ЛАБОРАТОРНАЯ РАБОТА № 5.
РАЗРАБОТКА МАКРОСОВ В ОФИСНОМ ПАКЕТЕ MS OFFICE 2007

Цель работы: научиться использовать возможности программирования в офисных приложениях MS Office.

Задачи работы:

- познакомиться со средствами MS Office 2007 для автоматической записи макросов;
- познакомиться с основными конструкциями языка Visual Basic for Applications (VBA);
- научиться разрабатывать макросы на языке VBA для приложений MS Word и MS Excel.

Лабораторная работа состоит из двух частей:

1. Создание макроса с помощью автоматической записи
2. Создание макроса вручную

Краткие теоретические сведения

Создание макросов

Макрос (макрокоманда) – последовательность команд, имеющая имя. С помощью макросов можно автоматизировать действия, часто выполняемые пользователем. Для написания макросов используется язык Visual Basic for Application (VBA), встроенный в пакет Microsoft Office.

Редактор Visual Basic доступен во всех приложениях Microsoft Office.

Макрос может быть создан в автоматическом режиме и вручную в редакторе Visual Basic. Для создания макроса в автоматическом режиме следует выполнить пункт **Запись макроса** группы **Макросы** вкладки **Вид.** Уже существующий макрос можно отредактировать или удалить, используя диалоговое окно **Макрос,** открывающееся при нажатии кнопки **Макросы** вкладки **Вид**.

Больше возможностей доступно при использовании вкладки **Разработчик**. Если она отсутствует, ее можно подключить в окне **Параметры Word – Настройки ленты**, отметив пункт **Разработчик** в правой панели настройки ленты (Рис. 5.1).

Для создания макроса вручную (запуск редактора Visual Basic for Application) достаточно использовать пункт **Создать** в диалоговом окне **Макрос**. Также редактор можно вызвать нажатием кнопки **Visual Basic** вкладки **Разработчик.**

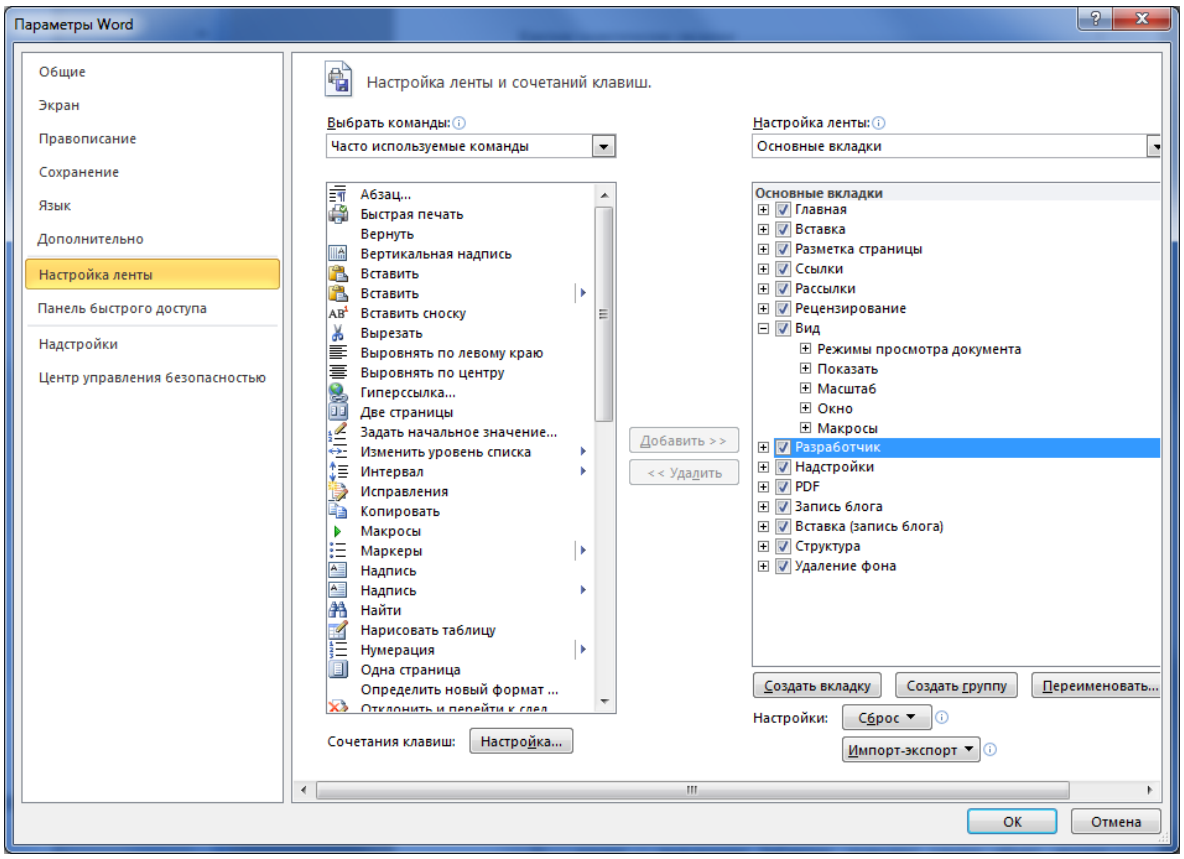

Рис. 5.1. Подключение вкладки **Разработчик**

Макросы сохраняются вместе с документом либо в шаблон Word в зависимости от области действия, указанной при создании макроса..

Структура проекта

Проект – это основная программа, которая объединяет и инициализирует все подключаемые модули, библиотеки и т.п. Обычно содержит описания объектов и интерфейса, вызовы процедур из подключенных модулей. В состав проекта входят следующие компоненты:

1) модули – содержат процедуры макросов ;
2) модули класса – содержат описания пользовательских классов;
3) формы – содержат описания форм и процедур обработки событий; позволяют организовать диалог с пользователем;
4) объекты Microsoft Word – делают доступными другие документы;
5) ссылки – подключение библиотек; позволяют сделать объект другого приложения доступным в программе.

Наиболее часто используемые компоненты – это модули и формы.

В VBA существует 2 типа проектов :

1) Normal – проект сохранен вместе с шаблоном Normal.dot; содержащиеся в нем макросы доступны для любого документа Word;
2) Project – хранит только макросы данного документа.

При написании макроса с использованием форм следует учитывать, какой тип проекта выбран. Если создается проект типа Normal, то и форма должна быть создана в проекте Normal, иначе объекты формы будут недоступны в данном проекте. Содержащиеся модули и макросы в проекте наглядно отображаются в окне проекта, находящемся слева вверху в редакторе VBA.

Основные конструкции языка Basic

Язык Visual Basic построен на синтаксисе языка Basic и поддерживает все его конструкции.

Компилятор языка нечувствителен к регистру клавиатуры. Каждый оператор записывается с новой строки. При необходимости записать два оператора на одной строке их разделяют двоеточием. Строки комментариев обозначаются командой REM или одинарной кавычкой.

Описание и типы переменных

Допускается неявное описание переменных, т.е. задание типа переменной по присвоенному ей значению. Например, оператор присваивания

X=4.5

при неявном описании указывает компилятору, что тип переменной X – вещественный.

При явном описании можно использовать следующие типы переменных: **Integer, Byte, Long** – целые, **Boolean** – логический, **Single, Double** – вещественные, **String** – строковый.

Для описания переменной используется оператор **DIM:**

DIM X as **Single**
DIM A(20) as **Integer** ‘ Описание массива

Базовые конструкции Visual Basic

Условный оператор:

IF логическое условие **Then**
Операторы
[**Else** ‘ в квадратных скобках - необязательная часть
Операторы]
End IF

Оператор выбора:

Select Case переменная- переключатель
Case Значение1
Операторы 1
Case Значение2
Операторы 2
[**Else**
Операторы]
End Select

Операторы циклов:

FOR переменная цикла= начальное значение **TO** конечное значение [**STEP** шаг]
Операторы
NEXT переменная цикла

While логическое условие
Операторы
Wend

Строчный комментарий:

' комментарий
REM комментарий

Типичные конструкции VBA

Открытие, закрытие и создание документов

Documents.Add - создание нового документа;
Documents.Open "D:\Users\myfile.doc" - открытие документа;
ActiveDocument.Save - сохранение активного документа;
Documents ("myfile.doc")**.Save** - сохранение заданного документа;
ActiveDocument.Close - закрытие активного документа;
Documents ("myfile.doc")**.Close** - закрытие заданного документа;
Documents.Save - сохранение всех открытых документов.

Добавление текста в документ

Selection.TypeText "My text" - добавление текста;
Selection.TypeParagraph - добавление знака абзаца.

Макрос запроса на печать документа

```
Public Sub OKToPrint()
    i = MsgBox("Хотите распечатать текущий документ?", vbYesNo,)
    If  i = 6 Then ActiveDocument.PrintOut
End Sub
```

Перемещение по документу

Selection.HomeKey unit:=wdStory, **Extend**:=wdMove - перемещение к началу документа;
Selection.EndKey unit:=wdStory, **Extend**:=wdMove - перемещение к концу документа;
Selection.MoveLeft unit:=wdCharacter, **Count**:=l, Extend:=wdMove - перемещение на один символ влево;
Selection.MoveRight unit:=wdWord, **Count**:=l, **Extend** :=wdMove перемещение на одно слово вправо;
Selection.Up unit:=wdParagraph, **Count**:=l, **Extend**:=wdMove - перемещение на один абзац вверх;
ActiveDocument.Bookmarks ("test") - выбор закладки "test"

Замена записи

```
With ActiveDocument.Content.Find
    .ClearFormatting
    .Replacement.ClearF ormatting
    .Text = "старая строка"
    .ReplacementText = "новая строка"
    .Execute Replace := wdReplaceAll
End With
```

Подсчет числа записей

```
    intCount = O
    Selection.HomeKey unit:=wdStory
    With Selection.Find
        .ClearFormatting
        .Text = "string"
        .Execute
        While .Found
                intCount = intCount + 1
                .Execute
        Wend
```

End With
MsgBox “Found” & Str$(intCount) & “occures”

Спецсимволы MS Word

Visual Basic:

- код_ASCII = Asc(символ)
- символ = Chr(код_ASCII)
- Код ASCII = от 0 до 255

Коды специальных символов, используемых в документах

- 1 - объект
- 2 - сноска
- 9 - табуляция (vbTab)
- 10 - разрыв строки (vbLf)
- 12 - разрыв страницы или раздела
- 13 - конец абзаца (vbCr)
- 13 + 7 - последний символ (Character) ячейки или строки таблицы
- 14 - разрыв колонки
- 19, 21 - поле
- 30 - неразрывный дефис
- 31 - мягкий перенос
- 160 - неразрывный пробел

Символы (Character) означающие конец параграфа (Paragraph)

- Chr(13) - конец параграфа
- Chr(13) + Chr(7) - конец последнего параграфа ячейки таблицы
- Chr(12) - разрыв раздела (код как и у разрыва страницы)

Символы (Character) не нарушающие целостности параграфа (Paragraph)

- Chr(11) - разрыв строки
- Chr(12) - разрыв страницы (код как и у разрыва раздела)
- Chr(14) - разрыв колонки

Методические рекомендации

При написании программы рекомендуется создать макрос в автоматическом режиме (см. методические указания к лаб. р. №4). Затем просмотреть сгенерированный текст в редакторе Visual Basic для определения объектов, методов и свойств, которые следует использовать при написании программы. При необходимости дополнительную информацию можно получить из справки в разделе Справка по Visual Basic.

Внимание! При использовании коллекции слов Words в состав слова включаются пробелы после него до следующего слова или знака препинания.

Прервать работу любого макроса MS Word можно парой клавиш Ctrl + Break.

Рекомендуется использовать следующие объекты:

ОБЪЕКТ	СВОЙСТВО или МЕТОД	ПОЯСНЕНИЕ
ActiveDocument	PageSetup Words	Параметры страницы активного документа Коллекция слов активного документа
Selection	Start End Startof Endof	выделенный текст; если текст не выделен, то текущая позиция курсора.
Range	Start End Startof Endof	любой заданный диапазон текста, который необходимо обработать
Bookmark	Start End Name Delete	закладка
Userform	Show Hide	форма – диалоговое окно для ввода-вывода данных

Примеры программ на языке VBA

Пример 1.
Весь документ представить в виде одного абзаца путем удаления символов признака конца абзаца ¶

Текст программы

```
Set r = ActiveDocument.Range(Start:=0, End:=0)
r.WholeStory
Selection.StartOf unit:=wdStory
k = 1
i = 0
While k < r.End – i
     Selection.EndKey unit:=wdLine
     Selection.Delete unit:=wdCharacter, Count:=1
     k = Selection.start
     i = i + 1
Wend
```

Текст программы с комментариями

```
    REM создание объектной переменной r типа диапазон,
    REM начальная и конечная позиции которого равны 0,
    REM т.е. диапазон пустой
Set r = ActiveDocument.Range(Start:=0, End:=0)
    REM задание значения диапазону r –весь документ
r.WholeStory
    REM выделения текста нет, поэтому объект
    REM Selection представляет из себя текущую позицию курсора,
    REM начало и конец этого объекта совпадают;
```

```
    REM установить начальную позицию – начало документа
    REM (единица измерения перемещений – весь документ)
Selection.StartOf unit:=wdStory
k = 1
i = 0
    REM цикл выполняется до тех пор, пока k меньше
    REM разности номера последнего символа в документе
    REM (позиция конца диапазона r) и i
While k < r.End – i
    REM переместить текущую позицию курсора в конец строки
Selection.EndKey unit:=wdLine
    REM удалить один символ в текущей позиции курсора
Selection.Delete unit:=wdCharacter, Count:=1
    REM переменной k присвоить значение начала выделения,
    REM т.е. в данном случае текущей позиции курсора
k = Selection.start
i = i + 1
Wend
```

Пример 2.

В выделенном фрагменте текста после фамилии «Петров» добавить его инициалы.

```
Set col = Selection.Words
For Each a In col
     If a = "Петров " Then a.InsertAfter "В.В. "
Next
```

Пример 3.

Определить с помощью макроса количество страниц в документе.
Для этого необходимо использовать в макросе команду

```
p = ActiveDocument.ComputeStatistics(wdStatisticPages)
```

Пример 4.

Требуется с помощью макроса определить текущий номер страницы, на которой найдена заданная строка при выполнении ее поиска по документу.

Для этого используются функции

```
Selection.Information(wdActiveEndPageNumber),
Selection.Information(wdActiveEndAdjustedPageNumber).
```

Первая возвращает номер страницы по счету, на которой располагается конец выделения, вторая возвращает присвоенный пользователем номер страницы, на которой располагается выделенный фрагмент.

При работе подпрограммы поиска каждое вхождение выделяется. Ниже приводится фрагмент кода макроса, который отображает в диалоговом окне номера страниц, на которых найден текст "text".

```
Selection.HomeKey Unit:=wdStory
Dim a As String
a =""
Selection.Find.ClearFormatting
Selection.Find.Replacement.CIearFormatting
With Selection.Find
   .Text = "text"
   .Forward = True
   .Wrap = wdFindStop
```

```
End With
While Selection.Find.Execute = True
   a = a + "" +'
   Str(Selection.Information(wdActiveEndPageNumber))
Wend MsgBox a
```

Пример 5.

Требуется из макроса вызвать стандартное диалоговое окно открытия и сохранения файла. Окно открытия файла можно вызвать командой

```
Dialogs(wdDialogFileOpen).Show,
```

а окно сохранения - командой

```
Dialogs(wdDialogFileSaveAs).Show
```

В обоих случаях будут вызваны соответствующие окна и выполнены соответствующие действия - открытие файла, его сохранение. Данные команды возвращают параметры: -2 - если нажато "Закрыть", - 1 - если "ОК", 0 - если "Отмена": код

```
If Dialogs(wdDiaIogFi1eOpen).Show = 0 Then Exit Sub
```

осуществит выход из программы, если в окне выбора файла будет нажата кнопка отмены.

Данные команды можно использовать и не только по "прямому" назначению - для открытия или сохранения файлов. Так, если вместо метода Show использовать Display, то окно отображаться будет, а связанное с ним действие не выполнится. Это позволяет использовать такие окна для выбора файлов с целью какого-либо действия над ними. Например, код

```
With Dialogs('wdDialogFileOpen)
   .Display
   myfile = .Name
End With
```

даст возможность пользователю выбрать файл в окне и запишет его имя в переменную myfile.

Пример 6.

Требуется в макросе получить номера строк и столбцов границ текущего выделения участка таблицы.

Объект **Selection** имеет свойство **Selection.Information**, которое содержит необходимые для решения задачи параметры:

```
Selection.Information(wdStartOfRangeColumnNumber)
```

- возвращает номер столбца начала выделения,

```
Selection.Information(wdStartOfRangeRowNumber)
```

- возвращает номер строки начала выделения ,

```
Selection.Information(wdEndOfRangeColumnNumber)
```

- возвращает номер столбца конца выделения,

```
Selection.Information(wdEndOfRangeRowNurnber)
```

- возвращает номер строки начала выделения.

Вне таблицы они возвращают "-1".

Пример 7.

Требуется из файлов *.doc, находящихся в определенном каталоге, извлечь созданное автоматически содержание.

Для этого необходимо в каждом документе выделить все элементы, входящие в коллекцию **TablesOfContents**, скопировать их в буфер обмена, а затем вставить как текст в новый документ.

Пусть документы, из которых надо извлечь содержание, находятся в папке "C:\Desktop\test", имеют расширение ".doc", а готовый результат надо вставить в уже открытый документ. Тогда макрос будет таким.

```
'начинаем поиск файлов .doc в указанной папке
'имена всех найденных файлов записываются
'в массив .FoundFiles (см, справку)
  With Application.FileSearch
     .NewSearch
     .Lookln = "C:\Desktop\test"
     .FileName = "*.doc"
     .SearchSubFolders = False
     .MatchTextExactly = True
     .FileType = msoFileTypeAllFiles
     .Execute
'теперь переберем все элементы этого массива
    For qqq = 1 To .FoundFiles.Count
'каждый откроем
       Documents.Open FileName:=.FoundFiles(qqq)
'выделим в нем содержание
       For Each a In ActiveDocumentTablesOfContents
          a.Range.Select
'скопируем его в буфер
          a.Range.Copy
       Next a
'закроем документ, при этом активируется предыдущий -
' тот, в который надо вставить содержание
       ActiveDocument.Close
'и вставим содержание "Специальной вставкой"
       Selection.PasteSpecial Link:=False,'
       DataType:=wdPasteText,
       Placement:=wdInLine, DispIayAsIcon:=False
'и так со всеми документами
    Next qqq
  End With
```

Пример 8.

В заданном файле требуется отметить текст после "=" в каждой строке стилем "tw4winlntemal",а остальной текст-стилем "Normal".

Следующий код, полученный в результате небольших исправлений записанного макроса, выполняет задачу:

```
'Выделить все:
  Selection. WholeStory
'Назначить стиль "Normal"
  Selection.Style = ActiveDocument.Styles(“Normal”)
'Снять выделение:
  Selection.MoveLeft Unit:=wdCharacter, Count:=l
'Установить параметры поиска - ищем знак =:
  Selection.Find.ClearFormatting
  With Selection.Find
       .Text = "="
       .Replacement.Text =""
       .Forward = True
       .Wrap = wdFindContinue
       .Format = False
       .MatchCase = False
       .Match WholeWord = False
```

```
            .MatchWildcards = False
            .MatchSoundsLike = False
            .Match AllWordForms = False
      End With
   'Теперь ищем все знаки = и, найдя каждый из них...
      While Selection.Find.Execute = True
    'выделяем текст от знака = до конца строки без символа конца
абзаца
        Selection.MoveRight Unit:=wdCharacter, Count:=l
        Selection.EndKey Unit:=wdLine, Extend :=wdExtend
        Selection.MoveLeft  Unit:=wdCharacter,  Count:=l,  Extend:
  =wdExtend
   'и применяем к нему стиль.
        Selection.Style = ActiveDocument.StyIes("tw4winIntemal")
        Selection.Coliapse
      Wend
```

После этого текст, отмеченный стилем "tw4winExternal", пропадает, поскольку текст в этом стиле помечается как "скрытый". Чтобы отобразить скрытый текст, следует включить отображение непечатаемых символов.

Чтобы сделать так, чтобы знак "=" тоже отмечался "tw4winExternal"?", необходимо внести следующие изменения:

```
      While Selection.Find.Execute = True
         Selection.MoveLeft Unit:=wdCharacter, Count:=l
         Selection.EndKey Unit:=wdLine, Extend:=wdExtend
         Selection.MoveLeft Unit:=wdCharacter, Count:=l,'
         Extend:=wdExtend
         Selection.Style = ActiveDocument.Styles("tw4winIntemal")
         Selection.Coliapse Direction:=wdCollapseEnd
      Wend
```

Пример 9.

Требуется определить путь к открытому dot-файлу.

Если к открытому в Word - то

`ActiveDocument.Path` – путь без имени,

`ActiveDocument.FullName` – - путь с именем.

Если к загруженному глобально, то

`Addlns("Имя шаблона.dot").Path.`

Пример 10.

Необходимо написать макрос, который прописывает в конец документа полный путь к файлу и его имя, а затем закрывает файл с сохранением.

Это действие выполняет следующий макрос:

```
      Sub cldoc()
           Selection.EndKey Unit:=wdStory Selection.TypeParagraph
           Selection.TypeText Text:=ActiveDocument.FulIName
           ActiveDocument.Save
           ActiveDocument.Close
      End Sub
```

Макрос можно поместить в Normal.dot через Редактор VBA, а затем назначить комбинацию клавиш или кнопку на панели инструментов.

Пример 11.
Следующий пример выполняет замену положительных значений на 1, а отрицательных – на -1 в диапазоне ячеек **B1:C3**.

```
Dim c As Range
For Each c In Range("B1:C3").Cells
     If c.Value > 0 Then
          c.Value = 1
     ElseIf c.Value < 0 Then
          c.Value = -1
     End If
Next
```

Другой макрос, выполняющий ту же задачу, использует свойство Cells, задающее относительное расположение ячеек к диапазону.

```
Dim i As Integer
Dim j As Integer
For i = 1 To Range("B1:C3").Rows.Count
     For j = 1 To Range("B1:C3").Columns.Count
          If Range("B1:C3").Cells(i, j).Value > 0 Then
               Range("B1:C3").Cells(i, j).Value = 1
          ElseIf Range("B1:C3").Cells(i, j).Value < 0 Then
               Range("B1:C3").Cells(i, j).Value = -1
          End If
     Next
Next
```

Пример 12.
Необходимо построить диаграмму на основе табличных данных, размещенных в диапазоне A1:C6.

```
Dim c As Chart
Set c = Charts.Add
c.ChartType = xlColumnClustered
c.SetSourceData Source:=Worksheets(1).Range("A1:C6"),
                PlotBy:=xlColumns
c.HasTitle = True
c.ChartTitle.Text = "Реальность и прогноз"
c.Name = "Продажи"
```

Порядок выполнения работы

В ходе работы необходимо сделать следующее:

1. Создать макрос в MS Word с помощью автоматической записи макросов согласно варианту;
2. Написать макрос на VBA в MS Word либо в MS Excel вручную согласно варианту.

Варианты заданий

№	Макрос, создаваемый автозаписью, в Word или Excel	Макрос в Word или Excel, выполняемый в среде VBA
1.	Переместить текущее слово в конец абзаца	Задать параметры шрифта (синий цвет, полужирный) для текущего абзаца, за исключением выделенного фрагмента текста.
2.	Сделать все внутренние границы текущей таблицы невидимыми линиями	Для выделенных ячеек строки вывести под ними все возможные перестановки значений ячеек (по одному набору в строку). Если выделено более одной строки – выдавать ошибку
3.	Применить параметры страницы: альбомная ориентация, верхнее и нижнее поля	Для двух выделенных ячеек одной строки (вторая ячейка – функция от первой) выполнить протяжение вдоль столбца с заданными пользователем шагом и конечным значением аргумента
4.	Поместить выделенный фрагмент текста в верхний колонтитул	Выполнить сортировку предложений в текущем абзаце по количеству слов
5.	Переместить текущий абзац в конец документа, отделив от предыдущего текста тремя пустыми строками	Для трех выделенных ячеек со значениями длин треугольника определить его тип: остроугольный, прямоугольный, тупоугольный.
6.	Настроить масштаб для отображения страницы целиком	Выполнить замену в выделенном фрагменте текста всех одно- и двухзначных чисел их текстовым начертанием (один, два…).
7.	Выделить курсивом слово, следующее за текущим словом, убрав курсив с остальных слов текущего абзаца	Найти в документе Word все символы латиницы и выделить их красным цветом, вывести в конце документа число символов латиницы.
8.	Переместить текущее слово на 1 слово вперед	Перед каждым предложением документа Word вывести его номер в абзаце.
9.	Вставить в позиции курсора новую пустую страницу альбомной ориентации	Найти в документе Word все цифры и выделить их синим цветом, вывести в конце документа число цифр.
10.	Назначить для выделенных ячеек Excel числовой формат с двумя знаками после запятой	Умножить значения выделенных ячеек Excel на число, заданное пользователем
11.	Выполнить масштабирование для отображения полной страницы	Создать в Excel таблицу умножения в пятеричной системе счисления
12.	Переместить текущее слово в конец абзаца	По введенным пользователем коэффициентам квадратного уравнения (в соответствующие ячейки) вычислить и вывести корни уравнения в соответствующие ячейки с анализом количества корней
13.	Установить ширину текущей ячейки по размеру текста	Преобразовать предложения в текущем абзаце в нумерованный список, каждое предложение – отдельным пунктом списка
14.	Сделать все внутренние границы	Выделить цветом в тексте все слова,

	текущей таблицы невидимыми линиями	длина которых превышает 8 символов.
15.	Назначить для выделенных ячеек Excel экспоненциальный формат	Выделить цветом в документе все предложения, состоящие более чем из 12 слов.