Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)
Кафедра экономической математики, информатики и статистики (ЭМИС)

АВТОМАТИЗАЦИЯ РАБОТЫ В MS WORD С ПОМОЩЬЮ VISUAL BASIC FOR APPLICATION

Отчет по практической работе по дисциплине «Информационные технологии»

Студент группы 549
 Баулин С.К.
«__» _____ 2020 г.
Старший преподаватель
кафедры ЭМИС
____ _____ Афанасьева И. Г.
«__» _____ 2020 г.

Томск 2020

Практическая работа №10

Автоматизация работы в Ms Word с помощью VBA

Цель работы: знакомство с основными объектами VBA Microsoft Word. На практике рассмотреть основные операции над объектами Application, Document, Selection, Range, Bookmark.

Теоретический материал

Объектная модель Microsoft Word, объекты Application, Document, Selection, Range, Bookmark.

На практике для решения большинства программных задач достаточно знать всего лишь пять объектов:

- объект Application;
- объект Document (с коллекцией Documents);
- объект Selection;
- объект Range;
- объект Bookmark (с коллекцией Bookmarks).

Ниже все эти самые важные объекты будут подробно рассмотрены. Поскольку наиболее часто встречающаяся задача программирования в Word – это создание документа (на основе шаблона) и запись в нужное место документа необходимой информации, то акцент будет сделан использовании соответствующих объектов для решения этой задачи.

Кроме того, для каждого объекта будут перечислены самые важные свойства, методы и события с кратким их описанием.

Объект Application, свойства, методы и события

ActiveDocument – возвращает объект активного документа в данном экземпляре Word. Используется очень активно, обычно без упоминания объекта Application, например:

ActiveDocument.Save

Это свойство доступно только для чтения, поэтому, чтобы сделать какой-нибудь документ активным, придется вызывать для его объекта метод Activate().

ActivePrinter – позволяет получить или настроить активный принтер в ходе работы программы. Также используется очень активно, если результаты работы вашего приложения необходимо печатать на определенном сетевом принтере.

AutomationSecurity – определяет уровень безопасности при программном открытии файлов. По умолчанию установлено значение msoAutomationSecurityLow, что значит – открывать со включенными макросами. Можно также использовать значения msoAutomationSecurityForceDisable – отключить макросы и msoAutomationSecurityByUI – то, что настроено на графическом интерфейсе.

BackgroundPrintingStatus – сколько заданий Word стоит в очереди на печать.

Caption – позволяет заменить слово Microsoft Word в заголовке окна на другой текст, например, "Мое приложение".

CheckLanguage – определяет ли Word в автоматическом режиме язык, на котором производится ввод текста. Если в системе установлено несколько языков ввода, то по умолчанию проверяет. При помощи этого свойства можно изменить режим работы Word.

CustomizationContext – свойство, которое позволяет указать шаблон или документ, на который будут распространяться внесенные вами изменения в меню, панели инструментов и клавиатурные комбинации. Например, код вида

CustomizationContext = NormalTemplate

говорит о том, что все изменения, которые вы будете вносить начиная с этого момента, будут сохраняться в шаблоне Normal.Dot (и, таким образом, будут применяться во всем документам).

Для диалоговых окон, которые предназначены для работы с файлами, в объекте Application предусмотрено отдельное свойство FileDialog, возвращающее одноименный объект.

DefaultSaveFormat – определяет формат сохранения файлов Word по умолчанию (тот, который будет предлагаться пользователь в диалоговых окнах Save As). Можно настроить на сохранение в формате обычного текста, текста Unicode, RTF и т.п.

DisplayAlerts – очень важное свойство. Оно позволяет подавить вывод ошибок и диалоговых окон при работе макросов и приложений VBA. Во многих ситуациях без него не обойтись. Особенно часто прибегать к этому свойству требуется, когда необходимо в ходе работы программы что-нибудь удалить или закрыть без сохранения.

DisplayAutoCompleteTips – включить/отключить подсказки для автозавершения текста. Чаще всего необходимо отключить. Остальные свойства Display… очевидны и поэтому здесь рассматриваться не будут.

FileDialog – возвращает объект FileDialog, то есть окно выбора файла, каталога, открытия файла или сохранения. Для открытия этого окна необходимо воспользоваться методом Show() этого объекта.

FileSearch – возвращает объект FileSearch, который может использоваться для поиска файлов по определенным параметрам.

IsObjectValid – очень удобное свойство для всевозможных проверок. Проверяет, существует ли еще объект, к которому хотим обратиться. Позволяет уберечься от ошибок, когда, к примеру, документ или объект в документе был удален пользователем.

KeyBindings – очень удобное во многих ситуациях свойство. Оно позволяет вернуть коллекцию KeyBindings – привязок клавиатурных комбинаций. Говоря проще, при помощи этого объекта и подобъектов вы можете назначить любую команду Word или любой макрос любому сочетанию клавиш (в том числе и сочетаниям, уже занятым служебными командами, например, <Alt>+<F4>).

MacroContainer – очень полезное свойство для программистов. Позволяет в ходе выполнения определить, откуда был запущен текущий программный код (обычно проверяются два варианта – normal.dot или обычный текущий документ).

NewDocument – одна из возможностей создать новый документ Word. Возвращает объект NewDocument. Для создания нового документа используется метод Application.NewDocument.Add().

NormalTemplate – Позволяет получить ссылку на объект Template, представляющий normal.dot – для внесения в него изменений.

Option – возвращает объект Option с огромным количеством свойств. Через этот объект программным способом можно настроить значения со всех вкладок, доступных на графическом экране через меню **Сервис - Параметры**.

Path – возвращает путь к программным файлам Word на диске.

PrintPreview – перейти в режим предпросмотра текущего документа или проверить, находимся ли мы в этом режиме. Очень удобно для показа документа пользователю или для реализации своей процедуры печати.

Selection – еще одно важнейшее свойство. Возвращает объект Selection – упрощенно говоря, место, в котором находится указатель вставки.

StatusBar – еще одно очень полезное свойство. Позволяет вывести текст в Status Bar – строке состояния, то есть строке в нижней части окна приложения, в которой выводится информация о страницах, столбцах, языке, режимах работы и т.п.

System – возвращает одноименный объект System, предназначенный для получения информации из операционной системы (региональный настройки, тип курсора мыши, разрешение экрана, тип процессора и т.п.). Позволяет также подключать сетевые диски и запускать приложение Microsoft System Information.

UserControl – очень важное свойство (оно есть и в Excel). Это свойство позволяет определить как именно был запущен Word – пользователем

вручную или программным образом. На основе этого можно, например, сделать вывод, нужно ли его программным образом закрывать.

UserInitials и UserName – возможность получить или определить информацию об инициалах или имени пользователя. Инициалы используются в исправлениях, а имя пользователя – в свойствах документа.

Version – свойство возвращает версию Word.

Visible – позволяет спрятать окно Microsoft Word.

Windows – возвращает информацию об одноименной коллекции Windows, представляющей объекты окон документов Word. Эта коллекция также используется очень часто.

WindowsState – позволяет свернуть/развернуть/восстановить окно Word.

Самые важные методы объекта Application:

Activate() – просто активизировать окно Word с текущим документом. Обычно нужно активизировать определенный документ, поэтому этот метод используется для объекта Document.

BuildKeyCode() – позволяет узнать уникальный номер для клавиатурной комбинации в Word. Пример применения этого метода приведен чуть выше при рассмотрении свойства Application.KeyBindings.

ChangeFileOpenDirectory() – этот метод позволяет изменить каталог, который по умолчанию открывает Word для работы с документами (по умолчанию, конечно, "Мои документы".

CleanString() – очень полезный метод. Позволяет "чистить" передаваемое символьное значение (полученное, например, от объектов Selection или Range) от специальных символов Word и превращать их в обычный текст – как будто он был набран в Блокноте.

GoBack() – этот метод обеспечивает переход на последнее место редактирования в документе. Word сохраняет с документом три последние точки редактирования, так что открыть последний документ в Word и перейти на точку, где вы остановились, можно очень просто:

RecentFiles(1).Open

Application.GoBack

Метод GoForward() обеспечивает переход вперед по точкам сохранения.

KeyBoard() – очень полезный метод. Позволяет программным способом переключать раскладку клавиатуры в Word, уберегая таким образом пользователей от ошибок. Переключение на русский выглядит как

Application.Keyboard 1049

а на английский

Application.Keyboard 1033

Если этому методу ничего не передавать, он вернет текущую раскладку клавиатуры.

OnTime() – очень интересный метод. Он позволяет выполнить макрос Word либо в указанное вами время, либо по прошествии какого-то времени. В Word одновременно может работать только один таймер.

OrganizerCopy() – еще один полезный метод. Позволяет скопировать макрос, панель инструментов, запись автотекста или стиль между документами. Для объекта Application предусмотрены также методы с самообъясняющими OrganizerDelete() и OrganizerRename().

PrintOut() – метод, который принимает огромное количество параметров и позволяет вывести на печать весь документ или его часть.

Quit() – метод, который используется, видимо, чаще всех. Позволяет закрыть Word с сохранением или без сохранения документов.

Repeat() – просто повторить последнюю выполненную команду указанное вами количество раз.

Run() –Позволяет запустить процедуру/макрос из открытого шаблона/документа и передать ей параметры.

ShowClipboard() – показать панель буфера обмена Word (если вы работаете с несколькими буферами).

У объекта Application множество событий – открытие/закрытие/сохранение/печать документа, щелчки мышью, активизация, выход из приложения и т.п.

Работа с объектом Selection

Объект Word.Selection, работа с выделенным участком текста, преимущества и недостатки

После того, как запущено приложение, найден и активизирован нужный нам файл, следующее действие, которое выполняется чаще всего – ввод или редактирование текста в нужном месте. Для этого используются объекты **Selection**, **Range** и **Bookmark**. Каждое из них используется в своих ситуациях и для своих задач.

Первый объект, который рассмотрим – это объект **Selection**.

Обычно перед тем, как что-либо сделать в окне документа Word, пользователь либо выделяет нужный участок текста, либо переставляет указатель вставки текста в нужное место. Объект Selection представляет именно такой выделенный участок текста (а если ничего не выделено, то место, где находится указатель вставки). Именно этот объект обычно использует макрорекордер.

Создавать объект Selection и получать на него ссылку в переменную не обязательно (а обычно и просто невозможно). Дело в том, что объект Selection в документе может быть только один. Он создается автоматически при запуске Word и всегда доступен. Обращаться к нему можно так:

```
Application.Selection.Text = "Вставляемый текст"
```

или просто

```
Selection.Text = "Вставляемый текст"
```

Обычно нужно правильно определить то место, на которое указывает объект Selection, чтобы выделить нужный нам участок текста или точку для ввода.

Как настроить выделение в документе Word

Самый простой способ – просто положиться на выделение нужного текста пользователем. Обычно такой способ применяется для сложного редактирования/форматирования участков текста и для ввода информации в указанное пользователем место документа, когда в автоматическом режиме нужное место не найти;

- воспользоваться методом Select(), который предусмотрен для огромного числа объектов (Document, Range, Bookmark, Table со всеми подобъектами типа столбцов и строк, PageNumber, Field и т.п.). Этот метод просто выделяет весь документ, закладку, таблицу и т.п.
- воспользоваться многочисленными методами объекта Selection, чтобы преобразовать уже существующее выделение;
- воспользоваться объектом Find для поиска нужного участка текста.

Если нужно вводить информацию в самое начало документа, можно вообще ничего не делать. По умолчанию указатель вставки устанавливается на начало документа.

Несмотря на то, что применение объекта Selection – самый простой и наглядный метод редактирования текста, и чаще всего именно он используется макрорекордером, на практике программисты используют его редко. Объясняется это очень просто: при использовании этого объекта существует зависимость от действий пользователя. Если во время выполнения нашего кода пользователь проявит инициативу и начнет щелкать по документу мышью, результат может быть совершенно непредсказуемым. Защититься от вмешательства пользователя можно двумя способами: работать со скрытым (то есть невидимым) документом или, возможно, со скрытым экземпляром Word. Для включения/отключения невидимости можно использовать свойство Visible для объектов Document и Application;
более удобный способ – вместо объекта Selection использовать объекты Range и Bookmark.

Объект Word.Bookmark, применение закладок в шаблоне, получение из объектов Bookmark объектов Selection и Range

Объект Bookmark – это просто закладка. На практике это – самый удобный способ навигации по документам, созданных при помощи шаблонов (например, отчетов). Принципиальное отличие его от объектов Selection и Range заключается в том, что все выделения и диапазоны теряются при закрытии документа (объекты Range вообще существуют только во время работы создавшей их процедуры, а закладки сохраняются вместе с документом. Если документ создан на основе шаблона, то все закладки, которые были определены в шаблоне, будут определены и в созданном на основе этого шаблона документе.

Создать закладку (меню **Вставка - Закладка**) намного проще, чем считать количество символов для объекта Range от начала документа/абзаца/предложения, или выполнять операции Move() (MoveDown(), MoveRight(), MoveNext()) для объекта Selection.

Функциональность объекта Bookmark невелика. Свойств и методов у этого объекта намного меньше, чем у объектов Selection и Range. Однако обычно никто и пытается использовать объект Bookmark для работы с текстом напрямую. Из объекта Bookmark очень просто получить объект Selection (при помощи метода Select()) или объект Range (при помощи свойства Range()) – и дальше можно пользоваться уже свойствами и методами этих объектов, например:

```
ThisDocument.Bookmarks("Bookmark1").Select
MsgBox Selection.Text
```

Создавать объекты Bookmark программным способом необязательно, но если есть необходимость, то можно использовать метод Add() коллекции Bookmark:

```
ThisDocument.Bookmarks.Add Name:="temp", Range:= Selection.Range
```

У этого метода – всего лишь два параметра, оба которых используются в примере.

Некоторые важные свойства объекта Bookmark

Empty – если это свойство возвращает True, то это значит, что закладка указывает на точку вставки, а не та текст;

Name – имя закладки. Очень удобно, что найти нужную закладку в коллекции закладок можно не только при помощи индекса (номера) закладки, но и по ее имени.

Range – возвращает объект Range на месте этой закладки.

Start, End, StoryType – аналогично таким же свойствам у объекта Selection.

Методов у объекта Bookmark всего три – Copy(), Delete() и Select(). Copy() – создает закладку на основе существующей, Delete() – удаляет ее, а Select() – выделяет то, на что ссылается закладка.

Объект Word.Range, программная работа с диапазоном в документе, свойства и методы объекта Range, преимущества по сравнению с объектом Selection

Как уже говорилось выше, чаще всего разработчиками для определения места ввода текста и навигации по документу используется объект Selection. Для этих же целей можно использовать и объект Range. Главное отличие между объектами Range и Selection заключается в том, что объект Selection может определить и пользователь (выделив текст мышью), а объект Range можно определить только программно, и он не зависит от текущего положения указателя или действий пользователя.

Формальное определение объекта Range выглядит так: это программный объект, который представляет непрерывный участок текста в документе. Этот объект не зависит от объекта Selection – вы можете работать с объектом Range, не изменяя текущего выделения. Он может не включать в себя ни одного символа (представлять курсор ввода текста).

Объектов Range в каждый момент времени может быть сколько угодно, а объектов Selection – только один.

Как создается объект Range

Первый способ – воспользоваться методом Range() объекта Document. В этом случае вам потребуется передать номера начального и конечного символа диапазона, а также document story, в который будут отсчитываться эти символы. Например, создать диапазон, который будет включать в себя первые 10 символов документа, можно так:

```
Dim rngDoc As Range
Set rngDoc = ActiveDocument.Range(Start:=0, End:=10)
```

Второй способ – воспользоваться свойством Range, которое предусмотрено для огромного количества объектов (Bookmark, Selection, Table-Row-Cell, Paragraph и т.п.). В этом случае при помощи этого свойства получаем объект Range, представляющий данный объект;

Третий способ – воспользоваться большим количеством вспомогательных свойств (Characters, Words, Sentences и т.п.), которые делят текст на отрезки – объекты Range. Эти свойства возвращают коллекции объектов Range.

Четвертый способ – переопределить существующий объект Range. Обычно для этой цели используется метод Range.SetRange();

Пятый способ, самый удобный в реальных приложениях. Он заключается в том, что вначале создается шаблон нужного вам документа (договора, приходного ордера, отчета и т.п.), в который при создании помещаете **закладки** в тем места, в которые потом потребуется произвести вставку данных. Затем программным способом для каждой закладки создается объект Range, и уже с его помощью производится ввод информации (данные о заказчике, сумма в кассовом ордере и т.п.)

Для целей отладки (чтобы убедиться, что объект Range действительно включает в себя тот участок текста, который вы планировали) можно создавать на основе объекта Range объект Selection (то есть выделять диапазон). Для этого у объекта Range предусмотрен метод Select().

Большая часть свойств и методов объекта Range совпадает с аналогичными свойствами и методами объекта Selection. Точно так же чаще всего для объекта Range используется одно-единственное свойство Text, которое позволяет ввести в место в документе, представленное этим объектом, нужный текст.

Задания на лабораторную работу

Необходимо разработать приложение следующего вида:

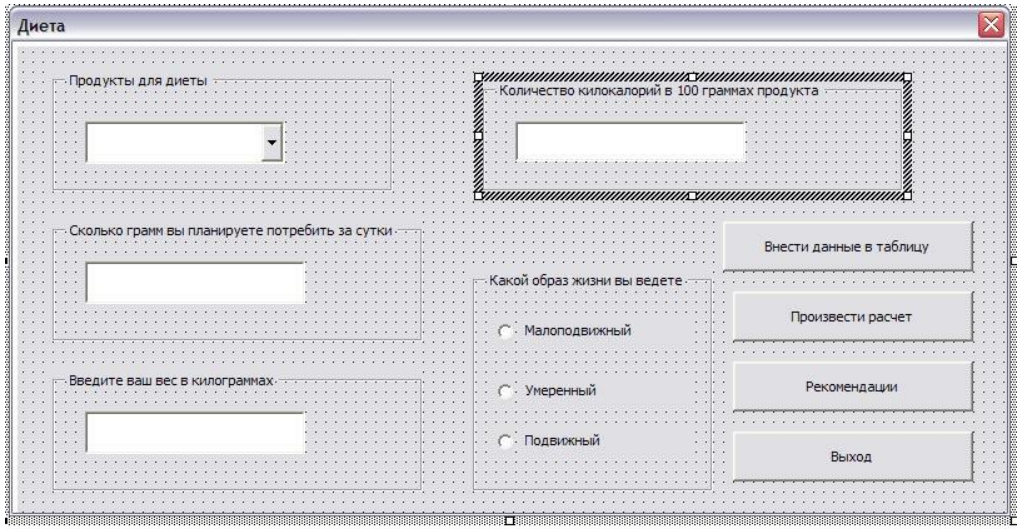

Рисунок 1 – Скриншот формы, которую необходимо создать

Данное приложение должно заполнять таблицу MSWord и производить расчет калорий согласно выбранным продуктам, а также давать рекомендации по выбранному вами меню.

1. В разделе «Продукты для диеты» пользователь должен выбрать предложенный из списка продукт. После этого, в разделе «Количество килокалорий в 100 граммах продукта» должна появиться информация необходимая информация.

2. Затем пользователь вводит в текстовые поля информацию о планируемом потреблении данного продукта и о своем весе в килограммах.

3. Также пользователь должен указать информацию о своем образе жизни.

4. Для того, чтобы внести информацию о выбранном продукте в таблицу, пользователю нужно нажать кнопку с одноименным названием.

5. Для итогового расчета, пользователь должен нажать на кнопку «Произвести расчет».

6. Для получения рекомендаций по диете – нажать на кнопку «Рекомендации».

7. Для выхода из приложения – нажать кнопку выход.

Ход выполнения работы

Ход работы:

1. Вызвать редактор VBA и создать макет данного приложения (разместить объекты на форме). Для работы с вложенным меню воспользуйтесь объектом ComboBox.

2. Заполнить список объектов (данные смотри в приложении 1).

3. Описание действий для кнопки «Внести данные в таблицу»:

После того, как пользователь выбрал продукт и внес всю необходимую информацию (в том числе, указал какой он образ жизни ведет) данная информация должна быть внесена в таблицу, следующего вида:

Продукты	Кол-во ккал в 100 граммах	Общая сумма ккал по продукту
Ваш вес = __ кг		
Ваше меню составляет = __ ккал		

Вес и общее количество килокалорий, которое содержит меню пользователя должны будут заполняться, только тогда, когда пользователь нажмет кнопку «Произвести расчет» (воспользуйтесь объектом FormFields для вывода данных).

Продуктов может быть выбрано несколько, по мере заполнения таблицы должны добавляться строчки.

4. При нажатии кнопки «Рекомендации» пользователь должен получить информацию о том подходит ему данный рацион или нет. Это должно быть рассчитано следующим образом: если пользователь ведет малоподвижный образ жизни, то его вес должен быть умножен на 24, если

умеренно активен, то вес необходимо умножить на 30, и на 44, если очень активен. Данные параметры будут считаться нормой, если меню в килокалориях будет меньше нормы или больше, то должны быть соответствующие рекомендации в виде сообщений (MsgBox). Например: «Ваше меню составляет 2300 калорий, что превышает вашу норму (1500). Советуем выбрать менее калорийное меню».

На рисунках 2 – 9 изображено выполненное задание №1.

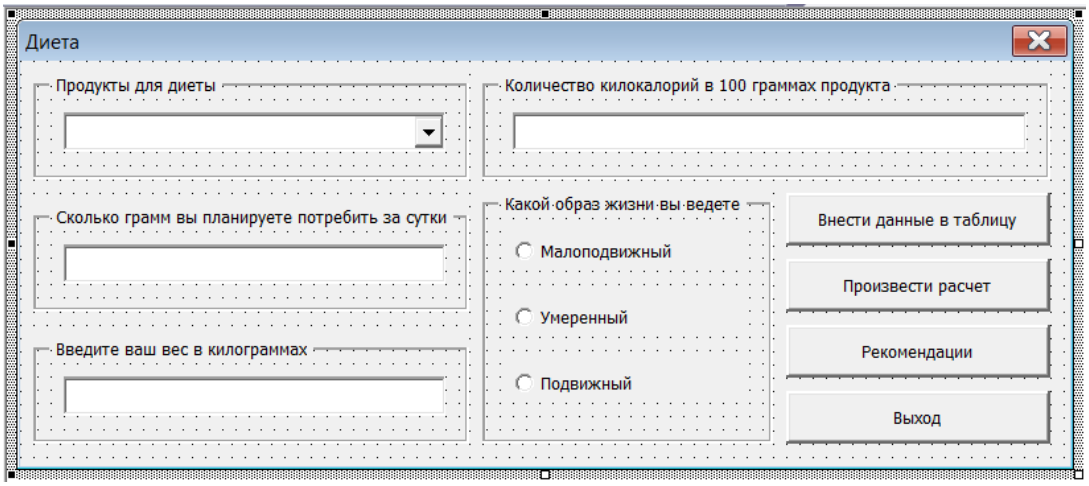

Рисунок 2 – Скриншот получившейся формы

```
Practica10 - UserForm1 (Code)
(General)                                   (Declarations)

Dim sum As Long
Private Sub CloseButton_Click()
UserForm1.Hide
End Sub

Private Sub CountButton_Click()

If ActiveDocument.Tables(1).Rows.Count = 5 Then
MsgBox "Добавьте хотя бы один продукт"
Else
If WeightBox.Text = "" Then
MsgBox "Введите свой вес"
Else
ActiveDocument.Bookmarks("weight").Select
Selection.Text = WeightBox.Text & " кг"
ActiveDocument.Bookmarks.Add Name:="weight", Range:=Selection.Range

ActiveDocument.Bookmarks("kalories").Select
Selection.InsertFormula Formula:="=SUM(C2:C100)"
ActiveDocument.Bookmarks.Add Name:="kalories", Range:=Selection.Range

End If
End If

End Sub

Private Sub InputButton_Click()

With ActiveDocument.Tables(1)

.Rows.Add (.Rows(.Rows.Count - 1))
.Cell(Row:=.Rows.Count - 4, Column:=1).Range.Text = ProductsList.Text
.Cell(Row:=.Rows.Count - 4, Column:=2).Range.Text = KilocaloriesBox.Text
.Cell(Row:=.Rows.Count - 4, Column:=3).Range.Text = KilocaloriesBox.Value * GrammsBox.Value \ 100
End With

sum = sum + KilocaloriesBox.Text * GrammsBox.Text \ 100

End Sub
```

Рисунок 3 – Скриншот кода формы (1)

```
Private Sub ProductsList_Change()

Select Case ProductsList.Value
Case "Рыба": KilocaloriesBox.Value = 170
Case "Мясо": KilocaloriesBox.Value = 264
Case "Яйца": KilocaloriesBox.Value = 160
Case "Фрукты": KilocaloriesBox.Value = 60
Case "Овощи": KilocaloriesBox.Value = 50
Case "Орехи": KilocaloriesBox.Value = 500
Case "Молочные": KilocaloriesBox.Value = 250
End Select

End Sub
```

Рисунок 4 – Скриншот кода формы (2)

```
Private Sub RecButton_Click()
Dim k As Integer
If ActiveDocument.Tables(1).Rows.Count = 5 Then
   MsgBox "Добавьте хотя бы один продукт"
Else
   If WeightBox.Text = "" Then
      MsgBox "Введите свой вес"
   Else

      If SlowOpt.Value = True Then
         k = 24
      End If

      If AverOpt.Value = True Then
         k = 30
      End If

      If FastOpt.Value = True Then
         k = 44
      End If

      If sum > WeightBox.Value * k Then
         MsgBox "Ваше меню составляет " & sum & _
         " калорий, что превышает вашу норму (" & WeightBox.Value * k & "). Советуем выбрать менее калорийное меню"
      Else
         MsgBox "Ваше меню составляет " & sum & _
         " калорий, что меньше вашей нормы (" & WeightBox.Value * k & "). Советуем выбрать более калорийное меню"
      End If

   End If
End If
```

Рисунок 5 – Скриншот кода формы (3)

```
Private Sub UserForm_Initialize()

ProductsList.AddItem "Рыба"
ProductsList.AddItem "Мясо"
ProductsList.AddItem "Яйца"
ProductsList.AddItem "Фрукты"
ProductsList.AddItem "Овощи"
ProductsList.AddItem "Орехи"
ProductsList.AddItem "Молочные"

AverOpt.Value = True

End Sub
```

Рисунок 6 – Скриншот кода формы (4)

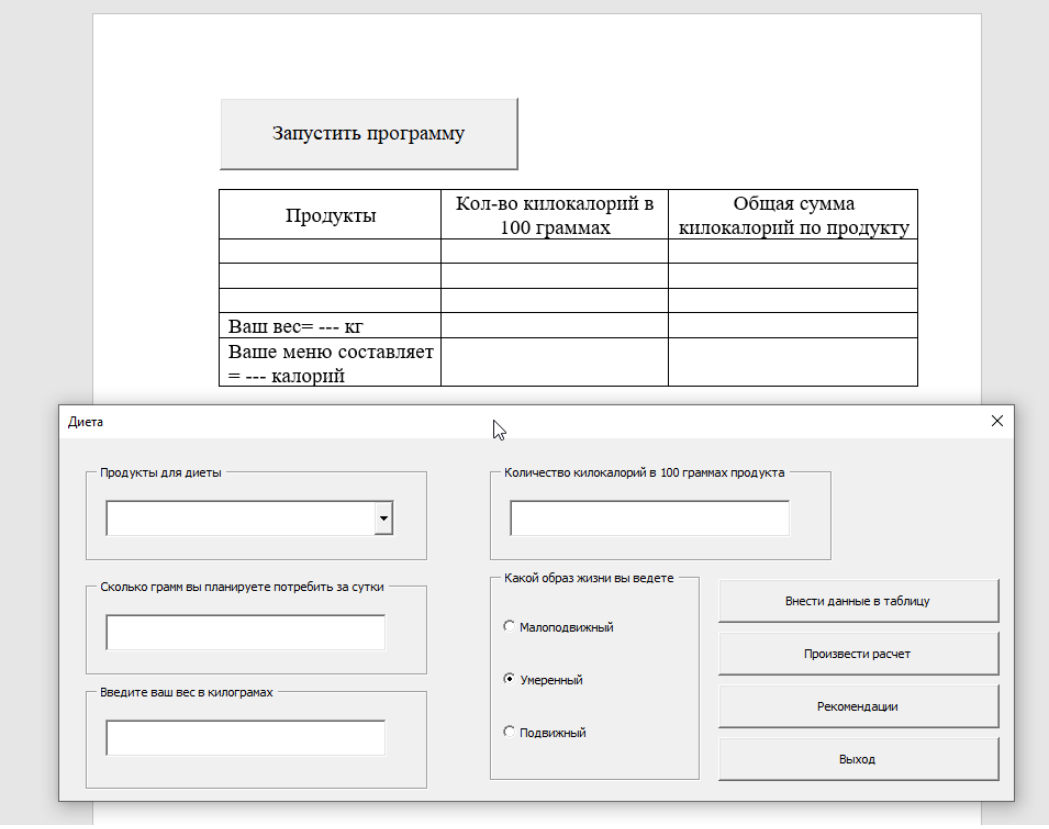

Рисунок 7 – Скриншот результата работы формы при запуске

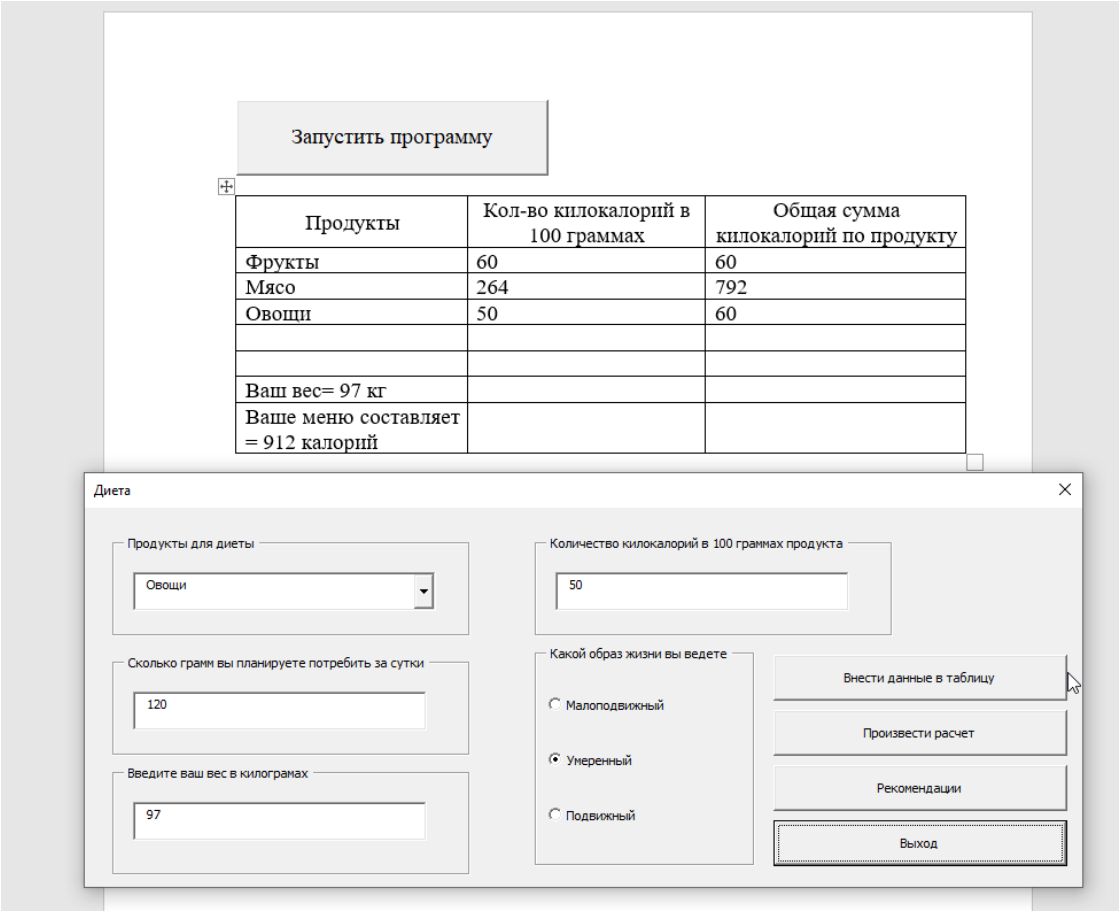

Рисунок 8 – Скриншот результата работы формы при нажатии кнопок: “Внести данные в таблицу” и “Произвести расчеты”

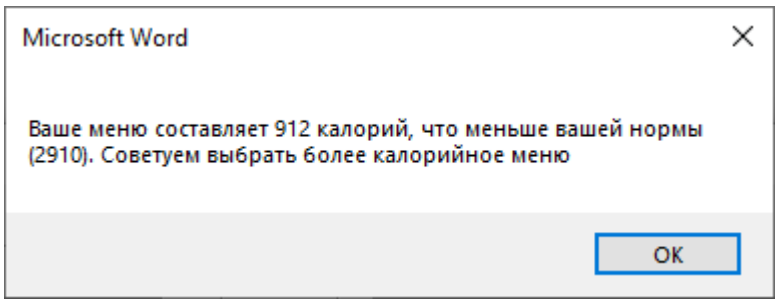

Рисунок 9 – Скриншот результата работы формы при нажатии кнопки “Рекомендации”

Вывод по лабораторной работе: в ходе лабораторной работы было произведено знакомство с основными объектами VBA Microsoft Word. На практике рассмотреть основные операции над объектами Application, Document, Selection, Range, Bookmark.