

**Министерство науки и высшего образования
Российской Федерации**

Федеральное государственное бюджетное образовательное учреждение
высшего образования

**«ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ» (ТУСУР)**

Кафедра автоматизации обработки информации (АОИ)

ИНФОРМАТИКА ДЛЯ МЕНЕДЖЕРОВ

Методические указания к лабораторным работам
и организации самостоятельной работы
для студентов направления
«Государственное и муниципальное управление»
(уровень бакалавриата)

2018

Потахова Ирина Владимировна

Информатика для менеджеров: Методические указания к лабораторным работам и организации самостоятельной работы для студентов направления «Государственное и муниципальное управление» (уровень бакалавриата) / И.В. Потахова. – Томск, 2018. – 50 с.

Оглавление

1. Введение	4
2. Методические указания по проведению лабораторных работ.....	5
2.1. Общие положения	5
2.2. Лабораторная работа «Интегрированная среда VBA. Создание диалоговых окон»	5
2.3. Лабораторная работа «Создание макросов Word».....	11
2.4. Лабораторная работа «Создание макросов Excel».....	17
2.5. Лабораторная работа «Основы программирования в VBA».....	22
2.6. Лабораторная работа «Строки VBA».....	28
2.7. Лабораторная работа «Массивы в VBA».....	32
2.8. Лабораторная работа «Создание собственных функций рабочего листа».....	35
2.9. Лабораторная работа «Работа с внешними файлами».....	38
3. Методические указания для организации самостоятельной работы ...	47
3.1. Общие положения	47
3.3. Самостоятельное изучение тем теоретической части курса	48
4. Рекомендуемые источники	50

1. ВВЕДЕНИЕ

Лабораторные работы выполняются в рамках курса «Информатика для менеджеров», предусматривающего изучение языка высокого уровня. В работах рассматриваются вопросы создания простых приложений, выполненных в среде разработки Visual Basic.

Работы рекомендуется выполнять в порядке их следования. В каждой работе предусмотрено обязательное выполнение индивидуального задания. В ходе выполнения лабораторных работ студенты изучают интегрированную среду разработки приложений, приобретают навыки построения интерфейса соответствующих приложений, знакомятся с основными элементами управления, принципами формирования программного кода.

По выполненным лабораторным работам студент отчитывается перед преподавателем. Отчет студента должен быть представлен выполненными программами и отчетом с пояснениями по ходу их выполнения.

2. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ПРОВЕДЕНИЮ ЛАБОРАТОРНЫХ РАБОТ

2.1. Общие положения

Основными целями лабораторных работ являются: закрепление теоретического материала, посредством решения индивидуальных задач; интерпретация полученных результатов.

Основной формой проведения лабораторных работ является решение задач с использованием средств вычислительной техники (Windows, MS Word, MS Excel)

Контроль формирования компетенций осуществляется посредством индивидуальной и (или) групповой защиты отчетов по лабораторным работам и (или) опроса студентов по вопросам, образующим тематическое наполнение каждого из заданий. По итогам или в ходе защиты отчетов (опроса) производится разбор типичных трудностей и ошибок, допущенных студентами в ходе выполнения заданий.

2.2. Лабораторная работа «Интегрированная среда VBA. Создание диалоговых окон»

Цель работы: приобретение первичных навыков создания приложения *Visual Basic for Application (VBA)*.

Форма проведения: выполнение индивидуального задания.

Форма отчетности: выполнение теста, защита отчета.

1. Среда разработки Visual Basic

Рабочая среда VBA называется интегрированной средой разработки или *IDE (Integrated Development Environment)* (рис. 2.1).

Окно про- Главное меню Форма Панель инструментов
екта

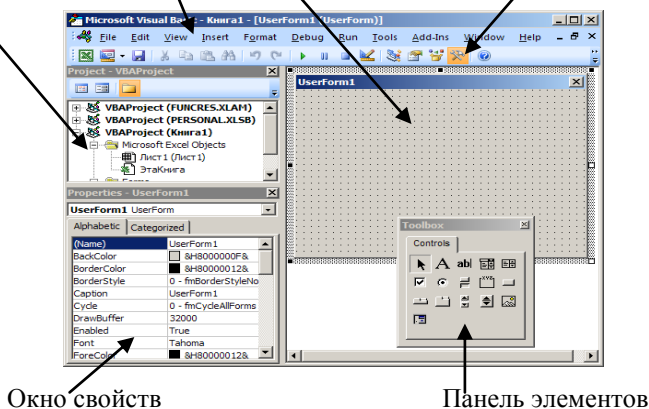


Рис.2.1 Интегрированная среда разработки

Элементы *IDE*:

- **Главное меню** (рис.2.1) содержит полный перечень команд среды разработки VBA.
- **Панель инструментов** (рис.2.2) предоставляет быстрый доступ к наиболее часто используемым командам. Отображается панель инструментов на экране после выполнения команды главного меню *View* ➤ *Toolbars* ➤ *Standart*.

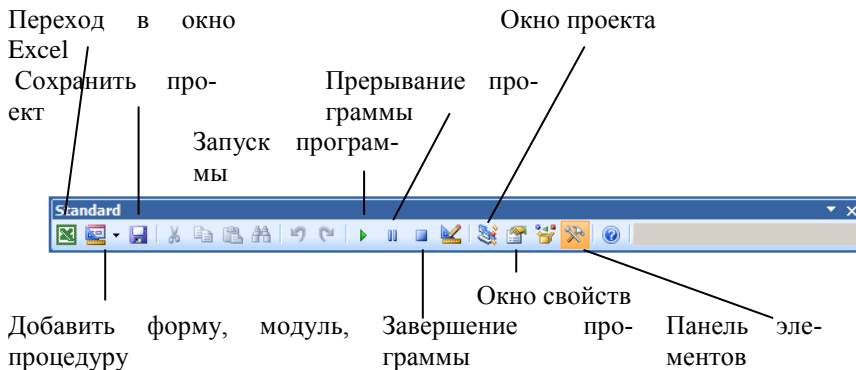


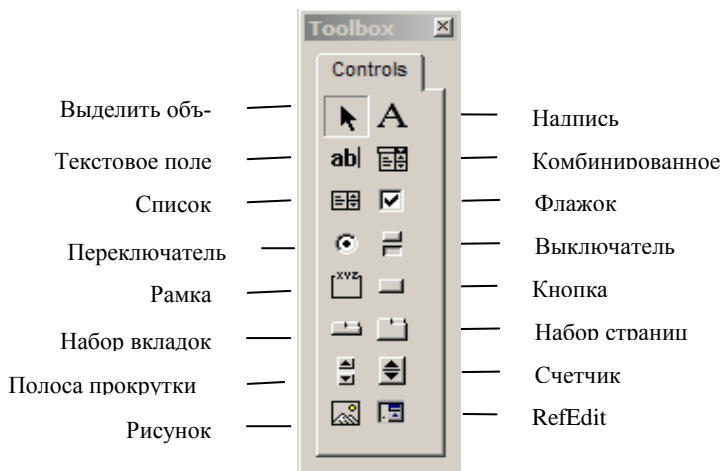
Рис.2.2 Панель инструментов Visual Basic

• **Панель элементов Toolbox** (рис.2.3) содержит объекты, которые размещаются на формах в разрабатываемых приложениях. Такие элементы называются элементами управления. Отображается панель элементов на экране после выполнения команды главного меню *View* ➤ *Toolbox*.

• **Окно проекта** отображается на экране после выполнения команды главного меню *View* ➤ *Project Explorer*. Проект – это набор файлов, используемых для построения приложения. В окне проекта отображаются все элементы программы: формы, модули, классы и т.д.

• **Окно свойств** отображается на экране после выполнения команды главного меню *View* ➤ *Propoties Window*. В этом окне приводится список свойств, относящихся к выделенному объекту. Набор свойств зависит от типа элемента управления. Список свойств состоит из двух столбцов: в первом перечислены названия свойств, во втором – их значения.

• **Форма.** Окно формы является главным элементом приложения. На форме размещаются требуемые элементы управления. Чтобы отобразить форму в окне конструктора форм достаточно выполнить двойной щелчок мыши по ее значку (имени) в окне проекта.

Рис.2.3 Панель инструментов *ToolBox***Задание 1**

1. Запустите программу *Excel* и выполните команду *Alt-F11*.
2. Поместите на форме элемент управления «Текстовое поле» два элемента «Кнопка». С помощью мыши измените размеры элементов и разместите их на форме, как показано на рис. 2.4.

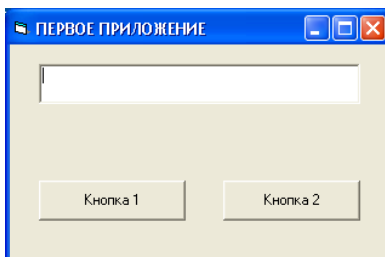


Рис.2.4 Форма первого приложения

3. Для созданных объектов в окне свойств установите следующие значения свойств:

Объект	Свойство	Значение
Форма	<i>Caption</i>	Первое приложение
Текстовое поле	<i>Text</i>	(пустое поле)
Кнопка 1	<i>Caption</i>	Кнопка 1
Кнопка 2	<i>Caption</i>	Кнопка 2

Обратите внимание, что в окне свойств отображаются свойства выделенного объекта!!!

4. Дважды щелкните по кнопке «**Кнопка 1**». Откроется окно редактора кода (рис.1.6), в котором записывается программный код приложения. Введите между строками *Private Sub CommandButton1_Click()* и *End Sub* следующий код:

Text1.Text = "Произошло событие Enter для кнопки 1!"

Процедура обработки события должна выглядеть так, как показано на рис. 2.5.

Имя объекта

Имя события

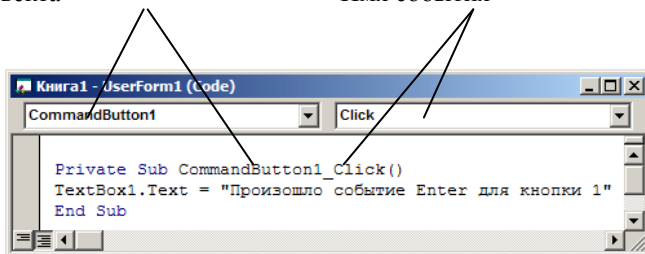


Рис.2.5 Окно редактора кода

Код приложения *VBA* делится на более мелкие блоки, называемые процедурами (*Procedure*). Процедура обработки события – это код, который выполняется, если происходит событие, например, щелчок кнопкой мыши по выделенному элементу. Имя процедуры составляется из имени объекта и имени события, разделенных символом подчеркивания «».

Первая строка программного кода *CommandButton1_Click()* определяет тип процедуры и имя процедуры. Последняя строка *End Sub* сообщает *VBA* о том, что выполнение процедуры пора завершить. Чтобы перейти к нужной процедуре любого объекта, нужно выбрать объект и событие в левом и правом полях со списком окна редактора кода (рис.2.6).

Для обращения к свойствам объекта используется следующий синтаксис:

[Форма.]Имя_объекта.Свойство =Значение

Имя формы указывать необязательно, если обращаются к элементу, принадлежащему этой форме.

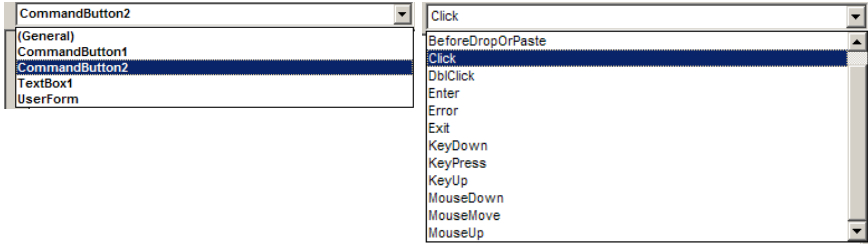


Рис. 2.6 Выбор процедуры

5. Самостоятельно определите код процедуры события нажатия кнопки 2.

6. Чтобы выполнить приложение, нажмите кнопку *Start* (запуск программы) панели инструментов. Нажмите поочередно кнопки на форме. Проследите за состоянием текстового поля.

7. Завершите работу приложения кнопкой *End* (завершение программы) на панели инструментов.

2 Форма, свойства форм.

Форма – это визуальная основа приложений *VBA*. Любое приложение, выводящее информацию на экран, построено на основе формы того или иного типа. Форма представляет собой окно, в котором размещены различные элементы управления (рис. 2.7). Форма появляется на экране при выполнении команды главного меню *VBA Insert ➤ UserForm*.

Insert ➤ UserForm.



Рис.2.7 Объект – форма

Свойства формы

Свойства формы, как и любого объекта, помещенного на форму, определяют внешний вид и его положение.

- *BackColor* определяет цвет фона для формы.
- *BorderStyle* определяет особенности границы, окружающей форму.
- *Caption* название формы, выводимое в строке заголовка окна.
- *ForeColor* определяет цвет текста, выводимого на форме.
- *Height* определяет высоту формы.
- *Left* определяет расстояние формы от левого края экрана.
- *Name* определяет имя объекта. Рекомендуется присваивать форме содержательное имя и снабдить его префиксом *frm*.
- *Top* определяет расстояние формы от верхнего края экрана.
- *Width* определяет ширину формы.

Задание 2.

1. Добавьте форму.
2. Изменяя различные свойства формы, запустите программу.
3. Сделайте выводы: как изменяется внешний вид формы.

3 Программная настройка свойств формы

Значение многих свойств можно задавать как в режиме конструирования, так и в режиме выполнения. Некоторые свойства можно определять только в режиме конструирования, некоторые – только в режиме выполнения. Свойства, доступные только в режиме выполнения, не отображаются в окне свойств в режиме конструирования.

Задание 3

1. Создайте новую форму. Свойство *Name* определите значением *frmForm1*. Свойству *Caption* присвойте значение **Привет**.

2. Дважды щелкните на элементе-кнопке, чтобы разместить на форме кнопку. Задайте свойству *Name* кнопки значение *cmdHello*, а свойству *Caption* – значение *&Hello*.

3. Дважды щелкните на кнопке, чтобы открыть окно программы с процедурой *cmdHello_Click*. Добавьте в процедуру следующий фрагмент:

```
If frmForm1.Caption = "Привет" Then
    frmForm1.Caption = "Пока"
Else
    frmForm1.Caption = "Привет"
End If
```

4. Запустите проект и нажмите кнопку. Сделайте анализ работы программы.

5. Внесите изменения в процедуру:

```
If frmForm1.Caption = "Привет" Then
    frmForm1.Caption = "Пока"
    cmdHello.Caption = "Привет"
Else
    frmForm1.Caption = "Привет"
    cmdHello.Caption = "Пока"
End If
```

Что изменилось?

Дважды щелкните по форме и строку *cmdHello.Caption = "Пока"* в процедуру *UserForm_Activate()* Запустите проект и сравните работу новой версии Вашей программы с предыдущей версией.

2.3. Лабораторная работа «Создание макросов Word»

Цель работы: приобретение навыков формирования элементарных приложений.

Форма проведения: выполнение индивидуального задания.

Форма отчетности: выполнение теста, защита отчета.

1. Автоматическое создание макросов

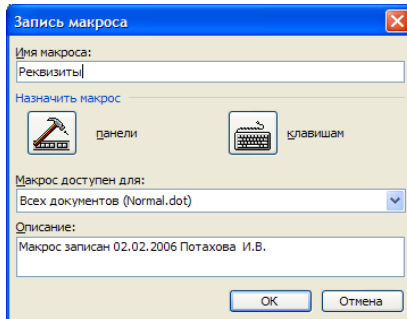
Макросы Microsoft Word хранятся в шаблоне документа. Шаблон — это специальный файл Microsoft Word с расширением .DOT, где хранится информация о стилях оформления документа, настройках, макросах и пользовательских инструментальных панелях. По умолчанию используется общий шаблон NORMAL.DOT, в который записываются макросы. Можно для своих документов создать другой шаблон под

произвольным именем и записывать туда вновь созданные макросы. Эти макросы будут доступны только из документа, открытого в этом шаблоне.

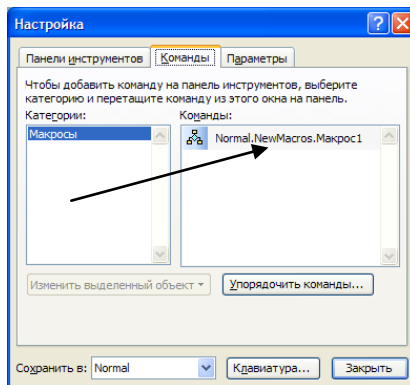
Чтобы записать макрос необходимо выполнить следующие шаги:

1. Выбрать пункт меню **Сервис\Макрос\Начать запись**.
2. В окне ввода **Имя макроса** введите имя макроса, например,

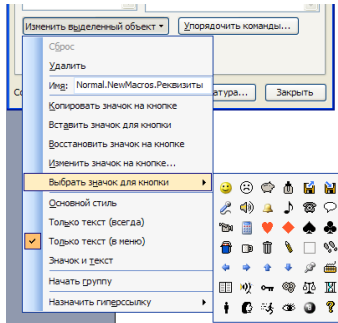
Реквизиты.



3. С помощью кнопок *Назначить макрос панели* поставить создаваемому макросу для быстрого вызова в соответствие свою кнопку на панели инструментов.



4. Нажать кнопку *Изменить выделенный объект* и в предложенном списке выбрать команду *Выбрать значок для кнопки* либо *Изменить значок для кнопки*.



5. Установить флаг *Только текст* (в меню)
6. В строке ввода *Макрос доступен для* выбрать шаблон, с которым требуется связать создаваемый макрос.
7. Щелкнуть по кнопке *OK*, чтобы перейти в режим записи макроса.


После выполнения этих шагов система переходит в режим протоколирования действий пользователя. Каждое действие записывается в виде команд Visual Basic. При этом манипулятор мышь можно использовать только для выбора команд меню и кнопок на панелях инструментов. Все остальные действия (выделение фрагментов текста, таблицы и т.п.) выполняются с помощью клавиатуры.

Например, введем реквизиты предприятия

Банковские реквизиты по учету бюджетных средств:

ИНН 7021000043 КПП 701701001
 УФК МФ РФ по Томской области
 (с/с 40105810020) ((ГОУВПО) Томский государственный университет систем управления и радиоэлектроники)
 Р/сч 4010581030000010001 БИК 046902001 ГРКЦ ГУ Банка России по Томской области
 л/сч 03073137150

Во время записи макроса на экране находится панель **Остановка**

записи макроса , на которой находятся две кнопки: **СТОП** и **ПАУЗА**. Первая прекращает запись макроса, а вторая приостанавливает

ет запись. Вторая кнопка используется только тогда, когда не все действия пользователя требуется записать в макрос.

Задания

Вариант 1

Создайте макрос для Word, изменяющий тип оформления (а также толщину и цвет оформления) фрагмента текста. Назначить макросу кнопку на панели инструментов, выбрав либо создав уникальный значок. Название значка определить только во всплывающей подсказке. Внести в код созданного макроса изменения, обеспечивающие возможность выбора устанавливаемых параметров макроса, например, выбор цвета для шрифта и т.п. Для этого разработать пользовательскую форму, в которой в качестве элемента управления могут быть использованы полоса прокрутки, список, поле со списком.

Изменить тело макроса таким образом, чтобы устанавливаемые характеристики передавались в процедуру макроса через параметры соответствующей процедуры

Вариант 2

Создайте макрос для Word, переносящий фрагмент текста в новую позицию и изменяющий его стиль. Назначить макросу кнопку на панели инструментов, выбрав либо создав уникальный значок. Название значка определить только во всплывающей подсказке. Внести в код созданного макроса изменения, обеспечивающие возможность выбора устанавливаемых параметров макроса, например, выбор цвета для шрифта и т.п. Для этого разработать пользовательскую форму, в которой в качестве элемента управления могут быть использованы полоса прокрутки, список, поле со списком.

Изменить тело макроса таким образом, чтобы устанавливаемые характеристики передавались в процедуру макроса через параметры соответствующей процедуры

Вариант 3

Создайте макрос для Word, преобразующий фрагмент текста в маркированный список. Назначить макросу кнопку на панели инструментов, выбрав либо создав уникальный значок. Название значка определить только во всплывающей подсказке. Внести в код созданного макроса изменения, обеспечивающие возможность выбора устанавливаемых параметров макроса, например, выбор цвета для шрифта и т.п. Для этого разработать пользовательскую форму, в которой в качестве

элемента управления могут быть использованы полоса прокрутки, список, поле со списком.

Изменить тело макроса таким образом, чтобы устанавливаемые характеристики передавались в процедуру макроса через параметры соответствующей процедуры

Вариант 4

Создайте макрос для Word, изменяющий в абзаце цвет, размер, начертание символов. Назначить макросу кнопку на панели инструментов, выбрав либо создав уникальный значок. Название значка определить только во всплывающей подсказке. Внести в код созданного макроса изменения, обеспечивающие возможность выбора устанавливаемых параметров макроса, например, выбор цвета для шрифта и т.п. Для этого разработать пользовательскую форму, в которой в качестве элемента управления могут быть использованы полоса прокрутки, список, поле со списком.

Изменить тело макроса таким образом, чтобы устанавливаемые характеристики передавались в процедуру макроса через параметры соответствующей процедуры

Вариант 5

Создайте макрос для Word, добавляющий в таблицу два смежных столбца слева от заданного столбца. Назначить макросу кнопку на панели инструментов, выбрав либо создав уникальный значок. Название значка определить только во всплывающей подсказке. Внести в код созданного макроса изменения, обеспечивающие возможность выбора устанавливаемых параметров макроса, например, выбор цвета для шрифта и т.п. Для этого разработать пользовательскую форму, в которой в качестве элемента управления могут быть использованы полоса прокрутки, список, поле со списком.

Изменить тело макроса таким образом, чтобы устанавливаемые характеристики передавались в процедуру макроса через параметры соответствующей процедуры

Вариант 6

Создайте макрос для Word, разбивающий текст на колонки. Назначить макросу кнопку на панели инструментов, выбрав либо создав уникальный значок. Название значка определить только во всплывающей подсказке. Внести в код созданного макроса изменения, обеспечивающие возможность выбора устанавливаемых параметров макроса, например, выбор цвета для шрифта и т.п. Для этого разработать пользова-

тельскую форму, в которой в качестве элемента управления могут быть использованы полоса прокрутки, список, поле со списком.

Изменить тело макроса таким образом, чтобы устанавливаемые характеристики передавались в процедуру макроса через параметры соответствующей процедуры

Вариант 7

Создайте макрос для Word, изменяющий параметры абзаца (выравнивание, интервал, отступ первой строки) Назначить макросу кнопку на панели инструментов, выбрав либо создав уникальный значок. Название значка определить только во всплывающей подсказке. Внести в код созданного макроса изменения, обеспечивающие возможность выбора устанавливаемых параметров макроса, например, выбор цвета для шрифта и т.п. Для этого разработать пользовательскую форму, в которой в качестве элемента управления могут быть использованы полоса прокрутки, список, поле со списком. Изменить тело макроса таким образом, чтобы устанавливаемые характеристики передавались в процедуру макроса через параметры соответствующей процедуры

Вариант 8

Создайте макрос для Word, Добавляющий номера строк и изменяющий начертание символов. Назначить макросу кнопку на панели инструментов, выбрав либо создав уникальный значок. Название значка определить только во всплывающей подсказке. Внести в код созданного макроса изменения, обеспечивающие возможность выбора устанавливаемых параметров макроса, например, выбор цвета для шрифта и т.п. Для этого разработать пользовательскую форму, в которой в качестве элемента управления могут быть использованы полоса прокрутки, список, поле со списком.

Изменить тело макроса таким образом, чтобы устанавливаемые характеристики передавались в процедуру макроса через параметры соответствующей процедуры

Вариант 9

Создайте макрос для Word, устанавливающий новые параметры страницы. Назначить макросу кнопку на панели инструментов, выбрав либо создав уникальный значок. Название значка определить только во всплывающей подсказке. Внести в код созданного макроса изменения, обеспечивающие возможность выбора устанавливаемых параметров макроса, например, выбор цвета для шрифта и т.п. Для этого разработать пользовательскую форму, в которой в качестве элемента управле-

ния могут быть использованы полоса прокрутки, список, поле со списком.

Изменить тело макроса таким образом, чтобы устанавливаемые характеристики передавались в процедуру макроса через параметры соответствующей процедуры

Вариант 10

Создайте макрос для Word, добавляющий к тексту номера страниц. Назначить макросу кнопку на панели инструментов, выбрав либо создав уникальный значок. Название значка определить только во всплывающей подсказке. Внести в код созданного макроса изменения, обеспечивающие возможность выбора устанавливаемых параметров макроса, например, выбор цвета для шрифта и т.п. Для этого разработать пользовательскую форму, в которой в качестве элемента управления могут быть использованы полоса прокрутки, список, поле со списком.

Изменить тело макроса таким образом, чтобы устанавливаемые характеристики передавались в процедуру макроса через параметры соответствующей процедуры

2.4. Лабораторная работа «Создание макросов Excel»

Цель работы: изучить встроенный сервис MacroRecorder для записи пользовательских макросов.

Форма проведения: выполнение индивидуального задания.

Форма отчетности: выполнение теста, защита отчета.

Пример для ознакомления

Вы — заведующий больницей. Вам предлагается составить штатное расписание, то есть определить, сколько сотрудников, с каким окладом и на какие должности необходимо принять на работу. Общий месячный фонд зарплаты составляет \$10000.

Предположим, что для нормальной работы больницы нужно 5–7 санитарок, 8–10 медсестер, 10–12 врачей, 1 заведующий аптекой, 3 заведующих отделениями, 1 главный врач, 1 заведующий хозяйством, 1 заведующий больницей.

Предлагается следующая модель решения задачи. За основу берется оклад санитарки. Размер оклада остальных сотрудников определяется по формуле

Оклад = A (Оклад санитарки) + B ,

где A — коэффициент оклада;

B — величина надбавки, \$.

Значения A и B назначаются, исходя из следующих соображений:

- медсестра должна получать в 1,5 раза больше санитарки;
- врач — в 3 раза больше санитарки;
- заведующий отделением — на \$30 больше, чем врач;
- заведующий аптекой — в 2 раза больше санитарки;
- заведующий хозяйством — на \$40 больше медсестры;
- главный врач — в 4 раза больше санитарки;
- заведующий больницей — на \$20 больше главного врача.

1. Оформите таблицу, используя следующие столбцы: **Должность, Количество сотрудников, Коэффициент оклада, Надбавка, Оклад, Итого.**

При решении задачи используйте сервисную функцию Excel «Подбор параметра»: **Сервис | Подбор параметра** (рис. 2.8).

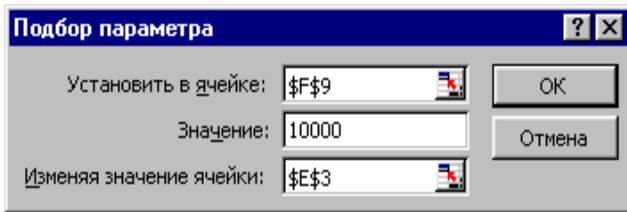


Рис. 2.8 Диалоговое окно **Подбор параметра**

В поле **Установить в ячейке** ввести адрес ячейки, где вычисляется общая месячная зарплата всех сотрудников больницы. В поле **Значение** ввести предельное значение месячного фонда зарплаты. В поле **Изменяя значение ячейки** ввести адрес ячейки, где находится оклад санитарки. После нажатия **ОК** произойдет автоматический подбор значения оклада санитарки таким образом, чтобы общий месячный фонд зарплаты составил \$10000.

Чтобы упростить эту работу, создайте простейший макрос — программу на языке VBA (Visual Basic for Application), встроенном в

офисные программы. Это можно сделать, не зная пока самого языка, с помощью транслятора MacroRecorder, который переводит на язык VBA действия пользователя с момента его запуска до окончания записи макроса. Для активизации MacroRecorder выбираем команду **Сервис | Макрос | Начать запись**. В появившемся диалоговом окне **Запись макроса** (рис. 2.9) задаем имя макроса (например, «Staff») и описание макроса (необязательно).

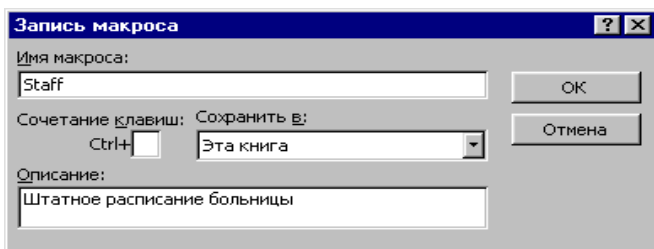


Рис. 2.9 Диалоговое окно **Запись макроса**

В поле **Сохранить в:** оставляем опцию по умолчанию **Эта книга** (тогда созданный макрос сохранится на новом листе модуля в активной рабочей книге). Будущий макрос можно запускать с помощью сочетания клавиш клавиатуры, например, Ctrl+z, если это указать в поле **Сочетание клавиш**.

После нажатия **ОК** все ваши действия над ячейками будут записываться. Для того чтобы остановить запись, необходимо нажать на кнопку **Остановить запись** на появившейся на экране плавающей панели или выполнить команду **Сервис | Макрос | Остановить запись**.

Итак, вызовем сервисную функцию **Подбор параметра**, выполним описанные ранее действия по расчету штатного расписания и остановим запись макроса.

Чтобы посмотреть, какая же все-таки VBA-программа «создана» нами, выполним команду **Сервис | Макрос | Макросы |**. В появившемся диалоговом окне выберем макрос с именем «Staff» и нажмем кнопку **Изменить**. Откроется главное окно редактора VBA с текстом записанного макроса, например:

```
Sub Staff()
    ' Staff Макрос
    ' Штатное расписание больницы
```

```

Range("I14").Select
Range("I14").GoalSeek
Goal:=10000, ChangingCell:=Range("H6")
End Sub

```

Именно эта процедура и выполняется, если в диалоговом окне **Макросы** нажать кнопку **Выполнить** или на клавиатуре набрать указанное сочетание **Ctrl+z**. Для заданного нового количества штатных единиц будут рассчитаны новые оклады.

Но можно и самому создать на листе **кнопку**, при нажатии на которую будут производиться нужные действия.

Кнопка является одним из элементов управления листа, создаваемых с помощью панели инструментов **Формы**. Обычно этой панели нет на экране, поэтому выполняем команду **Сервис | Настройка | Панели инструментов | Формы**. На экран выводится панель инструментов **Формы** (рис. 2.9). Выбираем на ней щелчком мыши форму **Кнопка**. При этом указатель мыши превращается в тонкий крестик. Щелкаем им по листу. На нем появляется кнопка с именем *Кнопка1* и одновременно открывается диалоговое окно **Назначение макроса объекту**. В поле **Имя макроса** выбираем имя нашего макроса «Stuff».

The screenshot shows an Excel spreadsheet with the following data table:

Должность	Количество сотрудников	Оклад \$	Итого \$
Санитарка	7	125,45	878,34
Медсестра	10	185,22	1852,17
Врач	1,2	376,43	4517,20
Зав. отделением	3	408,43	1219,30
Зав. аптекой	1	250,06	250,06
Зав. хозяйством	1	228,22	228,22
Главный врач	1	501,91	501,91
Зав. больницей	1	521,91	521,91
Итого	36	3апуск	10000,00

The spreadsheet also shows a button labeled 'Запуск' (Start) in cell I14, which is the target of the Goal Seek macro.

Рис. 2.9 Лист Excel с итоговой таблицей

Теперь указанная выше процедура расчета окладов будет выполняться простым нажатием кнопки.

Можно изменить формат кнопки (шрифт надписи, размер и т.п.). Для этого следует вызвать контекстно-зависимое меню и выполнить необходимые операции.

Кнопку вызова макроса можно разместить и на любой из панелей инструментов. Для этого выполняем команду **Сервис | Настройка | Команды | Макросы | Настраиваемая кнопка**. Удерживая левую кнопку мыши, перетаскиваем кнопку 😊 на панель инструментов. Вызвав контекстно-зависимое меню, выбираем пункт **Назначить макрос**. В появившемся диалоговом окне выбираем имя нашего макроса Staff. Закрываем диалоговое окно **Настройка**. Кнопка готова к работе. Можно отредактировать всплывающее имя кнопки и рисунок на ней. Для этого необходимо сначала щелкнуть по ней правой кнопкой мыши и в появившемся меню выбрать **Настройка**. Затем еще раз щелкнуть по ней правой кнопкой мыши и в контекстно-зависимом меню выполнить необходимые операции.

3. Измените макрос таким образом, чтобы можно было в некоторой ячейке задавать произвольное значение фонда зарплаты и под него рассчитывать оклады сотрудников.

Задания на лабораторную работу

Вариант 1

Создайте макрос для Excel, изменяющий цвет и размер символов в ячейке.

Вариант 2

Создайте макрос для Excel, производящий автозаполнение строки ячеек месяцами.

Вариант 3

Создайте макрос для Excel, устанавливающий название рабочего листа.

Вариант 4

Создайте макрос для Excel, меняющий местами содержимое двух ячеек.

Вариант 5

Создайте макрос для Excel, меняющий местами заданные строки.

Вариант 6

Создайте макрос для Excel, изменяющий формат вводимого в ячейке числа.

Вариант 7

Создайте макрос для Excel, вставляющий формулу в ячейку.

Вариант 8

Создайте макрос для Excel, меняющий местами заданные столбцы.

Вариант 9

Создайте макрос для Excel, присваивающий ячейке имя и центрирующий ее содержимое.

Вариант 10

Создайте макрос для Excel, добавляющий в ячейку текст заданного цвета.

2.5. Лабораторная работа «Основы программирования в VBA»

Цель работы: изучить основы программирования в среде VBA, закрепление навыков создания собственных приложений.

Форма проведения: выполнение индивидуального задания.

Форма отчетности: выполнение теста, защита отчета.

Основные понятия

1. Выражение. Правила построения выражений в VBA

В основе программирования на любом алгоритмическом языке лежит построение выражений. Выражение строится из переменных, констант, встроенных функций с использованием знаков, операций и скобок. Запись выражения задает правило вычисления его значения и типа.

Перечислим основные операции, которые можно использовать в VBA при построении выражений (табл. 2.1).

Таблица 2.1 Операции и их приоритет

Приорит	Арифметические	Сравнения	Логические	Описание некоторых операций
1	Возведение в степень (^)	Равенство	Отрицание	При возведении в степень основание и показатель

		(=)	(Not)	могут быть арифметическими выражениями любого типа. Результат имеет тип Double
2	Унарный минус (-)	Неравенство (<>)	Конъюнкция (And)	
3	Умножение, деление (*, /)	Меньше (<)	Дизъюнкция (Or)	
4	Деление нацело (\)	Больше (>)	Исключительное ИЛИ (Xor)	Деление нацело определено над целочисленными данными (применимо и к вещественным данным) и дает результат целого типа. Исключительное ИЛИ требует, чтобы один из операндов имел значение, отличное от True
5	Остаток от деления нацело (mod)	Меньше или равно (<=)	Эквивалентность (Eqv)	Операция mod определена над данными целого типа и возвращает результат целого типа
6	Сложение, вычитание (+, -)	Больше или равно (>=)	Импликация (Imp)	Среди логических операций определена операция следования (импликация), ложная в единственном случае — когда посылка истинна, а заключение ложно
7	Конкатенация строк (&)	Подобия (Like).		Операция Like проверяет соответствие строки образцу.

При построении выражений необходимо учитывать следующее:

- если выражение содержит операции разных категорий, то первыми выполняются арифметические операции, затем операции сравнения, и последними — логические;
- все операции сравнения имеют один и тот же приоритет. Арифметические и логические операции выполняются в соответствии с указанным приоритетом;

- одна и та же операция, записанная несколько раз подряд, или операции одного приоритета выполняются слева направо — из двух операций первой выполняется та, которая стоит левее в записи выражения;

- скобки позволяют изменить указанный порядок вычисления выражения, поскольку выражения в скобках имеют наивысший приоритет и вычисляются первыми. Внутри скобок действует обычный порядок следования;

- операция конкатенации не является арифметической, но при ее появлении в выражении она выполняется после всех арифметических операций, но до вычисления операций сравнения.

2. Функции обработки числовых данных

Помимо вышеуказанных операций VBA предоставляет набор математических функций, расширяющий возможности обработки числовых данных. Перечислим базовые функции:

- **Abs(число)** — абсолютное значение числа;
- **Atn(число)** — арктангенс (в радианах) аргумента, задающего тангенс угла;
- **Cos(число)** — косинус угла. Аргумент число задает угол в радианах;
- **Sin(число)** — синус угла;
- **Exp(число)** — экспонента, т.е. результат возведения в степень числа e ;
- **Log(число)** — натуральный логарифм числа;
- **Rnd[(число)]** — результат представляет число равномерно распределенное случайное число в интервале $[0-1]$. Если аргумент число не задан или больше нуля, то порождается очередное случайное число. Если он равен 0, то результатом будет предыдущее случайное число, а если число меньше нуля, то каждый раз порождается одно и то же число, определяемое аргументом. Перед тем, как получить последовательность случайных чисел, необходимо вызвать функцию `Randomize` для инициализации последовательности;
- **Sng(число)** — знак числа (если число больше нуля — 1, равно нулю — 0, меньше нуля — -1);
- **Sqr(число)** — квадратный корень;
- **Tan(число)** — тангенс угла.

Задания на лабораторную работу.

Для каждого задания соответствующего варианта разработать индивидуальную форму

Вариант 1

1. Написать программу вычисления выражения для введенных значений x , y , z :

$$H = 5 \operatorname{arctg}(x) - \frac{1}{4} \cos \frac{x + 3 \cdot |x - y| + x^2}{|x + y|^z + x^3} - e^{xyz}$$

2. Вычислить функцию:

$$Z = \begin{cases} \sqrt{3y - 5x}, \dots \text{если} \dots x < y \leq 2x \\ \sqrt{3y + 5x}, \dots \text{если} \dots y > 2x \end{cases}$$

Вариант 2

1. Написать программу вычисления выражения для введенных значений x , y , z :

$$H = \operatorname{Ln}(y^{-\sqrt{x}}) \cdot \left(x - \frac{y}{2}\right) + \sin^2(\operatorname{arctg}(z)) + e^{x+y}$$

2. Вычислить функцию:

$$Z = \begin{cases} \operatorname{Ln}(|x| + |y|), \dots \text{если} \dots 0 < x < y + 1 \\ \operatorname{Lg}(|x + y|), \dots \text{если} \dots x \geq y \end{cases}$$

Вариант 3

1. Написать программу вычисления выражения для введенных значений x , y , z :

$$H = \frac{e^{|x-y|} \cdot |x-y|^{x+y}}{\operatorname{arctg}(x) + \operatorname{tg}(z)} + \sqrt{x^6 + \ln^2(y)}$$

2. Вычислить функцию:

$$Z = \begin{cases} \sqrt{|x \cdot e^2 - y \cdot e^2|}, \dots \text{если} \dots |y| < x < 2 \cdot |y| \\ \sqrt{|x \cdot y|}, \dots \text{если} \dots x \geq 2 \cdot |y| \end{cases}$$

Вариант 4

1. Написать программу вычисления выражения для введенных значений x , y , z :

$$H = \left| x^{\frac{y}{x}} - \sqrt{\frac{y}{x}} \right| + (y - x) \frac{\cos y - e^{\frac{z}{y-x}}}{1 + (y - x)^2}$$

2. Вычислить функцию:

$$Z = \begin{cases} \frac{3x - y}{x^2 + y^2}, \dots \text{если} \dots |x| > |y| \\ x \cdot y, \dots \text{если} \dots \pi < |x| \leq |y| \end{cases}$$

Вариант 5

1. Написать программу вычисления выражения для введенных значений x , y , z :

$$H = y + \frac{e^{x-y}}{x^2} (1 + \operatorname{tg}^2 \frac{z}{2}) \sqrt{|y| + 6} \\ y + \frac{x^3}{y}$$

2. Вычислить функцию:

$$Z = \begin{cases} \sqrt{15x^2 + 20y^2}, \dots \text{если} \dots x > y \\ \sqrt{15y^2 + 20x^2}, \dots \text{если} \dots 1 < x \leq y \end{cases}$$

Вариант 6

1. Написать программу вычисления выражения для введенных значений x , y , z :

$$H = \frac{1 + \frac{x^2}{2} + \frac{x^2}{3} + 4x}{x(\sin \operatorname{arctg}(z) - \cos^2 y)} + \lg(z + yx)$$

2. Вычислить функцию:

$$Z = \begin{cases} \operatorname{tg}(x - 2y), \dots \dots \dots \text{если} \dots |x + y| \geq 2 \\ \ln(|x - 2y|), \dots \dots \dots \text{если} \dots 0.5 < |x + y| < 2 \end{cases}$$

Вариант 7

1. Написать программу вычисления выражения для введенных значений x , y , z :

$$H = \sqrt{10\sqrt{x + x^{y+2}}} \cdot (\sin^2 z + |x + y|) \cdot e^z$$

2. Вычислить функцию:

$$Z = \begin{cases} 3x^4 + 4y^3, \dots \dots \dots \text{если} \dots |x| + |y| \geq 1 \\ x^2 - y^2, \dots \dots \dots \text{если} \dots |x| + |y| < 1 \end{cases}$$

Вариант 8

1. Написать программу вычисления выражения для введенных значений x , y , z :

$$H = \frac{1 + \sin^2(x + y)}{\left| e^x - \frac{2y}{1 + x^2 y^3} \right|} \cdot x^{|y|} + \cos^2\left(\operatorname{arctg} \frac{1}{z}\right)$$

2. Вычислить функцию:

$$Z = \begin{cases} \lg(|2x - 3e \cdot y|), \dots \dots \dots \text{если} \dots |x| \leq 5y \\ \lg(|2 \cdot x - 3e^2 \cdot y|), \dots \dots \dots \text{если} \dots x > 5y \end{cases}$$

Вариант 9

1. Написать программу вычисления выражения для введенных значений x , y , z :

$$H = \left| \cos x - e^y \right|^{1+2\ln^2 y} \cdot \left(1 + z + \frac{z^2}{2} + \frac{z^3}{3} \right)$$

2. Вычислить функцию:

$$Z = \begin{cases} \ln(|x| + |y|), & \dots \text{если} \dots |x \cdot y| > 10 \\ e^{x+y}, & \dots \text{если} \dots |x \cdot y| \leq 10 \end{cases}$$

Вариант 10

1. Написать программу вычисления выражения для введенных значений x , y , z :

$$H = \frac{x^{y+1} + e^{y-1}}{1 + x(y - tgz)} (1 + |y - x|) + \frac{|y - x|^2}{2} - \frac{|y - x|^3}{3}$$

2. Вычислить функцию:

$$Z = \begin{cases} \arctg(x + |y|), & \dots \text{если} \dots x < y \\ e^{tg(|x|+y)}, & \dots \text{если} \dots x \geq y \end{cases}$$

2.6. Лабораторная работа «Строки VBA»

Цель работы: изучить возможности VBA по обработке текстовых данных, закрепление навыков создания собственных приложений.

Форма проведения: выполнение индивидуального задания.

Форма отчетности: выполнение теста, защита отчета.

Основные понятия

1. Операции над строками

Над строковыми переменными, определенными в проектах VB допустимо выполнение двух видов операций: сравнения и конкатенации строк.

Операция конкатенации используется для сцепления двух или нескольких строк. Обозначается данная операция знаком «+» либо знаком «&». В случае если применяется первый знак, то в качестве аргументов выражения, определяющего операцию конкатенации должны выступать переменные или константы строкового типа. В случае применения второго знака (&) – один из аргументов может быть переменной или константой типа число или дата. Например:

Dim stroka As String

Dim Ver As Single

Ver = 1

*stroka = "Компьютерная " + "подготовка " + " часть " & Ver
MsgBox (stroka)*

При сравнении строк применимы обычные операции сравнения. При этом сравнение может быть осуществлено в соответствии с расположением строк в словаре либо побитно. Второй тип сравнения обладает чувствительностью к регистру. Чтобы определить тип сравнения, необходимо в начале модуля поместить инструкцию Option Compare Text | Binary.

Сравнение строк с образцом осуществляется с использованием операции Like. При задании образца используются специальные символы (табл. 2.2), позволяющие разнообразить операцию сравнения.

Таблица 2.2. Специальные символы, используемые при задании шаблона

Символы	Интерпретация	Примеры
*	Любой текст — произвольное число	Шаблону Agent* соответствуют все тексты, начинающиеся со слова Agent. Строки Agent007 и Agent Майор Пронин удовлетворяют шаблону
?	Один любой символ	Шаблону К?к удовлетворяют строки Кок и Кук
#	Любая цифра от 0 до 9	Шаблону Agent### соответствует 1000 различных строк, среди которых и Agent007
[множество символов]	Любой символ, принадлежащий множеству	Задать множество можно с помощью перечисления и интервалов. Шаблоны К[аоу]к удовлетворяют слова «Как», «Кок», «Кук»
[!множество символов]	Любой не принадлежащий множеству символ	Шаблоны [!а-я] удовлетворяет символ, не являющийся буквой русского алфавита

2. Основные функции обработки строковых переменных

Функция **Len(string)** возвращает длину строки (число символов), которая задана аргументом String.

Функция **InStr** определяет позицию первого вхождения одной строки внутри другой строки. Синтаксис:

InStr([start,]string1, string2[, compare])

Необязательный аргумент start задает позицию, с которой начинается поиск (по умолчанию — с первого символа строки). String1 — строка, в которой осуществляется поиск, string2 — подстрока, вхождение которой ищется. Необязательный аргумент compare указывает способ сравнения строк. Его значение по умолчанию 0 используется для выполнения двоичного сравнения; 1 задает посимвольное сравнение без учета регистра.

Функция **Left(string, length)** выделяет в строке string указанное число length символов слева.

Функция **Rigth(string, length)** выделяет в строке string указанное число length символов справа.

Функция **Mid(string, start[, length])** позволяет выделить из строки string подстроку длины length, начиная с позиции start.

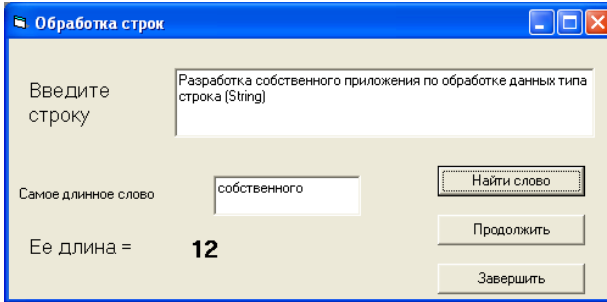
Функции **Ltrim(string)**, **Rtrim(length)**, **Trim(length)** возвращают копию строки, из которой удалены пробелы, находящиеся в начале строки, в конце строки или в начале и в конце строки соответственно.

Функции **Lcase(string)**, **UCase(string)** возвращают копию строки, символы которой приведены к нижнему или к верхнему регистру.

Replace

Пример выполнения задания.

Задана строка символов. Группы символов, разделенные пробелом и не содержащие пробелов внутри себя, будем называть словами. Найти самое длинное слово и определить его длину.



```
Private Sub Завершить_Click()
```

```
End
```

```
End Sub
```

```
Private Sub Продолжить_Click()
```

```
' Сбросить значения предыдущего решения
```

```
Строка.Text = ""
```

```
Слово.Text = ""
```

```
Длина.Caption = ""
```

```
' Передать управление полю ввода анализируемой строки
```

```
Строка.SetFocus
```

```
End Sub
```

```
Private Sub Поиск_Click()
```

```
Dim s As String
```

```
Dim slovo As String, slovo_max As String
```

```
Dim i, j, k, max_dlina
```

```
k = 1 ' начало первого слова
```

```
s = Строка.Text + " " ' Ввести значение исходной строки
```

```
max_dlina = 0
```

```
For i = 1 To Len(s)
```

```
  If Mid(s, i, 1) = " " Then
```

```
    ' Найдено очередное слово
```

```
    slovo = Mid(s, k, i - k) ' Выделить слово
```

```
    If max_dlina < Len(slovo) Then
```

```
      max_dlina = Len(slovo)
```

```
      slovo_max = slovo
```

```
    End If
```

```
    k = i + 1 ' Позиция начала очередного слова
```

End If
Next i
Слово.Text = slovo_max
Длина.Caption = max_dлина
End Sub

Задания.

1. Задана строка символов. Группы символов, разделенные пробелом и не содержащие пробелов внутри себя, будем называть словами. Найти количество слов в строке, у которых первый и последний символ совпадают.
2. Написать программу, которая проверяет, является ли введенная с клавиатуры строка дробным числом.
3. Задана строка символов. Группы символов, разделенные пробелом и не содержащие пробелов внутри себя, будем называть словами. Удалить из каждого слова строки все последующие вхождения его первой буквы.
4. Написать программу, которая преобразует введенное с клавиатуры десятичное число в шестнадцатеричное.
5. Задана строка символов. Группы символов, разделенные пробелом и не содержащие пробелов внутри себя, будем называть словами. Вывести все слова, которые заканчиваются на букву «я».
6. Написать программу, которая преобразует введенное с клавиатуры десятичное число в восьмеричное.
7. Задана строка символов. Группы символов, разделенные пробелом и не содержащие пробелов внутри себя, будем называть словами. Определить количество слов, которые содержат ровно две буквы *a*.
8. Написать программу, которая преобразует введенное с клавиатуры двухразрядное шестнадцатеричное число в десятичное.
9. Написать программу, которая преобразует строку текста, удаляя из нее все слова с четными номерами и переворачивая (печатая символы в обратном порядке) слова с нечетными номерами.
10. Дана строка символов. Подсчитать количество цифр, входящих в данную строку.

2.7. Лабораторная работа «Массивы в VBA»

Цель работы: изучить возможности VBA по обработке множества однотипных данных, закрепление навыков создания собственных приложений.

Форма проведения: выполнение индивидуального задания.

Форма отчетности: выполнение теста, защита отчета.

Основные понятия

Массив — это самый распространенный структурный тип данных. Массив представляет собой упорядоченную совокупность данных одного типа. Порядок элементов задается индексами, то есть порядковыми номерами элементов в соответствующей структуре. В VBA массивы могут одномерными и многомерными. Допустимое число измерений около 60.

При описании массивов к уже знакомой структуре описания переменных добавляется необходимость указания размерности:

```
{ Dim | Public | Private | Static } <имя переменной> (<список размерностей>) [As <имя типа>]
```

Каждое измерение в списке отделяется запятой и определяется заданием верхней и нижней границы изменения индексов. Массивы — это структуры с прямым доступом к элементам. Доступ осуществляется посредством указания имени массива и индекса элемента. Индекс элемента прописывается в круглых скобках.

Приведем пример:

```
Public Sub MyArray()  
    Const LowBound As Integer = -5, HighBound As Integer = 5  
    Dim MyArr(LowBound To HighBound) As Byte  
    Dim I As Integer  
    Debug.Print "Элементы массива MyArr:"  
    For I = LowBound To HighBound  
        MyArr(I) = I + 6  
        Debug.Print MyArr(I)  
    Next I  
End Sub
```

При исполнении данной процедуры будут напечатаны числа от 1 до 11.

Рассмотренный пример показывает работу со статическим массивом. Количество элементов такого массива определено в момент объявления его в программе. VBA имеет мощное средство работы с массивами — динамические массивы. Массив считается динамическим, если при первоначальном объявлении не указывается его размерность, но в последующем она может быть определена и переопределена оператором

ром ReDim. Размерность определяется динамически в той процедуре и в тот момент, когда она становится фактически известной. При этом границы изменения индексов можно задавать не только как константы, но и как выражения, зависящие от переменных. Если же при переопределении массива задать ключевое слово Preserve, можно сохранить все ранее полученные значения элементов и расширить массив, добавив новые элементы.

Приведем пример работы с динамическим массивом:

```
'Объявление динамического массива на уровне модуля
Public Vector() As Integer
```

```
Public Sub DinArray()
```

```
'Определяется фактическая размерность массива Vector
```

```
Dim N As Byte, I As Byte
```

```
N = InputBox("Введите число элементов вектора")
```

```
ReDim Vector(1 To N)
```

```
For I = 1 To N
```

```
Vector(I) = 2 * I + 1
```

```
Next I
```

```
' Массив расширяется с сохранением ранее вычисленных элементов
```

```
ReDim Preserve Vector(1 To 2 * N + 1)
```

```
For I = N + 1 To 2 * N + 1
```

```
Vector(I) = 2 * I
```

```
Next I
```

```
'А теперь печать
```

```
Debug.Print "Элементы Vector:" & Chr(13)
```

```
For I = 1 To N * 2 + 1
```

```
Debug.Print Vector(I)
```

```
Next I
```

```
End Sub
```

При исполнении данной процедуры будет выдан запрос на определение количества элементов массива, затем печать выведенных значений. Например, если ввести число 10, то вначале будут напечатаны нечетные числа от 3 до 21, а затем четные от 22 до 42.

Задания.

1. Записать положительные элементы массива $x=(x_1, x_2, \dots, x_n)$ подряд в массив $y=(y_1, y_2, \dots, y_k)$. Определить k - количество положи-

тельных элементов. Вычислить
$$S = \sum_{i=1}^k y_i$$
.

2. Записать элементы массива $A = (a_1, a_2, \dots, a_n)$ с четными индексами подряд в массив $B = (b_1, b_2, \dots, b_k)$. Определить k – количество

$$P = \prod_{i=1}^k b_i$$

четных элементов. Вычислить

3. Записать пять первых положительных элементов массива

$$S = \sum_{i=1}^5 y_i$$

$x=(x_1, \dots, x_n)$ подряд в массив $Y=(y_1, y_2, \dots, y_5)$. Вычислить

4. Записать элементы массива $x=(x_1, x_2, \dots, x_n)$, удовлетворяющие условию $x_i \in [1, 2]$, подряд в массив $y=(y_1, y_2, \dots, y_k)$. Определить k –

$$P = \prod_{i=1}^k y_i$$

количество таких элементов. Вычислить

5. Записать элементы массива $x=(x_1, x_2, \dots, x_n)$ в обратном порядке в массив $Y=(y_1, y_2, \dots, y_n)$. Вычислить произведение элементов массива Y с четными индексами.

6. Записать элементы массива $X=(x_1, x_2, \dots, x_{25})$ с индексами 1, 4,

$$S = \sum_{i=1}^5 y_i$$

9, 16, 25 подряд в массив $Y=(y_1, y_2, \dots, y_5)$. Вычислить

7. Записать положительные элементы массива $X=(x_1, x_2, \dots, x_n)$ подряд в массив $Y=(y_1, y_2, \dots, y_k)$. Определить k – количество положительных элементов. Вычислить произведение элементов массива Y с четными индексами.

8. Записать элементы массива $X=(x_1, x_2, \dots, x_{16})$ в обратном порядке в массив $Y=(y_1, y_2, \dots, y_{16})$. Вычислить $S=y_1+y_4+y_9+y_{16}$.

9. Записать элементы массива $X=(x_1, x_2, \dots, x_{12})$ в массив $Y=(y_1, y_2, \dots, y_{12})$, сдвинув элементы массива X вправо на три позиции. При этом 3 элемента из конца массива X перемещаются в начало, т.е. $(y_1, y_2, \dots, y_{12})=(x_{10}, x_{11}, x_{12}, x_1, x_2, \dots, x_9)$. Вычислить произведение элементов массива Y с четными индексами.

10. Записать отрицательные элементы массива $X=(x_1, x_2, \dots, x_n)$ подряд в массив $Y=(y_1, y_2, \dots, y_k)$. Определить k – количество отрица-

$$P = \prod_{i=1}^k y_i$$

тельных элементов. Вычислить

2.8. Лабораторная работа «Создание собственных функций рабочего листа»

Цель работы: изучить возможности *Excel* для создания функций, определяемых пользователем.

Форма проведения: выполнение индивидуального задания.

Форма отчетности: выполнение теста, защита отчета.

Основные понятия

1. Описание собственных функций рабочего листа

Функция рабочего листа, определенная пользователем, — это процедура *Function*, которую можно указать в формуле, хранящейся в ячейке. Эти функций работают точно так же, как и другие функций, используемые на рабочем листе *Excel*. Создавая собственные функции, можно:

1) Разработать функций, которые решают требуемые задачи. Если требуется производить специальные расчеты, то можно написать собственную функцию.

2) Указать вместо вложенных и сложных формул одну функцию.

3) Создать функции, содержащие математические выражения, встроенные функции рабочего листа *Excel* и инструкции *VBA*.

Функция, определяемая пользователем, выполняет вычисления и возвращает значение. После создания она становится доступной для всех листов рабочей книги. Функция, определяемая пользователем, используется точно так же, как и любая встроенная функция *Excel*, например, *СУММ()* или *СРЗНАЧ()*.

Функции, определяемые пользователем, обычно используются, чтобы произвести расчеты. Они не могут изменять значения свойств или выполнять методы. С помощью функций, заданных пользователем, нельзя выполнить следующие операции:

1) Задать свойства объектов.

2) Вставить, удалить или отформатировать ячейки.

3) Задать или изменить содержимое ячейки.

4) Переместить, переименовать, удалить или добавить рабочие листы.

5) Создать, открыть, закрыть или удалить рабочую книгу.

2. Собственные функций рабочего листа

Функции, определяемые пользователем, разрабатываются точно так же, как и любая функция *VBA*. Они хранятся в модуле и описываются с помощью ключевого слова *Public*. Чтобы создать функцию достаточно выполнить команду **Сервис\Макрос\Макросы**. В предложенном окне ввести имя макроса (функции) и нажать кнопку **Создать**.

Внести изменения в тело функции. Приведем пример функции, которая вычисляет налог на прибыль:

```
Public Function SalesTax(SalesAmount)
SalesTax = SalesAmount * 0.35
End Function
```

Параметром функции является значение или ссылка на ячейку. Чтобы использовать эту функцию в ячейке, введите следующую формулу:

```
=SalesTax(50)
=SalesTax(B1)
```

На рис.2.10 показано использование функции, определяемой пользователем. Обратите внимание на выражение в строке формулы.

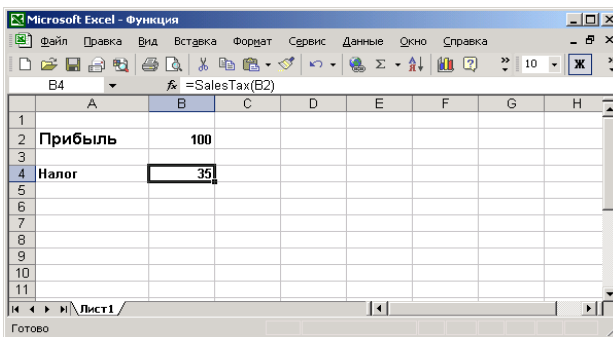


Рис. 2.10 Использование собственной функции рабочего листа

Задания

Создайте две функции пользователя в соответствии с формулами. Проверьте, что вновь созданные функции появились в общем списке функций, вызвав, например, **Мастер функций** в MS Excel. Используя функции пользователя протабулируйте каждую функцию в MS Excel.

Варианты функций определены в лабораторной работе «Основы программирования VBA».

$$Y = 3 \cos(\pi x) \frac{\sin(\pi x)}{1 + \sin(\pi x)} + \frac{\cos(\pi x)}{x}$$

$$g = \begin{cases} x^2 - |x - 2|, & x < 1, \\ 5x^2 + x - 6, & x \geq 1. \end{cases}$$

1. Для функций из задания №1 разработайте пользовательскую форму, в которой в качестве элемента управления для выбора варианта решения применяются переключатели. Добавьте в форму кнопку Отмена для завершения работы с пользовательской формой.

2.9. Лабораторная работа «Работа с внешними файлами»

Цель работы: изучить возможности VBA по сохранению больших массивов данных на внешнем носителе.

Форма проведения: выполнение индивидуального задания.

Форма отчетности: выполнение теста, защита отчета.

Основные понятия

При работе с большим количеством данных часто удобно записывать данные в файл или считывать из файла. Работа с файлами определяется следующими функциями языка программирования: создание файла, запись данных в файл, чтение данных из файла, закрытие файла. Кроме этого, существует дополнительная группа процедур обработки файловой системы, которые рассмотрим ниже. Операторы этой группы позволяют перемещать текущий каталог (папку) по дискам системы, создавать новые и удалять каталоги (папки); изменять атрибуты файлов, копировать и удалять файлы. Их набор, по существу, совпадает с набором соответствующих команд операционной системы

Создание файла: оператор *Open*

Инструкция *Open* позволяет создать файл и получить к нему доступ. Данная инструкция обеспечивает три типа доступа к файлам:

1. Последовательный доступ (определяется тремя режимами доступа):

- *Input* (для ввода данных из файла, то есть для чтения),
- *Output* (для вывода данных, то есть для записи),
- *Append* (добавление данных в существующий файл).

2. Произвольный доступ (режим *Random*). Данный режим используется, если необходимо считать и записать данные в файл без его закрытия.

3. Двоичный доступ (режим *Binary*). Двоичный режим позволяет организовать доступ, как по чтению, так и по записи в любую позицию файла.

Тип доступа к файлам определяет список допустимых инструкций чтения и записи данных (табл. 2.3).

Таблица 2.3 Инструкции чтения/записи данных в файл

Тип доступа	Запись данных	Чтение данных
Последовательный	Print #, Write #	Input #
Произвольный	Put	Get
Двоичный	Put	Get

Синтаксис инструкции *Open*:

Open путь *For* режим *As* [#]НомерФайла

Здесь:

путь — строковое выражение, указывающее имя файла, может содержать имя папки и диска.

режим — ключевое слово, указывающее режим работы с файлом.

НомерФайла — число, определяющее порядковый номер файла в программе. Допустимый номер файла может определяться из диапазона от 1 до 511 включительно. Для определения следующего свободного номера файла следует использовать функцию *FreeFile*.

Приведем пример описания файла, открываемого для последовательного чтения:

Open "TestFile" *For* *Input* *As* #1

Завершение операций ввода/вывода: оператор *Close*

Close [[#]НомерФайла],[#]НомерФайла...]

Необязательный аргумент *списокНомеровФайлов* может содержать один или несколько номеров файлов.

Если список пуст, то закрываются все активные файлы, открытые с помощью инструкции *Open*.

При закрытии файла, открытого в режиме *Output* или *Append*, в него добавляется содержимое последнего буфера вывода. Инструкция

Close разрывает связь между файлом на диске и соответствующим ему номером в программе.

Запись отформатированных данных в файл с последовательным доступом: оператор Print #

Print #НомерФайла, [СписокВывода]

НомерФайла — обязательный параметр.

СписокВывода — необязательный параметр, который может содержать следующие элементы:

- *Sps(n)* — используется для вставки пробелов в файл. Здесь *n* — число пробелов, которые следует вставить.
- *Tab (n)* — устанавливает курсор в столбец с указанным номером. *Tab* без аргумента устанавливает курсор в начало следующей зоны печати.
- *выражение* — числовые или строковые выражения, которые следует вывести в файл.
- *позиция* — указывает позицию, в которой следует печатать следующий символ. Для установки курсора сразу после последнего выведенного символа используется точка с запятой.

Рассмотрим на примере действие данной инструкции.

```
Private Sub MyPrint()
  Open "TESTFILE" For Output As #1 'Открываем файл
  Print #1, "Пример"; " форматирования вывода" 'Сплошная печать
  For i = 1 To 5
    Print #1, i * 10; ' Числа выводятся через пробел
  Next i
  Print #1, ' Пропустить строку в файле
  Print #1, "Зона 1", "Зона 2" ' Вторая зона печатается с 15 позиции
  Print #1, "Зона 1"; Tab; "Зона 2" Вторая зона печатается с 15 позиции
  Print #1,
  Print #1, Tab(15); "Вывод с заданной позиции"
  Print #1, Tab(15); "Очередной вывод"
  Print #1,
  Print #1, Spc(5); "Печатаем 5 пробелов"
  Print #1, "Между"; Spc(3); "словами"; Spc(3); "no"; Spc(3); _
    "три"; Spc(3); "пробела"
```


Close #1
End Sub

Запись неформатированных данных в файл с последовательным доступом: оператор *Write #*

Write #НомерФайла, [СписокВывода]

НомерФайла — обязательный параметр.

СписокВывода — необязательный параметр, который может содержать один или несколько разделяемых запятыми числовых или строковых выражений.

Чтение данных из открытого последовательного файла: оператор *Input #*

Input #НомерФайла, СписокПеременных

НомерФайла — обязательный параметр.

СписокПеременных — разделяемый запятыми список переменных, которым следует присвоить значения, считываемые из файла. Элементы данных должны располагаться в файле в том же порядке, что и переменные в *СпискеПеременных*. Данные в файл должны быть записаны с помощью оператора *Write*.

При достижении конца файла во время считывания элемента данных ввод прекращается и возникает ошибка. В этом случае в программе использую функцию *Eof*, которая позволяет определить признак конца файла и обработать его.

```
Private Sub MyInput()
  Dim MyString, MyNumber
  Open "TESTFILE1" For Input As #1
  Do While Not EOF(1)
    Input #1, MyString, MyNumber
    Debug.Print MyString, MyNumber '
  Loop
  Close #1
End Sub
```

Если требуется осуществить построчное чтение данных из открытого последовательного файла, то используется оператор *Line Input #*. Его синтаксис следующий:

Line Input #НомерФайла, ИмяПеременной

НомерФайла — обязательный параметр.

ИмяПеременной — обязательный параметр. Допустимое имя переменной типа *Variant* или *String*. Данные в файл предварительно записываются с помощью оператора *Print*.

Чтение данных из открытого файла: оператор *Get*

Get [#]НомерФайла, [НомерЗаписи], ИмяПеременной

НомерФайла — обязательный параметр.

НомерЗаписи — необязательный параметр (для файлов в режиме *Random*).

ИмяПеременной — обязательный параметр. Допустимое имя переменной, в которую следует помещать считываемые данные.

Первой записи в файле соответствует номер 1, второй – 2 и т.д. Если параметр *НомерЗаписи* опущен, читается запись, на которую был установлен указатель после выполнения последнего оператора *Get* или *Put*. Перевести указатель на нужную запись можно с помощью оператора *Seek*. Наличие запятых разделителей является обязательным, например:

Get #4,FileBuff

Рассмотрим пример чтения данных из файла, который содержит не менее пяти элементов типа запись.

Type Record

ID As Integer

*Name As String*20*

End Type

Private Sub RndFile()

Dim MyRecord As Record, Position

Open "TestFile3" For Random As #1 Len = Len(MyRecord)

Position = 3

Get #1, Position, MyRecord ' Читает третью запись в файле

Close #1

End Sub

Запись содержимого переменной в файл: оператор *Put*

Put [#]НомерФайла, [НомерЗаписи], ИмяПеременной

НомерФайла — обязательный параметр.

НомерЗаписи — необязательный параметр (для файлов в режиме *Random* – номер записи, для файлов *Binary* – номер байта, с которого следует читать данные).

ИмяПеременной — обязательный параметр. Допустимое имя переменной, содержащей данные, которые следует записать в файл.

Первой записи в файле соответствует номер 1, второй – 2 и т.д. Если параметр *НомерЗаписи* опущен, записывается запись, на которую был установлен указатель после выполнения последнего оператора *Get* или *Put*. Перевести указатель на нужную запись можно с помощью оператора *Seek*. Наличие запятых разделителей является обязательным, например:

Put #4,,FileBuff

Приведем пример записи в файл.

Type Record

ID As Integer

*Name As String*20*

End Type

Private Sub WriteRndFile()

Dim MyRecord As Record, RecordNumber

Open "TestFile3" For Random As #1 Len = Len(MyRecord)

For RecordNumber = 1 To 5

MyRecord.ID = RecordNumber

MyRecord.Name = "My Name" & RecordNumber

Put #1, RecordNumber, MyRecord

Next RecordNumber

Close #1

End Sub

Задания

Вариант 1

1. Написать процедуру, которая в одномерном массиве определяет минимальный и максимальный элементы и находит их среднее арифметическое. Размерность массива вводите с помощью оператора *InputBox*. Значения элементов массива определить датчиком случайных чисел. Для вывода скалярных результатов вычислений используйте оператор вывода *MsgBox*. Вывод массива выполнить в файл.

2. Известна информация о багаже пассажира – количество вещей и общий вес багажа. Ввести данные с клавиатуры и записать в файл типа Random.

Вариант 2

1. Написать процедуру вычисления $Z_i = (x_i - 50) - y_i$, где x изменяется от 50 до 100, а y является элементом массива y_1, y_2, \dots, y_{11} . Значения элементов массива вводите с помощью оператора *InputBox*. Вывод массива выполнить в файл и на экран с помощью оператора вывода *MsgBox*.

2. Дан список, состоящий из названия книг, фамилии авторов, названия издания и года издания. Ввести данные с клавиатуры и записать в файл типа Random.

Вариант 3

1. Написать процедуру поиска максимального элемента в массиве из N элементов, используя «кубковую систему»: на первом шаге алгоритма из каждой пары рядом стоящих элементов выбирается максимальный — он проходит в следующий тур. Элемент, которому не находится пара, переходит в следующий тур безусловно. На следующем шаге алгоритм повторяется и так до тех пор, пока в массиве не останется один элемент — он и будет максимальным. Размерность массива вводите с помощью оператора *InputBox*. Значения элементов массива определить датчиком случайных чисел. Для вывода скалярных результатов вычислений используйте оператор вывода *MsgBox*. Вывод массива выполнить в файл.

2. Дан список студентов группы и оценки экзаменационной сессии. Ввести данные с клавиатуры и записать в файл типа Random.

Вариант 4

1. Дан массив x_1, x_2, \dots, x_N . Удалить элемент массива, больший заданного числа. Если таких элементов нет, выдать сообщение: «Элементы для удаления не найдены». Если таких элементов несколько, то удалить последний из найденных. Размерность массива и заданное число вводите с помощью оператора *InputBox*. Значения элементов массива определить датчиком случайных чисел. Для вывода скалярных результатов вычислений используйте оператор вывода *MsgBox*. Вывод массива выполнить в файл.

2. Известна информация о сотрудниках фирмы – фамилия, имя, отчество, адрес и дата поступления на работу. Ввести данные с клавиатуры и записать в файл типа Random.

Вариант 5

1. Дан массив $A(N)$. В массиве найти сумму элементов после первого отрицательного и сумму элементов до него. Размерность массива вводите с помощью оператора *InputBox*. Значения элементов массива определить датчиком случайных чисел. Для вывода скалярных результатов вычислений используйте оператор вывода *MsgBox*. Вывод массива выполнить в файл.

2. Имеется K заявлений на получение жилплощади, в каждом указан срок подачи заявлений, площадь занимаемой квартиры и количество членов семьи. Ввести данные с клавиатуры и записать в файл типа Random.

Вариант 6

1. Даны три массива $A(N)$, $B(M)$, $C(L)$. Найти максимальный элемент в каждом массиве и максимальный среди них. Размерность массивов вводите с помощью оператора *InputBox*. Значения элементов массивов определить датчиком случайных чисел. Для вывода скалярных результатов вычислений используйте оператор вывода *MsgBox*. Вывод массивов выполнить в файл.

2. Дан список сотрудников лаборатории, должность и возраст каждого сотрудника. Ввести данные с клавиатуры и записать в файл типа Random.

Вариант 7

1. Дан массив a_1, a_2, \dots, a_N . Расположить положительные элементы массива, стоящие на нечетных местах, по возрастанию, остальные оставить на своих местах. Размерность массива вводите с помощью оператора *InputBox*. Значения элементов массива определить датчиком случайных чисел. Вывод массива выполнить в файл и на экран с помощью оператора вывода *MsgBox*.

2. Дан список группы с оценками экзаменационной сессии. Ввести данные с клавиатуры и записать в файл типа Random.

Вариант 8

1. Дан массив $A(N)$. Найти максимальный элемент массива и нормировать элементы массива по максимальному. Размерность массива вводите с помощью оператора *InputBox*. Значения элементов массива определить датчиком случайных чисел. Для вывода скалярных результатов вычислений используйте оператор вывода *MsgBox*. Вывод массива выполнить в файл.

2. Дан список абитуриентов, средний балл аттестата и оценки на вступительных экзаменах у каждого. Ввести данные с клавиатуры и записать в файл типа Random.

Вариант 9

1. Дан массив a_1, a_2, \dots, a_N . Расположить ненулевые элементы массива по убыванию; остальные элементы оставить на своих местах. Размерность массива вводите с помощью оператора *InputBox*. Значения элементов массива определить датчиком случайных чисел. Вывод массива выполнить в файл и на экран с помощью оператора вывода *MsgBox*.

2. Дан список группы и оценки экзаменационной сессии с названием предметов. Ввести данные с клавиатуры и записать в файл типа *Random*.

Вариант 10

1. Подсчитать число точек, находящихся внутри круга радиусом R , с центром в точке с координатами $(1, 1)$. Координаты заданы массивами $X(N)$, $Y(N)$. Размерность массивов вводите с помощью оператора *InputBox*. Значения элементов массивов определить датчиком случайных чисел. Для вывода скалярных результатов вычислений используйте оператор вывода *MsgBox*. Вывод массивов выполнить в файл.

2. Сведения о лекарствах содержатся в специальной ведомости: наименование, количество, цена и срок хранения. Ввести данные с клавиатуры и записать в файл типа *Random*.

3 МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ОРГАНИЗАЦИИ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

3.1 Общие положения

Самостоятельная работа предусмотрена учебным планом. Цель самостоятельной работы студента в рамках курса «Информатика для менеджеров» — закрепление и расширение знаний, полученных во время проведения аудиторных занятий.

Самостоятельная работа студента по дисциплине «Информатика для менеджеров» включает следующие виды деятельности:

- 1) проработка лекционного материала;
- 2) подготовка к лабораторным работам;
- 3) самостоятельное изучение тем (вопросов) теоретической части курса.

В ходе самостоятельной работы студент, ориентируясь на изложенные рекомендации, планирует свое время и перечень необходимых работ в зависимости от индивидуальных психофизических особенностей. Формат самостоятельной работы студентов может отличаться в зависимости от формы обучения и объема аудиторной работы.

3.2 Проработка лекционного материала и подготовка к лабораторным работам

Для качественного усвоения учебного материала целесообразно осуществлять проработку лекционного материала, которая направлена как на систематизацию имеющегося материала, так и на подготовку к освоению практических аспектов, связанных с содержанием дисциплины.

Проработка лекционного материала включает деятельность, связанную с изучением рекомендуемых преподавателем источников, в которых отражены основные моменты, затрагиваемые в ходе лекций. Кроме того, важное место отведено работе с собственноручно составленным конспектом лекций. При конспектировании во время лекции помните, что не следует записывать все, что говорит и/или демонстрирует лектор: старайтесь выявить главное и записать только это. Цель конспекта – формирование целостного логически выстроенного взгляда на круг во-

просов, затрагиваемых в ходе изучения соответствующей темы, а не механическая фиксация текстовой и графической информации.

Во внеаудиторное время проработка лекционного материала может быть выстроена в двух основных форматах:

а) отработка прослушанной лекции (прочтение конспекта и рекомендованных преподавателем источников с сопоставлением записей) и восполнение пробелов, если они имелись (например, если студент не понял чего-то, не успел записать);

б) прочтение перед каждой последующей лекцией предыдущей, дабы не тратилось много времени на восстановление контекста изучения дисциплины при продолжающейся или связанной теме.

В ходе проработки лекционного материала обращайтесь внимание на контрольные вопросы, которые, как правило, имеются в конце каждой темы учебника (учебного пособия). Отвечая на них, можно сделать вывод о степени понимания материала. Если ответы на какие-то вопросы вызвали затруднения, то следует предпринять еще одну попытку изучения отдельных вопросов.

При подготовке к лабораторным занятиям необходимо заранее изучить методические рекомендации по его проведению, обратить внимание на цель, формат и содержание занятия. Если какие-то моменты вызвали дополнительные вопросы, целесообразно обратиться к содержанию лекционного материала, рекомендациям преподавателя по изучению теоретической части курса (рекомендуемым источникам) или за личной консультацией. В ходе подготовки к лабораторным работам может потребоваться обращение к различным источникам. Проявляйте инициативу и самостоятельность в данном вопросе. При этом следует пользоваться только авторитетными изданиями, как печатными, так и электронными.

3.3 Самостоятельное изучение тем теоретической части курса

В ходе изучения дисциплины некоторые из тем курса выносятся исключительно на самостоятельное изучение. Следует обратить внимание на то, что работа по этим темам включает как подбор источников, так и изучение их содержания.

В зависимости от особенностей усвоения учебного материала студентами и объема аудиторной работы некоторые из вопросов, рассматриваемые в ходе проведения лекций и лабораторных работ, могут быть также вынесены в формат самостоятельного изучения.

Студент самостоятельно изучает дополнительные вопросы, связанные с построением и анализом моделей множественной регрессии и эконометрических моделей по временным рядам. Для достижения этой цели сформулированы следующие задания:

- Создание и использование классов.
- Создание и использование документов ActiveX.

Тема «Создание и использование классов»

Необходимо дать определение класса. Изучить модель классов Excel. Рассмотреть практические вопросы, связанные с созданием собственных классов. Описать процесс создания классов в VBA (Excel).

Тема «Создание и использование документов ActiveX»

Предложенная тема рассматривается в контексте использования панель инструментов "Элементы управления". Элементы, расположенные на панели инструментов Control Toolbox (Элементы управления), называются элементами Active X. Они несколько отличаются от элементов управления, расположенных на панели Forms (Формы). Скорее они ближе к элементам управления Visual Basic, т.к. при добавлении объекта Active X на рабочий лист создается макрос, который сохраняется вместе с этим элементом, а не только запускается при его выборе. При копировании или перемещении такого объекта автоматически будут скопированы или перемещены все процедуры, связанные с ним.

Необходимо рассмотреть несколько вариантов (от 3 и более) внедрения элементов управления на рабочий лист книги. Выполненная работа оформляется в виде реферата

4 РЕКОМЕНДУЕМЫЕ ИСТОЧНИКИ

1. Стивенс, Р. Visual Basic. Готовые алгоритмы [Электронный ресурс] / Р. Стивенс. — Электрон. дан. — Москва : ДМК Пресс, 2007. — 384 с. — Режим доступа: <https://e.lanbook.com/book/1221>. — Загл. с экрана.

2. Журавлев, А.Е. Информатика. Практикум в среде Microsoft Office 2016 [Электронный ресурс] : 2018-07-12 / А.Е. Журавлев. — Электрон. дан. — Санкт-Петербург : Лань, 2018. — 96 с. — Режим доступа: <https://e.lanbook.com/book/107927>. — Загл. с экрана.