«Белорусский государственный педагогический университет
имени Максима Танка»

Факультет физико-математический
Кафедра информатики и методики преподавания информатики

СОГЛАСОВАНО
Заведующий кафедрой
_____ С.В. Вабищевич
22 ___06___ 2018 г.

СОГЛАСОВАНО
Декан факультета
_____ С.И. Василец
21 ___06___ 2018 г.
(рег. № 24-2-201 19д 2018г.)

УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС ПО УЧЕБНОЙ ДИСЦИПЛИНЕ

«Информационные системы и сети»

Для специальности 1 – 02 05 02 «Физика и информатика»

Составители: Г.А. Заборовский, канд. физ-мат. наук, доцент

Рассмотрено и утверждено
на заседании Совета БГПУ 28 ___06___ 2018 г. протокол № 10

ОГЛАВЛЕНИЕ

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Электронный учебно-методический комплекс (ЭУМК) по учебной дисциплине «Информационные системы и сети»

для специальности 1-02 05 02 Физика и информатика

Профессиональная подготовка преподавателя информатики предполагает владение теоретическими основами и технологиями разработки и использования информационных систем и сетевых ресурсов.

Цель учебной дисциплины «Информационные системы и сети» — подготовка будущего преподавателя к разработке и использованию в профессиональной деятельности информационных систем и ресурсов сети Интернет, к активной профессиональной деятельности в условиях современного общества.

Основные задачи учебной дисциплины «Информационные системы и сети»:

- знакомство с объектными моделями текстового процессора MS Word, табличного процессора MS Excel;
- приобретение навыков автоматизации процессов при создании документов в офисных приложениях;
- приобретение знаний современных моделей представления данных;
- формирование системы знаний и умений, связанных с проектированием и разработкой баз данных;
- приобретение навыков работы в СУБД MS Access, формирования запросов различной степени сложности на языке SQL;
- формирование знаний и практических умений в области компьютерных сетей и веб-технологий;
- изучение сущности технологии веб-программирования.

При проведении занятий и работе с ЭУМК следует сочетать традиционные и инновационные методы обучения: лекция-визуализация, работа в малых группах, компьютерное тестирование, работа с электронным дидактическим комплексом.

ТЕОРЕТИЧЕСКИЙ РАЗДЕЛ

Раздел 1. АВТОМАТИЗАЦИЯ РАБОТЫ В ОФИСНЫХ ПРИЛОЖЕНИЯХ

Тема 1.1. Объектная модель MS Word. Исследование объектной модели MS Word Основы программирования в MS Office

Совокупность всех программ и форм документа называется проектом. В проект VBA входят:

Объекты MS Word

Формы

Модули (программы и процедуры)

Окно Project служит для просмотра, навигации по проекту и модификации его структуры. Если открыто несколько файлов, в окне проекта отображаются проекты всех документов. Для активизации окна проекта служит команда View - Project Explorer

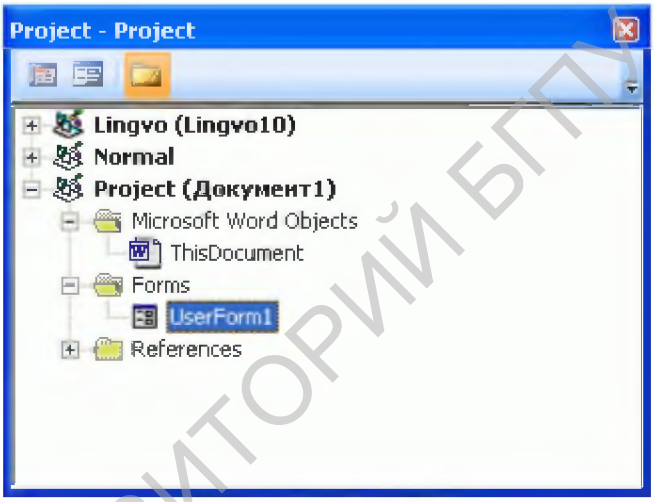


Рис. Окно Project

Каждый открытый документ представляет собой отдельный проект. На рис. показано окно Project для созданного документа MS Word. В проект добавлена форма.

Проект имеет иерархическую структуру. Так, в разделе Microsoft Word Objects (Объекты Microsoft Word) мы можем видеть объект ThisDocument (Этот документ), то есть документ, который мы создали. В разделе Forms (Формы) мы можем видеть объект UserForm1 — только что вставленную в проект форму.

Чтобы отобразить объект, достаточно сделать по нему двойной щелчок или нажать на кнопку View Object (вторая слева в верхней части окна Project). Чтобы просмотреть код объекта надо нажать на кнопку View Code (крайняя левая кнопка).

Щелкнув правой кнопкой мыши по названию проекта, можно увидеть его контекстное меню. Это меню содержит наиболее часто используемые команды для работы с проектами. В частности, особый интерес представляет команда Project Properties (Свойства проекта).

В окне Project Properties обратите внимание на вкладку Protection (Защита). Она позволяет защитить проект от просмотра и закрыть его паролем.

Для защиты проекта от просмотра установите галочку в поле Lock project for viewing (Закрыть проект от просмотра), для закрытия проекта паролем,

введите в поля Password (Пароль) и Confirm password (Подтвердить пароль) пароль.

Создаваемое приложение (программа или пакет программ) реализуется в виде набора взаимосвязанных модулей (блоков). Для каждого модуля можно задавать какие-либо входные и (или) выходные параметры. Добавляется новый модуль командой Insert - Module. Активизация уже существующего модуля - двойной щелчок по имени модуля в окне проекта. При этом в окне редактирования кода отображается содержимое модуля.

Модуль состоит из описания переменных модуля и процедур. Каждая процедура, в свою очередь, может содержать описание переменных процедуры, а также ссылаться на другие процедуры и формы.

Тема 1.2. Основы языка программирования Visual Basic for Application

Редактор VBA: интерфейс и возможности

Как вызвать окно редактора VBA:

вкладка Вид - Макросы - Макросы... - Изменить

вкладка Разработчик - Visual Basic

Alt + F11

В приложениях Microsoft Office предусмотрена специальная вкладка - Разработчик. Она служит для работы с VBA-программами, элементами управления, которые можно добавлять в документ и т.д. По умолчанию эта вкладка скрыта. Чтобы отобразить ее, например, в Microsoft Word, нужно в окне Параметры Word установить галочку в поле Показывать вкладку Разработчик на ленте.

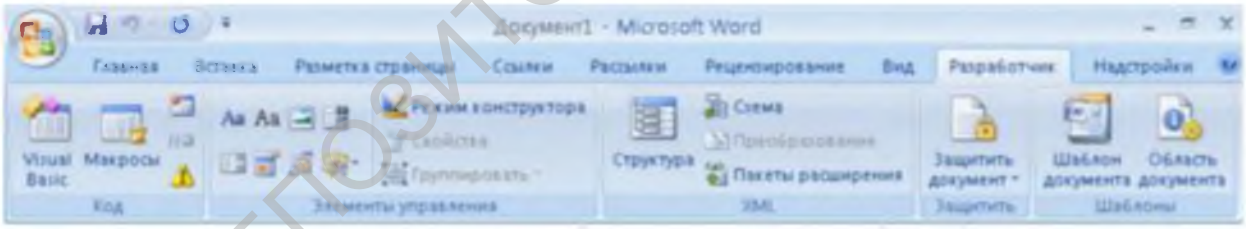


Рис. Вкладка Разработчик на ленте Microsoft Word

Если включено отображение вкладки Разработчик - в строке состояния Word появится кнопка, с помощью которой можно быстро начать или остановить запись макроса.

В области Элементы управления расположены миниатюры элементов управления, которые можно располагать в документах.

Обратите внимание на кнопку Visual Basic - она расположена в области Код. Нажав на эту кнопку, вы запустите редактор VBA.

Окно редактора Visual Basic выглядит одинаково во всех приложениях Microsoft Office и содержит следующие элементы:

- строка меню и панель инструментов;
- окно проекта;
- окно свойств проекта;
- окно редактирования кода (окно программы);
- окно формы.

Строка меню и панель инструментов.

• File (Файл) — служит для работы с файлами.

◦ File - Save — сохраняет файл.

◦ File - Import File — позволяет импортировать внешний файл в редактор. Например, таким образом можно добавить в свой проект модуль (то есть программный код) или форму. ◦ File - Export File — экспортирует данные из редактора во внешний файл. Например, этой командой можно сохранить редактируемую форму и передать ее другому разработчику. ◦ File - Close and Return To Microsoft Word — закрывает VBA-редактор и возвращается в Microsoft Word (аналогичная команда есть и для MS Excel).

Вы можете просто переключаться между редактором и основным приложением в Панели задач Windows, не закрывая редактор.

Edit (Правка) — содержит команды для правки. Помимо стандартных команд отмены и возврата последнего действия (Undo, Redo), вырезания, копирования, вставки (Cut, Copy, Insert), поиска (Find), это меню содержит несколько особенных команд. В частности, это List Properties/Methods (Список Свойств/Методов) и другие.

View (Вид) — содержит команды для отображения различных окон редактора. Названия команд соответствуют названиям окон.

Insert (Вставка) — служит для вставки в проект форм (Insert - User Form), модулей (Insert - Module), процедур (Insert - Procedure), файлов (Insert - File) и модулей класса (Insert - Class Module). Чаще всего вам придется вставлять в проект формы. Эти команды продублированы на панели инструментов редактора.

Format (Форматирование) — служит для управления расположением элементов управления на формах.

Debug (Отладка) — содержит команды для отладки программы.

Run (Запуск) — содержит команды для управления выполнением программ. В частности, команда Run - Sub/User Form (Запуск - Процедура/Форма) запускает на выполнение активную процедуру или форму (рядом с этой командой стоит характерный зеленый треугольник). Команда Run - Break (Запуск - Приостановить) — приостанавливает выполнение программы, команда Run - Reset (Запуск - Перезапуск) - останавливает выполнение программы. Эти команды продублированы на панели инструментов редактора в виде кнопок с соответствующими пиктограммами.

Tools (Инструменты) — содержит средства для настройки свойств редактора, подключения дополнительных библиотек объектов.

Add-Ins (Дополнения) — позволяет управлять дополнениями. По умолчанию это меню содержит лишь одну команду, запускающую менеджер дополнений.

Window (Окно) — стандартные команды для работы с окнами.

Help (Помощь) — помощь по VBA.

группа кнопок на панели инструментов для отображения

основных рабочих областей редактора VBA

Рабочие области редактора

Окно редактора включает в себя несколько рабочих областей - окон, служащих для выполнения различных действий.

По умолчанию в окне редактора присутствуют три рабочих области:

Code (Код) - это окно, в котором пишут тексты VBA-программ и редактируют макросы. Эта область расположена справа и занимает большую часть окна редактора.

Project Explorer (Проводник Проекта) - это окно обычно открыто в левой верхней части окна редактора. Оно отображает информацию о компонентах проекта, позволяет быстро перемещаться между компонентами.

Properties (Свойства) - отображает *свойства* выделенного объекта. Обычно располагается в левой нижней части рабочего окна.

Другие рабочие области:

UserForm (Форма) - служит для редактирования пользовательской формы в визуальном режиме.

Toolbox (Панель элементов управления) - содержит набор элементов управления (кнопки, поля ввода и т.д.), которые можно добавлять на формы или в документы.

Object Browser (Обозреватель объектов) - служит для просмотра информации об объектах, доступных в данном приложении.

Watch, Locals, Immediate - окна, средства которых используются при отладке приложений.

Окно редактирования кода служит в качестве редактора для создания процедур

Для переключения из режима проектирования формы в режим редактирования кода служит команда View - Code (обратный переход View - Object).

В верхней части окна редактора кода расположены два раскрывающихся списка. Левый список содержит все объекты и процедуры модуля, а правый - список событий для выбранного объекта. При редактировании программы используют метод drag-and-drop (перетаскивание) или буфер обмена.

При написании программы редактор кода предлагает пользователю список с возможными вариантами продолжения команды или подсказку с параметрами функции или оператора. Если вы набрали имя элемента управления правильно, то после ввода точки редактор предложит список свойств и методов, доступных данному элементу управления.

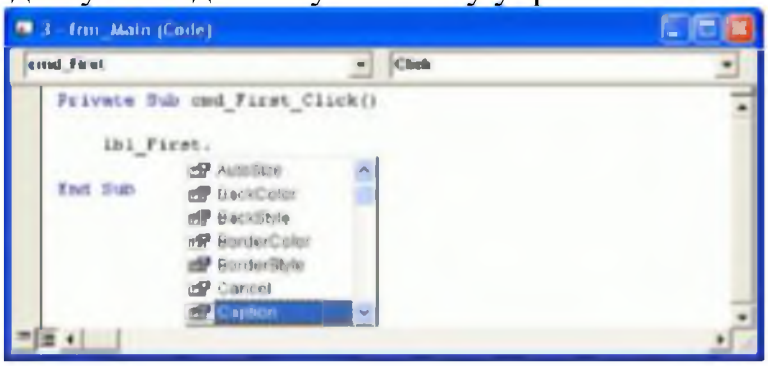


Рис. Подсказка о свойствах и методах объекта в редакторе кода Если в окне редактирования кода находится код модуля, существует возможность работать в двух режимах: просматривать все процедуры модуля или редактировать только одно процедуру. Переключение режимов осуществляется

при помощи команды Tools - Options - вкладка Editor - Default to Full Module View.

и

Окно формы позволяет в визуальном режиме проектировать модифицировать диалоговые окна.

Для вставки в проект новой формы служит команда Insert - User Form.

На форме можно разместить различные элементы управления - кнопки, флажки, переключатели, текстовые поля и т.д. При создании формы или эу VBA устанавл**ивает свойство** Name по умолчанию. Для добавления эу нужно активизировать панель эу: View - ToolBox.

Selected Items Only

На панели инструментов Toolbox в редакторе VBA отображается лишь малая часть элементов управления. Для того чтобы просмотреть установленные в системе элементы и вывести кнопкидля их вызова на панель инструментов, щелкните правой кнопкой мыши по панели и в появившемся меню выберите пункт Additional Control (Дополнительные элементы управления).

Даже если вы специально не устанавливали пакеты элементов - вы увидите большой список. Найдем в этом окне Календарь 12.0, выделим и нажмем OK — на панели инструментов VBA появится новый элемент управления.

Следующий шаг после размещения на форме эу - установка значений их свойств и свойств формы.

Окно свойств объекта содержит значение основных свойств выбранной формы или элемента управления. Состоит из двух частей: верхняя (список для выбора другого элемента или самой формы) и рабочая. Рабочая часть состоит из двух вкладок: «по алфавиту» и «по категориям», свойство Name расположено вверху. Изменить значение свойства можно непосредственным вводом нового значения с клавиатуры или выбором из списка.

На вкладке Categorized свойства объектов объединены в группы. Для удобства группы можно сворачивать и разворачивать.

Appearance (Внешний вид) — отвечает за отображение объекта, за надписи на нем, за его имя.

Behavior (Поведение) — отвечает за поведение объекта. Например, за отображение многострочного текста.Font (Шрифт) — содержит свойство, твечающее за шрифт, которым сделаны надписи на объекте.

Misc (Разное) — различные настройки. Например — настройки указателя мыши, когда он будет находиться над объектом.

Picture (Изображение) — информация об изображении, которое может отображаться на объекте.

Position (Расположение) — определяет размер и положение объекта.

У форм есть группа свойств Scrolling (Скроллинг) — свойства этой группы управляют прокруткой формы.

После проектирования внешнего вида формы приступают к следующему этапу - написанию процедур. Чаще всего разрабатываются процедуры, выполняющиеся при наступлении каких-либо событий.

Наиболее часто используемые события элементов управления:

Click - элемент выбран одинарным щелчком мыши;

DblClick - элемент выбран двойным щелчком мыши;

KeyPress - нажата любая клавиша на клавиатуре (кроме функциональных клавиш и клавиш управления курсором)

Error - во время работы программы произошла ошибка (сбой)

Например, процедура, выполняющаяся при щелчке по кнопке

Помимо обычных объектных моделей приложений Office, вы можете использовать в своих программах другие объектные модели. Например, программируя для Word, можете воспользоваться объектной моделью Excel а так же - любыми другими моделями, установленными в системе. Для этого нужно подключить нужные модели из редактора VBA с помощью команды меню Tools -> References (Инструменты - Ссылки).

В данном случае мы подключаем библиотеку Microsoft Excel (она ценна встроенными функциями) к Microsoft Word, то есть сможем пользоваться некоторыми функциями Excel в Word. Подключенные модели можно просматривать в окне Project Explorer.

Раздел 2. ВЕБ-КОНСТРУИРОВАНИЕ

Для создания Web-страницы можно воспользоваться специальными программами редактирования документов Всемирной паутины. Другой способ подготовки Web-страниц заключается в «ручном» создании кода документов на языке HTML - HyperText Markup Language - Язык разметки гипертекста. Данный язык представляет собой довольно простой набор команд, описывающий структуру документа. Язык HTML позволяет выделить в документе отдельные элементы - заголовки, абзацы, таблицы и т.д. Файлы с текстом кода на языке HTML имеют расширение .html или .htm. HTML является описательным языком разметки документов, в нем используются указатели разметки (теги). Теговая модель описывает документ как совокупность контейнеров, каждый из которых начинается и заканчивается тегами, то есть документ HTML представляет собой не что иное, как обычный файл, с добавленными в него управляющими HTML-кодами (тегами). В нем разрешено использовать только три управляющих символа: горизонтальную табуляцию, перевод каретки и перевод строки. Это облегчает взаимодействие с различными операционными системами.

Теги HTML-документов в большинстве своем просты и понятны, ибо они образованы с помощью общеупотребительных слов английского языка, понятных сокращений и обозначений. HTML-тег состоит из имени, за которым может следовать необязательный список атрибутов тега. Текст тега

заключается в угловые скобки ("<" и ">"). Простейший вариант тега — имя, заключенное в угловые скобки, например <HEAD> или <I>. Для ряда тегов характерно наличие атрибутов (т.е. параметров тега), которые могут иметь конкретные значения, устанавливаемые автором для изменения функции тега.

Гипертекст породил много специальных терминов:

Элемент - конструкция языка HTML. Это контейнер, содержащий данные и позволяющий отформатировать их определенным образом. Любая Web-страница представляет собой набор элементов. Одна из основных идей гипертекста - возможность вложения элементов.

Тег - начальный или конечным маркеры элемента. Теги определяют границы действия элементов и отделяют элементы друг от друга. В тексте Web-страницы теги заключаются в угловые скобки, а конечный тег всегда снабжается косой чертой.

Атрибут - параметр или свойство элемента. Это, по сути, переменная, которая имеет стандартное имя и которой может присваиваться определенный набор значений: стандартных или произвольных. Предполагается, что символьные значения атрибутов заключаются в прямые кавычки, но некоторые браузеры позволяют не использовать кавычки. Это объясняется тем, что тип атрибута всегда известен заранее. Атрибуты располагаются внутри начального тега и отделяются друг от друга пробелами.

Гиперссылка - фрагмент текста, который является указателем на другой файл или объект. Гиперссылки необходимы для того, чтобы обеспечить возможность перехода от данного документа к другому.

Структура Web-страницы

Структура HTML-документа позволяет задействовать вложенные друг в друга контейнеры. Собственно, сам документ — это один большой контейнер, который начинается с тега <HTML> и заканчивается тегом </HTML>. Он указывает браузеру, что данный текст представляет собой HTML-документ и, содержит в себе теги, которые браузер должен выявить, распознать и правильно интерпретировать.

Типичная Интернет-страница состоит из двух частей: головная часть1 (HEAD) и тела (BODY). Эту базовую структуру в простейшем виде можно представить следующим образом:

<HTML> Начало HTML-документа
  <HEAD> Начало головной части
    <TITLE> Начало строки названия страницы
... Строка названия страницы
    </TITLE> Конец строки названия страницы

1 Содержимое головной части не выводится в окне браузера, а содержит данные о документе.

</HEAD> Конец головной части
<BODY> Начало тела документа

</BODY> Конец тела документа
</HTML> Конец HTML-документа

<HTML> </HTML>

Элемент является самым внешним, так как между его начальным и конечным тегами должна находится вся страница. Этот элемент можно рассматривать как формальность.

<HEAD> </HEAD>

Область заголовка Web-страницы. Служит для формирования общей структуры документа. Должен включать элемент TITLE и допускает вложение элемента META.

<TITLE></TITLE>

Элемент разметки TITLE служит для именования документа в World Wide Web. Более прозаическое его назначение — именование окна браузера, в котором просматривается документ. Наличие конечного тега обязательно.

Синтаксис контейнера TITLE в общем виде выглядит

следующим образом: `<T1TbE>название документа</T1TbE>`

Заголовок не является обязательным контейнером документа. Его можно опустить. Роботы многих поисковых систем используют содержание элемента TITLE для создания поискового образа документа. Слова из TITLE попадают в индекс поисковой системы. Из этих соображений элемент TITLE всегда рекомендуется использовать на страницах Web- узла.

Нужно позаботиться о том, чтобы это строка, не будучи слишком длинной, достаточно точно отражала назначение документа. Если тег <TITLE></TITLE> отсутствует, в заголовке браузера выводится реальный адрес и имя просматриваемого html-файла.

<META></META>

Содержит служебную информацию, которая не отражается при просмотре. Внутри нет текста, поэтому он не имеет конечного тега. Этот тег специально рассчитан на программу поискового сервера, индексирующую web-страницы. Секция заголовка может содержать
несколько элементов <META>, каждый из которых отвечает за определенный набор параметров.

Может содержать:

- срок годности документа;
- адрес электронной почты;
- имя автора страницы;
- набор ключевых слов для поиска;
- краткое описание содержания Web-страницы;

- Описание типа и характеристик Web-страницы;
- Указание приложения, в котором была создана Web-страница;
- URL

Наличие этого тега значительно увеличивает шансы попасть в первую десятку адресов, найденных поисковым сервером.

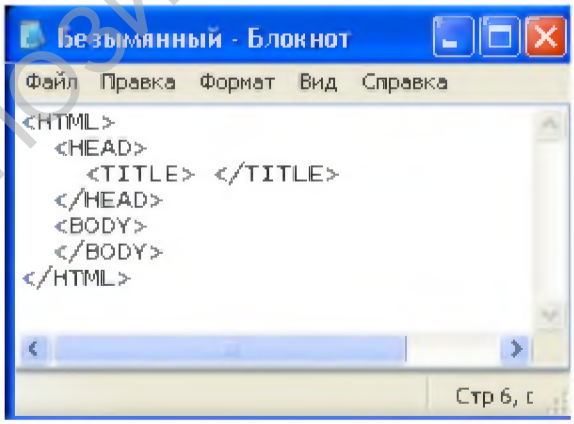
<BODY ></BODY >

Это собственно тело документа. Это та произвольная часть документа, которую разрабатывает автор страницы и которая отображается браузером. Конечный тег этого элемента располагается в конце html-кода. В этом элементе могут использоваться все элементы, предназначенные для дизайна web-страницы. Внутри начального тега <BODY> можно расположить ряд атрибутов, обеспечивающих установки для всей страницы целиком, такие как, цвет фона, фоновую картинку, цвет текста и гиперссылок и т.д.

Создание Web-страницы

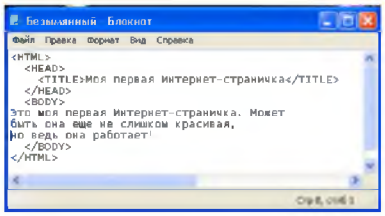
Так как все html-документы имеют одинаковую структуру, рекомендуется создать общий шаблон, в котором будут меняться только название (содержимое тега <TITLE>...</TITLE>) и содержательная часть документа (содержимое контейнера <BODY>...</BODY>

Создавать html-код лучше в простом текстовом редакторе, например, в программе Блокнот. Для создания документа необходимо запустить программу Блокнот и ввести общий для всех страниц код, который определяет структуру html-документа:



Этот документ можно сохранить под именем «шаблон» (Файл/ Сохранить как/ шаблонах!:) и использовать в дальнейшем в качестве заготовки для создания других документов.

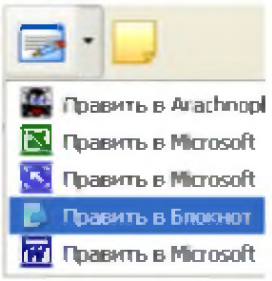
Теперь, чтобы создать web-страницу достаточно открыть файл шаблон-txt, прописать название документа и его содержательную часть:



далее сохранить документ в свою папку под другим именем и дать ему расширение .htm (предпочтительно) или .html (Файл/Сохранить как в строке имя файла ввести: *имя документа.Шш* и нажимает на кнопку Сохранить):

Теперь, чтобы просмотреть страницу, нам достаточно открыть полученный файл **начало.htm**. Однако, в браузере мы можем только просматривать страничку, а чтобы вносить изменения можем только html-коде. Есть несколько способов открыть html-код страницы:

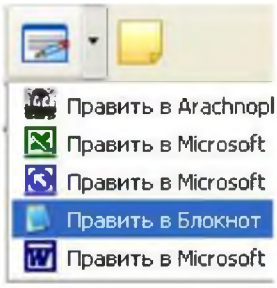
1. Выдрать в меню команду **Вид/Просмотр HTML-кода**.
2. Выбрать на **Панели инструментов** клавишу **Править в Блокнот**



3. Кликнуть по экрану правок кнопкой мыши и выбрать в контекстном меню команду **Просмотр HTML-кода**.

Теперь, чтобы просмотреть страницу, нам достаточно открыть полученный файл начало-htm. Однако, в браузере мы можем только просматривать страничку, а чтобы вносить изменения можем только html-коде. Есть несколько способов открыть html-код страницы:

1. Выдрать в меню команду Вид/Просмотр HTML-кода.
2. Выбрать на Панели инструментов клавишу Править в Блокнот



3. Кликнуть по экрану правок кнопкой мыши и выбрать в контекстном меню команду Просмотр HTML-кода.

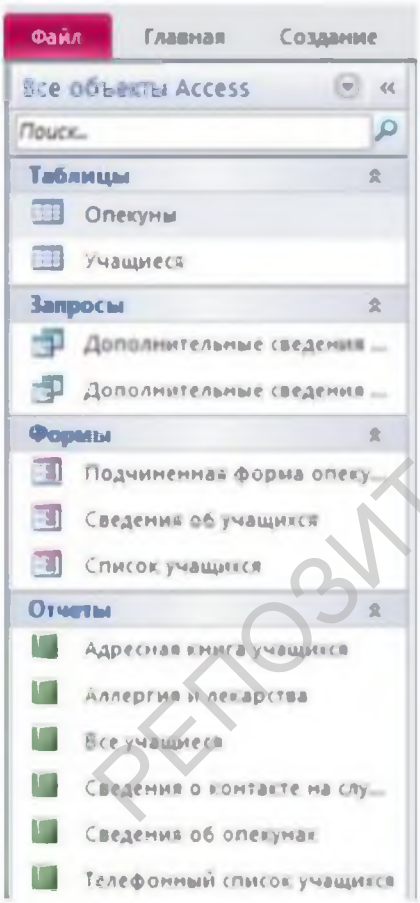
Далее нужно внести изменения в код документа и сохранить в Блокноте. Закрыть Блокнот и в окне браузера нажать на кнопкуОбновить на панели инструментов. После этого внесенные изменения отобразятся на экране.

Раздел 3. ТЕХНОЛОГИИ ОРГАНИЗАЦИИ, ХРАНЕНИЯ И ОБРАБОТКИ ДАННЫХ В СРЕДЕ СИСТЕМЫ УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ. СИСТЕМА УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ MS ACCESS

Объекты базы данных Access

1. *Таблицы* - предназначены для упорядоченного хранения данных.
2. *Запросы* - предназначены для поиска, извлечения данных и выполнения вычислений.
3. *Формы* - предназначены для удобного просмотра, изменения и добавления данных в таблицах.
4. *Отчеты* - используются для анализа и печати данных.
5. *Макросы* - используются для выполнения часто встречающегося набора макрокоманд, осуществляющих обработку данных.
6. *Модули* - предназначены для описания инструкций и процедур на языке VBA.

В Access 2010 объекты базы данных отображаются в области навигации.



В области навигации можно настроить категории и группы объектов. Кроме того, можно скрыть объекты, группы и саму область навигации.

Открытие объекта базы данных в оперативном режиме

- Дважды щелкните объект в области навигации.

-или-

- Выделите объект в области навигации и нажмите клавишу ВВОД.

-или-

- В области навигации щелкните объект правой кнопкой мыши и выберите в контекстном меню команду "Открыть".

Открытие объекта базы данных в режиме конструктора

- В области навигации щелкните объект правой кнопкой мыши и выберите в контекстном меню команду "Конструктор".

Запросы базы данных рекомендуется открывать в режиме конструктора, так как при открытии ***запросов на изменение*** в оперативном режиме можно внести в таблицы нежелательные правки или удалить данные.

Переключение между оперативным режимом и режимом конструктора

Открыв ***таблицу***, ***запрос***, ***форму*** или ***отчет*** можно переключаться между режимами просмотра этих объектов одним из способов:

- **вкладка** Главная ^

Режим **-или-**

- кнопки переключения режимов отображения в строке состояния (справа)

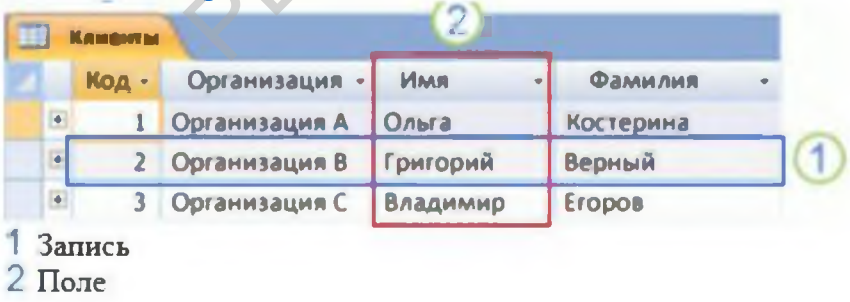
- Структура таблицы

Таблица — это объект базы данных, в котором хранятся сведения по определенной теме, например о сотрудниках или товарах. Таблица состоит из записей и полей.

Каждая ***запись*** содержит данные об одном элементе таблицы, например о конкретном сотруднике. Запись также часто называют строкой или экземпляром.

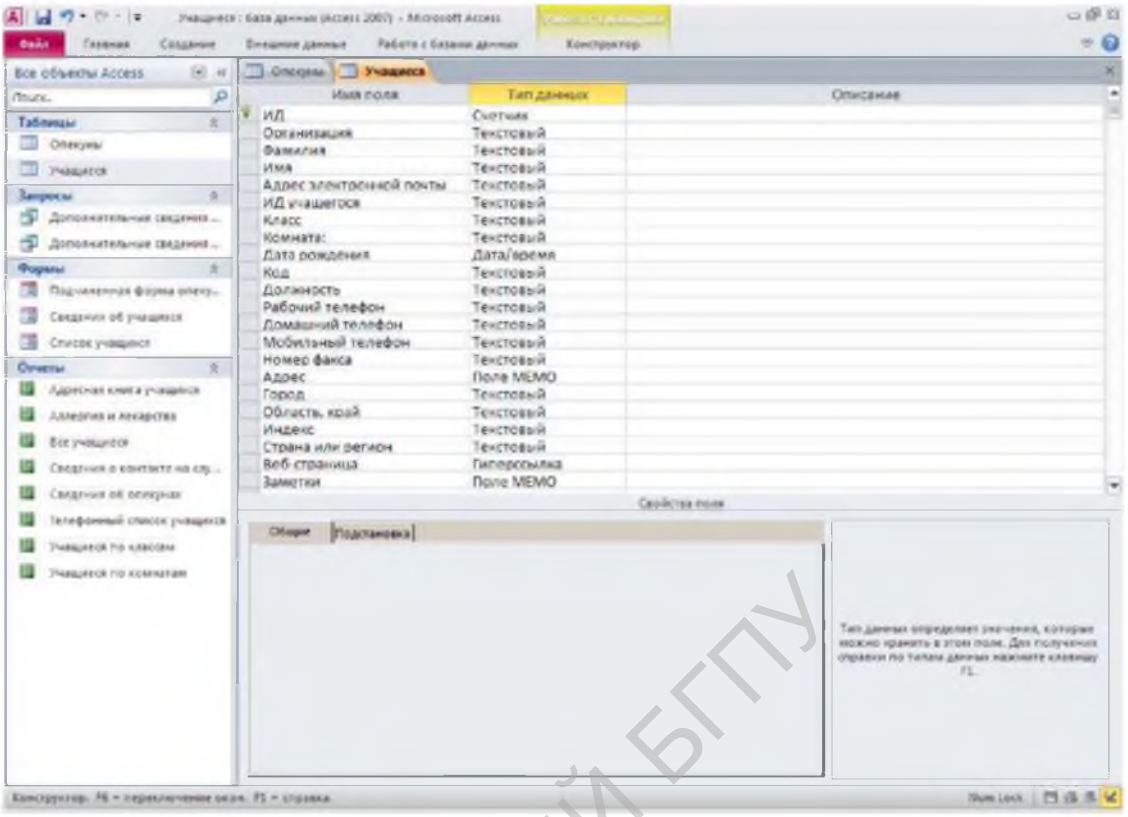
Каждое ***поле*** содержит данные об одном аспекте элемента таблицы, например имя пользователя или адрес электронной почты. Поле также часто называют столбцом или атрибутом.

Запись состоит из значений полей, таких как ***Иванов*** или ivanov@example.com. Значение поля также часто называют фактом.



База данных может содержать множество таблиц, в которых хранятся данные по различным темам. Каждая таблица может содержать множество полей различных типов, таких как текст, числа, даты и гиперссылки.

Таблица в режиме конструктора



Создание таблицы в новой базе данных

1. На вкладке ***Файл*** выберите команду ***Создать*** и щелкните элемент ***Новая база данных***.
2. В поле ***Файл*** введите имя файла новой базы данных.
3. Чтобы изменить место сохранения базы данных, щелкните значок папки.
4. Нажмите кнопку ***Создать***

Откроется новая база данных, в которой будет создана и открыта в режиме таблицы новая таблица с именем «Таблица1».

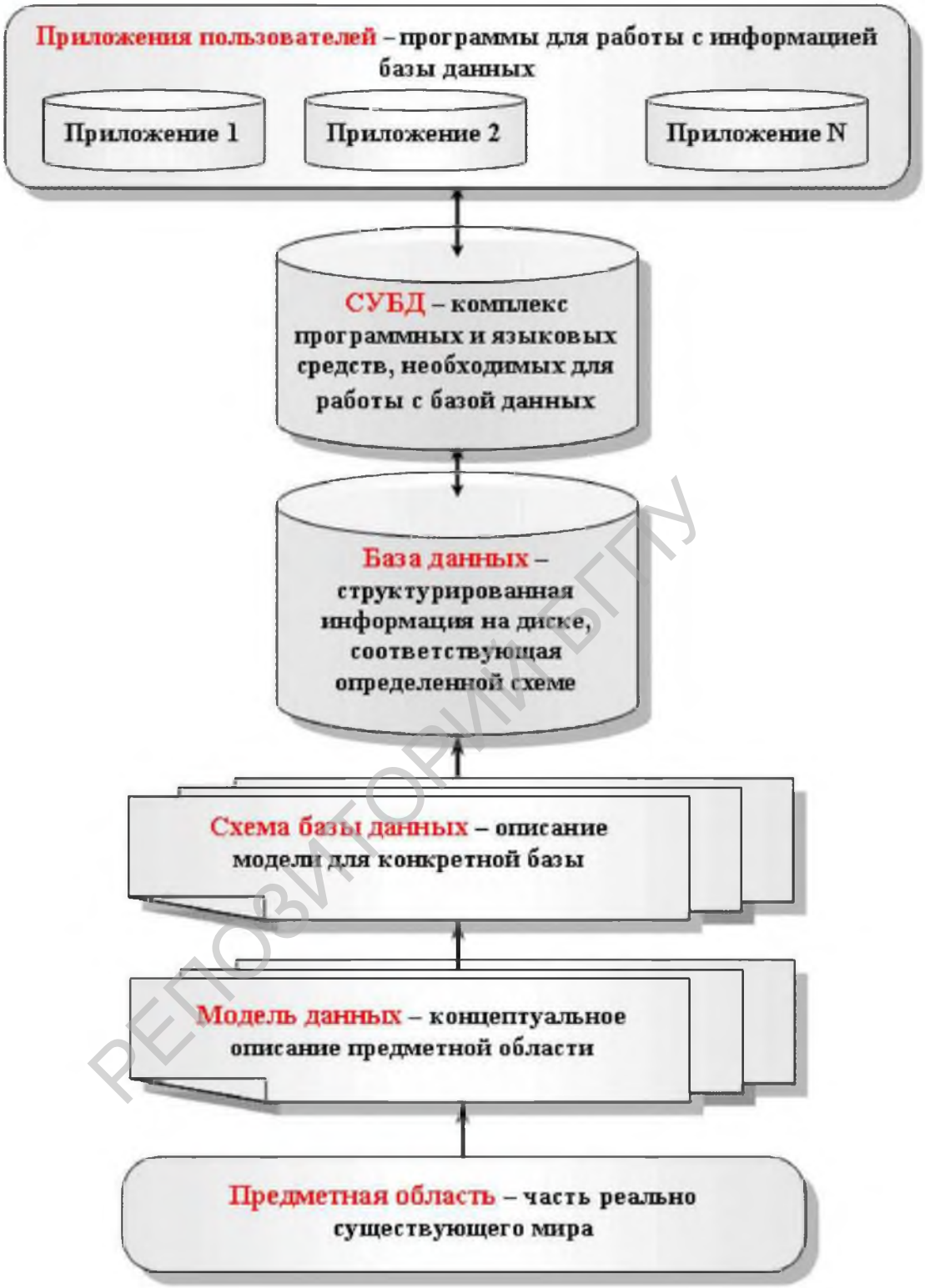
Связь между таблицами

Хотя в каждой таблице хранятся данные по определенной теме, данные в разных таблицах обычно связаны между собой. Для этого в базе данных устанавливают связи между таблицами. Связь — это логическое отношение между двумя таблицами, основанное на их общих полях.

Чтобы установить (удалить, отредактировать) связь между таблицами открывают окно «***Схема данных***» соответствующей командой на вкладке ***Работа с базами данных***.

База данных (БД) - совместно используемый набор логически связанных данных (и их описание), предназначенный для удовлетворения информационных потребностей в некоторой предметной области.

Схема, показывающая взаимосвязь основных терминов в области проектирования баз данных и работы с ними.



Базу данных можно определить как совокупность взаимосвязанных хранящихся вместе данных при наличии такой минимальной избыточности, которая допускает их использование оптимальным образом для одного или нескольких приложений.

Хорошо структурированная информация в БД позволяет не только без проблем эксплуатировать систему и выполнять ее текущее обслуживание, но и модифицировать и развивать ее при изменении информационных потоков и форм отчетности.

СУБД (система управления базами данных) - программное обеспечение, с помощью которого пользователи могут определять, создавать и поддерживать базу данных, а также получать к ней контролируемый доступ.

Различные представления о данных в базах данных

Создание БД предполагает интеграцию данных, предназначенных для решения нескольких прикладных задач разных пользователей. Соответственно, при интеграции данных должны учитываться требования к данным каждого пользователя, основанные на его представлении о данных и связях между ними. Далее эти требования должны обобщаться в единое представление, которое и будет служить основой для построения единой базы данных (рис. 1).

Обобщение представлений всех пользователей о данных называется ***концептуальной моделью (схемой) БД.*** Концептуальная модель представляет информационное описание предметной области с учетом логических взаимосвязей, поэтому её еще называют инфологической (информационно - логической) моделью. В модели отсутствуют какие-либо понятия, связанные с ЭВМ, памятью ЭВМ, способами размещения данных в памяти ЭВМ, и, по сути, это модель только предметной области.

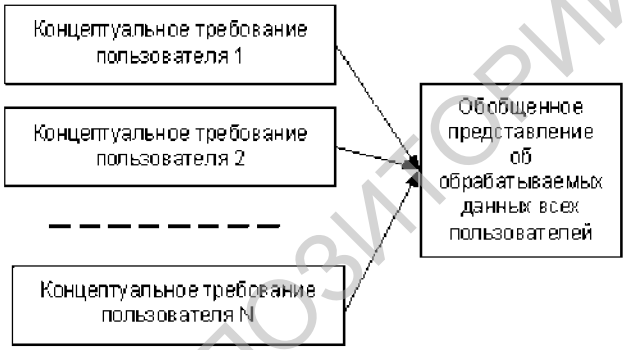


Рис. 1. Обобщение представления пользователей о данных

Для создания БД и работы с ней используется система управления базами данных. Каждая конкретная СУБД поддерживает определенный вид данных (форматов записей и отношений), называемый ***моделью данных СУБД.***

Следующий этап разработки БД предполагает выбор представления концептуальной модели с помощью модели данных конкретной СУБД. ***Полученное таким образом представление*** концептуальной модели ***называется*** логической моделью ***БД. Или другими словами,*** логическая модель ***- это концептуальная схема, специфицированная в языке конкретной СУБД.*** Логическая модель представляет данные и элементы данных вне зависимости от их содержания и среды хранения. Далее

разработчик системы средствами СУБД отображает полученную логическую модель БД в память ЭВМ и определяет методы доступа. Полученное представление данных в памяти ЭВМ называется ***внутренним***

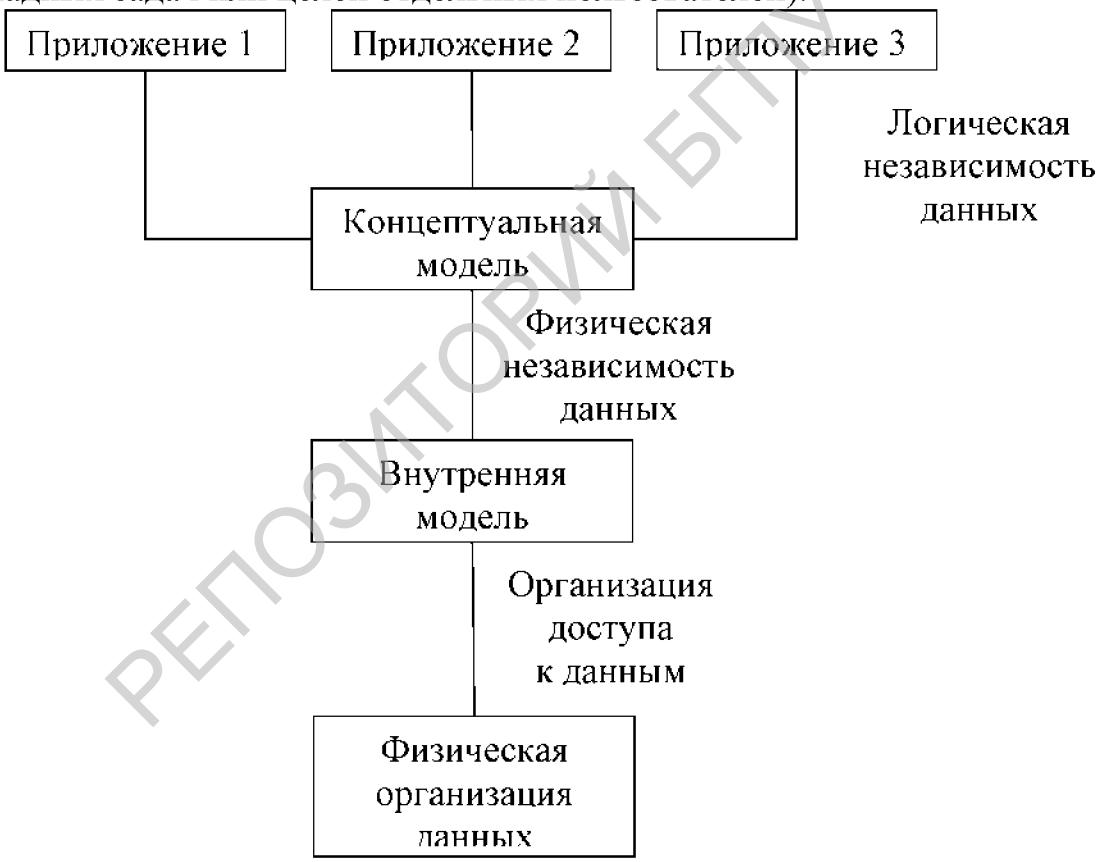
представлением или ***структурой хранения.*** Прикладные программы работают с логической моделью, причем каждому пользователю

представляется подмножество этой логической модели

(подсхема),

отражающее его представление о предметной области. Каждая прикладная программа "видит" и обрабатывает только те данные, которые необходимы именно ей.

Принято выделять несколько уровней архитектуры БД, связанных с разными представлениями и описаниями данных и разными целями этих представлений. Описание данных может быть выполнено на двух «целевых уровнях»: концептуальном (дающем сведения о данных в том виде, в котором они существуют в реальном мире, и не зависящем ни от выбора СУБД, ни от конкретной организации данных в памяти системы) и внутреннем, определяющем логическую организацию хранимых данных средствами конкретной СУБД. По схеме ANSI/SPARC (существует еще один уровень описания, внешний или пользовательский (зависящий от конкретных прикладных задач или целей отдельных пользователей).



На этой схеме есть только одна физическая БД (самый нижний уровень) - тогда как модели всех трех уровней (внутреннего, концептуального и внешнего) являются абстракциями, позволяющими разным пользователям создавать свои собственные представления о БД. Специальные программные средства СУБД позволяют осуществлять переходы от одного уровня к другому.

Одним из качеств организации информации в технологии БД является физическая и логическая независимость. Физическая независимость позволяет модифицировать внутреннюю модель без нарушения концептуальной и внешней моделей; логическая независимость означает, что

можно изменить или расширить концептуальную модель без изменения внешней модели.

Благодаря тому, что СУБД обеспечивает логическую и физическую независимость данных, пользователь получает возможность работать без знания конкретного способа размещения данных в памяти компьютера, то есть с логической структурой базы.

Понятие модели данных.

Одними из основополагающих в концепции баз данных являются обобщенные категории "данные" и "модель данных".

Понятие "данные" в концепции баз данных — это набор конкретных значений, параметров, характеризующих объект, условие, ситуацию или любые другие факторы. Примеры данных: Петров Николай Степанович, $30 и т.д. Данные не обладают определенной структурой, данные становятся информацией тогда, когда пользователь задает им определенную структуру, то есть осознает их смысловое содержание. Поэтому ***центральным понятием в области баз данных является понятие модели***. Не существует однозначного определения этого термина, у разных авторов эта абстракция определяется с некоторыми различиями, но тем не менее можно выделить нечто общее в этих определениях.

Модель данных - это некоторая абстракция, которая, будучи приложенной к конкретным данным, позволяет пользователям и разработчикам трактовать их уже как информацию, то есть сведения, содержащие не только данные, но и взаимосвязь между ними.

Модель данных, или концептуальное описание предметной области - самый абстрактный уровень проектирования баз данных.

Логическая структура данных зависит от СУБД, а точнее от той модели данных, которую поддерживает конкретная СУБД. ***Модель данных*** определяет ***совокупность правил порождения структур*** данных для данной СУБД, возможные ***операции над*** такими ***структурами, множество*** выводимых ***допустимых типов данных*** и ***отношений*** между ними и является основой для построения модели конкретной базы данных.

Или, проще говоря, ***моделью данных***_называют организацию данных и способы доступа к ним.

Понятие ***модели данных*** включает три компоненты:

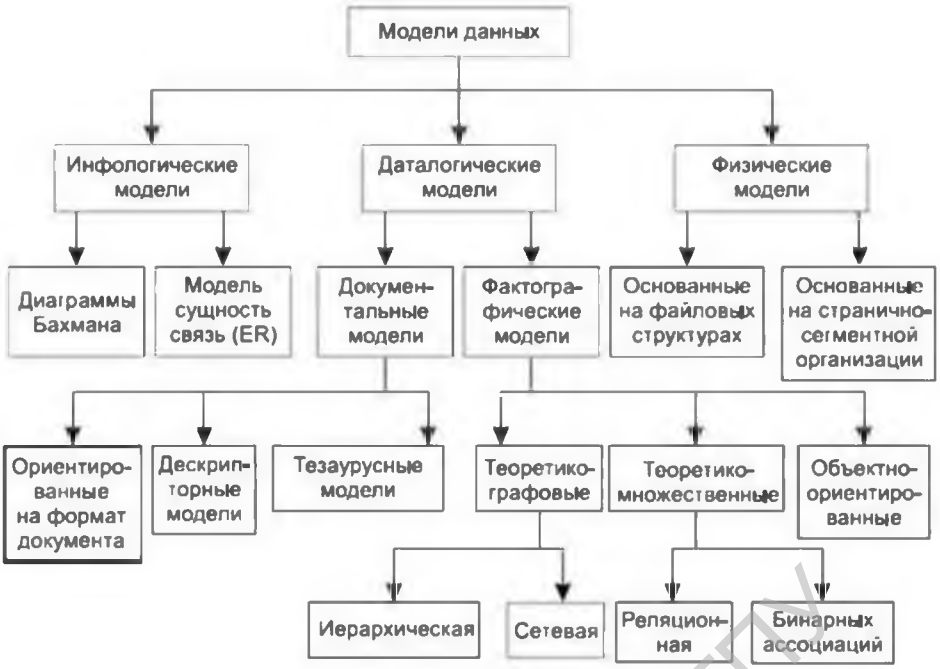
организацию (структуру) данных;

множество допустимых операций над данными;

средства обеспечения логической целостности и достоверности данных.

Классификация моделей данных.

В соответствии с трехуровневой архитектурой понятие модели данных рассматривается по отношению к каждому уровню.



3. Основные модели данных, поддерживаемые современными СУБД.

Теоретико-графовые модели данных отражают совокупность объектов реального мира в виде графа взаимосвязанных информационных объектов. В зависимости от типа графа выделяют иерархическую или сетевую модели. Исторически эти модели появились раньше, и в настоящий момент они используются реже, чем более современная реляционная модель данных. Однако до сих пор существуют системы, работающие на основе этих моделей, а одна из концепций развития объектно-ориентированных баз данных предполагает объединение принципов сетевой модели с концепцией реляционной.

Иерархическая модель данных

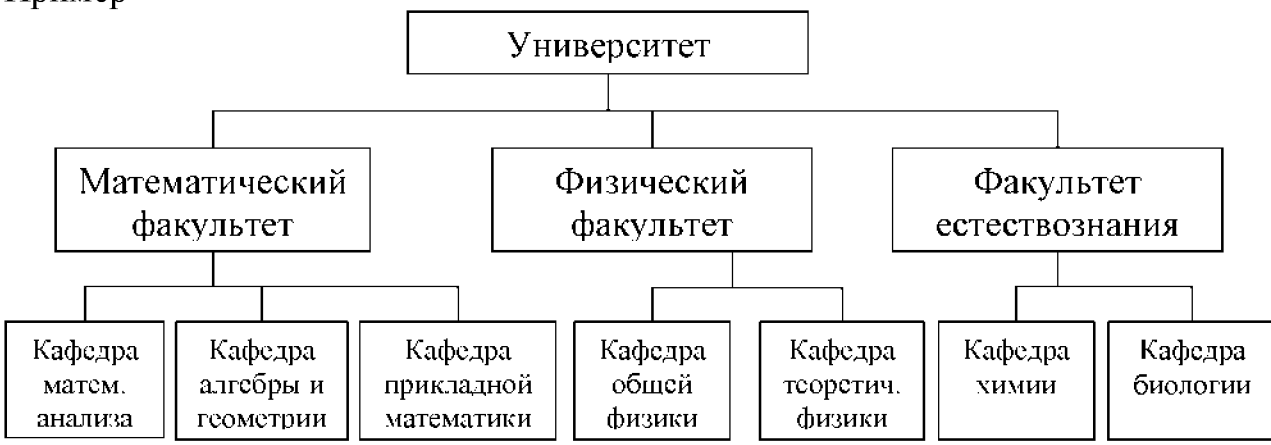
Иерархическая модель данных является наиболее простой среди всех даталогических моделей. Исторически она появилась первой среди всех даталогических моделей: именно эту модель поддерживает первая из зарегистрированных промышленных СУБД IMS фирмы IBM.

Появление иерархической модели связано с тем, что в реальном мире очень многие связи соответствуют иерархии, когда один объект выступает как родительский, а с ним может быть связано множество подчиненных объектов.

Структуру иерархической модели можно представить в виде графа, который называется деревом; объект самого верхнего уровня называется корнем, объекты самого нижнего уровня - листьями этого дерева. Древовидная

(иерархическая) структура может существовать лишь тогда, когда для каждого объекта указан только один исходный.

Пример



Сетевая модель данных

Когда структура данных оказывалась сложнее, чем обычная иерархия, простота иерархической модели данных становилась её недостатком. В связи с этим была разработана новая сетевая модель данных.

Стандарт сетевой модели впервые был определён в 1975 году организацией CODASYL (Conference of Data System Languages - Ассоциация по языкам и системам обработки данных), которая определила базовые понятия модели и формальный язык описания.

К известным сетевым СУБД относятся: DBHS, TOTAL, IDMS, DBVISTA.

Пример 1:

В сетевых базах данных основная структура представления информации имеет форму сети, в которой каждая вершина - узел, может иметь связь с любой другой вершиной (узлом). Данные в сетевой структуре более равноправны. Доступ к информации может быть осуществлён, начиная с любого узла и при наличии соответствующих связей с горизонтальной и вертикальной ориентацией. Однако доступ к информации всё же возможен только по заранее установленным связям.

Раздел 4. WEB-ПРОГРАММИРОВАНИЕ

Тема 4.1. Программирование на стороне клиента. Разработка интерактивных веб-страниц. Основы *JavaScript*

Клиентские скрипты. Обзор технологий для создания клиентских скриптов (JavaScript, VBScript, JScript, ActionScript), их особенности и поддержка в различных браузерах. Безопасность в Интернете. Защита данных и конфиденциальность, технология Cooki, безопасная передача данных и транзакций. Элементы интерактивных Web-страниц. Web программирование. Клиентские и серверные сценарии. Языки сценариев. Язык JavaScript. Алфавит, синтаксис, семантика. Типы данных. Переменные и выражения. Базовые алгоритмические конструкции. Строки. Чтение и

запись строк в файлы cookie.

Числа и даты. Массивы. Переменные, функции и управление последовательностью выполнения. Управление окнами. Формы. Динамические формы. Навигация по сайту. Позиционирование элементов HTML. Динамическое содержимое. Модель объектов JavaScript. Методы объектов и свойства объектов. События. Графика. Стеки гипертекстовых ссылок. Фреймы и окна. Наследование кода скриптов различными страницами. Определение возможностей браузера. Использование JavaScript для разработки ресурсов образовательного назначения.

Тема 4.2. Основы PHP

Язык PHP. Алфавит, синтаксис, семантика. Переменные. Типы данных. Функции. Стандартные функции PHP. Базовые алгоритмические конструкции. Массивы. Работа со строками. Регулярные выражения в языке PHP. Работа с формами в PHP. Целостность данных формы. Обработка форм. Постоянные данные, использующие сеансы и cookie-наборы. Использование шаблонов. Механизм шаблонов Smarty. Взаимодействие PHP и XML. Аутентификация пользователей. Защита PHP-кода. Шифрование данных. Объектноориентированное программирование в PHP. Подготовка сообщений электронной почты в PHP. Создание, использование и поиск Web-служб. Работа с файловой системой. Сетевой ввод-вывод. Программирование сокетов. Вывод графических данных с помощью PHP. Генерация печатаемых документов.

Тема 4.3. Основы MySQL. Язык запросов SQL. Формирование запросов к базе данных

Встроенные функции В VBA имеется большой набор встроенных функций, использование которых существенно упрощает программирование. Эти функции можно разделить на следующие основные категории: математические функции Функции времени и даты функции обработки строк функции проверки типов функции преобразования форматов Встроенные математические функции Функция Описание Abs Абсолютное значение Atn Арктангенс Cos Косинус числа. Exp Возвращает число e (2.718282), возведенное в степень аргумента функции. Fix Отбрасывает дробную часть числа и возвращает целую. В результате для положительных чисел получается число меньшее, чем входное (Fix(2.5) возвратит 2), для отрицательных - большее (Fix(-2.5) возвратит -2) Int Отбрасывает дробную часть числа и возвращает целую. Для положительных получается число меньшее введенного (Int(2.5) возвратит 2), для отрицательных - так же меньшее (Int(-2.5) возвратит -3). Log Возвращает натуральный логарифм числа Rnd Возвращает случайное число типа Single, причем, это число находится между 0 и 1. Для инициализации генератора случайных чисел используйте директиву Randomize - ее надо вызвать до вызова Rnd. Sgn Функция предназначена для определения знака числа. Если число положительное - она возвращает 1. Для нуля функция возвратит 0, для отрицательного числа -1. Sin Синус Sqr Квадратный корень Tan Тангенс

Функции даты и времени Основными функциями дат и времени VBA являются Функция Назначение Date Возвращает или устанавливает текущую дату Time Возвращает или устанавливает текущее время Now Возвращает или устанавливает текущую дату и время

DateSerial

Преобразовывает в дату три последовательных числа: год, месяц, число

TimeSerial

Преобразовывает в дату три последовательных числа: часы, минуты, секунды DateValue Преобразует в дату ее символьное представление TimeValue Преобразует во время его символьное представление Timer Возвращает временной интервал от полуночи (в сек.) Day Преобразует дату в день месяца Month Преобразует дату в месяц года Weekday Преобразует дату в день недели Year Преобразует дату в год Hour Преобразует дату в часы Minute Преобразует дату в минуты Second Преобразует дату в секунды

Встроенные функции обработки строк Функция Назначение StrComp(Строка1,Строка2) Сравнивает две строки Lcase(Строка) Преобразует строку в нижний регистр Ucase(Строка) Преобразует строку в верхний регистр

Space(Число)

Создает строку пробелов, в соответствии с заданным количеством

String(Число, “Символ”)

Создает строку символов, в соответствии с заданным количеством

Len(Строка)

Вычисляет длину строки по количеству символов Instr(Строка, Подстрока) Ищет подстроку в строке Lset Выравнивает строку по левому краю Rset Выравнивает строку по правому краю

Left(Строка, Число)

Выделение левой части строки. Количество символов отсчитывается слева

Right(Строка, Число)

Выделение правой части строки. Количество символов отсчитывается справа Mid(Строка, Число,Число) Выделяет и перемещает строку Ltrim(Строка) Удаляет пробелы в начале строки Rtrim(Строка) Удаляет пробелы в конце строки Trim(Строка) Удаляет пробелы и вначале, и в конце строки Str(Число) Преобразует число в строку Val(Строка) Преобразует строку в число

Format(Число, Шаблон)

Преобразует число в строку по заданному формату

Функции проверки типа данных Если вам нужно узнать тип данных переменной, вы можете воспользоваться функцией TypeName. Чтобы проверить, являются ли данные, хранимые в переменной типа Variant, числом, можно воспользоваться функцией IsNumeric. Для точного определения типа данных, которые хранятся в переменной типа Variant, вы

можете воспользоваться функцией VarType.

Функции преобразования типов Val — тип String в тип Double Функция Val применяется для конверсии строковых переменных в числовые, а именно – переменных типа String в тип Double. Val (" 12345привет") возвратит число 12345. Val читает предлагаемую ей строку слева направо, игнорируя пробелы. Она считывает все числовые знаки до первого символьного знака и преобразует считанное в число. В качестве дробных символов функция понимает лишь точки. Val ("1 2 3") возвратит число 123 Val ("1 2 и 3") возвратит число 12. Иногда нужно провести обратное преобразование — превратить число в строку. Str — числовые типы в String Функция Str конвертирует данные различных числовых типов в тип String. Особенность функции заключается в том, что первый символ полученного строкового значения зарезервирован для знака числа. Если в строку конвертируется число отрицательное — первый символ полученной строки — знак -. Если конвертируется положительное число, первым символом полученной строки будет пробел, а дальше будут идти числовые символы. Например, функция Str (12) возвратит строку " 12". Существуют и другие функции, предназначенные для конверсии типов данных. Их названия состоят из сокращенного слова "Convert" и сокращенного же названия типа данных, в который они конвертируют входные значения. Например, это CBool, CByte, CCur, CDate, CDbl, CDec, CInt, CLng, CSng, CStr, CVar. Скажем, функция CInt конвертирует данные в формат Integer. Учитывая особенности этого типа данных, корректно могут быть сконвертированы лишь значения от -32768 до 32767. Причем, дробные числа округляются при конверсии до ближайшего четного числа — 0.5 округляется до 0, 1.5 — до 2. Если вам понадобятся подробности о каждой из этих функций — обратитесь к справочной системе VBA.

ПРАКТИЧЕСКИЙ РАЗДЕЛ

Раздел 1. АВТОМАТИЗАЦИЯ РАБОТЫ В ОФИСНЫХ ПРИЛОЖЕНИЯХ

Тема 1.1. Объектная модель MS Word. Исследование объектной модели MS Word Основы программирования в MS Office

Лабораторная работа

Цель: ознакомиться с несколькими вариантами реализации тестов; уметь применять на практике текстовые элементы, элементы управления «флажок» и «переключатель», использовать список; самостоятельно разработать сценарии для всех видов тестов.

Флажки

Элемент управления «флажок» используется в случае, когда из предложенных вариантов можно выбрать как один, так и несколько. Каждый вариант выбора задается флажком, который можно либо установить, либо сбросить. Флажок определяется в теге <input> значением checkbox параметра type. Обязательным параметром является параметр value, значение которого будет передано на обработку в случае выбора нажатием кнопки.

Выбор характеристик издания

Предположим, читателю предлагается заполнить анкету, в которой требуется указать название любимого издания и выбрать из предложенного списка характеристики, которые присущи рассматриваемому изданию.

Для задания характеристик можно воспользоваться флажком. Пользователь устанавливает флажки для тех свойств, которыми, по его мнению, обладает издание. Обработка анкеты будет состоять в том, что выбранные свойства будут отражены в поле ввода многострочного текста (рисунок 1).

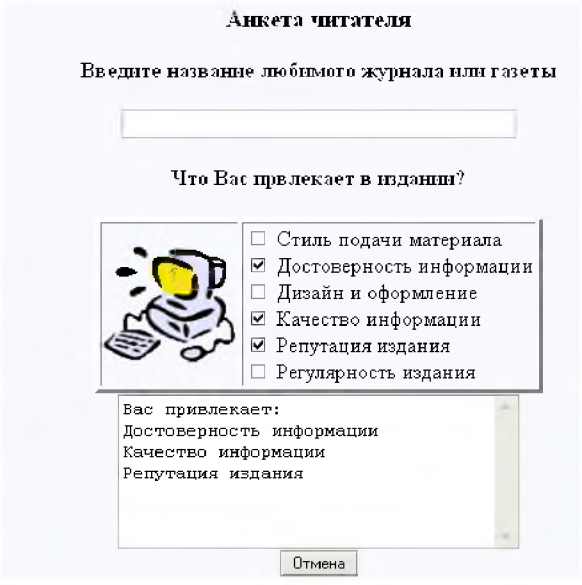


Рисунок 1 – Пример обработки анкеты читателя

При щелчке мышью по флажку возникает событие Click, обработка которого состоит в вызове функции set с одним параметром, принимающим значение параметра value флажка. Для формирования строки результата служит глобальная переменная s; к имеющемуся значению добавляется значение параметра функции и помещается в текстовое поле. Если нажать на кнопку **Отмена**, то очистятся все поля формы. Однако следует позаботиться о том, чтобы значение переменной s изменилось на начальное. Значение параметра реакции на событие Click при щелчке по кнопке **Отмена** задается оператором присваивания, обеспечивающим начальные условия.

HTML-код представлен в листинге 4.1, а.

Листинг 4.1,а Анкета читателя

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Анкета читателя</title>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
<script>
var s="Вас привлекает:  \r\n"
function set(vch)
{ s=s+vch+"\r\n"; document.form1.area.value=s }
</script>
</head>
<body bgcolor="f8f8ff">
<center>
<h3 align="center">Анкета читателя</h3>
<form name="form0">
<H4>Введите название любимого журнала или газеты</H4>
<input type="text" name="n1" size="45"><br>
</form>

<form name="form1">
<h4> Что Вас привлекает в издании?</h4>
<table border="3" align="center">
<tr>
<td><img src="n1.gif" align="center"></td>
        <td align="left"><input type="checkbox" name="m1"
        value="Стиль подачи материала"
                        onClick="set(form1.elements[0].value)">
                        Стиль подачи материала<br>
                <input type="checkbox" name="m2"
        value="Достоверность информации"
                        onClick="set(form1.elements[1].value)">
                        Достоверность информации<br>
                <input type="checkbox" name="m3"
        value="Дизайн и оформление"
                        onClick="set(form1.elements[2].value)">
                        Дизайн и оформление<br>
                <input type="checkbox" name="m4"
        value="Качество информации"
                        onClick="set(form1.elements[3].value)">
```

```
                    Качество информации<br>
                <input type="checkbox" name="m5"
        value="Репутация издания"
                    onClick="set(form1.elements[4].value)">
                    Репутацияиздания<br>
                <input type="checkbox" name="m6"
        value="Регулярностьиздания"
                    onClick="set(form1.elements[5].value)">
                    Регулярность издания<br>
        </td>
</tr>
</table>
<center>
<textarea   name="area" cols=35 rows=7></textarea><br>
</center>
<input type="reset" value="Отмена"
onClick="s=' Вас привлекает: \r\n' ">

</form>
</body>
</html>
```

Если флажок получает фокус, то происходит событие Focus, в качестве значения параметра обработки события, как и в предыдущем случае, может быть вызов функции set:

```
<input type="checkbox" name="m2"
        value="Достоверность информации"
                    onFocus="set(form1.elements[1].value)">
Достоверность информации<br>
```

И, наконец, потеря объектом фокуса вызовет событие Blur, обработка которого может быть произведена аналогичным способом:

```
<input type="checkbox" name="m2"
        value="Достоверность информации"
                    onBlur="set(form1.elements[1].value)">
                    Достоверность информации<br>
```

Объект checkbox обладает свойствами value, name, type, которые соответствуют параметрам тега, описывающего флажок. Функция set получает в качестве параметра объект checkbox и формирует для выбранного флажка значения соответствующих ему свойств так, как показано на рисунке.

Тема 1.2. Основы языка программирования Visual Basic for Application

Раздел 2. ВЕБ-КОНСТРУИРОВАНИЕ

Лабораторная работа

Тема «Стилевое оформление веб-страниц»

Цель - формирование навыков создания и использования стилей CSS

Упр. 1. Создать стили оформления элементов страницы веб-справочника Запустим веб-редактор. Откроем документ inter1.htm. В нем отсутствуют теги и атрибуты оформления (font, color, align, …), а размечена только структура (заголовки h2, h3; абзацы p, списки ul…). Такую разметку называют логической. Изменим заданные по умолчанию стили оформления элементов, т.е. будем использовать их в качестве селекторов типа. Начнем с тега <body>, который задает вид всей страницы. Для этого из меню Формат (Format) вызовем диалоговое окно Стиль (Style). В списке тегов выберем <body> и нажмем Изменить (Change). В выпадающем списке кнопки Формат выберем пункт Шрифт (Font) и в появившемся окне выберем Arial. После подтверждений OK,… OK в разделе <head> веб-документа появился контейнер <style>, куда и записано правило стиля: body { font-family:Arial }. Текст на всей странице стал отображаться этим шрифтом. Аналогичными действиями будем изменять стили оформления других элементов нашей страницы. После каждого изменения будем сохранять ее в папке pro и просматривать в браузере. Заголовок h2 - абзац (paragraph): выравнивание по центру; граница (border): сплошная, цвет: оливковый, поля (padding) по 10px со всех сторон; заливка (background): цвет фона светлосерый, цвет текста темно-синий. В контейнер <style> добавятся правила: h2 { text-align:center; color:#000080; border: 1px solid #808000; padding:10px; background-color:#CCCCCC } Заголовок h3 - шрифт: цвет темно-синий. В контейнер <style> добавится правило: h3 { color:#000080 }

Создадим пользовательский стиль, применимый к нескольким элементам (селектор класса). Для этого из меню Формат вызовем диалоговое окно Стиль, нажмем Создать (Create) и введем имя my. Действия по заданию свойств стиля не отличаются от уже описанных. Зададим шрифт: цвет оливковый, начертание курсив. Добавились правила .my { color:#808000; font-style:italic }. Заметим, что имени селектора класса предшествует точка. Это имя появилось в списках стилей. Созданный стиль класса можно применять к разным элементам (выделяя их), например, к заголовкам 1. Основы Internet и 2. Сервисы Internet, что указывается в их тегах: <h3 class="my">. Наконец, создадим стиль, применимый лишь к одному элементу с именем art (селектор ID). Ввод имени (ID) этого стиля начнем с символа # (решетка). Действия по заданию свойств не отличаются от уже описанных: граница: сплошная, цвет: оливковый, поля по 20px; заливка: цвет фона светлосерый; В контейнер <style> добавятся правила: #art { border: 1px solid #808000; padding:20px; background-color:#CCCCCC } Созданный стиль ID применим лишь к элементу с именем art, например, контейнеру div.

Поместим в этот контейнер все шесть статей нашего справочника. Для этого заключим их в теги ……. . Текст всех статей будет

отображаться на сером прямоугольнике. В заключение подчеркнем, что стили можно создавать и изменять как с помощью диалоговых окон меню Формат, так и редактированием стилевых правил вручную.

Упр. 2. Перенести созданные стилевые правила в отдельный файл и сохранить под именем style1.css. Использовать их для оформления двух страниц inter1.htm и inter2.htm. С помощью меню Создать (Create) создадим пустой текстовый файл. Переместим в него (Ctrl+X Ctrl+V) созданную таблицу стилей (содержимое контейнера <style>…</style> Сохраним под именем style1.css в той же папке pro. С помощью меню Формат (Format) свяжем страницу inter1.htm с файлом style1.css. При этом в разделе <head> страницы inter1.htm появится запись <link rel="stylesheet" type="text/css" href="style1.css"> Откроем в веб-редакторе вторую страницу справочника inter2.htm.

Свяжем и ее с той же таблицей стилей style1.css.

Сравним отображение обеих страниц в браузере. Добавим в inter2.htm имена селекторов: <h3 class="my"> и контейнер ….. . После каждого изменения будем сохранять страницу inter2.htm в папке pro и просматривать в браузере.

Упр. 4сам Создать фрагмент веб-сайта "Песняры беларускай зямлі", используя CSS.

- Все страницы сайта будем сохранять в папке **pismen**, куда скопируем заранее подготовленные рисунки.
- Создадим стили оформления элементов страниц. Начнем со стиля контейнера-обертки ** .** Для этого в диалоговом окне **Положение (Position)** установим его ширину 720 и высоту 560 пикселей, а на вкладке **Заливка** окна **Границы и заливка** укажем имя файла **ramka.gif** с фоновым рисунком.
- Создадим стили вложенных элементов: 1) заголовка, 2) портрета и биографии, 3) гиперссылок.
- Оформим страницу в соответствии с рис.1: скопируем заранее подготовленный текст из файла **biograf.txt**, вставим рисунок из файла **kupala.jpg**. Сохраним страницу под именем **Kupala.htm** в папке **pismen**.

Рис 1

Рис 2

- Аналогично оформим страницы «Якуб Колас» и «Максім Багдановіч», сохранив под именами **Kolas.htm** и **Bagdanovich.htm** в папке **pismen**.
- Главную страницу оформим в соответствии с рисунком 2. Сохраним страницу под именем **index.htm.**
- Создадим все ссылки на главной и персональных страницах в соответствии со схемой навигации по сайту.
- Продемонстрируем страницы в браузере. Проверим работу гиперссылок.

Раздел 3. ТЕХНОЛОГИИ ОРГАНИЗАЦИИ, ХРАНЕНИЯ И ОБРАБОТКИ ДАННЫХ В СРЕДЕ СИСТЕМЫ УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ. СИСТЕМА УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ MS ACCESS

Лабораторная работа

Тема «Построение запросов SQL»

Запрос 1: Найти среднюю стоимость услуг:

```
SELECT AVG(price)
FROM tbl_service
```

Запрос 2: С помощью следующего оператора можно найти среднюю стоимость всех услуг в случае удвоения их цены:

```
SELECT AVG(price*2)
FROM tbl_service
```

Если в столбце, к которому применяется агрегирующая функция, имеются пустые (NULL) значения, то они просто игнорируются.

Исключением является функция COUNT(*), которая всегда подсчитывает общее количество строк, независимо от наличия в них нулевых значений.

Запрос 3: Подсчитать количество расторгнутых договоров:

```
SELECT count(retire_date)
From tbl_contract
Where retire_date not like ''
```

Поскольку поле retire_date содержит дату расторжения договора и пустую строку в случае, если договор не был расторгнут, функция COUNT не учитывает строки, содержащие значения пустой строки в поле retire_date.

Запрос 4: Нужно получить список услуг, указав для каждой из них текущую цену, процентное снижение цены и новую стоимость. Стоимость услуг до 100 у.е. снижается на 10 процентов, для услуг, стоимостью от 100 у.е до 200 у.е снижение составит 20 процентов, для услуг стоимостью выше 200у.е. — 30 процентов.

Без оператора UNION вам потребовалось бы выполнить три отдельных запроса и поместить результаты в новую таблицу.

Оператор UNION выполняет все это за один проход:

```
SELECT '10% off' as sale, price as old_price,
price*0.9 as new_price
From tbl_service
Where price<100
UNION
SELECT '20% off' as sale, price as old_price,
price*0.8 as new_price
From tbl_service
where price Between 100 and 200
UNION
SELECT '30% off' as sale, price as old_price,
price*0.7 as new_price
```

```
From tbl_service
where price>200;
```

Раздел 4. WEB-ПРОГРАММИРОВАНИЕ

Тема 4.1. Программирование на стороне клиента. Разработка интерактивных веб-страниц. Основы *JavaScript*

ЛАБОРАТОРНАЯ РАБОТА

Тема: «Общие сведения по написанию сценариев в JavaScript»

Цель: ознакомиться с правилами размещения сценариев JavaScript в HTML-документе, изучить различные способы использования функций, рассмотреть основные обработчики событий.

Размещение сценариев JavaScript в HTML-документе

Сценарии, написанные на языке JavaScript, могут располагаться непосредственно в HTML-документе между тегами <script>и </script>.

Одним из параметров тега <script> является language, который определяет используемый язык сценариев. Для языка JavaScript значение параметра равно “JavaScript”. Если применяется язык сценариев VBScript, то значение параметра должно быть равным “VBScript”. В случае использования языка JavaScript параметр language можно опустить, так как этот язык выбирается браузером по умолчанию.

Обычно браузеры, не поддерживающие какие-либо теги HTML, эти теги просто игнорируют. Попытка браузера проанализировать содержимое не поддерживаемых тегов может привести к неверному отображению страницы. Чтобы избежать такой ситуации, рекомендуется помещать операторы языка JavaScript в теги комментария <! – –... – –>. Для правильной работы интерпретатора перед закрывающим тегом комментария – –> следует поставить символы //.

Итак, для размещения сценария в HTML-документе следует написать следующее:

```
<script language="JavaScript”>
<!--
операторы языка JavaScript
//-->
</script>
```

Документ может содержать несколько тегов <script>. Все они последовательно обрабатываются интерпретатором JavaScript.

1.1 Вычисление площади треугольника

Необходимо написать сценарий, определяющий площадь прямоугольного треугольника по заданным катетам. Сценарий разместим в

разделе <body> HTML-документа (листинг 1.1).

Листинг 1.1 Первый сценарий в документе

```
<html>
<head>
<title>Первый сценарий в документе</title>
</head>
<body>
<p>Страница, содержащая сценарий.</p>
<script language="JavaScript" type="text/javascript">
<!--
var a=8; h=10
document.write("Площадь прямоугольного треугольника равна ",a*h/2,".")
//-->
</script>
<p>Конец форматирования страницы, содержащей сценарий</p>
</body>
</html>
```

В сценарии описываются и инициализируются две переменные, затем значение выражения записывается в документ. Для формирования вывода в HTML-страницу используется метод write объекта document. Строки, записываемые в документ, могут включать в себя теги HTML и выражения JavaScript.

Тег <noscript> определяет HTML-код, отображаемый на экране в случае, если JavaScript не поддерживается браузером или поддержка отключена. Этот тег следует после кода, заключенного в теги <script> и </script>. Если поддержка включена, то тег <noscript> игнорируется.

В дальнейших примерах будем считать, что поддержка JavaScript включена.

Формальный параметр type="text/javascript" будем опускать.

Современные браузеры понимают язык сценариев JavaScript, поэтому комментарии тоже предлагается опускать.

Достаточно сценарий заключить в теги <script>и </script>.

Индивидуальные задания

Вариант 1

Написать сценарий JavaScript вычисления площади параллелограмма. Ниже представлен рекомендуемый вид HTML-документа во время выполнения сценария.

Вычисление площади параллелограмма.
Введите исходные данные:
Длина (см) → ***9***

Ширина (см) → ***7.5***

Площадь параллелограмма: 67.50 кв.см.

Вариант 2

Написать сценарий JavaScript вычисления объема параллелепипеда с использованием функции, реализующей передачу параметров по значению. Ниже представлен рекомендуемый вид HTML-документа во время выполнения сценария.

Вычисление объема параллелепипеда.
Введите исходные данные:
Длина (см) → ***9***
Ширина (см) → ***7.5***
Высота (см) → ***5***

Объем: 337.50 куб.см.

Тема 4.2. Основы PHP

Лабораторная работа

Тема «Обработка запросов с помощью PHP»

Цель: рассмотреть основные понятия клиент-серверных технологий, охарактеризовать методы POST и GET, изучить механизм получения данных из HTML-форм и их обработки с помощью PHP. Проверить правильность выполнения программы создания формы для регистрации пользователей на сайте и отправке «универсального письма» всем зарегистрировавшимся. Разработать тестовую программу с отправкой результатов тестирования на сервер и их обработке с помощью PHP.

Протокол http и способы передачи данных на сервер

Internet построен по многоуровневому принципу, от физического уровня, связанного с физическими аспектами передачи двоичной информации и до прикладного уровня, обеспечивающего интерфейс между пользователем и сетью.

HTTP (HyperText Transfer Protocol, протокол передачи гипертекста) – это протокол прикладного уровня, разработанный для обмена гипертекстовой информацией в Internet.

HTTP предоставляет набор методов для указания целей запроса, отправляемого серверу. Эти методы основаны на дисциплине ссылок, где для указания ресурса, к которому должен быть применен данный метод, используется универсальный идентификатор ресурсов (Universal Resource Identifier) в виде местонахождения ресурса (Universal Resource Locator, URL) или в виде его универсального имени (Universal Resource Name, URN).

Сообщения по сети при использовании протокола HTTP передаются в формате, схожем с форматом почтового сообщения Internet (RFC-822) или с форматом сообщений MIME (Multipurpose Internet Mail Exchange).

Стоить отметить, что документации по протоколу HTTP огромное множество, при желании всегда можно более детально ознакомиться с ним самостоятельно.

Использование HTML-форм для передачи данных на сервер

Как передавать данные серверу? Для этого в языке HTML есть специальная конструкция – формы. Формы предназначены для того чтобы получать от пользователя информацию. Например, вам нужно знать логин и пароль пользователя для того, чтобы определить, на какие страницы сайта его можно допускать. Или вам необходимы личные

данные пользователя, чтобы была возможность с ним связаться. Формы как раз и применяются для ввода такой информации. В них можно вводить текст или выбирать подходящие варианты из списка. Данные, записанные в форму, отправляются для обработки специальной программе (например, скрипту на PHP) на сервере. В зависимости от введенных пользователем данных эта программа может формировать различные web-страницы, отправлять запросы к базе данных, запускать различные приложения и так далее.

Разберемся с синтаксисом HTML-форм. Возможно, многие с ним знакомы, но все же повторим основные моменты, поскольку это важно.

Итак, для создания формы в языке HTML используется тег FORM. Внутри него находится одна или несколько команд INPUT. С помощью атрибутов action и method тега FORM задаются имя программы, которая будет обрабатывать данные формы, и метод запроса, соответственно. Команда INPUT определяет тип и различные характеристики запрашиваемой информации. Отправка данных формы происходит после нажатия кнопки input типа submit.

Создадим форму для регистрации участников спецкурсов по программированию

PHP-код программы и результаты его выполнения изображены на рисунке 31.

```
<h2> Форма для регистрации </h2>
<form action="31_1.php" method=POST><!--создаем форму-->
<!--данные формы будет обрабатывать файл31_1.php при
    отправке запроса будет использован метод POST-->
Имя <br><input type=text name="first_name"
     value="ВведитеВаше имя"><br>
Фамилия <br><input type=text name="last_name"><br>
E-mail <br><input type=text name="email"><br>
<p>
Выберите спецкурс, который вы хотели посещать: <br>
<input type=radio name="kurs" value="PHP">PHP<br>
<input type=radio name="kurs" value="Web-дизайн: HTML">Web-дизайн: HTML<br>
<input type=radio name="kurs" value="Web-дизайн: Flash">Web-дизайн: Flash<br>
<p> Что вы хотите, чтобы мы знали о вас? <br>
<textarea name="comment" cols=32 rows=5></textarea>
<p><input name="confirm" type=checkbox
checked>Подтвердить получение <br>
<input type=submit value="Отправить">
<input type=reset value="Отменить">
</form>
```

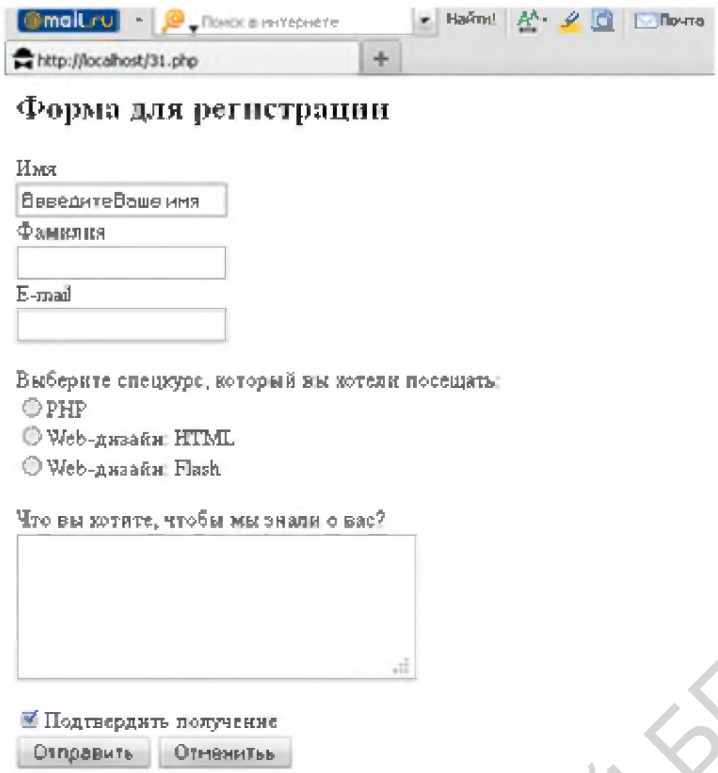


Рисунок 31

Метод GET

При отправке данных формы с помощью метода GET содержимое формы добавляется к URL после знака вопроса в виде пар имя=значения, объединенных с помощью амперсанта &:

action?namel=valuel&name2=value2&name3=value3

Здесь action – это URL-адрес программы, которая должна обрабатывать форму (это либо программа, заданная в атрибуте action тега form, либо сама текущая программа, если этот атрибут опущен). Имена namel, name2, name3 соответствуют именам элементов формы, а value1, value2, value3 – значениям этих элементов. Все специальные символы, включая = и &, в именах или значениях этих параметров будут опущены. Поэтому не стоит использовать в названиях или значениях элементов формы эти символы и символы кириллицы в идентификаторах.

Если в поле для ввода ввести какой-нибудь служебный символ, то он будет передан в его шестнадцатеричном коде, например, символ $ заменится на %24. Так же передаются и русские буквы.

Для полей ввода текста и пароля (это элементы input с атрибутом type=text и type=password), значением будет то, что введет пользователь. Если пользователь ничего не вводит в такое поле, то в строке запроса будет присутствовать элемент name=, где name соответствует имени этого элемента формы.

У передачи данных методом GET есть один существенный недостаток – любой может подделать значения параметров. Поэтому не советуем использовать этот метод для доступа к защищенным паролем страницам, для передачи информации, влияющей на безопасность работы программы или сервера. Кроме того, не стоит применять метод GET для передачи информации, которую не разрешено изменять пользователю.

Несмотря на все эти недостатки, использовать метод GET достаточно удобно при отладке скриптов (тогда можно видеть значения и имена передаваемых переменных) и для передачи параметров, не влияющих на безопасность.

Индивидуальное задание:

Разработать серверный сценарий обработки одностраничной или многостраничной тестовой формы. Содержание теста взять из 4 лабораторной работы.

Глобальные переменные

Чтобы использовать внутри функции переменные, заданные вне ее, эти переменные нужно объявить как глобальные. Для этого в теле функции следует перечислить их имена после ключевого слова global. Например, как это показано в

```
1  <?php
2  $a=1;
3      function Test_g(){
4          global $a;
5          $a=$a*2;
6          echo 'в результате работы функции $a=',$a;
7      }
8      echo 'вне функции $a=',$a,', ';
9      Test_g();
10     echo"<br>";
11     echo 'вне функции $a=',$a,', ';
12     Test_g();
13 ?>
```

примере, изображенном на рисунке 41

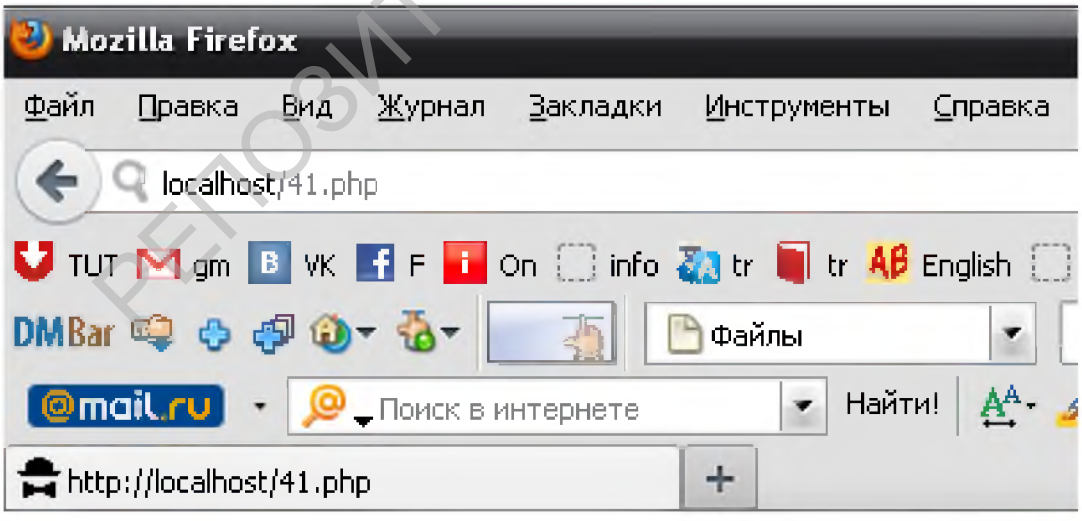


Рисунок 41 – Глобальные переменные

Когда переменная объявляется как глобальная, фактически создается ссылка на глобальную переменную. Поэтому такая запись эквивалентна следующей (массив

GLOBALScодержит все переменные, глобальные относительно текущей области видимости):

```
$var1=& $ GLOBALS[“var1”]
$var2=& $ GLOBALS[“var2”]
```

Это значит, например, что удаление переменной $var1 не удаляет глобальной пееменной$ GLOBALS[“var1”].

Тема 4.3. Основы MySQL. Язык запросов SQL. Формирование запросов к базе данных

Лабораторная работа

Тема «Построение запросов на выборку SQL»

Задание 1.

Изучите схему БД из файла **Описание базы данных db_telco.** Откройте БД и установите связи между таблицами.
Результат выполнения продемонстрируйте преподавателю.

Изучите справочные материалы из файлов **1Выборка данных_простые запросы**, **2Фильтрация данных**, **3Создание вычисляемых полей.**

Создайте в режиме SQL запросы 1 - 23, представленные в документах. Просмотрите запросы в оперативном режиме.
Результат выполнения продемонстрируйте преподавателю.

Задание 2.

Изучите схему БД из файла **Описание базы данных db_sales.** Откройте БД и установите связи между таблицами.
Результат выполнения продемонстрируйте преподавателю.

Создайте в режиме SQL запросы 1 - 7 по условию из файла **Задание 2**.
Продемонстрируйте преподавателю запросы в двух режимах.

Задание 3.

Изучите справочные материалы из файла **4_Объединение таблиц в запросах.**

Создайте запросы 11 – 15, представленные в документе. Просмотрите результаты отображения данных запроса на веб-странице.

Результат выполнения продемонстрируйте преподавателю.

РАЗДЕЛ КОНТРОЛЯ ЗНАНИЙ

Перечень вопросов

по дисциплине «Информационные системы и сети» для студентов 4 курса дневной формы получения образования факультета физико-математического

1. ИНФОРМАЦИОННЫЕ СИСТЕМЫ НА БАЗЕ ОФИСНЫХ ТЕХНОЛОГИЙ

1. Классификации информационных систем. Информационные системы на базе офисных технологий.

2. Современные офисные приложения. Работа с офисными приложениями.

Редактор MS Word как издательская система.

3. Основы типографики. Компьютерная верстка.

4. Объектная модель текстового документа. Типовые задачи, инструменты и методы работы с текстовым документом.

5. Редактор MS Word как издательская система. Создание и использование стилей. Автоматизация работы.

6. Создание ссылок, оглавлений, списков литературы, предметных указателей.

7. Создание электронного справочника. Публикация в формате PDF.

8. Информационные системы на основе электронных таблиц.

9. Электронные таблицы как инструмент обработки данных. Типовые задачи, инструменты и методы обработки и представления данных.

10. Электронные таблицы как инструмент обработки данных. Визуализация данных.

11. Электронные таблицы как инструмент обработки данных. Создание и применение фильтров, форм, сводных таблиц, шаблонов.

12. Электронные таблицы как инструмент обработки данных. Использование элементов управления для создания интерактивного пользовательского интерфейса.

13. Офисное программирование. Объектные модели офисных приложений. Электронные таблицы как среда программирования.

14. Электронные таблицы как среда программирования. Технология макросов. Запись и запуск макросов.

15. Основы языка Visual Basic for Application (VBA). Типы данных. Переменные, константы. Процедуры. Функции.

16. VBA. Базовые алгоритмические конструкции.

17. Электронные таблицы как среда программирования. Редактор VBA. Интерфейс.

18. Основы VBA. Элементы управления. События. Использование VBA для решения практических задач.

19. Базы данных. Основные понятия и концепции. Модели данных. Инфологические модели. Даталогические модели. Реляционная модель.

20. Реляционные базы данных. Логическое и физическое проектирование баз данных. Проектирование структуры базы данных. Нормализация.

21. Язык запросов SQL. Основные понятия. Выражения и операторы. Функции.

22. Базы данных. Построение и выполнение запросов с помощью SQL.

23. Системы управления базами данных (СУБД). Визуальные средства проектирования. СУБД MS Access. Интерфейс. Объекты Access.

24. СУБД MS Access. Инструменты и методы разработки таблиц, форм, запросов, отчетов. Использование конструкторов и мастеров.

25. СУБД MS Access. Создание таблиц базы данных. Ввод и редактирование данных. Связывание таблиц.

26. СУБД MS Access. Разработка и использование форм. Построение запросов. Создание отчетов.

2. ОСНОВЫ WEB-КОНСТРУИРОВАНИЯ

1. Языки гипертекстовой разметки. (x)HTML.
2. Структура web-документа. Основные теги. Атрибуты тегов.
3. Структура web-документа. Метатеги.
4. Физическая разметка web-документов. Форматирование текста.
5. Гипертекстовая разметка документов. Списки. Таблицы. Табличная верстка.
6. Гипертекстовая разметка документов. Гиперссылки. Изображения на web-страницах.
7. Гипертекстовая разметка документов. Фреймы. Iframe.
8. Инструменты и методы разработки web-страниц и сайтов. Разметка web-документов с помощью текстового редактора. Разработка фрагмента сайта с помощью визуального web-редактора.
9. Основы web-дизайна. Логическая разметка web-документов.
10. Стили CSS. Способы подключения. Правила построения. Селекторы.
11. Стили CSS. Блочные и строчные элементы. Блочная модель. Позиционирование.
12. Информационная модель сайта. Безтабличная верстка. Навигация.
13. Стили CSS. Текст на web-страницах.
14. Графика в Internet. Подготовка графических элементов web-страниц.

Оптимизация графики для web-страниц.

15. Разработка фрагмента сайта с использованием таблиц стилей.

16. Язык XML. Семантика и синтаксис. Структура и содержимое XML-документа. Правила разметки.

17. Визуализация XML-документа. Представление XML-документа средствами CSS.

18. Язык XML. Представление и преобразование данных средствами XSL и XSLT.

19. Развитие языков и технологий разметки. Стандарт HTML5. Новые элементы и возможности. Семантические элементы.

20. Стандарт HTML5. Мультимедиа в Internet. Использование звука и видео. Геолокация.

21. CSS3. Новые элементы и возможности. Визуальные эффекты.

22. CSS3. Трансформации. Эффекты анимации.

23. Интерактивные web-страницы. Формы. Методы форм: GET, POST. Элементы форм: Input.

24. Элементы форм: Textarea, Select. Новые элементы и свойства форм в HTML5.

25. JavaScript. Алфавит, синтаксис, семантика. Типы данных. Переменные и выражения. Функции.

26. JavaScript. Базовые алгоритмические конструкции. Объекты. События. Объектно-событийная модель JavaScript.

ВСПОМОГАТЕЛЬНЫЙ РАЗДЕЛ

Учреждение образования
«Белорусский государственный педагогический
университет имени Максима Танка»

УТВЕРЖДАЮ
Проректор по учебной и информационно-
аналитической работе БГПУ
_____ В.М.Зеленкевич
_____ 2016 г.

Регистрационный № УД-_____/ уч.

ИНФОРМАЦИОННЫЕ СИСТЕМЫ И СЕТИ

**Учебная программа учреждения высшего образования
по учебной дисциплине для специальности
1-02 05 02 Физика и информатика**

2016 г.

Учебная программа составлена на основе Образовательного стандарта высшего образования первая ступень специальность 1-02 05 02 Физика и информатика (ОСВО 1-02 05 02-2013) и Учебного плана специальности (регистрационный № 39-2013/у от 25.07.2013 г.)

СОСТАВИТЕЛИ:

С.В. Вабищевич, доцент кафедры информатики и методики преподавания информатики учреждения образования «Белорусский государственный педагогический университет имени Максима Танка», кандидат педагогических наук, доцент;

Г.А. Заборовский, доцент кафедры информатики и методики преподавания информатики учреждения образования «Белорусский государственный педагогический университет имени Максима Танка», кандидат физ.-мат. наук, доцент.

С.И. Зенько, заведующий кафедрой информатики и методики преподавания информатики учреждения образования «Белорусский государственный педагогический университет имени Максима Танка», кандидат педагогических наук, доцент

РЕЦЕНЗЕНТЫ:

кафедра "Информационные технологии" Республиканского института инновационных технологий учреждения образования "Белорусский национальный технический университет";

О.Л. Сапун, заведующий кафедрой экономической информатики учреждения образования "Белорусский государственный аграрный технический университет", кандидат педагогических наук, доцент

РЕКОМЕНДОВАНА К УТВЕРЖДЕНИЮ:

Кафедрой информатики и методики преподавания информатики
(протокол № 10 от 26.05.2016 г.);
Заведующий кафедрой _____ С.И. Зенько

Научно-методическим советом БГПУ
(протокол № 8 от 15.06.2016 г.).

Оформление учебной программы и сопровождающих ее материалов действующим требованиям Министерства образования Республики Беларусь соответствует

Методист учебно-методического управления БГПУ

_____ С.А. Стародуб

Ответственный за редакцию: С.И. Зенько
Ответственный за выпуск: С.И. Зенько

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Владение технологиями разработки и использования информационных систем является важным элементом профессиональной подготовки преподавателя информатики и неотъемлемым компонентом его будущей профессиональной деятельности.

Цель учебной дисциплины «Информационные системы и сети» - подготовка будущего преподавателя физики и информатики к разработке и использованию информационных систем и ресурсов Интернет.

Задачи изучения дисциплины:

- формирование теоретических знаний и практических умений разработки и использования информационных систем на основе офисных технологий;
- формирование навыков разработки и использования ресурсов Интернет;
- освоение языков и технологий веб-программирования;
- формирование навыков разработки распределенных информационных систем.

Место учебной дисциплины в системе подготовки специалиста

Изучение учебной дисциплины «Информационные системы и сети» опирается на основные академические, социально-личностные и профессиональные компетенции, сформированные у студентов при изучении дисциплин «Технологии программирования и методы алгоритмизации», «Компьютерная графика и мультимедиа», «Архитектура и программное обеспечение вычислительных систем», «Методика преподавания информатики».

Профессиональные компетенции студента

Учебная дисциплина «Информационные системы и сети» входит в государственный компонент специальных дисциплин, что определяет роль данной дисциплины в профессиональной подготовке будущего учителя информатики. Изучение учебной дисциплины «Информационные системы и сети» должно обеспечить формирование у студентов академических, социально-личностных и профессиональных компетенций.

Требования к академическим компетенциям

Специалист должен:

- АК-1. Уметь применять базовые научно-теоретические знания для решения теоретических и практических задач.
- АК-2. Владеть системным и сравнительным анализом.
- АК-4. Уметь работать самостоятельно.
- АК-5. Быть способным порождать новые идеи (обладать креативностью).
- АК-6. Владеть междисциплинарным подходом при решении проблем.
- АК-7. Иметь навыки, связанные с использованием технических устройств, управлением информацией и работой с компьютером.
- АК-9. Уметь учиться, повышать свою квалификацию в течение всей жизни.

Требование к социально-личностным компетенциям

Специалист должен:

- СЛК-7. Быть способным к осуществлению самообразования и самосовершенствования профессиональной деятельности.

Требования к профессиональным компетенциям

Специалист должен быть способен:

Обучающая деятельность

- ПК-1. Эффективно реализовывать обучающую деятельность.
- ПК-3. Использовать оптимальные методы, формы и средства обучения.
- ПК-6. Организовывать самостоятельную работу обучающихся.

Воспитательная деятельность

- ПК-11. Формировать базовые компоненты культуры личности воспитанника.

Развивающая деятельность

- ПК-14. Развивать навыки самостоятельной работы обучающихся с учебной, справочной, научной литературой и др. источниками информации.

Ценностно-ориентационная деятельность

- ПК-22. Осуществлять самообразование и самосовершенствование профессиональной деятельности

В результате изучения дисциплины обучаемый должен **знать:**

- типовые задачи и методы обработки информации в офисных приложениях;
- основы офисного программирования;
- модели представления данных и знаний, принципы построения баз данных;
- принципы функционирования глобальных компьютерных сетей;
- языки гипертекстовой разметки, стили CSS;
- инструменты и методы разработки web-страниц, основы web-дизайна;
- основы web-программирования, объектные модели браузера и документа;
- принципы построения распределенных информационных систем;
- социальные, этические и правовые аспекты информационных систем.

Обучаемый должен **уметь:**

- решать практические задачи обработки информации в офисных приложениях; использовать программирование для автоматизации офисных приложений;
- разрабатывать и администрировать базы данных с применением языка запросов SQL и различных технологий доступа;
- разрабатывать простые информационно-справочные системы;
- разрабатывать web-страницы с помощью различных инструментов и методов;
- создавать web-сайты с использованием Java Script и PHP;
- использовать технологии web-программирования для разработки распределенных информационных систем.

Обучаемый должен владеть:

- методами автоматизации работы в офисных приложениях;
- навыками программирования в среде офисных приложений;
- средствами разработки локальных и удаленных баз данных;
- навыками разработки интерактивных web-страниц и web-приложений;
- навыками управления web-сайтами и удаленными базами данных.

Структура содержания учебной дисциплины.

Дисциплина изучается на протяжении двух семестров и содержит четыре раздела. В первом разделе изучаются теоретические и практические аспекты построения информационных систем на основе офисных приложений, во втором рассматриваются основы гипертекстовой разметки документов и web-дизайна, в третьем – основы web-программирования, в четвертом - построение распределенных информационных систем на основе web-технологий.

Методы обучения.

В лекционном курсе рассматриваются современные концепции и подходы к построению информационных систем, обсуждаются возможности web-технологий. При чтении лекций особое внимание следует уделять демонстрации реальных информационных систем и мультимедийным презентациям, которые должны служить для будущих учителей образцом объяснения материала.

Лабораторные занятия направлены на формирование навыков практического использования полученных знаний при выполнении конкретных заданий. Методика их проведения должна содействовать развитию индивидуально-творческих способностей каждого студента и приобретению навыков самостоятельной работы. С целью подготовки будущего учителя к решению задач информатизации сферы образования рекомендуется предусмотреть задания по разработке информационных систем (баз данных и фрагментов сайтов) образовательного назначения.

Для управления самостоятельной работой рекомендуется использовать интерактивные учебные пособия, тренажеры, тестирующие программы и др. Текущий контроль ссуществляется при выполнении и сдаче лабораторных работ. Для промежуточной аттестации студентов предлагается тематический контроль (тестирование, коллоквиум). В качестве итогового контроля рекомендуется проведение двух экзаменов.

Распределение общего количества часов по формам обучения и семестрам

Специальность 1-02 05 02 Физика и информатика

Дневная форма получения высшего образования:

Всего на учебную дисциплину – 276 часов.

7 семестр – 68 часа аудиторных (22 часа – лекции, 46 часов – лабораторные занятия), 40 часов – самостоятельная работа.

8 семестр – 60 часов аудиторных (14 часов – лекции, 46 часов – лабораторные занятия), 36 часов – самостоятельная работа.

Всего за 7 и 8 семестры: 128 часов аудиторных (36 часов – лекции, 92 часов – лабораторные работы), 76 часов – самостоятельная работа.

Формы контроля – экзамен (7 семестр), экзамен (8 семестр).

СОДЕРЖАНИЕ УЧЕБНОГО МАТЕРИАЛА

РАЗДЕЛ 1. ИНФОРМАЦИОННЫЕ СИСТЕМЫ НА БАЗЕ ОФИСНЫХ ТЕХНОЛОГИЙ

Тема 1.1. Работа с офисными приложениями.

Классификации информационных систем. Современные офисные приложения. Основы типографики. Компьютерная верстка. Редактор MS Word как издательская система. Объектная модель текстового документа. Типовые задачи, инструменты и методы работы с текстовым документом. Создание и использование стилей. Автоматизация работы. Создание ссылок, оглавлений, списков литературы, предметных указателей. Создание электронного справочника. Публикация в формате PDF.

Информационные системы на основе электронных таблиц. Электронные таблицы как инструмент обработки данных. Типовые задачи, инструменты и методы обработки и представления данных. Визуализация данных. Создание и применение фильтров, форм, сводных таблиц, шаблонов. Использование элементов управления для создания интерактивного пользовательского интерфейса.

Тема 1.2. Офисное программирование.

Объектные модели офисных приложений. Электронные таблицы как среда программирования. Технология макросов. Запись и запуск макросов. Основы языка Visual Basic for Application (VBA). Типы данных. Переменные, константы. Процедуры. Функции. Базовые алгоритмические конструкции. Редактор VBA. Интерфейс. Элементы управления. События. Использование VBA для решения практических задач.

Тема 1.3. Базы данных.

Модели данных. Реляционная модель. Системы управления базами данных (СУБД). Реляционные базы данных. Логическое и физическое проектирование баз данных. Инфологическая модель предметной области. Проектирование структуры базы данных. Нормализация. Визуальные средства проектирования. СУБД MS Access. Интерфейс. Объекты Access. Инструменты и методы разработки таблиц, форм, запросов, отчетов. Использование конструкторов и мастеров. Создание таблиц базы данных. Ввод и редактирование данных. Связывание таблиц. Разработка и использование форм. Построение запросов. Условия и вычисления в запросах. Создание отчетов. Импорт, экспорт, преобразование данных. Транзакции и блокировки. Защита данных. Использование макросов. Язык запросов SQL. Типы данных. Выражения и операторы. Функции. Построение и выполнение запросов с помощью SQL.

Тема 1.4. Развитие информационных систем.

Представление данных и знаний. Базы знаний. Информационно-справочные системы. Геоинформационные системы. Информационно-поисковые системы. Механизмы поиска информации. Анализ текста докумен-

тов, индексирование. Экспертные системы. Тенденции развития информационных систем.

РАЗДЕЛ 2. ОСНОВЫ WEB-КОНСТРУИРОВАНИЯ

Тема 2.1. Гипертекстовая разметка документов.

Языки гипертекстовой разметки. (x)HTML. Структура web-документа. Основные теги. Атрибуты. Метатеги. Физическая разметка web-документов. Форматирование текста. Списки. Таблицы. Табличная верстка. Изображения. Гиперссылки. Фреймы. Iframe. Инструменты и методы разработки web-страниц и сайтов. Разметка web-документов с помощью текстового редактора. Разработка фрагмента сайта с помощью визуального web-редактора.

Тема 2.2. Основы web-дизайна.

Логическая разметка web-документов. Стили CSS. Способы подключения. Правила построения. Селекторы. Блочные и строчные элементы. Блочная модель. Позиционирование. Безтабличная верстка. Информационная модель сайта. Навигация. Эргономика. Текст на web-страницах. Графика в Internet. Подготовка графических элементов web-страниц. Оптимизация графики для web-страниц. Разработка фрагмента сайта с использованием таблиц стилей.

Тема 2.3. Развитие языков и технологий разметки.

Язык XML. Семантика и синтаксис. Структура и содержимое XML-документа. Правила разметки. Визуализация XML-документа. Представление XML-документа средствами CSS. Представление и преобразование данных средствами XSL и XSLT. Стандарт HTML5. Новые элементы и возможности. Семантические элементы. Новые элементы и свойства форм. Сокеты. Web-хранилища. Геолокация. Мультимедиа в Internet. Использование звука и видео. CSS3. Новые элементы и возможности. Трансформации. Эффекты анимации. Основы кроссбраузерной и кроссплатформенной верстки. Адаптивный дизайн. Разработка адаптивных web-ресурсов. Особенности разработки web-ресурсов для мобильных устройств. Разработка образовательного web-сайта.

РАЗДЕЛ 3. ОСНОВЫ WEB-ПРОГРАММИРОВАНИЯ

Тема 3.1. Основы JavaScript.

Языки сценариев. JavaScript. Алфавит, синтаксис, семантика. Типы данных. Операторы. Переменные и выражения. Функции. Базовые алгоритмические конструкции. Объекты. События. Объектно-событийная модель JavaScript. Встроенные объекты языка JavaScript. Объект Math. Работа с массивами. Работа со строками. Работа с датой и временем. Создание и использование функций и объектов. Формат JSON.

Тема 3.2. Объектные модели браузера и документа.

Объектная модель браузера. Объекты Navigator, Screen. Получение информации о пользователе. Объектная модель документа (DOM). Доступ к объектам web-страницы. Объект Window. Управление окнами. Объект Document. Создание динамических страниц. Разработка интерактивных web-ресурсов. Формы.

Методы форм: Get, Post. Элементы форм: Input, Textarea, Select. Обработка форм на web-странице. Проверка вводимых данных. Регулярные выражения. Работа с графикой. Объект Canvas. Рисование на холсте. Динамические эффекты. Формат SVG. Использование JavaScript для разработки образовательных ресурсов. Разработка систем тестирования.

Тема 3.3. Основы серверного программирования.

Клиентские и серверные приложения. Протокол HTTP. Принципы работы с web-сервером. Язык PHP. Алфавит, синтаксис, семантика. Переменные и константы. Типы данных. Разработка PHP-сценариев. Базовые алгоритмические конструкции. Функции. Массивы. Строки. . Объектная модель PHP. Регулярные выражения в языке PHP. Взаимодействие клиента и сервера. Отправка данных HTML-форм на сервер методами GET и POST. Получение и обработка данных средствами PHP. Формирование динамических web-страниц на сервере. Технология SSI. Работа с графикой. Рисование фигур. Использование растровых изображений. Работа с файловой системой. Чтение и запись данных. Счетчики посещений сайта. Cookies. Загрузка файлов на сервер. Разработка серверных web-приложений. Авторизация доступа. Механизм сессий. Создание web-галереи.

РАЗДЕЛ 4. РАСПРЕДЕЛЕННЫЕ ИНФОРМАЦИОННЫЕ СИСТЕМЫ

Тема 4.1. Проектирование и использование удаленных баз данных.

Концепции распределенных информационных систем (ИС). Web-приложения и web-сервисы. Средства и методы разработки web-приложений. Управление удаленными базами данных. Сервер баз данных MySQL. Создание и администрирование базы данных с помощью phpMyAdmin (MySQL Work-Bench). Работа с таблицами. Добавление, извлечение, редактирование, поиск и удаление данных. Работа с базой данных средствами PHP. Функции доступа к базе данных. Использование PHP и MySQL для разработки ИС. Системы опроса и тестирования. Разработка системы управления web-сайтом (CMS). Создание стилей и шаблонов оформления web-страниц. Разработка интерфейса пользователя. Разработка интерфейса администратора.

Тема 4.2. Перспективные технологии и средства разработки web-приложений.

Тенденции и перспективы развития информационных систем и коммуникационных технологий. Использование библиотек и шаблонов. Библиотека jQuery. Обмен данными между браузером и сервером без перезагрузки web-страницы. Технология Ajax. Гостевая книга. Чат. Фреймворки. Адаптивный дизайн с помощью Bootstrap. Модель MVC. AngularJS. Шаблонизаторы. Обзор открытых CMS.

Основы безопасности при разработке и использовании ИС. Скриптинг. SQL-инъекции. Проверка вводимых данных. Шифрование. Социальные, этические и правовые аспекты разработки и использования ИС. Интеллектуальная собственность.

УЧЕБНО-МЕТОДИЧЕСКАЯ КАРТА ДИСЦИПЛИНЫ
для дневной формы получения образования

Номер раздела, темы	Название раздела, темы	Количество аудиторных часов				Количество часов самостоятельная работа	Форма контроля знаний
		Лекции	Лабораторные занятия	Практические занятия	Иное		
1	2	3	4	5	6	7	8
1	**Информационные системы на базе офисных технологий**	**12**	**26**			**20**	
1.1	**Работа с офисными приложениями**	**6**	**8**			**8**	
	Классификации информационных систем. 1. Современные офисные приложения. 2. Основы типографики.	2					
	Редактор MS Word как издательская система. 1. Объектная модель текстового документа. 2. Типовые задачи, инструменты и методы работы с текстовым документом.	2				2	
	Компьютерная верстка сложных документов. 1. Создание и использование стилей. Автоматизация работы. 2. Создание ссылок, оглавлений, списков литературы, предметных указателей. 3. Создание электронного справочника. Публикация в формате PDF.		4			2	Выполнение контрольных заданий (КЗ). Защита отчета по лаб работе
	Информационные системы на основе электронных таблиц. 1. Электронные таблицы как инструмент обработки данных. 2. Типовые задачи, инструменты и методы обработки и представления данных.	2				2	
	Автоматизация работы со сложными документами 1. Визуализация данных. 2. Создание и применение фильтров, форм, сводных таблиц, шаблонов. 3. Использование элементов управления для создания интерактивного пользовательского интерфейса.		4			2	Выполнение КЗ. Защита отчета по лаб работе

1.2	**Офисное программирование**	**2**	**8**			**4**	
	Электронные таблицы как среда программирования. 1. Объектные модели офисных приложений. 2. Технология макросов. 3. Основы языка Visual Basic for Application (VBA). Типы данных.	2					
	Основы программирование на VBA. 1. Запись и запуск макросов. 2. Редактор VBA. Интерфейс. Элементы управления. События. 3. Переменные, константы. Процедуры. Функции. **4.** Базовые алгоритмические конструкции.		4			2	Выполнение КЗ. Защита отчета по лаб работе
	Использование VBA для решения практических задач. 1. Решение задач с контролем результатов. 2. Создание интерактивных демонстраций. 3. Создание простейших тестов.		4			2	Рейтинговая контрольная работа № 1
1.3	**Базы данных.**	**2**	**8**			**6**	
	Логическое и физическое проектирование баз данных. 1. Модели данных. Реляционная модель. 2. Инфологическая модель предметной области. 3. Реляционные базы данных. Системы управления базами данных (СУБД). 4. Нормализация. 5. Язык запросов SQL. Типы данных. Выражения и операторы. Функции.	2				2	
	Создание базы данных. Использование конструкторов и мастеров. 1. СУБД MS Access. Интерфейс. Объекты Access. 2. Проектирование структуры базы данных. 3. Создание таблиц базы данных. Ввод и редактирование данных. 4. Связывание таблиц. 5. Разработка и использование форм.		4			2	Выполнение КЗ. Защита отчета по лаб работе
	Инструменты и методы разработки запросов, отчетов. 1. Построение запросов. Условия и вычисления в запросах. 2. Создание отчетов. 3. Импорт, экспорт, преобразование данных. Защита данных. 4. Построения и выполнение запросов с помощью SQL		4			2	Рейтинговая контрольная работа № 2

1.4	**Развитие информационных систем.**	**2**	**2**			**2**	
	Тенденции развития информационных систем. 1. Представление данных и знаний. Базы знаний. 2. Информационно-справочные системы. Геоинформационные системы. 3. Информационно-поисковые системы. 4. Механизмы поиска информации. Анализ текста документов, индексирование. 5. Экспертные системы.	2				2	
	Разработка информационно-справочной системы образовательного назначения		2				
2	**Основы web-конструирования**	**10**	**20**			**20**	
2.1	**Гипертекстовая разметка документов.**	**2**	**4**			**4**	
	Языки гипертекстовой разметки. (x)HTML. 1. Структура web-документа. Основные теги. Атрибуты. Метатеги. 2. Физическая разметка web-документов. Форматирование текста. Списки. 3. Таблицы. Табличная верстка. Изображения. Гиперссылки. 4. Фреймы. Iframe.	2				2	
	Инструменты и методы разработки web-страниц и сайтов. 1. Разметка web-документов с помощью текстового редактора. 2. Разработка фрагмента сайта с помощью визуального web-редактора.		4			2	Выполнение КЗ. Защита отчета по лаб работе
2.2	**Основы web-дизайна.**	**2**	**4**			**4**	
	Логическая разметка web-документов. 1. Стили CSS. Способы подключения. Правила построения. Селекторы. 2. Блочные и строчные элементы. Блочная модель. Позиционирование. 3. Информационная модель сайта. Навигация. 4. Эргономика. Графика в Internet.	2				2	
	Разработка фрагмента сайта с использованием таблиц стилей 1. Логическая разметка страниц. 2. Безтабличная верстка. 3. Подготовка графических элементов web-страниц. Оптимизация графики.		4			2	Выполнение КЗ. Защита отчета по лаб работе

2.3	**Развитие языков и технологий разметки.**	**6**	**12**			**12**	
	Язык XML. 1. Семантика и синтаксис. 2. Структура и содержимое XML-документа. Правила разметки. 3. Представление и преобразование XML-документов.	2				2	
	Визуализация XML-документа. 1. Представление XML-документа средствами CSS. 2. Представление и преобразование данных средствами XSL и XSLT.		4			2	Выполнение КЗ. Защита отчета по лаб работе
	Стандарт HTML5. Новые элементы и возможности 1. Семантические элементы. 2. Новые элементы и свойства форм. 3. Сокеты. Web-хранилища. Геолокация.	2				2	
	Новые элементы и возможности HTML5. 1. Использование элементов форм. 2. Использование звука и видео		4			2	Выполнение КЗ. Защита отчета по лаб работе
	Основы кроссбраузерной и кроссплатформенной верстки. 1. CSS3. Новые элементы и возможности 2. Адаптивный дизайн. 3. Особенности разработки web-ресурсов для мобильных устройств.	2				2	
	Разработка адаптивных web-ресурсов 1. Трансформации. 2. Эффекты анимации. 3. Разработка образовательного web-сайта.		4			2	Рейтинговая контрольная работа № 3
			2				
	Итого за 7 семестр	**22**	**46**			**40**	**Экзамен**

3	**Основы web-программирования**	**8**	**32**			**20**	
3.1	**Основы JavaScript.**	**2**	**8**			**4**	
	Языки сценариев. JavaScript. 1. Алфавит, синтаксис, семантика. 2. Типы данных. Переменные и выражения. 3. Функции. Объекты. События. 4. Объектно-событийная модель JavaScript.	2				2	
	Встроенные объекты языка JavaScript. 1. Объект Math. 2. Базовые алгоритмические конструкции.		4				Защита отчета по лаб работе
	Создание и использование функций и объектов. Формат JSON. 1. Работа с массивами. 2. Работа со строками. 3. Работа с датой и временем.		4			2	Выполнение КЗ. Защита отчета по лаб работе
3.2	**Объектные модели браузера и документа.**	**2**	**12**			**6**	
	Объектная модель браузера. Объектная модель документа (DOM). 1. Доступ к объектам web-страницы. 2. Разработка интерактивных web-ресурсов. Формы. 3. Работа с графикой. Объект Canvas. Формат SVG. 4. Регулярные выражения.	2					
	Управление окнами. Создание динамических страниц. 1. Объекты Navigator, Screen. Получение информации о пользователе 2. Объект Window. 3. Объект Document.		4			2	Выполнение КЗ. Защита отчета по лаб работе
	Обработка форм на web-странице. 1. Методы форм: Get, Post. 2. Элементы форм: Input, Textarea, Select. 3. Проверка вводимых данных.		4			2	Выполнение КЗ. Защита отчета по лаб работе
	Использование JavaScript для разработки образовательных ресурсов. 1. Объект Canvas. Рисование на холсте. Динамические эффекты. 2. Использование регулярных выражений. 3. Разработка систем тестирования.		4			2	Рейтинговая контрольная работа № 4

3.3	**Основы серверного программирования.**	**4**	**12**			**10**	
	Клиентские и серверные приложения. 1. Принципы работы с web-сервером. Протокол HTTP. 2. Язык PHP. Алфавит, синтаксис, семантика. 3. Переменные и константы. Типы данных. Функции. 4. Объектная модель PHP.	2				2	
	Разработка PHP-сценариев. 1. Базовые алгоритмические конструкции. 2. Работа с массивами. Работа с строками. 3. Регулярные выражения в языке PHP.		4			2	Защита отчета по лаб работе
	Взаимодействие клиента и сервера. 1. Отправка данных HTML-форм на сервер методами GET и POST. 2. Получение и обработка данных средствами PHP.		4			2	Выполнение КЗ. Защита отчета по лаб работе
	Формирование динамических web-страниц на сервере. 1. Технология SSI. 2. Работа с графикой. 3. Работа с файловой системой. 4. Авторизация доступа. Механизм сессий.	2				2	
	Разработка серверных web-приложений 1. Рисование фигур. Использование растровых изображений. 2. Чтение и запись данных. Счетчики посещений сайта. Cookies. 3. Загрузка файлов на сервер. Создание web-галереи.		4			2	Рейтинговая контрольная работа № 5
4	**Распределенные информационные системы**	**6**	**14**			**16**	
4.1	**Проектирование и использование удаленных баз данных.**	**2**	**6**			**6**	
	Управление удаленными базами данных. 1. Концепции распределенных информационных систем. 2. Web-приложения и web-сервисы. 3. Средства и методы разработки web-приложений. 4. Сервер баз данных MySQL.	2				2	

	Использование PHP и MySQL для разработки ИС 1. Создание и администрирование базы данных с помощью phpMyAdmin (MySQL WorkBench). 2. Работа с таблицами. Добавление, извлечение, редактирование, поиск и удаление данных. 3. Работа с базой данных средствами PHP. Функции доступа к базе данных. 4. Системы опроса и тестирования.		4			2	Выполнение КЗ. Защита отчета по лаб работе
	Разработка системы управления web-сайтом (CMS). 1. Создание стилей и шаблонов оформления web-страниц. 2. Разработка интерфейса пользователя. 3. Разработка интерфейса администратора.		2			2	Выполнение КЗ. Защита отчета по лаб работе
4.2.	**Перспективные технологии и средства разработки web-приложений.**	**4**	**8**			**10**	
	Тенденции и перспективы развития информационных систем и коммуникационных технологий. 1. Технология Ajax. 2. Фреймворки. Шаблонизаторы. 3. Обзор открытых CMS.	2				4	
	Использование библиотек и шаблонов. 1. Библиотека jQuery. 2. Обмен данными между браузером и сервером без перезагрузки web-страницы. 3. Гостевая книга. Чат.		4			2	Выполнение КЗ. Защита отчета по лаб работе
	Использование фреймворков. 1. Адаптивный дизайн с помощью Bootstrap. 2. Модель MVC. AngularJS.		4			2	Рейтинговая контрольная работа № 6
	Основы безопасности и социальные аспекты разработки и использования ИС. 1. Скриптинг. SQL-инъекции. Шифрование. 2. Этические и правовые аспекты разработки и использования ИС	2				2	
	Итого за 8 семестр	**14**	**46**			**36**	**Экзамен**
	Итого за 7–8 семестры	**36**	**92**			**76**	

ИНФОРМАЦИОННО-МЕТОДИЧЕСКАЯ ЧАСТЬ

ЛИТЕРАТУРА

Основная

1. Колисниченко Д. Н., PHP и MySQL. Разработка Web-приложений. - СПб.: БХВ-Петербург. 2015. – 592 с.
2. Кузин А.В., Базы данных. ./А.В.Кузин, С.В.Левонисова . — М. “Академия”, 2012. — 320 с.
3. Никсон Р., Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript , CSS и HTML5- СПб.: Питер, 2015. — 688 с.
4. Пирогов, В.Ю. Информационные системы и базы данных: организация и проектирование. – СПб. : БХВ-Петербург, 2009. – 528с.
5. Прохоренок Н.А. HTML, JavaScript, PHP и MySQL. Джентельменский набор web-мастера – СПб.: БХВ-Петербург. 2015. – 768 с.
6. Слепцова Л.Д. Программирование на VBA в Microsoft Office 2010. — М. “Вильямс”, 2010. — 432 с.
7. Хоган Б. HTML5 и CSS3. Веб-разработка по стандартам нового поколения. СПб.: Питер, 2014. — 320 с
8. Фрейн Б. HTML5 и CSS3. Разработка сайтов для любых браузеров и устройств. — СПб.: Питер, 2014. — 304 с.

Дополнительная

1. Бекаревич Ю. Б. Самоучитель Access 2010. / Ю. Б. Бекаревич, Н. В. Пушкина. — СПб.: БХВ-Петербург, 2011. — 432 с.:
2. Вальтер Ш. Создание приложений для Windows 8 с помощью HTML5 и JavaScript. - М.: ДМК Пресс, 2013. - 344 с.
3. Емельянова Н.З. Основы построения автоматизированных информационных систем../Н.З.Емельянова, Т.Л.Партыка, И.И.Попов - М.: ИНФРА-М, 2007. - 416 с.
4. Закас Н., Java Script для профессиональных веб-разработчиков. - СПБ, Питер. 2015.960 с.
5. Кит Вуд. Расширение библиотеки jQuery – М.: ДМК Пресс, 2014. – 400 с.
6. Клименко Б. Microsoft Word: комфортная работа с помощью макросов./ Б.Клименко, М.Розенберг – БХВ-Петербург, 2006. 474 с.
7. Уокенбах Дж. Excel 2010: профессиональное программирование на VBA.:— М. “Вильямс”, 2012. — 944 с.
8. Феличи Дж. Типографика: шрифт, верстка, дизайн. – СПб.: БХВ, 2014.- 496 с.

Электронные учебные ресурсы

1. Опорные конспекты лекций [Электронный ресурс].
   Режим доступа: S:\COURSE04isis \um\
2. Материалы к лабораторным работам. [Электронный ресурс].
   Режим доступа: S:\COURSE04\isis\labs\

Методические рекомендации по организации и выполнению самостоятельной работы студентов

Содержание и формы самостоятельной работы студентов разрабатываются кафедрами в соответствии с целями и задачами подготовки специалиста. Для управления самостоятельной работой рекомендуется использовать электронные средства обучения, тестирующие программы. Текущий контроль осуществляется в ходе выполнения и защиты лабораторных работ.

Перечень используемых средств диагностики результатов учебной деятельности

Для оценки достижений и уровня знаний студента при изучении дисциплины целесообразно применить комплексный инструментарий, который включает:

- контроль выполнения внеаудиторных заданий;
- отчеты о самостоятельной работе;
- контроль ведения рабочих тетрадей;
- выборочный отчет по внеаудиторным заданиям;
- устный экспресс контроль по блоку тем;
- устное собеседование, коллоквиум;
- компьютерное тестирование;
- отчет о выполнении заданий самостоятельного цикла;
- контроль выполнения самостоятельной работы по темам;
- зачетное занятие с учетом результатов рейтинг-листа, составленного по данным прохождения дисциплины в семестре.

ПРОТОКОЛ СОГЛАСОВАНИЯ УЧЕБНОЙ ПРОГРАММЫ

Название учебной дисциплины, с которой требуется согласование	Название кафедры	Предложения об изменениях в содержании учебной программы учреждения высшего образования по учебной дисциплине	Решение, принятое кафедрой, разработавшей учебную программу (с указанием даты и номера протокола)
Методика преподавания физики	Кафедра физики и методики преподавания физики	Использовать согласованную терминологию при рассмотрении вопросов, связанных с использованием электронных таблиц для решения практических задач (тема 1.2,), а также разработкой образовательных web-ресурсов (темы 2.1, 2.3, 3.2)	Протокол № 10 от 26.05.2016 г.