ФЕДЕРАЛЬНОЕ АГЕНТСТВО ВОЗДУШНОГО ТРАНСПОРТА

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ
БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ ГРАЖДАНСКОЙ АВИАЦИИ» (МГТУ ГА)**

Кафедра прикладной математики

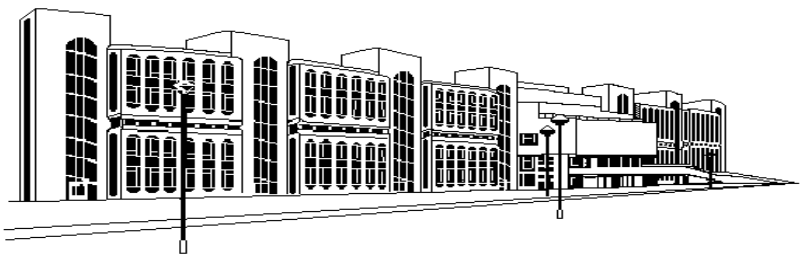
В.М. Коновалов

Проектирование программного обеспечения

Пособие

по выполнению лабораторных работ

*для студентов IV курса
направления 231300
дневного обучения*



Москва - 2014

Рецензент: к.т.н., доцент Рощин А.Г.

Коновалов В.М.

Проектирование программного обеспечения: пособие по выполнению лабораторных работ. – М.: МГТУ ГА, 2014. – 32с.

Методическое пособие издается в соответствии с рабочей программой учебной дисциплины «Проектирование программного обеспечения» по Учебному плану для студентов IV курса направления 231300 дневного обучения.

Рассмотрено и одобрено на заседаниях кафедры 21.01.14 г. и методического совета 21.01.14 г.

СОДЕРЖАНИЕ

ПРЕДИСЛОВИЕ

Данная методическая разработка определяет содержание и последовательность выполнения студентами лабораторного практикума. Основная задача состоит в приобретении практических навыков работы в инструментальных системах Visual Basic for Applications (VBA) – для разработки **интерактивных** пользовательских приложений на базе MS Office, и Visual Studio .NET – для проектирования приложений любых типов: консольных, Windows-приложений, Web-приложений, Web-сервисов и др.

VBA – это средство **офисного** программирования. Но в Office многие задачи, связанные с автоматизацией действий пользователя, можно решать без всякого программирования. Тем не менее, класс задач, решение которых может быть получено "руками", ограничен, и все что нельзя сделать руками, можно сделать программно. Для программирования используется VBA, являющийся расширением ядра языка VB (Visual Basic) объектами, определяющими офисные приложения (т.е. входящие в состав пакета MS Office).

Подобно другим средствам, VBA позволяет создавать полностью автоматические программные продукты, которые можно использовать, в частности, при оформлении документов (подготовка текстов), при вычислениях и анализе данных таблиц (электронных таблиц), при манипулировании данными, содержащимися в таблицах базы данных. Перечисленные функции по обработке данных реализуются офисными прикладными программами: Word, Excel, Access и др. VBA, в этом случае, является инструментом, расширяющим их функциональные возможности.

При решении задач с помощью VBA, все создаваемые программные компоненты объединяются в одно целое, называемое **проектом**. Проекты VBA выполняются совместно с другими приложениями. **Офисное приложение**, в котором разрабатывается и выполняется проект VBA, называется **основным**. Например, можно создать проект VBA, который работает вместе с Microsoft Excel. В этом случае Excel является основным приложением. Не используя основное приложение, нельзя построить приложение VBA. В этом – основное различие между VBA и "чистым" языком программирования Visual Basic (VB), с помощью которого можно создать **полностью** самостоятельное приложение. VBA учитывает специфические особенности основного приложения. Поэтому при разработке функционально ориентированных приложений, для которых можно явно определить основное приложение, предпочтение следует отдавать VBA, а не VB, что **снизит трудоемкость** программирования.

Для реализации цели обучения в рамках настоящего лабораторного практикума достаточно ограничиться основными приложениями, в качестве которых можно использовать офисные прикладные программы MS Excel, MS Access, выполняющие функции по работе с вычислениями и обработкой хранимых данных в базе данных.

Лабораторная работа 1. ИЗУЧЕНИЕ ПРИЁМОВ РАБОТЫ В ИНТЕГРИРОВАННЫХ ИНСТРУМЕНТАЛЬНЫХ СРЕДАХ ПРОЕКТИРОВАНИЯ

Продолжительность работы – **4 часа.**

Цель работы: изучить состав, структуру, назначение элементов управления, освоить приемы работы в средах проектирования программного обеспечения – VBE (Visual Basic Editor) и VS.NET(Visual Studio.NET)

1. Знакомство с инструментальной средой VBE (Visual Basic Editor)

1.1. Выбор основного приложения

На начальном этапе создания проекта необходимо выбрать **основное** (офисное) приложение для VBA-проекта. Если разрабатываемое Вами приложение (проект) будет применяться для настройки и автоматизации процесса или задачи в некотором офисном приложении, то, очевидно, следует использовать **это** приложение. В общем случае выбрать основное приложение не столь просто, как кажется. При выборе следует руководствоваться следующими принципами:

- в качестве основного приложения следует выбирать программу, которая будет использоваться для запуска проекта;
- в качестве основного приложения следует выбирать программу, в которой можно выполнить **большую** часть требуемых операций. Например, если требуется работать с диаграммами, производить вычисления, а также формировать документы, то лучше использовать в качестве основного приложения Excel, а не Word.

Разработка приложения в системе VBA выполняется практически полностью в редакторе VBE. Поэтому **инструментальной средой** является среда редактора VBE. В приложениях Excel и Word эти среды практически идентичны. В Access имеется специфика в структуре среды при сохранении всех ее инструментальных возможностей. Первоначальное знакомство с инструментальной средой можно начать, например, используя в качестве основного приложения – Excel.

1.2. Запуск редактора VBE

Для запуска редактора VBE выбрать в основном приложении (в Excel, в данном случае) команду **Сервис\Макрос\Редактор Visual Basic**. Сразу отметим, что практически всегда имеется альтернативная возможность отработки команды с помощью соответствующей пиктограммы на панели инструментов в верху экрана. Кроме того, некоторые названия в меню могут претерпевать изменения в различных версиях пакета MS Office. Последовательность же выполняемых действий и их сущность остаются практически неизменными.

Теперь следует изучить возможности по **конфигурированию среды**, а так же назначение и **способы работы с ее компонентами**. Конфигурирование выполняется с помощью команд основного меню редактора, в частности, путем отработки команд меню **Вид** и **Вставка**. В результате в окне редактора могут быть активизированы различные компоненты – рабочие окна, панели инструментов, в той конфигурации, какая необходима для эффективной работы.

Порядок выполнения лабораторной работы

Изучить основные действия, осуществляемые при работе:

● с окном проекта

Окно проекта – это специальное окно редактора VBE, в котором выводятся все элементы проекта в виде иерархической структуры, включающей все формы, модули кода и объекты основного приложения, например, рабочие листы книги Excel. В окне проекта выводится проект VBA каждого открытого в основном приложении документа.

● с окном свойств

Окно свойств используется для просмотра и задания свойств объектов проекта. С его помощью можно установить свойства и просмотреть их значения для любого компонента проекта и объекта основного приложения.

● с окном модуля

Разрабатываемые в VBA-проекте программы можно хранить в виде взаимосвязанных процедур в модуле проекта. Различают **стандартные** модули и **модули класса**. Код, используемый для создания объектов, их свойств и методов, содержится в модуле класса. В качестве имени модуля класса обычно используется имя созданного объекта. В стандартном модуле могут храниться любые процедуры, разрабатываемые пользователем для автоматизации работы приложения.

● с панелью элементов

Панель элементов позволяет разместить элементы управления на форме. Хотя окно с инструментами отображается автоматически при создании формы, может понадобиться скрыть его, чтобы освободить место на экране. Для вывода панели элементов на экран следует отработать команду **Вид\Панель элементов**.

● с формой

При разработке формы часто необходимо посмотреть на ее внешний вид. Для этого необходимо открыть форму и либо нажать клавишу <F5>, либо отработать команду **Запуск\Запуск подпрограммы/формы**. Для возврата в окно редактора просто закройте окно формы.

● выполнить операции импорта и экспорта файлов; переключения в основное приложение

Редактор VBE обеспечивает возможность импорта и экспорта компонентов приложения. Это дает возможность использовать имеющиеся данные в других приложениях и проектах Visual Basic.

Если имеется компонент проекта VBA, который необходимо включить в текущий проект, то следует сначала экспортировать компонент из исходного проекта, а затем импортировать его в текущий проект. Компоненты приложения Visual Basic сохраняются при экспорте в файлах, имеющих стандартные расширения: frm – для формы; cls – для модуля класса; bas – для модуля программы.

В процессе разработки проекта часто приходится переключаться в основное приложение, не закрывая проект. Для выполнения данной операции можно использовать несколько альтернативных способов.

● произвести запуск форм, процедур и макросов, сохранение изменений

До получения окончательной версии разрабатываемого приложения потребуется в процессе отладки неоднократно запускать на выполнение формы, процедуры и макросы, находясь в среде разработки – в среде редактора VBE.

● привести настройку инструментальной среды VBE в соответствие с условиями разрабатываемого проекта.

С помощью диалогового окна **Параметры** можно установить параметры работы редактора VBE. Для вывода окна **Параметры** отработайте команду **Сервис\Параметры**.

С помощью диалогового окна **VBAProject – свойства проекта** задаются свойства проекта. Войти в это диалоговое окно можно, отработав команду **Сервис\Свойства VBAProject**

2. Знакомство с инструментальной средой VS .NET (Visual Studio .NET)

Имена нынешнего поколения продуктов от Microsoft сопровождаются окончанием .Net (читается Dot Net), отражающим реалии современного глобального информационного мира, в котором **коммуникативная** составляющая любых программных продуктов начинает играть определяющую роль.

Инструментальная среда **Visual Studio .NET** базируется на разработанной корпорацией Microsoft платформе **.NET Framework** и представляет собой универсальный инструмент, с помощью которого можно создавать самые разнообразные приложения:

● Консольные приложения, позволяющие выполнять вывод на "консоль", то есть в окно командного процессора;

● Windows-приложения, использующие интерфейс Windows, включая формы с элементами управления на них;

● Web-приложения, представляющие собой web-страницы, которые могут просматриваться любым web-браузером;

● Web-службы (сервисы), представляющие собой распределенные приложения, которые позволяют обмениваться по Интернету практически любыми данными с использованием единого синтаксиса XML, независимо от того, какой язык программирования применялся при создании web-службы и на какой системе она размещена.

Такие приложения могут создаваться с использованием **разных** языков программирования, известных как языки группы .NET, в частности, поставляемых Microsoft вместе с Visual Studio .NET:

● Visual Basic .NET;
● Visual C# .NET (читается Си-шарп);
● Visual J# .NET (читается Джей-шарп);
● Visual C++ .NET;
● ASP.NET (для создания web-приложений).

Для работы с этими языками Visual Studio .NET предоставляет один и тот же **интерфейс IDE** (Integrated Development Environment) - интегрированную среду разработки. Здесь слово «интегрированная» означает, что в составе среды имеются все инструменты, необходимые для написания, редактирования, компиляции, отладки и запуска программ.

В программных продуктах **.NET** за этим именем стоит вполне конкретное содержание, которое предполагает, в частности, наличие открытых стандартов коммуникации, переход от создания монолитных приложений к созданию компонентов, допускающих повторное использование в разных средах и приложениях. Возможность повторного использования уже созданных компонентов и легкость расширения их функциональности - все это непременные атрибуты новых технологий. Важную роль в этих технологиях играет язык XML, ставший стандартом обмена сообщениями в сетях.

Сущность разработанных Майкрософт технологий поясняет схема (рис.1). Кратко рассмотрим архитектуру .NET Framework, а также место инструментальной среды Visual Studio .NET и разных систем программирования в этой архитектуре.

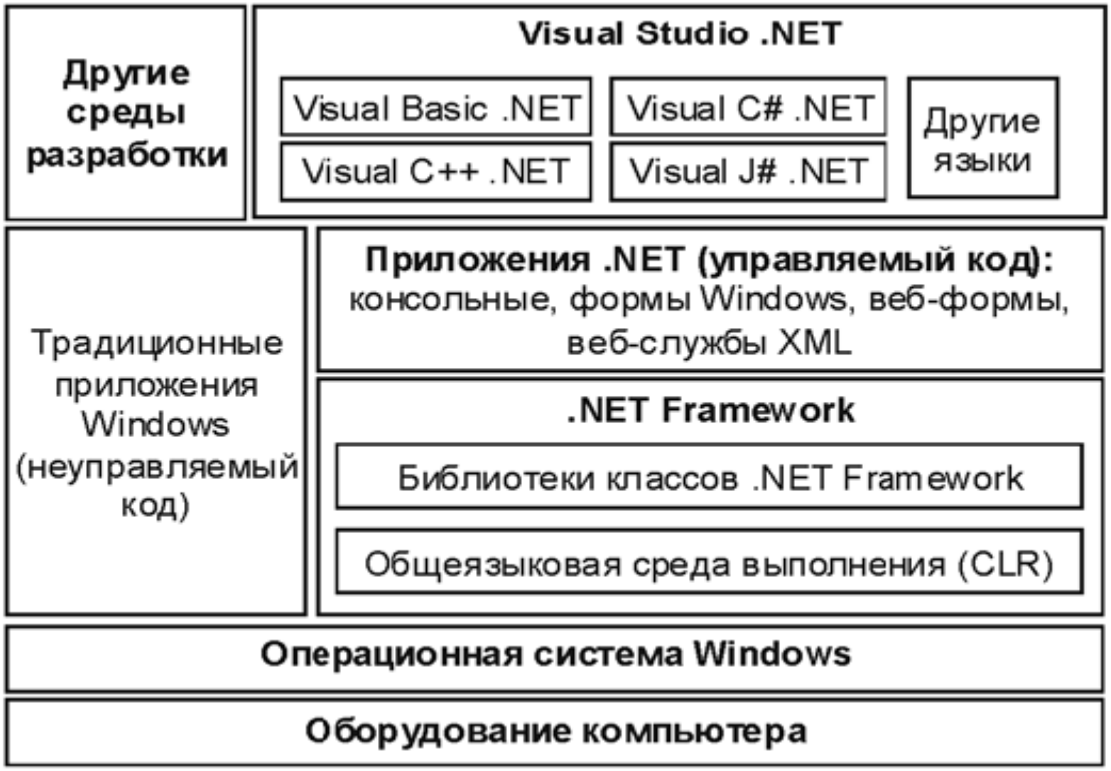


Рис1. Visual Studio .NET. и .NET Framework.

Основным элементом .NET Framework является **общеязыковая среда выполнения приложений** (Common Language Runtime, или **CLR**), обеспечивающая выполнение следующих процессов:

● компиляцию кода по мере вызова тех или иных компонентов программы;

● распределение памяти для кэширования откомпилированного кода и размещения данных;
● управление потоками вычислений и удаленное взаимодействие программ;
● строгую проверку типов данных и другие виды проверки точности кода, что гарантирует безопасность и надежность выполнения программ.

Таким образом, основным принципом работы CLR является управление программным кодом. Именно поэтому код, который выполняется в .NET Framework, называют ***управляемым*** кодом (managed code), а код, который выполняется на компьютере, минуя CLR, - ***неуправляемым*** (unmanaged). Более того, данные, с которыми работает управляемый код, также находятся под полным контролем CLR и поэтому называются ***управляемыми данными*** (managed data).

Другой основной компонент .NET Framework – общая для всех языков программирования **библиотека классов –** FCL (Framework Class Library). Ее наличие позволяет разработчикам использовать единую систему типов данных и вызываемых функций (точнее, программных объектов, их свойств и методов). Соответственно, большая часть функциональности программы, которая ранее реализовывалась за счет функций и процедур конкретного языка программирования, теперь обеспечивается использованием библиотеки классов. Например, чтобы вычислить квадратный корень в предыдущих версиях Visual Basic, программисту нужно было воспользоваться конструкцией вида:

X=SQR (y)

В Visual Basic .NET аналогичный оператор будет выглядеть иначе:

X=System.Math.Sqrt (y)

В этой конструкции уже будет задействован один из стандартных классов библиотеки .NET Framework, относящийся к так называемому ***пространству имен*** (namespace) System.Math. В .NET Framework пространством имен называется некоторая область, в которой определены названия, свойства и методы используемых классов (как системных, так и создаваемых разработчиком программы).

Существенным преимуществом данного подхода является возможность использовать **одни и те же** классы в программах**,** написанных на **разных** языках программирования: и на тех, которые разработаны корпорацией Майкрософт, и на языках **сторонних производителей**. Более того, разработчики могут создавать свои собственные библиотеки классов и использовать их в дальнейшей работе. А межъязыковое взаимодействие и общая среда разработки позволяют, например, в Visual Basic .NET-приложениях использовать компоненты, написанные на других языках .NET

Как видно из схемы, Visual Studio .NET – это **единая** среда разработки приложений, как традиционных, так и работающих в среде выполнения .NET Framework. И если первые весьма жестко привязаны к особенностям интерфей-

са прикладного программирования в операционной системе Windows (Win32 API), то для вторых CLR «экранирует» эти особенности. Такой подход делает написанные для общеязыковой среды программы легко переносимыми на компьютеры, работающие не под управлением Windows. Примерами такого переноса являются **проект Portable .NET** для Linux и других Unix-подобных систем или **проект Rotor** для Free BSD и Mac OS X.

Таким образом, совместное использование Visual Studio .NET и .NET Framework предоставляет в распоряжение разработчиков один из самых мощных на сегодняшний день инструментов для создания программных проектов.

Система объектно-ориентированного программирования Visual Basic .NET является составной частью единой среды разработки приложений Visual Studio .NET и в полной мере реализует объектный стиль разработки программных проектов. Для программиста, владеющего основами этого стиля, не столь важно, на каком **конкретном языке** программирования или в какой среде ему необходимо разработать тот или иной программный проект – на любом языке он будет создавать программный продукт требуемого качества. Тем не менее, у каждого программиста есть свои предпочтения, свой любимый язык и среда разработки.

Вашей задачей теперь, в рамках данной лабораторной работы, является знакомство с интерфейсом IDE и приобретение практических навыков работы в Visual Studio .NET, **ориентируясь**, в определенном смысле, на методику, изложенную выше для среды IDE VBA.

Для самостоятельного изучения можно использовать любой, доступный для Вас, выпуск Visual Studio .NET из линейки программных продуктов корпорации Microsoft: Visual Studio 2003 – Visual Studio 2013.

Некоторые компоненты Visual Studio NET в варианте Express Edition распространяются Майкрософт бесплатно; их дистрибутивы доступны для «скачивания» с сайтов корпорации:

http://www.microsoft.com/rus/express/
http://www.microsoft.com/ru/ru/default.aspx
http://msdn.microsoft.com/ru-ru/vstudio/default.aspx

Следует отметить, что в различных выпусках Visual Studio .NET различаются как среда **разработки**, так и среда **выполнения** приложений (версия 2003 базируется на .NET Framework 1.1, версия 2005 – на .NET Framework 2.0 и 3.0, версия 2008 – на .NET Framework 3.5, версия 2010 – на .NET Framework 4.0 и т.д.). Различия необходимо учитывать и в процессе разработки программ с помощью данных систем, и при использовании программ в дальнейшем.

Кроме того, в Visual Studio .NET включена электронная справочная система (так называемая библиотека **MSDN**). Русскоязычная справка размещена в Интернете по адресу: http://msdn.microsoft.com/ru-ru/library/default.aspx

Результаты выполнения работы представить в формате файла-отчета по лабораторной работе – **lr1Отчет.doc**, содержащего экранные копии диалоговых

окон с данными соответствующего этапа выполнения работы, с необходимыми комментариями.
Форма отчетности – стандартная**: именованная** (фамилией студента) **папка**, содержащая текстовый **файл с отчетом**.
(Например: Иванов \ lr1Отчет.doc).

Контрольные вопросы

1. Чем определяется выбор офисного приложения **в качестве основного** при создании проекта пользовательского приложения в инструментальных средах VBA?
2. Назовите инструментальные компоненты, которые можно вывести в окне редактора VBE.
3. Что содержит окно проекта при его открытии?
4. Перечислите несколько альтернативных способов добавления модуля в проект.
5. Каким способом можно быстро найти процедуру обработки события для текущего объекта в том случае, когда объем модуля достигает больших размеров?
6. Как можно увеличить стандартный набор элементов на панели элементов?
7. Как создать процедуру обработки события, связанного с элементом управления на форме?
8. Можно ли использовать компонент некоторого VBA-приложения в текущем проекте?

Источники информации

1. Коновалов В.М. Прикладное программное обеспечение: Пособие по выполнению лабораторных работ. – М.: МГТУ ГА, 2010. [с. 5 – 22]
2. Голощапов А.Л. Microsoft Visual Studio 2010. – СПб: БХВ – Петербург, 2011

Лабораторная работа 2. ПРОЕКТИРОВАНИЕ ИНТЕРАКТИВНЫХ ПРИЛОЖЕНИЙ В БАЗИСЕ MS Excel

Продолжительность работы – **4 часа.**
Цель работы: освоить технику применения макросов в MS Excel; научиться пользоваться ими для повышения производительности разрабатываемых приложений.

Повышение производительности приложений с помощью макросов.

Большинству пользователей Excel приходится часто выполнять повторяющиеся задачи – например, вводить серии заголовков в финансовых отчетах или увеличивать ширину нескольких столбцов в книге. Подобные действия отнимают время, поэтому имеет смысл записать их в виде макроса.
Макросом называется именованная последовательность команд, которая в VBA синтаксически оформляется в виде процедуры Sub.

Прежде, чем использовать макрос, следует убедиться, что в основном приложении нет встроенного средства для решения нужной задачи. Например, если часто приходится выделять полужирным начертанием текст заголовков

столбцов и увеличивать в них размер шрифта, то можно записать макрос для автоматического форматирования заголовков. Но на самом деле значительно быстрее будет воспользоваться командой **Формат\Стиль**, которая накладывает на ячейки стиль заголовка и позволяет добиться того же эффекта форматирования. Другими словами, не пользуйтесь макросами, если только их применение не оправдывается сложностью записываемых команд.

Это предостережение вовсе не означает, что не следует пользоваться макросами для автоматизации работы – дело обстоит как раз наоборот. Но перед тем, как приступить к записи макроса, стоит все же ознакомиться с возможностями **основного** приложения в части встроенных средств автоматизации и знать, когда использовать готовое средство, а когда для максимальной эффективности следует приступить к созданию макроса.

Используемый для создания макросов в Excel язык Visual Basic обладает достаточными возможностями для решения многих нетривиальных задач, например, для связи с другими приложениями Windows. Однако самыми полезными макросами нередко оказываются те, которые заменяют всего четыре или пять простых команд.

Хотя макросы можно создать "из ничего" в среде Visual Basic, лучшим способом изучения макросов все же является автоматическая генерация кода VBA (запись макрорекодером) с последующим редактированием процедуры макроса.

Порядок выполнения лабораторной работы

Изучить основные действия, осуществляемые при создании и применении макросов:

● произвести автоматическую запись макроса

Самый простой способ создать процедуру макроса – это записать ее с помощью макрорекодера. Этот метод пригоден для Excel, Word, PowerPoint. В Access макросы создаются с помощью собственного языка макросов, а не создаются автоматически. Более того, макросы в Access не являются процедурами VBA.

Прежде чем записать макрос, требуется продумать свои действия:

1. Какие условия должны выполняться при запуске макроса:
   - Должен ли быть открыт некоторый файл?
   - Должен ли курсор находиться в определенном положении или должна быть активной определенная ячейка?
   - Требуется ли включить определенный режим работы приложения?
2. Какие действия должен выполнять макрос:

   Рекомендуется перед записью потренироваться и выполнить пробный заход. Во многих случаях не играет роли, что используется при записи – мышь, клавиатура или команды меню, но в записи будет меньше ошибок, если заранее продумать последовательность своих действий.
3. Тщательно продумайте, что необходимо проделать при завершении работы макроса:
   - Требуется ли установить курсор в некоторую позицию или сделать активной некоторую ячейку?

• Требуется ли закрыть открытый файл или вернуть к исходному состоянию любые измененные установки?

Хороший макрос характеризуется тем, что по завершении своей работы приложение имеет то же состояние, что и до его запуска, конечно, если назначение макроса не состоит в том, чтобы изменить это состояние.

Для записи макроса следует воспользоваться командой **Сервис\Макрос\Начать запись,** а имя макроса будет задано в окне диалога **Запись макроса**

● выполнить макрос

Чтобы сделать автоматизацию работы более гибкой, Excel предоставляет пользователю три способа выполнения макроса.

• Двойной щелчок на имени макроса в окне диалога **Макрос**.

• Нажатие сочетания клавиш (если оно было присвоено макросу).

• Щелчок кнопки макроса на панели инструментов (если для него была создана кнопка).

● редактировать макрос

По умолчанию Excel сохраняет макросы в модуле с именем *Модуль1*. Его можно вывести на экран, чтобы просмотреть команды языка Visual Basic, из которых состоит макрос, снабдить его комментариями относительно использования или отредактировать для изменения поведения или повышения эффективности. Для редактирования макроса следует выполнить команду **Сервис\Макрос\Макросы**. и продолжить работу в окне диалога **Макрос**.

● использовать личную книгу макросов

Личная книга макросов хранится в папке XLStart и автоматически загружается при запуске Excel. Обычно она используется для хранения макросов, но при желании в нее можно заносить и обычные листы. Поскольку для записи или выполнения макросов видеть ее необязательно, она остается скрытой, если только не открыть ее командой **Окно\Показать**. Личная книга макросов не существует до момента сохранения в ней первого макроса.

● назначить макросу кнопку на панели инструментов

Макросы становятся более доступными, если для их запуска имеется кнопка на панели инструментов. Для того чтобы воспользоваться этой услугой, необходимо предварительно записать макрос и присвоить ему имя. Если требуется, чтобы кнопка постоянно присутствовала на панели инструментов, макрос должен быть записан в личную книгу. Для добавления кнопки макроса на панель инструментов следует использовать окно диалога **Настройка** после отработки команды **Вид\Панели инструментов\Настройка**.

Сделаем одно важное замечание. Во многих случаях записанные макросы выполняют не совсем те действия, которые требуются. Основная причина заключается в том, что информация в командах макроса в точности соответствует введенной при записи. Например, в коде записанного макроса указано именно то имя файла, который был открыт при записи, либо задано значение параметра, установленное на момент записи в диалоговом окне. По этой причине часто необходимо отредактировать записанные макросы. Тем не менее, запись макроса позволяет существенно сократить время на создание приложений и является эффективным способом освоить VBA.

После ознакомления с вопросами, относящимися к технологии создания макросов в MS Excel, **Вашей задачей** теперь является:

1. Подготовка объекта автоматизации - алгоритма вычислений на таблице (одной или нескольких) MS Excel, в соответствии с **вариантом заданий**. Метод

вычислений (соответственно, реализующий этот метод вычислительный алгоритм) с применением таблиц MS Excel выбирается самостоятельно.

2. Запись макроса, автоматизирующего работу пользователя с таблицами.

3. Редактирование процедуры макроса с целью внесения в нее изменений, которые не могут быть воспроизведены в режиме автоматической записи.

4. Разработка **пользовательского интерфейса** интерактивного документа Excel.

5. Оформление результатов в формате исполняемого файла – **lr2.xls** и файла-отчета по лабораторной работе – **lr2Отчет.doc**.

Форма отчетности – стандартная**: именованная** (фамилией студента) **папка**, содержащая файл/файлы с исполняемым кодом; текстовый **файл с отчетом**.
(Например: Иванов \ lr2.xls, Иванов \ lr2Отчет.doc).

Варианты заданий для разработки макросов в MS Excel

Вариант 1.
Взять матрицу M x N. Определить сумму положительных элементов каждой строки и поместить на место элементов главной диагонали. Вывести результирующую матрицу рядом с исходной. M = 5, N = 5.

Вариант 2.
Взять матрицу M x N. Определить сумму отрицательных элементов каждой
строки и поместить на место первого элемента соответствующей строки.
Вывести результирующую матрицу рядом с исходной. M=4, N=6.

Вариант 3.
Взять матрицу M x N. Определить сумму четных элементов каждой строки и поместить на место последнего элемента соответствующей строки. Вывести результирующую матрицу под исходной матрицей. M=6, N=8.

Вариант 4.
Взять матрицу M x N. Определить произведение элементов кратных 3 в каждой строке и поместить на место элемента, номер которого в строке совпадает с номером строки. Вывести результирующую матрицу рядом с исходной. M=3, N=6.

Вариант 5.
Взять одномерный массив из L элементов. Отсортировать его по возрастанию. Вывести результирующий массив под исходным. L = 30.

Вариант 6.
Взять одномерный массив из L элементов. Определить количество различных элементов и поместить их в начале массива. Вывести результирующий массив под исходным. L=20.

Вариант 7.
Взять одномерный массив из L элементов. Поместить в начало массива отрицательные элементы, в середину – положительные, а в конец – нулевые. Вывести результирующий массив под исходным. L=25.

Вариант 8.
Взять матрицу M x N. Определить строку с максимальной суммой элементов и выделить ее цветом. M = 8, N = 6.

Вариант 9.

Взять матрицу M x N. Определить сумму элементов каждого столбца. Вывести результирующую матрицу–строку под исходной. M=4. N=6.

Вариант 10.

Взять матрицу M x N. Определить максимальный элемент каждой строки. Выделить их цветом. M=4, N=6.

Вариант 11.

Взять матрицу M x N. Определить минимальный элемент каждого столбца и выделить их цветом. M = 4, N = 5.

Вариант 12.

Взять матрицу M x N. Вычеркнуть строку с заданным номером. Вывести результирующую матрицу рядом с исходной. M =5, N=6.

Вариант 13.

Взять матрицу M x N. Вычеркнуть столбец с заданным номером. Вывести результирующую матрицу рядом с исходной. Вывести номер вычеркнутого столбца. M=6, N=7.

Вариант 14

Взять матрицу M x N. Определить строку с максимальной суммой положительных элементов. Выделить ее цветом. M = 6, N = 4.

Вариант 15

Взять матрицу M x N. Определить суммы элементов над и под главной диагональю. Вывести полученные суммы справа и снизу, соответственно, относительно исходной матрицы. . M = 6, N = 6.

Вариант 16

Взять матрицу M x N. Определить в каждой строке произведение элементов, кратных 4. Поместить их на место соответствующих элементов побочной диагонали. Вывести результирующую матрицу рядом с исходной. M = 5, N = 5.

Вариант 17

Взять матрицу M x N. Расставить по возрастанию элементы заданной строки. Вывести результирующую матрицу рядом с исходной. M=6, N=5.

Вариант 18

Взять матрицу M x N, Преобразовать ее в симметричную по строке M/2. Вывести результат рядом с исходной матрицей. M=6, N=7.

Вариант 19

Взять матрицу M x N. Определить максимальный элемент над главной диагональю. Выделить его цветом. M=6, N=6.

Вариант 20

Взять матрицу M x N. Обнулить элементы с четной суммой индексов. Вывести результирующую матрицу рядом с исходной. M=7, N=4.

Вариант 21

Взять матрицу M x N. Определить элементы, сумма индексов которых кратна трем. Выделить их цветом. M=4, N=7.

Вариант 22

Взять матрицу M x N. Определить сумму элементов над побочной диагональю.
Результат-сумму вывести над исходной матрицей. M=8 N=8.

Вариант 23

Взять матрицу M x N. Вычеркнуть строки с отрицательной суммой элементов. Вывести результирующую матрицу рядом с исходной. M = 7, N = 5.

Вариант 24

Взять матрицу M x N. Вычеркнуть столбцы с четными номерами. Вывести результирующую матрицу рядом с исходной. M = 5, N = 8.

Вариант 25

Взять матрицу M x N. Определить количество ненулевых элементов каждого столбца. Результат-строку вывести под исходной матрицей. M=5, N=7.

Вариант 26

Взять матрицу M x N. Транспонировать. Вывести результирующую матрицу рядом с исходной. M = 7, N = 4.

Контрольные вопросы

1. В каком виде автоматически записанный в Excel макрос сохраняется в VBA-проекте?
2. Назовите известные способы запуска макросов.
3. Где должен храниться макрос, чтобы им можно было пользоваться во всех книгах Excel?
4. Как назначить макросу кнопку на панели инструментов?
5. Как назначить макросу сочетание клавиш, осуществляющих его вызов?

Источники информации

1. Коновалов В.М. Прикладное программное обеспечение: Пособие по выполнению лабораторных работ. – М.: МГТУ ГА, 2010. [с. 22 – 34]
2.. Биллиг В.А. Основы офисного программирования и документы Excel. http://www.intuit.ru/department/office/vbaexcel/
3. Биллиг В.А.. Основы офисного программирования и язык VBA. http://www.intuit.ru/studies/courses/112/112/info
4. Заика А.А. **VBA в MS Office 2007**. http://www.intuit.ru/department/se/vbamsoffice2007/

Лабораторная работа 3. АВТОМАТИЗАЦИЯ ОБРАБОТКИ ДАННЫХ В MS Access

Продолжительность работы – **4 часа**.

Цель работы: изучить особенности работы с макросами в Access; научиться пользоваться ими для повышения производительности работы с данными базы.

Автоматизация приложений в Access

В Microsoft Access многие действия выполняются с помощью макросов. Вместе с тем, для решения задач автоматизации требуется программирование на языке VBA. Выбор между созданием макроса или разработкой программы VBA обычно определяется требуемыми действиями.

В каких случаях следует создавать макрос?

Макрос является удобным средством выполнения простых задач, таких как открытие и закрытие форм, вывод на экран и скрытие панелей инструментов или запуск отчетов. Действия, связывающие различные объекты базы данных, выполняются легко и просто, поскольку пользователь не должен запоминать правила синтаксиса – все аргументы, требуемые каждой макрокомандой, выводятся в нижней половине окна макроса.

В дополнение к упомянутым простым действиям, макросы необходимо использовать для выполнения следующих задач.

• Определение общих назначенных клавиш.

• Выполнение макрокоманды или набора макрокоманд при открытии базы данных. При этом, определенные действия, которые должны производиться при открытии базы данных, например, открытие формы, могут быть заданы в диалоговом окне **Параметры запуска**.

В каких случаях следует создавать программу VBA?

Программы Visual Basic используют вместо макросов в случаях, когда необходимо:

• Упростить управление базой данных. Поскольку макросы являются объектами, существующими отдельно от использующих их форм и отчетов, поддержание базы данных, в которой реакция на события в формах и отчетах определяется многими макросами, становится достаточно затруднительным.

В отличие от этого, процедуры обработки события Visual Basic являются встроенными в описания соответствующих форм и отчетов. При переносе формы или отчета из одной базы данных в другую встроенные процедуры обработки события автоматически переносятся вместе с формой или отчетом.

• Создавать собственные специализированные функции. В Microsoft Access определен ряд встроенных функций, например, функция IPmt, которая рассчитывает проценты по платежам. Пользователь имеет возможность использовать для проведения расчетов встроенные функции без необходимости разрабатывать сложные выражения. Однако язык VBA позволяет пользователям создавать собственные функции как для решения задач, выходящих за рамки возможных для встроенных функций, так и для замены сложных выражений, содержащих встроенные функции. Кроме того, создаваемые пользователем функции используются для выполнения одинаковых операций над разными объектами.

• Скрывать сообщения об ошибках. Стандартные сообщения об ошибках Microsoft Access, выводящиеся при возникновении нештатных ситуаций во время работы пользователя с базой данных, могут оказаться малопонятными для пользователя, в особенности, для не имеющего большого опыта работы с Microsoft Access. Средства VBA позволяют перехватывать ошибку при ее возникновении и либо выводить собственное сообщение об ошибке, либо предпринимать определенные действия, направленные на ее исправление.

• Создавать или обрабатывать объекты. В большинстве случаев удобнее создавать или изменять объекты в режиме конструктора. Однако в некоторых ситуациях приходится работать с описанием объекта в программе. Средства VBA позволяют выполнять обработку любых объектов в базе данных и самой базы данных.

• Выполнять действия на системном уровне. Выполнение в макросе макрокоманды **ЗапускПриложения** (RunApp) позволяет запускать из собственного

приложения другое приложение, работающее в среде Windows, однако, это практически все, что можно сделать вне Microsoft Access из макроса. Средства Visual Basic позволяют проверять существование файлов, использовать механизм программирования объектов или динамического обмена данными (DDE) для связи с другими приложениями, работающими под управлением Windows, например, Microsoft Excel, а также вызывать функции из библиотек динамической компоновки (DLL) Windows.

• Обрабатывать записи по одной. Инструкции VBA позволяют перебирать наборы записей по одной и выполнять определенные действия над отдельной записью. В отличие от этого, макросы позволяют работать только с целым набором записей.

• Передавать аргументы в специальные процедуры VBA. Пользователь имеет возможность задать в окне макроса значения аргументов макрокоманды при создании макроса, однако, невозможно изменить значения этих аргументов при выполнении макроса. В отличие от макросов, в программах Visual Basic допускается передача аргументов в программу при запуске программы или использование в качестве аргументов значений переменных. Поэтому использование программ VBA дает более широкие возможности работы с данными.

Создание макроса

Макросом в Access называют набор из одной или нескольких макрокоманд, выполняющих определенные операции, такие как открытие форм или печать отчетов. Макросы особенно полезны при создании небольших персональных приложений или создании прототипов больших приложений. Но даже если Вы считаете, что готовы сразу перейти к VBA, лучше сначала изучить макрокоманды. Почти все из них придется использовать в VBA. Поэтому изучение макросов является прекрасным введением в программирование в Access в целом.

Порядок выполнения лабораторной работы

Изучить особенности (в отличие от Excel) действий, осуществляемых при создании и применении макросов в Access:

● подготовить базу данных

В качестве полигона для изучения можно использовать учебную базу данных "Студенты и занятия", которую можно создать в автоматическом режиме с помощью мастера. Для этого воспользуйтесь командой **Файл\Создать базу данных…**.

● подготовить форму

Создать форму можно несколькими способами, но в данном случае лучше всего начать с кнопки **Создать** на вкладке **Формы** окна базы данных.

● записать макрос

Для записи макроса в окне базы данных выбрать вкладку **Макросы.** Нажать кнопку **Создать**. Откроется окно нового макроса. Верхняя часть окна используется для определения нового макроса, а нижняя – для ввода значений аргументов макрокоманд, включенных в макрос.

● запустить и отладить макрос

При запуске макроса выполнение макрокоманд начинается с первой строки макроса и продолжается до конца макроса, или, если макрос входит в группу макросов, до начала следующего макроса.

Выполнение макроса может начинаться по команде пользователя, при вызове из другого макроса или процедуры обработки события, а также в ответ на событие в форме, отчете или элементе управления. Допускается также создание специальной команды меню или кнопки на панели инструментов, запускающей макрос, определение сочетания клавиш, одновременное нажатие которых запускает макрос, а также автоматический запуск макроса при открытии базы данных.

● запуск макроса пользователем

Некоторые макросы могут быть запущены непосредственно из окна базы данных или окна макроса (прямой запуск), поскольку они не зависят от элементов управления открытой формы или отчета. Изучите варианты запуска макроса пользователем.

● запуск макроса. входящего в группу макросов.

В большинстве форм, создаваемых для приложения, требуется применение значительного числа макрокоманд. Одни используются для редактирования полей, другие открывают отчеты, третьи реагируют на нажатие командных кнопок. Можно создать макросы для каждой отдельной операции, но в этом случае в приложении накопятся сотни различных макросов.

Существует эффективный способ упростить ситуацию. Он связан с группированием макросов. Группирование макросов можно произвести для каждой формы или отчета. Другой подход состоит в группировании макросов по типу операций.

● назначение макросу кнопки в форме

● создание специальной команды меню, запускающей макрос

В некоторых случаях требуется использовать одни и те же средства в разных частях приложения. Например, желательно выполнять какую–либо команду независимо от того, с чем Вы работаете в данный момент – с формой, отчетом или чем-нибудь еще. Возможность подобных действий дает процедура вставки в существующее меню требуемой команды или добавление в строку меню нового меню, содержащего требуемую команду.

● создание кнопки панели инструментов, запускающей макрос

Добавление кнопки на панель инструментов является еще одним удобным средством выполнения некоторой команды, находясь в различных местах приложения.

● назначение сочетания клавиш для запуска макроса

В Access допускается связывание макрокоманды или набора макрокоманд с конкретной клавишей или сочетанием клавиш с помощью специальной группы макросов **AutoKeys**. После этого при нажатии клавиши или сочетания клавиш Access будет выполнять данную макрокоманду.

● запуск макроса при открытии базы данных

Специальный макрос с именем **Autoexec** позволяет автоматически выполнить макрокоманду или набор макрокоманд при открытии базы данных. В процессе открытия базы данных Access проводит поиск макроса с этим именем и, если такой макрос существует, автоматически запускает его.

● преобразовать созданные макросы в модули VBA

При необходимости, создаваемые макросы можно автоматически преобразовать в модули VBA, которые выполняют эквивалентные действия с помощью программ Visual Basic. Допускается преобразование макросов, определенных в форме или отчете, или преобразование общих макросов, не связанных с конкретными формами и отчетами.

После ознакомления с особенностями, относящимися к технологии создания макросов в Access, **Вашей задачей** теперь является:

1. Воспроизведение описанных этапов работы применительно к самостоятельно разработанному объекту автоматизации – базе данных, в соответствии **с вариантом** заданий.
2. Оформление результатов в формате исполняемого файла – **lr3.mdb** и файла-отчета по лабораторной работе – **lr3Отчет.doc**.
Форма отчетности – стандартная**: именованная** (фамилией студента) **папка**, содержащая файл/файлы с исполняемым кодом; текстовый **файл с отчетом**.
(Например: Иванов \ lr3.mdb, Иванов \ lr3Отчет.doc).

Варианты заданий для разработки макросов в MS Access.

Вариант 1.
А. Создать в MS Access проект БД, содержащей сведения о месячной зарплате рабочих завода. Каждая запись содержит поля: фамилия рабочего; наименование цеха; размер зарплаты за месяц. Количество записей ≥ 10.
Б. Разработать для пользователя интерактивную форму взаимодействия с БД, позволяющую вывести общую сумму выплат за месяц по цеху X, а также среднемесячный заработок рабочего этого цеха. Напечатать для бухгалтерии ведомость для начисления зарплаты рабочим этого цеха.
Вариант 2.
А. Создать в MS Access проект БД, содержащий сведения о количестве изделий, собранных сборщиками цеха за неделю. Каждая запись содержит поля: фамилия сборщика; количество изделий, собранных им ежедневно в течении шестидневной недели. Количество записей ≥ 10.
Б. Разработать для пользователя интерактивную форму взаимодействия с БД, для визуализации следующей информации: фамилии сборщика и общего количества деталей, собранных им за неделю; фамилии сборщика собравшего наибольшее число изделий, и день, когда он достиг наивысшей производительности труда.
Вариант 3.
А. Создать в MS Access проект БД, содержащей сведения о количестве изделий категорий A, B, C, собранных рабочими за месяц. Структура записи: фамилия сборщика; наименование цеха; количество изделий по категориям, собранных рабочим за месяц. Количество записей ≥ 10.
Б. Разработать для пользователя интерактивную форму взаимодействия с БД, позволяющую вывести (считая заданными значения расценок S_a, S_b, S_c за выполненную работу по сборке единицы изделия категорий A, B, C соответственно), следующую информацию:
- общее количество изделий категорий A, B, C, собранных рабочим цеха X;
- ведомость заработной платы рабочих цеха X;
- средний размер заработной платы работников этого цеха.

Вариант 4.
А. Создать в MS Access файл БД, содержащий сведения о телефонах абонентов. Каждая запись имеет поля: фамилия абонента; год установки телефона; номер телефона.
Количество записей ≥ 10.
Б. Разработать для пользователя интерактивную форму взаимодействия с БД, позволяющую вывести информацию следующего вида:
- по вводимой фамилии абонента выдается номер абонента;
- определяется количество установленных телефонов с XXXX г.

Вариант 5.

А. Создать в MS Access файл БД, содержащий сведения об ассортименте игрушек в магазине. Структура записи: название игрушки; цена; количество; возрастные границы.
Количество записей ≥ 10.

Б. Разработать для пользователя интерактивную форму взаимодействия с БД, позволяющую вывести следующие сведения:
- названия игрушек, которые подходят детям от 1 до 3-х лет;
- стоимость самой дорогой игрушки;
- названия игрушек, которые по стоимости не выше X руб. и подходят детям от A до B лет.

Значения X, A, B вводить на форме.

Вариант 6.

А. Создать в MS Access файл БД, содержащий сведения о сдаче студентами 4 курса кафедры “ПМ” сессии. Структура записи: индекс группы; фамилия студента; оценки по пяти предметам; признак участия в общественной работе: "1" – активное участие, "0" – неучастие.
Количество записей ≥ 25.

Б. Разработать для пользователя интерактивную форму взаимодействия с БД, позволяющую вывести информацию о зачислении студентов группы X на стипендию. Студент, получивший все оценки "5" и активно участвующий в общественной работе, зачисляется на повышенную стипендию (доплата 50%). Не активно участвующий – доплата 25%. Студенты, получившие "4","5", зачисляются на обычную стипендию. Студенты, получившие одну "3", но активно занимающиеся общественной работой, также зачисляются на стипендию, в противном случае зачисление не производится. Индекс группы вводится на форме.

Вариант 7.

А. Создать в MS Access файл БД, содержащий сведения о сдаче студентами сессии. Структура записи: индекс группы; фамилия студента; оценки по пяти экзаменам и пяти зачетам ("з" означает зачет, "нз" – незачет). Количество записей ≥ 25.

Б. Разработать для пользователя интерактивную форму взаимодействия с БД, позволяющую вывести следующие сведения:
- фамилии неуспевающих студентов с указанием индексов групп и количества задолженностей;
- средний балл, полученный каждым студентом группы X, и всей группы в целом.

Вариант 8.

А. Создать в MS Access файл БД, содержащий сведения о личной коллекции книголюба. Структура записи: шифр книги; автор; название; год издания; местоположение (номер стеллажа, шкафа и т.п.). Количество записей ≥ 20.

Б. Разработать для пользователя интерактивную форму взаимодействия с БД, позволяющую вывести следующую информацию:
- местоположение книги автора X названия Y. Значения X и Y вводятся на форме;
- список книг автора Z ,находящихся в коллекции;
- число книг издания XXXX года, имеющихся в библиотеке.

Вариант 9.

А. Создать в MS Access файл БД, содержащий сведения о наличии билетов и рейсах Аэрофлота. Структура записи: номер рейса; пункт назначения; время вылета; время прибытия; количество свободных мест в салоне. Количество записей ≥ 20.

Б. Разработать для пользователя интерактивную форму взаимодействия с БД, позволяющую вывести информацию следующего вида:
- время отправления самолетов в город X.
- наличие свободных мест на рейс в город X с временем отправления Y.

Значения X, Y по запросу вводятся на форме.

Вариант 10.

А. Создать в MS Access файл БД, содержащий сведения об ассортименте обуви в магазине фирмы. Структура записи: артикул; наименование; количество; стоимость одной пары. Количество записей ≥ 20. Артикул начинается с буквы Д для дамской обуви, М - для мужской, П - для детской.

Б. Разработать для пользователя интерактивную форму взаимодействия с БД, позволяющую вывести следующую информацию:

- о наличии и стоимости обуви артикула X;
- ассортиментный список дамской обуви с указанием наименования и имеющегося в наличии числа пар каждой модели.

Вариант 11.

А. Создать в MS Access файл БД, содержащий сведения о 10 нападающих хоккейных команд «Спартак» и «Динамо». Соответствующие таблицы должны содержать: имена нападающих; число заброшенных ими шайб; сделанных голевых передач; заработанное штрафное время.

Б. Разработать для пользователя интерактивную форму взаимодействия с БД, позволяющую вывести имя, команду, сумму очков (голы + передачи) для шести лучших игроков обеих команд.

Вариант 12.

А. Создать в MS Access файл БД, содержащий сведения о том, какие из пяти предлагаемых дисциплин по выбору желает слушать студент. Структура записи: фамилия студента; индекс группы; 5 дисциплин; средний бал успеваемости. Выбираемая дисциплина отмечается символом "1", иначе – пробел. Количество записей ≥ 25.

Б. Разработать для пользователя интерактивную форму взаимодействия с БД, позволяющую вывести список студентов, желающих прослушать дисциплину X. Если число желающих студентов превысит 8 человек, то отобрать студентов, имеющих более высокий средний бал успеваемости.

Вариант 13.

А. Создать в MS Access файл БД, содержащий сведения об отправлении поездов дальнего следования с Ярославского вокзала. Структура записи: номер поезда; станция назначения; время отправления; время в пути; наличие билетов. Количество записей ≥ 20.

Б. Разработать для пользователя интерактивную форму взаимодействия с БД, позволяющую вывести следующую справочную информацию:

- время отправления поездов в город X во временном интервале от A до B часов;
- наличие билетов на поезд с номером XXX.

Вариант 14.

А. Создать в MS Access файл БД, содержащий сведения о сотрудниках института. Структура записи: фамилия работающего; название отдела; год рождения; стаж работы; должность; оклад. Количество записей ≥10.

В. Разработать для пользователя интерактивную форму взаимодействия с БД, позволяющую вывести следующую информацию:

- список сотрудников пенсионного возраста на сегодняшний день с указанием стажа работы;
- средний стаж сотрудников, работающих в отделе X.

Вариант 15.

А. Создать в MS Access файл БД, содержащий сведения о пациентах поликлиники. Структура записи: фамилия пациента; пол; возраст; место проживания (город); диагноз. Количество записей ≥10.

Б. Разработать для пользователя интерактивную форму взаимодействия с БД, позволяющую вывести следующую справочную информацию:

- количество иногородних пациентов, прибывших в клинику;

- список пациентов, старше X лет с диагнозом Y. Значения X, Y по запросу вводятся на форме.

Вариант 16.

А. Создать в MS Access базу данных для работы виртуального магазина по продаже продуктов питания.

Б. Разработать объектно-ориентированный интерфейс для взаимодействия с базой данных для покупателя и сотрудника магазина.

Вариант 17.

А. Создать в MS Access базу данных для работы виртуального магазина по продаже книг и печатной продукции.

Б. Разработать объектно-ориентированный интерфейс для взаимодействия с базой данных для покупателя и сотрудника магазина.

Вариант 18.

А. Создать в MS Access базу данных для работы виртуального магазина по продаже аудио, видео, CD–продукции.

Б. Разработать объектно-ориентированный интерфейс для взаимодействия с базой данных для покупателя и сотрудника магазина.

Вариант 19.

А. Создать в MS Access базу данных для работы виртуального магазина по продаже туристических путевок.

Б. Разработать объектно-ориентированный интерфейс для взаимодействия с базой данных для покупателя и сотрудника магазина.

Вариант 20.

А. Создать в MS Access базу данных для работы виртуального магазина по продаже автомобилей.

Б. Разработать объектно-ориентированный интерфейс для взаимодействия с базой данных для покупателя и сотрудника магазина.

Вариант 21.

А. Создать в MS Access базу данных для работы виртуального магазина по продаже компьютерной техники.

Б. Разработать объектно-ориентированный интерфейс для взаимодействия с базой данных для покупателя и сотрудника магазина.

Примечание. В вариантах №16-21, при разработке интерфейсных форм, необходимо учесть, чтобы покупатель имел возможность выбрать товар по каталогу, сформировать заказ и получить счет без посещения офиса или торгового зала. Сотрудникам магазина - получить информацию о сделанных заказах.

Вариант 22.

А. Создать в MS Access базу данных для автоматизированной системы управления работой магазина.

Б. Разработать объектно-ориентированный интерфейс для взаимодействия с базой данных для должностного лица.

Вариант 23.

А. Создать в MS Access базу данных для автоматизированной системы управления работой гостиницы.

Б. Разработать объектно-ориентированный интерфейс для взаимодействия с базой данных для должностного лица.

Вариант 24.

А. Создать в MS Access базу данных для автоматизированной системы управления работой школы.
Б. Разработать объектно-ориентированный интерфейс для взаимодействия с базой данных для должностного лица.

Примечание. В вариантах №22-24, при разработке интерфейсных форм, необходимо обеспечить соответствующему должностному лицу возможность формировать структуру и штатное расписание организации, принимать сотрудников на работу, увольнять сотрудников, распределять денежные вознаграждения за отдельные виды работ, составлять график работы, объявлять взыскания и поощрения.

Вариант 25.
А. Создать в MS Access базу данных для функционирования информационно-справочной системы морского порта.
Б. Разработать объектно-ориентированный интерфейс для взаимодействия с базой данных для пользователя и сотрудника информационно-справочной службы.
Вариант 26.
А. Создать в MS Access базу данных для функционирования информационно-справочной системы автовокзала.
Б. Разработать объектно-ориентированный интерфейс для взаимодействия с базой данных для пользователя и сотрудника информационно-справочной службы.
Вариант 27.
А. Создать в MS Access базу данных для функционирования информационно-справочной системы библиотеки.
Б. Разработать объектно-ориентированный интерфейс для взаимодействия с базой данных для пользователя и сотрудника информационно-справочной службы.
Вариант 28.
А. Создать в MS Access базу данных для функционирования информационно-справочной системы гостиницы.
Б. Разработать объектно-ориентированный интерфейс для взаимодействия с базой данных для пользователя и сотрудника информационно-справочной службы.
Вариант 29.
А. Создать в MS Access базу данных для функционирования информационно-справочной системы железнодорожного вокзала.
Б. Разработать объектно-ориентированный интерфейс для взаимодействия с базой данных для пользователя и сотрудника информационно-справочной службы.
Вариант 30.
А. Создать в MS Access базу данных для функционирования информационно-справочной системы аэропорта.
Б. Разработать объектно-ориентированный интерфейс для взаимодействия с базой данных для пользователя и сотрудника информационно-справочной службы.

Примечание. В вариантах №25-30, при разработке интерфейсных форм, необходимо предоставить пользователю возможность многокритериального поиска необходимой информации и формирования заказа, а сотрудникам - возможность наполнения системы новыми данными и управления заказами.

Контрольные вопросы

1. В чем отличие и сходство макросов в Access от макросов в Excel?
2. Можно ли в Access создать макрос в автоматическом режиме, подобно тому, как это делается в Excel?
3. Каким образом можно изменить значения передаваемых макросу аргументов?
4. Объясните, как происходит автоматический запуск макроса при открытии базы данных?

Источники информации

1. Коновалов В.М. Прикладное программное обеспечение: Пособие по выполнению лабораторных работ. – М.: МГТУ ГА, 2010. [с. 42 – 67]
2. Гандерлой М., Харкинз С. Автоматизация Microsoft Access с помощью VBA. – М.: Вильямс, 2006.

Лабораторная работа 4. ПРОЕКТИРОВАНИЕ ИНТЕРАКТИВНЫХ ПРИЛОЖЕНИЙ С ПРЯМЫМ ДОСТУПОМ К ДАННЫМ В БАЗИСЕ MS Access

Продолжительность работы – **4 часа**.

Цель работы: освоить приемы работы с объектами прямого доступа к данным в Access; научиться пользоваться ими для повышения производительности работы с данными базы.

Этапы работы:
1. Работа с объектом Recordset типа таблицы
2. Работа с объектом Recordset типа динамического набора записей
3. Работа с объектом Recordset типа статического набора записей
4. Работа с объектом Recordset статического набора записей с последовательным доступом
5. Оценка производительности работы с объектами прямого доступа к данным

Порядок выполнения лабораторной работы

● Разработать и отладить **процедуры создания** каждого из **четырех** типов объектов Recordset, в зависимости от типа открываемого сеанса (рабочей области). Источник внешних данных выбирается самостоятельно.

● В соответствии с целью создания результирующего набора записей, подготовить **процедуры работы** (модификация - добавление, удаление, изменение; поиск; фильтрация; сортировка; транзакции и т.п.) **с записями** в наборах.
Количественный состав процедур **зависит** от типа объекта Recordset и определяется его функциональностью. Результаты выполнения операций – **визуализировать** с помощью **графического интерфейса** (пользовательские интерактивные формы).

● Получить **временные оценки** выполнения **однотипных** операций над **соответствующими** наборами записей для анализа производительности в **разных** рабочих областях. Построить

диаграммы зависимости времени выполнения операций от типа набора записей для каждой рабочей области. Выполнить **анализ** и сделать **выводы** о предпочтительности использования типов объектов прямого доступа в каждой конкретной конфигурации доступа к источнику внешних данных.

● В качестве источника данных можно использовать имеющуюся разработку базы данных, выполненную в Лабораторной работе 3. Основные таблицы в базе данных, для адекватности временных оценок, должны содержать **не менее** 50 записей.

● Форма отчетности – стандартная**: именованная** (фамилией студента) **папка**, содержащая файл/файлы с исполняемым кодом; текстовый **файл с отчетом**.
(Например: Иванов \ lr4.mdb, Иванов \ lr4Отчет.doc).

Контрольные вопросы

1. Приведите описание существующих способов решения задачи обеспечения доступа к данным. Опишите преимущества и недостатки каждого из них.
2. Перечислите и коротко охарактеризуйте основные (наиболее популярные) известные технологии доступа к данным.
3. Опишите архитектуру концепции доступа к данным фирмы Microsoft. Опишите каждый уровень данной архитектуры, а также роль, которую он играет в общем процессе обеспечения доступа к данным.
4. Опишите объектную модель DAO и основные приемы обеспечения доступа к БД с помощью СУБД Access.
5. Опишите технологию ODBC. Приведите ее особенности, преимущества, недостатки. Опишите архитектуру ODBC.
6. С какой целью используется утилита **ODBC Data Sources Administrator?**
7. Объясните типы: файловое DSN, машинное DSN, пользовательское DSN, системное DSN.

Источники информации

1. Коновалов В.М. Прикладное программное обеспечение: Пособие по выполнению лабораторных работ. Ч. II. – М.: МГТУ ГА, 2011
2. Эволюция технологий доступа к данным
http://www.osp.ru/win2000/2003/04/176027/_p2.html
3. Универсальный доступ к данным: Microsoft UDA
http://www.compress.ru/Article.aspx?id=9396
4 Обзор методов доступа к данным
http://alibek09.narod.ru/vb/articles/dba/index.html

Лабораторная работа 5. ПРОЕКТИРОВАНИЕ ИНТЕРАКТИВНЫХ ПРИЛОЖЕНИЙ С ИСПОЛЬЗОВАНИЕМ ТЕХНОЛОГИЙ ADO, ADO**.**NET

Продолжительность работы – **4 часа**.

Цель работы: изучить приемы работы с объектами прямого доступа к источникам внешних данных, научиться пользоваться ими для повышения производительности работы с удаленными источниками данных.

Этапы работы:

1. Работа с объектом Recordset, созданным с типом курсора keyset cursor.
2. Работа с объектом Recordset, созданным с типом курсора static cursor.
3. Работа с объектом Recordset, созданным с типом курсора forward-only cursor.
4. Работа с объектом Recordset, созданным с типом курсора dynamic cursor.
5. Оценка производительности работы с объектами ActiveX Data Objects (**ADO**).
6. Работа с объектами ActiveX Data Objects. NET (**ADO.NET**).

Порядок выполнения лабораторной работы

ActiveX Data Objects (ADO):

● Разработать и отладить **процедуры создания** объектов Recordset для каждого из **четырех** типов курсоров.

● В соответствии с целевым назначением результирующего набора записей, подготовить **процедуры работы** (модификация - добавление, удаление, изменение; поиск; фильтрация; сортировка; транзакции и т.п.) **с записями** в наборах.
Количественный состав процедур **зависит** от функциональности используемого типа курсора, определяемого, к тому же, поддержкой провайдера. Результаты выполнения операций – **визуализировать** с помощью **графического интерфейса** (пользовательские интерактивные формы).

● Получить **временные оценки** выполнения **однотипных** операций над **соответствующими** наборами записей для анализа производительности для разных провайдеров - **Microsoft OLE DB Provider for ODBC, Microsoft.Jet.OLEDB.4.0**, **Microsoft OLE DB Provider for SQL Server**. Построить **диаграммы** зависимости времени выполнения используемых процедур работы от типа курсора для каждого провайдера. Выполнить **анализ** и сделать **выводы** о предпочтительности использования провайдеров в каждой конкретной конфигурации доступа к источнику внешних данных.

● В качестве источника данных можно использовать имеющуюся разработку базы данных, выполненную в Лабораторной работе 3.. Основные таблицы в базе данных, для адекватности результатов выполнения предыдущего задания, должны содержать не менее 50 записей.

● Форма отчетности – стандартная**: именованная** (фамилией студента) **папка**, содержащая файл/файлы с исполняемым кодом; текстовый **файл с отчетом**.
(Например: Иванов \ lr5.mdb, Иванов \ lr5Отчет.doc).

ActiveX Data Objects .NET (ADO.NET):

С появлением и активным развитием платформы Framework дальнейшее развитие получила и технология ADO, которая стала более универсальной и ориентированной на создание распределенных приложений. Универсальность

ее достигается во многом благодаря использованию языка XML, являющегося сегодня стандартом обмена данными между различными системами и средами.

Рассмотрим кратко архитектуру и основные принципы взаимодействия служб ADO.NET. Основные компоненты ADO.NET и их взаимодействие в процессе обработки данных (по MSDN) представлены на рис.2.

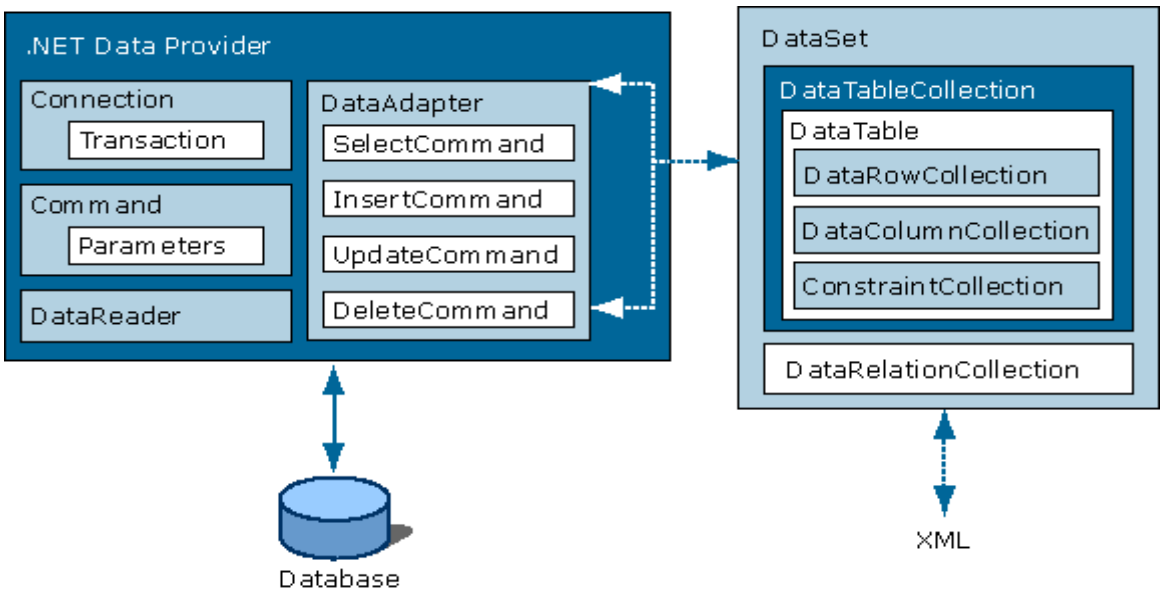


Рис.2. Основные компоненты архитектуры ADO.NET

Как видно из рисунка, доступ к данным основан на использовании двух компонентов: **набора данных (**DataSet**)**, в котором данные хранятся на локальном компьютере и **провайдера данных** (Data Provider), выполняющего функции посредника при взаимодействии программы с БД.

Объект DataSet – это представление в памяти компьютера данных, изолированных от источника данных. Этот объект можно также рассматривать как локальную копию фрагмента БД. В DataSet данные можно загрузить из любого допустимого источника, например из БД SQL Server, Microsoft Access или XML-файла. Объект DataSet хранится в памяти, его содержимым разрешено манипулировать и обновлять независимо от БД, играющей роль источника данных. При необходимости, объект DataSet может служить шаблоном для обновления серверной БД.

Объект DataSet (см. рис.3) содержит набор объектов DataTable (этот набор может быть и пустым, то есть не содержать ни одного объекта DataTable). Каждый объект DataTable представляет в памяти компьютера одну таблицу.

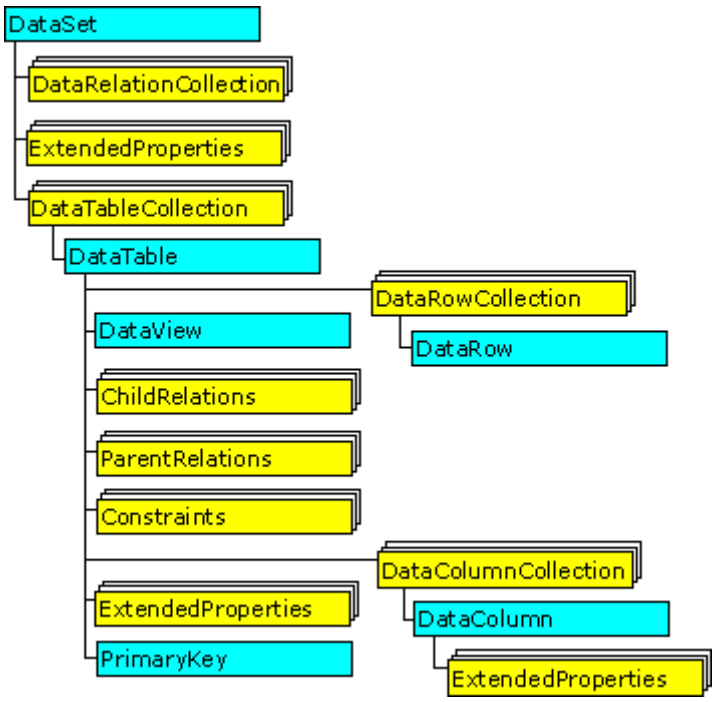


Рис.3. Внутреннее устройство объекта DataSet.

Структура объекта DataTable определяется двумя наборами: DataColumns, куда входят все столбцы таблицы, которую представляет объект DataTable, и набором ограничений (Constrains) таблицы. Вместе эти два набора составляют схему таблицы.

В DataTable также входит набор Data Rows, где, собственно, и хранятся данные объекта DataSet. Конечно, структура объекта DataSet сложнее, чем та, которая приводится здесь и состоит из гораздо большего числа элементов, однако целью данного рассмотрения не является детальное рассмотрение объектной структуры ADO.NET, а лишь знакомство с ее основными элементами.

Связь с БД создается и поддерживается при помощи провайдера данных. В действительности провайдер – это набор взаимосвязанных компонентов, обеспечивающих эффективный высокопроизводительный доступ к данным. В составе .NET Framework поставляются **два типа** провайдеров данных: SQL Server .NET Data Provider (SQL **Managed** Provider), созданный для работы с SQL Server версии 7.0 и выше, и OLE DB .NET Data Provider (ADO **Managed** Provider) – для подключения к OLE DB-источникам данных.

SQL Server .NET Data Provider оптимизирован для работы с Microsoft SQL Server. Он работает по специальному протоколу, называемому Tabular Data Stream (TDS) и не использует ни ADO, ни ODBC, ни какую-либо другую технологию. Ориентированный специально на MS SQL Server, этот протокол позволяет увеличить скорость передачи данных и тем самым повысить общую производительность приложения. За это приходится платить потерей переносимо-

сти на другие сервера баз данных, вследствие использования в TDS специализированных классов.
Находится этот провайдер в пространстве имен System.Data.SqlClient.

OLE DB .NET Data Provider это более общий провайдер, способный работать с почти любым источником данных, поддерживающим OLE DB. (исключения представлены ниже). Поскольку для своей работы он должен использовать COM, то между провайдером и источником данных в этом случае присутствуют две прослойки: обертка, позволяющая managed-приложению использовать COM, и собственно провайдер OLE DB для COM. Естественно, в этом случае производительность уступает предыдущему решению, но сохраняется возможность использования практически любых источников данных.
Находится этот провайдер в пространстве имен System.Data.OleDb.

К числу провайдеров, прошедших тестирование Microsoft на возможность работы в ADO .NET, относятся следующие OLE DB-провайдеры:

Driver	Provider
SQLOLEDB	Microsoft OLE DB Provider for SQL Server
MSDAORA	Microsoft OLE DB Provider for Oracle
Microsoft.Jet.OLEDB.4.0	OLE DB Provider for Microsoft Jet

Не поддерживается OLE DB Provider for ODBC (MSDASQL), а также OLE DB version 2.5 (в частности, Microsoft OLE DB Provider for Exchange и Microsoft OLE DB Provider for Internet Publishing). Для корректной работы провайдера OLE DB следует установить MDAC 2.6 и выше.
Не рекомендуется использование баз данных Microsoft Access для многоуровневых приложений.
Для работы **с источниками данных ODBC** в среде ADO .NET, Microsoft предоставляет соответствующую поддержку – провайдер ODBC .NET Data Provider. Этот **третий** тип поставщика данных .NET (ODBC **Managed** Provider) можно загрузить с сайта http://www.microsoft.com/downloads/

На Рис.4 показаны различные конфигурации подключений, по которым приложение может связываться с базой данных через ADO.NET. При выборе пути сначала определяется, какой поставщик данных .NET будет использоваться. Если база данных – SQL Server 7.0 или более поздняя версия, то подключается поставщик данных SQL Server.NET. Если база данных – SQL Server 6.5 или отличная от SQL Server (например, Oracle), понадобится поставщик данных OLE DB .NET. Заметим, что можно задействовать поставщика данных OLE DB .NET для баз данных SQL 7.0 и выше, но тогда потеряется выигрыш в производительности, который дает прямое подключение к SQL Server через протокол

TDS. Однако в этом неспецифическом способе есть свой плюс — мобильность, т. е. можно менять базы данных без модификации кода.

Далее необходимо определить, какую задачу требуется выполнить. Если надо просто прочитать и отобразить данные из источника данных, - объекта DataReader вполне достаточно. Но если предстоит манипулировать данными (например, редактировать или удалять), нужно использовать объект DataSet. Задействовать этот объект целесообразно только в случае необходимости, потому что он работает медленнее, чем DataReader (DataSet использует DataReader для заполнения таблиц).

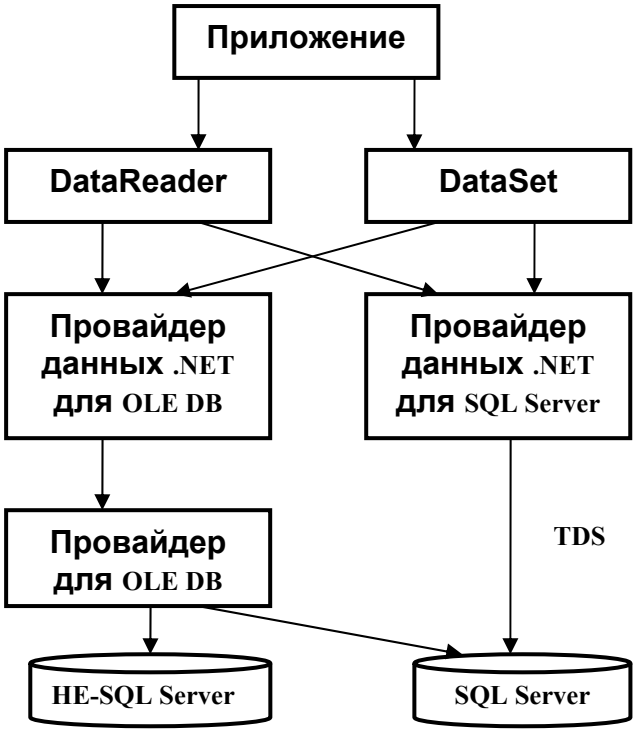


Рис.4. Конфигурации подключений к источникам данных в ADO.NET

Любой провайдер данных состоит из близких версий следующих универсальных классов компонентов:

- Connection – обеспечивает подключение к БД;
- Command – применяется для управления источником данных; позволяет исполнять команды, не возвращающие данные, например, INSERT, UPDATE и DELETE, либо команды, возвращающие объект DataReader (такие, как SELECT);
- DataReader – предоставляет доступный только для однонаправленного чтения набор записей, подключенный к источнику данных;
- DataAdapter – заполняет отсоединенный объект DataSet или DataTable и обновляет его содержимое.

Доступ к данным в ADO.NET осуществляется так: объект Connection устанавливает между приложением и БД соединение, напрямую доступное объек-

там Command и DataAdapter. Объект Command позволяет исполнять команды непосредственно над БД. Если исполненная команда возвращает несколько значений, Command открывает к ним доступ через объект DataReader. Полученные результаты можно обрабатывать напрямую, используя код приложения, либо через объект DataSet, заполнив его при помощи объекта DataAdapter. Для обновления БД также применяют объекты Command или DataAdapter.

Контрольные вопросы

1. Опишите технологию OLE DB. Приведите ее особенности, преимущества, недостатки. Опишите архитектуру OLE DB, а также механизм взаимодействия компонентов.
2. Опишите технологию ADO. В чем заключается особенность ADO? Перечислите ее преимущества и недостатки. Опишите объектную модель ADO и ее отличия от DAO и RDO.
3. Опишите технологию доступа к данным ADO.NET. В чем ее отличия от ADO? Перечислите основные компоненты этой технологии и опишите механизмы взаимодействия между ними.
4. Охарактеризуйте функциональность **Провайдера данных** с позиции технологии OLE DB.
5. При доступе к данным каких источников предпочтительно использовать провайдер **Microsoft Jet 4.0 OLE DB** ?
6. Для каких применений разработан провайдер, специфицируемый как **Microsoft OLE DB Provider for SQL Server**?
7. В чем отличие настольных СУБД от их сетевых многопользовательских версий?
8. В чем отличие файл-серверных систем доступа к данным от клиент-серверных систем?
9. В чем причина большей универсальности технологии ADO.NET по сравнению с ADO?

Источники информации

1. Коновалов В.М. Прикладное программное обеспечение: Пособие по выполнению лабораторных работ. Ч. II. – М.: МГТУ ГА, 2011
2. OLE DB и ADO
http://compress.ru/article.aspx?id=11477&iid=451
3. Универсальный доступ к данным и ADO
http://www.intuit.ru/department/office/vbaexcel/5/
4. ADO.NET: Обзор технологии
http://www.cyberguru.ru/dotnet/ado-net/adonet-overview.html
5 Архитектура ADO.NET
http://msdn.microsoft.com/ru-ru/library/27y4ybxw%28v=VS.90%29.aspx
6. Технологии и средства доступа к реляционным базам данных. ADO .NET
http://club.shelek.ru/viewart.php?id=148