

**В.Ю. Петров**

# **информационные технологии в менеджменте**

**Учебное пособие**



**Санкт-Петербург**

**2015**

Петров Вадим Юрьевич. Информационные технологии в менеджменте. Учебное пособие. – СПб: Университет ИТМО, 2015. – 76с.

Учебное пособие позволяет студентам познакомиться с использованием информационных технологий для прикладных задач менеджмента, овладеть навыками разработки приложений в среде Office для создания простых информационных систем. Рассмотрены начальные сведения о программировании на Visual Basic for Application (VBA), использовании встроенных функций, вопросы создания пользовательского интерфейса. Многие примеры взяты из реальных разработок офисных программ для бухгалтеров, экономистов и инженеров.

Для студентов специальностей 080200.62 «Менеджмент», 080200.62.01 «Производственный менеджмент», 080200.62.02 «Управление проектом», 080200.62.05 «Финансовый менеджмент», 080801 «Прикладная информатика в экономике» и др.

Рекомендовано к печати на заседании ученого совета Гуманитарного факультета, протокол № 11 от 16/12/2014.



**Университет ИТМО** – ведущий вуз России в области информационных и фотонных технологий, один из немногих российских вузов, получивших в 2009 году статус национального исследовательского университета. С 2013 года Университет ИТМО – участник программы повышения конкурентоспособности российских университетов среди ведущих мировых научно-образовательных центров, известной как проект «5 в 100». Цель Университета ИТМО – становление исследовательского университета мирового уровня, предпринимательского по типу, ориентированного на интернационализацию всех направлений деятельности.

© Университет ИТМО, 2015  
© В.Ю. Петров, 2015

## ОГЛАВЛЕНИЕ

<b>Введение</b> .....	<b>5</b>
<b>1. Исследование встроенных функций Excel и запись макросов</b> .....	<b>6</b>
1.1. Основные положения.....	6
1.2. Исследование выполнения некоторых функций.....	7
1.2.1. Логические функции.....	7
1.2.2. Функции даты и времени.....	9
1.2.3. Математические функции.....	9
1.2.4. Функции просмотра и ссылок.....	11
1.3. Запись макросов.....	16
<b>2. Исследование встроенных функций и проверка работы операций</b> .....	<b>17</b>
2.1. Основные положения.....	17
2.1.1. Основные окна, используемые при отладке.....	17
2.1.2. Управление печатью в окне «Проверка» с использованием разделителей.....	19
2.1.3. Управление печатью с помощью функции «Format».....	20
2.1.4. Особые условия.....	22
2.2. Исследование работы строковых функций.....	22
2.3. Исследование работы функций из категории «Дата и время».....	24
2.4. Исследование работы логических операторов.....	26
2.5. Функции логических проверок.....	28
<b>3. Редактор VBA</b> .....	<b>29</b>
3.1. Интегрированная среда разработки. Общие положения.....	29
3.2. Окна редактора VBA.....	30
3.2.1. Окно проекта (Project explorer).....	30
3.2.2. Окно модуля.....	31
3.2.3. Окно форм и окно свойств.....	34
3.3. Создание простейших программ и экранных форм.....	34
3.4. Задание параметров редактора.....	36
<b>4. Создание функций, процедур и их отладка. Применение циклов и повторяемых структур</b> .....	<b>38</b>
4.1. Основные положения.....	38
4.2. Запуск и отладка процедур.....	39
4.3. Создание пользовательских функций.....	41
4.3.1. Определения и особенности создания пользовательских функций.....	41
4.3.2. Вызов пользовательских функций для использования в формулах рабочего листа.....	43
4.3.3. Аргументы пользовательских функций.....	44
4.3.4. Использование функций рабочего листа в процедурах и функ-	

циях VBA.....	44
4.4. Создание процедур.....	45
5. <b>Встроенные диалоговые окна</b> .....	47
5.1. Ввод и вывод информации. Функции MsgBox() и InputBox().....	47
5.1.1. Поиск ошибок в процедуре .....	47
5.1.2. Создание собственных процедур.....	48
5.2. Вызов диалоговых окон приложения и работа с ними.....	48
6. <b>Использование элементов управления, встраиваемых на рабочие листы</b> .....	50
6.1. Использование кнопок.....	50
6.2. Использование выключателей, полей ввода.....	54
6.3. Полоса прокрутки как элемент управления.....	54
7. <b>Создание диалоговых окон пользователей. Исследование работы элементов управления. Процедуры обработки событий</b> ..	58
7.1. Основные положения.....	58
7.2. Страницы, флажки и переключатели в диалоговых окнах.....	58
7.3. Метки (надписи) и поля ввода.....	61
7.4. Список и выключатель.....	62
7.5. Комбинированный список.....	63
7.6. Полоса прокрутки и счетчик.....	65
8. <b>Создание приложения Excel для автоматизации расчетов</b> .....	67
9. <b>Построение диаграмм</b> .....	70
<b>Литература</b> .....	73

## ВВЕДЕНИЕ

Начиная с Office 95, офисная среда отражает тенденции развития информационных технологий и претерпевает изменения как платформа разработки программных решений, создания информационных систем. Встроенный язык программирования *Visual Basic for application (VBA)*, начиная с использования его в Excel 5, все более соответствует требованиям объектно-ориентированного программирования и универсальных языков высокого уровня и в совокупности с возможностями среды Office позволяет творить чудеса.

Программирование в среде Office называют «Офисным программированием». Успех создания приложений на современном этапе зависит, в конечном итоге, от того, насколько широко программисты и пользователи будут использовать платформу Office для построения своих решений. В связи с этим важным моментом является знание VBA и основ офисного программирования.

Пособие последовательно, шаг за шагом, вводит пользователя в мир объектов Office, их методов, свойств, событий, знакомит с интегрированной средой разработки, технологией визуального программирования. В работе приводится большое количество примеров и заданий, требуемых конкретного или альтернативного решения. Такой подход всегда полезней и информативней, чем разговоры о том, как создать программы. Автор постарался указать и на типичные, характерные ошибки пользователей, возникающие при программировании, и пути их обхода или преодоления.

Основное внимание в пособии уделено табличному (Excel) процессору, имеющему более зрелые модели объектов и наиболее востребованным с точки зрения офисного программирования. Учитывая, что в ряде случаев студенты используют старые версии Office, а именно Office 97-2003, в пособии отмечено, как произвести те или иные действия для этих пакетов программ.

# 1. ИССЛЕДОВАНИЕ ВСТРОЕННЫХ ФУНКЦИЙ EXCEL И ЗАПИСЬ МАКРОСОВ

## 1.1 ОСНОВНЫЕ ПОЛОЖЕНИЯ

Простые формулы в ячейки можно вводить вручную. Но при сложных вычислениях, а также для ускорения, упрощения организации вычислений, для сокращения числа ошибок следует использовать *Мастер функций*.

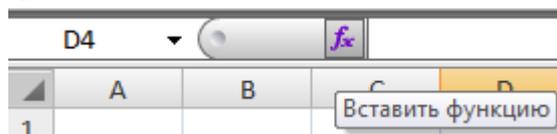
*Мастер функций* - это диалог, позволяющий выбрать и выполнить большинство из часто употребляемых, существующих функций. Все для выполнения этих функций в Excel уже готово, и Вам необходимо только указать на их аргументы, а если точнее, адреса ячеек, где эти аргументы хранятся.

Во многих случаях в качестве аргументов можно использовать все те же встроенные функции пакета. Такой подход позволяет реализовать достаточно сложные алгоритмы и математические выкладки.

Для запуска мастера функций необходимо активизировать ту ячейку, в которую хотите поместить функцию (формулу), после чего:

✓ или щелкнуть мышкой по кнопке  $f_x$ , расположенной рядом со строкой формул,

✓ или выполнить команду *Формулы/Библиотека функций/Вставить функцию*.



*! Русский алфавит при наборе адресов ячеек не используйте.*

Как оформлять работу? Новый материал должен быть законспектирован. Все исследования оформите с помощью средств Excel, включая панель рисования. Результаты работы сохраните на внешнем носителе. Для примера на рисунке 1.1 приведен вариант оформления работы для случая изучения логической функции «ИЛИ».

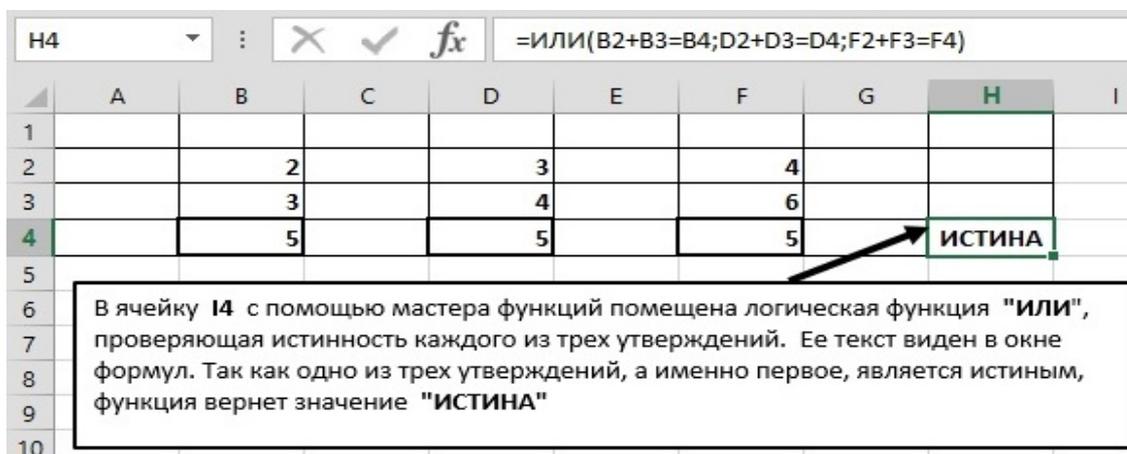


Рисунок 1.1 - Пример оформления выполненного задания

Что нужно сделать в работе? Самостоятельно решите задачи и проверьте правильность работы функций, указанных ниже. Все функции выполните, следуя диалогам, которые предложит вам мастер функций. В том случае, если что-то не получается, используйте справку в рабочей таблице мастера функций.

## 1.2. ИССЛЕДОВАНИЕ ВЫПОЛНЕНИЯ НЕКОТОРЫХ ФУНКЦИЙ

### 1.2.1. Логические функции

**Задание № 1.** С помощью электронных таблиц решите задачу в соответствии с приведенным алгоритмом.

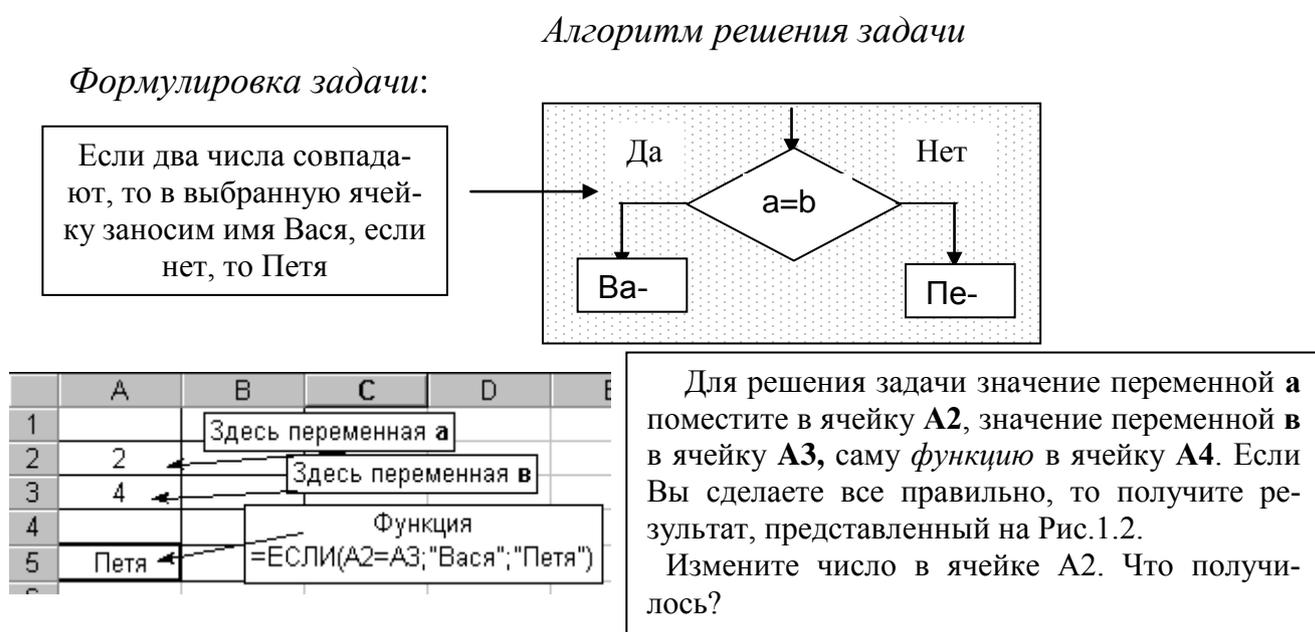


Рисунок 1.2 Пример оформления выполненного задания №1

*Важная информация.* В задании №1 использовалась функция «**ЕСЛИ**». Отметим, что при наборе аргументов этой функции (как, впрочем, и других впоследствии) адреса ячеек в поля ввода (рисунок 1.3) или в рабочие ячейки листа следует вводить, щелкая по ним. Например, (см. рис.1.3.), поле "**Лог\_выражение**" заполняем так: сначала щелкните по ячейке "**А2**", затем введите символ ">" с клавиатуры, потом щелкните по ячейке "**А3**".

После заполнения всех полей окна «*Аргументы функции*» в строке *формул* появится текст формулы, а в *поле адреса*, слева от строки формул – название функции (см. рисунок 1.3.)

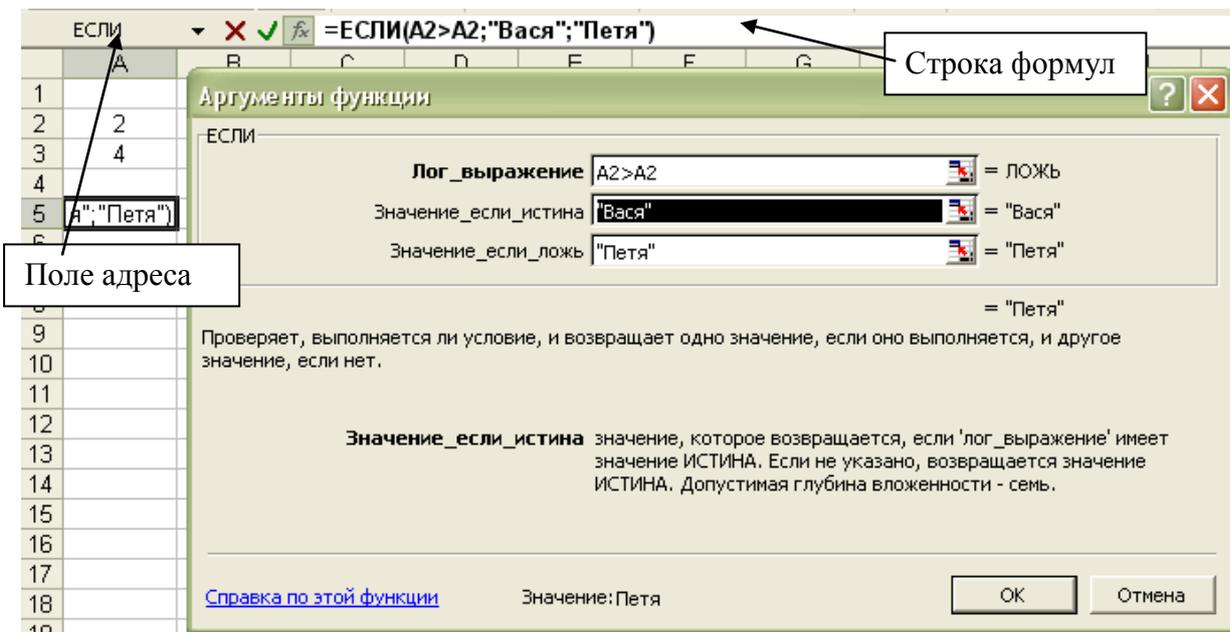


Рисунок 1.3. Ввод аргументов функции в рабочее окно мастера функций

В окне "*Аргументы функции*" есть еще масса полезной информации:

- ✓ сообщение о том, что делает функция;
- ✓ при щелчке по каждому полю для ввода аргументов выводится информация о том, что и в каком виде необходимо в него вводить;
- ✓ около каждого поля приводится возвращаемое им значение;
- ✓ выводится значение, возвращаемое функцией, причем уже здесь при неверно введенной информации будет прописана ошибка;
- ✓ если щелкнуть по "*Справка по этой функции*" будет выведена не только справочная информация, но и примеры;
- ✓ если щелкнуть по полю для ввода аргумента, а после этого по *полю адреса*, то в качестве аргумента можно использовать любую из встроенных функций Excel.

### Задание № 2.

В качестве аргумента функции «*ЕСЛИ*», можно использовать и ту же самую функцию «*ЕСЛИ*», и другие функции. Причем, уровень вложенности функций может быть достаточно велик. Это позволяет создавать функции для сложных алгоритмов.

Используя приведенный на рисунке 1.4 алгоритм, решите задачу с помощью электронных таблиц. Здесь Вам понадобится применить вложение функций.

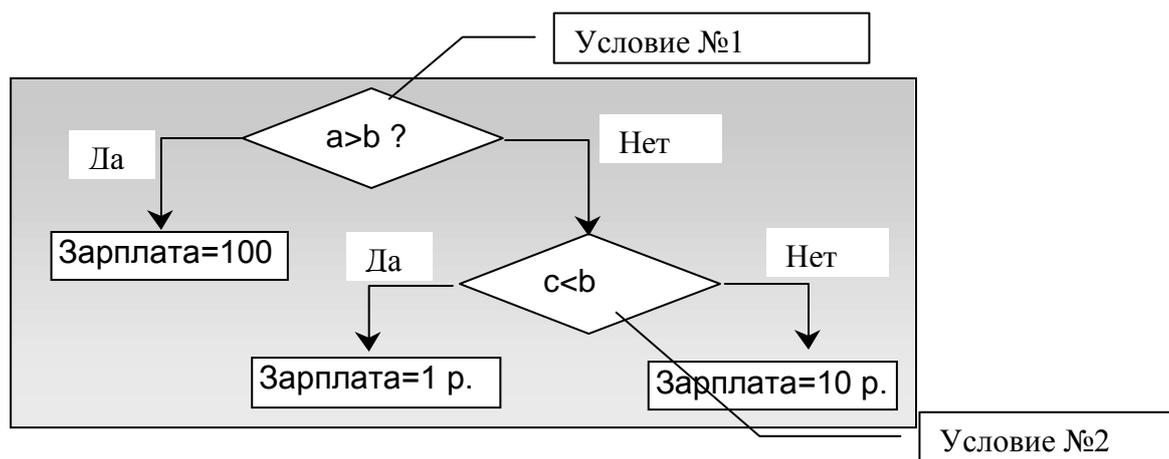


Рисунок 1.4 –Алгоритм для задания №2

## 1.2.2 Функции даты и времени

	A	B	C	D	E	F	G	H
1								
2	03.08.99	← 3	←	=ДЕНЬНЕД(A2), оперделяет, каким по счету днем недели является дата , хранящаяся в				
3								
4	09.09.99	← 36	←	=ДНЕЙ360(A2;A4), определяет количество дней между двумя датами.				
5								
6	07.09.88	← 9	←	=МЕСЯЦ(A6), Оперделяет номер меся в году , у даты из ячейки A6				
7								
8	В ячейки колонки A даты вводим сами. Это исходные данные		20.08.99	=СЕГОДНЯ(), вводит в ячейку сегодняшнюю дату.				
9								
10			6	=ДЕНЬНЕД(СЕГОДНЯ()), возвращает какой сегодня день недели. Здесь нужно использовать вложенные функции				
11								
12								
13								
14								

Рисунок 1.5 – Результаты выполнения задания №3

### Задание № 3.

- ✓ Произведите действия, которые требуется выполнить для получения результатов, представленных на рисунке 1.5.
- ✓ Используя табличный процессор, определите, в какой день недели Вы родились?
- ✓ Определите, сколько дней Вы прожили от рождения до сегодняшнего дня?

## 1.2.3 Математические функции

### 1.2.3.1. Простые, часто встречающиеся функции

Задание № 4. Самостоятельно исследуйте работу функций, таких как:

- ✓ **ЗНАК, ABS, ОСТАТ, ОТБР,**
- ✓ **МИН, МАКС,**

✓ исследуйте функцию **РАНГ**, которая определяет, какое именно место по возрастанию занимает ваше число среди массива чисел.

✓ Придумайте и приведите примеры использования функций округления чисел: **ОКРУГЛ**, **ЦЕЛОЕ**, **ОКРВВЕРХ**, **ОКРВНИЗ**. Чем они отличаются?

### 1.2.3.2. Функции **СЧЕТЕСЛИ**. **СУММЕСЛИ**

**Задание № 5.** Приведены данные, определяющие должности и оклады сотрудников предприятия, которые сведены в таблицу 1.1

Таблица 1.1 – Исходные данные

Фамилия	Должность	Оклад
Петров	инж	100
Иванов	тех	50
Кацман	менеджер	1000
Ли	бармен	20
Портная	инж	100
Худая	тех	70
Веселова	инж	120

Таблица 1.2 – Результирующие данные

Должность	Кол-во	Суммарный оклад
инж	?	?
тех	?	?
бармен	?	?

Ваша задача: так составить таблицу 1.2, чтобы в ней, исходя из данных таблицы 1.1, автоматически определялось:

- ✓ сколько по отдельности инженеров, техников и барменов работает на предприятии?
- ✓ сколько человек на предприятии имеет оклад менее 120?
- ✓ в примере используйте функции **СЧЕТЕСЛИ** и **СУММЕСЛИ**.

**Задание № 6.** Задана таблица с названиями фирм, к которым сегодня обращались клиенты, и текстом: «Успех», «Крах».

Таблица 1.3 - Вариант для задания №6

	А	В	С	Д	Е
1					
2		Дом +	Геоид	Нева	Техника
3		Геоид	Дом +	Нева	Геоид
4		Геоид	Геоид	Дом +	Нева
5		Нева	Техника	Нева	Дом +
6					
7			Успех	Крах	

Необходимо поместить в ячейку **Е7**, функцию, которая бы выполняла следующее:

- ✓ считала бы количество встретившихся имен «Геоид» и «Нева»,
- ✓ сравнивала бы эти две суммы.

✓ Помещала бы в ячейку **Е7** слово, содержащееся в ячейке **С7**, если количество имен «Геоид» *больше* количества имен «Нева».

✓ Помещала бы в ячейку **E7** слово, содержащееся в ячейке **D7**, если количество имен «Геоид» *меньше* количества имен «Нева».

В данном примере Вам понадобится использовать несколько функций. Можно, например, применить функции **ЕСЛИ** и **СЧЕТЕСЛИ**, но можно и какие-то другие. Сначала попробуйте выполнить отдельные задания, используя для выполнения каждого из них свою отдельную ячейку. В этом случае задача будет решена по частям, в несколько этапов.

После того, как будет получен положительный результат, синтезируйте требуемую функцию в одной ячейке. Результаты запомните.

## 1.2.4. Функции просмотра и ссылок

**1.2.4.1. Функция «ВЫБОР».** Возвращает одно значение из списка. На это значение указывает отдельный аргумент - номер индекса. (См. справку по функции)

**Задание № 7.** На чистом листе составьте таблицу с исходными данными, хранящимися в столбцах **B** и **C**, изображенную на рисунке 1.6. В столбце **D**, в указанные ячейки вставьте функции в соответствии с указаниями, содержащимися на том же рисунке.

	A	B	C	D	E	F	G	H
1	<b>Ведомость выдачи ЗП</b>				<p>В эту ячейку будем вводить индекс. Для данного случая он совпал с № п.п. В принципе, индекс может быть и числом рассчитанным по формуле.</p> <p>Функция, введенная в эту ячейку вернет фамилию гражданина в соответствии с введенным в ячейку <b>D2</b> индексом.            =ВЫБОР(D3;B3;B4;B5;B6;B7;B8;B9)</p>			
2	№ п.п	Список (Фамилия)	Список_0 (Сумма)					
3	1	Петров	100	2				
4	2	Иванов	200	Иванов				
5	3	Сидоров	300					
6	4	Уткин	400					
7	5	Гусев	500					
8	6	Собакин	600	100				
9	7	Свиньин	700	300				
10								
11	<p>Сюда поместим функцию, которая суммирует премию с ЗП выбранного по индексу гражданина:            =СУММ(D8;ВЫБОР(D3;C3;C4;C6;C7;C8;C9))</p>				<p>В этой ячейке находится сумма премии, которую мы добавим к зарплате гражданина, которого вычислим через индекс из ячейки <b>D3</b>.</p>			
12								
13								
14								
15								

Рисунок 1.6 – Исходные данные и результат выполнения заданий №№ 7-8

**Задание № 8.** В списке на рисунке 1.6 содержится 7 фамилий.

Ваша задача: сделать так, чтобы из этого списка автоматически производился выбор дежурного на каждый день недели. В понедельник - Пет-

ров, во вторник - Иванов и т.д. Здесь необходимо использовать функции *Даты и времени* и **ВЫБОР**.

#### 1.2.4.2. Функция «ДВССЫЛ»

Функция «ДВССЫЛ» используется для того, чтобы получить значение из ячейки по ссылке, которая находится в другой ячейке. В данном случае в процессе обмена задействованы три ячейки:

- **ячейка с функцией**, куда должна быть возвращена информация,
- **ячейка с информацией**. Информация из этой ячейки возвращается в ячейку с функцией,
- **ячейка содержащая адрес второй ячейки, той, где хранится информация**. (подробней см. справку).

Процесс обмена информацией для функции «Двойная ссылка» представлен на рисунке 1.7.

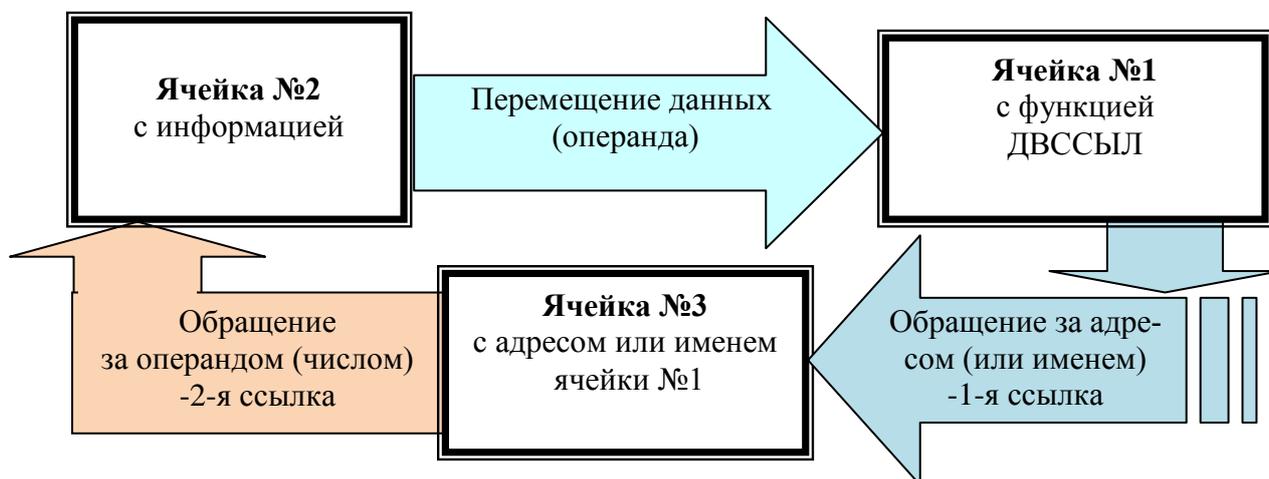


Рисунок 1.7- Перемещение данных при выполнении команды «ДВССЫЛ»

**Задание № 9.** У Вас есть огромная таблица (назовем ее основная), в которую собраны сведения о сотрудниках вашего предприятия, их фамилии, должности и оклады. Сотрудников много, должностей значительно меньше. Это значит, что сотрудников с одинаковыми должностями несколько. Причем, каждой должности соответствует своя заработная плата.

Представьте, что каждой должности поставлена в соответствие новая заработная плата. Это повлечет за собой необходимость всем гражданам во всей таблице изменить величину заработной платы. Если предприятие большое это займет много времени.

**Ваша задача** автоматизировать процесс замены окладов у сотрудников.

*Конкретизируем эту задачу.*

Пусть «Основная таблица» для вашего предприятия представлена на рисунке 1.8. В нее включено всего шесть человек, которые могут занимать три должности. Создайте ее на чистом листе.

Для автоматизации процесса замены окладов у сотрудников, создадим вспомогательную таблицу с должностями и соответствующими им окладами, не касаясь фамилий. В принципе, эта таблица должна быть у вас в любом случае, поскольку несет в себе определяющие, во многих случаях, справочные данные. Пусть в ней будет всего 3 строчки - 3 должности и 3 оклада.

**Необходимо сделать так, чтобы:**

- ✓ смена оклада в этой вспомогательной таблице влекла за собой изменение окладов во всей большой таблице;
- ✓ изменение должности у сотрудника в основной таблице автоматически изменяло и его оклад.

В представленном варианте вы выигрываете всего в два раза. Но если предприятие большое, то выигрыш будет велик.

Для того чтобы добиться сказанного необходимо использовать функцию **ДВССЫЛ**, причем, со ссылкой по имени. Завершите изменения таблиц, представленных на рисунке 1.8, и докажите, что они работоспособны.

	A	B	C	D	E	F	G
1	Вспомогательная таблица			<p>1. Ячейкам столбца "Оклад" присвойте имена: "Матрос", "Солдат", "Санитар"</p> <p>2. Данные в ячейки вводите любые. При этом в основной таблице д.б. изменены соответствующие оклады</p>			
2	№	Должность	Оклад				
3	1	Матрос	100				
4	2	Солдат	200				
5	3	Санитар	300				
6							
7	Основная таблица			<p>В ячейки столбца "Оклад" вставьте функцию ДВССЫЛ для того, чтобы получить представленный результат. При изменении должности человека должен измениться оклад</p>			
8	Гражданин	Должность	Оклад				
9	Денис	Матрос	100				
10	Вика	Солдат	200				
11	Оксана	Санитар	300				
12	Ваня	Матрос	100				
13	Катя	Матрос	100				
14	Вадим	Солдат	200				
15	Наталья	Санитар	300				

Рисунок 1.8. - Исходные данные и результат выполнения функции «ДВССЫЛ»

**1.2.4.3 Функция «СМЕЩ»**

Возвращает ссылку на диапазон, отстоящий от ячейки или диапазона ячеек на заданное число строк и столбцов. Возвращаемая ссылка может

быть отдельной ячейкой или диапазоном ячеек. Можно задавать количество возвращаемых строк и столбцов.

### **Синтаксис**

**СМЕЩ**(ссылка; смещ\_по\_строкам; смещ\_по\_столбцам; высота; ширина)

Эта функция хорошо прокомментирована в справке. Кроме того, как уже отмечалось, в справке есть примеры.

Здесь же отметим только то, что при описании функции (в справке либо в окне диалога) некоторые ее аргументы - высота и ширина представлены текстом с обычным начертанием, не полужирным. Это означает, что означенные параметры не являются обязательными и при обращении к функции их можно опустить.

**Задание № 10.** По справочным данным разберитесь с использованием функции «СМЕЩ» и приведите свои примеры ее использования.

При изучении работы этой функции удобнее всего заполнить поле из рядом стоящих ячеек (например, поле размером 5x5) разными цифрами. В центральную ячейку этого поля следует ввести функцию. Проанализируйте результат.

#### ***1.2.4.4 Функция «ЯЧЕЙКА»***

Это функция, которая позволяет получить массу полезных сведений о ячейке: адрес, номер строки или столбца, содержимое и т.д. (см. справку).

**Задание № 11.** Приведите примеры использования функции «ЯЧЕЙКА».

#### ***1.2.4.5. Функция «ИНДЕКС»***

Возвращает значение элемента таблицы или массива.

Функция **ИНДЕКС** имеет две синтаксические формы: ссылка и массив.

Для примера будем использовать форму массива, которая возвращает значение или массив значений. Ее синтаксис имеет вид:

**ИНДЕКС**(массив; номер\_строки; номер\_столбца)

#### **Задание № 12.**

✓ Изучите самостоятельно справочные данные для функции «ИНДЕКС».

✓ Создайте простейшую справочную систему, используя функцию «ИНДЕКС», в основу которой положены две таблицы, представленные на рисунке 1.9.

Пусть единая тарифная сетка задана в виде таблицы «Скрытая таблица». В принципе она должна быть скрытой от пользователей.

Для того, чтобы пользователь имел возможность ввести свой разряд и получить от машины тарифный коэффициент необходимо создать справочное табло.

Это табло представляет собой электронную таблицу, с двумя рабочими ячейками: в одну из которых граждане вводят свой разряд, а в другой получают тарифный коэффициент, который машина должна найти в скрытой таблице.

Скрытость таблицы не следует понимать формально в данной задаче.

Справочное табло	
Введи разряд	2
Данному разряду соответствует тарифный коэффициент	1,5

Скрытая таблица	
Разряд	Коэффициент
1	1
2	1,5
3	2
4	3
5	4
6	4,5

Рисунок 1.9 - Исследование работы функции «ИНДЕКС»

#### 1.2.4.6 Функция «ВПР»

Функция ищет значение в крайнем левом столбце таблицы и возвращает значение в найденной строке из указанного столбца таблицы.

По умолчанию таблица должна быть отсортирована по возрастанию.

	A	B	C	D	E	F	G	H
1								
2		Таблица				=ВПР(3,В3:Е8,4) - это пример использования ВПР		
3		1	3	5	30			
4		2	6	33	20		100	
5		3	2	55	100			
6		5	9	44	40			
7		10	8	11	50		9	
8		13	100	2	70		50	
9								
10						Ячейка с функцией ВПР	Ячейка для ввода	
11	В ячейку G8 поместите функцию "ВПР". Она должна выбрать необходимые данные из закрашенной части таблицы. Причем так, чтобы искомое значение, которое будем искать в первом столбце закрашенной таблицы, можно было бы вводить в ячейку G7							
12								
13								
14								

Рисунок 1.10 – Исследование работы функции «ВПР»

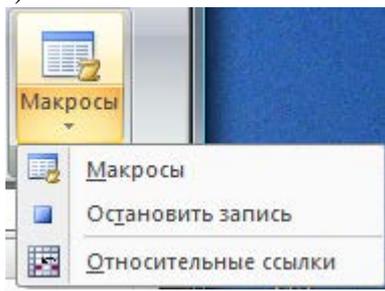
#### Задание № 13.

- ✓ Выполните действия, указанные на рисунке 1.10.
- ✓ Получите тот же результат, используя функцию «ГПР»

### 1.3. ЗАПИСЬ МАКРОСОВ

*Макрос* — это программный код, написанный на языке Visual Basic for Application (VBA). В ранних версиях Excel - 5 и 7 версиях- он записывался на отдельных листах. Для версий Excel 97 и выше макрос создается в редакторе программ VBA. По классификации макрос — это такой программный код, который создал *Макрорекордер*.

Макрорекордер автоматически записывает все действия, произведенные в Excel в виде программного кода и запоминает его как отдельную процедуру - Макрос. Для запуска Макрорекордера следует выполнить команду *Вид/Макросы/Запись макроса* (для старых версий -*Сервис/Макрос/Начать запись*).



Для остановки процесса записи макроса макрорекордером выполните команду «*Остановить запись*» с вкладки макросы. Нижняя кнопка на этой панели включает режим относительных или абсолютных ссылок.

#### **Задание № 14.**

Запустите табличный процессор Excel и включите макрорекордер.

После присвоения макросу имени, на рабочем листе активизируйте ячейку A3. В этой ячейке напишите любую фразу, например, «Дай миллион».

Перейдите к ячейке B2.

Остановите запись.

Перейдите к *Модулю*, в котором находится записанный текст программы. Убедитесь, что программа существует. Для этого следует:

- ✓ перейти в редактор VBA - щелкните на клавиатуре по клавишам <Alt +F11>,
- ✓ в редакторе вызвать вызовите окно проекта – клавиши <Ctrl+R>,
- ✓ в окне проекта найти строку Modules/Module1 и щелкнуть по ней мышью,
- ✓ справа в рабочем окне модуля просмотреть программный код.

Запустите макрос на другом листе Excel, используя команду *Вид/Макросы/Макросы/имя макроса в рабочем окне*, и убедитесь, что он выполнен.

**Задание № 15.** Создайте еще один макрос, но с относительными ссылками. Ответьте, чем он отличается от первого, записанного (по умолчанию) с абсолютными ссылками.

## 2. ИССЛЕДОВАНИЕ ВСТРОЕННЫХ ФУНКЦИЙ И ПРОВЕРКА РАБОТЫ ОПЕРАЦИЙ

При работе со вторым разделом пособия рекомендуется составить отчет или конспект. Он должен включать все выполненные задания (1-6), с таблицами, примерами и комментариями по работе рассмотренных функций. В тех случаях, когда это возможно и целесообразно, материал разместите на «флешке», остальное законспектируйте с соответствующими выводами.

*! Учтите, что информация в "Окне Проверки", "Окне контрольных значений" и др. сбрасывается после закрытия приложения, и сохранить полученные в этом окне результаты на «флешке» или ином носителе не представляется возможным. Поэтому данные из этого окна конспектируйте или сохраняйте в каком-либо редакторе как рисунок.*

### 2.1. ОСНОВНЫЕ ПОЛОЖЕНИЯ

#### 2.1.1. Основные окна, используемые при отладке программ

Набор текста программ и изменение его в среде Office происходит в редакторе *Visual Basic for Application (VBA)*. Его вызов может быть произведен из любого приложения Office горячими клавишами  $\langle Alt+F1 \rangle$ , или из вкладки меню «Разработчик» из группы команд «Код» кнопкой «Visual Basic».

Для отладки программ, проверки работы большинства операций и функций в Excel предусмотрена удобная возможность использования ряда окон, которые открываются после запуска редактора.

	<u>Английское название</u>		<u>Русское название</u>
	Immediate Window	Ctrl+G	=Окно "Проверка"
	Locals Window		=Окно "Локальные переменные"=
	Watch Window		=Окно "Контрольные значения"=

Окно "*Локальные переменные*" автоматически отображает все объявленные переменные текущей процедуры и их значения.

Окно "*Контрольные значения*" применяется для просмотра значений выражений и переменных. Но для этого необходимо сначала перетащить мышью такие переменные из текста программы в окно.

Окно "*Проверка*" позволяет ввести инструкцию, которая будет немедленно выполнена после нажатия клавиши Enter:

✓ при необходимости вывода результатов расчета в этом окне перед выполняемой операцией поставьте слово «**print**» или знак ?

✓ . Значение любой *переменной* при выполнении программы может быть выведено на панель «Проверка» методом *Debug.Print переменная*

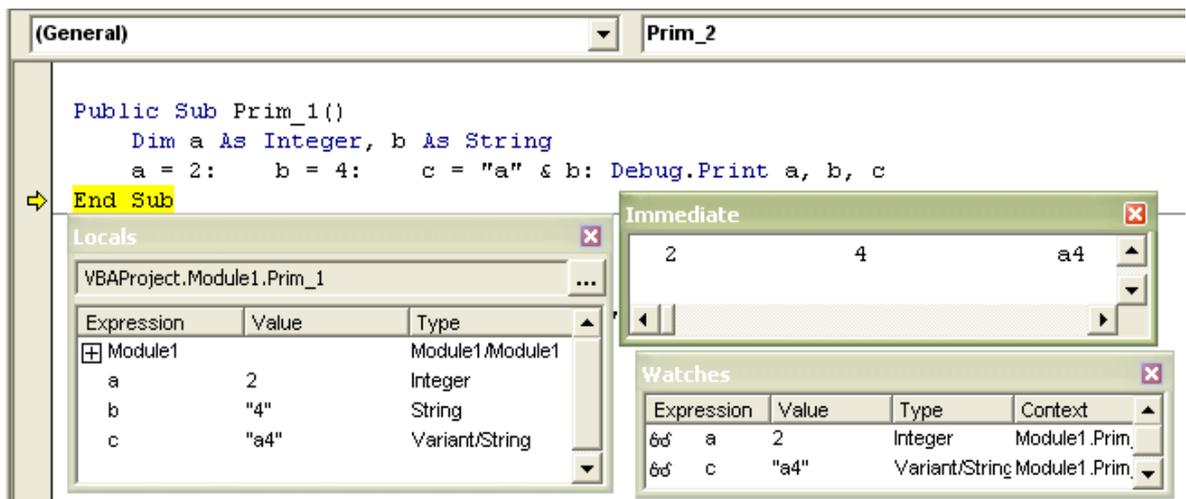


Рисунок 2.1. - Пояснения к заданию №1

**Задание №1.** Выполните инструкции, представленные на рисунке 2.1 и убедитесь в вышеизложенном. Для этого:

- ✓ запустите редактор VBA;
- ✓ в окне модуля наберите текст процедуры, представленной на рисунке 4.1;
- ✓ сделайте активными три указанных окна;
- ✓ щелкнув по имени процедуры, и нажимая после этого кнопку <F8> на клавиатуре, выполните программу в пошаговом режиме, оценивая и объясняя результаты, появляющиеся в окнах.

**Задание №2.** Проверьте действия, выполняемые инструкциями (функциями), в соответствии с рисунком 2.2.

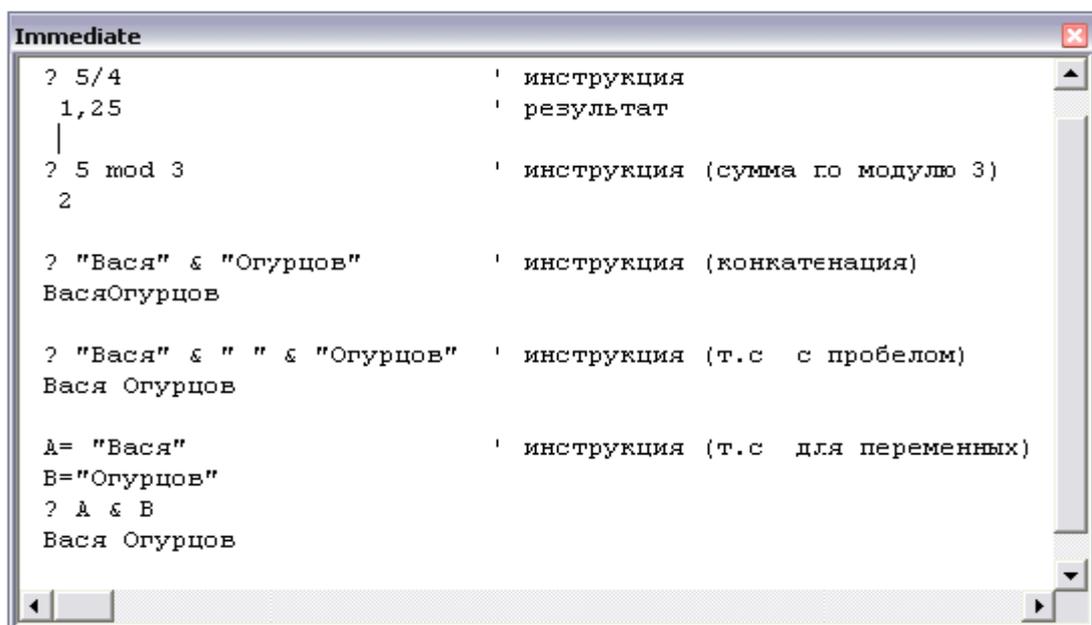


Рисунок 2.2 – Вид результатов для задания №2

## 2.1.2. Управление печатью в окне «Immediate» (Проверка) с использованием разделителей

При выводе информации на печать в окне «Immediate» (Проверка) (см. рисунок 2.3) имеют место следующие закономерности:

✓ при размещении в строке для печати нескольких элементов, разделенных *запятыми* (,), они будут напечатаны в блоках, изначально заполненных пробелами по 14 символов в блоке;

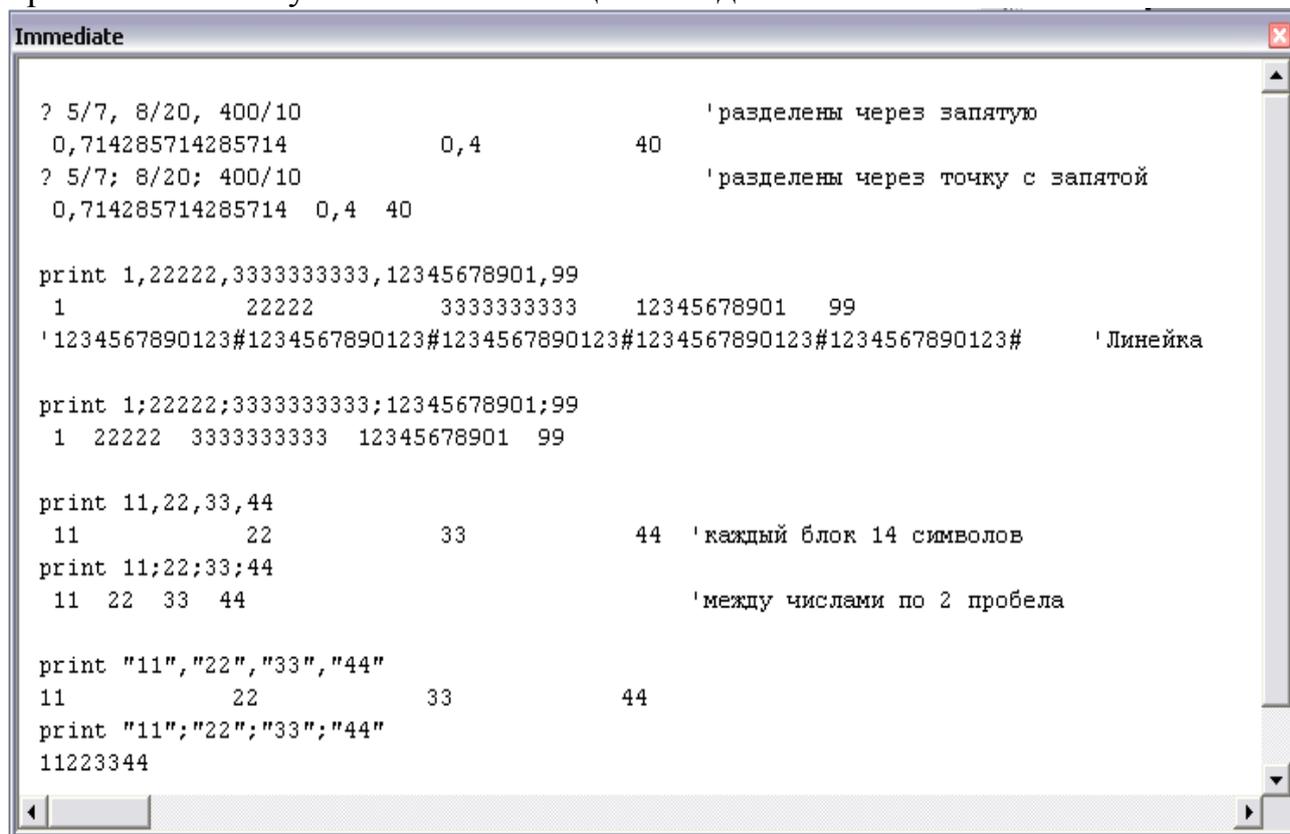
✓ при разделении элементов символом (;), напечатанная строка будет содержать по два пробела между числами: один ведущий, перед числом и второй конечный, после числа. Они появляются в результате преобразования числа в строку оператором вывода на печать.

✓ при разделении элементов символом (;) и печати символьных переменных пробелы не ставятся, и все слова на панели будут написаны слитно.

**Задание № 3.** Выполните инструкции, представленные на рисунке 2.3. При этом, обратите внимание на то, что было сказано выше, а также и то, что:

✓ ? и **Print** имеют один смысл как управляющие операторы;

✓ строка '1234567890123#12....использована в примере как линейка, по которой можно определить количество символов в строке над ней, причем символ # указывает на позицию каждого 14-го символа.



```
Immediate
? 5/7, 8/20, 400/10          'разделены через запятую
0,714285714285714          0,4          40
? 5/7; 8/20; 400/10        'разделены через точку с запятой
0,714285714285714 0,4 40

print 1,22222,3333333333,12345678901,99
1          22222          3333333333  12345678901  99
'1234567890123#1234567890123#1234567890123#1234567890123#1234567890123#  'Линейка

print 1;22222;3333333333;12345678901;99
1 22222 3333333333 12345678901 99

print 11,22,33,44
11          22          33          44  'каждый блок 14 символов
print 11;22;33;44
11 22 33 44          'между числами по 2 пробела

print "11","22","33","44"
11          22          33          44
print "11";"22";"33";"44"
11223344
```

Рисунок 2.3 – Печать с использованием разделителей

### 2.1.3. Управление печатью с помощью функции «Format»

В упрощенной форме синтаксис функции формат «*Format*» выглядит так:

*Format* (выражение, "формат")

Содержит выражение, представляющее число, строку или дату, подлежащую форматированию

Любой известный или созданный пользователем формат, в соответствии с которым должно быть отображено выражение (в кавычках)

#### *Преобразования числовых величин*

Первым аргументом функции *Format* является само число, которое необходимо напечатать (в данном случае напечатать на панели «Отладка»).

Вторым аргументом является строка форматирования, определяющая в каком виде это число будет напечатано, и содержащая ряд символов для указания этого вида:

- ⇒ символы (#) - используются в качестве цифровых заполнителей;
- ⇒ символы (, - запятая), указывают на то что каждые 3 символа отделяются запятыми;
- ⇒ символы (. - точка), указывают на положение десятичной точки;
- ⇒ символы (0 - ноль), указывают на позицию обязательного символа.

#### Задание № 4.

1. Выполните инструкции, представленные на рисунке 2.4. Объясните полученные результаты.

```
a=33.1234 'жду целыми и дробными десятичная точка !!!
b=0.12345
c=12345
print a, b, c
33,1234      0,12345      12345

Print format(a, "#.0"), format(a, "#.00"), format(a, "0.000"), format(a, "000.000")
33,1          33,12          33,123          033,123
Print format(b, "#.0"), format(b, "#.00"), format(c, "0.000"), format(c, "#,###.000")
,1            ,12            12345,000      12 345,000
```

Рисунок 2.4 – Результаты преобразования числовых величин

2. Заданы следующие числа: a=98733,12345 и b=0,0012345

Задайте такой формат, чтобы напечатать их в следующем виде

а	98733,1	98733,12	98733,123	098733,1234	98 733,1
в	,0	,001234500			

3. Выполните преобразования с использованием *именованных форматов*, представленные на рисунке 2.5.

```

Immediate
? Format (1234.1234567, "General Number") 'Число без выделения разрядов тысяч
1234,1234567
? Format (1234.1234567, "Fixed") 'Как минимум один знак слева и два справа
1234,12 'от десятичного разделителя
? Format (.1, "Fixed") 'без выделения разрядов тысяч
0,10

? Format (1234.1234567, "Standard") 'Число с выделением разрядов чисел и
1 234,12 'как минимум одним знаком слева и двумя справа
'от десятичного разделителя
? Format (1234.1234567, "Currency") 'Число с выделением разрядов чисел, с двумя знаками
1 234,12р. 'после десятичного разделителя,|
1 234,12р. 'а также символом денежной единицы

```

Рисунок 2.5 - Преобразования с использованием именованных форматов

### ***Преобразования даты и времени по заданным форматам***

#### **Задание № 5.**

Выполните преобразования в соответствии с материалом, представленным на рисунке 2.6

```

Immediate
? Format (Date, "mm/dd/yy")
11.10.03
? Format (Time, "hh:mm")
02:43
? Format (Now, "mm/dd/yy hh:mm")
11.10.03 02:44

'Именованные форматы
'-----
? Format (Date, "General date") 'дата, в соответствии с установками в панели управления.
10.11.2003
? Format (Date, "Long Date") 'дата в виде, заданном в панели управления
10 Ноябрь 2003 г. 'для полного формата даты.

? Format (Date, "Medium Date") 'для среднего формата даты.
10-ноя-03
? Format (Date, "Short Date") 'для краткого формата даты.
10.11.2003

'АНАЛОГИЧНО для форматов - Long Time, Medium Time, Short Time

```

Рисунок 2.6 – Преобразование даты и времени по заданным форматам

**2.1.4. Особые условия.** Если необходимо в окне Immediate проверить работоспособность выражения, состоящего из нескольких операторов, то следует записать их в одну строчку и поставить между каждой парой операторов двоеточие - «:» .

**Задание № 6.** Запустите в окне отладки трехстрочный цикл, выводящий цифры от 1 до 10. Слово «вывести» замените необходимым для этого оператором.

```
For i=1 to 10
вывести i
Next i
```

## 2.2. ИССЛЕДОВАНИЕ РАБОТЫ СТРОКОВЫХ ФУНКЦИЙ

Таблица № 2.1 – Строковые функции

№ п.п.	Функция	Описание	Примечание (Заполняем сами)
1	Lcase	Преобразовывает строку в нижний регистр	
2	Ucase	Преобразовывает строку в верхний регистр	
3	Len	Определяет длину строки	
4	Instr	Ищет подстроку	
5	Lset	Выравнивает строку по левому краю	
6	Rset	Выравнивает строку по правому краю	
7	Left	Выделяет левую часть строки	
8	Right	Выделяет правую часть строки	
9	Mid	Выделяет или перемещает подстроку	
10	LTrim	Удаляет ведущие пробелы	
11	RTrim	Удаляет завершающие пробелы	
12	Trim	Удаляет пробелы с двух сторон строки	
13	Asc	Возвращает ASCII код символа	Что такое ASCII код ?
14	Chr	Возвращает символ по ASCII коду	
15	Str	Преобразует число в строку	
16	Val	Преобразует строку в число	
17	StrComp	Сравнивает 2 строки	

### Задание № 7.

Повторите примеры, в которых представлено выполнение строковых функций, приведенные ниже. Курсивом в примерах выделены комментарии. Поэкспериментируйте с функциями. Перечертите таблицу, и, по мере выполнения функций, заполните столбец комментариев в таблице своими замечаниями. Она пригодится вам в дальнейшем.

#### *Примеры выполнения некоторых строковых функций*

*Инициализируем две переменные*

**A**="Маленький мальчик нашел пулемет"

**B**="Больше в деревне никто не живет"

? **Lcase(A)** *преобразуем в строчные символы*  
маленький мальчик нашел пулемет *это результат*

? **Instr(B,"никто")** *ищем место подстроки "никто"*  
18 *это результат*

? **Left(A,9)** *возвращает 9 первых, левых символов переменной A*  
Маленький *это результат*

? **Mid(B,18,5)** *возвращает 5 символов, начиная с 18-го из переменной B*  
Никто *это результат*

**C**=" Hello " *переменной C присваиваем значение, содержащее пробелы.*

? **"|" & Ltrim( C ) & "|"** *удаляет пробелы слева*  
|Hello | *это результат*

? **Val("rwq123")**  
0 *-это результат*

? **Val(" 123")**  
123 *-это результат*

? **Val(" 123.7fds")**  
123,7 *-это результат*

*Функция VAL игнорирует любое количество пробелов слева и преобразовывает строку в число до тех пор, пока не встретится символ, который не может быть частью числа.*

*Между соседними цифрами м.б. любое количество пробелов.*

*Дробное число должно иметь десятичную точку в аргументе.*

### Задание № 8.

Исследуйте работу функций, приведенных в Таблице № 4.1, которые не были рассмотрены в примерах.

*Примечание.* Для исследования функций **Lset** и **Rset** сначала напишите процедуру - **qq()**, необходимую для объявления фиксированной длины строки. Это связано с тем, что для этих функций нужно знать, где у строки правый край, а где левый.

Более того, при выполнении процедуры необходимо помнить то, что локальные переменные "умирают" при выходе из подпрограммы, поэтому их значения необходимо анализировать до момента окончания процедуры. Для этого процедуру останавливают до момента ее окончания, используя оператор **Stop**.

Длину строки задает оператор **Dim**. А так как в окне отладки оператор **Dim** не работает и длину строковой переменной не задать, приходится это делать в «фиктивной» процедуре **qq()**.

! *Заметим*, что эти используемые переменные и процедура пригодятся при исследовании функций логических проверок.

Ниже приведен текст, который следует набрать в модуле редактора VBA.

*'Объявления переменных (3 строки) сделайте в разделе деклараций модуля.*

```
Dim D As String * 10  ' объявляем D строковой переменной, длиной
10 символов
```

```
Dim D01 As String * 10, D02 As String * 10
```

```
Dim D3 As Variant, D4 As Object, DD(10) As Integer, D1 As Date
```

```
Sub qq()
```

```
  '-----
```

```
  D = "123456789012345"
```

```
  D1 = Day(Now)
```

```
  D3 = Null
```

```
  LSet D01 = "qaz"
```

```
  RSet D02 = "qaz"
```

```
  Stop  ' Это замечательный оператор. Он останавливает, но не
        прерывает программу. А это позволяет проверить значения
        переменных и использовать их в окне отладки.
```

```
End Sub
```

После набора кода программы запустите ее и покажите, чему равно **D1**.

Ответьте, как сделать так, чтобы **D1** было числом?

### 2.3. ИССЛЕДОВАНИЕ РАБОТЫ ФУНКЦИЙ ИЗ КАТЕГОРИИ «ДАТА И ВРЕМЯ».

#### Задание № 9 .

Перечертите таблицу 2.2.

Исследуйте работу приведенных в ней функций.

Повторите инструкции, приведенные на рисунке 2.7.

Учтите, что функции *Day()*, *Month()*, *Weekday()*, *Year()*, *Hour()*, *Minute()* используют в качестве аргумента последовательный номер даты так, как это показано в примере, на рисунке 2.7.

Таблица 2.2 Функции даты и времени

№ п.п.	Функция	Описание
1	Date	Устанавливает или возвращает текущую дату
2	Time	Устанавливает или возвращает текущее время
3	Now	Возвращает текущие дату и время
4	DateSerial	Преобразовывает в последовательную дату три целых числа: год, месяц, день
5	Day	Преобразовывает последовательную дату в день месяца
6	Month	Преобразовывает последовательную дату в месяц года
7	Weekday	Преобразовывает последовательную дату в день недели
8	Year	Преобразовывает последовательную дату в год
9	Hour	Преобразовывает последовательную дату в часы дня
10	Minute	Преобразовывает последовательную дату в минуты в часе

Задание № 11. Исследуйте функции, производящие вычисления над датами: *Timer*, *DateAdd* и *DateDiff*.

Синтаксис функции *DateDiff*, вычисляющей разницу между *Датой1* и *Датой2*:

**DateDiff(interval, Дата1, Дата2)**

где: первый аргумент - *interval* в *DateAdd* и *DateDiff* может принимать код, который следует указывать в кавычках:

“Yyy” – Год, “Q” - Квартал, “m” - Месяц, “ww” - Неделя,  
“H” - Часы, “N” - Минуты, “S” - Секунды.

```

Проверка
? Now
26.08.99 16:44:40

? DateSerial(99,10,18),DateSerial(1999-5,10-3,18+5)
18.10.99      23.07.94
18.10.99      23.07.94
'Использует 3 аргумента: год, месяц,день, причем строго в указанном порядке.
'С аргументами можно производить арифметические действия

? DateValue("1 февраль 1973"),DateValue("13.03.1950")
01.02.73      13.03.50
? DateValue("30 августа"),DateValue("24-авг-99"), DateValue("Сентябрь 1 1999")
30.08.99      24.08.99      01.09.99
' Переводит аргумент-строку в дату. Правильно обрабатывает все допустимые даты

?Day(Now), Day(DateSerial(99,10,18)),Day(DateValue("30 августа"))
26            18            30

```

Рисунок 2.7 –Проверка работы функций времени

✓ Повторите примеры выполнения некоторых таких функций, при веденных ниже.

```

? Timer          'Возвращает число секунд прошедших после полуночи
61290,47

? Now
26.08.99 16:56:27

?DateAdd("N",15,Now) 'Добавляет или вычитает заданный временной интервал из даты
26.08.99 17:22:02  '1-й арг. - строка,указывающая тип добавляемого временного интервала
                   '2-й арг. - число временных интервалов
                   '3-й арг - дата к которой добавляют (вычитают) временной 2-й арг.

```

- ✓ Посчитайте сколько секунд, минут, часов, дней, месяцев Вы прожили от рождения до сего момента.
- ✓ Подсчитайте сколько секунд и минут осталось до полуночи.
- ✓ Определите каким днем недели будет день вашего рождения через 1 год, 10 лет (используйте функцию DateAdd)

## 2.4. ИССЛЕДОВАНИЕ РАБОТЫ ЛОГИЧЕСКИХ ОПЕРАТОРОВ

**Задание № 12.** Исследуйте операторы логического сравнения, приведенные в таблице № 2.3. Пример такого исследования приведен ниже на рисунке 2.8.

? 1=1,	1=2,	1>2,	1<2,	2<=2
True	False	False	True	True
? "qaz" >"qaz1",	"qaz"="qaz"			
False	True			
? StrComp("qaz","qaz",1)	, StrComp("qaz","qaz",0)			
0	0			
? StrComp("qaz","wsx",1)	, StrComp("qaz","wsx",0)			
-1	-1			
? "Вася" Like "В*"	, "Вася" Like "Ba??"	, "Вася" Like "Ba??"		
True	True	True		
? "Вася" Like "Ba[нс]я"	, "Вася" Like "Ba[лера]?"			
True	False			

Рисунок 2.8 – Пример выполнения логических операторов

Таблица № 2.3 Логические операции

№ п.п	Оператор	Описание
1	=	Равно
2	><	Не равно
3	<	Меньше
4	>	Больше
5	<=	Меньше или равно
6	>=	Больше или равно
7	Is	Идентично (только для объектов). Возвращает True, если две переменные ссылаются на один и тот же объект.
8	Like	Операция, позволяющая использовать при сравнении строк подстановочные символы. <ul style="list-style-type: none"> <li>■ * - любое кол-во символов, ? - любой символ, # -любая цифра,</li> <li>■ [список символов] - любой символ из списка,</li> <li>■ [! список символов] - любой символ не из списка</li> </ul>
9	StrComp	Сравнивает 2 строки. Имеет 3 аргумента. Синтаксис: StrComp(строка1, строка2,код_сравнения). Если строка1> строка2, то возвращает +1, если строка1=строка2, то возвращает 0, если строка1 строка2, то возвращает -1. Цифровой код код_сравнения определяет двоичное (0) или текстовое (1) сравнение.

## 2.5. ФУНКЦИИ ЛОГИЧЕСКИХ ПРОВЕРОК

Функции, приведенные в таблице № 2.4 служат для проверки: относится ли выбранная переменная к величинам нужного вам типа и возвращают *True*, если это так, и *False* в противном случае.

### Задание № 13.

✓ Перед исследованиями в окне "*Проверка*" сначала запустите процедуру *qq()*, приведенную выше в задании № 9. Объясните зачем ?

✓ Что будет, если в общую область модуля (проекта) вставить оператор *Option Explicit*?

✓ Проверьте работу функций, представленных в таблице 2.4.

Таблица №2.4 Функции логических проверок

№ п.п.	Функция	Описание	Примечание
1	IsArray()	Возвращает True, если аргумент - массив.	
2	IsDate()	Возвращает True, если аргумент - дата.	
3	IsNull()	Возвращает True, если аргумент - Null.	
4	IsNumeric	Возвращает True, если аргумент - число	
5	IsObject()	Возвращает True, если аргумент - объект.	
6	IsMissing()	Возвращает True, если аргумент опущен.	Проверяет передан ли аргумент процедуре?

Примеры исследований в окне "Проверка" должны иметь вид:

? D

1234567890

? D1

15.01.1900

? IsArray(D), IsArray(DD), IsNull(D3)

False True True

? IsDate(D), IsDate(D1), IsNumeric(DD), IsObject(D4)

False True False True

## 3. РЕДАКТОР VBA

### 3.1. ИНТЕГРИРОВАННАЯ СРЕДА РАЗРАБОТКИ ОБЩИЕ ПОЛОЖЕНИЯ

Разработка программного обеспечения на VBA для работы с приложениями в Office практически полностью выполняется в редакторе VBA. Для приложений Microsoft Excel, Word, PowerPoint редактор представляет полноценную *интегрированную среду разработки (IDE)*, позволяющую создавать, редактировать, отлаживать, запускать программы, связанные с документами Office

Работа в редакторе предполагает, что Вы создаете *проект* программного продукта, который включает и модули с программным кодом, и модули классов, и пользовательские формы, и отдельные листы и документы и др., то есть все то, к чему в рамках данной разработки пользователь имеет доступ.

Для вызова редактора VBA следует после запуска основного приложения (например, Excel) выполнить команду меню: *Разработчик/Код/Visual Basic*, или нажать горячие клавиши <Alt+F11>. В общем случае редактор VBA включает большое количество окон, панелей инструментов, которые могут быть, при желании, убраны или добавлены в пользовательский интерфейс. Один из его вариантов может иметь вид, представленный на рисунке 3.1.

Для открытия окон редактора VBA выполните соответствующие команды из его меню - *Вид\Окно проверки (View\Immediate Windows)*, или *Вид\Окно проекта (View\Project Explorer)*. Вместо этих команд можно использовать горячие клавиши, которые обозначены в меню рядом с командой. Например, для *открытия окна проекта* - <Ctrl + R>, а *окна свойств* – F4 (при этом в последнем выводятся свойства активного объекта).

Используя команды меню *Вставка (Insert)*, можно добавлять в проект (окно проекта) *Формы (User Form)*, *Модули (Module)*, *Модули класса (Class Module)*, а в окно с программным кодом *процедуры (Procedure)* или *файл (File...)*.

Для активизации (вывода) на экран *окна с программным кодом (окна модуля (Module)* следует дважды щелкнуть мышью в окне проекта по имени того приложения, для которого необходимо создать какую-либо программную единицу. Аналогично можно вызвать окно кода для любой экранной формы или ее элемента управления.

Примечание. Щелчок правой кнопки мыши по элементам интерфейса (окнам, формам, кнопкам и т.д.) выводит контекстно-связанные с этими элементами меню.

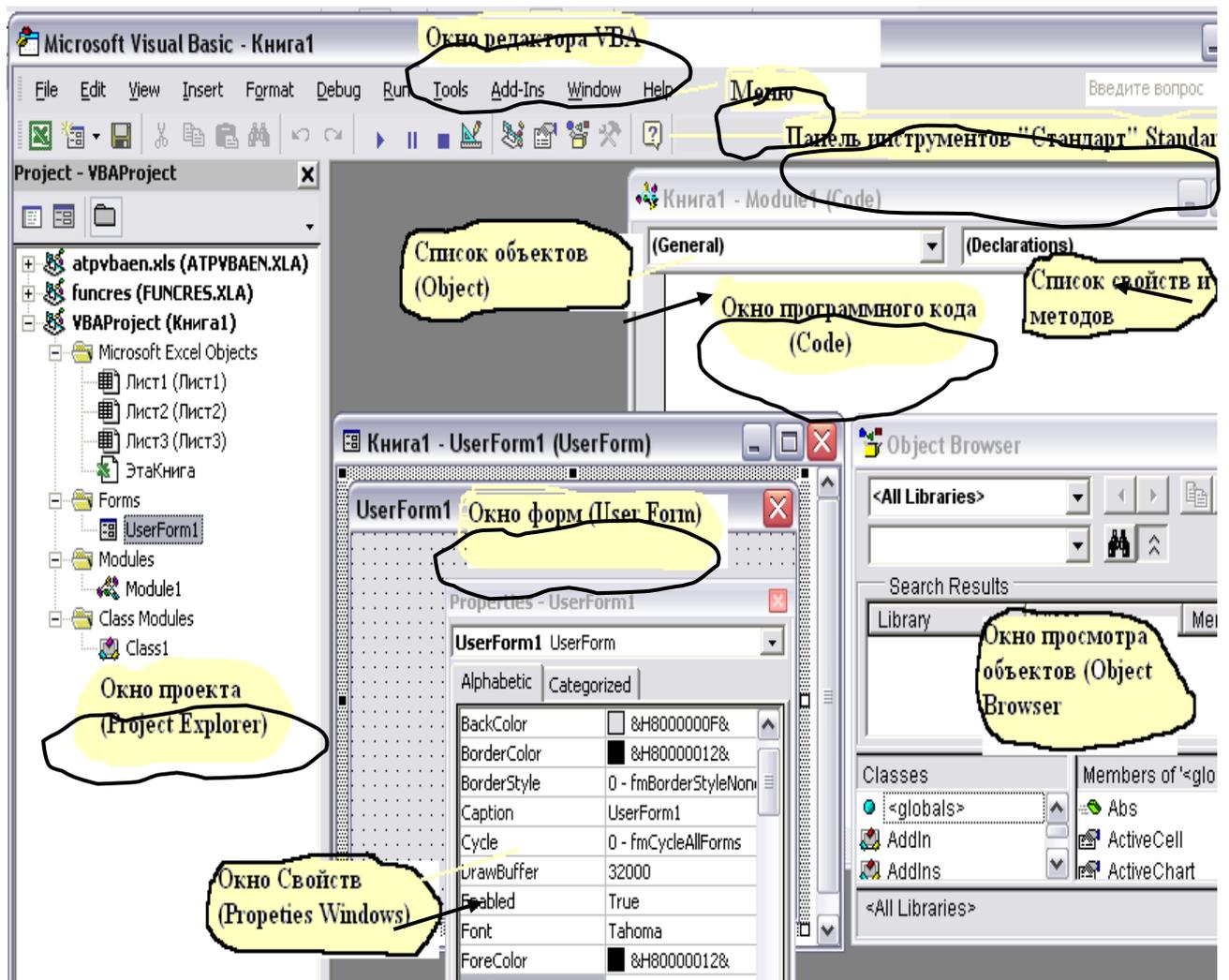


Рисунок 3.1 – Основные окна редактора VBA

**Задание № 1.** Научитесь вызывать и закрывать редактор VBA; открывать окна, изменять их размеры, двигать по экрану, открывать панели инструментов; вводить программный код; изменять свойства элементов, используя окно свойств.

## 3.2. ОКНА РЕДАКТОРА VBA

### 3.2.1. Окно проекта (Project explorer)

Для скрытия окна используйте кнопку -  или команду *Hide* (*Скрыть*) из контекстного меню.

Если из этого же меню выполнить команду *Dockable* (*Закрепить*), то окно всегда будет располагаться поверх других окон, и вместо 3-х кнопок в его правом верхнем углу останется одна.

Используя команду *Remove...* (*Удалить*) из контекстного меню, можно удалять формы, классы и модули.

Щелкнув по объекту в окне проекта, вы активизируете его, а двойной щелчок по объекту из окна проекта вызывает, как правило, окно модуля (программного кода) с набором процедур написанных для этого объекта.

Для вывода информации «ветвей дерева» в окне проекта щелкните по символу "+" около имени ветви. Щелчок по символу " – " скрывает информацию ветвей.

### 3.2.2. Окно модуля

Это окно используется для создания текста любой программы. Его особенностью является присутствие двух раскрывающихся списков под строкой заголовка окна. В левом выводятся все объекты модуля (см. рисунок 3.2). В данном случае из них выбран Worksheet. В правом списке выводится набор процедур (событий), связанных с данным объектом. При щелчке мышью по имени одной из них (у нас это Activate) в окне модуля появляются ее первая и последняя строки (такая процедура называется - пустой процедурой), и вам остается только написать тело процедуры.

Редактирование текста программного кода происходит точно так же, как и в любом текстовом редакторе.

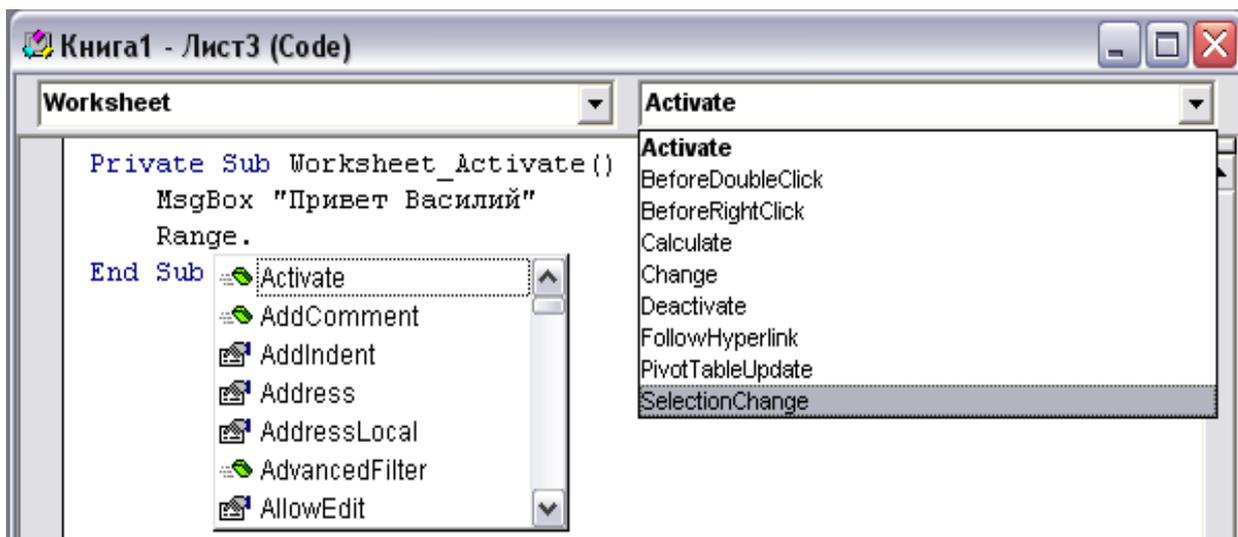


Рисунок 3.2 – Вид окна кода (модуля)

*Процесс работы по написанию кода в VBA автоматизирован.*

✓ Добавление пустых процедур и функций пользователя происходит по команде меню *Вставка\Процедура (Insert Procedure...)*. При этом, используя появившееся рабочее окно, можно задать необходимые параметры процедур и функций.

✓ При описании переменных (например, переменной *qq*), с использованием конструкций вида - *Dim qq As*, редактор предложит список их допустимых типов сразу после набора слова *As*.

✓ В том случае, если нужно вставить в текст программы метод или свойство, можно обратиться к команде основного меню редактора - *Правка\*

*Список Свойств\Методов (Edit\List Properties/Methods)*, или к команде контекстного меню окна кода (щелкните правой кнопкой мыши в любом его месте) - *Список Свойств\Методов (List Properties/Methods)*.

✓ При вводе имени объекта и точки после него редактор отобразит список его методов и свойств так, как это отображено на рисунке.5.2 для объекта Range. Указанный режим будет действовать в том случае, если в диалоговом окне *Параметры* (команда вызова из меню – *Сервис\Параметры \Редактор (Tools\Options...\Editor)*) предварительно был поднят флажок *Список компонентов (Auto List Members)*.

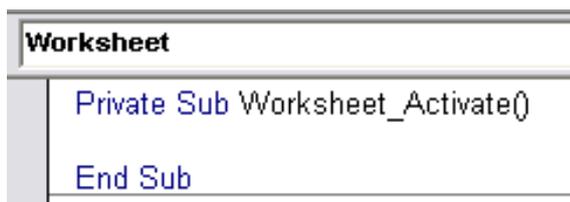
✓ Команда меню *Вид\Просмотр объектов (View\Object Browser)* или горячая клавиша **F2** вызывают Окно просмотра объектов (см. рисунок 3.1). С помощью этого окна можно определить к какой из библиотек принадлежит объект, посмотреть его свойства и методы, уточнить значения констант.

*Проанализируем, что же можно узнать из информации на рисунке 3.2.*

Во-первых, код программы создается для объекта «*Лист3*», который вложен в объект «*Книга1*». Об этом сообщает надпись в заголовке окна кода -«*Книга1 – Лист3(Code)*». В терминах VBA на объект – рабочий лист указывает и имя выбранного объекта в левом раскрывающемся списке – *Worksheet*.

Во-вторых, в правом раскрывающемся списке, который на рисунке открыт и содержит методы и свойства объекта *Worksheet*, выбран метод *Activate*.

В-третьих, как только такой выбор будет сделан, в окне кода редактор сделает заготовку для программы обработки события – активизировать *Лист3* – первую и последнюю инструкции (рисунок 3.3).



```
Worksheet
Private Sub Worksheet_Activate()
End Sub
```

Рисунок 3.3 – Пустая процедура

Осталось заполнить инструкциями тело процедуры и все.

Первая команда (рисунок 3.2) выводит на экран окно сообщения с надписью: «Привет Василий»

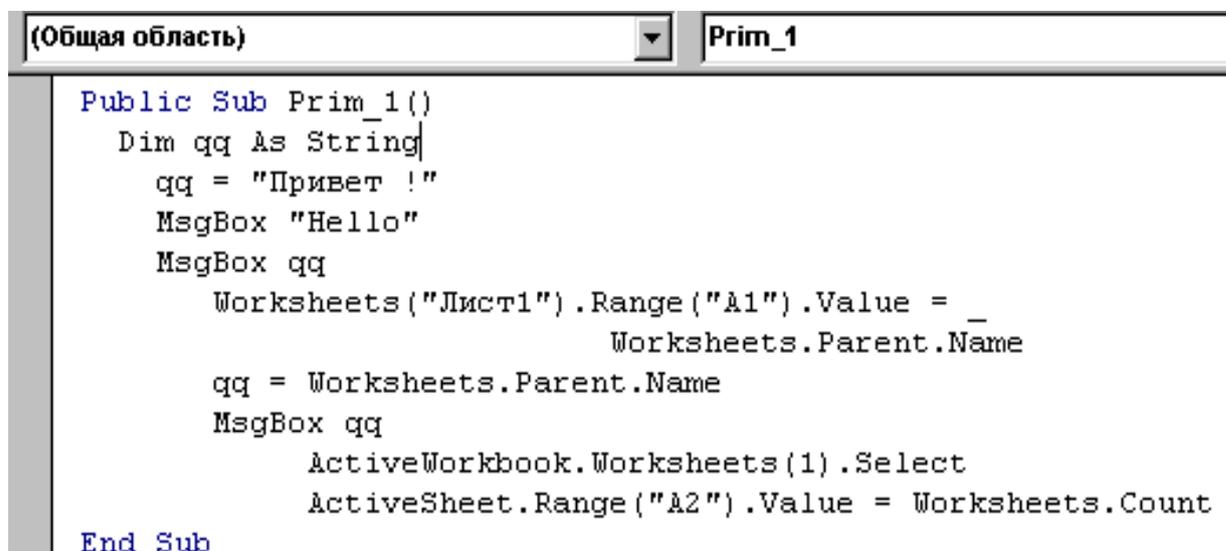
Вторую команду нужно несколько исправить. В том виде, в каком она представлена, конкретная ячейка рабочего листа, на которую должен ссылаться объект Range, не определена. Методы или свойства объекта Range также не указаны. Сделаем так, чтобы в ячейку «A3» рабочего листа записать информацию. Команда может иметь такой вид:

`Range("A3").Value="1234567"`

**Задание №2.** Наберите текст рассмотренной выше процедуры, исправив второй оператор. Перейдите на первый или второй рабочий лист. Активизируйте Лист3. Проанализируйте результат.

**Задание №3.**

А) Наберите текст процедуры, приведенный на рисунке 3.4.



```
Public Sub Prim_1()  
    Dim qq As String  
    qq = "Привет !"  
    MsgBox "Hello"  
    MsgBox qq  
    Worksheets("Лист1").Range("A1").Value = _  
        Worksheets.Parent.Name  
    qq = Worksheets.Parent.Name  
    MsgBox qq  
    ActiveWorkbook.Worksheets(1).Select  
    ActiveSheet.Range("A2").Value = Worksheets.Count  
End Sub
```

Рисунок 3.4 - Вариант рабочей программы для задания №3

Набирая текст, обратите внимание на максимально возможную автоматизацию ввода текста программы. Ответьте где здесь объекты, свойства, методы, что они обозначают и (или) выполняют.

В) Запустите процедуру на выполнение. Для этого курсор мыши поставьте на какой-нибудь оператор программы и выполните команду меню *Run/Run Macro* или щелкните на функциональную клавишу **F5** на клавиатуре.

С) Попробуйте выполнить программу в пошаговом режиме. В этом случае система будет ждать команду на выполнение очередного оператора. Такой командой является инструкция меню редактора VBA *Debug/StepInto* или щелчок по клавише **F8** на клавиатуре.

Если при выполнении программы необходимо уточнить значение какой-нибудь переменной поставьте курсор мыши на эту переменную в тексте программы или воспользуйтесь окнами просмотра или отладки.

В том случае, если требуется закончить программу, не дожидаясь выполнения всех операторов (прервать программу), щелкните мышью по кнопке **Reset**.



Д) Поэкспериментируйте!

Предложите свои варианты. Попробуйте все то, о чем говорилось выше.

### 3.2.3. Окно форм и окно свойств

В проект можно вставить любое количество форм. Каждая форма вставляется командой меню редактора VBA *Вставка\Форма (Insert\UserForm)* Отдельные элементы управления: кнопка, поле ввода и другие, вставляются в форму перетаскиванием их с *панели инструментов (ToolBox)*.

Каждый элемент, размещенный на форме, обладает свойствами, отображаемые в *окне свойств*, которые могут быть при необходимости изменены. *Окно свойств (Properties Window)* открывается командой меню *Вид\Окно свойств (View\ Properties Window)* или горячей клавишей **F4**.

Изменение элементов в размерах и их перемещение в пределах формы производится обычным для Window образом (активизируйте и тащите или изменяйте мышью). Для выделения нескольких элементов одновременно щелкните последовательно по ним мышью, удерживая нажатой кнопку *Shift*.

Выделяя несколько элементов можно добиться желаемого размещения с помощью команд из меню, сгруппированных под общей командой *Формат*.

#### Задание №4.

**А)** Разместите на форме 4 кнопки. Покажите, как работают команды из пункта меню *Формат*. Измените свойства установленных элементов, используя *окно свойств*.

С каждым объектом связывают код. Причем, для обработки каждого события, связанного с конкретным объектом, это своя оригинальная программа, написанная в окне модуля (кода).

Вызов окна модуля для элементов формы производится двойным щелчком мыши по форме или нажатием горячей клавиши **F7**. Выбор объекта и события для создания процедуры обработки этого события производится так, как это было рассмотрено выше.

**В)** Попробуйте для нескольких кнопок написать программы по обработке события –щелкнуть мышью по кнопке – *CommandButton1\_Click* (здесь цифра один – это номер кнопки), например:

- щелчок по первой кнопке – должен изменить надпись на третьей кнопке (свойство –*Caption*);
- щелчок по второй кнопке должен информацию, помещенную в ячейке «A4» первого листа записать в ячейку «B2» второго листа.

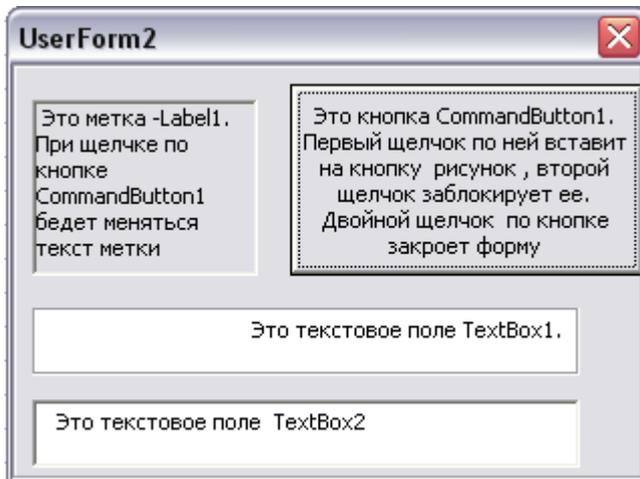
### 3.3. СОЗДАНИЕ ПРОСТЕЙШИХ ПРОГРАММ И ЭКРАННЫХ ФОРМ

#### Задание №5. (см. рисунок 3.5)

✓ Создайте форму с двумя текстовыми полями (полями ввода), одной меткой и кнопкой.

✓ Отформатируйте форму, и ее элементы так, как Вам хочется. На кнопку поместите рисунок.

- ✓ С событиями – изменение текстового поля 2 и щелчок по кнопке свяжите процедуры, текст которых приведен ниже (см. рисунок 3.6).
- ✓ Для запуска формы, для ее активизации – щелкните мышью по любой точке формы, после чего, нажмите кнопку **F5** на клавиатуре.
- ✓ Введите что-нибудь в текстовые поля.



- ✓ Пощелкайте мышью по кнопке.
- ✓ Ответьте и проверьте, что будет, если вместо ключевого слова **Static** ввести **Dim**.
- ✓ Сделайте выводы и обоснуйте ответы.
- ✓ Попробуйте создать свои варианты формы.

Рисунок 3.5 - Пример для выполнения задания №5

```

CommandButton1  Db1Click

Private Sub CommandButton1_Click()
    Static j As Integer
    j = j + 1
    If j = 1 Then
        'Изменяем программно рисунок на кнопке и текст метки
        CommandButton1.Picture = LoadPicture("D:\MSOffice\Clipart\Cat.wmf")
        Label1.Caption = " Вы нажали кнопку 1 раз, нажмите еще"
    Else
        CommandButton1.Enabled = False
        Label1.Caption = " Привет коллега ! Теперь кнопка заблокирована."
    End If
End Sub

Private Sub CommandButton1_Db1Click(ByVal Cancel As MSForms.ReturnBoolean)
    Unload Me          'Закрываем форму
End Sub

Private Sub TextBox2_Change()
    Static i As Integer
    i = i + 1
    TextBox1.Value = " Во втором текстовом поле - изменение № " & CStr(i)
End Sub

```

Рисунок 3.6 – Текст программы для задания №5

### 3.4. ЗАДАНИЕ ПАРАМЕТРОВ РЕДАКТОРА

Для того, чтобы задать параметры редактора необходимо выполнить команду *Сервис\Параметры(Tools\Options.)* и в появившемся окне диалога на соответствующих вкладках установить их. Помимо явно понятных параметров, отметим следующие:

#### ***Вкладка редактор:***

- ◆ *Проверка синтаксиса (Auto syntax Check).*
- ◆ *Явное описание переменных (Require VariableDeclaration)* – вставляет оператор *Option Explicit*, требующий явного описания переменных, во вновь создаваемом проекте.
- ◆ *Список компонентов (Auto List Members)* – автоматически отображает список данных, логически завершающих инструкцию, расположенных вместе ее вставки.
- ◆ *Краткие сведения (Auto Quick Inform)* – после ввода имени процедуры отобразит сведения о них и их параметрах.
- ◆ *Подсказки значений данных (Auto Data Trips)* – при установке курсора на переменную выводится ее значение. Работает только в режиме прерывания (отладки программ).
- ◆ *Перетаскивание текста (Drag and Drop Text Editing)* – определяет, возможно ли мышью перемещать и копировать выделенные куски текста в программах.
- ◆ *Просмотр всего модуля (Default to Fill Module View)* – определяет можно ли просматривать процедуры в модуле все сразу или по одной.

#### ***Вкладка общие(General):***

- ◆ *Всплывающие подсказки (Show ToolTips)* – возникают около кнопок панелей инструментов.

#### **Задание №6.**

- ✓ Создайте новый проект.
- ✓ Получите информацию о параметрах редактора, щелкнув по кнопке *Справка*, которая находится на каждой вкладке.
- ✓ Попробуйте изменить установки. Проверьте, что изменилось.
- ✓ Используя команду *Сервис\Свойства VBA Project\ Защита (Tools\VBAProject Properties\Protection)* попробуйте защитить свой проект от несанкционированного вмешательства.

#### **Задание №7.**

Ниже представлен вариант программы, состоящий из 4-х процедур. Основная процедура- main запускает в соответствующем порядке 3 остальные процедуры.

Требуется заполнить таблицу, представленную справа, поместив в нее значения переменных-  $a$ ,  $b$ ,  $c$ . Значения переменных фиксируем в момент окончания соответствующей процедуры.

После заполнения таблицы, покажите ее преподавателю. Далее наберите текст программы на ПК в окне программного кода и запустите *main* на выполнение. В пошаговом режиме проследите, как изменяются значения переменных и их соответствие тем значениям, которые записаны вами в таблице. Объясните все несоответствия, если они есть.

The screenshot shows a Visual Basic IDE window with the following code in the 'General' tab:

```

Public a As Integer
Dim b As Integer
Sub qq1()
Static c As Integer
a = a + 1: b = b + 2: c = c + 3
End Sub
Sub qq2()
Dim b As Integer
a = a + 3: b = b + 4: c = c + 5
End Sub
Sub qq3()
Static a As Integer
Dim c As Integer
a = a + 7: b = b + 6: c = c + 9
End Sub
Sub main()
a = 1: b = 2: c = 3
qq1: qq2: qq1: qq3: qq3: qq2: qq1
c = a + b + c
End Sub
    
```

To the right of the code is a table with the following structure:

Процедура	Значения переменных при выходе из процедуры		
	a	b	c
main	1	2	3
qq1			
qq2			
qq1			
qq3			
qq3			
qq2			
qq1			
main			

Рисунок 3.7 – Программа для исследования области видимости переменных.

## 4. СОЗДАНИЕ ФУНКЦИЙ, ПРОЦЕДУР И ИХ ОТЛАДКА. ПРИМЕНЕНИЕ ЦИКЛОВ И ПОВТОРЯЕМЫХ СТРУКТУР

### 4.1. ОСНОВНЫЕ ПОЛОЖЕНИЯ

Для создания и отладки процедур и функций в Excel, а также других приложениях используют редактор VBA. Он имеет свои панели инструментов и меню. Вызовите редактор VBA и убедитесь в этом. Многие основные команды повторяются в головном меню и панелях инструментов, и Вы можете использовать их так, как удобнее.

Основные из тех команд, которые Вам понадобятся при отладке и запуске процедур представлены на рисунке 4.1

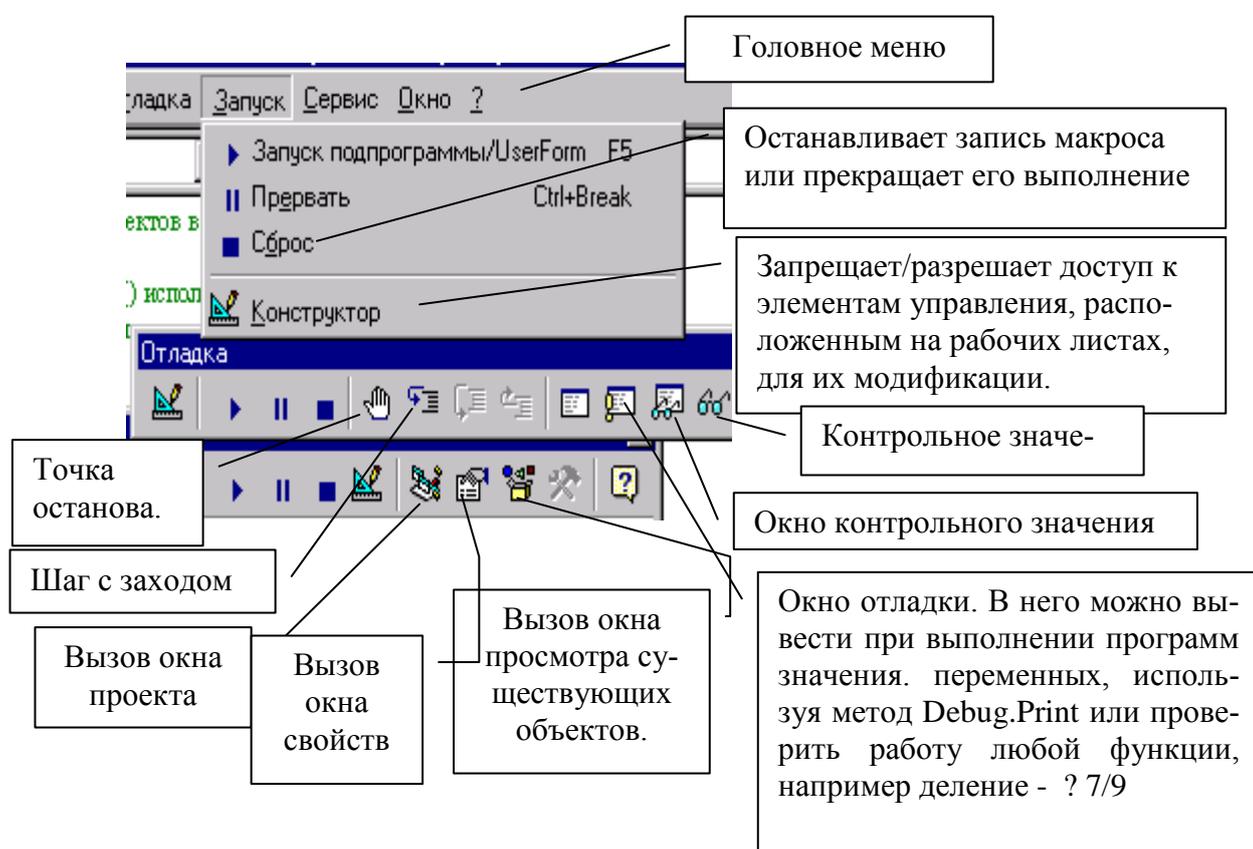


Рисунок 4.1 - Вид меню и панели «Отладка» редактора VBA.

Кнопки «*Запуск программы*» и «*Шаг с заходом*» вызывают выполнение процедуры. Причем, во втором случае в пошаговом режиме. Все команды данного макроса будут выполняться по очереди. В этом режиме для запуска очередной команды удобно использовать горячую клавишу **F8**.

Кнопка "**Точка останова**" определяет место в процедуре, где необходимо остановить ее выполнение. Повторный щелчок по кнопке выключает данный режим.

Кнопка "**Контрольное значение**" позволяет вывести имена переменных и их значения в окно "**Контрольного значения**". Для этого выделите в исходном тексте процедуры переменную, информацию о которой Вы хотите получить, и перетащите ее мышью в указанное окно.

## 4.2. ЗАПУСК И ОТЛАДКА ПРЦЕДУР

### Задание №1.

Внимательно ознакомьтесь с текстом исходной программы.

Предложенная процедура использует вычисляемый цикл *-For-Next* и заполняет текущий выбор (выделенную область ячеек) случайными числами. Ее исходный текст с подробными комментариями приведен ниже.

1. Наберите текст этой программы в окне модуля (оно появится после выполнения команды меню "**Вставка/Модуль**")

Option Explicit

' Запись случайных чисел в текущий выбор (выделенную область ячеек).

'=====

' Метод Rows возвращает набор всех строк в текущем выборе.

' Метод Columns возвращает набор всех столбцов в текущем выборе.

' Свойство Count возвращает количество объектов в текущем выборе.

,

' Метод Cells, свойство Value и функция Rnd() используются

' для записи случайного числа в текущую ячейку

,

Sub QQ\_Rand()

Dim numRows As Integer, numCols As Integer

Dim theRow As Integer, theCol As Integer

'определение размера текущего набора

numRows = Selection.Rows.Count       ' количество строк

numCols = Selection.Columns.Count     ' количество столбцов

Randomize       ' инициализация генератора случайных чисел.

                  'Rnd() - функция возвращающая случ.величину.

For theRow = 1 To numRows             ' цикл заполнения строк

    For theCol = 1 To numCols         ' цикл заполнения столбцов

        Selection.Cells(theRow, theCol).Value = Rnd

    Next theCol

Next theRow

End Sub

2. Запустите процедуру *QQ\_Rand* и на рабочем листе "Лист1" получите картинку, изображенную на рисунке 4.2. Для этого существует несколько вариантов. Рассмотрим их.

	A	B	C	D	E
1					
2		0,45671	0,833824	0,037694	
3		0,83691	0,894278	0,294622	
4		0,563746	0,783413	0,730796	
5					

Рисунок 4.2 – Результат работы программы *QQ\_Rand*

Первый вариант. Активизируйте рабочий лист "Лист1", выделите область ячеек, которую хотите заполнить случайными числами, и выполните команду меню: *Сервис/Макрос/Макросы...* . В появившемся диалоговом окне "Макрос" выберите процедуру *QQ\_Rand* и щелкните по командной кнопке "Выполнить".

Второй вариант. Вызовите панель инструментов "Visual Basic" (команда меню - *Вид/Панели инструментов/Visual Basic*), и щелкните по кнопке "Выполнить макрос". Находясь на рабочем листе и выделите необходимую область. Остальное аналогично.

Третий вариант. Выполните ту же процедуру, но в режиме выполнения операторов по одному -*по шагам*.

Для этого, находясь в "Окне Модуля" щелкните по какой-либо команде процедуры *QQ\_Rand*, а потом по кнопке  или клавише *F8* на клавиатуре.

Каждый щелчок по кнопке "*Шаг с заходом*" будет приводить к выполнению очередной команды макроса.

Получите в "*Окне контрольных значений*" информацию о переменных процедуры. Добейтесь того, чтобы это окно имело вид, представленный на рисунке 4.3.

Expression	Value	Type	Context
Rnd	0,8195155	Single	Module1.QQ_Rand
numCols	4	Integer	Module1.QQ_Rand
numRows	5	Integer	Module1.QQ_Rand
theCol	2	Integer	Module1.QQ_Rand
theRow	1	Integer	Module1.QQ_Rand

Рисунок 4.3. - Вид окна контрольных значений

Выполните последовательно операторы процедуры, щелкая каждый раз по кнопке "**Шаг с заходом**". При этом контролируйте изменения значений переменных и ход выполнения макроса.

### 4.3. СОЗДАНИЕ ПОЛЬЗОВАТЕЛЬСКИХ ФУНКЦИЙ

#### 4.3.1. Определения и особенности создания пользовательских функций

Все программные единицы делят на процедуры и функции. Функции имеют следующий синтаксис:

```
[Public / Private] [Static] Function Имя_функции [(Список_аргументов)] [As Тип]  
    Тело функции  
End Function
```

Ключевые слова **Function**, **End Function** и *Имя\_функции* являются обязательными параметрами.

*Тело функции* может включать любые допустимые операторы VBA и, в частности, оператор Exit Function, прекращающий выполнение функции и передачу управления вызываемой процедуре. Здесь же *Имя\_функции* получает значение, которое эта функция и будет возвращать.

Описывая функцию, можно задать ее *Тип*. Функцию можно сделать статической, используя описатель **Static**, указывая на то, что переменные между вызовами функции сохраняются.

Функция, описанная как **Private**, является доступной в программных единицах текущего модуля, как **Public** – всех модулей. Созданные таким образом функции можно вызывать из любых других процедур и функций. *Но если требуется создать пользовательскую функцию, то необходимо:*

- ✓ описать функцию как **Public**;
- ✓ программный код для функции поместить в окне модуля. Если модуль уже создан, то его можно активизировать из *Окна проекта*. Если модуля еще нет, то его следует создать командой меню редактора VBA – **Insert/Module**)

**Задание №2.** Учитывая сказанное, создайте пользовательскую функцию так, как это представлено на рисунке 4.4.

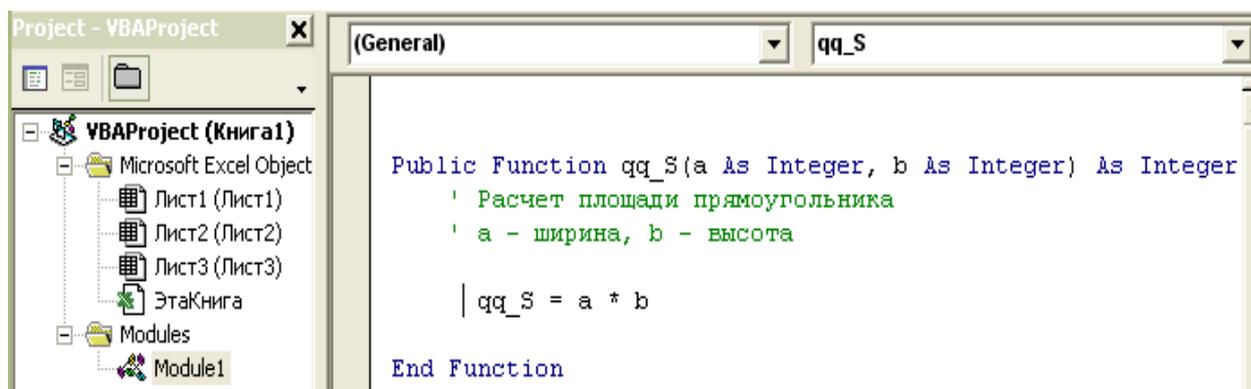


Рисунок 4.4- Окно проекта и модуля для пользовательской программы

При желании для написанной функции можно добавить комментарии. Для этого, находясь в редакторе VBA, выполните команду меню **Вид/Просмотр объектов (View/Object Browser)**. В появившемся окне, в раскрывающемся списке выбора **Проекта/Библиотек** активизируйте строку **VBAProject** - см. рисунок 4.5, ниже.

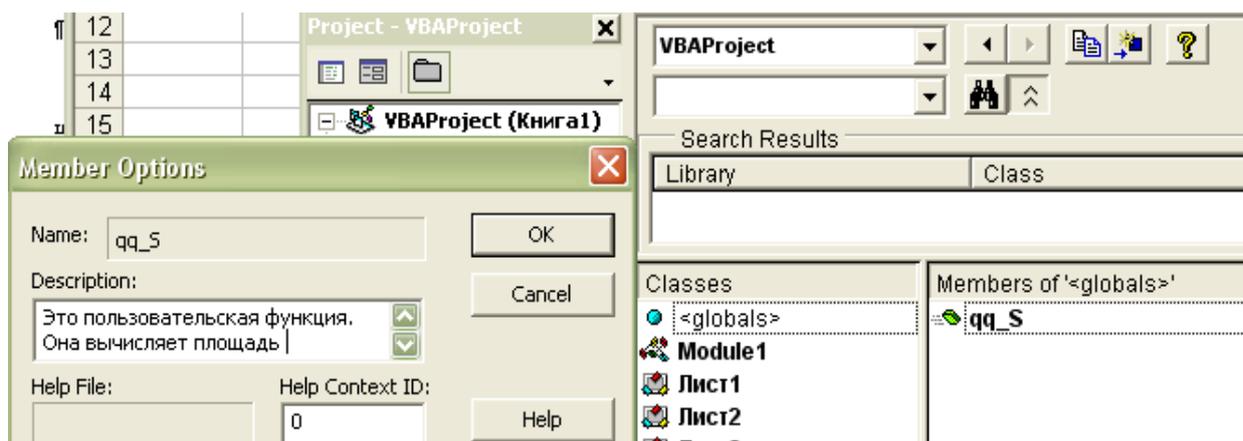


Рисунок 4.5 – Добавление комментариев в пользовательскую функцию

Щелкните правой кнопкой мыши по имени **qq\_S** вашей пользовательской функции в окне "**Компонент '<глобальные>' (Members of '<globals>')**", и выберите в контекстном меню команду "**Свойства (Properties)**", откройте окно "**Параметры компонента (Member Options)**". В этом окне в поле "**Описание (Description)**" введите поясняющий текст к Вашей функции.

Все созданные таким образом *функции* будут записаны в категорию "**Пользовательских функций**" или "**Определенных пользователем**". После этого **Мастер функций** будет работать с ними так, как он это делает со всеми встроенными функциями.

Проверьте работу созданной пользовательской функции. На рабочем листе, используя мастер функций, введите написанную функцию в какую-нибудь ячейку. В рабочем окне мастера появится поля для ввода аргументов, результат выполнения операции и поясняющий текст, как на рисунке 4.6.

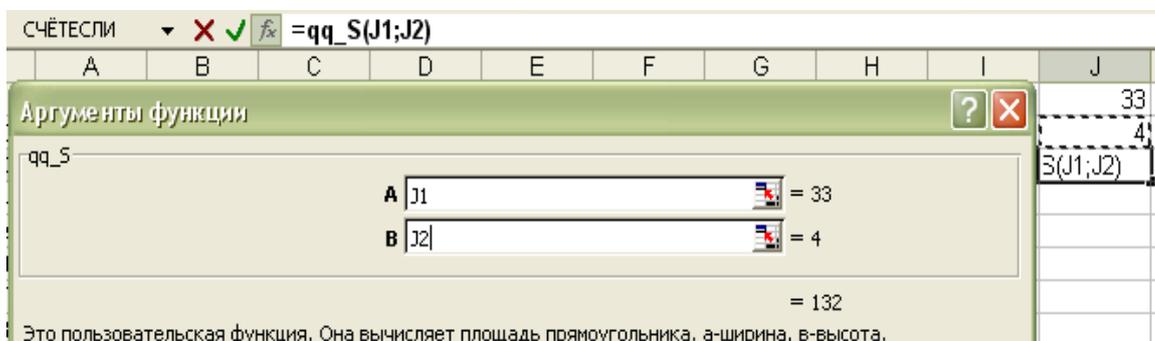


Рисунок 4.6 – Запуск пользовательской функции

#### 4.3.2. Вызов пользовательских функций для использования в формулах рабочего листа

**А. Самый простой способ** вызова предполагает использование мастера функций. Он рассмотрен в задании №2, выше. Категория «*Пользовательских функций*» будет доступна мастеру всякий раз, когда открыт файл (книга), в котором были созданы пользовательские функции. Если книгу с пользовательскими функциями закрыть, то в ячейках оставшихся открытых книг, там, где в формулах используются эти функции будет возвращено ошибочное значение *#ссылка*.

Для доступа к созданным функциям всех пользователей запишите свой файл с функциями в каталог *XLSTART*. В этом случае ваши функции станут доступны для любого пользователя, работающего с табличным процессором.

##### Задание №3.

✓ Попробуйте выполнить созданную пользовательскую функцию из той книги, в которой ее нет.

✓ Запишите книгу с созданной пользовательской функцией в каталог *XLSTART*. Закройте Excel. Откройте Excel. Создайте новую книгу и вызовите из нее пользовательскую функцию.

##### Задание №4.

Создайте собственную пользовательскую функцию для расчета подоходного налога. Проверьте ее работоспособность.

$$\text{Подох\_налог} = (\text{Зар\_плата} - \text{МРОТ} * (\text{Кол\_иждевенцев} + 1) - 1\% * \text{Зар\_плата}) * 12\%$$

### Задание №5.

Создайте пользовательскую функцию:

***Function Translat\_MesToNum(anMes As String) As Integer***

которая по имени месяца (например, Март) возвращает его номер (в данном случае это - 3).

Используйте структуру Select Case. Проверьте ее работоспособность.

### **4.3.3. Аргументы пользовательских функций**

Во всех функциях, созданных в данной работе, были использованы аргументы различного типа в количестве от одного до нескольких. Тем не менее, следует иметь в виду то, что:

- ✓ аргументы могут отсутствовать;
- ✓ аргументы могут быть обязательными (как в примерах выше), так и необязательными. Перед необязательными следует вставлять ключевое слово ***Optional***.
- ✓ количество аргументов может быть неопределенным. В этом случае перед именем аргумента нужно вставить ключевое слово ***ParamArray***.

### Задание № 6.

На рисунке 4.7 приведен текст пользовательской функции без аргументов и пример ее использования.

Создайте свою пользовательскую функцию, которая не использует аргументов, и покажите, что она работоспособна.

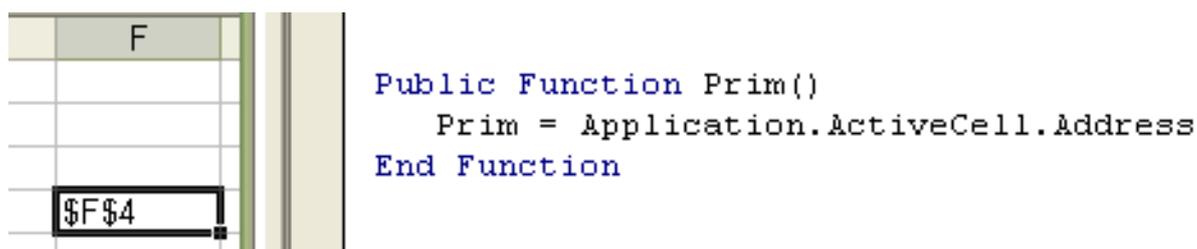


Рисунок 4.7 – Пользовательская функция без аргументов

### Задание № 7

Создайте функцию с необязательным параметром. В том случае, если этот параметр не получит никакого значения функция должна возвращать одно значение (например, имя Катерина), а если параметр задан – то другое значение (другое имя).

### 4.3.4. Использование функций рабочего листа в процедурах и функциях VBA

Полный перечень функций рабочего листа можно получить, открыв в интернете *List of Worksheet Functions Available to Visual Basic (список функций раб.листа доступных для VB)* или открыв в справке по Excel *Список функций рабочего листа (по категориям)*. Все эти функции можно использовать в том случае, если при обращении к ним перед именем функции вставить *WorkSheetFunction* и точку.

Для примера была выбрана функция *Large(массив, i)*. Функция возвращает *i-е* наибольшее значение из *массива* или *интервала данных* (диапазона рабочих ячеек). На рисунке 4.9 она использована для получения 2-х максимальных значения, хранящихся в пяти выделенных ячейках рабочего листа.

Очевидно, что если массив содержит *N* элементов, то *Large(массив,N)* вернет наименьшее значение.

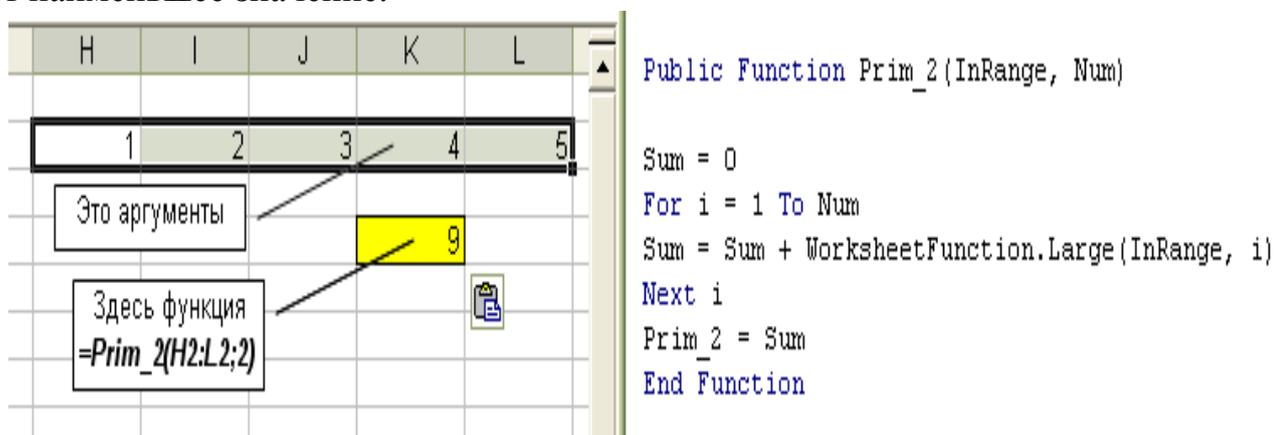


Рисунок 4.9 – Пример использования функции рабочего листа

#### Задание № 9.

1. Повторите пример, приведенный на рисунке 4.9.
2. Напишите функцию, которая бы возвращала сумму двух минимальных значений массива. Массив должен быть представлен набором выделенных ячеек Excel. В данной задаче нужно определить и размерность массива.
3. Напишите функцию, которая использует в своем теле любую другую функцию рабочего листа. Проверьте ее работоспособность.

## 4.4. СОЗДАНИЕ ПРОЦЕДУР

#### Задание № 10.

1. Создайте процедуру, название и формальные параметры которой представлены ниже.

***Sub Zam\_Simvol (Sim1 As String; Sim2 As String; \_  
Sim3 As String; Stroka As Variant)***

- ' Sim1 - число, Sim2 - месяц, Sim3 - год  
' из строки Stroka : 25 января 1998 г.

	A	B	C
1			
2		25 января 1998	
3			
4			25
5		января	
6			1998
7			

Указанная процедура должна получать строку, содержащую дату: число, месяц и год (**25 января 1998 г.**). Причем, число пробелов между словами и цифрами в исходной строке, а также перед первым числом и после последнего может быть любым. Задача, стоящая перед процедурой, заключается в том, чтобы выделить и вернуть

по отдельности три составляющие даты: число, месяц и год.

2. Проверьте созданную программную единицу. Для этого напишите еще одну, проверочную процедуру, которая должна:

- из первой ячейки (B2) считывать исходную строку в переменную *Stroka*,
- обращаться к процедуре *Zam\_Simvol*, которая данные переменной *Stroka* должна делить на три отдельные части (число, месяц и год) и присваивать их соответственно переменным *Sim1*, *Sim2*, *Sim3*.
- во 2-ю, 3-ю, 4-ю ячейки, и записывать результаты, возвращаемые проверяемой процедурой *Zam\_Simvol* (в ячейку B4 - число, B5 - месяц, B6 - год).

Примечание.

1. Для решения этой задачи подойдут строковые функции, представленные в таблице 2.1.
2. Совсем необязательно эту задачу решать предложенным способом. Используйте свои варианты.

**Задание № 11** Создайте процедуру для замены любого количества пробелов между символами одним символом табуляции.

Вместо символа табуляции используйте константу ***vbTab***.

Для замены используйте встроенную функцию ***Replace***.

Создайте и проверьте процедуру для случаев ее использования в ***Excel*** и ***Word***.

## 5. ВСТРОЕННЫЕ ДИАЛОГОВЫЕ ОКНА

### 5.1. ВВОД И ВЫВОД ИНФОРМАЦИИ. ФУНКЦИИ *MsgBox()* и *InPutBox()*

Используя всего две функции - *MsgBox()* и *InPutBox()*, можно создать простой пользовательский интерфейс для различных приложений. С их помощью можно указывать на ошибки пользователя и исправлять их, вводить информацию и др. Удобнее всего работать с этими функциями в интерактивном режиме.

Перед выполнением первого задания следует вспомнить синтаксис этих функций и допущения, накладываемые на их параметры. Для этого следует либо обратиться к лекциям, либо к соответствующей литературе.

#### 5.1.1. Поиск ошибок в процедуре

**Задание № 1.** Наберите в окне модуля исходный текст процедуры *ZapPrem()*.

#### *Sub ZapPrem()*

```
'=====
Dim theDefault As String, thePrompt As String, theTitle As String
Dim theReply As String, OKFlag As Boolean, theB As Single

thePrompt = " Hello, гражданин ! " & _
" Введите пожалуйста сумму премии за квартал с учетом копеек."
theDefault = "123456789"
theTitle = "Ввод суммы премии за квартал"
Do
  theReply = InputBox(thePrompt, theTitle, theDefault)
  If (theReply) = Then Exit Sub
  theReply = Trim(theReply)
  theB = Val(theReply)
  IfNot IsNumeric(theReply) Then
    MsgBox «Повторите ввод еще раз, пожалуйста.» & _
      " Но введите, все таки, число, гражданин!", , theTitle
    OKFlag = False
  ElseIf theReply = theDefault Then
    MsgBox «Вы ничего не ввели. Повторите попытку, сделайте " & _
      " хоть что-нибудь сами.", , " Это заголовок !"
    OKFlag = False
  Else
    OKFlag = False
  End If
Loop Until OKFlag
Sheets("Лист1").Range("A1").Value = "Премия=" & theB
End Sub
```

Процедура содержит множество ошибок. Разберитесь в программе, составьте алгоритм ее работы и исправьте ошибки. Около каждого оператора введите текст, поясняющий его работу. Запустите процедуру и убедитесь в том, что:

- ✓ она работает;
- ✓ сумму премии невозможно ввести символами вместо цифр, и нельзя ошибаться при вводе десятичного знака;
- ✓ невозможно оставить все без изменения, а именно, не вводить сумму премии вообще.

Измените программу так, чтобы при вводе пустой строки процедура не заканчивалась, а требовала у пользователя повторного ввода данных.

### **5.1.2 Создание собственных процедур**

**Задание № 2.** Создайте процедуру, в которой бы использовались встроенные окна ввода/вывода и был бы организован диалог с пользователем. В ней необходимо проверить вводимые данные на подлинность. Например, возраст не может быть более 100 лет.

Не забывайте сохранять свой материал на дискету.

#### ***Примерный алгоритм***

1. Спросить имя пользователя.
2. Если оно введено и соответствует разрешенному имени, то поздравиться с пользователем, введя соответствующий текст в окно диалога. Если имя не введено или введено неверно, сообщить пользователю об этом. После третьего неверного ответа выйти из процедуры.
3. Спросить дату рождения пользователя, и, если она является недопустимой, указать ему на это и попросить ввести дату снова.
4. Посчитать и вывести количество дней, прожитых пользователем.

## **5.2. ВЫЗОВ ДИАЛОГОВЫХ ОКОН ПРИЛОЖЕНИЯ И РАБОТА С НИМИ**

В любом приложении Office существует огромное количество рабочих окон, которые можно вызвать, используя команды меню и инструкции VBA.

### **Задание № 5.**

Наберите текст приведенной ниже программы. Объясните работу всех операторов и напишите комментарии в каждой строчке процедуры. Исследуйте процедуру в пошаговом режиме, используя окно отладки. Если какие-то операторы не работают, то исправьте их. Добавьте в процедуру несколько своих подобных операторов.

```

Sub qq_10
Workbooks.Add
Workbooks.Open ("C:/VUP/Teh/Лит-98/Кн2")
Worksheets("Лист_xxx").Select
ActiveWorkbook.Worksheets.Add
ActiveSheet.Name = "Январь"
Worksheets(5).Name = "Февраль"
ActiveWorkbook.SaveAs "Книга_Иванова"
Workbooks("Книга_Иванова ").Close
Workbooks.Close
End Sub

```

**Задание № 6.**

Наберите текст процедуры, приведенной ниже. Заставьте ее работать. Установите, используя эту процедуру, для области ячеек *A3:C7* денежный формат. Убедитесь, что это так!

```

Sub qq()
Application.Dialogs(xlDialogFormatNumber).Show
End Sub

```

**Задание № 7.**

Используя процедуру, приведенную ниже, откройте какой-нибудь файл и запомните его с другим именем. Объясните, куда и как попадают Ваши файлы при их записи и чтении.

```

Sub qq()
Application.Dialogs(xlDialogOpen)
Application.Dialogs(xlDialogSaveAs).Show
Workbooks.Close
End Sub

```

**Задание № 8.**

Создайте максимально возможное для Вас число диалоговых окон приложения. Покажите, как их можно использовать.

Для вызова окон используйте набор *Dialogs* объекта *Application*, с аргументами в виде *числа-индекса*.

Эти *индексы* соответствуют константам Excel, которые можно выбрать, используя просмотр объектов. Посмотреть объекты можно после выполнения команды меню *View/Object Browser* (выбирая объекты класса *XlBuiltInDialog* библиотеки *Excel*).

Все результаты попробуйте объяснить. Сколько у Вас получилось окон?

## 6. ИСПОЛЬЗОВАНИЕ ЭЛЕМЕНТОВ УПРАВЛЕНИЯ, ВСТРАИВАЕМЫХ НА РАБОЧИЕ ЛИСТЫ

### 6.1. ИСПОЛЬЗОВАНИЕ КНОПОК

Во всех случаях жизни от кнопки можно ждать только одного из действий: включить или выключить что-то, запустить или остановить какой-нибудь процесс.

**Задание № 1.** Исследуйте процесс размещения, форматирования и использования кнопок на рабочем листе. В принципе, точно также как кнопки, на лист вставляют и используют и другие элементы управления. С помощью кнопок оцените время между началом процесса и его отдельными фазами. Для этого выполните действия, приведенные ниже.

1. Дайте рабочему листу книги любое название, например, «*Test*».
2. Создайте на листе таблицу для расчета произвольных интервалов времени между началом и концом отдельных событий, а также между временем начала текущего события и началом работы (упражнения). В ее отдельные ячейки вставьте формулы и поясняющий текст так, как на рисунке 6.1.

	А	В	С	Д	Е
1					
2			Время начала/конца	Время между соседними упражнениями	Время от начала теста до конца текущего упражнения
3		<i>Начало теста (упр-я №1)</i>	16:20:40		
4		<i>Конец упражнения №1</i>	16:20:40	=C4-C3	=C4-C3
5		<i>Конец упражнения №2</i>	16:20:40	=C5-C4	=C5-C3
6		<i>Конец упражнения №3</i>	16:20:40	=C6-C5	=C6-C3
7		<i>Конец упражнения №4</i>	16:20:40	=C7-C6	=C7-C3
8					
9	<b>Начать тест (упражнение №1)</b>	Выполнено упражнение №1	Выполнено упражнение №2	Выполнено упражнение №3	Выполнено упражнение №4
10					
11					
12					

Рисунок 6.1 - Вид рабочего листа

3. Поставьте (перетащите мышью) на этот лист пять кнопок, используя панель инструментов «*Элементы управления*».

Панель «*Элементы управления*» появится после щелчка по кнопке «*Вставить*», расположенной в окне «*Элементы управления*» на вкладке ленты «*Разработчик*». Ее вид представлен на рисунке 6.2. Все элементы с панели управления доступны для использования на рабочем листе.

4. Группа элементов «ActiveX» доступна при включенном «Режиме конструктора» (см. рисунок 6.2).

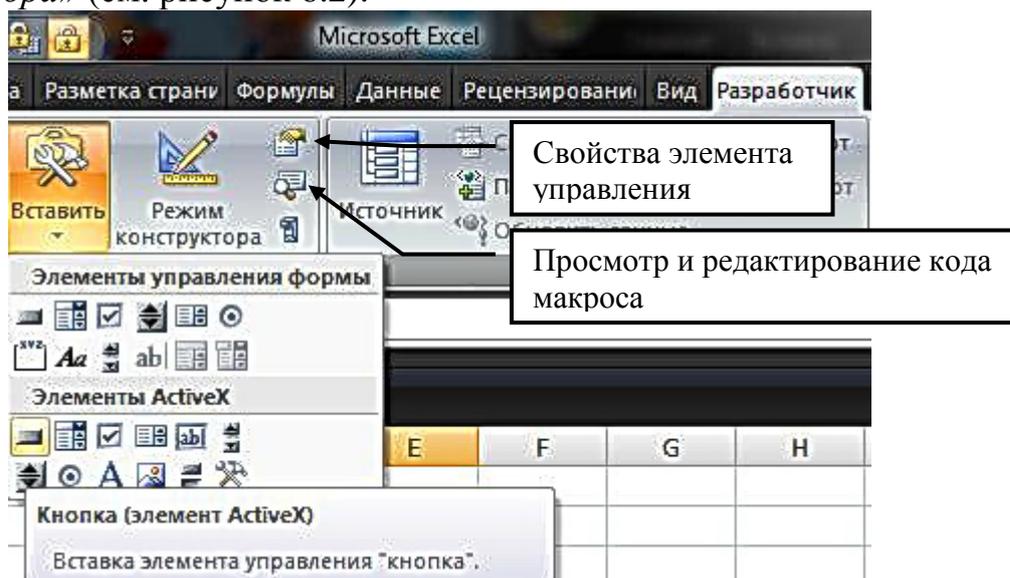


Рисунок 6.2 – Доступные элементы управления

Чтобы отформатировать кнопку (либо любой другой элемент управления), размещенную на листе (изменить ее размеры, цвет, определить доступность, передвинуть в другое место и т.д.), или написать для этого элемента управления программный код-макрос, должен быть включен *режим конструктора*. После отключения *режима конструктора* установленные элементы переходят в *рабочий режим*. Форматирование таких элементов становится невозможным.

Управляющие кнопки для этих действий представлены на рисунке 6.2. Кроме того, действие этих кнопок задублировано соответствующими командами контекстного меню, появляющегося при щелчке по элементу правой кнопкой мыши.

Вид окна для просмотра кода, в случае использования элемента ActiveX «Кнопка», представлен на рисунке 6.3.

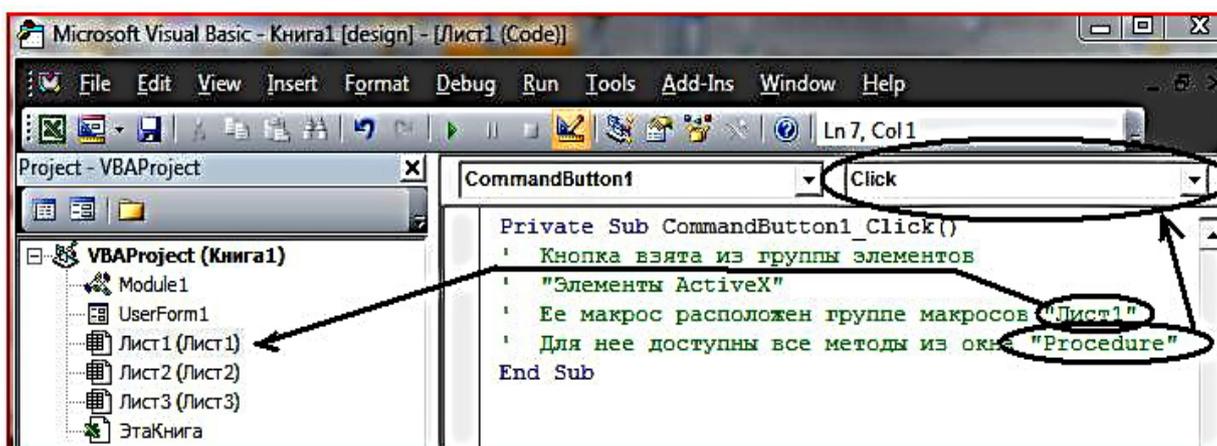
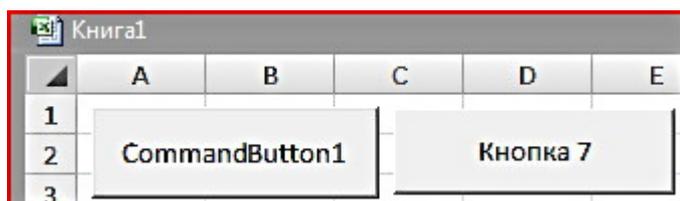


Рисунок 6.3 – Окно редактирования программного кода

Имя у кнопки, установленной на рабочем листе, будет «*CommandButton1*». Потом его можно будет изменить.



5. При установке кнопки из группы «*Элементы управления Формы*» ей будет дано имя «*Кнопка 1*» и т.д.

Измениться расположение макроса в проекте (см. рисунок 6.4).

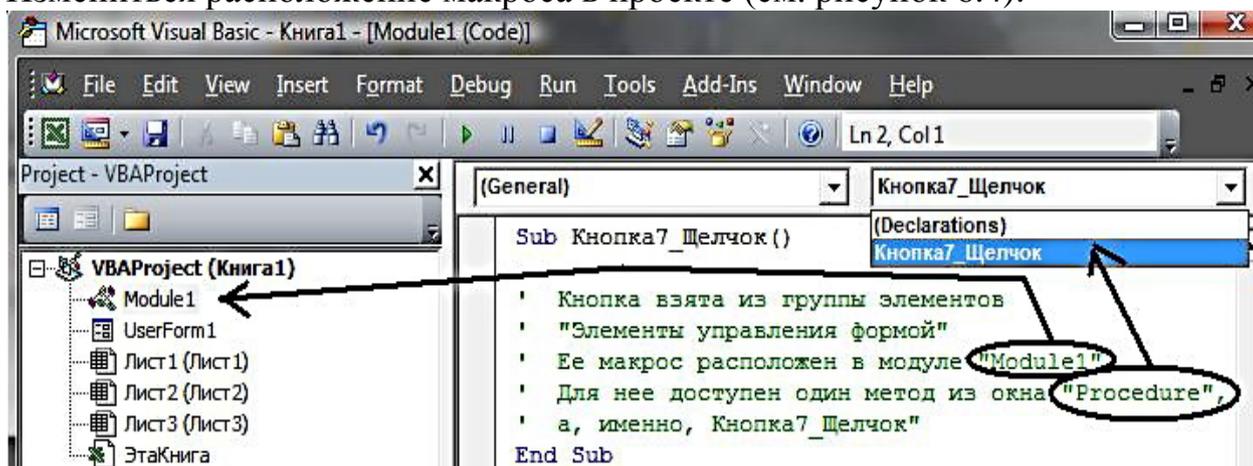


Рисунок 6.4 – Окно редактирования программного кода

Все перемещения и изменения параметров у кнопок доступны без изменений режима *Конструктора*. Достаточно щелкнуть по ним правой кнопкой мыши и выбрать нужную команду.

*! Замечание.* В дальнейшем в пособии использованы элементы управления из категории «*Элементы ActiveX*»

6. Дайте кнопкам имена и отформатируйте.

Активизировав установленную на листе кнопку, вызовите *Окно свойств (Properties)* – см. Рисунок 6.5) для каждой из них, щелкнув мышью по кнопке "*Свойства элементов управления*" на панели инструментов "*Элементы управления*", либо другим способом.



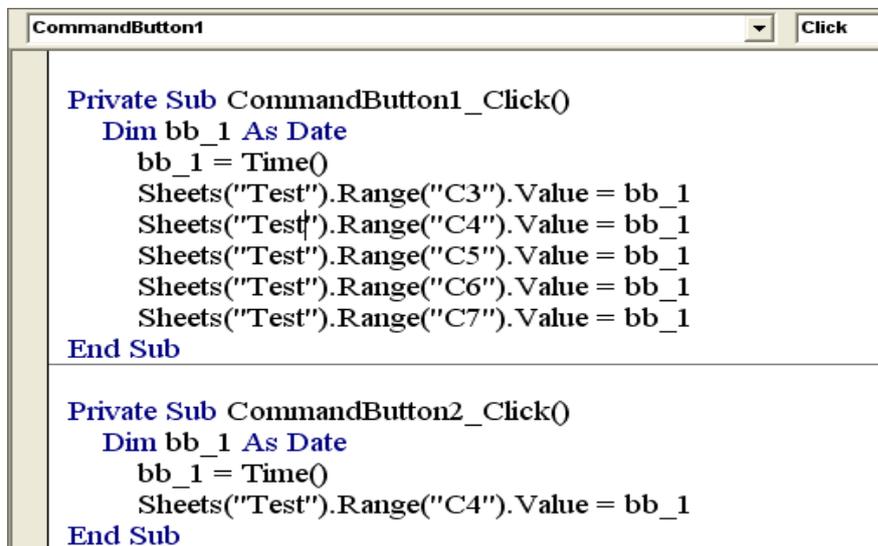
Рисунок 6.5 – Установка свойств для кнопки

Для ввода текста, появляющегося на кнопке, используйте свойство *Caption*. В этом же окне задайте цвет кнопки, цвет надписи на ней, откорректируйте другие свойства.

7. Определите те действия, которые будут происходить после щелчка по кнопкам. Такое определение называют связью событий с макросами. В нашем случае макросы - это процедуры (методы), которые должны считывать системное время и помещать его в необходимые ячейки на листе.

Для вызова окна программного кода (см. рисунок 6.3), в котором пишут текст программ, необходимо дважды щелкнуть по кнопке, установленной пользователем на листе, или щелкнуть по кнопке "Просмотр и редактирование кода", расположенной на панели инструментов "Элементы управления" (режим конструктора должен быть включен).

Примерный текст процедур, которые должны быть выполнены при активизации двух первых кнопок, приведен ниже. Остальное допишите самостоятельно.



```
CommandButton1 Click
Private Sub CommandButton1_Click()
    Dim bb_1 As Date
    bb_1 = Time()
    Sheets("Test").Range("C3").Value = bb_1
    Sheets("Test").Range("C4").Value = bb_1
    Sheets("Test").Range("C5").Value = bb_1
    Sheets("Test").Range("C6").Value = bb_1
    Sheets("Test").Range("C7").Value = bb_1
End Sub

Private Sub CommandButton2_Click()
    Dim bb_1 As Date
    bb_1 = Time()
    Sheets("Test").Range("C4").Value = bb_1
End Sub
```

Рисунок 6.6 - Примерный вариант программного кода

8. Сохраните книгу.
9. Убедитесь в работоспособности программного продукта.
10. Измените модуль с программой.
  - 10.1. Вставьте оператор, требующий обязательного описания типов переменных.
  - 10.2. Опишите переменную `bb_1` так, чтобы можно было удалить ее описание внутри каждой из процедур.
  - 10.3. Измените программу так, чтобы при заполнении таблиц в ячейках не появлялись служебные символы ##### (если такие у вас получились).
  - 10.4. Попробуйте вставить кнопки из группы элементов управления «Элементы управления Формы». Используя эти кнопки, добейтесь того же результата.
11. *Дополнительное сложное задание.* Попробуйте.

Вставьте еще одну кнопку на лист. Напишите для нее такой макрос, чтобы он удалял одну кнопку из пяти, ранее поставленных.

Здесь нужно учесть, что при установке кнопок был включен режим конструктора, а при их использовании он выключен.

## 6.2. ИСПОЛЬЗОВАНИЕ ВЫКЛЮЧАТЕЛЕЙ, ПОЛЕЙ ВВОДА

### Задание № 2.

1. Откройте книгу, созданную при выполнении задания №1.
2. Вставьте на рабочий лист еще два элемента управления: *Выключатель* и *Текстовое поле* так, как на рисунке 6.7.



Рисунок 6.7 - Вид рабочего листа

3. Отформатируйте установленные элементы управления.
4. Событие *Выключателя* **Change** свяжите с макросом, который бы блокировал *Кнопку 3* и *Кнопку 5*, а также выводил в *текстовое поле* информацию о состоянии *Выключателя*: «*Включено*» или «*Выключено*».
5. Добейтесь работоспособности разработанных программ.

## 6.3. ПОЛОСА ПРОКРУТКИ - КАК ЭЛЕМЕНТ УПРАВЛЕНИЯ

**А).** Полоса прокрутки возвращает число (*свойство Value*), которое определяется положением ее движка и двумя числами, соответствующими его начальному и конечному положению.

Эти данные можно задать программно, а можно и при форматировании данного объекта - конкретной полосы прокрутки. Для ее форматирования используют *окно свойств* точно также, как и для кнопки (см. п. 6.1). Основные свойства, определяющие ее поведение, ограничения на возвращаемые значения, указаны на рисунке 6.8.

Практически все объекты управления могут быть отформатированы таким образом до момента их использования в приложениях

**Б).** Полосу прокрутки можно помещать и на рабочих листах, и в окнах диалога (формах). В данной работе будем размещать ее на рабочих листах.

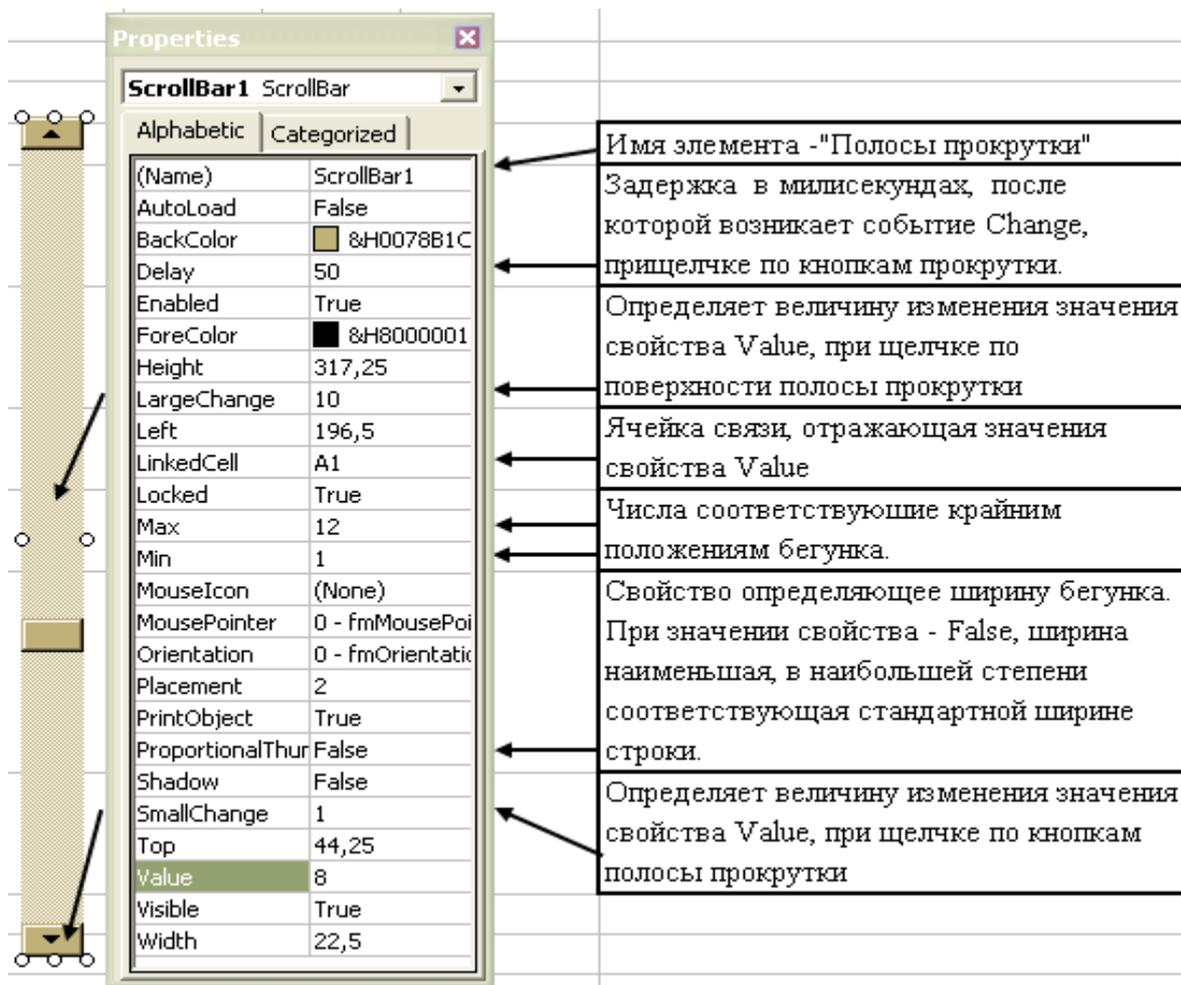


Рисунок 6.8 - Свойства полосы прокрутки



Рисунок 6.9 - Рабочий лист с полосой прокрутки

Обычно, положение движка полосы прокрутки определяет число, которое указывает на какой-то элемент списка (5-й, 10-й и т. д.). Всегда можно сделать еще и так, чтобы это положение движка не только определяло, но и указывало на конкретную строку или столбец заданной таблицы. Для этого необходимо чтобы движок располагался напротив искомой строки или столбца.

Например, на рисунке 6.9 движок расположен напротив одиннадцатой строки таблицы. При этом может быть активизирована одна из ячеек, находящаяся напротив движка, например, A13, B13, C13, D13.

**В).** Для того, чтобы это произошло, необходимо:

*во-первых*, расположить полосу прокрутки рядом с таблицей и установить у нее размер поля для движения движка таким, чтобы в него попадали все строки (столбцы) таблицы (см. рисунок 6.9, строки с 3-й по 14-ю);

*во-вторых*, используя форматирование полосы прокрутки, установить соответствие между начальными и конечными крайними положениями движка в этой полосе прокрутки и соответствующими им числами (в данном случае - это 1 и 12), а также числом, определяющим величину шага между соседними положениями движка (в данном случае шаг должен быть равен 1);

*в-третьих*, связать полосу с макросом, который должен быть написан заранее, так, чтобы при каждой инициализации полосы прокрутки (щелчка по ней мышью) этот макрос выполнялся.

### **Задание № 3.**

Создайте таблицу для отображения заработка трех сотрудников по месяцам, приведенную на рисунке 6.9. Используя полосу прокрутки, добейтесь того, чтобы при нахождении ее движка напротив ячейки с соответствующим названием месяца, эта ячейка была бы активизирована.

Для этого сделайте следующее.

1. Полосу прокрутки расположите так, чтобы поле для движения движка начиналось с верхней границы строки "*Январь*" и заканчивалось на нижней границе строки "*Декабрь*". В этом случае, с учетом выполнения условий форматирования, представленных на рисунке, каждый шаг движка в полосе прокрутки будет перемещать движок на один месяц (на одну строку) вверх или вниз.

2. Отформатируйте полосу прокрутки в соответствии указаниями, представленными на рисунке 6.9.

3. Для того чтобы выполнить заданные требования нужно написать макрос. Вызовите окно модуля и в нем для события ***Change*** (изменение) объекта прокрутка напишите процедуру, текст которой представлен ниже на рисунке 6.10.

4. Проверьте работоспособность созданного приложения.

5. Совершенно очевидно, что предложенный код, хоть и предельно понятен, но страдает безумными излишествами. Измените текст процедуры

так, чтобы он занимал не более 3-х строк. Для этого целесообразно вспомнить, что у объекта *Range* есть замечательное свойство *Cells*.

6. Проверьте работоспособность измененного приложения.

```

ScrollBar1
Change
Private Sub ScrollBar1_Change()
    Dim N_Mes As Integer
        ' N_Mes - вспомогательная переменная, позволяющая избежать
        ' излишне длинной записи при проверке условных операторов.
    Sheets(2).Range("A1").Value = ScrollBar1.Value
        ' Размещаем на втором листе в ячейку A1 значение св-ва Value прокрутки
        ' это св-во должно соответствовать номеру месяца от 1 до 12
    N_Mes = Sheets(2).Range("A1").Value
        ' Присваиваем переменной N_Mes значение, равное числу из ячейки "A1",
        ' с рабочего листа "Лист2" (т.е. номер месяца)
    Sheets("Лист2").Select      ' Явно указываем на лист "Лист! "
    If N_Mes = 1 Then          ' В том случае, если это 1-й месяц
        Range("A3").Select      ' Активизируем ячейку "A3".
        ElseIf N_Mes = 2 Then Range("A4").Select:
        ElseIf N_Mes = 3 Then Range("A5").Select
        ElseIf N_Mes = 4 Then Range("A6").Select:
        ElseIf N_Mes = 5 Then Range("A7").Select
        ElseIf N_Mes = 6 Then Range("A8").Select:
        ElseIf N_Mes = 7 Then Range("A9").Select
        ElseIf N_Mes = 8 Then Range("A10").Select:
        ElseIf N_Mes = 9 Then Range("A11").Select
        ElseIf N_Mes = 10 Then Range("A12").Select:
        ElseIf N_Mes = 11 Then Range("A13").Select
        ElseIf N_Mes = 12 Then Range("A14").Select
    End If
End Sub

```

Рисунок 6.10 - Примерный программный код

#### Задание № 4.

1. Таблицу на рисунке 6.9 дополните еще одним столбцом (столбцом *E* с аналогичными данными для Смирнова).

2. На этот же лист добавьте еще одну полосу прокрутки - на этот раз, *Горизонтальную*, для того чтобы можно было выбрать не только строку, но и столбец (*B, C, D, E*).

3. Напишите процедуры для обработки соответствующих событий таким образом, чтобы можно было активизировать любую ячейку таблицы (*B3:E14*), пользуясь горизонтальной и вертикальной полосами прокрутки.

При написании макроса используйте метод *Cells* объекта *Range*.

4. Текст процедур не должен занимать место более 3-х строк.

5. Проверьте работоспособность программного продукта.

6. ! *Добейтесь такой работы программ, чтобы одна ячейка таблицы (B3:E14) при любых положениях движков прокруток всегда была активна.* Как правило, этого сначала не получится. При смене положения движка у полос прокруток активизация ячеек будет проходить через раз.

## 7. СОЗДАНИЕ ДИАЛоговых ОКОН ПОльзовАТЕЛЕЙ. ИССЛЕДОВАНИЕ РАБОТЫ ЭЛЕМЕНТОВ УПРАВЛЕНИЯ. ПРОЦЕДУРЫ ОБРАБОТКИ СОБЫТИЙ

### 7.1. ОСНОВНЫЕ ПОЛОЖЕНИЯ

В отличие от предыдущей работы элементы управления будут вставлены на экранные формы, а не рабочие листы. Для того, чтобы это сделать необходимо запустить редактор VBA, выполнив команду меню *Разработчик/Код/Visual Basic*. Того же результата можно добиться при использовании комбинации горячих клавиш  $\langle \text{Alt+F11} \rangle$ .

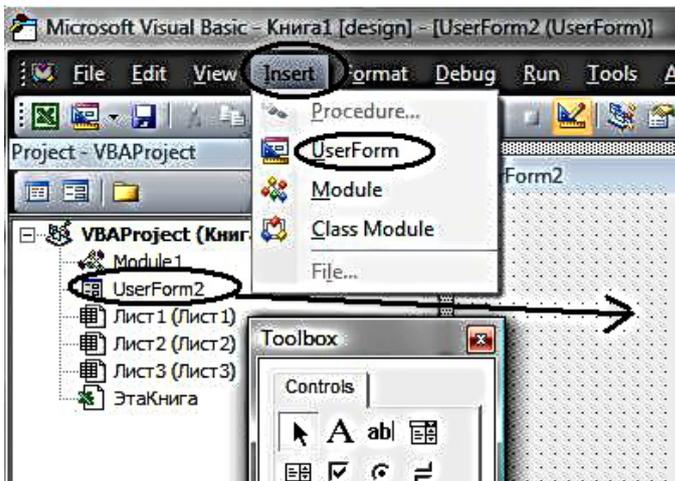


Рисунок 7.1 - Окно редактора VBA

После этого командой меню редактора VBA *Insert/UserForm* следует перейти в режим редактирования экранной формы (рисунок 7.1). В окне справа появится заготовка формы для размещения на ней элементов управления, а в окне проекта (*Project-VBAProject*) – значок, отражающий факт появления в проекте окна форм- *UserForm2*. Одновременно с этим появится окно “*Toolbox*” с набором элементов управления, которые можно перетащить на заготовку экранной формы.

Выполняя задания, вспомните о том, какие свойства имеют различные элементы с панели инструментов «*Элементы управления*» (*Toolbox*), что они собой представляют, и как с ними обращаться.

В этой работе не указано с какими событиями и как следует связывать те или иные макросы. Эти вопросы достаточно полно рассмотрены выше, поэтому при необходимости решите их самостоятельно.

Номера и имена листов, флажков и переключателей, которые Вы будете создавать, могут быть любыми и не совпадать с теми, которые указаны в тексте пособия. Словосочетания: *диалоговое окно, экранная форма, пользовательская форма* будем считать синонимами.

Номера и имена листов, флажков и переключателей, которые Вы будете создавать, могут быть любыми и не совпадать с теми, которые указаны в тексте пособия. Словосочетания: *диалоговое окно, экранная форма, пользовательская форма* будем считать синонимами.

### 7.2. СТРАНИЦЫ, ФЛАЖКИ И ПЕРЕКЛЮЧАТЕЛИ В ДИАЛОГОВЫХ ОКНАХ

#### ✓ Страницы

**Задание № 1.** Создайте новую пользовательскую форму (диалоговое окно). На эту форму перетащите с панели элементов “*Элементы управ-*

ления «Набор страниц» (*MultiPage*). Используя свойства элементов, дайте названия окну диалога, страницам, так, как это сделано на рисунке 7.2. При создании (добавлении) новых страниц элемента *MultiPage* используйте его контекстное меню.

✓ **Флажки**

**Задание №2.** Исследуйте работу флажков (*CheckBox*) и переключателей (*OptionButton*), для этого:

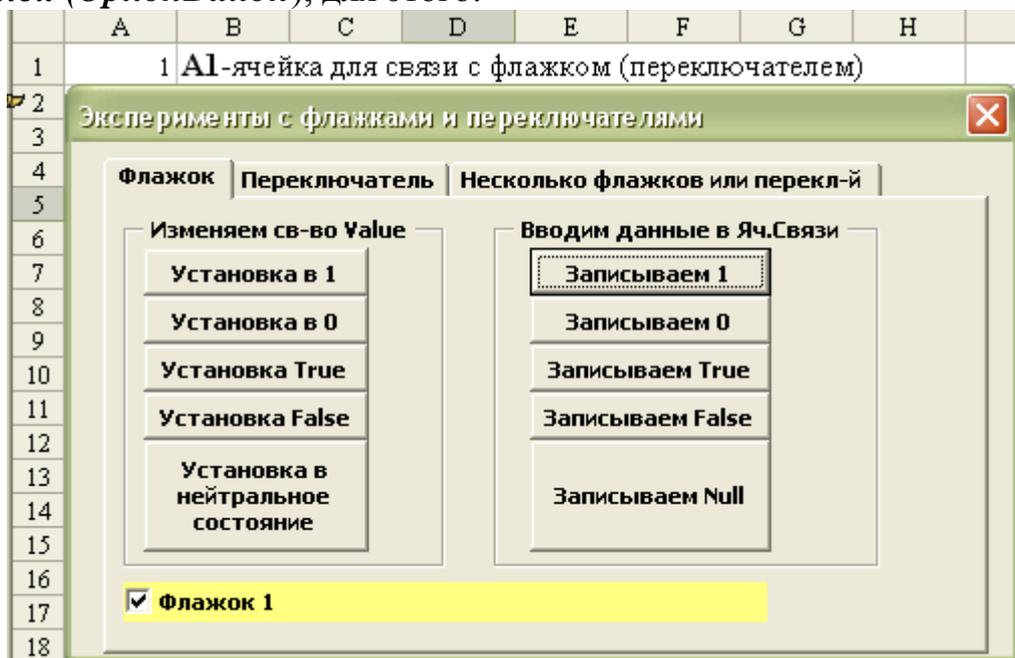


Рисунок 7.2 - Вид страницы «Флажок» на экранной форме

1. Разместите в окне созданного диалога на странице "Флажок" элементы управления, изображенные на рисунке 7.2.

2. Используя свойства элемента, свяжите **Флажок 1** с ячейкой **A1** рабочего листа.

Установите свойство *TripleState = True*. При этом у флажка появится 3-е состояние -нейтральное (*Null*). По умолчанию у флажка 2 состояния, а свойство *TripleState = False*.

3. Поэкспериментируйте.

3.1. Активизируйте групповое окно "Изменяем свойство Value".

3.2. Создайте 5 отдельных процедур, устанавливающих свойство *Value* у флажка в **1, 0, True, False и Null**. Присвойте (назначьте) созданные макросы соответствующим кнопкам этого группового окна.

3.3. Расположив предварительно рядом два окна - одно с диалогом, другое с листом "Лист1", активизируйте окно диалога (щелкните по нему мышкой, а потом нажмите клавишу **F5** на клавиатуре). Добейтесь поднятия и опускания флажка щелкая по соответствующим кнопкам на форме, оценивая при этом содержимое ячейки связи **A1**.

3.4. Активизируйте групповое окно "Вводим данные в Яч.Связи".

Проделайте то же самое, что и п.п.3.2 - 3.3, создав предварительно процедуры, записывающие непосредственно в ячейку связи *1, 0, True, False и Null*.

### ✓ Переключатели

**Задание № 3.** На странице "*Переключатель*" элемента *MultiPage*, создайте интерфейс подобный представленному на рисунке 3.2, но для переключателя.

Произведите исследования одного переключателя аналогичные тем, которые были сделаны для элемента управления «*Флажок 1*», расположенного на странице «*Флажок*» (в п.п. 2-3). При этом учтите, что у переключателя только два состояния.

### ✓ Несколько флажков или несколько переключателей

**Задание № 4.** Несколько флажков или несколько переключателей могут быть объединены групповыми окнами (*Frame*).

На странице "*Несколько флажков или перекл-й*" расположите элементы так, как указано на рисунке 7.3. Примечания разместите на полях ввода (*TextBox*), предварительно задав для них режим ввода многостраничного текста.

Напишите 4 макроса:

✓ первый, для записи конкретного имени "*Иван*" в ячейку *A2* и свяжите его с флажком "*Флажок 2*";

✓ второй, для записи конкретного отчества "*Иванович*" в ячейку *B2* и свяжите его с флажком "*Флажок 3*";



Рисунок 7.3 - Вид страницы «Несколько флажков или перекл-й»

на экранной форме

✓ третий, для записи конкретной имени "Сидоров" в ячейку C2 и свяжите его с флажком "Флажок 4";

✓ четвертый, для стирания информации из ячеек A2, B2, C2 и свяжите его с переключателем "Перекл.2".

Запустите диалог. Исследуйте работу флажков и переключателей, расположенных в этих групповых окнах.

Сделайте письменные выводы.

### 7.3 МЕТКИ (НАДПИСИ) и ПОЛЯ ВВОДА

**Задание № 5.** Исследуйте процесс размещения и работы с *метками (Label)* и полями *ввода (TextBox)*.

1. Создайте рабочую форму «*Метки и поля ввода*» так, как показано на рисунке 7.4.

Рисунок 7.4 - Вид экранной формы «Метки и поля ввода»

2. Присвойте имена меткам группового поля "*Результат ввода*": MET1, MET2, MET3, а рабочим полям ввода в окне "*Ввод*" соответственно: Поле\_Фам, Поле\_Им, Поле\_Отч.

3. Напишите макрос, который бы изменял текст меток группового поля "*Результат ввода*", на текст, приведенный на рисунке, и присвойте его кнопке "*Исходное состояние*"

4. Напишите макрос, который бы изменял текст меток группового поля "*Результат ввода*" на текст, набираемый в соответствующих полях ввода группового окна "*Ввод*", и присвойте его кнопке "*Ввести новую информацию*".

5. Запустите диалог. Отладьте макросы. Сделайте выводы.

## 7.4. СПИСОК и ВЫКЛЮЧАТЕЛЬ

### Задание № 6.

Исследуйте свойства и методы элемента *список (ListBoxes)* и *выключателя (Togglebutton)*.

1. Рассмотрите рисунок 7.5. Он содержит список, 3 групповых окна, и расположенные отдельно 2 кнопки и выключатель. Создайте такую же свою экранную форму.

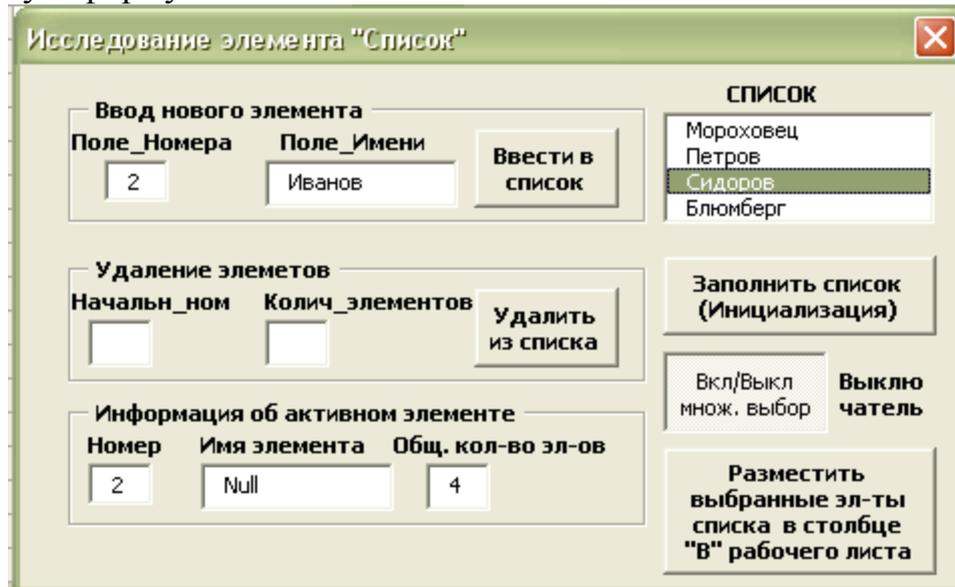


Рисунок 7.5 - Экранная форма для исследования элементов управления «Список» и «Выключатель»

2. Элемент **Выключатель** по своим свойствам и поведению практически не отличается от флажка. После его установки напишите макрос, который бы переключал режим множественного выбора записей в списке обычного на множественный, используя свойство *MultiSelect*.

3. Для заполнения пустого списка используйте свойство *List*. Напишите макрос, который бы производил заполнение списка пятью фамилиями и свяжите его с кнопкой «**Заполнить список (Инициализация)**».

4. Создайте процедуру, которая при подсвечивании элемента списка, например - "Сидоров", определяла бы и записывала в текстовые поля группового поля "**Информация об активном элементе**" его номер по списку и имя, а также общее количество элементов – фамилий. Выберите событие, которое нужно связать с этим макросом.

5. Для группового поля «**Ввод нового элемента**» создайте процедуру для записи фамилии из текстового поля "**Поле\_Имени**" в список. Сделайте так, чтобы вводимая в список фамилия расположилась после той, которая имеет порядковый номер в списке, введенный Вами в поле ввода "**Поле\_номера**". Присвойте этот макрос кнопке "**Ввести в список**".

6. В создаваемой процедуре сделайте защиту от попытки ввода неправильного номера в поле ввода «Поле\_Номера». Например, в списке всего 5 фамилий, а Вы ввели номер равный 10 и т.д.

7. Создайте процедуру для удаления нескольких записей из списка и присвойте ее кнопке "Удалить из списка". Запись в списке, начиная с которой будут удаляться элементы списка, и количество этих записей должны вводиться в поля ввода "Начальн\_номер" и "Колич\_элементов", соответственно.

Для удаления нескольких записей вам понадобится цикл *For... Next*. При этом учтите, что первый элемент списка имеет номер =0, а при удалении каждого элемента список сдвигается вверх, заполняя пустое пространство. После этого место удаленного элемента займет нижерасположенный элемент.

8. Установив режим множественного выбора элементов списка, добейтесь того, чтобы подсвеченные элементы из списка можно было бы записать в столбец **В** любой рабочий лист. Разработанную процедуру свяжите с кнопкой «Разместить выбранные эл-ты списка в столбце «В» рабочего листа».

9. Проверьте и отладьте созданное Вами программное обеспечение.

## 7.5. КОМБИНИРОВАННЫЙ СПИСОК

Комбинированный список (*ComboBox*) является таким элементом, который включает в себя одновременно и поле ввода, и раскрывающийся список. Обращение к тексту, расположенному в поле ввода, осуществляется через свойство *Text*, а к выбранному элементу списка через свойство *Value*. Раскрывающийся список практически идентичен обычному списку - *ListBox*. Основное отличие – отсутствие у него множественного выбора.

	А	В	С	Д	Е	F
1	Петров	Вася	Сторож			
2	Иван	Васильевич	Сидоров			
3	Сидоров	Коля	Поэт			
4	Чертков	Ваня	Шофер			
5	Кащеев	Леня	Певец			
6	Вий	Жора	Шеф			
7	Леший	Вова	Босс			
8	Ягов	Женя	Директор			
9	Бабов	Сима	Банщик			
10	Горьныгчев	Змей	Пожарник			
11						
12	Ваня					

Этот диапазон ячеек для формирования раскрывающегося списка. Три столбца

Ячейка связи со списком

Рисунок 7.6 – Вид рабочего листа с исходными данными

Создайте комбинированный список соблюдая следующие условия:  
*во-первых*, поместите в него элементы 3 столбцов с рабочего листа (см. рисунок 7.6), используя свойство **RowSource**;

*во-вторых*, свяжите ячейку рабочего листа A12 со списком, чтобы в ней дублировалось значение свойства **Value**;

При желании можно несколько изменить внешний вид списка, изменив свойство **ListStyle** (0 -по умолчанию).

В результате работы Вы должны получить диалоговое окно, представленное на рисунке 7.7.

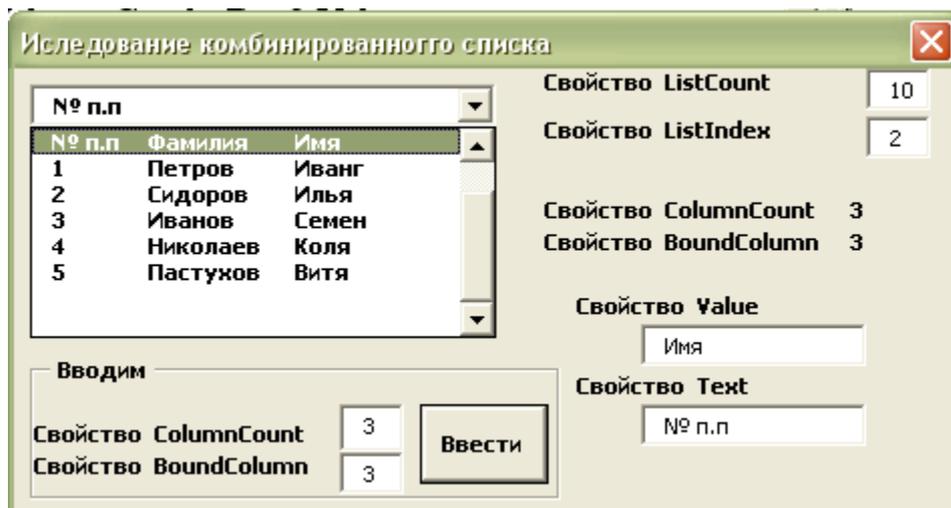


Рисунок 7.7 – Окно диалога для исследования элемента управления «Комбинированный список»

### Задание № 7.

1. В групповом окне «**Вводим**» обеспечьте возможность ввода и изменения у раскрывающего списка двух его свойств.

**ColumnCount** – определяет число столбцов в списке, выводимом на экран. Может принимать следующие значения:

=0 - столбцы не выводятся;

=1, 2, 3 - выводится 1, 2, 3 ..... столбца;

= -1 - выводятся все столбцы.

**BoundColumn** – определяет столбец, в котором происходит выбор данных для установки значения свойства **Value**. Например, на рисунке 3.6 свойство **BoundColumn**=3, а это значит то, что значение свойства **Value** будет сформировано из третьего столбца списка.

2. Справа в окне диалога расположите текстовые поля и метки для вывода в них свойств комбинированного списка.

Напомним, что свойство **ListCount** – возвращает количество элементов в списке. Это свойство только читается, а **ListIndex** – задает номер выбранной строки. В том случае, если Вы не выбрали ни один элемент, **ListIndex**= -1, а **ListCount**=0.

3. Проведите эксперименты.

- ✓ Сделайте так, чтобы раскрывающийся список содержал один столбец, два, ни одного. Обеспечьте выбор данных из первого столбца.
- ✓ Используя свойство *ColumnWidths*, установите различную ширину у каждого столбца. Это можно сделать в *пунктах* (1/72 дюйма), например, ... (46;31;120), или в *сантиметрах* - ... (2 cm; 3 cm; 4cm).
- ✓ Используя обращение к элементам списка *Объект.List(строка, столбец)*, создайте процедуру для записи в какую-нибудь ячейку рабочего листа одного из элементов списка.

## 7.6. ПОЛОСА ПРОКРУТКИ И СЧЕТЧИК

Счетчик (*SpinButton*), как и полоса прокрутки (*ScrollBar*), изменяет значение свойства Value. Один щелчок мышью изменяет это свойство на величину, заданную свойством *SmallChange* в пределах, определенных свойствами *Min* и *Max*. Вертикальное или горизонтальное расположение счетчика определяется свойством *Orientation*.

**Задание № 8.** Исследуйте полосу прокрутки и счетчик

1. Создайте окно диалога (форму) "*Исследование прокрутки и спиннера*" представленное на рисунке 7.8. Если необходимо, то объектам присвойте имена. На рабочем листе "*Лист3*" создайте список, изображенный на том же рисунке. Этот список используйте для задания исходных данных при создании макросов и исследовании прокрутки и счетчика.



Рисунок 7.8 - Окно диалога и рабочий лист для исследования работы элемента управления «Полоса прокрутки» и «Счетчик»

2. Напишите процедуру, которая бы при изменении состояния (положения движка) полосы прокрутки заполняла бы соответствующие поля ввода в групповом окне "*Полоса прокрутки*". Эту процедуру свяжите с изменением полосы прокрутки.

3. Аналогичным образом поступите со *счетчиком (спиннером)*.

Обратите внимание на то, что номер по порядку в списке, представленном на рисунке 3.7 справа не совпадает с номером строки на рабочем листе. Например, у Петрова номер п.п. – 1, а запись с его фамилией находится в 4 строке.

4. Напишите процедуру для того, чтобы задать диапазон значений, возвращаемых свойством *Value* прокрутки и счетчика, соответственно. Свяжите ее с кнопкой «*Установка диапазона .....*»

5. Отладьте процедуры и добейтесь работы окна диалога.

## 8. СОЗДАНИЕ ПРИЛОЖЕНИЯ EXCEL ДЛЯ АВТОМАТИЗАЦИИ РАСЧЕТОВ

В результате ультразвукового обследования щитовидной железы получают линейные размеры (длину, ширину, толщину) ее двух долей - левой и правой. После этого определяют объем каждой доли и общий объем железы.

Объем долей определяют по формуле:

*Объем доли = длина \* ширина \* толщина \* коэффициент,*

*Общий объем = Объем левой доли + Объем правой доли,*

а коэффициент выбирают исходя из условия:

*коэффициент 0.526, если возраст человека > 16 лет,*

*0.474, если возраст человека ≤ 16 лет.*

После сравнения рассчитанных объемов с их допустимыми нормами, представленными в таблице 8.1, делают вывод о том: находятся истинные размеры желез в норме, меньше или больше нормы. Результат обследования оформляют в виде бланка.

### Задание.

Автоматизируйте расчеты анализа обследования щитовидной железы. Для этого выполните действия, представленные ниже.

1. На одном из листов рабочей книги наберите таблицу 8.1.
2. В редакторе VBA создайте экранную форму (окно пользовательского интерфейса) для заполнения данных по расчету и анализу щитовидной железы, например, такую, как представленная на рисунке 8.2.
3. В *пользовательское меню* поместите команду для запуска экранной формы, так как это показано на рисунке 8.1.

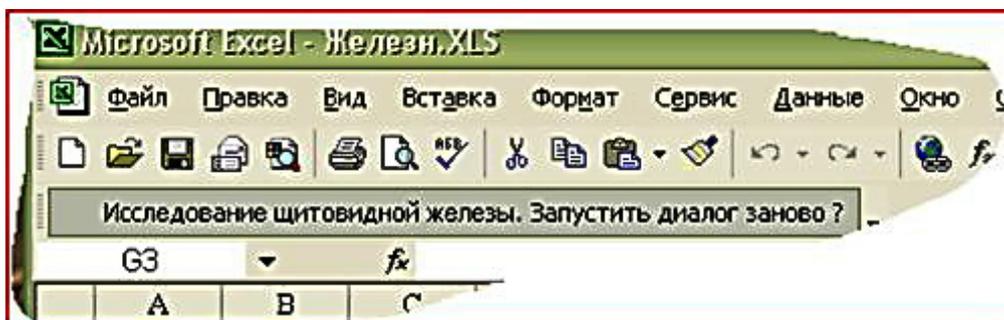


Рисунок 8.1 – Пользовательское меню

4. На пустом листе Excel создайте бланк для размещения исходных данных и результатов анализов (см. рисунок 8.3).
5. Предусмотрите автозапуск экранной формы при открытии рабочей книги.

6. Предусмотрите удаление пользовательского меню, в том случае, если наступит событие "Закреть книгу".

*По требованию преподавателя:*

7. Предусмотрите печать бланка из экранной формы.

8. Предусмотрите возможные ошибки пользователя при вводе чисел в текстовые поля экранной формы.

Например, когда пользователь вместо запятой, как разделителя дробной и целой части, ввел точку. Для обработки ошибок используйте встроенные средства Excel (объект Error).

Таблица 8.1- Нормы объема железа

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1															
2		<b>Нормы объема щитовидной железы</b>													
3															
4		<b>Женщины</b>						<b>Мужчины</b>							
5	<b>Возраст</b>	<b>Правая</b>		<b>Левая</b>		<b>Общий (Пр+Лев)</b>		<b>Правая</b>		<b>Левая</b>		<b>Общий (Пр+Лев)</b>			
6		<b>min</b>	<b>max</b>	<b>min</b>	<b>max</b>	<b>min</b>	<b>max</b>	<b>min</b>	<b>max</b>	<b>min</b>	<b>max</b>	<b>min</b>	<b>max</b>		
7	4	0,51	1,25	0,44	1,18	0,97	2,41	0,54	1,24	0,5	1,18	1,06	2,4		
8	5	0,61	1,19	0,54	1,14	1,17	2,31	0,54	1,66	0,52	1,38	1,06	3,01		
9	6	0,7	1,76	0,68	1,62	1,4	3,34	0,86	1,78	0,78	1,64	1,66	3,38		
10	7	0,91	1,97	0,8	1,82	1,75	3,75	0,96	2,32	0,79	2,17	1,79	4,45		
11	8	1,24	2,48	1,09	2,33	2,38	4,76	1,19	2,37	1,11	2,21	2,32	4,56		
12	9	1,3	3,38	1,19	3,09	2,53	6,43	1,28	2,66	1,21	2,51	2,53	5,13		
13	10	1,65	3,03	1,44	2,42	3,13	5,91	1,49	2,83	1,28	2,66	2,83	5,43		
14	11	1,72	3,43	1,62	3,18	3,38	6,58	1,76	3,2	1,57	3,09	3,38	6,54		
15	12	3,33	4,06	2,39	4,59	5,18	9,18	1,85	3,39	1,67	3,81	3,57	7,67		
16	13	2,54	5,74	2,37	5,05	5,03	10,69	2,13	5,31	1,96	4,6	4,19	9,81		
17	14	2,99	5,79	2,42	5,32	5,57	10,95	2,68	5,58	2,19	5,09	4,96	10,58		
18	15	3,2	7,36	2,65	6,79	5,98	14,04	3,02	6,7	2,46	5,84	5,58	12,14		
19	16	3,38	6,58	3,03	5,75	6,61	12,15	3,7	7,86	3,5	6,76	7,43	14,39		
20	17	4,43	7,95	4,13	7,19	8,7	14,98	4,22	8,74	3,72	8,18	8,06	16,8		
21	18	3,94	7,46	3,49	7,59	7,45	15,05	3,68	5,5	3,38	4,58	7,46	9,68		
22	19	2,85	7,51	2,82	7,06	5,7	14,56	3,68	5,5	3,38	4,57	7,46	9,68		
23	20	3,74	6,89	3,47	6,75	7,26	13,62	7,57	7,57	7,57	7,57	15,13	15,13		
24	21-25	3,94	8,64	3,81	7,51	7,88	16	4,74	9,14	4,22	8,14	9,16	17,28		
25	26-30	4,16	8,78	3,98	7,86	8,28	16,5	5,13	9,37	3,94	9,18	9,31	18,33		
26	31-35	3,4	10,14	2,11	9,61	5,84	19,44	4,65	10,97	4,17	10,37	8,98	21,58		
27	36-40	3,83	8,57	3,66	7,32	7,78	15,64	4,6	9,26	4,15	8,35	8,43	17,43		
28	41-45	4,39	10,11	2,92	9,66	7,59	19,47	5,72	8,72	4,92	7,68	11,32	15,72		
29	46-50	3,65	7,21	3,19	6,55	7,06	13,56	4,31	10,41	3,8	10,3	8,17	20,65		
30	51-55	2,65	9,84	2,37	8,15	5,64	17,42	4,5	9,48	4,12	9,3	8,68	18,72		
31	56-60	3,55	7,57	3,35	7,69	6,9	15,26	3,42	9,7	3,72	9,02	7,78	18,6		
32	61-65	2,38	7,96	2,45	6,49	5,15	14,09	3,19	8,57	2,3	10,04	5,55	18,55		
33	66-70	1,77	5,23	1,5	4,62	3,32	9,8	1,89	8,47	1,85	8,47	3,77	17,05		
34	71-75	1,36	4,14	1,29	4,09	2,65	8,21	2,7	5,12	2,59	4,81	5,39	9,89		
35	76-80	1,2	3,44	1,2	3,18	2,45	6,57	2,47	8,19	1,87	8,07	4,88	16,2		
36	81-95	1,02	2,92	1,01	2,92	2,06	5,86	1,02	2,92	1,01	2,92	2,06	5,86		

9. Создайте дополнительную таблицу (базу данных) для хранения информации, полученной в результате исследования всех пациентов.

10. Создайте дополнительную экранную форму или добавьте необходимые элементы управления в существующую, чтобы появилась возможность просматривать созданную базу данных, производить поиск информации (например, по фамилии), стирать и корректировать записи.

**Сравнение результатов исследования с нормативными значениями**

Фамилия И.О.  Коэффициент 0,526

Возраст (лет)  Пол  Мужской  Женский

Размеры щитовидной железы (см)

Левая доля		Правая доля	
Длина	<input type="text" value="10,7"/>	Длина	<input type="text" value="2,2"/>
Ширина	<input type="text" value="5,7"/>	Ширина	<input type="text" value="2,7"/>
Толщина	<input type="text" value="7,4"/>	Толщина	<input type="text" value="2,1"/>

Произвести расчеты

Распечатать лист "Результат"

Результаты расчета

Объем левой доли	<input type="text" value="237,3975"/>
Объем правой доли	<input type="text" value="6,561324"/>
Общий объем	<input type="text" value="243,9588"/>

Нормы объема

Объем левой доли	<input type="text" value="422-8,14"/>
Объем правой доли	<input type="text" value="4,74-9,14"/>
Общий объем	<input type="text" value="9,16-17,28"/>

Отклонения объема

Объем левой доли	<input type="text" value="-184,6"/>
Объем правой доли	<input type="text" value="Норма"/>
Общий объем	<input type="text" value="226,67"/>

Общее заключение

Левая доля	<input type="text" value="Меньше нормы"/>
Правая доля	<input type="text" value="В норме"/>
Общий объем	<input type="text" value="Больше нормы"/>

Рисунок 8.2 - Пользовательский интерфейс

	A	B	C	D	E	F	G	H
1								
2								
3		<b>Результат ультразвукового исследования щитовидной железы</b>						
4								
5		<b>Фамилия И.О.</b>						
6		<b>Возраст (лет)</b>		17				
7		<b>Пол (муж/жен)</b>		Мужской				
8								
9		<b>Размеры щитовидной железы</b>						
10		<b>Правая доля</b>		(см)		Объем <b>правой</b> доли	0,486024	
11		длина	<b>1,2</b>			Норма	4,22-8,74	
12		ширина	<b>0,7</b>			Отклонение	-3,733976	
13		толщина	<b>1,1</b>			Заключение	Меньше нормы	
14		<b>Левая доля</b>						
15		длина	<b>3,7</b>			Объем <b>левой</b> доли	4,631956	
16		ширина	<b>1,7</b>			Норма	3,72-8,18	
17		толщина	<b>1,4</b>			Отклонение	Норма	
18						Заключение	В норме	
19								
20						<b>Общий</b> объем	5,11798	
21						Норма	8,06-16,8	
22						Отклонение	-2,94202	
23						Заключение	Меньше нормы	
24								
25								

Рисунок 8.3 – Отчет о результате обследования

## 9. ПОСТРОЕНИЕ ДИАГРАММ

Чаще всего диаграммы строят вручную с помощью мастера диаграмм. Но в ряде случаев это требуется сделать программным путем. В последнем варианте используют объект *Chart* и все вложенные в него объекты, их свойства и методы. Программный способ построения диаграмм не является интерактивным и имеет меньше возможностей.

Создать диаграмму можно используя метод *SetSourceData* объекта *Chart*. Текст программы, использующей этот метод, представлен ниже на рисунке 9.1, а результат ее работы на рисунке 9.2.

```
(General) Prim0
Sub Prim0()
'=====
'Пример построения диаграммы на 1-ом листе активной книги

Dim myRange As Range, myChart As ChartObject
With Worksheets(1)
    Set myRange = .Range("B95:E99")
    myRange.Activate
    'Данные для построения возьмем из области ячеек "B95:E99"
    'Активизируем область построения диаграмм

    Set myChart = .ChartObjects.Add(myRange.Left - 10, _
        myRange.Top + myRange.Height, _
        250, 200)

    'Создадим объект myChart (диаграмму)
    'Свяжем его размеры с размерами области данных для диаграммы.
    'Объект пока пустой. Имеет размеры, но без данных.
End With

With myChart.Chart
    ' Свойство Chart возвращает объект Chart
    .ChartType = xl3DColumn
    .SetSourceData Source:=myRange, PlotBy:=xlRows
    ' Метод SetSourceData указывает на источник данных
    .Location where:=xlLocationAsObject, Name:="Лист1"
    ' Метод Location передвигает диаграмму в новое место
End With

ActiveChart.RightAngleAxes = True
    ' Свойство повышающее наглядность диаграмм
    ' Изменяет угол зрения на диаграмму
End Sub
```

Рисунок 9.1 – Вид окна модуля с программой

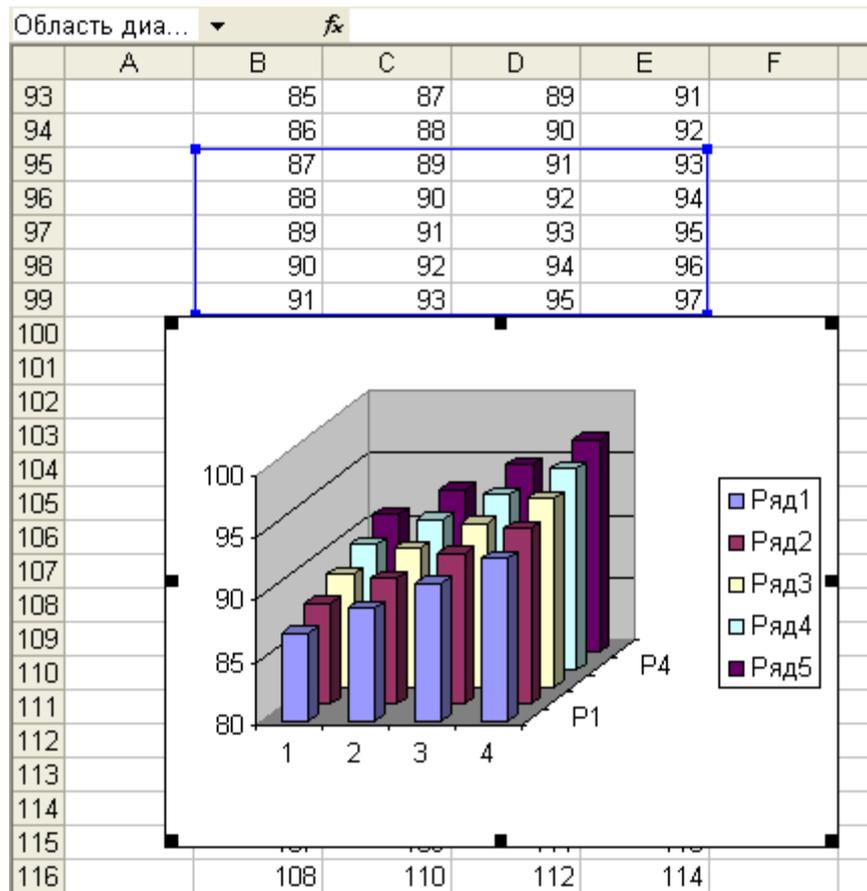


Рисунок 9.2 - Результат работы программы

### Задание №1.

1. Наберите текст программы и убедитесь в ее работоспособности.
2. Программным путем:
  - 2.1. Измените положение диаграммы на листе (сместите ее).
  - 2.2. Измените размер диаграммы на листе.
  - 2.3. Свяжите с другой область данных.
  - 2.4. Поменяйте значения свойств объектов-участников объекта *Chart* (измените названия осей, заголовков и т.д.)
  - 2.5. Используя метод *Location*, поместите диаграмму на другой лист.
  - 2.6. Создайте диаграмму на отдельном листе диаграмм (не на рабочем листе).
  - 2.7. Раскрасьте объекты в другие цвета.

### Задание №2.

Создайте самостоятельно диаграмму, используя метод *ChartWizard* вместо *SetSourceData*.

Подобъекты объекта Chart и их названия приведены на рисунке 9.3.

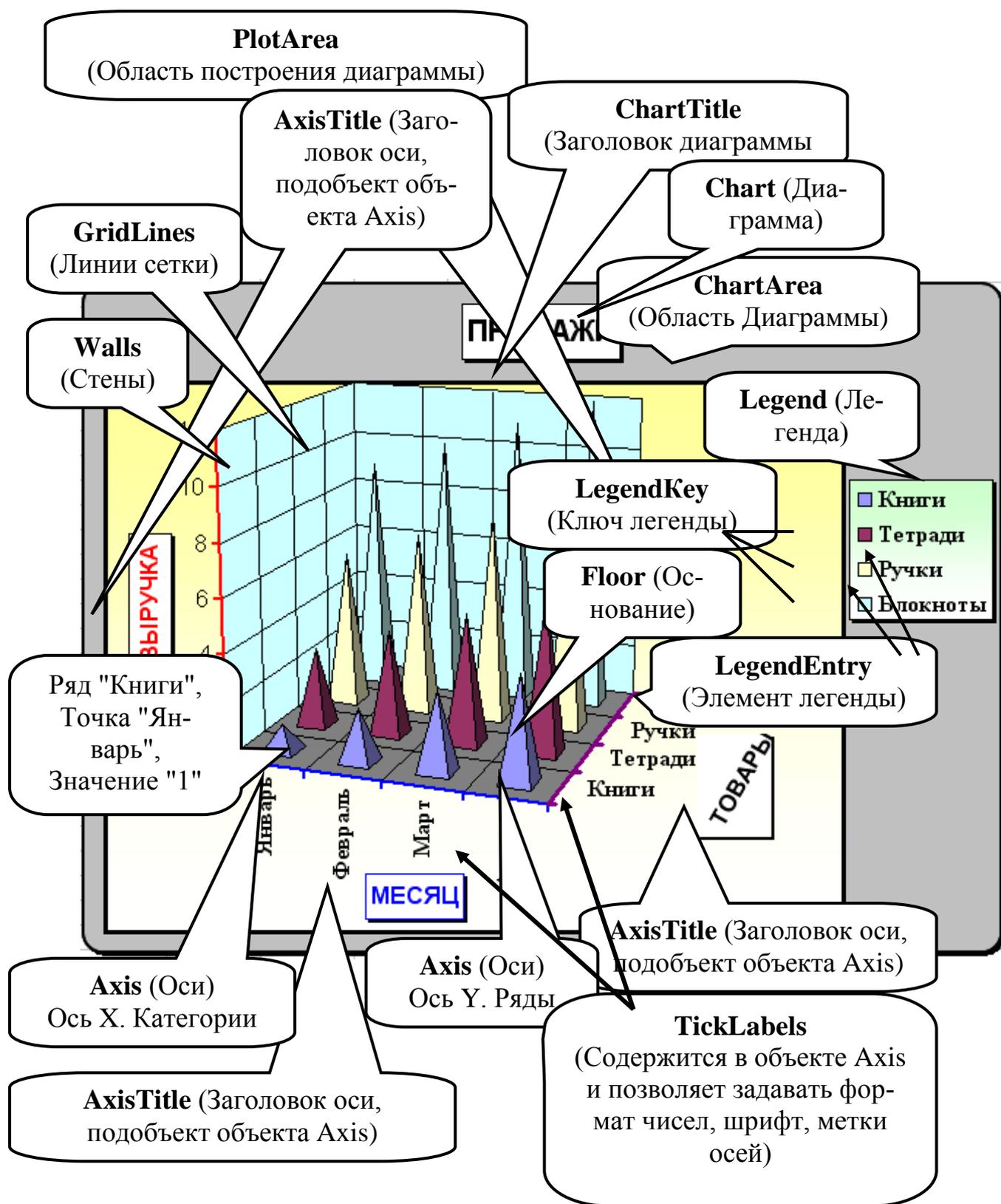


Рисунок 9.3 - Подобъекты объекта Char

## ЛИТЕРАТУРА

1. Уокенбах Дж. Excel 2013: профессиональное программирование на VBA – М.: Издательство Диалектика, 2014 г. -2014. – 960 с..
2. Электронно-библиотечная система. Издательство «Лань» [Электронный ресурс] Осетрова И.С., Осипов Н.А Microsoft Visual Basic for Application. учебник- Изд. СПб НИУ ИТМО 2013. -120 с. — Режим доступа: [http://e.lanbook.com/books/element.php?pl1\\_id=43575](http://e.lanbook.com/books/element.php?pl1_id=43575)
3. Бураков П.В., Петров В.Ю. Информационные системы в экономике. Учебное пособие. - СПб.: СПбГУ ИТМО, 2010. - 66с.
4. Зикратов И.А., Петров В.Ю. Информационные технологии в управлении. Учебное пособие. - СПб.: СПбГУ ИТМО, 2010. - 64с
4. Биллиг В.А.. Средства разработки VBA- программиста. Офисное программирование. Т.1. -М.: Издательско-торговый дом "Русская редакция", 2001. - 480 с.
6. Васильев А., Андреев А. VBA в OFFICE 2000. -СПб.: "Питер", 2001. - 432 с.

**Миссия университета** – генерация передовых знаний, внедрение инновационных разработок и подготовка элитных кадров, способных действовать в условиях быстро меняющегося мира и обеспечивать опережающее развитие науки, технологий и других областей для содействия решению актуальных задач.

---

## **КАФЕДРА ЭКОНОМИКИ И СТРАТЕГИЧЕСКОГО МЕНЕДЖМЕНТА**

Кафедра экономики и стратегического менеджмента с 9 февраля 2015г. является приемником кафедры прикладной экономики и маркетинга.

Кафедра прикладной экономики и маркетинга была основана 25 мая 1995 года в связи с началом подготовки в СПбГУ ИТМО бакалавров по направлению 521600 «Экономика». В 1997 году кафедра стала готовить сначала бакалавров, а затем и специалистов по специальности 071900 «Информационные системы в экономике». Со дня основания и по настоящее время кафедрой руководит Почётный работник высшего профессионального образования, доктор экономических наук, профессор, действительный член Российской академии естествознания Олег Валентинович Васюхин.

В настоящее время кафедра прикладной экономики и маркетинга обучает студентов по специальности 080801 «Прикладная информатика в экономике», а также готовит бакалавров и магистров по направлению 080100 «Экономика»

Кадровый состав кафедры представлен специалистами высшей квалификации – три доктора экономических наук, профессора; 9 кандидатов наук, доцентов и 6 старших преподавателей и ассистентов. Около 30% преподавателей – это молодые специалисты, обучающиеся в аспирантуре или недавно закончившие её.

В соответствии с утверждёнными учебными планами, преподаватели кафедры читают более 40 экономико-управленческих и информационных дисциплин как для студентов своих специальностей и направлений, так и для студентов всего университета. С целью обеспечения более эффективного учебного процесса преподавателями кафедры разработаны более 25 учебно-методических пособий, в том числе, часть из них в виде электронных учебников.

Кафедра обладает современной материально-технической базой. Большая часть учебного процесса реализуется в компьютерных классах Гуманитарного факультета, подключённых к сети Интернет. Все виды заня-

тий, текущий контроль знаний, а также разнообразные виды самоподготовки студентов осуществляются на основе балльно-рейтинговой системы организации учебного процесса.

Ежегодно кафедра выпускает около 50 специалистов, бакалавров и магистров, которые успешно работают на предприятиях различных форм собственности и направлений деятельности. Часть выпускников каждый год продолжают обучение в аспирантуре СПбГУИТМО. Практически все студенты кафедры, начиная с 3-4 курса, в свободное время работают на предприятиях Санкт-Петербурга, что в большинстве случаев является основой для прохождения различного рода практик и подготовки выпускной квалификационной работы.

Преподаватели и аспиранты кафедры ведут активную научно-исследовательскую деятельность, участвуя в хоздоговорных исследованиях для предприятий и организаций, а также в крупных госбюджетных НИР. В учебном процессе кафедры принимают участие представители промышленности и науки Санкт-Петербурга.

В настоящее время кафедра прикладной экономики и маркетинга является одной из ведущих выпускающих кафедр Гуманитарного факультета СПбГУ ИТМО.

Вадим Юрьевич Петров

## **Информационные технологии в менеджменте**

Учебное пособие

В авторской редакции

Дизайн

В.Ю.Петров

Верстка

В.Ю.Петров

Редакционно-издательский отдел Санкт-Петербургского государственного университета информационных технологий, механики и оптики

Зав. РИО

Н.Ф. Гусарова

Лицензия ИД № 00408 от 05.11.99

Подписано к печати \_\_\_\_\_

Заказ № \_\_\_\_\_

Тираж 50

Отпечатано на ризографе

**Редакционно-издательский отдел  
Университета ИТМО**

197101, Санкт-Петербург, Кронверкский пр., 49