAX для подгрузки дополнительных полей формы, зависящих от выбранного значения?
6. Как отображать прогресс отправки формы с использованием AJAX?
7. Как обработать ошибки, возникающие при отправке формы через AJAX?
8. Как добавить функциональность автозаполнения полей формы с использованием AJAX и базы данных?
9. Как обрабатывать частично заполненные формы (частичное сохранение) при использовании AJAX?
10. Как реализовать функцию автоматического заполнения формы на основе предыдущих вводов пользователя с использованием AJAX?

Практическая работа №19 по теме «Технология Ajax»

Теоретическая часть

Теоретический материал, необходимый для выполнения практической работе можно найти в лекции, по ссылке https://yadi.sk/i/-Bjt8K_RmbjQZg

Цель работы: научиться создавать web-приложения с использованием технологии AJAX

Время выполнения: 6 часов

Рабочее задание:

1)Разработать произвольный html-шаблон сайта, состоящий из трех страниц (можно взять любой готовый в интернете)

2)В выбранный шаблон встроить следующие скрипты

а) регистрация и авторизация (с использованием ajax)

б) обработка контактной формы (с использованием ajax)

в) какой либо вариант гостевой книги с отображением комментариев (с использованием ajax)

г)реализовать возможность загрузки файлов на сайт пользователем (с использованием ajax)

Материалы для выполнения практических заданий находятся в папке «**AJAX материалы к практическим занятиям**».

Пример сайта со встроенными скриптами авторизации, регистрации, гостевой книги находится в папке «**oll**».

Контрольные вопросы

1. Что такое AJAX и какую роль она играет в разработке web-приложений?
2. Какие основные преимущества использования AJAX в web-приложениях?
3. Как отправить AJAX-запрос на сервер для получения данных для web-приложения?
4. Как обработать ответ от сервера после отправки AJAX-запроса и обновить содержимое страницы?
5. Как обрабатывать ошибки или неудачные AJAX-запросы в web-приложении?
6. Как осуществить асинхронную загрузку данных в web-приложении с помощью AJAX?
7. Как использовать AJAX для отправки данных формы без перезагрузки страницы?
8. Как обновлять содержимое определенных элементов страницы с использованием AJAX?
9. Как реализовать функциональность динамического поиска с помощью AJAX в web-приложении?
10. Как обращаться к удаленному API с использованием AJAX для получения данных в web-приложении?

Практическая работа №20

СТОИМОСТНАЯ ОЦЕНКА ПРОЕКТА

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

Порядок выполнения работы:

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

Теоретическая часть:

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:

1) Расчет трудоемкости реализации функций до и после внедрения ИС;
2) Расчет затрат на реализацию функций ИС;

3) Расчет затрат на проектирование и внедрение системы

4) 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, заключенных с подрядчиками и актов выполненных работ	15	20	300	0,75
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 "Ремонт/модернизация" БД	2	140	280	0,7
Внесение данных материальных ценностей в паспорт помещения и в БД	10	30	300	0,75
Ввод данных заявок от	0,2	20	4	0,01

сотрудников в БД				
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:

1. в Пенсионный фонд Российской Федерации уплачивается 22 процента;
2. в Фонд социального страхования Российской Федерации - 2,9 процента;

3. в Федеральный фонд обязательного медицинского страхования - 3,1 процента;

4. 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
1.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, $\Delta К$ = 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

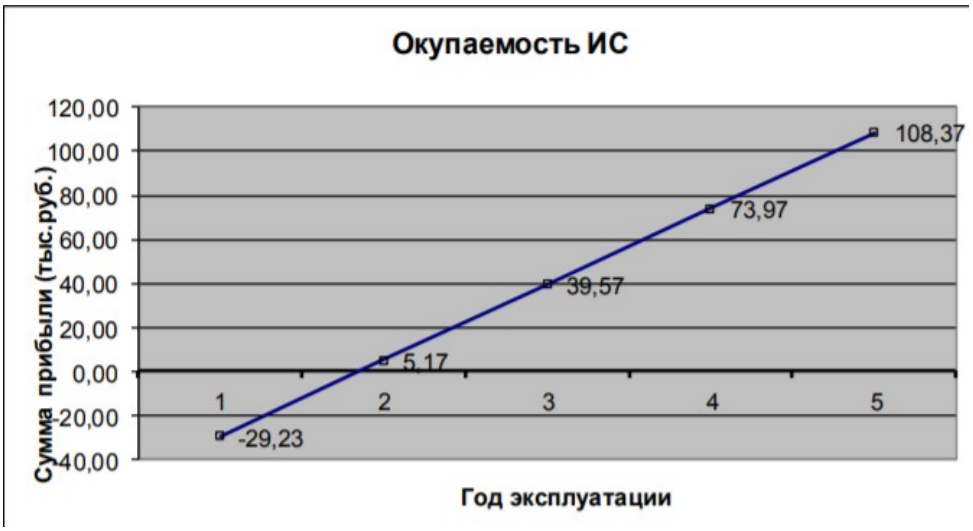


Рис.5.1 - График окупаемости затрат

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.

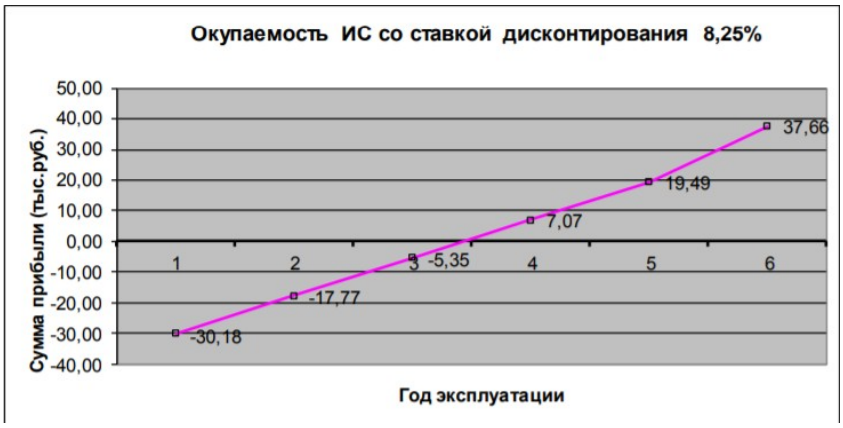


Рисунок 2 - График окупаемости ИС с учетом дисконтирования простым процентом

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.

Контрольные вопросы

1. Как определить стоимостные затраты на проект внедрения информационной системы (ИС)?
2. Какие факторы следует учесть при расчете стоимостей проекта внедрения ИС?
3. Как определить сроки окупаемости проекта внедрения ИС?
4. Какие методы расчета стоимости и сроков окупаемости проекта внедрения ИС существуют?
5. Каким образом можно учесть возможные риски и неопределенности при расчете стоимостей и сроков окупаемости проекта внедрения ИС?
6. Как влияют изменения в исходных данных, например, стоимость или объемы продаж, на расчеты стоимости и сроков окупаемости проекта внедрения ИС?
7. Как оценить эффективность проекта внедрения ИС на основе стоимостных показателей, таких как ROI (Return on Investment) или NPV (Net Present Value)?
8. Каким образом можно учесть дополнительные выгоды, такие как повышение эффективности работы или улучшение качества услуг, при расчете стоимостей и сроков окупаемости проекта внедрения ИС?
9. Какие инструменты и методики могут помочь в проведении стоимостной оценки проекта внедрения ИС и расчете сроков окупаемости?
10. Какие факторы следует учесть при прогнозировании будущих изменений стоимостей и сроков окупаемости внедрения ИС?

Практическая работа №21

ПОСТРОЕНИЕ ДИАГРАММ ПОТОКОВ ДАННЫХ И ГЕНЕРАЦИЯ КОДА

Цель работы: получение навыков построения диаграмм потоков данных.

Порядок выполнения работы

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

Теоретическая часть

Понятие диаграммы потоков данных. Элементы диаграммы потоков данных. Хранилищаданных. Потокиуправления.

Рабочее задание

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).

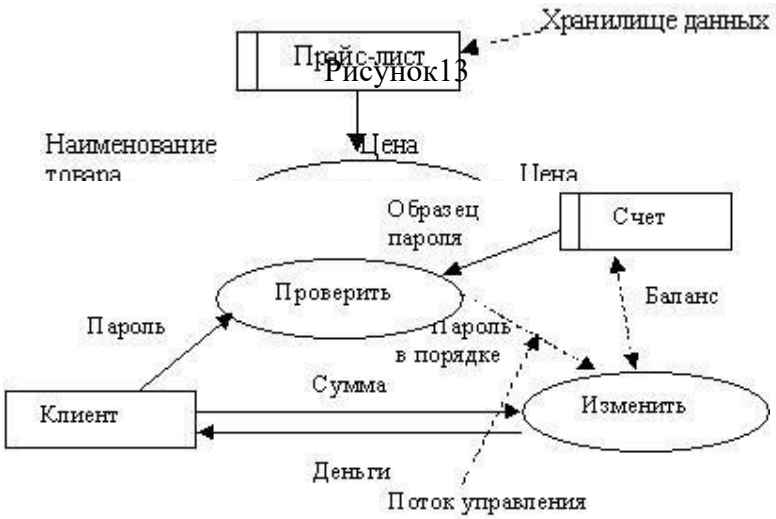


Рисунок14

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).

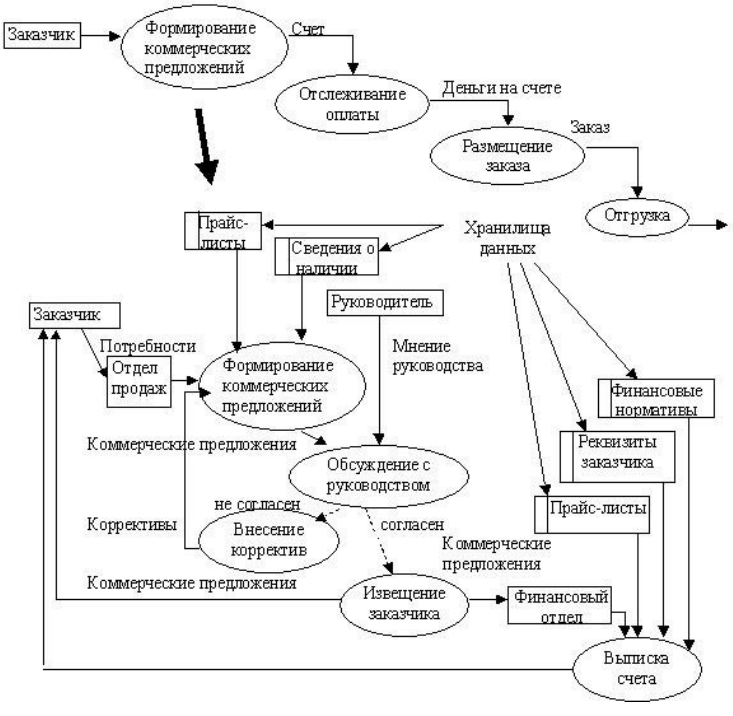


Рисунок 15

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

Задание № 4. Оформите отчет.

Контрольные вопросы

1. Что такое диаграмма потоков данных (DFD) и какую роль она играет в анализе и проектировании систем?
2. Какие основные компоненты включает в себя диаграмма потоков данных?
3. Каким образом определяются и представляются потоки данных на диаграмме?
4. Как обозначаются процессы или функции на диаграмме потоков данных?
5. Какие символы используются для обозначения хранилищ данных на диаграмме?
6. Как связываются компоненты на диаграмме потоков данных, чтобы показать потоки данных между ними?
7. Каким образом можно представить разные уровни детализации на диаграмме потоков данных?
8. Как использовать диаграммы потоков данных для выявления потенциальных проблем или улучшений в системе?
9. Каким образом диаграммы потоков данных могут быть использованы для документации и коммуникации с заинтересованными сторонами?
10. Какие инструменты или программное обеспечение могут помочь в построении диаграмм потоков данных?

Практическая работа №22

УСТАНОВКА И НАСТРОЙКА СИСТЕМЫ КОНТРОЛЯ ВЕРСИЙ С РАЗГРАНИЧЕНИЕМ РОЛЕЙ

Цель работы: получение навыков установки и настройки системы контроля версий.

Порядок выполнения работы

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

Время выполнения: 2 ч

Теоретическая часть

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.

Рабочее задание

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.

Контрольные вопросы

1. Что такое система контроля версий (Version Control System, VCS) и для чего она используется?
2. Какие основные виды систем контроля версий существуют и в чем их отличия?
3. Как установить систему контроля версий на свой компьютер?
4. Какие файлы и папки нужно исключить из контроля версий?
5. Как создать новый репозиторий в системе контроля версий?
6. Как настроить основные параметры системы контроля версий, такие как имя пользователя и адрес электронной почты?
7. Как добавить файлы в репозиторий и сделать первый коммит?
8. Как создать новую ветку (branch) в системе контроля версий и переключиться на нее?
9. Как работать с различными версиями файлов, отменять изменения и восстанавливать предыдущие версии?
10. Как сотрудничать с другими разработчиками и обмениваться изменениями с помощью системы контроля версий?

Практическая работа №23

ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ИНТЕРФЕЙСАПОЛЬЗОВАТЕЛЯ

Цель работы: получение навыков проектирования и разработки интерфейса пользователя.

Порядок выполнения работы:

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

Теоретическая часть

Понятие пользовательского интерфейса. Виды пользовательских интерфейсов.
Основные элементы пользовательского интерфейса. Требования к разработке пользовательского интерфейса.

Рабочее задание

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.

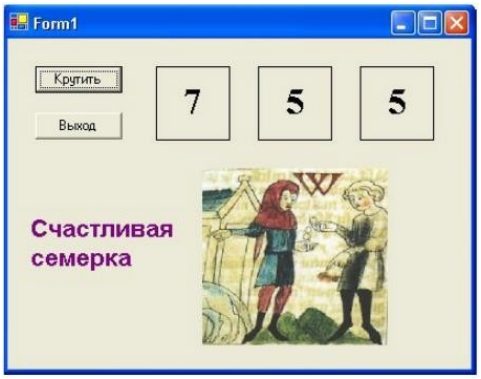
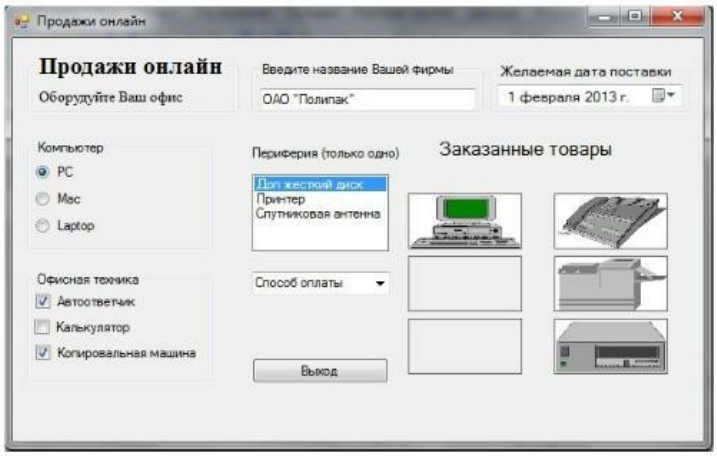


Рисунок 16

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».

Контрольные вопросы

1. Какие основные принципы проектирования интерфейса пользователя следует учитывать при разработке приложений на C#?
2. Как создать пользовательский интерфейс в C# с использованием Windows Forms или WPF?
3. Какие элементы управления (controls) доступны в C# для создания интерактивного интерфейса пользователя?
4. Каким образом можно настроить внешний вид элементов управления и стилей в C#?
5. Как обрабатывать события (events) интерфейса пользователя в C#?
6. Как обеспечить валидацию ввода данных в интерфейсе пользователя на C#?
7. Какой подход использовать для создания многопоточного интерфейса пользователя в C#?
8. Как интегрировать различные типы мультимедиа (например, изображения, звук, видео) в интерфейс пользователя на C#?
9. Каким образом можно улучшить навигацию и поток работы пользователей в интерфейсе C#?
10. Как организовать тестирование и отладку интерфейса пользователя в C# для обеспечения качества и удобства использования?

Практическая работа №24
РЕАЛИЗАЦИЯ АЛГОРИТМОВ ОБРАБОТКИ ЧИСЛОВЫХ ДАННЫХ. ОТЛАДКА ПРИЛОЖЕНИЯ

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

Порядок выполнения работы:

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

Время выполнения: 2 ч

Теоретическая часть

Элементы управления, используемые для обработки числовыхданных.

Рабочее задание

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.

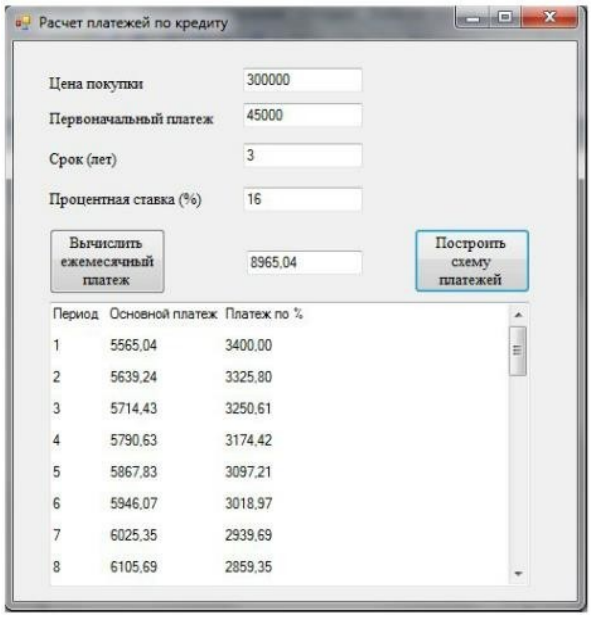


Рисунок 18

Контрольные вопросы

1. Какие методы и алгоритмы используются для сортировки числовых данных?
2. Что такое поиск в массиве и какие методы и алгоритмы используются для его реализации?
3. Какие алгоритмы применяются для поиска минимума и максимума в числовых данных?
4. Какие алгоритмы и методы используются для обработки числовых данных в формате матриц или таблиц?
5. Какие алгоритмы применяются для вычисления суммы или среднего значения числовых данных?
6. Как реализовать операции поиска, добавления, удаления и обновления элементов в динамической структуре данных, такой как список или дерево?
7. Какие алгоритмы используются для решения задач аппроксимации и интерполяции числовых данных?
8. Какие методы применяются для анализа и фильтрации временных рядов или сигналов?
9. Как реализовать алгоритмы поиска паттернов или совпадений в числовых данных?
10. Как использовать алгоритмы машинного обучения и искусственного интеллекта для анализа и обработки числовых данных?

Практическая работа №25

РЕАЛИЗАЦИЯ АЛГОРИТМОВ ПОИСКА. ОТЛАДКА ПРИЛОЖЕНИЯ

Цель работы: получение навыков реализации алгоритмов поиска данных, отладки приложений.

Порядок выполнения работы:

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

Теоретическая часть Алгоритмы поиска в тексте. Алгоритмы поиска в массивах.

Рабочее задание

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

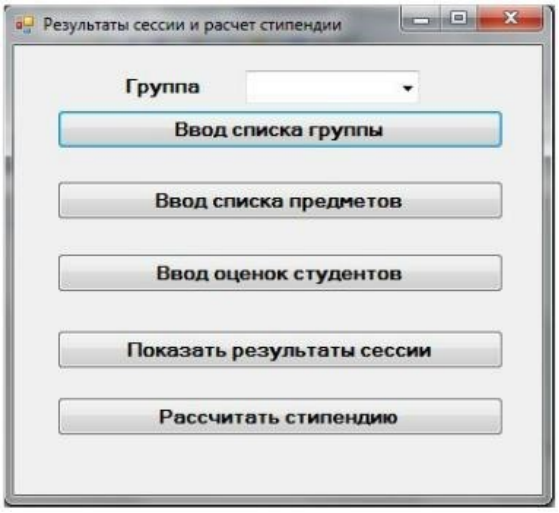


Рисунок 20

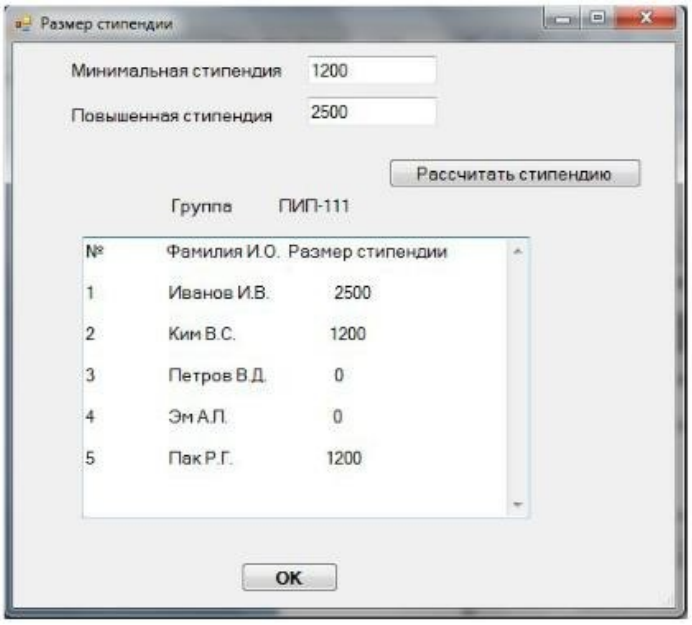


Рисунок21

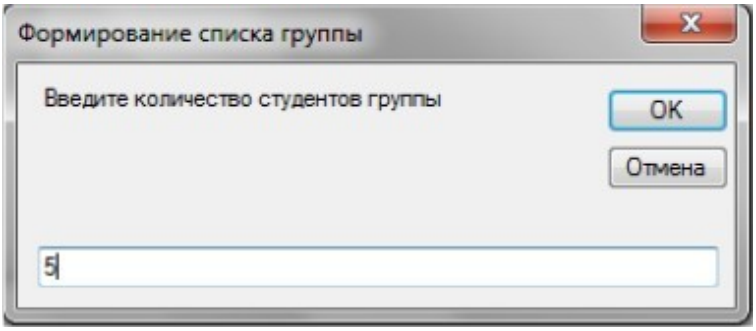


Рисунок22

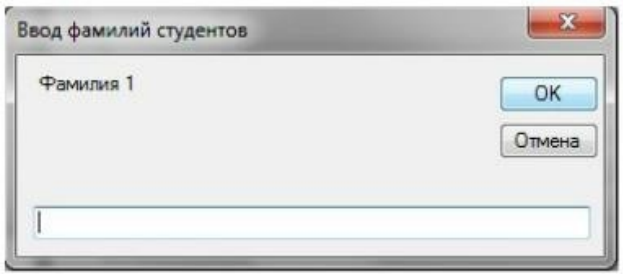


Рисунок2

Рисунок 24

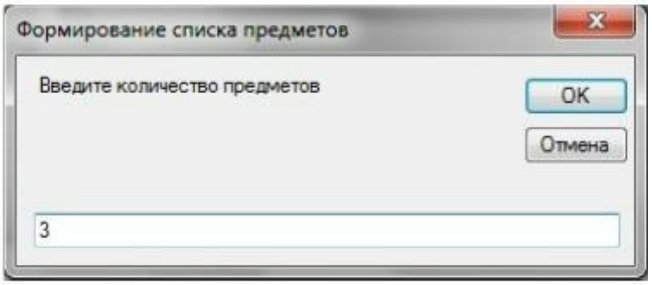


Рисунок25

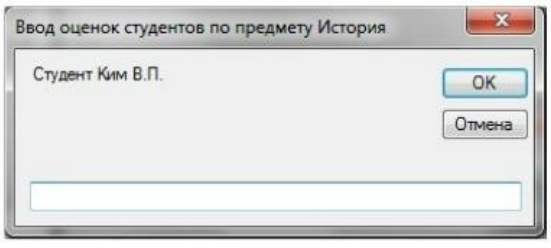


Рисунок26

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

Контрольные вопросы

1. Какие основные алгоритмы поиска используются для поиска данных в массиве или списке?

2. Как реализовать алгоритм бинарного поиска для быстрого поиска данных в отсортированном массиве?
3. Какой алгоритм использовать для поиска подстроки или шаблона в строке или тексте?
4. Какие алгоритмы могут быть использованы для поиска пути или оптимального пути в графе или сети?
5. Как реализовать алгоритм поиска наилучшего совпадения или наиболее подходящего элемента с использованием алгоритма наименьших квадратов (Least Squares)?
6. Какие алгоритмы можно использовать для поиска определенной структуры данных, такой как дерево или граф, в другой структуре данных?
7. Каким образом можно реализовать алгоритмы поиска с использованием хэш-функций и хэш-таблиц?
8. Какие алгоритмы можно применять для анализа и классификации данных с использованием машинного обучения или интеллектуальных алгоритмов?
9. Как реализовать алгоритмы вероятностного поиска для поиска оптимальных решений или перебора значений в большом пространстве поиска?
10. Какие алгоритмы можно использовать для поиска и фильтрации данных в режиме реального времени, например, в потоковых данных или онлайн-системах?

Практическая работа №26
РЕАЛИЗАЦИЯ ОБРАБОТКИ ТАБЛИЧНЫХ ДАННЫХ. ОТЛАДКА ПРИЛОЖЕНИЯ

Цель работы: получение навыков обработки табличных данных, отладки приложений.

Порядок выполнения работы:

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

Время выполнения: 2 ч

Теоретическая часть

Обработка табличных данных в приложениях.

Рабочее задание

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.

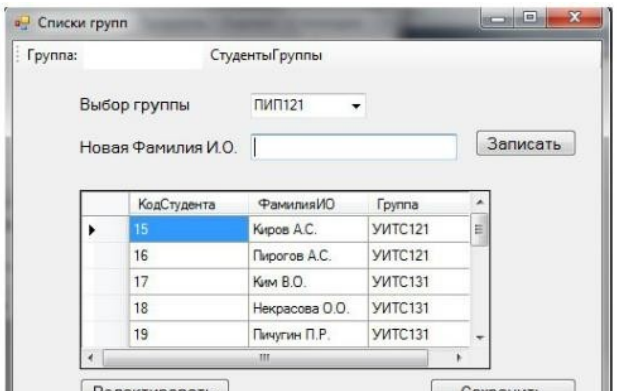


Рисунок 27

Контрольные вопросы

1. Как создать и инициализировать таблицу данных в C#?
2. Как добавить новые строки и столбцы в таблицу данных в C#?
3. Каким образом можно обращаться к данным в таблице и изменять их значения в C#?
4. Как отфильтровать и сортировать данные в таблице данных в C#?
5. Как реализовать поиск и выборку данных по определенным критериям или условиям в таблице данных в C#?
6. Каким образом можно объединять, разделять или преобразовывать таблицы данных в C#?
7. Как выполнять агрегацию и вычисление статистических значений (например, сумма, среднее, максимум) в таблице данных в C#?
8. Как реализовать связь и отношения между таблицами данных в C#?
9. Каким образом можно экспортировать и импортировать таблицы данных из различных форматов (например, CSV, Excel) в C#?
10. Как обрабатывать ошибки и обеспечить контроль целостности данных при работе с таблицами данных в C#?

Практическая работа №27

РАЗРАБОТКА И ОТЛАДКА ГЕНЕРАТОРА СЛУЧАЙНЫХ СИМВОЛОВ

Цель работы: получение навыков разработки и отладки генератора случайных символов.

Порядок выполнения работы

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

Теоретическая часть

Понятие генератора случайных символов. Управление генератором случайных символов.

Рабочее задание

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>usingnamespacestd;
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.

Контрольные вопросы

1. Как создать генератор случайных символов в выбранном языке программирования?
2. Как сгенерировать случайный символ из определенного набора символов?
3. Как убедиться, что сгенерированный символ является уникальным или не повторяется в предыдущих генерациях?
4. Как создать генератор случайной строки из символов определенной длины?
5. Как обеспечить настройку вероятности появления определенных символов при генерации случайных символов?
6. Каким образом можно отлаживать генератор случайных символов и проверять правильность его работы?
7. Как создать функцию для генерации случайной последовательности символов заданной длины?
8. Каким образом можно обрабатывать возможные ошибки и исключения при генерации случайных символов?
9. Как проверить равномерность распределения случайных символов, чтобы избежать предвзятости или смещения в результате генерации?
10. Как реализовать генератор случайных символов с заданными параметрами, такими как длина строки, допустимые символы и особые условия (например, уникальность символов)?

Практическая работа №28

ИНТЕГРАЦИЯ МОДУЛЯ В ИНФОРМАЦИОННУЮ СИСТЕМУ

Цель работы: получение навыков интеграции модулей в информационную систему.

Порядок выполнения работы

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

Теоретическая часть

Понятие модуля.
Управление модулями.

Рабочее задание

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.

Контрольные вопросы

1. Каким образом можно интегрировать модули в существующую информационную систему?
2. Какие методы или протоколы можно использовать для связи и обмена данными между модулями в информационной системе?
3. Как реализовать механизм автоматического обнаружения и регистрации модулей при интеграции в информационную систему?
4. Каким образом можно обеспечить безопасность и аутентификацию при обмене данными между модулями в информационной системе?
5. Как реализовать механизм публикации и подписки на события между модулями в информационной системе?
6. Как управлять зависимостями и версиями модулей при их интеграции в информационную систему?
7. Каким образом можно мониторить и отслеживать работу интегрированных модулей в информационной системе?
8. Как реализовать механизм конфигурации и настройки параметров интегрированных модулей в информационной системе?
9. Как обеспечить масштабируемость и гибкость в интеграции модулей в информационную систему для возможности добавления новых модулей в будущем?
10. Как оценить и улучшить производительность и эффективность интеграции модулей в информационной системе?

Практическая работа №29
РАЗРАБОТКА ПРИЛОЖЕНИЙ ДЛЯ МОДЕЛИРОВАНИЯ ПРОЦЕССОВ И ЯВЛЕНИЙ. ОТЛАДКА ПРИЛОЖЕНИЯ

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

Порядок выполнения работы:

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

Теоретическая часть

Понятие модели.
Моделирование процессов и явлений.
Технологии моделирования процессов и явлений в приложениях.

Рабочее задание

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.

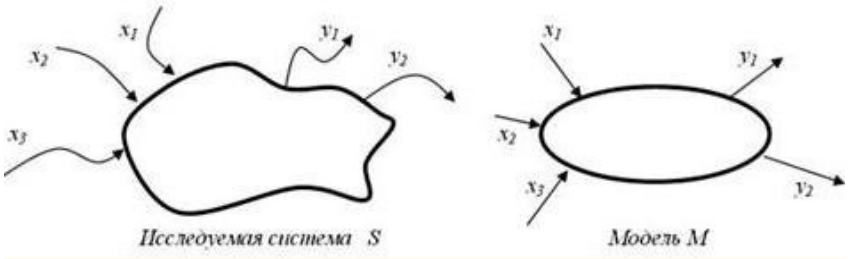


Рисунок 28
Если на вход M поступают сигналы из X и на входе появляются сигналы из Y, то задан закон, правило f функционирования модели, системы.
Классификацию моделей проводят по различным критериям.
Модель – статическая, если среди параметров описания модели нет (явно) временного параметра.
Модель – 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 (c), 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 о от x (например, k = dx): $x_{i+1} = x_i + ax_i - dx_i^2, x_0 = c$.

Контрольные вопросы

1. Как выбрать подходящую технологию для разработки приложения моделирования процессов и явлений?
2. Каким образом можно создать модель процесса или явления в приложении для

моделирования?
3. Как реализовать визуализацию моделирования процессов и явлений в приложении?
4. Каким образом можно симулировать временные параметры и динамику в приложении моделирования?
5. Как обработать входные данные и параметры для моделирования процессов и явлений в приложении?
6. Каким образом можно отлаживать и проверять правильность работы моделирования в приложении?
7. Как реализовать возможность настройки и изменения параметров моделирования в приложении?
8. Каким образом можно обрабатывать и представлять результаты моделирования процессов и явлений в приложении?
9. Как оценить эффективность и точность моделирования в приложении и провести его валидацию?
10. Как реализовать возможность взаимодействия и интеграции с другими системами и сервисами в приложении моделирования процессов и явлений?

Практическая работа №30
ПРОГРАММИРОВАНИЕ ОБМЕНА СООБЩЕНИЯМИ МЕЖДУ МОДУЛЯМИ

Цель работы: получение навыков программирования обмена сообщениями между модулями.

Порядок выполнения работы:

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

Теоретическая часть

Понятие и структура сообщения. Обмен сообщениями между модулями.

Рабочее задание

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

Контрольные вопросы

1. Каким образом можно реализовать обмен сообщениями между модулями при помощи технологии программирования?
2. Как выбрать подходящий протокол обмена сообщениями для взаимодействия между модулями?
3. Как реализовать механизм отправки и получения сообщений между модулями в программе?
4. Как обеспечить синхронизацию и параллельную обработку сообщений между модулями?
5. Как управлять очередью сообщений и приоритетами обработки взаимодействия между модулями?
6. Каким образом можно реализовать сериализацию и десериализацию сообщений для передачи данных между модулями?
7. Как обработать ошибки и исключения при обмене сообщениями между модулями?
8. Как реализовать механизм подписки и рассылки сообщений между модулями для возможности реакции на определенные события?
9. Каким образом можно обеспечить безопасность и аутентификацию при обмене сообщениями между модулями?
10. Как оценить производительность и эффективность обмена сообщениями между модулями в программе и провести его оптимизацию?

Практическая работа №31
ОРГАНИЗАЦИЯ ФАЙЛОВОГО ВВОДА-ВЫВОДА ДАННЫХ

Цель работы: получить практические навыки программирования задач ввода-вывода с использованием файлов.

Порядок выполнения работы:

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

Время выполнения: 2 ч

Теоретическая часть

Организация ввода и вывода. Файловая система

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

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

В языке С (С++) формат стандартных файлов ввода/вывода описан в заголовочном файле stdio.h. Имена стандартных файлов ввода/вывода для языка С (С++) представлены в табл. 7.1. В момент начала выполнения 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, связанные в основном с вводом/выводом объектов. Описание средств для

создания потоков в C++ представлено в заголовочном файле iostream.h. Когда начинает работать программа на C++, открываются потоки, приведенные в табл. 7.2.

Таблица 7.2

Потоки, определяемые в языке C++

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 файлов, с которыми логические файлы всегда связаны.

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

Основными действиями, производимыми над файлами, являются их открытие, обработка и закрытие. Обработка файлов может заключаться в считывании блока данных из потока в оперативную память, запись блока данных из оперативной памяти в поток, считывание определенной записи данных из потока, занесение определенной записи данных в поток. При этом необходимо помнить, что понятие файла в памяти ЭВМ не 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, содержащую параметры реализации потока, такие как адреса буферов, указатели позиций потока, маркеры ошибок потока и т.д.

При работе с дисковыми файлами в момент их открытия следует задать режим доступа, чтобы определить, к какому файлу осуществляется доступ: к текстовому или двоичному, а также способ доступа: чтение или запись. Все 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
r	Открыть файл только для чтения, модификации файла запрещены.
w	Создать новый файл только для записи. При попытке открыть таким способом существующий файл происходит перезапись файла. Чтение данных из файла запрещено.
a	Открыть файл для дозаписи. Если файла с указанным именем не существует, он будет создан.
r+	Открыть существующий файл для чтения и записи.
w+	Создать новый файл для чтения и записи.
a+	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 пути к нужному файлу. Это вызывает ошибку исполнения программы. Следует предвидеть подобные ситуации и проводить проверку возможности открытия файла. Такую проверку осуществить довольно легко, так как функция fopen() возвращает значение указателя в случае успешного его открытия и значение NULL, если доступ к файлу невозможен. Поэтому достаточно написать:

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

Текстовой файл состоит из последовательности символов, разбитой на строки путем использования управляющего символа \n. На диске текстовые файлы хранятся в виде сплошной последовательности символов, и их деление на строки становится заметным лишь в момент вывода на экран или печать, так как именно при выводе управляющие символы начинают выполнять свои функции. Текстовые файлы легко переносятся с одного типа компьютера на другой лишь в случаях, когда они содержат только символы, принадлежащие стандартному набору 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(), но имеют 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 обращаться к нему вторично, закрыли связанный с ним файл и освободили, таким образом, указатель.

В приведенном примере имя файла было записано непосредственно в операторе открытия файла. Но можно 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 файл лишь 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 стандартное 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 символов в 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 таблицы 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 оператор записи в файл, фактическая запись может и не происходить, поскольку реально сначала происходит заполнение данными текстового буфера, а потом его содержимое записывается на диск. Запись буфера происходит как только он окажется полностью заполненным или при выполнении специальных команд принудительной записи на диск. Процесс записи недозаполненного буфера на диск называется *флэшированием* и обычно выполняется с помощью функции fflush(f_out). При необходимости завершить работу сразу со всеми открытыми файлами пользуются функцией flushall().
Закрытие файла посредством функции fclose(f_out) также включает процесс флэширования, то есть перенос информации из буфера на диск.
Доступ к элементам текстовых файлов возможен только в последовательном режиме как при записи файла, так и при его чтении.

Рабочее задание

Напишите программу согласно Вашему варианту задания.

Варианты заданий

Номер варианта	Задание
1, 14	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 X и Y местами. Результат записать в другой текстовый файл такого же формата.
2, 15	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) ---------- - ----------
Номер варианта	Задание
3, 16	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
4, 17	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.
5, 18	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).
Номер варианта	Задание
6, 19	Создать текстовый файл, содержащий целочисленные значения, следующего формата 5 21 4 37 52 9 . . .Определить, являются ли значения, находящиеся в файле, упорядоченными по возрастанию.
7, 20	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.
8, 21	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
9, 22	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.
10, 23	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).
11, 24	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.
12, 25	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 #
Номер варианта	Задание
13, 26	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) ---------- - ----------

Контрольные вопросы

1. Каким образом можно открыть и закрыть файл для выполнения операций ввода-вывода в программе?
2. Как прочитать данные из файла в программу при помощи технологии программирования?
3. Как записать данные из программы в файл при помощи технологии программирования?
4. Как обработать ошибки при чтении или записи данных в файл при помощи технологии программирования?
5. Каким образом можно перемещаться по файлу и читать его данные порционно в программе?
6. Как реализовать отображение файла в память и работу с ним как с массивом данных в программе?
7. Как обрабатывать текстовые файлы и выполнять операции чтения и записи в них в программе?
8. Каким образом можно работать с двоичными файлами и производить операции чтения и записи бинарных данных в программе?
9. Как реализовать работу с файлами на удаленных серверах или в сети при помощи технологии программирования?
10. Как оценить производительность и эффективность операций ввода-вывода с файлами в программе и провести их оптимизацию?

Практическая работа №32

РАЗРАБОТКА МОДУЛЕЙ ЭКСПЕРТНОЙ СИСТЕМЫ

Цель работы: освоение технологии и методики построения экспертных систем на примере разработанной учебной экспертной системы

Порядок выполнения работы:

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

Время выполнения: 2 ч

Теоретическая часть

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:

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

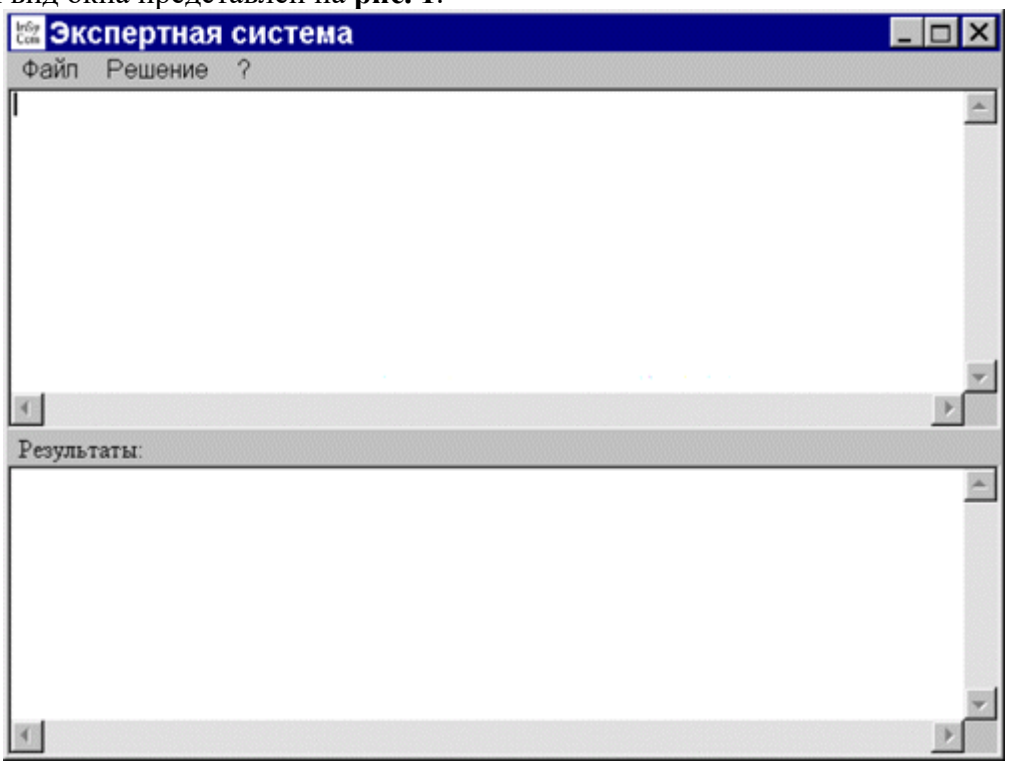
Рабочее задание

Выполнить инструкции приведенные ниже

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**).

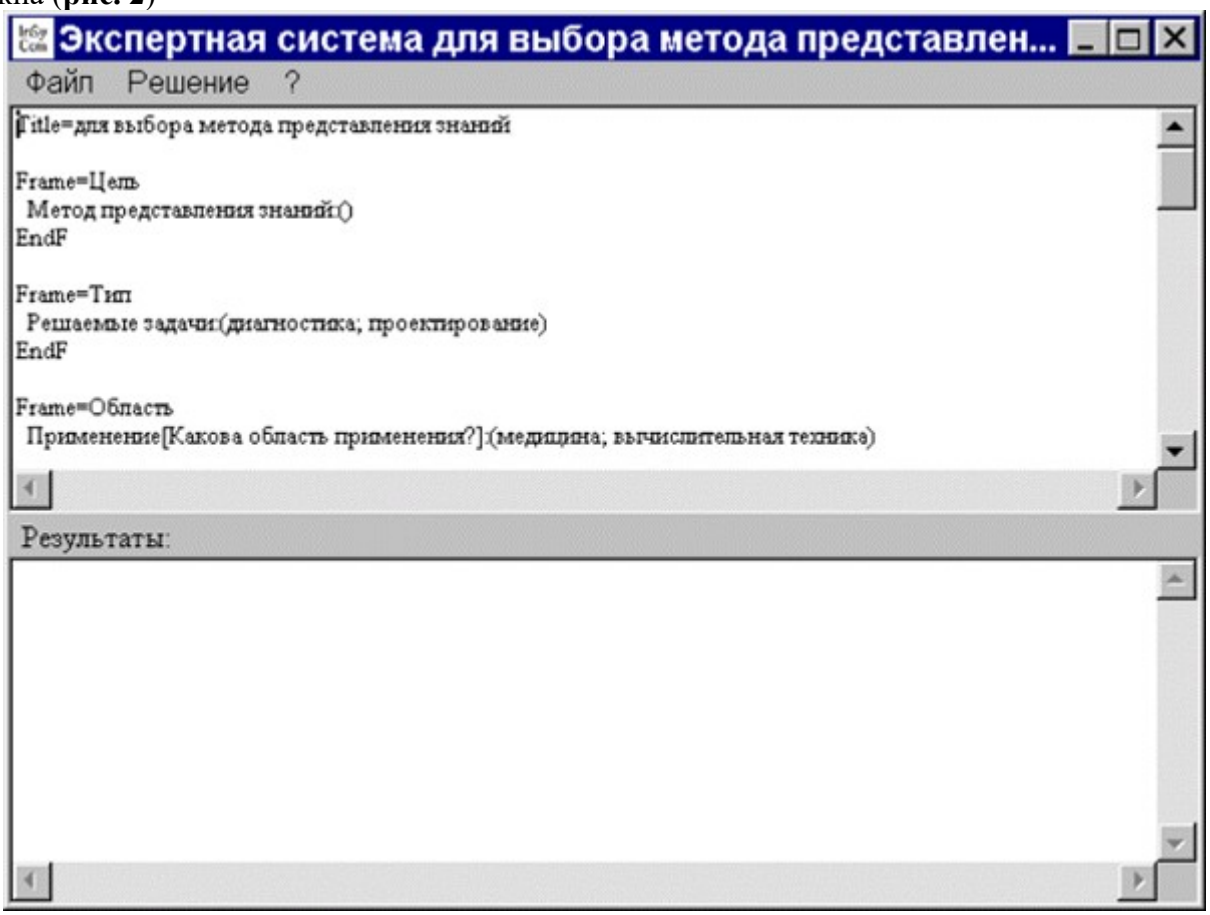
Строка меню состоит из пунктов: "**Файл**", "**Решение**" и "**?**".

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.

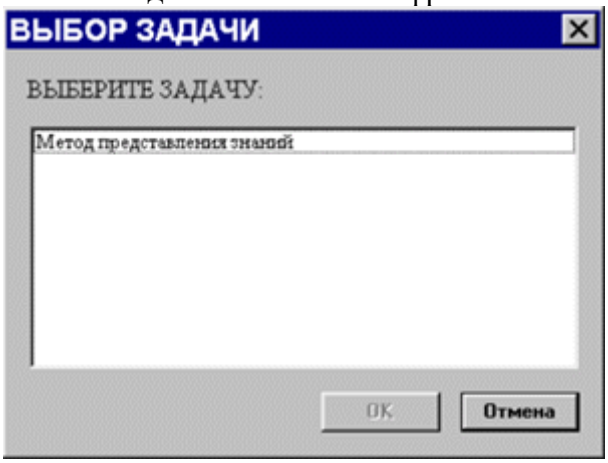
При необходимости можно загрузить базу данных из одноименного файла с расширением ***.dtb**. Для этого используется пункт меню "**Файл**"/"**Загрузить базу данных**".

После загрузки фреймы и правила базы знаний отображаются в верхней части основного

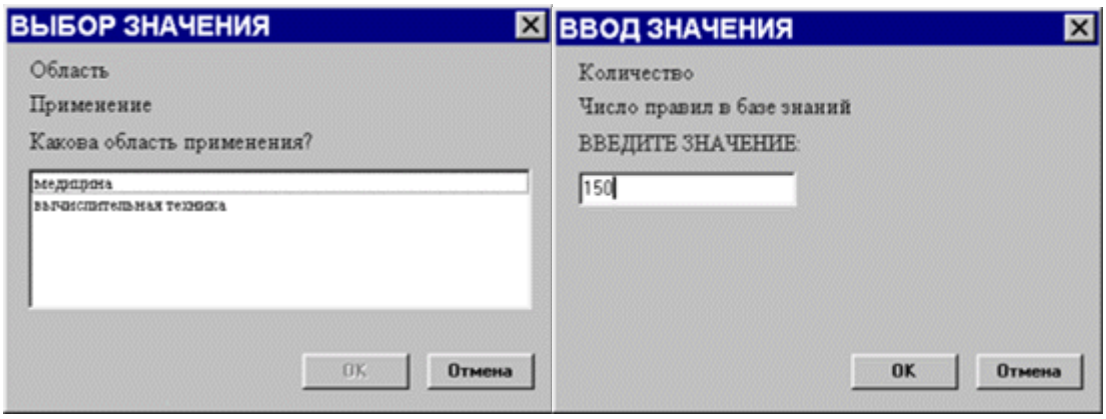
окна (**рис. 2**)



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**".

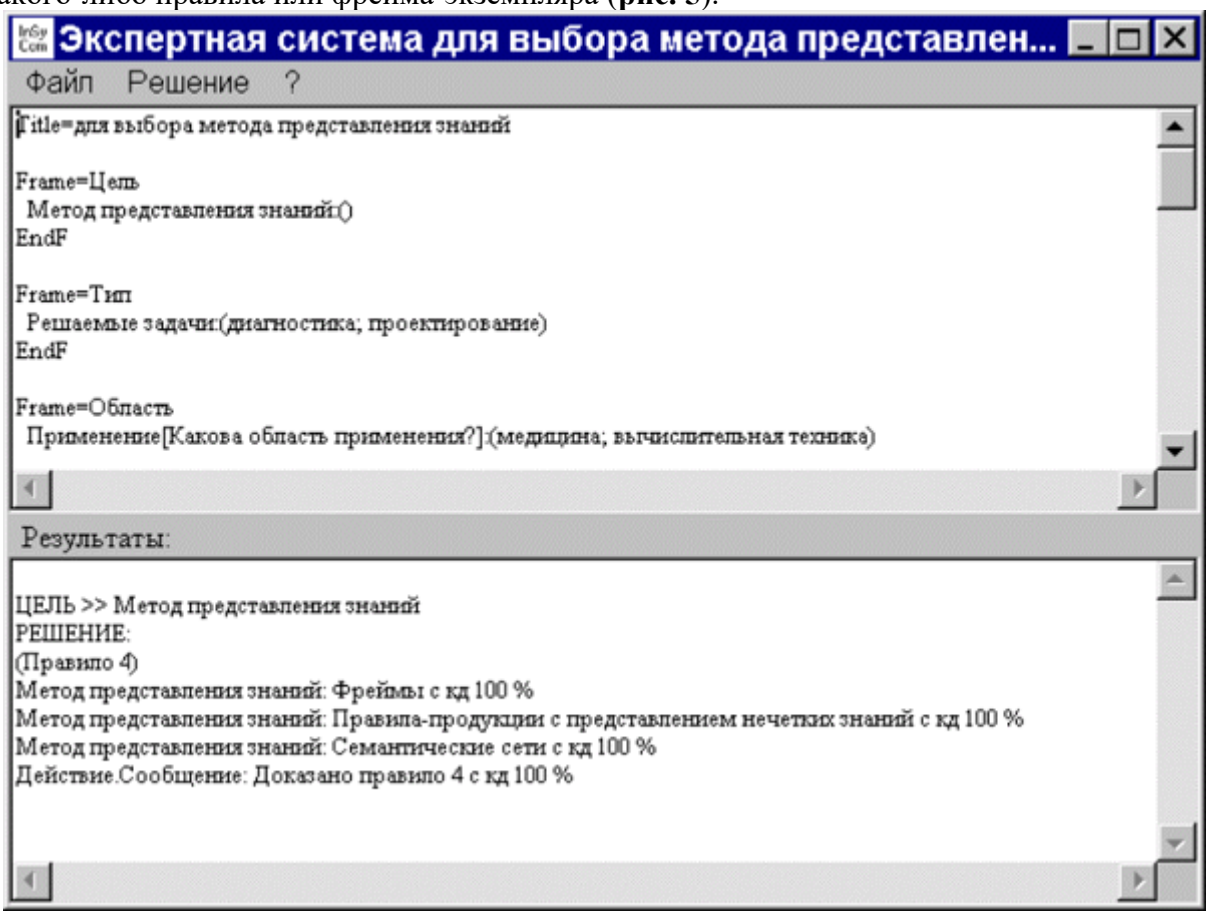


В процессе логического вывода, в качестве ответов на вопросы, задаваемые программной оболочкой, пользователю предлагается выбирать одно из символьных значений или вводить численное значение (**рис. 4**).

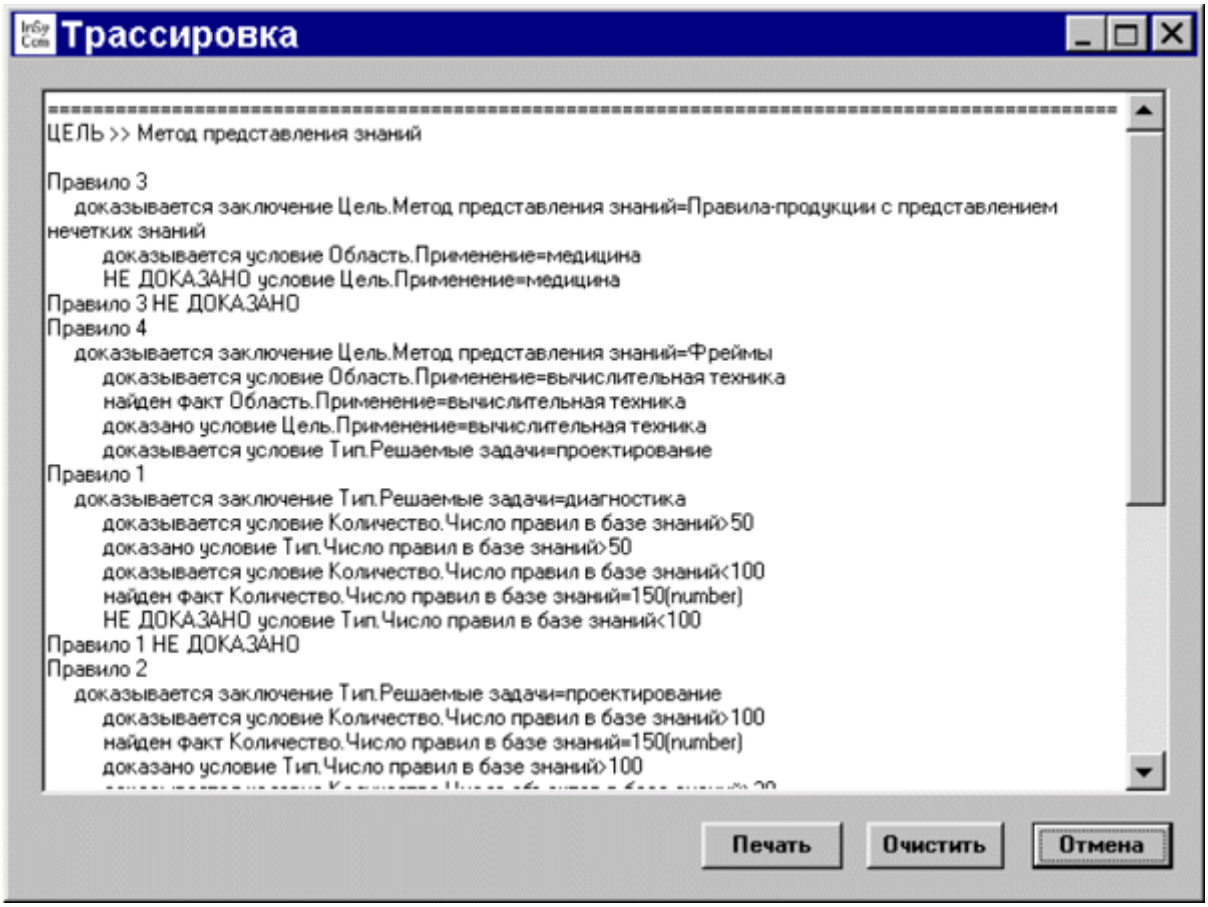


В случае с лингвистической переменной в одном окне будет предложено выбрать одно из символьных значений или ввести численное значение.

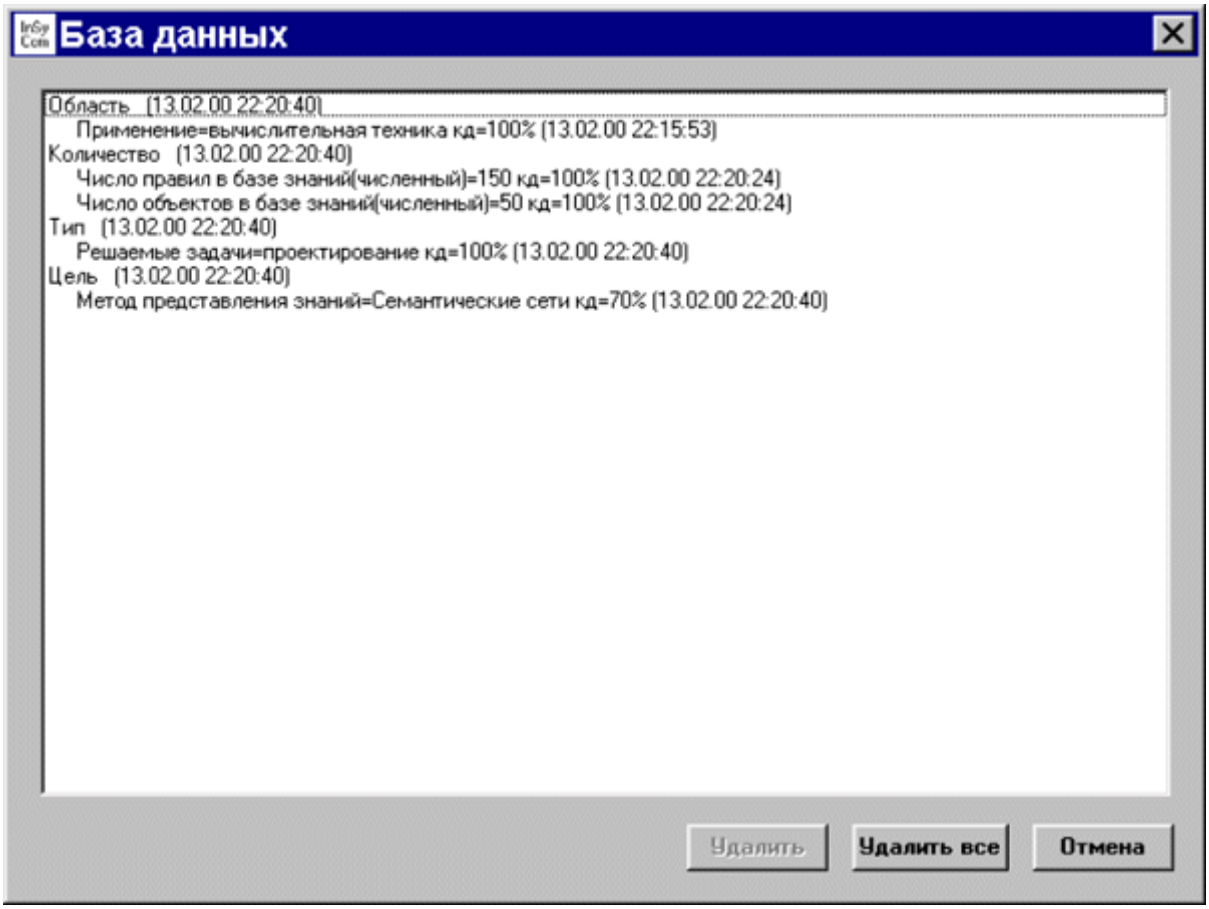
Результаты логического вывода отображаются в нижней части основного окна с комментариями, каким образом было получено решение: в результате доказательства какого-либо правила или фрейма-экземпляра (**рис. 5**).



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.

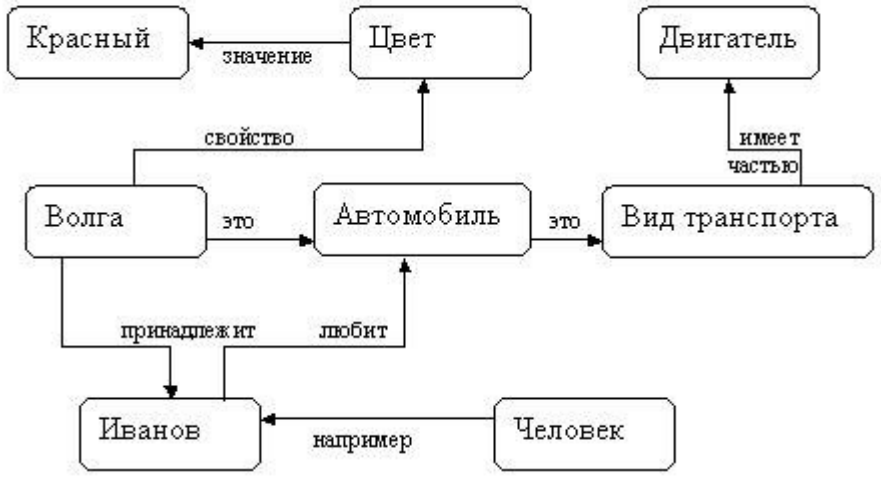


Рис. Семантическая сеть.

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.

Контрольные вопросы

1. Как выбрать подходящую технологию для разработки учебной экспертной системы?
2. Каким образом была спроектирована структура и архитектура учебной экспертной системы?
3. Какой метод или подход к инженерии знаний был использован при разработке учебной экспертной системы?
4. Какое базовое знание и правила были определены для учебной экспертной системы?
5. Каким образом были собраны и структурированы данные для обучения учебной экспертной системы?
6. Как проводилась верификация и валидация учебной экспертной системы?
7. Каким образом пользователи могли взаимодействовать с учебной экспертной системой?
8. Как реализована система вывода и интерпретации результатов в учебной экспертной системе?

9. Как реализован механизм обновления и поддержания знаний в учебной экспертной системе?
10. Как можно оценить эффективность и практическую ценность учебной экспертной системы и провести ее апробацию в реальной среде?

Практическая работа №33
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.

Порядок выполнения работы:

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

Время выполнения: 2 ч

Рабочее задание: 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.

Теоретическая часть

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.

Контрольные вопросы

1. Что представляет собой семейство протоколов TCP/IP и какие протоколы оно включает?
2. Каким образом установливается и разрывается TCP-соединение между клиентом и сервером?
3. Каким образом гарантируется доставка данных в протоколе TCP?
4. Какие функции и методы предоставляет стандарт WINSOCK для программирования сокетов в ОС?
5. Как реализовать создание сокета и его связь с конкретным портом при помощи стандарта WINSOCK?
6. Как реализовать отправку и прием данных через сокет при помощи стандарта WINSOCK?
7. Каким образом обрабатывать ошибки и исключения, возникающие при работе с сокетами в стандарте WINSOCK?

8. Каким образом реализована мультиплексирование и обработка нескольких сокетов одновременно при помощи стандарта WINSOCK?
9. Каким образом управлять уровнем использования пропускной способности и настройками сокетов при помощи стандарта WINSOCK?
10. Как оценить производительность и эффективность работы с сокетами при помощи стандарта WINSOCK и провести оптимизацию сетевого взаимодействия в программе?

Перечень использованных информационных ресурсов

№	ISBN	Название	Назначение	Автор	Издательство	Год издания	Издание	Кол-во в библиотеке	Объем	Наличие на электронных носителях
1		2	3	4	5	6	7	8	9	10
	978-5-4486-0513-0	Объектно-ориентированное программирование и программная инженерия		Мейер Б.	Москва: Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Эр Медиа	2019	Объектно-ориентированное программирование и программная инженерия	1	285 с.	http://www.iprbookshop.ru/79706.html
	978-5-4468-6992-3	Разработка модулей программного обеспечения для компьютерных систем	учебник для студ. СПО	Федорова Г.Н.	Академия	2018	2-е изд, стер.	25	384 с.	
	978-5-4468-6739-4	Разработка и эксплуатация автоматизированных информационных систем	учеб. пособие для студ. СПО	Фуфаев Д.Э.	Академия	2018	6-е изд.,стер.	25	304 с.	
	978-5-4488-0101-3	Алгоритмы и структуры данных	Электрон.текстовые данные (электронный ресурс)	Никлаус, Вирт	Саратов:Профобразование	2017		1	272 с.	http://www.iprbookshop.ru/63821.html
	978-5-4468-6169-9	Основы алгоритмизации и программирования. Практикум	учеб. пособие для сред. проф. образования	Семакин,И.Г.	М.:Академия	2018	2-е изд.	25	144 с.	