



В.И. Сысков

ПРОГРАММИРОВАНИЕ БАЗОВЫХ АЛГОРИТМОВ НА VISUAL BASIC FOR APPLICATIONS (MS EXCEL)

Учебное текстовое электронное издание
Подготовлено кафедрой вычислительной техники
Научный редактор: проф., д-р т. наук С.Л. Гольдштейн

Методические указания созданы в соответствии с: рабочей программой дисциплины «Информатика» для специальностей: 151001 – Технология машиностроения; 151002 – Металлообрабатывающие станки и комплексы; 190201 – Автомобиле- и тракторостроение (автор В.И. Сысков, 2007 г.); рабочей программой дисциплины «Информатика» для специальности: 080507 – Менеджмент организации (автор В.И. Сысков, 2007 г.).

На примере расчета параметров движения тела, брошенного под углом к горизонту, рассматриваются различные варианты вычислений в MS Excel: по формуле рабочего листа и с помощью пользовательских процедур (функций и подпрограмм). Изучается технология создания в Редакторе Visual Basic процедур, реализующих базовые алгоритмы: линейный, условный и циклический. Рассматриваются вопросы взаимодействия программы с рабочим листом и с внешними файлами. Приведен пример создания пользовательской формы, панели инструментов и меню. Рассматриваются примеры использования элементов управления (кнопки, списки, флажки и пр.) для автоматизации управления ходом вычислительного процесса и автоматизации ввода данных.

ОГЛАВЛЕНИЕ

1.	Цель курсовой работы	4
2.	Постановка задачи.....	4
3.	Подготовка рабочих листов и исходных параметров.....	5
3.1.	Использование элементов панели Формы для ввода данных	6
3.1.1.	Счетчик и Полоса прокрутки	6
3.1.2.	Список и Поле со списком	6
3.1.3.	Переключатели	7
3.1.4.	Флажок	7
3.2.	Использование команды Проверка меню Данные.....	7
4.	Расчет параметров	8
4.1.	Угол броска в радианах	8
4.1.1.	Формула.....	8
4.1.2.	Функция.....	8
4.1.2.1.	Общие сведения о Редакторе Visual Basic	8
4.1.2.2.	Создание функции.....	8
4.1.2.3.	Использование функции.....	9
4.1.3.	Макрос	10
4.1.3.1.	Создание макроса.....	10
4.1.3.2.	Использование макроса	10
4.1.4.	Блок-схема.....	11
4.2.	Время движения до максимальной высоты.....	11
4.2.1.	Формула.....	11
4.2.2.	Функция.....	11
4.2.3.	Макрос	12
4.2.4.	Блок-схема.....	12
4.2.5.	Проверка корректности исходных данных.....	12
4.3.	Максимальная высота	13
4.3.1.	Формула.....	13
4.3.2.	Функция.....	13
4.3.3.	Макрос	13
4.4.	Общее время движения.....	14
4.4.1.	Формула.....	14
4.4.2.	Функция.....	14
4.4.3.	Макрос	14
4.5.	Дальность полета.....	14
4.5.1.	Формула.....	14
4.5.2.	Функция.....	14
4.5.3.	Макрос	14
5.	Получение файла с таблицей, просмотр и чтение файла	15
5.1.	Построение таблицы зависимости параметров движения тела от Угла броска α_1 , и запись таблицы в файл Tabl.txt	15
5.2.	Просмотр файла Tabl.txt в Блокноте	17
5.3.	Чтение файла Tabl.txt на лист Таблица.....	18

6.	Создание формы для работы с таблицей	20
6.1.	Создание формы	20
6.2.	Запуск формы.....	23
7.	Построение графика	24
7.1.	Использование Создателя макросов	24
7.2.	Автоматизация выбора графика	25
8.	Процедура выдачи информации о курсовой работе.....	27
8.1.	Создание процедуры Тема	27
8.2.	Вызов процедуры Тема с помощью кнопки	28
8.3.	Автоматический вызов процедуры Тема	28
9.	Создание панели инструментов и меню	28
10.	Приложения	30
10.1.	Требования по оформлению отчета по курсовой работе	30
10.2.	Структура отчета по курсовой работе.....	30
10.3.	Литература по VBA.....	34
10.4.	Ссылки в Интернете.....	34

1. Цель курсовой работы

Целью курсовой работы является изучение различных способов вычислений в MS Excel, изучение программирования базовых алгоритмов на языке Visual Basic for Applications (MS Excel), получение знаний и навыков по работе с объектами в офисных программах MS Excel и MS Word.

2. Постановка задачи

Курсовая работа заключается в расчете параметров заданного объекта – это может быть геометрическая фигура, физический объект или процесс, электрическая схема и т.п. Расчеты выполняются в Excel-книге; программы создаются в Редакторе Visual Basic на языке Visual Basic for Applications (VBA).

Необходимо предусмотреть автоматизацию ввода и проверку значений исходных параметров – с помощью элементов управления панели инструментов Формы или с помощью команды Проверка меню Данные.

Параметры объекта вычислять тремя способами: 1) путем вставки формул в ячейки рабочего листа; 2) с помощью процедуры типа Function (функция); 3) с помощью процедуры типа Sub (подпрограмма).

Предусмотреть проверку корректности исходных данных – путем использования функции ЕСЛИ (в формулах рабочего листа) и инструкции условного перехода If (в процедурах).

Получить таблицу значений параметров в зависимости от одного из циклически изменяемого исходного параметра, если заданы его начальное и конечное значения и шаг изменения. Таблицу записать в файл.

Предусмотреть программное открытие полученного файла в Блокноте.

Предусмотреть программное чтение таблицы из файла на рабочий лист.

Автоматизировать построение графика на основе полученной на рабочем листе таблицы – путем использования команды Записать макрос.

Предусмотреть автоматизацию выбора исходных данных для графика – с помощью элемента управления Список и назначенного ему макроса.

Создать пользовательский интерфейс – с использованием кнопок, пользовательских панелей инструментов, меню и форм.

Выполнение курсовой работы рассмотрим на примере **расчета параметров движения тела, брошенного под углом к горизонту** (см. Рис. 1).

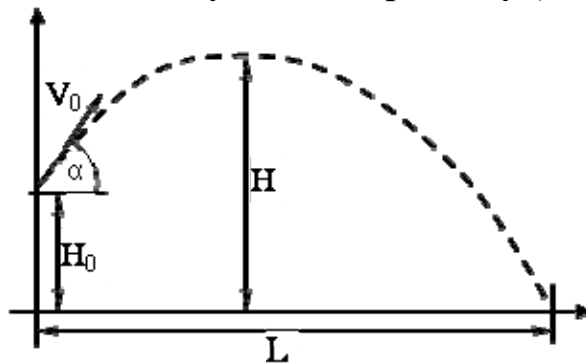


Рис. 1. Параметры движения тела, брошенного под углом к горизонту

Исходные параметры:

Высота начальной точки броска, м – H_0

Начальная скорость, м/с – V_0

Угол броска, град. – α_1

Ускорение свободного падения, м/с² – $g = 9,80665$

Рассчитываемые параметры:

Угол броска, рад – $\alpha_2 = \alpha_1 \frac{\pi}{180}$

Время движения до максимальной высоты, с – $T_H = \frac{V_0 \sin \alpha_2}{g}$

Максимальная высота, м – $H = H_0 + \frac{V_0^2 \sin^2 \alpha_2}{2g}$

Общее время движения, с – $T_L = \frac{V_0 \sin \alpha_2 + \sqrt{V_0^2 \sin^2 \alpha_2 + 2gH_0}}{g}$

Дальность полета, м – $L = V_0 T_L \cos \alpha_2$

	A	B	C	D	E	F	G	H	I	J	K
1			РАСЧЕТ ПАРАМЕТРОВ ДВИЖЕНИЯ ТЕЛА, БРОШЕННОГО ПОД УГЛОМ К ГОРИЗОНТУ								
2			ИСХОДНЫЕ ПАРАМЕТРЫ:								
3											
4											
5		Высота начальной точки броска, м - H_0		0							
6		Начальная скорость, м/с - V_0		10							
7		Угол броска, град. - α_1		45							
8		Ускорение свободного падения, м/с ² - g		9,80665							
9											
10			РАССЧИТЫВАЕМЫЕ ПАРАМЕТРЫ:								
11											
12					ФОРМУЛА	ФУНКЦИЯ	МАКРОС				
13		Угол броска, рад - α_2		0,785398163		0,785398163	0,785398163		$\alpha_2 = \alpha_1 \frac{\pi}{180}$		
14		Время движения до макс. высоты, с - T_H		0,721048249		0,721048249	0,721048249		$T_H = \frac{V_0 \sin \alpha_2}{g}$		
15		Максимальная высота, м - H		2,549290532		2,549290532	2,549290532		$H = H_0 + \frac{V_0^2 \sin^2 \alpha_2}{2g}$		
16		Общее время движения, с - T_L		1,442096498		1,442096498	1,442096498		$T_L = \frac{V_0 \sin \alpha_2 + \sqrt{V_0^2 \sin^2 \alpha_2 + 2gH_0}}{g}$		
17		Дальность полета, м - L		10,19716213		10,19716213	10,19716213		$L = V_0 T_L \cos \alpha_2$		
18											
19		Построение таблицы зависимости параметров движения тела от угла броска a1 и запись таблицы в файл Tabl.txt			Просмотр файла Tabl.txt в Блокноте			Чтение файла Tabl.txt на лист Таблица			
20											
21											
22											
23		Вызов формы для построения таблицы									
24											
25											

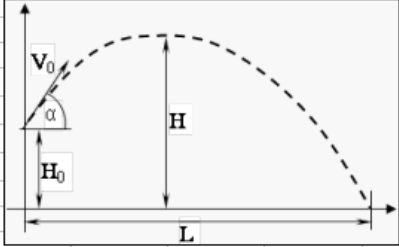


Рис. 2. Расчетный лист

3. Подготовка рабочих листов и исходных параметров

Создайте и сохраните Excel-книгу с тремя рабочими листами: Титульный лист, Расчетный лист и Таблица.

На Расчетном листе введите заголовок Расчет параметров движения тела, брошенного под углом к горизонту, и создайте иллюстрирующий рисунок (см. Рис. 2).

Введите таблицу с исходными параметрами. Введите значения неизменяемых исходных параметров, в нашем примере это ускорение свободного падения – 9,80665. Значения других исходных параметров будут вводиться с помощью элементов управления (счетчики, списки и пр.); их создадим позднее.

Введите таблицу с рассчитываемыми параметрами. Она состоит из трех столбцов – по числу вариантов вычисления: по формуле; с помощью функции; с помощью макроса.

Ячейки с названиями параметров (на рисунке столбец D) отформатированы с выравниванием по правому краю; для задания полужирного начертания, верхнего и нижнего индекса используйте меню Формат-Ячейки; для вставки математических символов (α , β и т.п.) используйте меню Вставка-Символ.

Вставьте формулы (объект Microsoft Equation 3.0), которые нужны будут для вызова макросов, вычисляющих соответствующие параметры.


Кнопки для работы с таблицей создадим позднее.

3.1. Использование элементов панели Формы для ввода данных

Активизируйте панель инструментов Формы.

3.1.1. Счетчик и Полоса прокрутки

Используем элемент Счетчик для автоматизации ввода значения высоты начальной точки броска в ячейке E5 (см. Рис. 2).

Для этого выберите на панели Формы элемент -Счетчик и очертите его контуры правее ячейки E5. Щелчком в стороне сбросьте маркеры с полученного элемента. Убедитесь: щелчок по нему пока не приводит ни к какому результату.



В контекстном меню элемента управления выберите Формат объекта, и на вкладке Элемент управления появившегося диалогового окна в поле Связь с ячейкой мышью задайте ячейку E5. (Остальные параметры этой вкладки – Текущее значение, Минимальное, Максимальное и Шаг изменения – задаются по смыслу).

Теперь щелчок по Счетчику увеличивает или уменьшает значение ячейки E5.

Для Счетчика нельзя задать дробное значение Шага изменения. Но если все-таки надо в E5 вводить дробные значения, то Счетчик надо связать с какой-нибудь другой ячейкой, например, с G5, а в E5 ввести формулу: $=0,1*G5$. Теперь (если Шаг изменения равен 1) значение ячейки E5 будет меняться на 0,1.

Если надо переместить элемент или изменить его размеры, щелкните по нему правой кнопкой и клавишей Esc уберите контекстное меню.

В примере с брошенным телом Счетчик вполне годится для задания и других исходных параметров: Начальной скорости и Угла броска. Поэтому вставьте еще два Счетчика и свяжите их, соответственно, с E6 и с E7.

Технология использования -Полосы прокрутки аналогична -Счетчику.

3.1.2. Список и Поле со списком

В других вариантах курсовой работы надо обеспечить выбор одного из множества заданных значений.

Допустим, необходимо, чтобы в E3 появлялось значение диэлектрической проницаемости среды, выбранной из заданного списка (см. Рис. 3).

	A	B	C	D	E	F	G	H	I	J
1			=ИНДЕКС(J1:J5;G3)						Воздух	1
2						Воздух			Парафин	2
3		Диэлектрическая проницаемость среды		2,2		Парафин	4		Бумага	2,3
4						Бумага			Масло	2,2
5						Масло			Ткань	3,5
6						Ткань				
7			=ЕСЛИ(G8=1;60;55)							
8		Пенсионный возраст		55		<input type="radio"/> Мужчины	2			
9						<input checked="" type="radio"/> Женщины				
10										
11			=ЕСЛИ(G12;100;0)							
12		Сумма гарантии		100		<input checked="" type="checkbox"/> Гарантия	ИСТИНА			
13										

Рис. 3. Примеры использования Списка, Переключателей и Флажка

Заведите сначала список значений (ед. изм. ф/м) – в диапазоне I1:J5.

Вставьте элемент -Список. В его контекстном меню выберите Формат объекта, и задайте параметры: Формировать список по диапазону: I1:I5; Связь с ячейкой: G3. В G3 будет появляться номер выбранного пункта в Списке.

В E3 введите формулу: =ИНДЕКС(J1:J5;G3). Функция ИНДЕКС вернет из диапазона J1:J5 ячейку, номер которой указан в G3.

Технология использования -Поля со списком аналогична -Списку.

3.1.3. Переключатели

Допустим, необходимо, чтобы в E8 отображался пенсионный возраст, зависящий от выбора: 60 – для мужчин, 55 – для женщин (см. Рис. 3).

Вставьте два элемента -Переключатель, и измените их текст, как на рисунке. Свяжите их с ячейкой G8. В ней будет отображаться номер выбранного переключателя: 1 или 2.

В E8 введите формулу: =ЕСЛИ(G8=1;60;55).

3.1.4. Флажок

Допустим, в E12 должна отображаться сумма 100 р., если продаваемый товар ставится на гарантийное обслуживание, и 0 р., если нет гарантии (см. Рис. 3).

Вставьте -Флажок на рабочий лист, и свяжите его с ячейкой G12. В ней будет отображаться одно из возможных значений: ИСТИНА – если флажок установлен, ЛОЖЬ – если нет.

В E12 введите формулу: =ЕСЛИ(G12;100;0).

3.2. Использование команды Проверка меню Данные

Для выбора диэлектрической проницаемости был использован элемент Список в комбинации с функцией ИНДЕКС (см. Рис. 3).

Вводить данные в ячейки, выбирая значения из списка, можно и по-другому.

Выделите ячейку E3, и выполните команду Проверка меню Данные. На вкладке Параметры появившегося диалогового окна выберите тип вводимых данных Список, и в качестве источника данных выделите диапазон J1:J5.

Теперь появится возможность выбирать значения непосредственно в E3.

Диалог Проверка позволяет выбрать кроме Списка и другие типы данных, а также задать сообщение для ввода (отображаемое при активизации ячейки), и сообщение об ошибке (при попытке ввести не допустимые значения).

4. Расчет параметров

Расчет каждого из параметров выполняется тремя способами: 1) путем вставки формулы в ячейки рабочего листа; 2) с помощью процедуры типа Function (функция); 3) с помощью процедуры типа Sub (подпрограмма).

4.1. Угол броска в радианах $\alpha_2 = \alpha_1 \frac{\pi}{180}$

4.1.1. Формула

В ячейку E13 вставьте формулу: =E7*ПИ()/180. Изменяя значение ячейки E7, убедитесь, что E13 меняется соответствующим образом.

4.1.2. Функция

Через меню Сервис-Макрос перейдите в -Редактор Visual Basic.

4.1.2.1. Общие сведения о Редакторе Visual Basic

Редактор Visual Basic служит для создания процедур (функций и подпрограмм) на языке Visual Basic for Applications.

Процедуры вводятся в модули (как в текстовом редакторе). Перечень модулей представлен в виде структуры в окне Project – VBAProject. Для его отображения вызовите команду меню View – Project Explorer (Рис. 4).

С точки зрения программиста Excel-книга рассматривается как проект. Проект только что созданной книги состоит из модулей ЭтаКнига, Лист1, Лист2 и Лист3. В них записываются процедуры, автоматически выполняемые по наступлению определенного события: открытие или закрытие книги, активизация листа, изменение значения ячеек и пр. Модули новых рабочих листов в проекте добавляются автоматически, автоматически и удаляются.

Модули другого типа добавляются по необходимости – через меню Insert:

UserForm – модули форм – содержат процедуры, откликающиеся на события, связанные с формой, и с элементами на ней;

Module – модули общего назначения – содержат процедуры, выполняемые не по наступлению какого-то события, а по вызову пользователя;

Class Module – модули для создания новых классов объектов.

Для открытия модуля служит двойной щелчок по нему в окне проекта. Пункт Remove контекстного меню модуля служит для его удаления (при этом будет предложено сохранить удаляемый модуль в отдельном файле).

Одновременно можно открыть несколько модулей; упорядочить их окна можно через меню Window.

Создайте модуль общего назначения: меню Insert-Module. Созданному модулю присвоится имя Module1. (Второй модуль будет называться Module2 и т.д.). В этом модуле мы будем создавать процедуры.

4.1.2.2. Создание функции

Чтобы в модуль вставить процедуру, выберите меню Insert-Procedure – появится диалоговое окно Add Procedure (Рис. 4).

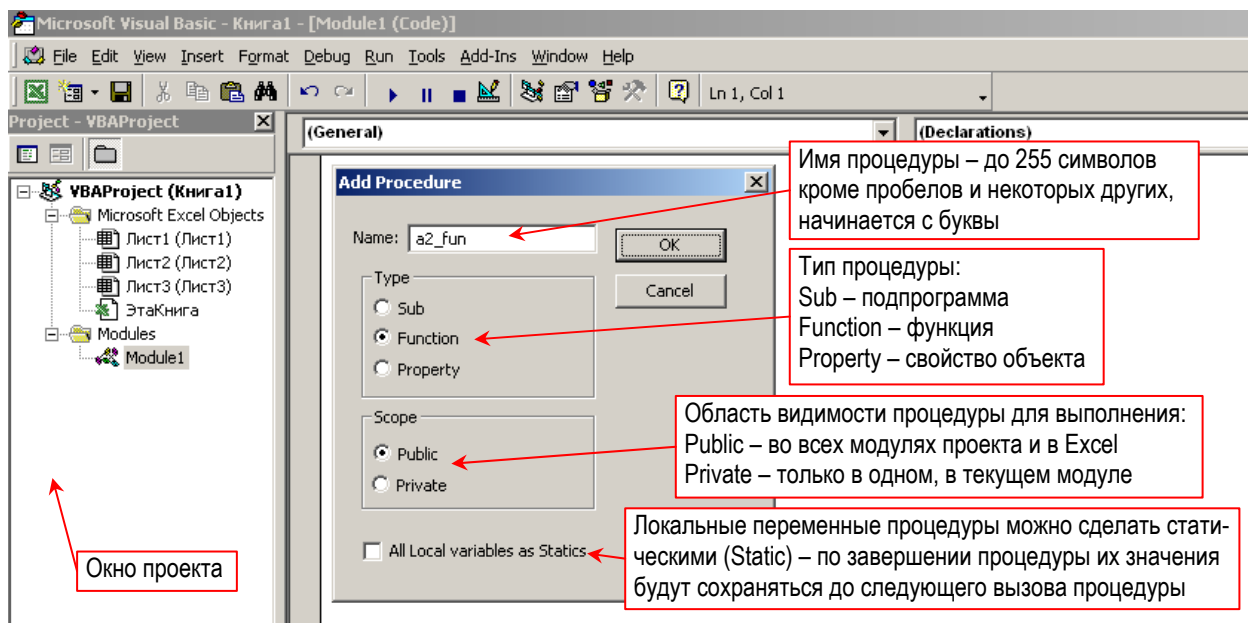


Рис. 4. Вставка процедуры в модуль общего назначения Module1

Введите имя процедуры a2_fun. Тип процедуры выберите Function. После нажатия OK в модуле появится заготовка функции – её первая и последняя строки:

```
Public Function a2_fun()  
End Function
```

Для вычисления угла броска в радианах α_2 исходными данными служит угол броска в градусах α_1 . Его значение функция будет принимать в качестве аргумента. Поэтому в скобках первой строки функции введите аргумент a1.

Function-процедура может возвращать значение в место её вызова (в отличие от Sub-процедуры, рассматриваемой ниже).

Чтобы функция возвращала значение, она должна содержать оператор присваивания, в левой части которого задается переменная, одноименная с функцией, а в правой – выражение, значение которого после вычисления и будет возвращаться в место вызова функции. Введите этот оператор присваивания:

```
Public Function a2_fun(a1)  
    a2_fun = a1 * 3.14159265358979 / 180  
End Function
```

4.1.2.3. Использование функции

В ячейку F13 вставьте формулу, вызывающую функцию: =a2_fun(E7).

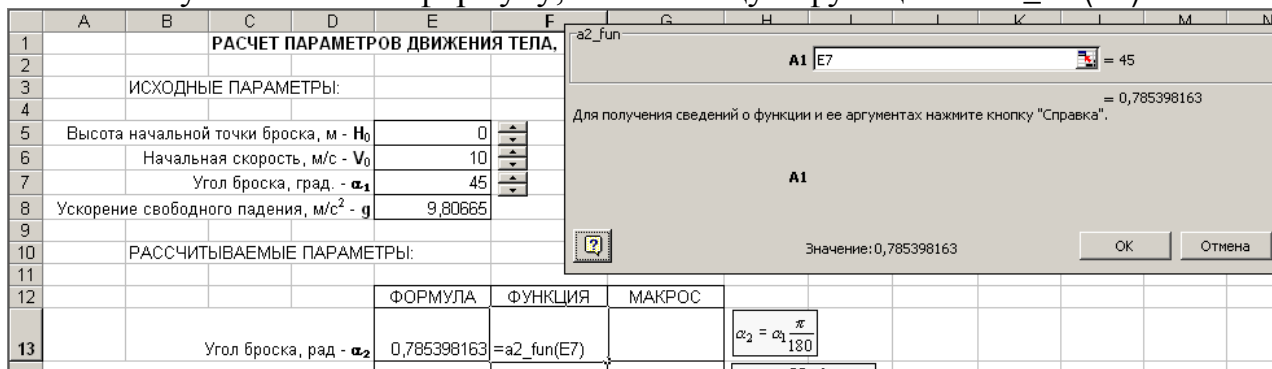


Рис. 5. Вставка функции a2_fun(E7) в ячейку F13 с помощью Мастера функций

При использовании Мастера функций функцию a2_fun ищите в категории Определенные пользователем. В качестве аргумента задайте ссылку на E7 (Рис. 5).

Изменяя значение ячейки E7, убедитесь, что F13 тоже меняется (как и E13).

4.1.3. Макрос

Наряду с Function-процедурами в VBA используется еще один вид процедур – Sub-процедуры, их еще называют подпрограммами или макросами.

Как и функция, подпрограмма может принимать исходные данные через аргументы, перечисляемые в скобках. Однако при вызове таких подпрограмм из рабочего листа Excel нет возможности задать значения аргументов.

Подпрограмма не может возвращать значение, как это делается в функции.

Поэтому передачу исходных данных в подпрограмму и передачу вычисленного значения из подпрограммы организуем по-другому, не так, как в функции.

4.1.3.1. Создание макроса

В Редакторе Visual Basic откройте модуль Module1. Для вставки подпрограммы вызовите меню Insert-Procedure, введите имя a2_sub и задайте тип Sub. После нажатия ОК появится заготовка:


```
Public Sub a2_sub()  
End Sub
```

В отличие от функции a2_fun, в подпрограмму a2_sub значение исходного параметра a1 будем передавать не через аргумент, а путем обращения к ячейке E7 с помощью функции Range. С помощью этой же функции вычисленное значение a2 запишем в ячейку G13. Введите соответствующие команды:

```
Public Sub a2_sub()  
    a1 = Range("E7")  
    Range("G13") = a1 * 3.14159265358979 / 180  
End Sub
```

Строго говоря, функция Range возвращает не только значение ячейки, а саму ячейку (или диапазон ячеек), как объект, вместе со всеми свойствами: значение, шрифт, размер, цвет и пр. Если специально не указывать, о каком свойстве идет речь (как в нашем примере), то подразумевается свойство значение.

4.1.3.2. Использование макроса

Для выполнения подпрограммы a2_sub в Редакторе Visual Basic, надо установить курсор на текст подпрограммы и выполнить команду -Run Sub/UserForm (меню Run или панели инструментов Standard) или нажать клавишу F5. Проверьте: в G13 должно получиться вычисленное значение.

Для выполнения подпрограммы a2_sub на рабочем листе (здесь Public Sub-процедуры чаще называются макросами) вызовите меню Сервис-Макрос-Макросы (или нажмите сочетание клавиш Alt+F8), в появившемся списке макросов выберите a2_sub и нажмите кнопку Выполнить (Рис. 6).

Можно упростить вызов макроса, предварительно назначив его какому-нибудь объекту (рисунку, формуле-объекту MS Equation 3.0, элементу управления и др.), размещенному на рабочем листе.

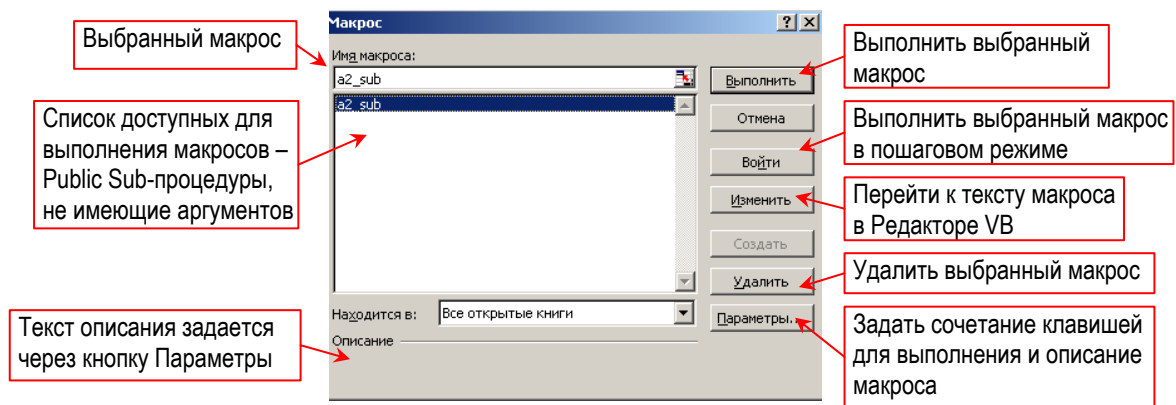


Рис. 6. Диалоговое окно Макрос

Для этого в контекстном меню формулы-объекта $\alpha_2 = \alpha_1 \frac{\pi}{180}$ выберите Назначить макрос, в появившемся окне выберите макрос a2_sub и нажмите ОК. Формула-объект превратится в кнопку вызова макроса, убедитесь в этом.

Назначить макрос можно также кнопке панели инструментов и пункту меню – об этом речь пойдет в разделе Создание панели инструментов и меню.

4.1.4. Блок-схема

Процедуры вычисления α_2 реализует линейный алгоритм. Блок-схема алгоритма представлена на Рис. 7.



Рис. 7. Блок-схема алгоритма вычисления α_2

4.2. Время движения до максимальной высоты $T_H = \frac{V_0 \sin \alpha_2}{g}$

Этот параметр, и все следующие, рассчитываются так же, как и рассмотренный выше Угол броска в радианах.

4.2.1. Формула

В ячейку E14 вставьте формулу: =(E6*SIN(E13))/E8.

4.2.2. Функция

Введите функцию в Module1 и в F14 вставьте формулу: =TH_fun(E6;F13;E8).

```
Public Function TH_fun(V0, a2, g)
```

```
    TH_fun = V0 * Sin(a2) / g
```

```
End Function
```

4.2.3. Макрос

Введите макрос и назначьте его соответствующей формуле-объекту:

```
Public Sub TH_sub()  
    V0 = Range("E6")  
    a2 = Range("E13")  
    g = Range("E8")  
    Range("G14") = V0 * Sin(a2) / g  
End Sub
```

4.2.4. Блок-схема

В таком виде процедуры вычисления T_H реализуют линейный алгоритм. Его блок-схема представлена на Рис. 8 слева. (Она аналогична блок-схеме алгоритма вычисления α_2 , Рис. 7).

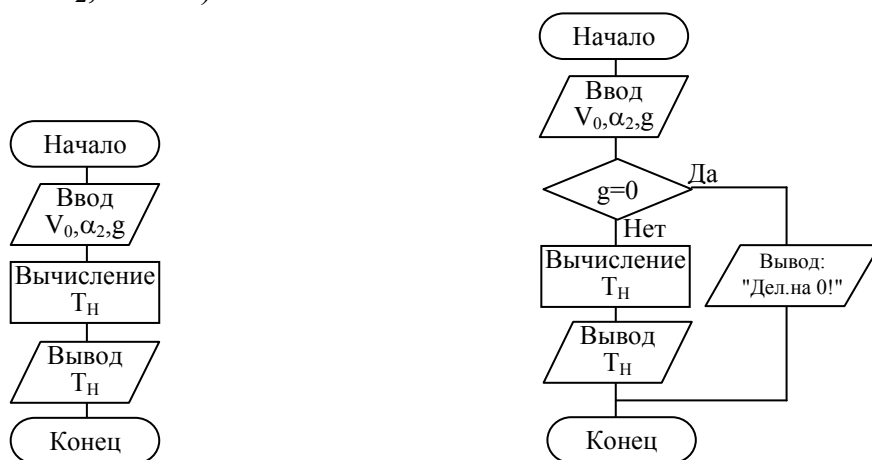


Рис. 8. Блок-схемы вычисления T_H . Вариант линейного алгоритма (слева) и вариант с проверкой корректности исходных данных (справа)

4.2.5. Проверка корректности исходных данных

Чтобы при вычислении параметров избежать таких ошибок, как, например, деление на ноль или попытка извлечения корня из отрицательного числа, то необходимо предусмотреть проверку корректности исходных данных: в формулах ячеек рабочего листа это можно сделать с помощью функции ЕСЛИ, а в процедурах – с помощью оператора условного перехода `If`.

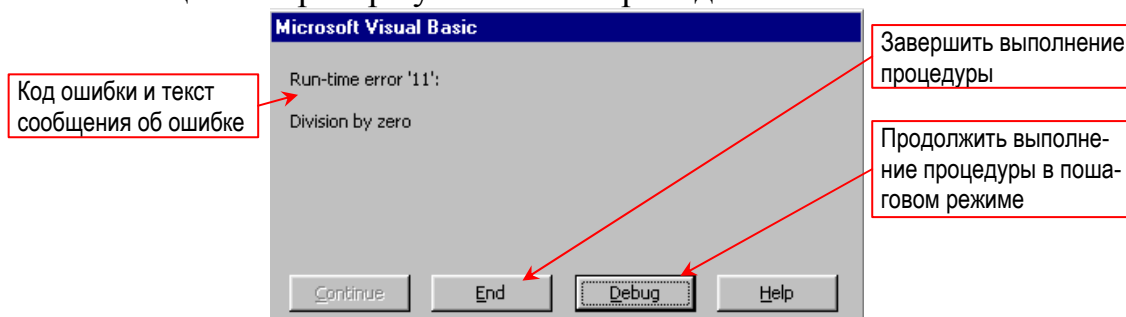


Рис. 9. Сообщение об ошибке при выполнении процедуры

Например, если значение ускорения свободного падения в E8 будет равно 0, то при расчете времени движения до максимальной высоты по формуле $T_H = \frac{V_0 \sin \alpha_2}{g}$ в ячейке E14 (содержащей формулу) появится стандартное сооб-

щение об ошибке #ДЕЛ/0!, а в ячейке F14 (содержащей вызов функции) – сообщение об ошибке #ЗНАЧ!. И при выполнении макроса, назначенного соответствующей формуле-объекту, также будет выдано сообщение об ошибке (Рис. 9).

Чтобы стандартное сообщение об ошибке заменить своим, в E14 введите новую формулу: =ЕСЛИ(E8=0;"Деление на 0!";(E6*SIN(E13))/E8).

А процедуры TH_fun и TH_sub дополните оператором условного перехода If:

```
Public Function TH_fun(V0, a2, g)
    If g = 0 Then
        TH_fun = "Деление на 0!"
    Else
        TH_fun = V0 * Sin(a2) / g
    End If
End Function

Public Sub TH_sub()
    V0 = Range("E6")
    a2 = Range("E13")
    g = Range("E8")
    If g = 0 Then
        Range("G14") = "Деление на 0!"
        ' MsgBox "Деление на 0!"           ' - Вариант
    Else
        Range("G14") = V0 * Sin(a2) / g
    End If
End Sub
```

Введите 0 в E8 – в результате в E14, F14, G14 запишется Деление на 0!.

В таком виде процедуры вычисления T_n реализуют алгоритм условного перехода. Блок-схема этого алгоритма представлена на Рис. 8 справа.

4.3. Максимальная высота $H = H_0 + \frac{V_0^2 \sin^2 \alpha_2}{2g}$

4.3.1. Формула

В ячейку E15 вставьте формулу: =E5+(E6^2*SIN(E13)^2)/(2*E8).

4.3.2. Функция

Введите функцию в Module1 и в F15 вставьте формулу: =H_fun(E5;E6;F13;E8).

```
Public Function H_fun(H0, V0, a2, g)
    H_fun = H0 + (V0 ^ 2 * Sin(a2) ^ 2) / (2 * g)
End Function
```

4.3.3. Макрос

Введите макрос и назначьте его соответствующей формуле-объекту:

```
Public Sub H_sub()
    H0 = Range("E5")
    V0 = Range("E6")
    a2 = Range("E13")
    g = Range("E8")
    Range("G15") = H0 + (V0 ^ 2 * Sin(a2) ^ 2) / (2 * g)
End Sub
```

Блок-схема алгоритма вычисления H аналогична блок-схеме алгоритма вычисления α_2 (Рис. 7).

4.4. Общее время движения $T_L = \frac{V_0 \sin \alpha_2 + \sqrt{V_0^2 \sin^2 \alpha_2 + 2gH_0}}{g}$

4.4.1. Формула

Формула ячейки E16: =(E6*SIN(E13)+КОРЕНЬ(E6^2*SIN(E13)^2+2*E8*E5))/E8.

4.4.2. Функция

Введите функцию в Module1 и в F16 вставьте формулу: =TL_fun(E5;E6;F13;E8).

```
Public Function TL_fun(H0, V0, a2, g)
    TL_fun = (V0 * Sin(a2) + _
        Sqr(V0 ^ 2 * Sin(a2) ^ 2 + 2 * g * H0)) / g
End Function
```

4.4.3. Макрос

Введите макрос и назначьте его соответствующей формуле-объекту:

```
Public Sub TL_sub()
    H0 = Range("E5")
    V0 = Range("E6")
    a2 = Range("E13")
    g = Range("E8")
    Range("G16") = (V0 * Sin(a2) + _
        Sqr(V0 ^ 2 * Sin(a2) ^ 2 + 2 * g * H0)) / g
End Sub
```

Блок-схема алгоритма вычисления T_L аналогична блок-схеме алгоритма вычисления α_2 (Рис. 7).

4.5. Дальность полета $L = V_0 T_L \cos \alpha_2$

4.5.1. Формула

Формула ячейки E17: =E6*E16*COS(E13).

4.5.2. Функция

Введите функцию в Module1 и в F17 вставьте формулу: =L_fun(E6;F16;F13).

```
Public Function L_fun(V0, TL, a2)
    L_fun = V0 * TL * Cos(a2)
End Function
```

4.5.3. Макрос

Введите макрос и назначьте его соответствующей формуле-объекту:

```
Public Sub L_sub()
    V0 = Range("E6")
    TL = Range("G16")
    a2 = Range("E13")
    Range("G17") = V0 * TL * Cos(a2)
End Sub
```

Блок-схема алгоритма вычисления L аналогична блок-схеме алгоритма вычисления α_2 (Рис. 7).

5. Получение файла с таблицей, просмотр и чтение файла

5.1. Построение таблицы зависимости параметров движения тела от Угла броска a_1 , и запись таблицы в файл *Tabl.txt*

В модуль Module1 введите процедуру:

```
Public Sub ПостроениеТаблицы()  
    Кнопка = MsgBox("Продолжить?", vbYesNo + vbQuestion, _  
                    "Построение таблицы")  
  
    If Кнопка = vbNo Then Exit Sub  
    n = InputBox("Задайте начальное значение a1")  
    k = InputBox("Задайте конечное значение a1")  
    s = InputBox("Задайте шаг изменения a1")  
    n = CDBl(n)  
    k = CDBl(k)  
    s = CDBl(s)  
    H0 = Range("E5")  
    V0 = Range("E6")  
    g = Range("E8")  
    Open "Tabl.txt" For Output As #1  
    Write #1, "a1", "a2", "TH", "H", "TL", "L"  
    For a1 = n To k Step s  
        a2 = a2_fun(a1)  
        TH = TH_fun(V0, a2, g)  
        H = H_fun(H0, V0, a2, g)  
        TL = TL_fun(H0, V0, a2, g)  
        L = L_fun(V0, TL, a2)  
        Write #1, a1, a2, TH, H, TL, L  
    Next a1  
    Close #1  
    MsgBox "Таблица записана в файл Tabl.txt", vbInformation, _  
        "Построение таблицы"  
  
End Sub
```

Команда

```
Кнопка = MsgBox("Продолжить?", vbYesNo + vbQuestion, _  
                "Построение таблицы")
```

отобразит диалоговое окно, как на Рис. 10.

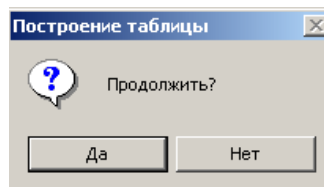


Рис. 10. Диалоговое окно MsgBox

Значение первого аргумента "Продолжить?" будет отображено в диалоговом окне. Значение второго аргумента `vbYesNo + vbQuestion` задает комбинацию кнопок и значок: `vbYesNo` – кнопки **Да** и **Нет**, `vbQuestion` – значок. Значение третьего аргумента "Построение таблицы" отображается в заголовке окна. Переменной `Кнопка` присвоится значение, соответствующее нажатой кнопке.

Оператор условного перехода

```
If Кнопка = vbNo Then Exit Sub
```

проверяет, какая кнопка была нажата. При нажатии Нет (vbNo) выполнится команда Exit Sub, осуществляющая выход из процедуры. При нажатии Да выполнение процедуры продолжится.

С помощью функции InputBox вводятся параметры изменения a_1 :

```
n = InputBox("Задайте начальное значение a1")  
k = InputBox("Задайте конечное значение a1")  
s = InputBox("Задайте шаг изменения a1")
```

Поскольку функция InputBox принимает с клавиатуры значения в виде строки, эти значения необходимо преобразовать в числовой тип:

```
n = CDbl(n)  
k = CDbl(k)  
s = CDbl(s)
```

Другие исходные параметры считываются из соответствующих ячеек:

```
H0 = Range("E5")  
V0 = Range("E6")  
g = Range("E8")
```

Команда

```
Open "Tabl.txt" For Output As #1
```

откроет файл Tabl.txt в режиме Output (для вывода данных во вновь создаваемый файл) и присвоит ему номер 1 (в дальнейшем обращение к файлу будет осуществляться по этому номеру). Файл будет сохранен в папке Мои документы. Для сохранения его в другом месте, это место надо указать, например, так: "C:\Папка\Tabl.txt".

Команда

```
Write #1, "a1", "a2", "TH", "H", "TL", "L"
```

выводит в файл заголовки таблицы.

В цикле

```
For a1 = n To k Step s  
    a2 = a2_fun(a1)  
    TH = TH_fun(V0, a2, g)  
    H = H_fun(H0, V0, a2, g)  
    TL = TL_fun(H0, V0, a2, g)  
    L = L_fun(V0, TL, a2)  
    Write #1, a1, a2, TH, H, TL, L  
Next a1
```

для каждого значения переменной a_1 , изменяющейся от n до k с шагом s , вычисляются значения других переменных (путем вызова ранее созданных функций) и затем оператором Write выводятся в файл.

Команда Close #1 закрывает файл.


А команда

```
MsgBox "Таблица записана в файл Tabl.txt", vbInformation, _  
        "Построение таблицы"
```

выводит сообщение об окончании работы.

Процедура ПостроениеТаблицы реализует циклический алгоритм типа «цикл со счетчиком». Он используется, когда заранее известно, сколько раз надо выполнить цикл. Блок-схема алгоритма представлена на Рис. 11 слева.

Для вызова этой процедуры на рабочем листе Расчетный лист создайте кноп-

ку (элемент  панели инструментов Формы) и назначьте ей макрос ПостроениеТаблицы. На кнопке введите текст: Построение таблицы зависимости параметров движения тела от угла броска a1 и запись таблицы в файл Tabl.txt (см. Рис. 2).

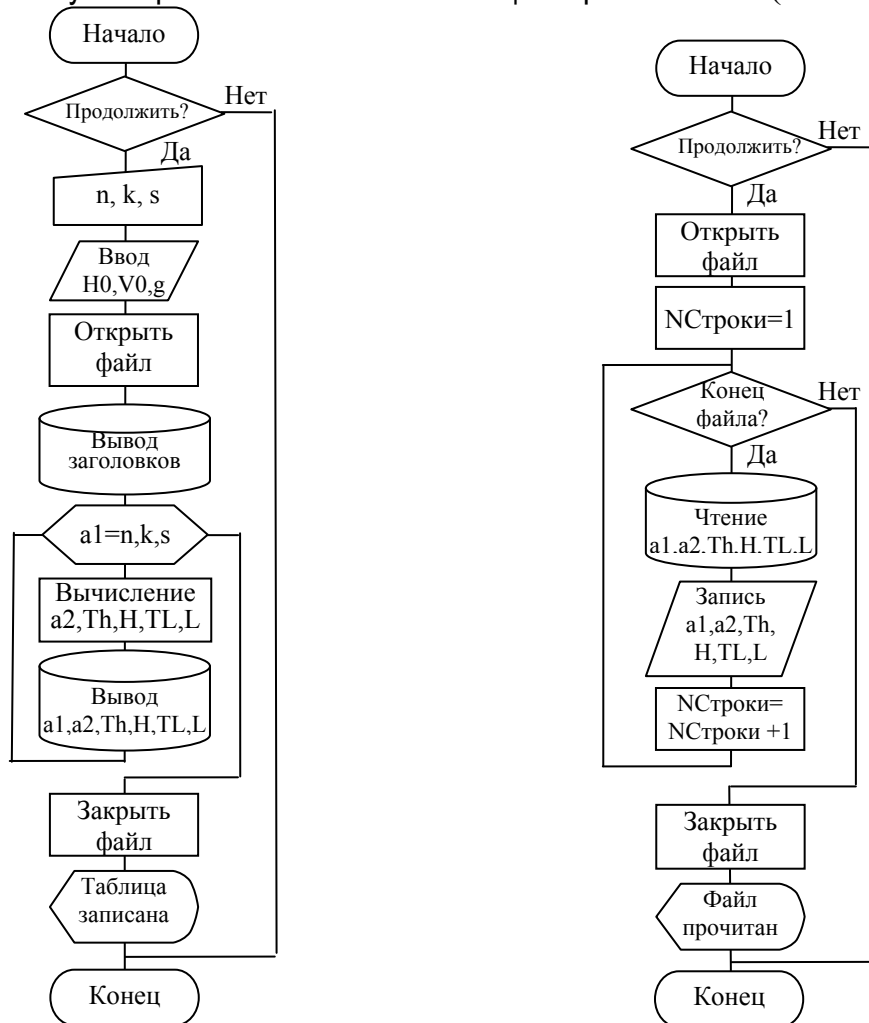


Рис. 11. Блок-схемы процедуры ПостроениеТаблицы (слева) и ЧтениеФайла (справа)

Выполните процедуру ПостроениеТаблицы, задав параметры изменения Угла броска в градусах a_1 : начальное значение 0, конечное 90, шаг 1. После выполнения процедуры в папке Мои документы должен появиться файл Tabl.txt (Рис. 12).

5.2. Просмотр файла Tabl.txt в Блокноте

В модуль Module1 введите процедуру:

```
Public Sub ПросмотрФайла()
    Shell "notepad.exe Tabl.txt", vbNormalFocus
End Sub
```

Стандартная процедура Shell выполняет команду notepad.exe, запускающую Блокнот. Аргументом команды является имя файла Tabl.txt, он будет открыт в Блокноте. Значение аргумента vbNormalFocus означает открыть Блокнот в обычном окне (не в свернутом и не в распахнутом на весь экран).

Для вызова этой процедуры создайте кнопку и назначьте ей макрос ПросмотрФайла. Текст на кнопке: Просмотр файла Tabl.txt в Блокноте (см. Рис. 2).

Выполните процедуру ПросмотрФайла – откроется Блокнот с файлом Tabl.txt, примерно как на Рис. 12.

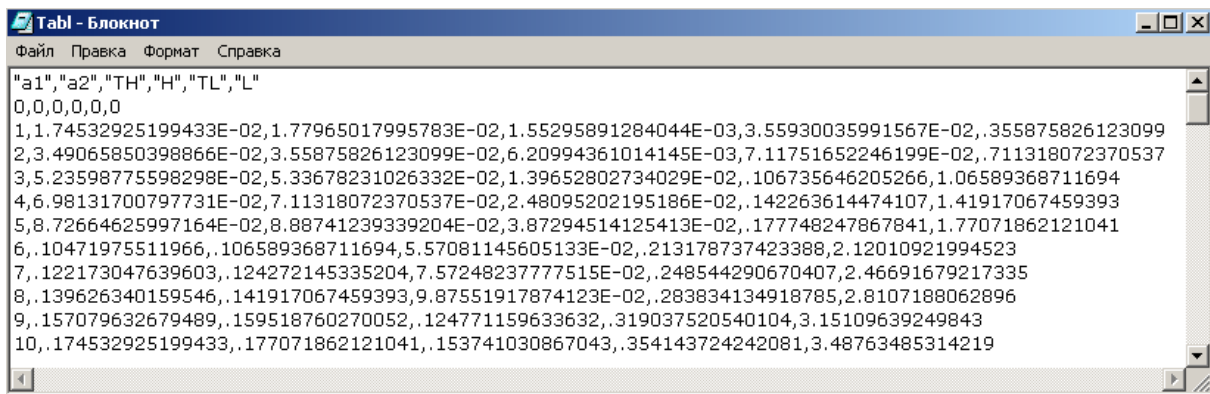


Рис. 12. Файл Tabl.txt, открытый в Блокноте

5.3. Чтение файла Tabl.txt на лист Таблица

В модуль Module1 введите процедуру:

```
Public Sub ЧтениеФайла()
    Кнопка = MsgBox("Продолжить?", vbYesNo + vbQuestion, _
        "Чтение файла")

    If Кнопка = vbNo Then Exit Sub
    Open "Tabl.txt" For Input As #1
    NСтроки = 1
    Do While Not EOF(1)
        Input #1, a1, a2, ТН, Н, ТЛ, L
        Worksheets("Таблица").Cells(NСтроки, 1) = a1
        Worksheets("Таблица").Cells(NСтроки, 2) = a2
        Worksheets("Таблица").Cells(NСтроки, 3) = ТН
        Worksheets("Таблица").Cells(NСтроки, 4) = Н
        Worksheets("Таблица").Cells(NСтроки, 5) = ТЛ
        Worksheets("Таблица").Cells(NСтроки, 6) = L
        NСтроки = NСтроки + 1
    Loop
    Close #1
    MsgBox "Файл Tabl.txt прочитан и записан на лист Таблица!", _
        vbInformation, "Чтение файла"
End Sub
```

Команды

```
Кнопка = MsgBox("Продолжить?", vbYesNo + vbQuestion, _
    "Чтение файла")
```

```
If Кнопка = vbNo Then Exit Sub
```

обеспечивают возможность продолжить выполнение, или отказаться.

Команда

```
Open "Tabl.txt" For Input As #1
```

открывает файл Tabl.txt в режиме Input (ввод из файла, то есть чтение файла) и присваивает ему номер 1.

Переменная NСтроки будет отслеживать номер строки рабочего листа, в которую будет записываться очередная строка значений таблицы из файла.

Команды, находящиеся в теле цикла, то есть между командами

```
Do While Not EOF(1)
```

и

```
Loop
```

будут выполняться, пока не закончится файл, то есть пока будет истинно выражение `Not EOF(1)`. Стандартная функция `EOF(1)` возвращает логическое значение ИСТИНА, если файл (с номером 1) закончился, и ЛОЖЬ, если нет. А логическая операция `NOT` инвертирует это значение.

Значения очередной строки таблицы читаются из файла и записываются в указанные переменные по команде

```
Input #1, a1, a2, TH, H, TL, L
```

Команды

```
Worksheets("Таблица").Cells(NСтроки, 1) = a1
Worksheets("Таблица").Cells(NСтроки, 2) = a2
Worksheets("Таблица").Cells(NСтроки, 3) = TH
Worksheets("Таблица").Cells(NСтроки, 4) = H
Worksheets("Таблица").Cells(NСтроки, 5) = TL
Worksheets("Таблица").Cells(NСтроки, 6) = L
```

записывают значения переменных на рабочий лист Таблица в ячейки, стоящие на пересечении строки с номером `NСтроки` и столбца 1-го, 2-го, ..., 6-го.

По команде

```
NСтроки = NСтроки + 1
```

значение номера строки увеличивается на 1.

После этого опять осуществляется проверка, не закончился ли файл, и если нет, цикл повторяется.

Когда файл будет прочитан весь, он будет закрыт по команде `Close #1`.

А команда

```
MsgBox "Файл Tabl.txt прочитан и записан на лист Таблица!", _
vbInformation, "Чтение файла"
```

выведет сообщение об окончании работы.

Процедура `ЧтениеФайла` реализует циклический алгоритм типа «цикл пока». В отличие от «цикла со счетчиком», здесь заранее не известно, сколько раз выполнится цикл. Выполнение будет происходить, пока выполняется заданное условие, в нашем примере – пока не закончится файл. Блок-схема алгоритма представлена на Рис. 11 справа.

Для вызова этой процедуры создайте кнопку и назначьте ей макрос `ЧтениеФайла`. Текст на кнопке: `Чтение файла Tabl.txt на лист Таблица` (см. Рис. 2).

Выполните процедуру `ЧтениеФайла`. В результате на листе Таблица, появятся данные, прочитанные из файла `Tabl.txt` (Рис. 13).

	A	B	C	D	E	F	
1	a1	a2	TH	H	TL	L	
2	0	0	0	0	0	0	
3	1	0,017453	0,017797	0,001553	0,035593	0,355876	
4	2	0,034907	0,035588	0,00621	0,071175	0,711318	
5	3	0,05236	0,053368	0,013965	0,106736	1,065894	
6	4	0,069813	0,071132	0,02481	0,142264	1,419171	
7	5	0,087266	0,088874	0,038729	0,177748	1,770719	
8	6	0,10472	0,106589	0,055708	0,213179	2,120109	
9	7	0,122173	0,124272	0,075725	0,248544	2,466917	
10	8	0,139626	0,141917	0,098755	0,283834	2,810719	
11	9	0,15708	0,159519	0,124771	0,319038	3,151096	
12	10	0,174533	0,177072	0,153741	0,354144	3,487635	

Рис. 13. Фрагмент рабочего листа Таблица с данными, прочитанными из файла `Tabl.txt`

6. Создание формы для работы с таблицей

6.1. Создание формы

Для запуска процедур ПостроениеТаблицы, ПросмотрФайла и ЧтениеФайла мы использовали кнопки на рабочем листе. Рассмотрим вариант с пользовательской формой. На форме мы разместим поля ввода значений исходных параметров и кнопки для запуска процедур – так что форма станет своего рода пультом управления для работы с таблицей и файлом (Рис. 14).

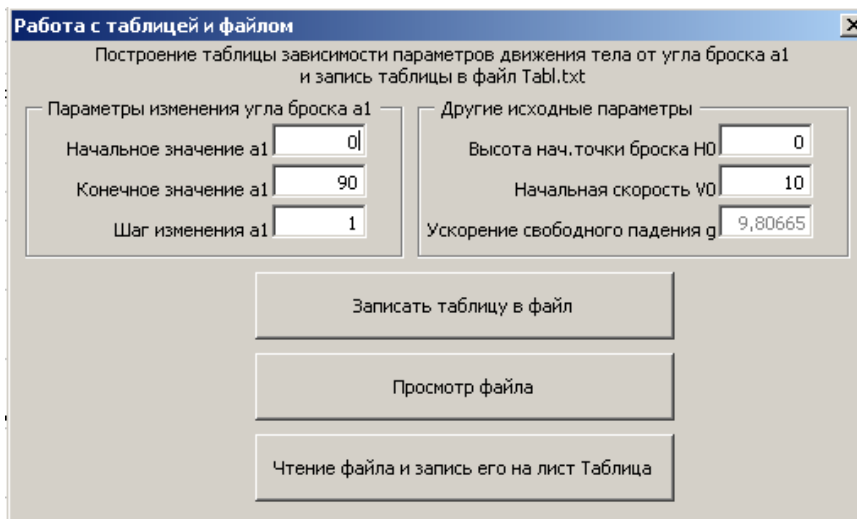


Рис. 14. Пользовательская форма для работы с таблицей и файлом (этап выполнения)

Для создания формы перейдите в Редактор Visual Basic и вставьте модуль формы: меню Insert–UserForm. Появится форма UserForm1 – Рис. 15.

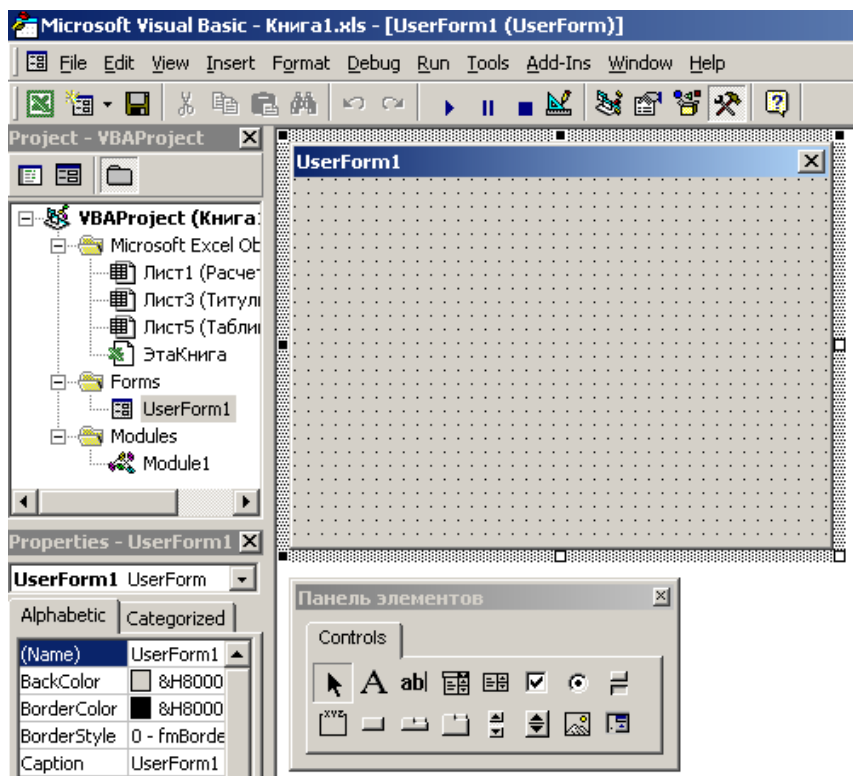



Рис. 15. Новая пользовательская форма




Запустите её на выполнение (команда -Run или клавиша F5). Появится окно пустой формы с заголовком UserForm1. Пока эта форма бесполезна. Так что закройте её и вернитесь к проектированию.

Сама форма и элементы, размещаемые на ней, обладают определенными свойствами, значения которых можно изменять в окне Properties (Рис. 15). Открывается окно свойств через меню View (или клавишей F4).

Для свойства Caption (заголовок формы) вместо UserForm1 введите новое значение: Работа с таблицей и файлом.

Для вставки элементов в форму служит Панель элементов (Toolbox) (Рис. 15); она появляется автоматически при выделении формы.

Мы будем использовать следующие элементы:

 A-Label – Надпись	для отображения текста на форме
 -Frame – Рамка	для группировки элементов на форме
 a6-TextBox – Поле ввода	для ввода данных

Вставьте элементы в форму, как на Рис. 16.

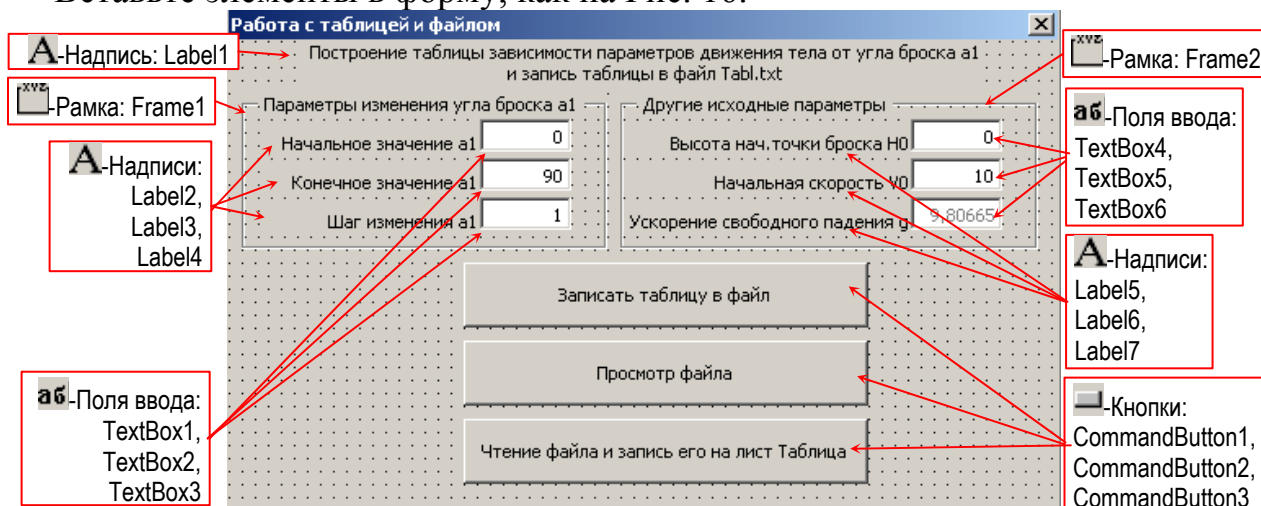


Рис. 16. Пользовательская форма для работы с таблицей и файлом (этап проектирования)

Имейте в виду: вставляемому элементу автоматически присваивается имя, зависящее от порядка ввода, например: Label1, Label2, Label3 и т.д. Так что возможно, у вас получатся имена с другими номерами.

Свойства формы и свойства вставляемых элементов задайте на свое усмотрение, используя окно Properties.

Вот некоторые общие свойства:


Name	Имя элемента, имя формы
Caption	Текст на элементе, текст заголовка формы
TextAlign	Выравнивание текста
Font	Параметры шрифта: тип, начертание, размер
ForeColor	Цвет текста
BackColor	Цвет фона

Для полей ввода (элементы TextBox) будем использовать также свойства:

Value	Значение, введенное в поле
ControlSource	Содержит имя ячейки, с которой синхронизируется значение поля
Enabled	True – элемент доступен для изменения; False – нет


Свойство Value служит для хранения значения, введенного в поле. Значения некоторых полей для удобства можно ввести сразу, то есть на этапе проектирования, в качестве «значений по умолчанию». Например, начальное и конечное значение и шаг изменения угла броска α_1 известны заранее, и поэтому свойству Value для TextBox1, TextBox2 и TextBox3 присвойте соответственно: 0, 90 и 1.

В качестве значения свойства ControlSource задается (при необходимости) имя ячейки рабочего листа, с которой надо синхронизировать значение поля. Для TextBox4 (Высота начальной точки броска H_0) в свойство ControlSource впишите E5; для TextBox5 (Начальная скорость V_0) – E6; для TextBox6 (Ускорение свободного падения g) – E8.

Запустите форму на выполнение (команда -Run или клавиша F5) и убедитесь: значения заданных ячеек автоматически вписываются в соответствующие поля ввода на форме, и изменение значений полей влечет за собой изменение значений ячеек. (Речь идет о ячейках активного листа).

Чтобы предотвратить возможность изменения уже заданного значения Ускорения свободного падения g , свойству Enabled поля TextBox6 присвойте значение False – поле станет недоступным для изменения (и цвет его поблекнет).

Теперь о кнопках на форме – элементы CommandButton.

Мы должны связать событие Click (Щелчок по кнопке) с процедурой, обрабатывающей это событие. В контекстном меню кнопки CommandButton1 выберите пункт -View Code – откроется окно кода формы UserForm1(Code) (Рис. 17), и в нем будет создана процедура обработки клика по кнопке. В эту процедуру введите команду вызова ранее созданной процедуры ПостроениеТаблицы:

```
Private Sub CommandButton1_Click()  
    ПостроениеТаблицы  
End Sub
```

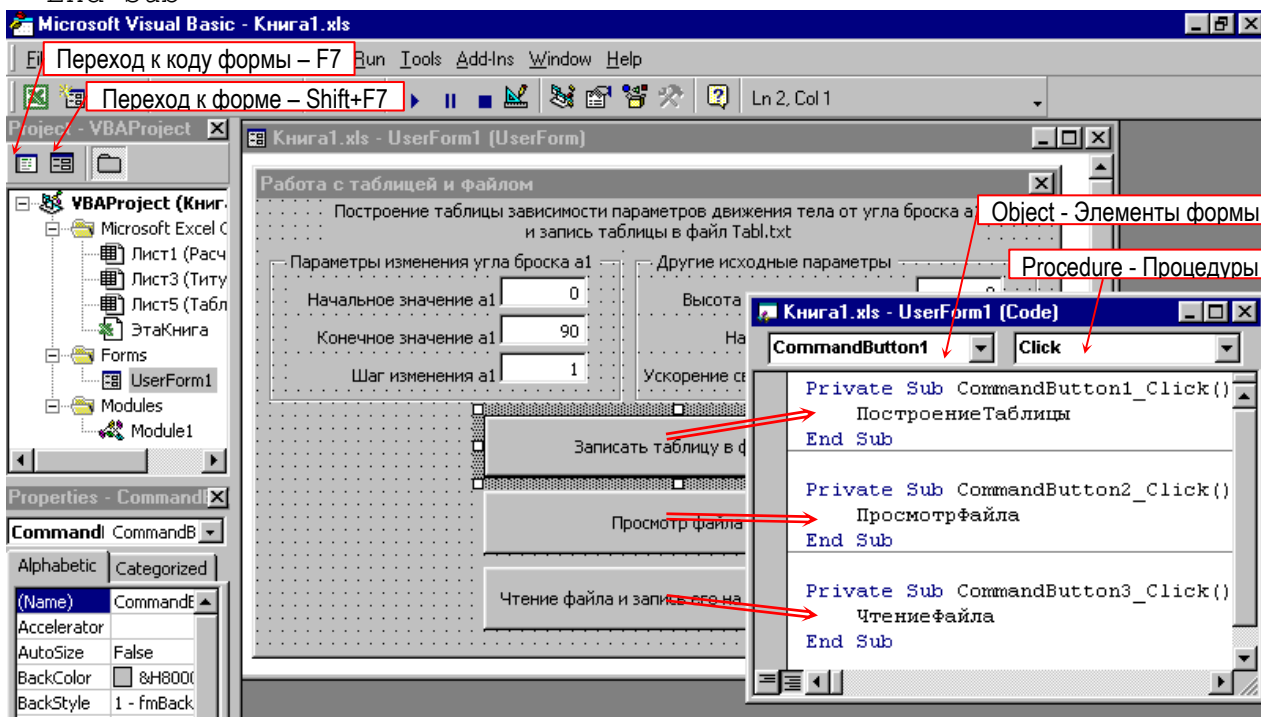


Рис. 17. Окно проектирования формы (UserForm) и окно кода формы (Code)

Таким же образом создайте процедуры для двух других кнопок (Рис. 17):

```

Private Sub CommandButton2_Click()
    ПросмотрФайла
End Sub
Private Sub CommandButton3_Click()
    ЧтениеФайла
End Sub

```

Для перехода от формы к коду и обратно использовать можно не только контекстное меню, но и команды меню View, или кнопки окна проекта Project (Рис. 17), или клавиши F7 и Shift+F7. Сдвоенный щелчок по форме или по её элементу также осуществляет переход в окно кода.

Замечание. Переход от формы к коду сопровождается созданием процедуры, и если мы не собирались этого делать, её придется удалить «вручную». Если же процедура была создана раньше, то осуществляется переход к ней.

По умолчанию создается процедура обработки клика (Click). Для создания процедуры обработки другого события надо обратиться в поле Procedure вверху окна кода (Рис. 17). Поле Object служит для выбора элемента формы.

Чтобы иметь возможность запускать процедуру ПостроениеТаблицы как с помощью кнопки на рабочем листе так и с помощью формы, вместо команд

```

n = InputBox("Задайте начальное значение a1")
k = InputBox("Задайте конечное значение a1")
s = InputBox("Задайте шаг изменения a1")

```

в этой процедуре введите условный оператор:

```

If UserForms.Count = 0 Then
    n = InputBox("Задайте начальное значение a1")
    k = InputBox("Задайте конечное значение a1")
    s = InputBox("Задайте шаг изменения a1")
Else
    n = UserForm1.TextBox1.Value
    k = UserForm1.TextBox2.Value
    s = UserForm1.TextBox3.Value
End If

```

Свойство Count объекта UserForms возвращает количество открытых пользовательских форм. Если открытых форм 0, значит, запуск процедуры осуществляется не из формы, а с помощью кнопки на рабочем листе, и потому значения переменных n, k, s будут вводиться с клавиатуры с помощью InputBox; иначе переменным n, k, s надо присвоить значение свойства Value элементов TextBox1, TextBox2, TextBox3 формы UserForm1.

6.2. Запуск формы

Обеспечим запуск формы из рабочего листа.

Сначала в модуле Module1 вставьте процедуру, в которой осуществляется вызов стандартной процедуры Show, связанной с формой UserForm1 (по другому: к объекту UserForm1 применяется метод Show):

```

Public Sub ВызовФормы()
    UserForm1.Show
End Sub

```


На рабочем листе Расчетный лист вставьте кнопку с надписью Вызов формы для построения таблицы (см. Рис. 2), и назначьте ей макрос ВызовФормы.

Проверьте работу формы.

7. Построение графика


На основе данных на листе Таблица надо построить график зависимости одного параметра от другого, например, a_2 от a_1 . График должен строиться с помощью макроса. Сам макрос получим с помощью Создателя макросов.

Необходимо также обеспечить автоматизацию выбора графика зависимости любого из параметров от a_1 .

7.1. Использование Создателя макросов

Создатель макросов (его еще называют Рекордером) это средство автоматизации создания макросов путем записи действий пользователя в виде команд VBA.

Перейдите на рабочий лист Таблица. Перед тем как воспользоваться Создателем макросов, «прорепетируйте»: постройте с помощью Мастера диаграмм график зависимости a_2 (второй столбец таблицы) от a_1 (первый столбец).

Запустите Создатель макросов командой -Начать запись (меню Сервис-Макрос или панели инструментов Visual Basic).

В появившемся диалоговом окне Запись макроса введите его имя: ПостроениеДиаграммы (параметры Сочетание клавиш и Описание можно оставить без изменений), и нажмите ОК. Автоматически появится панель Остановить запись.

С этого момента все ваши действия будут записываться в макрос!

Запустите Мастер диаграмм, и на 1 шаге выберите тип диаграммы График.

На 2 шаге задайте диапазон данных, выделив второй столбец таблицы B1:B92. Задайте: Ряды в столбцах.

На вкладке Ряды задайте подписи оси X, выделив первый столбец таблицы, начиная со второй ячейки: A2:A92.

На 3 шаге на вкладке Легенда уберите флажок Добавить легенду.

На 4 шаге поместите диаграмму на имеющемся листе Таблица. Готово.

После появления диаграммы нажмите кнопку -Остановить запись.

Чтобы посмотреть результат, перейдите в Редактор Visual Basic. Обратите внимание: в окне проекта Project появился еще один модуль – модуль Module2, он создан Создателем макросов. Откройте его (сдвоенным щелчком) и убедитесь, что в нем записан макрос:

```
Sub ПостроениеДиаграммы()  
    ' <описание макроса - текст, задаваемый в окне Запись макроса>  
    Charts.Add  
    ActiveChart.ChartType = xlLineMarkers  
    ActiveChart.SetSourceData _  
        Source:=Sheets("Таблица").Range("B1:B92"), PlotBy:=xlColumns  
    ActiveChart.SeriesCollection(1).Xvalues = "=Таблица!R2C1:R92C1"  
    ActiveChart.Location Where:=xlLocationAsObject, Name:="Таблица"  
    ActiveChart.HasLegend = False  
End Sub
```


Команда `Charts.Add` создает диаграмму.

```
ActiveChart.ChartType = xlLineMarkers
```

– задает тип диаграммы График.

```
ActiveChart.SetSourceData
```

```
Source:=Sheets("Таблица").Range("B1:B92"), PlotBy:=xlColumns
```

– диапазон с исходными данными, ряды в столбцах.

```
ActiveChart.SeriesCollection(1).XValues = "=Таблица!R2C1:R92C1"
```

– диапазон с подписями по X (стиль ссылок R1C1).

```
ActiveChart.Location Where:=xlLocationAsObject, Name:="Таблица"
```


– место размещения диаграммы.

```
ActiveChart.HasLegend = False
```

– убрать легенду.

Так что Создатель макросов можно использовать в целях изучения языка VBA.

Обеспечим запуск полученного макроса из рабочего листа.

На рабочем листе Таблица вставьте кнопку (элемент  панели инструментов Формы) с надписью Построить график зависимости a2 от a1 (см. Рис. 18), и назначьте ей макрос ПостроениеДиаграммы.

Проверьте: щелчок по кнопке добавит новую диаграмму с графиком. Таким образом, мы автоматизировали выполнение всех действий, которые выполняли с помощью Мастера диаграмм.

Если при выполнении макроса возникли ошибки, удалите макрос (меню Сервис-Макрос-Макросы, кнопка Удалить; или непосредственно в Редакторе Visual Basic), и попытайтесь создать его ещё раз. При записи макроса не выполняйте лишних действий, например, не перемещайте и не меняйте размеры диаграммы.

7.2. Автоматизация выбора графика

Мы получили график одного параметра – a2. Теперь обеспечим получение графика любого из параметров: a2, Tн, H, Tл, L – на выбор.

Для каждого параметра создайте процедуру, корректирующую диаграмму с графиком, а именно переопределяющую диапазоны с исходными значениями и подписями по X. Процедуры создавайте в модуле Module2 (где уже есть процедура ПостроениеДиаграммы). Поскольку процедуры очень похожи, воспользуйтесь буфером обмена с последующей корректировкой.

```
Sub График_a2()
```

```
ActiveChart.SetSourceData
```

```
Source:=Sheets("Таблица").Range("B1:B92"), PlotBy:=xlColumns
```

```
ActiveChart.SeriesCollection(1).XValues = "=Таблица!R2C1:R92C1"
```

```
End Sub
```

```
Sub График_ТН()
```

```
ActiveChart.SetSourceData
```

```
Source:=Sheets("Таблица").Range("C1:C92"), PlotBy:=xlColumns
```

```
ActiveChart.SeriesCollection(1).XValues = "=Таблица!R2C1:R92C1"
```

```
End Sub
```

```
Sub График_H()
```

```
ActiveChart.SetSourceData
```

```
Source:=Sheets("Таблица").Range("D1:D92"), PlotBy:=xlColumns
```


```
ActiveChart.SeriesCollection(1).XValues = "=Таблица!R2C1:R92C1"
```

```
End Sub
```

```

Sub График_TL()
    ActiveChart.SetSourceData _
        Source:=Sheets("Таблица").Range("E1:E92"), PlotBy:=xlColumns
    ActiveChart.SeriesCollection(1).XValues = "=Таблица!R2C1:R92C1"
End Sub
Sub График_L()
    ActiveChart.SetSourceData _
        Source:=Sheets("Таблица").Range("F1:F92"), PlotBy:=xlColumns
    ActiveChart.SeriesCollection(1).XValues = "=Таблица!R2C1:R92C1"
End Sub

```

Выбор, какую процедуру выполнять, обеспечим с помощью элемента -Список панели инструментов Формы. Вставьте его на листе Таблица (Рис. 18).

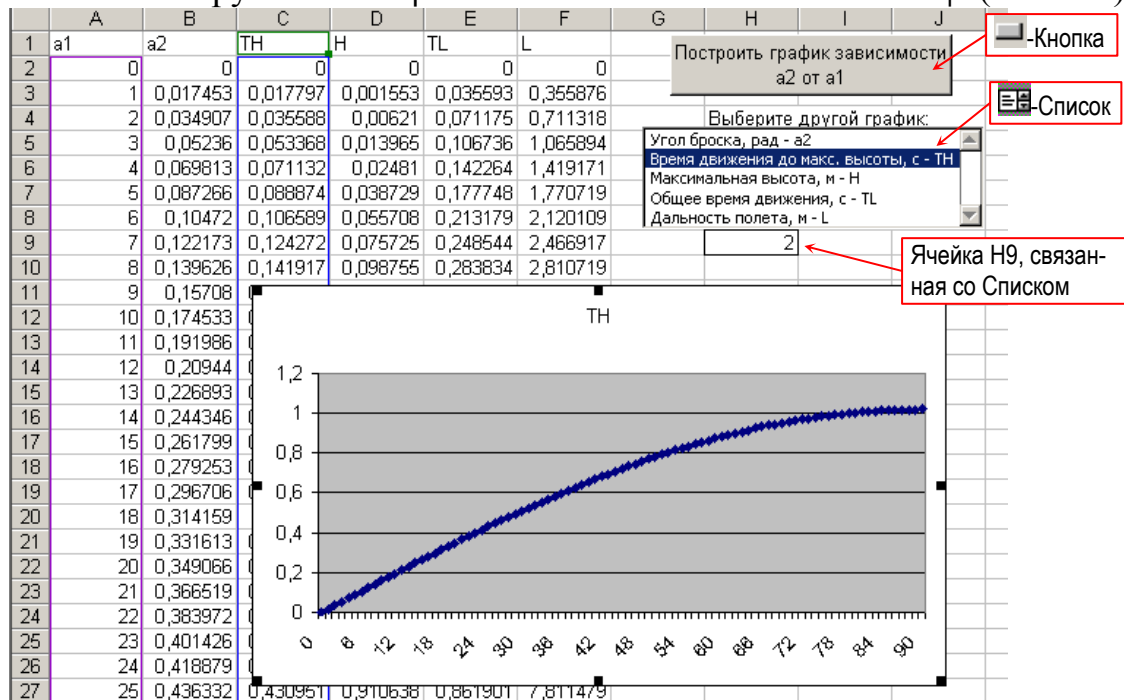


Рис. 18. Автоматизация построения графика с возможностью выбора (рабочий лист Таблица)

Вызовите Формат элемента управления. Для задания значения параметра Формировать список по диапазону выделите D13:D17 на Расчетном листе (в этом диапазоне названия параметров, ими будет заполняться список). Значение параметра Связь с ячейкой – H9 (в неё будет заноситься номер выбранного пункта списка).

В модуле Module2 введите процедуру ВыборГрафика:

```

Sub ВыборГрафика()
    Select Case Range("H9")
        Case "1"
            График_a2
        Case "2"
            График_TH
        Case "3"
            График_H
        Case "4"
            График_TL
        Case "5"
            График_L
    End Select
End Sub

```

– с помощью оператора выбора `Select Case` значение ячейки Н9 сравнивается с одним из возможных номеров выбранного пункта списка "1", "2", "3", "4", "5", и затем запускается соответствующая процедура корректировки диаграммы.

Через контекстное меню Списка назначьте ему макрос `ВыборГрафика`.

Убедитесь: выбор графика в списке отражается на диаграмме.

Мы заменили несколько стандартных действий по корректировке диаграммы одним щелчком мыши.

Замечание. Диаграмма должна быть выделена перед выбором графика, иначе будет выдано сообщение об ошибке, так как наши процедуры не содержат команд для предварительного выделения корректируемой диаграммы.

8. Процедура выдачи информации о курсовой работе

8.1. Создание процедуры Тема

Перейдите в Редактор Visual Basic, и в модуле `Module1` вставьте процедуру:

```
Public Sub Тема()  
    MsgBox "РАСЧЕТ ПАРАМЕТРОВ ДВИЖЕНИЯ ТЕЛА," & vbNewLine & _  
        "БРОШЕННОГО ПОД УГЛОМ К ГОРИЗОНТУ" & vbNewLine & vbNewLine & _  
        "Студент: Иванов И.И." & vbNewLine & _  
        "Группа: ЭУ-18092" & vbNewLine & _  
        "Учебный год: 2008-2009", _  
        vbInformation, _  
        "КУРСОВАЯ РАБОТА по ИНФОРМАТИКЕ"  
End Sub
```

Здесь вызывается стандартная процедура вывода `MsgBox` с тремя аргументами. Каждая команда VBA должна занимать одну строку. Команда вызова `MsgBox` очень длинная, её пришлось записать в несколько строк, используя символ продолжения команды – знак подчеркивания с пробелом перед ним.

Первый аргумент процедуры `Msgbox` это текст, который будет отображаться в окне. Фрагменты текста связываются операцией сцепления `&`. Для вывода фрагмента с новой строки используется стандартная константа `vbNewLine`.

Второй аргумент со значением `vbInformation` задает значок.

Третий аргумент содержит текст для заголовка окна.

Результат выполнения процедуры `Тема` представлен на Рис. 19.

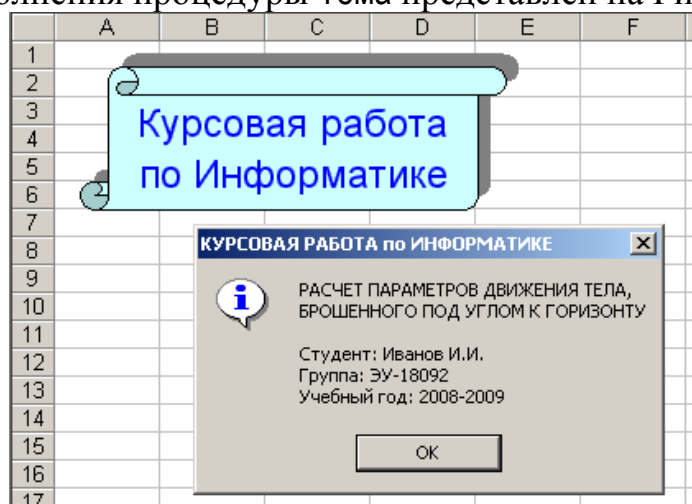


Рис. 19. Фрагмент Титульного листа с автофигурой-кнопкой, вызывающей макрос `Тема`

8.2. Вызов процедуры Тема с помощью кнопки

На Титульном листе вставьте какой-нибудь объект – это может быть иллюстрирующий рисунок, картинка, или (как на Рис. 19) автофигура. И через контекстное меню назначьте ему макрос Тема.

Убедитесь: щелчок по автофигуре запустит макрос.

8.3. Автоматический вызов процедуры Тема

Сделаем так, чтобы открытие книги с курсовой работой сопровождалось автоматическим запуском процедуры Тема.

Перейдите в Редактор Visual Basic, и откройте модуль ЭтаКнига – сдвоенным щелчком в окне проекта Project (Рис. 20).

Процедуры этого модуля (и модулей листов), событийно-ориентированные, то есть запускаются автоматически по наступлению определенного события. Среди прочих событий есть и событие Open – открытие книги.

В поле Object сверху окна модуля выберите Workbook – в результате вставится процедура обработки события Open (это событие принято по умолчанию; другие события можно выбрать в поле Procedure). Вставьте в эту процедуру команду вызова процедуры Тема:

```
Private Sub Workbook_Open()  
    Тема  
End Sub
```

Для проверки закройте книгу с курсовой работой (с сохранением) и откройте снова – автоматически отобразится окно с информацией о курсовой работе.

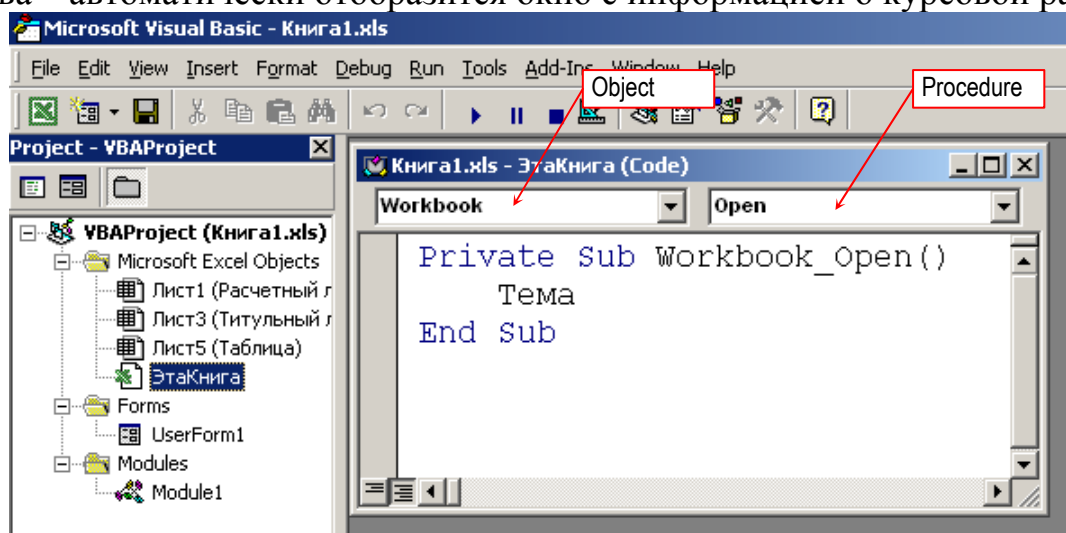


Рис. 20. Окно модуля ЭтаКнига с процедурой, обрабатывающей открытие книги

9. Создание панели инструментов и меню

В дополнение к штатным средствам пользовательского интерфейса (меню, панели инструментов и пр.) в Excel есть возможность создавать специфические средства интерфейса, ориентированные на решение конкретной задачи. К ним относятся объекты, создаваемые с помощью панелей инструментов **Формы** и **Элементы управления**: кнопки, списки, счетчики и пр., – а так же объекты, превращаемые в кнопки путем назначения им макросов: формулы, рисунки и пр.

Примеры их использования мы уже рассматривали. Еще одним элементом интерфейса могут служить пользовательские панели инструментов и меню.

Через меню Сервис-Настройка откройте диалоговое окно Настройка (Рис. 21).

Для создания новой панели надо на вкладке Панели инструментов щелкнуть кнопку Создать и затем задать имя панели.

Для создания нового меню надо на вкладке Команды выделить категорию Новое меню, и перетащить команду Новое меню в область меню.

Для добавления кнопок на панели и в меню надо на вкладке Команды выделить категорию Макросы и перетащить оттуда нужную команду.

Через контекстное меню кнопки можно задать её имя, выбрать или изменить (отредактировать) значок и назначить макрос.

Для удаления надо просто стащить кнопку с панели или из меню.

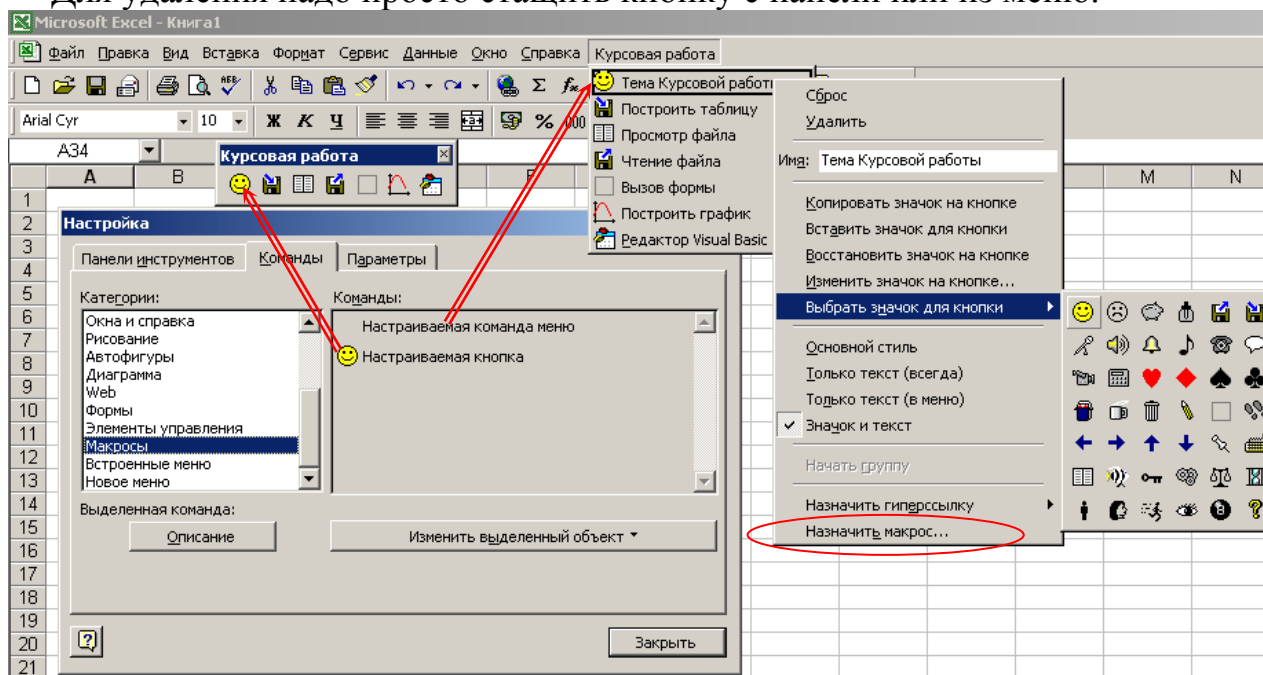


Рис. 21. Создание панели инструментов и меню

Создайте панель инструментов и меню с именами Курсовая работа (Рис. 21), снабдите их кнопками для вызова соответствующих макросов:

Значок и имя кнопки	Назначенный макрос
😊 Тема Курсовой работы	Тема
📊 Построить таблицу	ПостроениеТаблицы
📄 Просмотр файла	ПросмотрФайла
📖 Чтение файла	ЧтениеФайла
📄 Вызов формы	ВызовФормы
📈 Построить график	ПостроениеДиаграммы
🔧 Редактор Visual Basic	

В контекстном меню кнопки Построить график выберите пункт Изменить значок на кнопке, и в Редакторе кнопок создайте значок кнопки, как на Рис. 22.

Для создания кнопки 🗑️-Редактор Visual Basic перетащите соответствующую команду из категории Сервис.

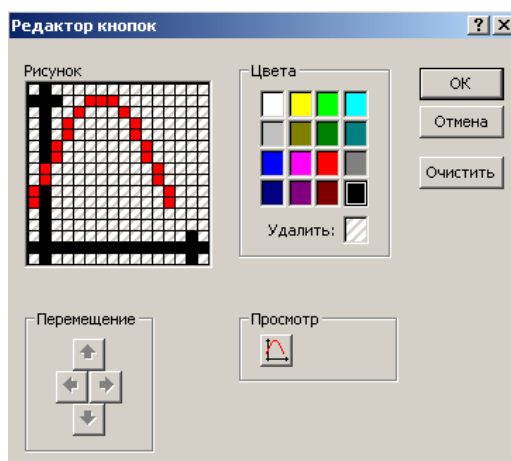


Рис. 22. Редактирование значка кнопки Построить график

10. Приложения

10.1. Требования по оформлению отчета по курсовой работе

Отчет по курсовой работе создается в MS Word, печатается на листах формата А4, и сдается преподавателю вместе с дискетой с Excel-книгой.

Все заголовки в отчете оформлять в стиле Заголовок1, Заголовок2 и т.д.

Текст основного набора Отчета выполнять шрифтом прямого начертания гарнитуры Times New Roman, размером 14пт, интервалом 1.

Формулы (объекты MS Equation 3.0) набираются шрифтом прямого начертания гарнитуры Times New Roman, размером 14пт. Крупный индекс набирается шрифтом 10пт, мелкий – 8пт.

Страницы Отчета пронумеровать.

10.2. Структура отчета по курсовой работе

1. Титульный лист

Пример титульного листа:

Федеральное агентство по образованию
ГОУ ВПО «Уральский Государственный Технический Университет –
УПИ»

КУРСОВАЯ РАБОТА
по информатике

«Программирование базовых алгоритмов на VBA
(MS Excel)»

Вариант 1
«Расчет параметров движения тела, брошенного под
углом к горизонту»

Студент гр. ЭУ-18092ир	Иванов И.И.
Преподаватель	Сысков В.И.

2009 г.

2. Оглавление

Перечень заголовков отчета с указанием номеров страниц.

Рекомендуется использовать команду Оглавление и указатели меню Вставка. (А для этого все заголовки в отчете должны быть оформлены в стиле Заголовок1, Заголовок2 и т.д.).

3. Постановка задачи

Текст задания по курсовой работе.

4. Описание решения задачи

Здесь необходимо дать общее описание решения задачи. Перечень и назначение рабочих листов. Варианты выполнения вычислений. Объекты, элементы управления, размещаемые на рабочих листах; их назначение. Диаграммы. Пользовательские формы. Интерфейс пользователя.

5. Распечатки

5.1 Рабочие листы

Распечатать все рабочие листы книги.

Перед печатью вызовите диалог Параметры страницы (меню Файл) и:

– на вкладке Страница задайте альбомную ориентацию;

– на вкладке Лист включите флажки: ☒-сетка и ☒-заголовки строк и столбцов.

Чтобы объекты рабочих листов выводились на печать, вызовите для них через контекстное меню диалог Формат и на вкладке Свойства включите флажок ☒-выводить объект на печать.

5.2 Расчетный лист с формулами

Через меню Сервис вызовите диалог Параметры и на вкладке Вид включите флажок ☒-формулы – в результате в ячейках будут отображаться не значения, а формулы ячеек. При необходимости измените ширину столбцов.

После печати листа с формулами выключите флажок ☐-формулы.


5.3 Диаграммы (графики)

Чтобы печатался не весь рабочий лист, а только одна диаграмма, размещенная на нем, её надо выделить.

5.4 Файл с таблицей

Печать файла с таблицей удобнее выполнить из Блокнота.

5.5 Пользовательские формы

Для вставки в Отчет пользовательской формы (а также окна с информацией о курсовой работе) отобразите форму и нажмите клавишу Print Screen – содержимое экрана окажется в буфере обмена; в Отчете вставьте рисунок из буфера обмена. Используя команду -Обрезка панели Настройка изображения, оставьте на рисунке только то, что нужно.

5.6 Окно с информацией о курсовой работе

6. Тексты процедур и блок-схемы

Тексты всех процедур должны быть упорядочены и сгруппированы по

рассчитываемым параметрам и по решаемым задачам.

Для вставки процедур в Отчет из Редактора Visual Basic используйте буфер обмена.

Текст процедур сопровождать блок-схемой – не обязательно для всех процедур, но в Отчете должно быть, по крайней мере, четыре блок-схемы, представляющих базовые алгоритмы: 1) линейная последовательность (процедура для одного из параметров); 2) условный переход (процедура для одного из параметров); 3) цикл типа For (процедура ПостроениеТаблицы); 4) цикл типа While (процедура ЧтениеФайла).

Для создания блок-схем используйте панель инструментов Рисование–Автофигуры–Блок-схема.

7. Руководство пользователя по решению задачи

7.1 Техническое и программное обеспечение

Описать технические параметры компьютера (тип процессора, объем ОЗУ, и пр.), тип и версия ОС, версия MS Excel.

7.2 Инструкция по выполнению расчетов

Назначение рабочих листов и объектов на них. Пользовательские панели инструментов и меню, их активизация и использование. Опишите порядок выполнения расчетов. Допустимые исходные данные. Причины возможных ошибок. Действия по устранению ошибок.

8. Выводы

Дать сравнительную характеристику вариантов расчета параметров: достоинства и недостатки использования формул, функций и макросов. Преимущества знания языка программирования VBA. Преимущества использования элементов управления.

10.3. Литература по VBA

1. Туркин О.В. VBA. Практическое программирование. – М.: СОЛОН – ПРЕСС, 2007. – 128 с.
2. Гарнаев А. Ю. Самоучитель VBA. – СПб. : БХВ – Санкт-Петербург, 2000. – 512 с. : ил.
3. Дубина А.Г. Машиностроительные расчеты в среде Excel 97/2000. – СПб.: БХВ – Санкт-Петербург, 2000. – 416 с.
4. Дубина А.Г., Орлова С.С., Шубина И.Ю. MS Excel в электротехнике и электронике. – СПб.: БХВ – Санкт-Петербург, 2001. – 304 с.
5. Ларсен, Рональд, У. Инженерные расчеты в Excel. : Пер. с англ. – М. : Издательский дом «Вильямс», 2002. – 544 с.
6. Уокенбах Джон. Профессиональное программирование на VBA в Excel 2002.: Пер. с англ. – М.: Издательский дом «Вильямс», 2003. – 784 с.
7. Роман Стивен. Использование макросов в Excel. 2-ое изд. – СПб.: Питер, 2004. – 507 с.
8. Гладкий А.А., Чиртик А.А. Excel. Трюки и эффекты. – СПб.: Питер, 2007. – 368 с.

10.4. Ссылки в Интернете

1. Сайт «Office Online» <http://office.microsoft.com/ru-ru/default.aspx> содержит раздел «Обучающие курсы» и среди них следующие курсы:
<http://office.microsoft.com/ru-ru/excel/HA011189581049.aspx?pid=CH062528391049>
– Общие сведения о пользовательских макросах в Microsoft Excel.
<http://office.microsoft.com/training/training.aspx?AssetID=RC011506201049&pid=CR061831141049>
– Использование макросов Microsoft Excel для создания циклов.
2. На сайте «Планета Excel.ru» (<http://www.planetaexcel.ru/index.php>), среди прочей полезной информации, можно найти обучающий курс:
<http://www.planetaexcel.ru/tip.php?aid=84>
– Создание макросов и пользовательских функций на VBA.
3. На сайте «RusFAQ.RU» – <http://rusfaq.ru/> – можно получить бесплатные консультации по VBA: <http://rusfaq.ru/issues/5/3/50>.
4. На сайте «Всё о Visual Basic» – <http://vbnet.ru/> – также можно проконсультироваться на форуме VBA: <http://vbnet.ru/forum/?forumid=6>.
5. На сайте Филиала УГТУ-УПИ в г. Ирбите <http://www.upi-irbit.by.ru/> в разделе «Студенту» доступна для загрузки документация по VBA.

Учебное текстовое электронное издание

Сысков Владимир Иванович

**ПРОГРАММИРОВАНИЕ БАЗОВЫХ АЛГОРИТМОВ
НА VISUAL BASIC FOR APPLICATIONS (MS EXCEL)**

Редактор	<i>Лугова Н.В.</i>
Компьютерная верстка:	<i>Сысков В.И.</i>

Рекомендовано РИС ГОУ ВПО УГТУ-УПИ
Разрешен к публикации 28.05.08

Электронный формат – PDF
Формат 60x90 1/8 Объем 2 уч.-изд. л.

Издательство ГОУ-ВПО УГТУ-УПИ
620002, Екатеринбург, ул. Мира, 19

Информационный портал
ГОУ ВПО УГТУ-УПИ
<http://www.ustu.ru>