КАЗАНСКИЙ ГОСУДАРСТВЕННЫЙ ЭНЕРГЕТИЧЕСКИЙ УНИВЕРСИТЕТ

Р. Г. ТАХАВУТДИНОВ

ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ

ЧАСТЬ IV:
ЭЛЕМЕНТЫ СТРУКТУРНОГО ПРОГРАММИРОВАНИЯ. ТИПЫ ПЕРЕМЕННЫХ

Методические указания к лабораторным работам, практическим занятиям, расчетному заданию и самостоятельной работе студентов по дисциплинам «Информатика», «Программные и аппаратные средства информатики», «Алгоритмические языки и программирование», «Вычислительная техника и программирование»

Казань 2008

УДК 516.62(077)
ББК 31.31
М37

Тахавутдинов Р.Г.

Основы алгоритмизации и программирования. Часть IV: Элементы структурного программирования. Типы переменных. Методические указания к лабораторным работам, практическим занятиям, расчетному заданию и самостоятельной работе студентов по дисциплинам «Информатика», «Программные и аппаратные средства информатики», «Алгоритмические языки и программирование», «Вычислительная техника и программирование». Казань: Казан. гос. энерг. ун-т, 2008.

Содержатся рекомендации по выполнению компьютерного практикума, который предусмотрен Государственным образовательным стандартом. Предназначены для студентов всех специальностей при изучении дисциплин «Информатика», «Программные и аппаратные средства информатики», «Алгоритмические языки и программирование», «Вычислительная техника и программирование», а также других информационно-ориентированных дисциплин. Могут быть использованы в ходе лабораторных работ, практических занятий, при выполнении расчетного задания и для самостоятельной работы студентов.

ВВЕДЕНИЕ

Современное состояние развития программных средств и вычислительной техники позволяет решать многие практически важные задачи в самых различных областях с использованием готовых программных комплексов. При этом зачастую пользователю нет необходимости самому составлять методику решения, алгоритм и программу на каком-либо искусственном языке, как это было десятилетие назад. Вместе с тем, нередко встречаются ситуации, когда эти программные комплексы не предоставляют стандартных опций для решения каких-либо специфических задач пользователя, или же эти стандартные опции являются неудобными для решения конкретных проблем. Поэтому практически во всех программных комплексах предусмотрена возможность их дополнения собственными модулями пользователя, написанными на каком-либо языке программирования. Таким образом, эти программные комплексы, как правило, имеют свою интегрированную среду программирования или же свой язык макрокоманд, с использованием которых пользователь может ***автоматизировать*** работу с этими программными комплексами применительно к собственным задачам, или же ***расширить возможности*** программных комплексов, вводя туда свои собственные модули. Поэтому наряду с умением применять готовые программные продукты, современные пользователи должны обладать также навыками алгоритмизации и программирования, и понимать основные принципы алгоритмизации, которые являются общими для всех искусственных языков и программных сред.

Основной ***целью*** работ, которые изложены в данных методических указаниях, является содействие студентам в приобретении основных навыков алгоритмизации и программирования. При этом также решаются ***задачи*** овладения основами автоматизации работы с документами и расширения возможностей программных продуктов путем создания дополнительных модулей на алгоритмическом языке программирования.

В данном компьютерном практикуме предусмотрено обучение основам алгоритмизации и программирования в рамках программного комплекса Microsoft Office с использованием интегрированного с ним языка программирования Visual Basic for Applications (сокращенно общепринята аббревиатура VBA). Таким образом, при прохождении компьютерного практикума как бы стирается грань между использованием готового программного комплекса и программированием: пользователи программируют, находясь в рамках готового программного комплекса, и

используют программирование для расширения возможностей этого программного комплекса и автоматизации работы с документами. Данный подход, вне всякого сомнения, является передовым и полностью соответствует современному состоянию развития вычислительной техники и программных средств. Несомненным достоинством принятого подхода является также то, что преподавание компьютерного практикума по дисциплине «Информатика» и другим информационно-ориентированным дисциплинам осуществляется в единой канве, включая как изучение офисных программных средств, так и алгоритмизацию и программирование.

ОСНОВЫ СТРУКТУРНОГО ПРОГРАММИРОВАНИЯ

В современной классификации методов программирования обычно выделяют ***алгоритмическое, структурное и объектно-ориентированное программирование***. При выполнении заданий данного компьютерного практикума мы, возможно, сами того не замечая, фактически использовали все три перечисленных выше метода программирования.

Под ***алгоритмическим программированием*** понимают составление программы в виде конечной последовательности элементарных операций. Примером реализации алгоритмического программирования служит любая программа, составленная Вами при создании отдельного модуля. Она состоит из конечного количества операторов присвоения, условного или безусловного перехода, цикла.

Ранее было обращено Ваше внимание на то, что VBA представляет собой ***объектно-ориентированный язык программирования***, и при выполнении заданий с помощью программы на VBA совершались действия над диапазонами ячеек и листами Excel, которые являются примерами объектов, а также изменялись свойства объектов, например, цвет и тип шрифта.

Структурное программирование предполагает выделение отдельных структур, каждая из которых имеет свое назначение. Примерами этих структур служат процедуры и модули, которые Вы создавали при выполнении заданий данного компьютерного практикума. Преимущества структурного программирования наиболее явно проявляются при реализации сложного алгоритма, когда размещение программы в одном модуле сделало бы ее трудной для восприятия и отладки. Поэтому в таких случаях создают несколько модулей, каждый из которых выполняет понятные и достаточно простые функции. Модуль представляет собой самостоятельную программу, но различные модули могут быть связаны между собой для передачи данных, и из одного модуля может вызываться другой модуль. Для вызова процедуры типа *Sub* из другой процедуры можно использовать команду *Call*, имеющую следующий синтаксис:

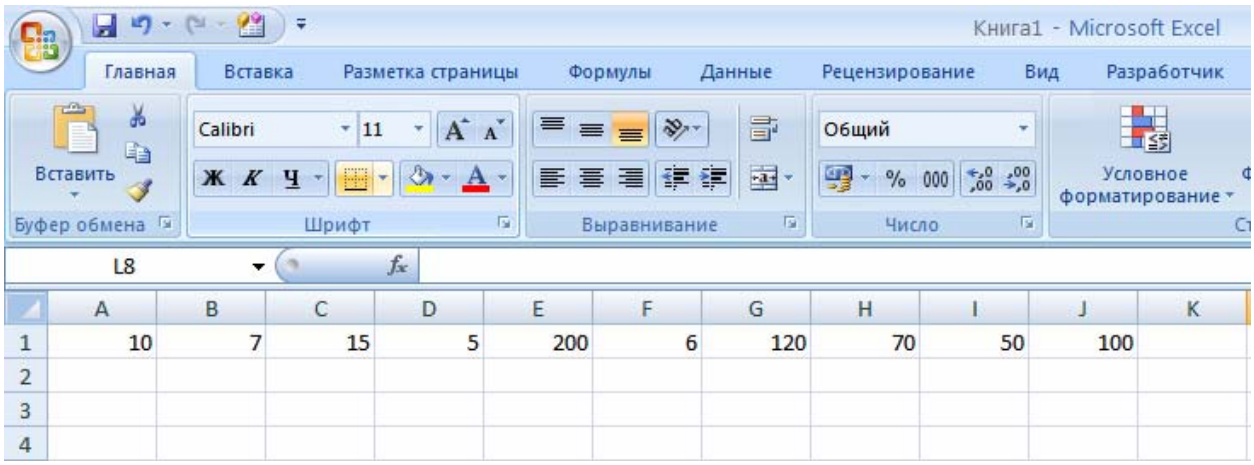
Call name(arglist),

где *name* – имя вызываемой процедуры, *arglist* – список ***фактических*** параметров, которые передаются вызываемой процедуре. Он должен соответствовать по количеству и типу списку ***формальных*** параметров, заданных при определении процедуры.

При выполнении приведенных ниже заданий Вы приобретете первоначальные навыки структурного программирования.

ЗАДАНИЕ 1. Заполните десять ячеек первой строки таблицы Excel произвольными числами. Составьте модуль на VBA, который меняет местами элемент в столбце с порядковым номером *m* и наименьший из элементов, расположенных правее столбца с порядковым номером *m* и заполняет вторую строку преобразованным массивом с сохранением первой строки в неизменном виде. Исполните программу и удостоверьтесь в правильности результата путем сопоставления первой и второй строки таблицы Excel. Данный модуль нам будет необходим при выполнении задания 2, поэтому не уничтожайте его.

РЕШЕНИЕ ЗАДАНИЯ 1. Заполним таблицу Excel:



	A	B	C	D	E	F	G	H	I	J	K
1	10	7	15	5	200	6	120	70	50	100	
2											
3											
4											

Вызовем редактор VBA: ***Разработчик > Visual Basic*** , или же нажатие комбинации клавиш Alt и F11. Добавим стандартный модуль:***Insert>Module***. В появившемся окне напишем следующий текст программы:

```
Sub   minsear()
  n = 10
  m = 4

  For   i = 1   To   n
      Cells(2, i) = Cells(1, i)
  Next

  mina = Cells(2, m)
  imin = m
   For   i = m   To   n
     If   Cells(2, i) < mina   Then
```

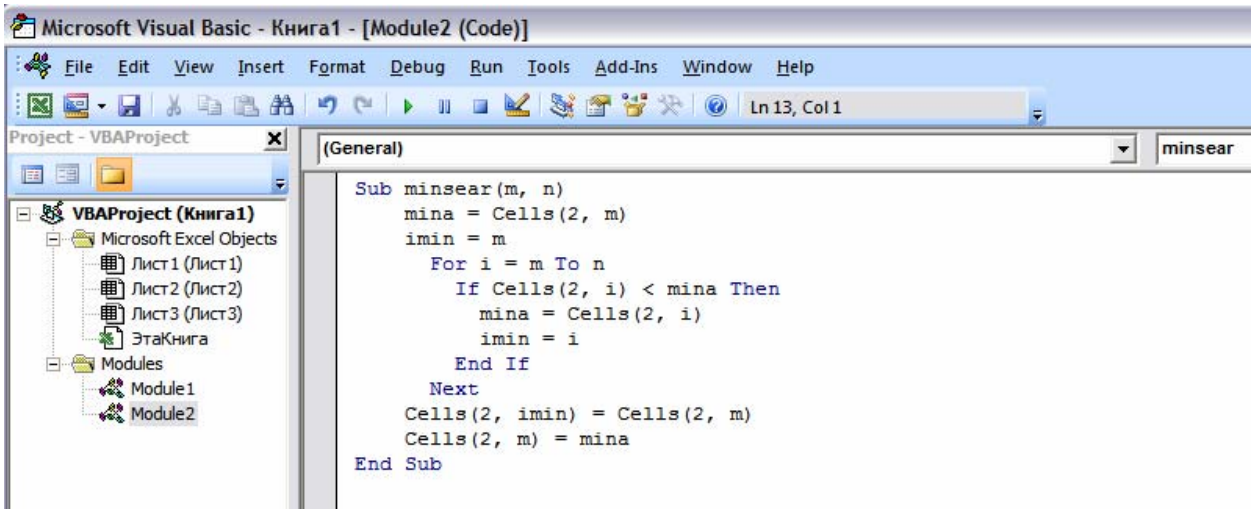
```
        mina = Cells(2, i)
        imin = i
      End   If
    Next
   Cells(2, imin) = Cells(2, m)
   Cells(2, m) = mina
End   Sub
```

Поменяйте несколько раз значение *m* в программе и проверяйте получаемые после каждого исполнения модуля результаты. Данный модуль не следует уничтожать, так как он нам пригодится при выполнении следующего задания.

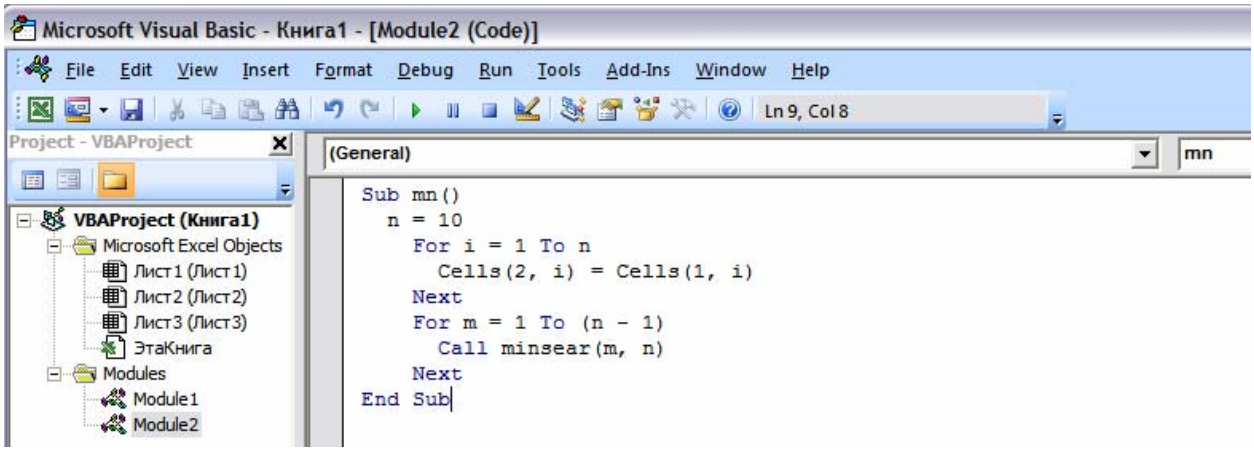
ЗАДАНИЕ 2. Составьте программу, которая упорядочивает элементы предыдущей таблицы в порядке возрастания. *(Примечание: в Excel есть встроенные возможности автоматической сортировки чисел, которые позволяют расположить элементы по возрастанию, но в данном случае не следует пользоваться этим. Данные задания носят учебный характер, поэтому от Вас требуется самим составить программу на VBA. У Вас будет возможность воспользоваться встроенными функциями сортировки при выполнении задания 10)*

РЕШЕНИЕ ЗАДАНИЯ 2. Данную задачу будем решать с использованием наработок, сделанных в ходе выполнения предыдущего задания. Действительно, если запустить предыдущий модуль со значением *m=1*, то наименьший элемент всего массива встанет на первое место. Далее, если запустить его со значением *m=2*, то на второе место встанет наименьший из расположенных правее элементов. Продолжая запускать подобным образом модуль со значениями *m=3*, *m=4*, и так далее до *m=9*, мы получим массив, упорядоченный в порядке возрастания. Описанную процедуру реализуем в виде цикла и разместим в отдельном модуле, из которого будет вызываться модуль, меняющий местами элемент столбца с номером *m* и наименьший из расположенных правее элементов.

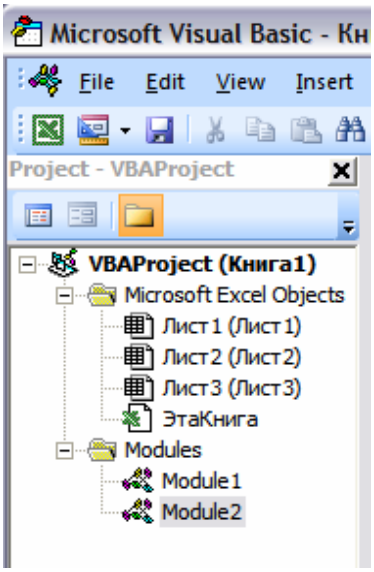
Сначала скорректируем предыдущий модуль, убрав операторы присвоения переменным *m* и *n* конкретных чисел и введя их в список аргументов процедуры в качестве формальных параметров: *Sub minsear(m, n)*. Также уберем цикл, формирующий первоначальную вторую строку таблицы. Тогда это окно программы VBA примет следующий вид:



Не уничтожая данный модуль, создадим новый модуль: ***Insert>Module*** и в появившемся окне наберем следующую программу, которая формирует первоначальную вторую строку, а затем, вызывая процедуру *minsear* в общей сложности (n-1) раз, упорядочивает элементы второй строки в порядке возрастания:



Таким образом, здесь мы в явном виде использовали структурное программирование, распределив весь алгоритм между двумя модулями. Исполните программу, нажав клавиш *F5*, и проверьте, что массив во второй строке упорядочен в порядке возрастания. Вы можете переходить в окно первого или второго модуля двойным щелчком мыши по *Module1* или *Module2* в списке модулей данного проекта:

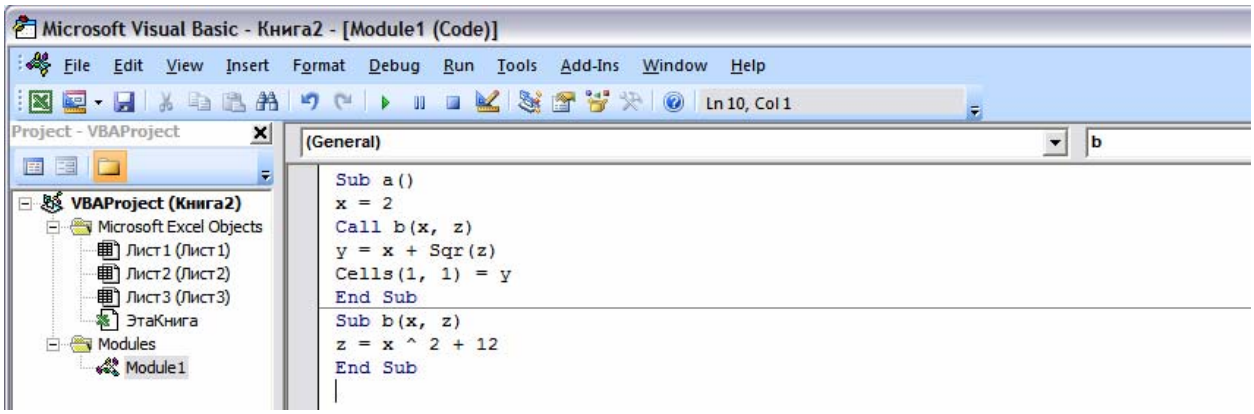


Наверняка данное задание Вам показалось сложным для начала ознакомления со структурным программированием, поэтому Вы могли не понять принцип организации модулей и связей между ними. Следующее задание является очень простым, поэтому в ходе его выполнения Вы несомненно поймете, как можно разбить алгоритм на две независимые процедуры и вызывать одну процедуру из другой.

ЗАДАНИЕ 3. Составьте программу на VBA для вычисления и занесения в ячейку листа Excel значения *y* при *x=2* по формуле:

$y = x + \sqrt{z}$, где *z*, в свою очередь, определяется по формуле $z = x^2 + 12$.

РЕШЕНИЕ ЗАДАНИЯ 3. Одна процедура будет вычислять значение z, а вторая – значение *y* с использованием значения *z*, вычисленного первой процедурой. Для наглядности мы можем расположить тексты программ обеих процедур в одном окне. При этом разделяющая их линия вычерчивается автоматически:



Поясним работу программы. Начинает работу верхняя процедура. Во второй строке переменной x присваивается число 2. Далее управление передается нижней процедуре оператором вызова процедуры Call b(x,z). В качестве

фактических параметров, заключенных в скобки, выступает значение x, равное 2, и переменная z, значение которой вычисляется нижней процедурой и возвращается в верхнюю процедуру после завершения работы нижней процедуры на строке End Sub. Далее работает верхняя процедура с четверной строки, которая вычисляет значение y и далее передает это число в ячейку Cells(1,1) активного листа Excel, расположенную в первом столбце первой строки. Исполните программу и проверьте, что в результате ячейка A1 листа Excel заполнилась числом 6.

Расположение текста двух процедур в одном окне поясняет нам происхождение ключевого слова Sub. Это сокращение английского слова Subroutine, означающего «Подпрограмма». Мы видим, что верхняя процедура a() без списка аргументов использует расположенную ниже процедуру b(x,z) с двумя аргументами в качестве как бы вспомогательной программы. Исторически в подобных случаях программу, не имеющую аргументов, располагали наверху и говорили, что она является «основной программой». Программы, имеющие список аргументов и вызываемые из основной программы, располагали ***под*** основной ***программой*** и называли «подпрограммами». Данная терминология сохранилась до сих пор во многих языках программирования, включая и большинство разновидностей Basic. Но в VBA, как и во многих других современных средах программирования, нет разделения на основную программу и подпрограммы. В VBA все модули являются в некотором смысле равноправными. Мы можем располагать тексты различных процедур как в одном окне, так и разных окнах, соответствующих различным модулям. Именно этим «равноправием» модулей можно объяснить особенность синтаксиса объявления процедуры Sub nane(), заключающуюся в том, что открывающаяся и закрывающаяся скобки выставляются даже в том случае, если у процедуры нет аргументов. Таким образом, различные модули, необходимые для решения конкретной задачи, могут располагаться произвольно и использоваться не только для решения данной задачи, но и других задач.

ЗАДАНИЕ 4. Составьте программу на VBA для вычисления и занесения в ячейку листа Excel значения *y* при *x=5* по формуле:
$y = x^2 + \sqrt{z + x}$, где *z*, в свою очередь, определяется по формуле $z = x^{1/3} + 1$.
Программу организуйте в виде двух процедур типа Sub, одна из которых вычисляет значение z, а другая – значение *y* с использованием вычисленного значения *z*.

ЗАДАНИЕ 5. Выполните предыдущее задание с использованием процедуры типа Function для вычисления z.

РЕШЕНИЕ ЗАДАНИЯ 5. Мы знакомы с вызовом процедур типа Function из листа Excel, когда числа в ячейках вычислялись с использованием созданных нами пользовательских функций. Эти функции можно вызывать также и из других процедур. В нашем случае программа может выглядеть следующим образом:

```
Sub  a()
  x = 5
  y = x ^ 2 + Sqr(z(x) + x)
  Cells(1, 1) = y
End  Sub
```

```
Function  z(x)
  z = x ^ (1 / 3) + 1
End  Function
```

Исполните программу и удостоверьтесь в том, что получается тот же результат вычисления *y*, что и в предыдущем задании. Данный модуль не следует уничтожать или изменять, так как он нам пригодится при выполнении следующего задания.

ЗАДАНИЕ 6. Составьте программу на VBA для вычисления и занесения в ячейку листа Excel значения *y* при *x=2* по формуле:

$$y = \frac{p^{1/4} + \ln|s + x| + q + x^3 + \sqrt{z^2 + x}}{s + 1},$$

где *z*, p, q, s, в свою очередь, определяются по формулам

$$z = x^{1/3} + 1$$

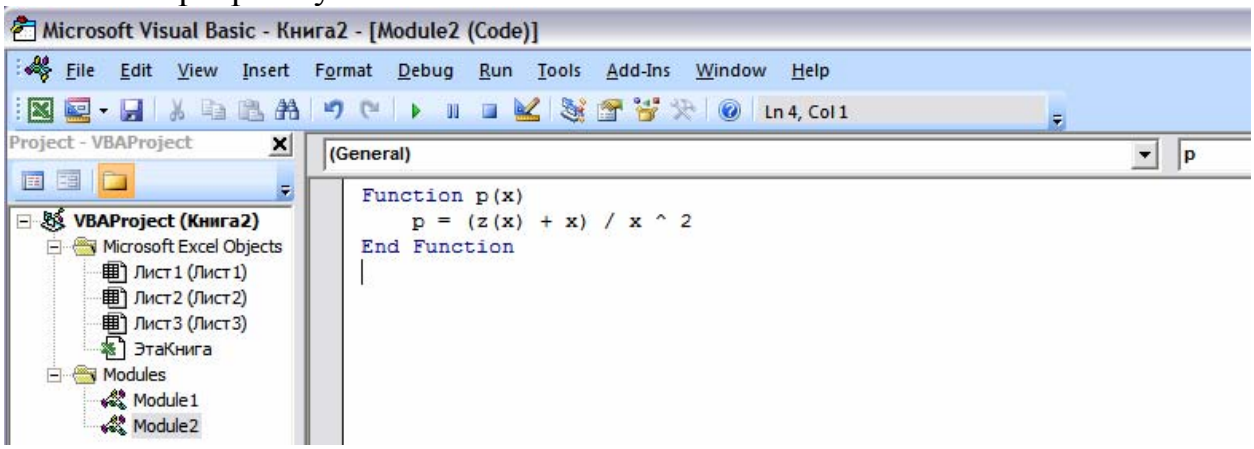
$$p = \frac{z + x}{x^2}$$

$$q = p + z^2 + x$$

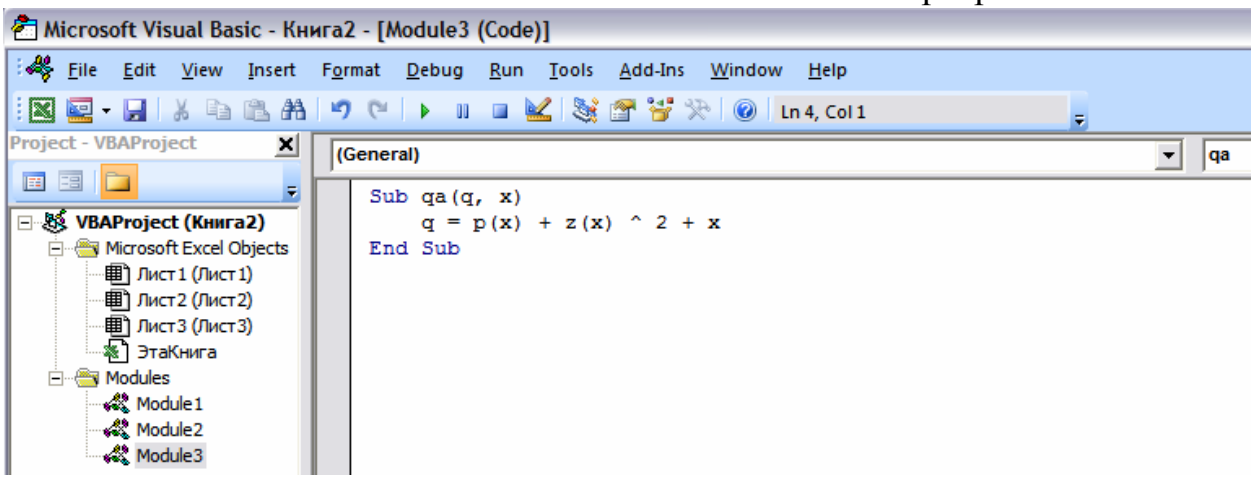
$$s = \ln(x) + p^2$$

Для вычисления *z* и *p* используйте процедуры типа Function, а для *s, q, y* – процедуры типа Sub.

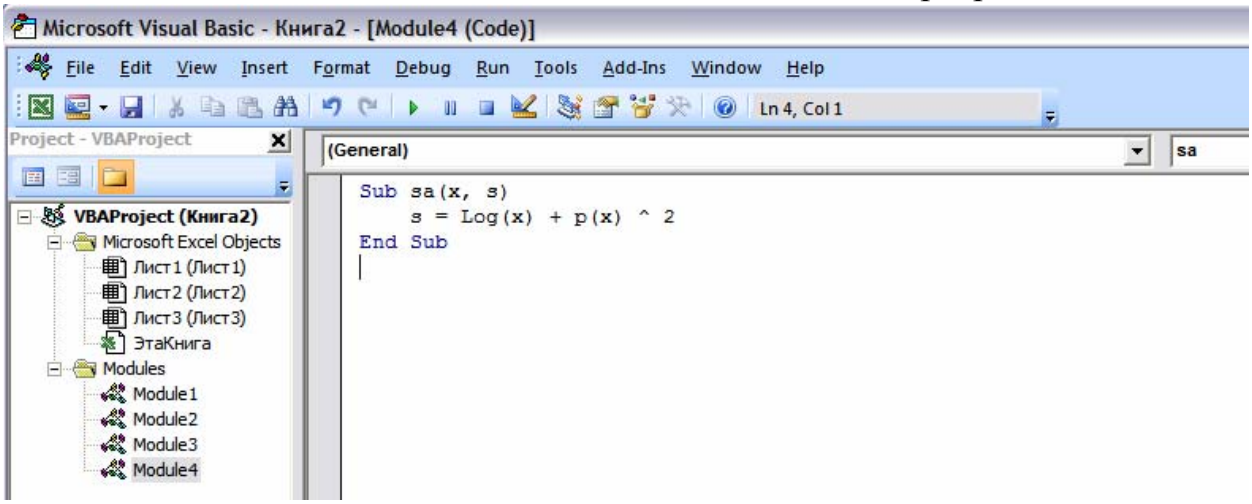
РЕШЕНИЕ ЗАДАНИЯ 6. Для вычисления *z* используем готовую процедуру, составленную при выполнении предыдущего задания, поэтому новый модуль для этого создавать нет необходимости. Процедуры для *p, s, q, y* разместим каждую в отдельном модуле: *Вставка>модуль.* В появившемся окне напишем программу:



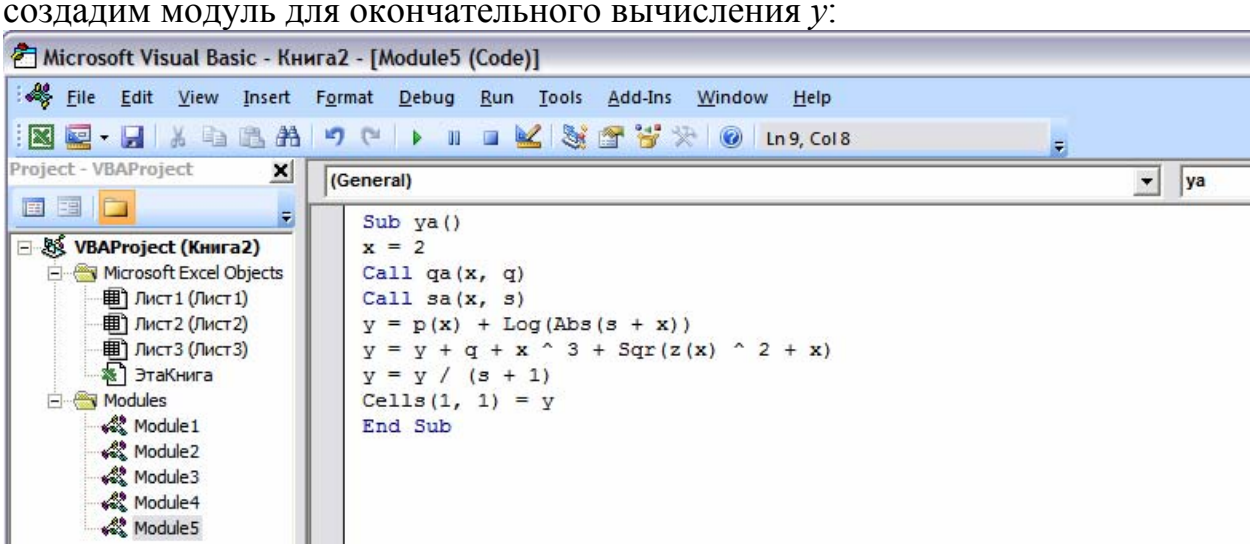
Не уничтожая и не изменяя данный модуль, создадим новый: ***Insert>Module***. В появившемся окне напишем текст программы:



Не уничтожая и не изменяя предыдущие модули, создадим еще один модуль: ***Insert>Module***. В появившемся окне напишем текст программы:



Наконец, опять таки не уничтожая и не изменяя предыдущие модули, создадим модуль для окончательного вычисления *y*:



ЗАДАНИЕ 7. Составьте программу на VBA для вычисления и занесения в ячейку листа Excel значения *y* при *x=85* по формуле:

$$y = \frac{s^2 + z + x + \ln(q) + x^2 + f}{p + s + h},$$

где *z, p, q, s* определяются по формулам предыдущего задания, а *f* и *h* по следующим формулам:

$f=x+q^{1/3}+z$

$h=x+p^2+s+q$

Модуль для вычисления *f* оформите в виде процедуры типа Function, а модуль для вычисления *h* – в виде процедуры типа Sub. Для *z, p, q, s* используйте готовые модули, созданные при выполнении двух предыдущих заданий. *(Вычисленное значение y должно получиться равным 35,89905.)*

ЗАДАНИЕ 8. Составьте программу на VBA, которая считывает значение *x* из ячейки Excel, затем вычисляет значение *y* по формуле, приведенной ниже, и передает это значение *y* в ячейку Excel:

$$y = \frac{z + p^2 - p^{1|/3} + 5}{z + x^2},$$

где $z = \frac{x+5}{x+10}$, $p=z+x^2$

Программы для вычисления *z* и *p* оформите в виде отдельных модулей.

ЗАДАНИЕ 9. Заполните десять ячеек первой строки таблицы Excel произвольными числами. Аналогично заданию 2, составьте программу, которая упорядочивает элементы столбца по убыванию. Программу оформите в виде двух модулей, первый их которых меняет местами произвольный элемент с наибольшим из элементов, расположенных правее. Второй модуль, работая по циклу, многократно вызывает первый модуль и меняет местами элементы строки таким образом, чтобы в конце образовалась убывающая последовательность чисел. ***Наряду с созданием новых модулей, при выполнении данного и последующего задания Вы можете использовать модули, созданные ранее, например, в ходе выполнения задания 2.***

ЗАДАНИЕ 10. Заполните произвольными числами таблицу Excel, состоящую из 10 строк и 5 столбцов. Составьте и исполните программу, которая расставляет элементы нечетных строк в порядке возрастания, а элементы четных строк – в порядке убывания.

ЗАДАНИЕ 11. Заполните произвольную таблицу, которая включает как столбцы, содержащие текст (например, фамилии), так и столбцы с числами. Используя встроенные возможности автоматической сортировки

Данные>Сортировка, расположите элементы таблицы по различным признакам (в алфавитном порядке, по возрастанию, убыванию, по нескольким критериям).

ТИПЫ ДАННЫХ И ОБЪЯВЛЕНИЕ ПЕРЕМЕННЫХ

Тип данных относится к самым фундаментальным понятиям любого языка программирования. Тип данных определяет множество допустимых значений, которые может принимать переменная. При ***объявлении переменных*** указывается, какой тип данных хранится в ней. Учитывая основополагающее значение этих понятий, обычно при прохождении компьютерных практикумов они изучаются в самом начале. Однако в рамках данного компьютерного практикума методологически целесообразнее рассмотреть эти вопросы в завершение. Дело в том, что на начальном этапе в принципе можно вполне обходиться и без объявления переменных и знания типа данных. Действительно, мы с Вами успешно выполнили все предыдущие задания по программированию без объявления переменных. Поэтому все используемые нами переменные изначально имели тип, принятый по умолчанию (*variant*), а их настоящий тип автоматически распознавался транслятором и преобразовывался к требуемому типу. Однако объявление переменных делает программу более надежной, ускоряет ее работу, позволяет проводить вычисления с большей точностью и облегчает процесс отладки программы. Поэтому программисты, как правило, объявляют все используемые переменные. Перечислим некоторые типы данных (в VBA их существенно больше), которые мы будем использовать при выполнении последующих заданий:

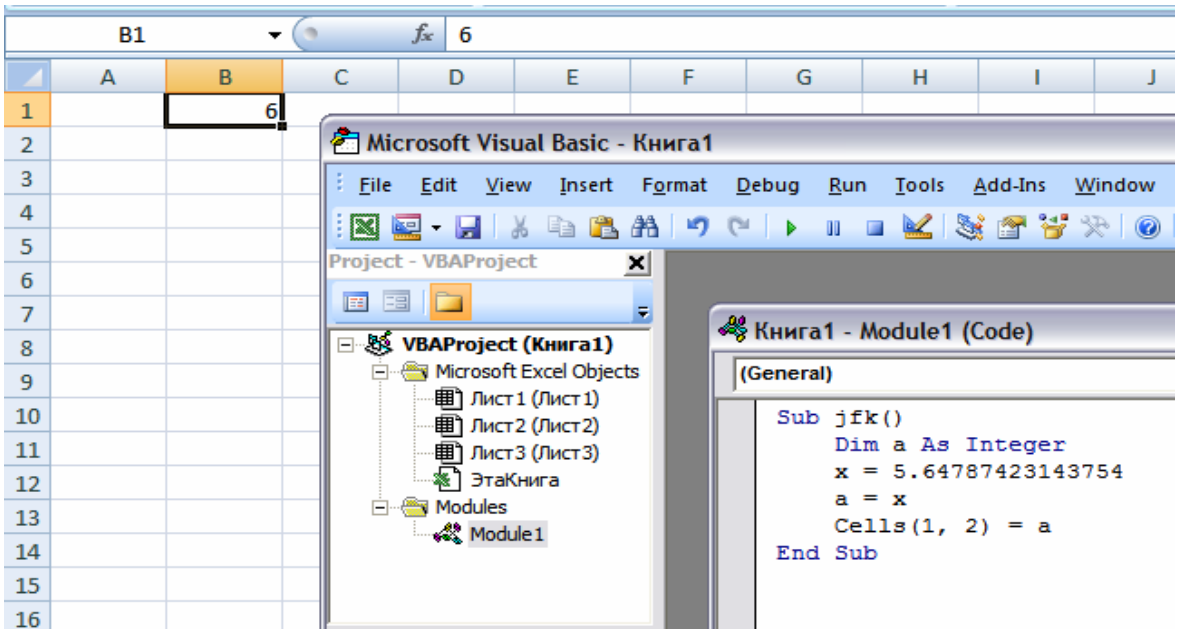
Тип	Описание	Объявление
Integer	Целочисленное	*Dim Name as Integer*
Single	Обычной точности (до шестого знака)	*Dim Name as Single*
Double	Двойной точности (до четырнадцатого знака)	*Dim Name as Double*
String	Строка	*Dim Name as String*

Переменные типов *Integer, Single, Double* являются числовыми, а типа *String* - символьными. Если Вы объявили, что некая переменная является строковой, то есть имеет тип *String*, то Вы с этой переменной никаких арифметических операций производить не сможете, если даже присвоили этой переменной последовательность символов, похожую на число. Дело в том, что эта последовательность цифр транслятором будет восприниматься не как число, -а как ***набор символов***. В VBA есть единственная операция для работы с

данными типа строка, называемая *конкатенцией*, которая объединяет несколько строк в одну. Эти и некоторые другие особенности объявления переменных и использования различных типов данных Вам станут более понятными в ходе выполнения следующих заданий.

ЗАДАНИЕ 12. В окне стандартного модуля VBA введите программу, в которой присвойте переменной *x* число 5.64787423143754 объявите целочисленную переменную *a* и присвойте переменной *a* значение переменной *x*. Передайте значение переменной в ячейку Cells(1,2) листа Excel и посмотрите результат.

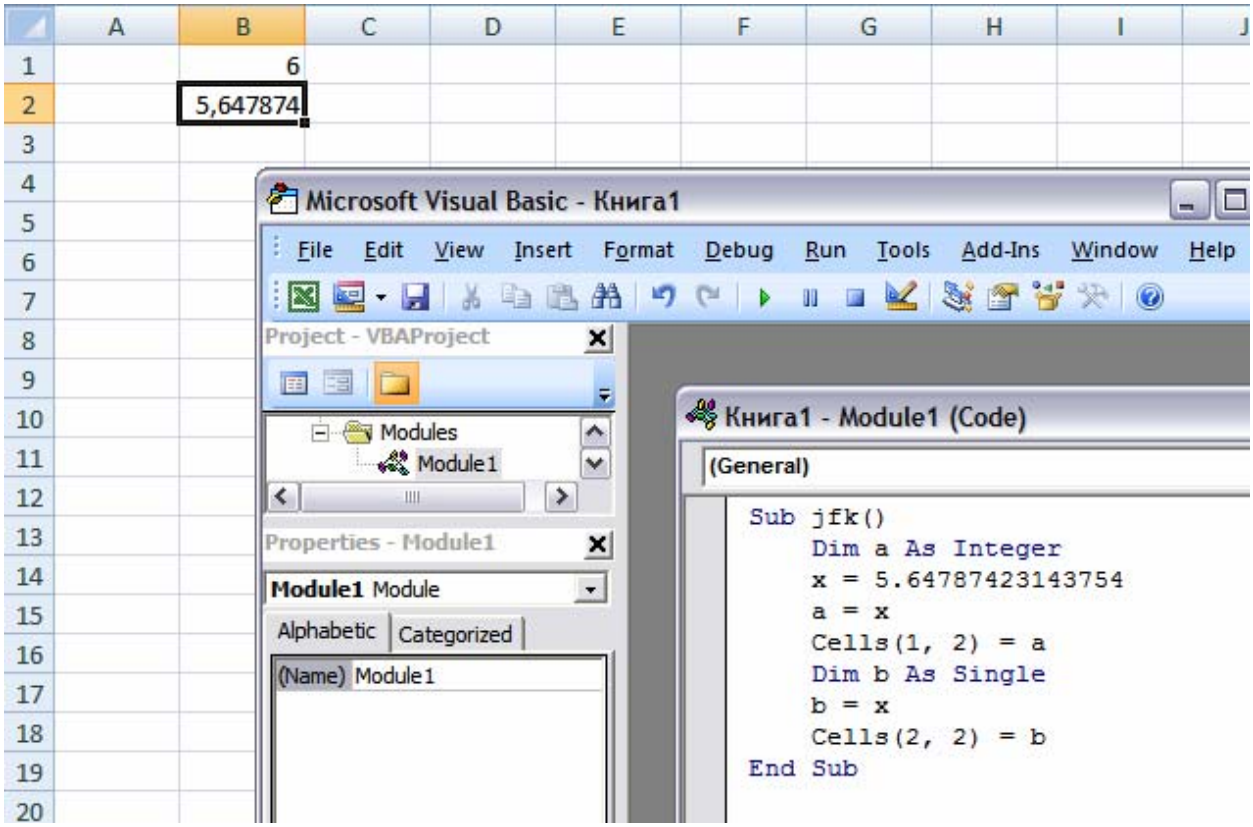
РЕШЕНИЕ ЗАДАНИЯ 12. Программа и результат ее исполнения:



Мы видим, что поскольку тип переменной *a* был объявлен как целочисленный, то в результате исполнения программы в ячейку Excel передалось целое число 6, которое представляет собой округление числа 5.64787423143754 до ближайшего целого.

ЗАДАНИЕ 13. Не стирая имеющиеся строки, дополните предыдущую программу объявлением переменной *b* обычной точности, присвойте ей значение переменной *x* и передайте в ячейку Cells(2,2). Исполните программу и сопоставьте значение переменной *x* и переданное в ячейку Excel значение переменной *b*.

РЕШЕНИЕ ЗАДАНИЯ 13:

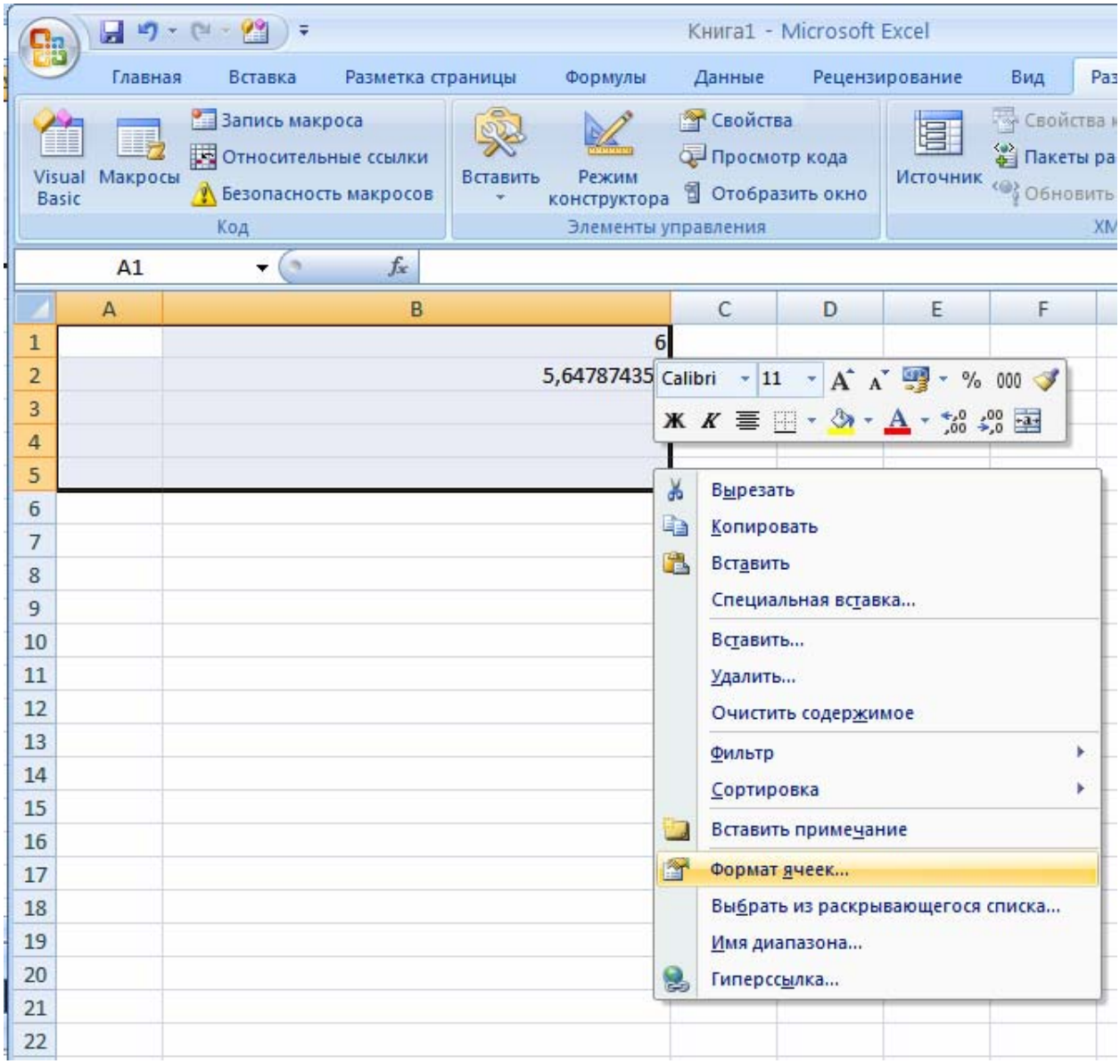


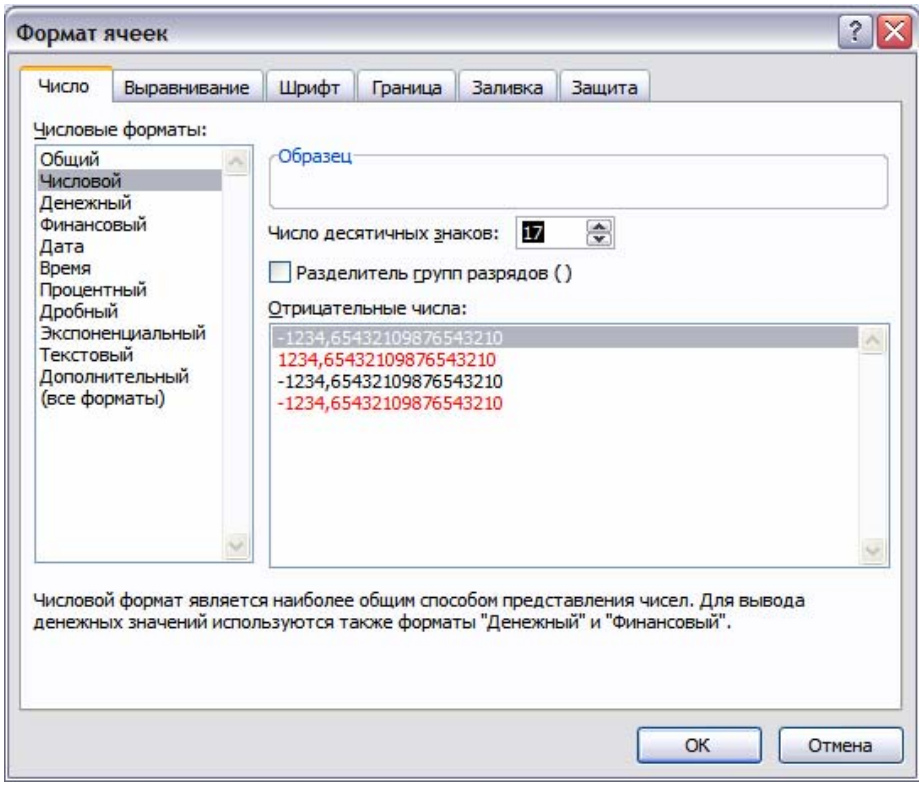
Сопоставляя значение переменной *x* и переданное в ячейку Excel значение переменной *b* мы видим, что они совпадают только ***до шестого знака*** после разделителя целой и дробной частей.

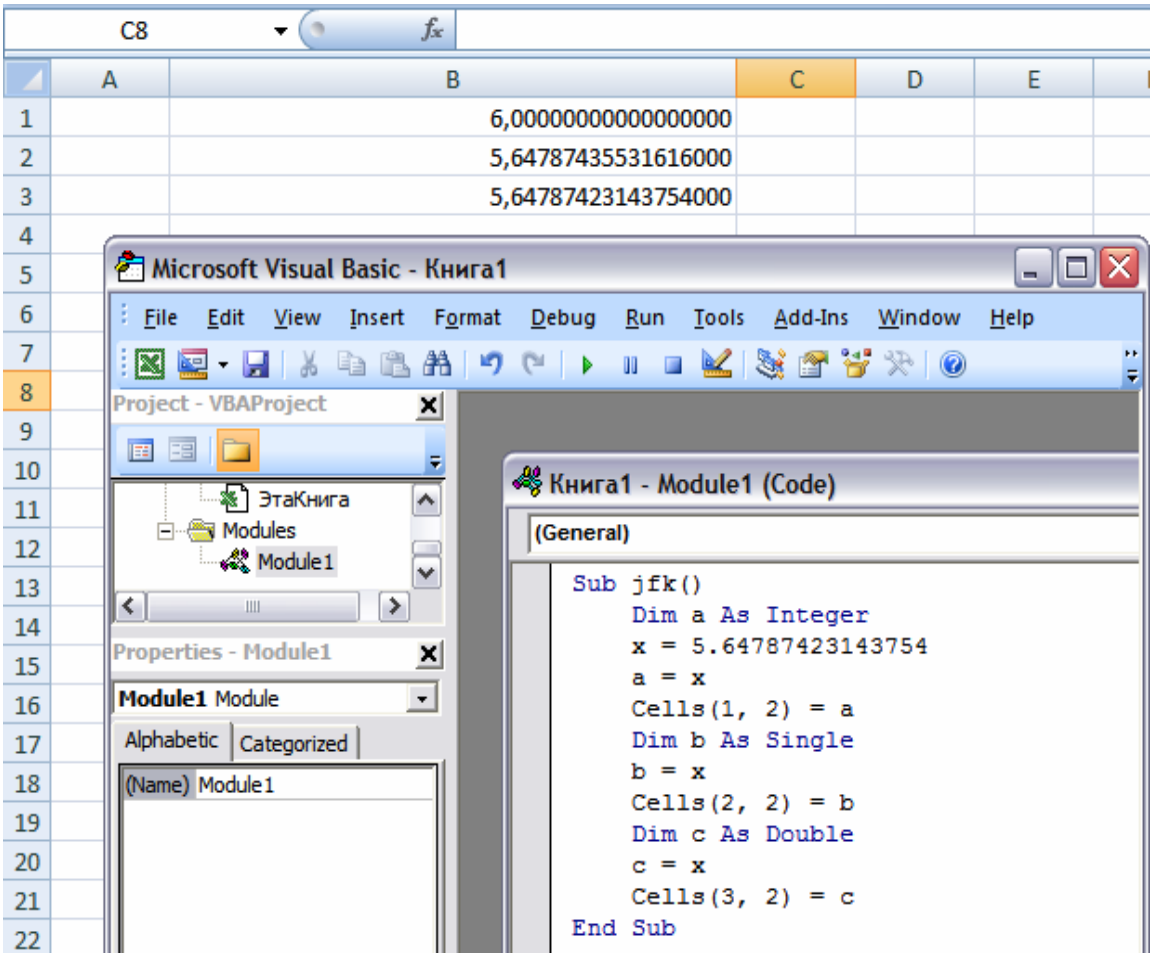
ЗАДАНИЕ 14. Дополните предыдущую программу объявлением переменной *c* двойной точности, присвойте ей значение переменной *x* и передайте в ячейку Cells(3,2). Исполните программу и сопоставьте значение переменной *x* и переданное в ячейку Excel значение переменной *c*.

РЕШЕНИЕ ЗАДАНИЯ 14. Для того, чтобы отображались все значащие цифры, нам необходимо расширить протяженность ячеек столбца *B* по горизонтали, затем выделить необходимые ячейки и выставить формат отображения чисел, для определенности, с 17 знаками после разделителя целой и дробной частей (это вовсе не означает, что вычисления будут проводиться с точностью до 17 знака, речь идет о форме отображения числа.). Потянув мышью за вертикальную линию между *B* и *C*, растяните столбец *B*. Затем выделите ячейки столбца *B*, нажмите правую кнопку мыши и в появившемся окне выберите *формат ячеек* (см. следующий рисунок). Затем в окне формата ячеек выберите *числовой,* далее *число десятичных знаков:17*, как показано на рисунке ниже. Еще ниже на рисунке представлена

программа и результат ее исполнения. Сопоставление значения переменной *x* и переданного в ячейку Excel значения переменной *c* показывает, что они совпадают ***до четырнадцатого знака*** после разделителя целой и дробной частей. Три дополнительных нуля не являются значащими цифрами, так как они возникли из-за того, что мы потребовали отображать 17 десятичных знаков.

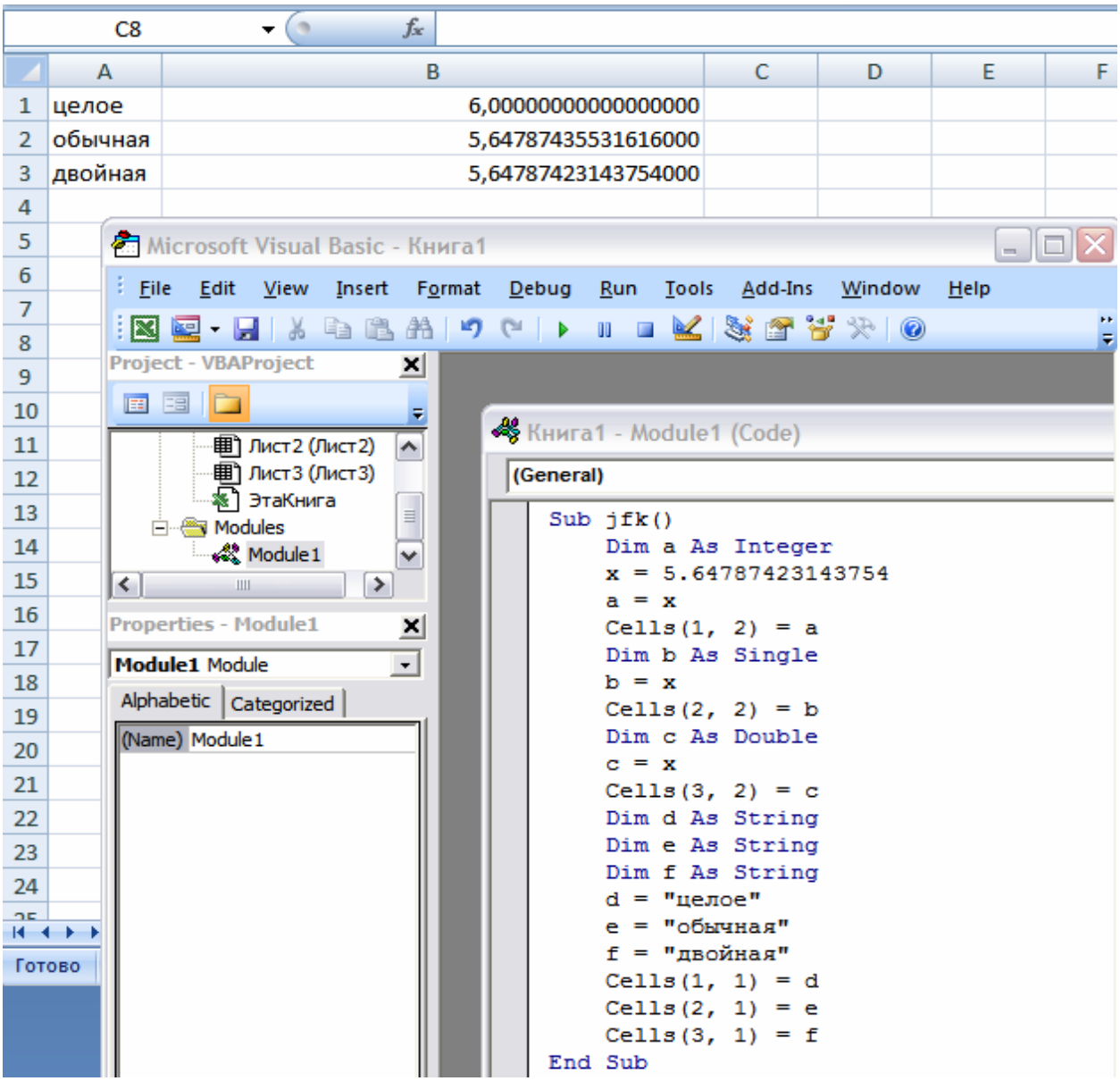



Формат ячеек
Число
Выравнивание
Шрифт
Граница
Заливка
Защита
Числовые форматы:
Общий
Числовой
Денежный
Финансовый
Дата
Время
Процентный
Дробный
Экспоненциальный
Текстовый
Дополнительный
(все форматы)
Образец
Число десятичных знаков: 17
Разделитель групп разрядов ( )
Отрицательные числа:
-1234,65432109876543210
1234,65432109876543210
-1234,65432109876543210
-1234,65432109876543210
Числовой формат является наиболее общим способом представления чисел. Для вывода денежных значений используются также форматы "Денежный" и "Финансовый".
OK
Отмена


C8
A
B
C
D
E
1
6,00000000000000000
2
5,64787435531616000
3
5,64787423143754000
Microsoft Visual Basic - Книга1
File Edit View Insert Format Debug Run Tools Add-Ins Window Help
Project - VBAProject
ЭтаКнига
Modules
Module1
Properties - Module1
Module1 Module
Alphabetic
Categorized
(Name) Module1
Книга1 - Module1 (Code)
(General)
Sub jfk()
Dim a As Integer
x = 5.64787423143754
a = x
Cells(1, 2) = a
Dim b As Single
b = x
Cells(2, 2) = b
Dim c As Double
c = x
Cells(3, 2) = c
End Sub

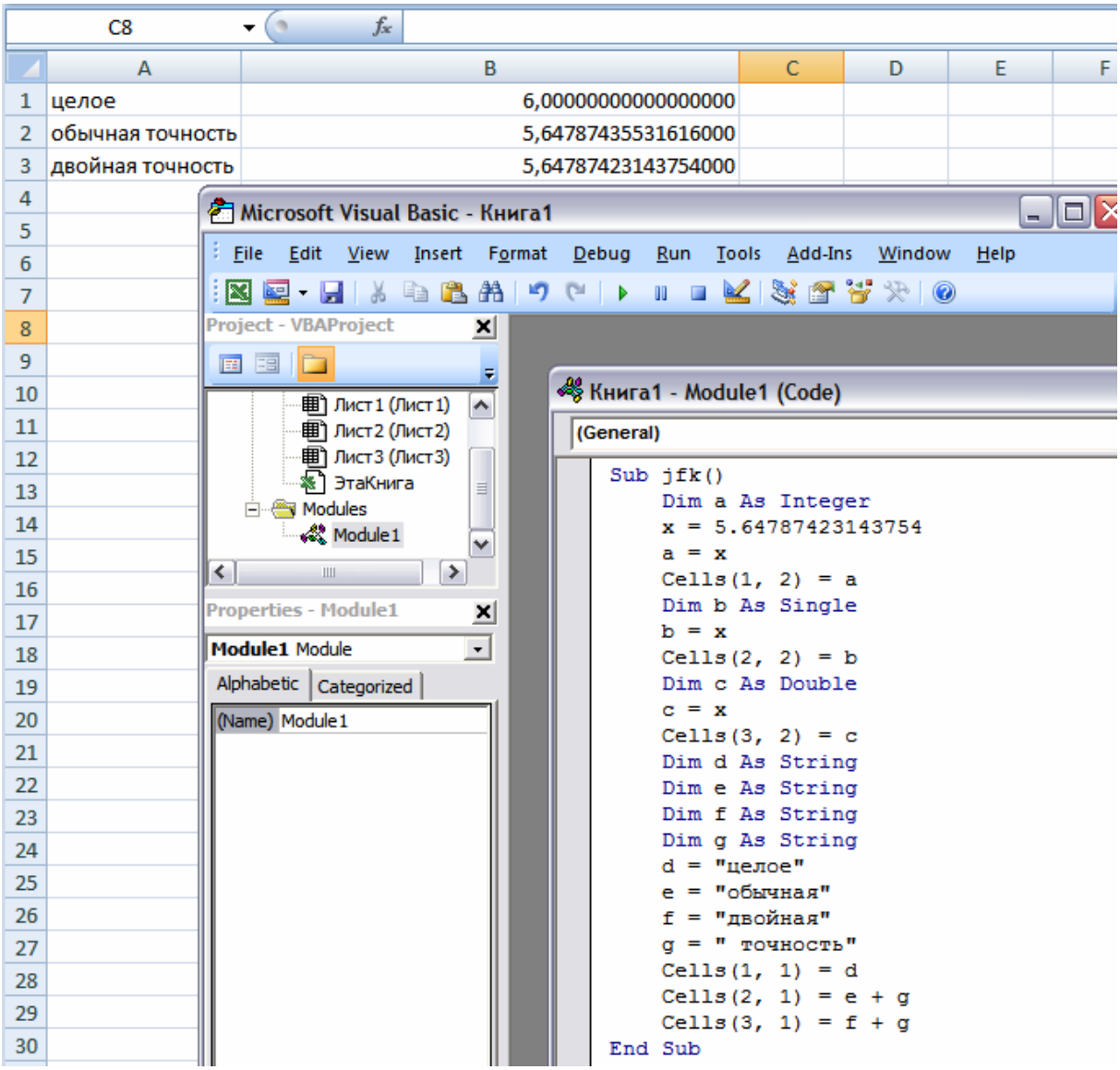
ЗАДАНИЕ 15. Дополните предыдущую программу объявлением переменных *d, e, f* типа строка, присвойте им соответственно тексты *целое, обычная* и *двойная* и передайте строковые значения этих переменных соответственно в ячейки Cells(1,1), Cells(1,2) и Cells(1,3) в качестве пояснений к числам второго столбца таблицы.

РЕШЕНИЕ ЗАДАНИЯ 15: Программа и результат исполнения:



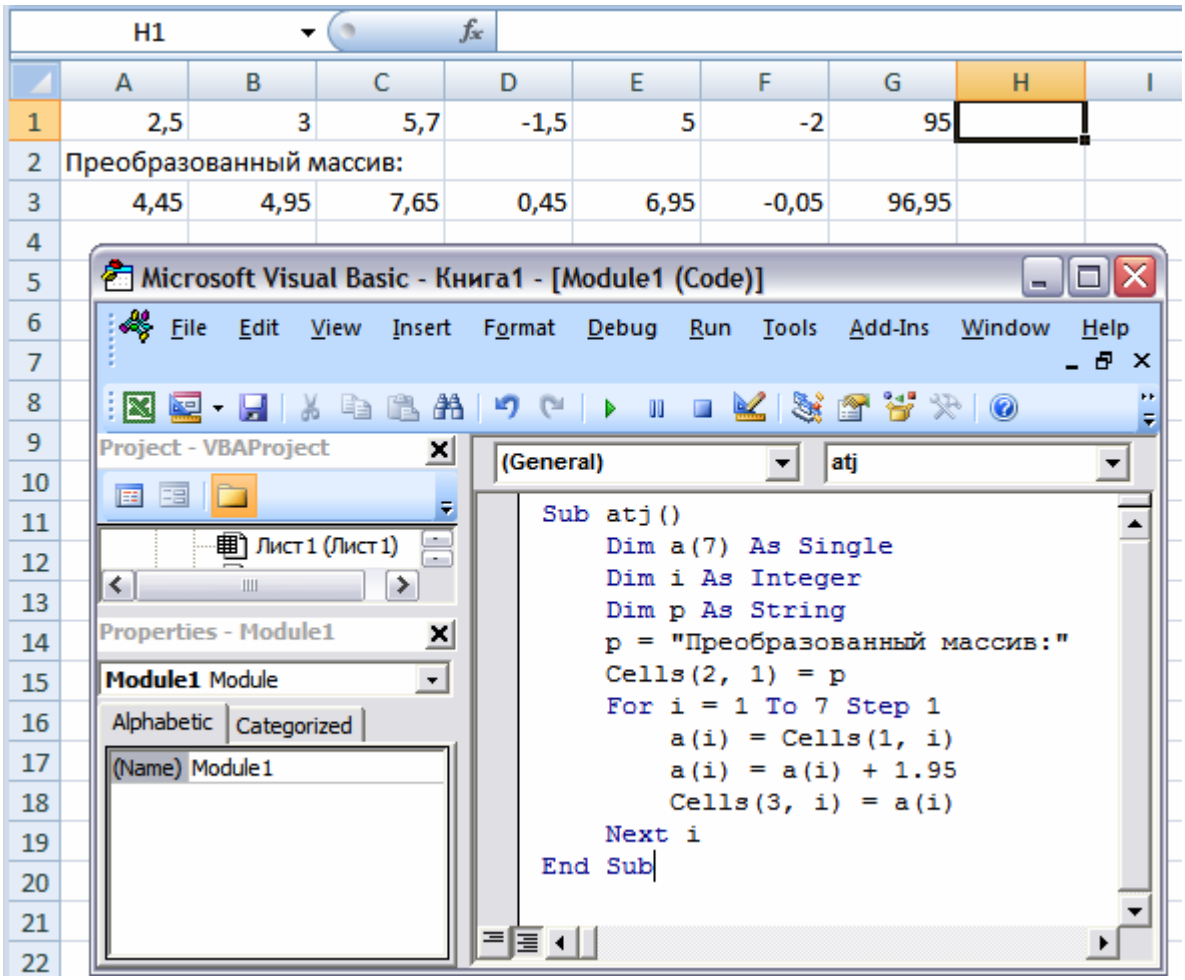
ЗАДАНИЕ 16. Дополните предыдущую программу объявлением переменной *g* типа строка, присвойте ей текст *точность* и используя операцию конкатенции строк, обозначаемую простым символом +, сделайте так, чтобы в результате исполнения программы в первом столбце листа Excel пояснения имели вид *обычная точность* и *двойная точность*.

РЕШЕНИЕ ЗАДАНИЯ 16. При необходимости растяните первый столбец листа Excel с помощью мыши. Программа и результат ее исполнения представлены на рисунке:



Операцию конкатенции строк вместо символа + можно также обозначать символом амперсанда &. Отредактируйте программу, заменив в ней + на & и удостоверьтесь в том, что после ее исполнения получается прежний результат.

ЗАДАНИЕ 17. Составьте программу на VBA, которая объявляет одномерный массив *a* обычной точности, состоящий из 7 элементов, считывает его значения из таблицы Excel, преобразовывает каждый элемент по формуле *a(i)=a(i)+1.95* и передает преобразованный массив в свободные ячейки листа Excel, располагая его после пояснения “Преобразованный массив”. Исполните программу, а после этого преобразуйте формат



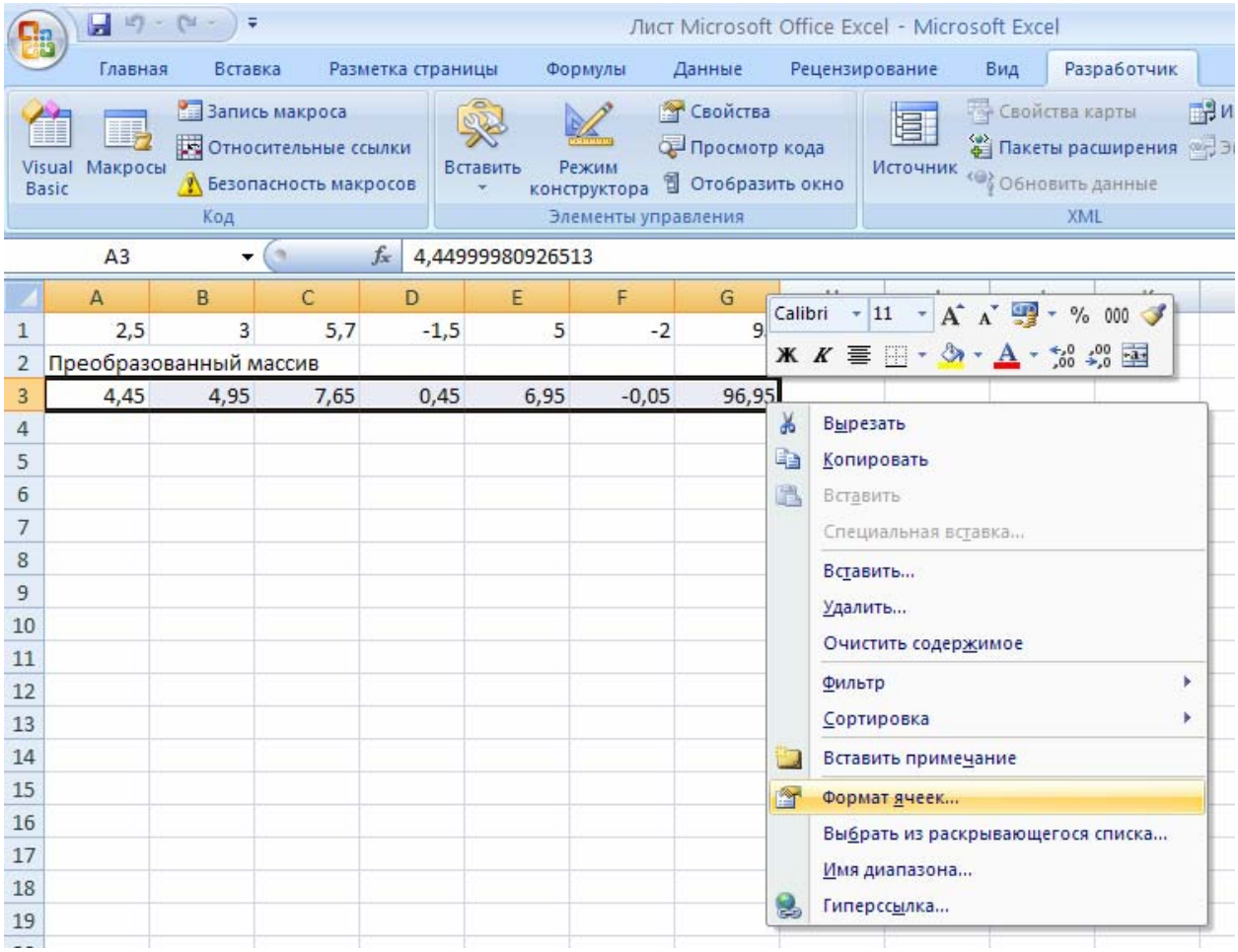
отображения чисел к экспоненциальному виду. *(В программе необходимо объявить все используемые переменные!)*

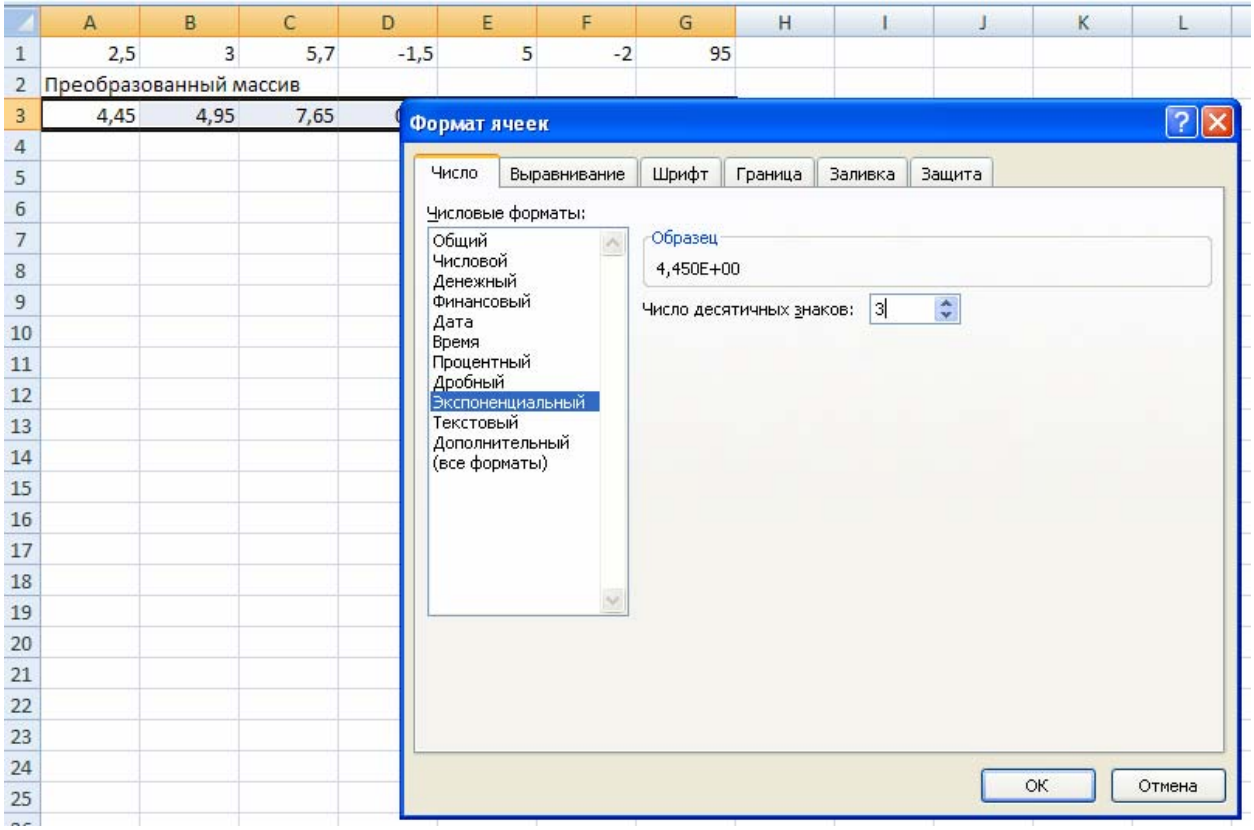
РЕШЕНИЕ ЗАДАНИЯ 17: Ниже на рисунке представлена программа и результат ее исполнения.

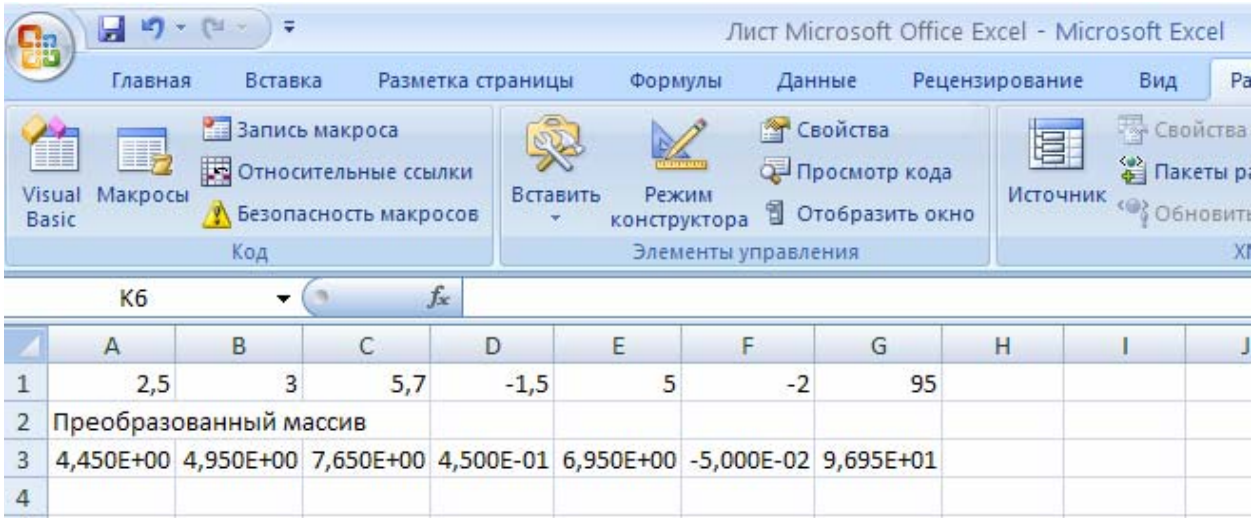
Принцип представления чисел в экспоненциальном формате станет понятным из следующих примеров: 1000=1.0E+03; $1.5\cdot10^{5}$=1.5E+05; $7.8\cdot10^{-15}$=7.8E-15; $5.7\cdot10^{-3}$=5.7E-03; $-2\cdot10^{5}$=-2E+05

Для преобразования формата чисел к экспоненциальному виду выделите требуемые ячейки, как показано на рисунке ниже, затем нажмите правую кнопку. Выберите *формат ячеек* и в появившемся окне, представленном на следующем рисунке, установите *числовой формат: экспоненциальный*, а число десятичных знаков поставьте равным 3. После нажатия *OK*, лист Excel примет вид, приведенный на следующем рисунке.

Экспоненциальный формат удобен для представления больших, или же наоборот, маленьких чисел. Например, число 0.00000000000000000159 в экспоненциальном формате представляется компактно 1.59E-19. Однако не следует путать тип данных с форматом чисел, так как первое определяет множество допустимых значений переменной, а второе является лишь формой представления числа.


Лист Microsoft Office Excel - Microsoft Excel
Главная
Вставка
Разметка страницы
Формулы
Данные
Рецензирование
Вид
Разработчик
Visual Basic
Макросы
Запись макроса
Относительные ссылки
Безопасность макросов
Код
Вставить
Режим конструктора
Свойства
Просмотр кода
Отобразить окно
Элементы управления
Источник
Свойства карты
Пакеты расширения
Обновить данные
XML
A3
4,44999980926513
2,5 3 5,7 -1,5 5 -2
Преобразованный массив
4,45 4,95 7,65 0,45 6,95 -0,05 96,95
Calibri
11
Вырезать
Копировать
Вставить
Специальная вставка...
Вставить...
Удалить...
Очистить содержимое
Фильтр
Сортировка
Вставить примечание
Формат ячеек...
Выбрать из раскрывающегося списка...
Имя диапазона...
Гиперссылка...

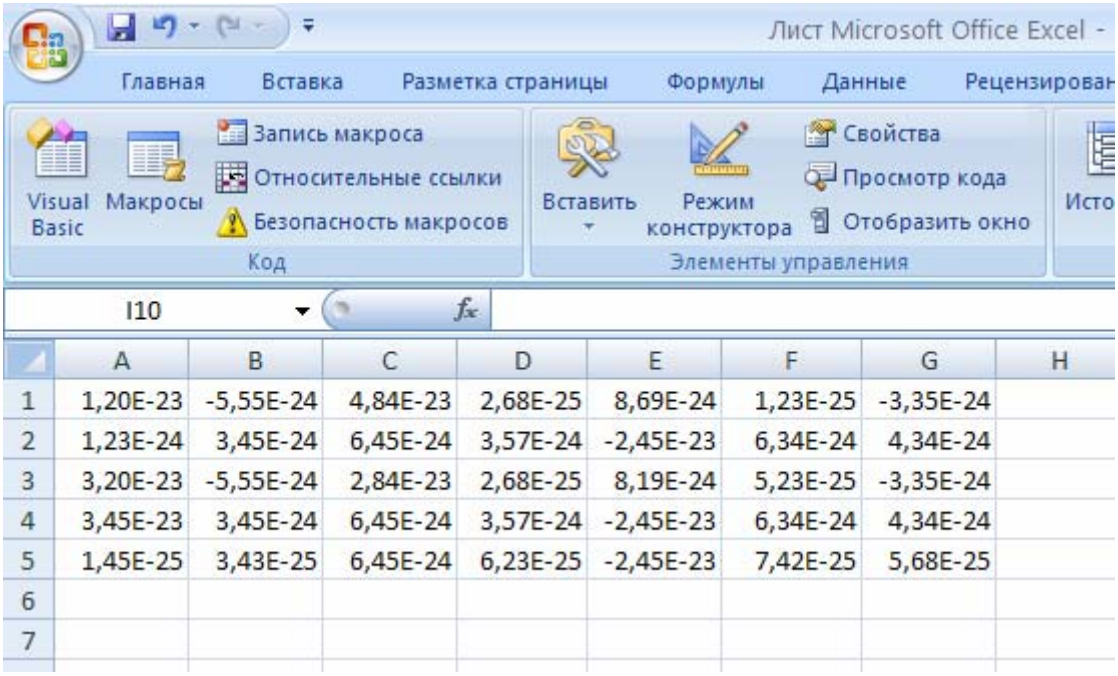

2,5 3 5,7 -1,5 5 -2 95
Преобразованный массив
4,45 4,95 7,65
Формат ячеек
Число
Выравнивание
Шрифт
Граница
Заливка
Защита
Числовые форматы:
Общий
Числовой
Денежный
Финансовый
Дата
Время
Процентный
Дробный
Экспоненциальный
Текстовый
Дополнительный
(все форматы)
Образец
4,450E+00
Число десятичных знаков: 3
OK
Отмена



	A	B	C	D	E	F	G	H	I	J
1	2,5	3	5,7	-1,5	5	-2	95			
2	Преобразованный массив									
3	4,450E+00	4,950E+00	7,650E+00	4,500E-01	6,950E+00	-5,000E-02	9,695E+01			
4										

ЗАДАНИЕ 18. Заполните произвольными числами, имеющими порядок от 10^{-25} до 10^{-23}, таблицу Excel в экспоненциальном формате, состоящую из 5 строк и 7 столбцов. Составьте программу на VBA, которая объявляет двумерный массив *a* размерности 5x7 обычной точности, считывает его значения из таблицы Excel, преобразует каждый элемент по формуле $a(i,j)=a(i,j)+1.5\cdot10^{-24}$ и передает преобразованный массив в свободные ячейки листа Excel, располагая его после пояснения “Преобразованный массив”. *В программе необходимо объявить все используемые переменные!* Исполните программу и проверьте результат.

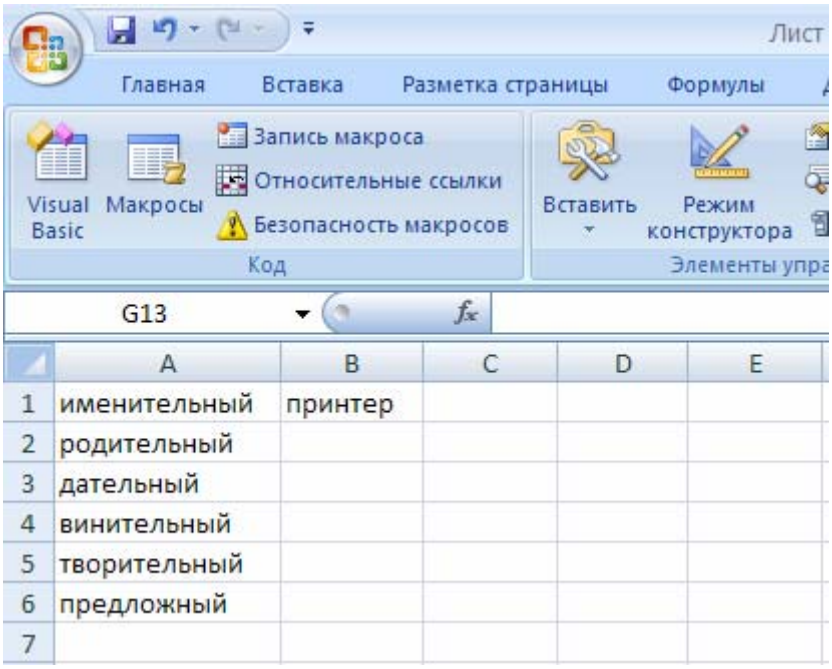
Исходная таблица может состоять, например, из следующих чисел:



	A	B	C	D	E	F	G	H
1	1,20E-23	-5,55E-24	4,84E-23	2,68E-25	8,69E-24	1,23E-25	-3,35E-24	
2	1,23E-24	3,45E-24	6,45E-24	3,57E-24	-2,45E-23	6,34E-24	4,34E-24	
3	3,20E-23	-5,55E-24	2,84E-23	2,68E-25	8,19E-24	5,23E-25	-3,35E-24	
4	3,45E-23	3,45E-24	6,45E-24	3,57E-24	-2,45E-23	6,34E-24	4,34E-24	
5	1,45E-25	3,43E-25	6,45E-24	6,23E-25	-2,45E-23	7,42E-25	5,68E-25	
6								
7								

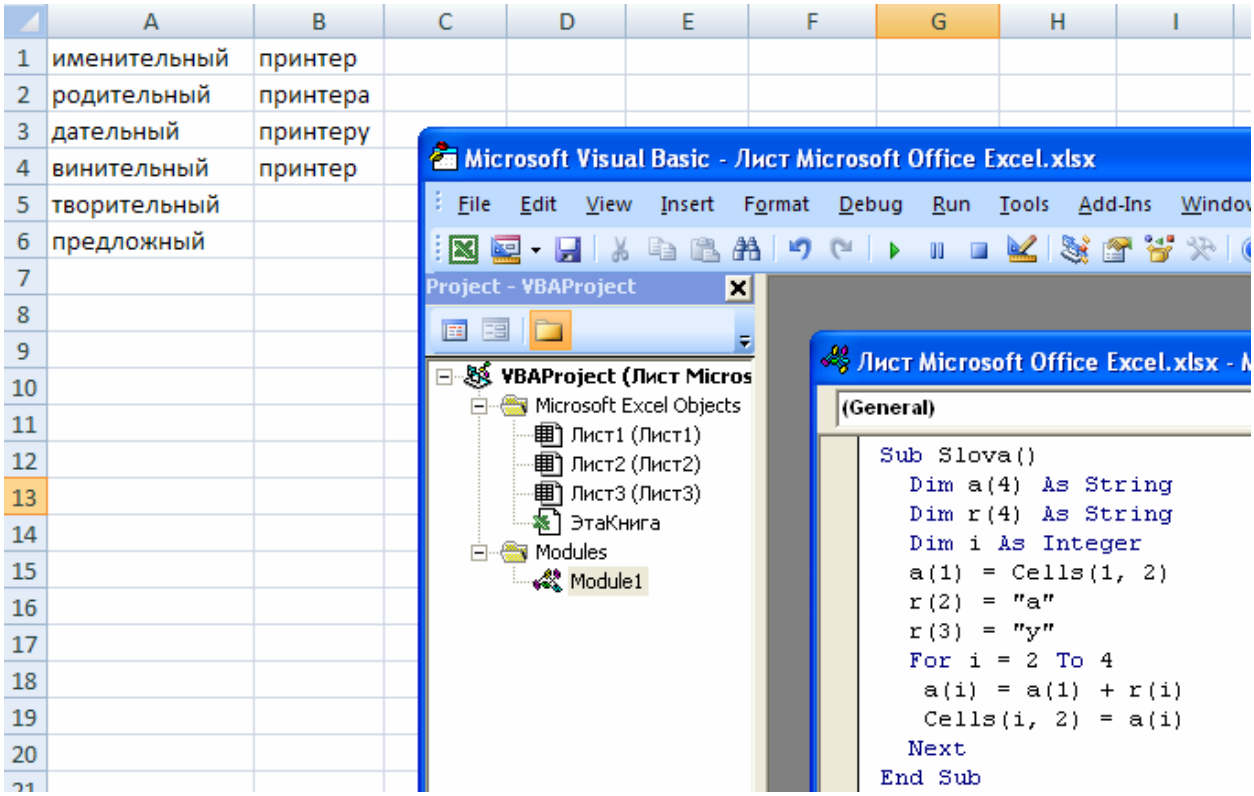
ЗАДАНИЕ 19. Составьте программу, которая именительный падеж слова типа *компьютер, стол, стул, блок,. принтер и т.д.* преобразует в родительный, дательный и винительный падежи и формирует соответствующую таблицу изменения слова по падежам.

РЕШЕНИЕ ЗАДАНИЯ 19. Составим первоначальную таблицу Excel:



	A	B	C	D	E
1	именительный	принтер			
2	родительный				
3	дательный				
4	винительный				
5	творительный				
6	предложный				
7					

Программа и результат ее исполнения представлены ниже:



	A	B	C	D	E	F	G	H	I
1	именительный	принтер							
2	родительный	принтера							
3	дательный	принтеру							
4	винительный	принтер							
5	творительный								
6	предложный								
7									
8									
9									
10									
11									
12									
13									
14									
15									
16									
17									
18									
19									
20									

ЗАДАНИЕ 20. Дополните предыдущую программу для преобразования слова в творительный и предложный падежи и занесения их в соответствующие ячейки таблицы. (Формы творительного и предложного падежей: *принтером, принтере*)

ЗАДАНИЕ 21. Составьте программу, которая автоматически дополняет следующую таблицу, изменяя приведенные слова по всем падежам. *При выполнении данного задания рекомендуется объявить двумерный массив типа String и использовать вложенные циклы.*

	A	B	C	D	E	F	G	H	I
1	именительный	принтер	компьютер	дисковод	блок	сканер	кулер	процессор	
2	родительный								
3	дательный								
4	винительный								
5	творительный								
6	предложный								
7									

ЗАДАНИЕ 22. Составьте программу, которая автоматически дополняет следующую таблицу, изменяя приведенные слова по всем падежам:

	A	B	C	D	E	F	G	H	I	J
1	именительный	клавиатура	дискета	плата	микросхема	процедура	сигнатура	программа		
2	родительный									
3	дательный									
4	винительный									
5	творительный									
6	предложный									
7										
8										

ЗАДАНИЕ 23. Составьте программу, которая автоматически дополняет следующую таблицу, изменяя приведенные словосочетания по всем падежам:

	A	B	C	D	E	F	G	H
1	именительный	стандартная клавиатура	новая дискета	материнская плата	интегральная микросхема	встроенная процедура	постоянная сигнатура	собственная программа
2	родительный							
3	дательный							
4	винительный							
5	творительный							
6	предложный							
7								

РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА

1. Гарнаев А.Ю. Excel, VBA, Internet в экономике и финансах. – Спб.:БХВ-Петербург, 2002, - 816 с.
2. Информатика. Базовый курс. Под. ред. С.В.Симоновича: СПб, «Питер», 2006, 640с.

Рустам Гумерович Тахавутдинов

Основы алгоритмизации и программирования. Часть IV: Элементы структурного программирования. Типы переменных. Методические указания к лабораторным работам, расчетному заданию и самостоятельной работе студентов
(Кафедра Информатики и информационных управляющих систем КГЭУ)

Редактор издательского отдела Г. Я. Дарчинова

Изд. лиц. № 03480 от 8.12.00 Темплан издания КГЭУ 2003 г.
Подписано к печати Формат 60 х 84/16
Гарнитура “Times” Вид печати РОМ Бумага “Business”
Физ. печ. л. Усл. печ. л. Уч-изд. л.
Тираж 200 Заказ

Издательский отдел КГЭУ
420066, Казань, Красносельская, 51

Типография КГЭУ
420066, Казань, Красносельская, 51