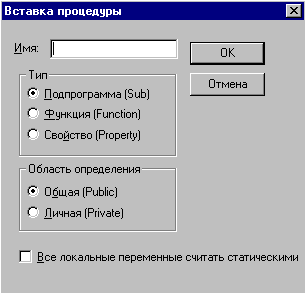
Лабораторная работа №8

VBA. ПОЛЬЗОВАТЕЛЬСКИЕ ПРОЦЕДУРЫ.

**Процедура** – это набор описаний и инструкций, сгруппированных для вычисления. Существует три типа процедур **Sub, Function, Property**. Имя процедуры всегда определяется на уровне модуля. В процедурах должны содержаться все исполняемые программы. Вложенность процедуры в другие процедуры не допускается. Для того, чтобы вставить процедуру необходимо вызвать редактор **VBA** и в меню редактора **Вставка** выбрать подменю **Модуль**. Затем снова выбрать **Вставка** и, в открывшемся подменю выбрать **Процедура**…. Появляется диалоговое окно ***Вставка процедуры*** следующего вида:



В окне **Имя** пишется название процедуры. В рамке **Тип** выбирается необходимая процедура. А в рамке Область определения выбирается необходимая область определения для вашей процедуры.

Инструкция **Sub** описывает имя, аргументы и текст программы, составляющий тело процедуры **Sub**.

Синтаксис

[Private | Public] [Static] Sub имя [(списокАргументов)]

инструкции

[Exit Sub]

[инструкции]

End Sub

Синтаксис инструкции **Sub** содержит следующие элементы:

**Элемент** Описание

**Public** Необязательный. Указывает, что процедура **Sub** доступна для всех других процедур во всех модулях. При использовании в личном модуле (модуле, который содержит инструкцию Option Private) такая процедура является недоступной вне проекта.

**Private** Необязательный. Указывает, что процедура **Sub** доступна для других процедур только того модуля, в котором она описана.

**Static** Необязательный. Указывает, что локальные переменные процедуры **Sub** сохраняются в промежутках времени между вызовами этой процедуры. Атрибут Static не действует на переменные, описанные вне процедуры **Sub**, даже если они используются в процедуре.

**имя** Обязательный. Имя процедуры **Sub**, удовлетворяющее стандартным правилам именования переменных.

**списокАргументов** Необязательный. Список переменных, представляющий аргументы, которые передаются в процедуру **Sub** при ее вызове. Имена переменных разделяются запятой.

**инструкции** Необязательный. Любая группа инструкций, выполняемых в процедуре **Sub**.

**Дополнительные сведения**

Процедуры **Sub**, тип которых не указан явно с помощью слов Public или Private, являются общими по умолчанию. Если не используется слово Static, значения локальных переменных не сохраняются между вызовами процедур.

Все выполняемые команды должны содержаться в процедурах. Не допускается определение процедуры **Sub** внутри другой процедуры **Sub, Function или Property**.

Инструкция Exit Sub приводит к немедленному выходу из процедуры **Sub**. Выполнение программы продолжается с инструкции, следующей за инструкцией, содержащей вызов процедуры **Sub**. В любом месте процедуры **Sub** допускается наличие любого числа инструкций **Exit** **Sub**.

Подобно процедурам **Function**, процедура **Sub** является самостоятельной процедурой, которая может получать аргументы, выполнять последовательность инструкций и изменять значения своих аргументов. Однако в отличие от процедуры **Function**, которая возвращает значение, процедура **Sub** не может применяться в выражении.

Если в процедуре используются общие переменные **(Public)** и они получают там новые, то при вызове этой процедуры другой процедурой **Sub**, в которой так же используются эти переменные, они будут содержать уже полученные значения. А вызвать одну процедуру другой можно следующим способом:

Процедура **Sub** вызывается в выражении по своему имени, за которым следует список аргументов в скобках.

Для вызова процедур **Sub** можно использовать два способа.

В первом случае используется ключевое слово **Call**.

**Call Primer1(0)**

Вышеуказанная строка вызывает процедуру Primer с аргументом 0 .

Если используется процедура без аргументов, скобки не используются

- **Call Primer2**

Во втором случае ключевое слово **Call** не используется – указывается лишь имя процедуры, при наличии аргументов они перечисляются без скобок.

**Primer1 0**

**Primer2**

Переменные, используемые в процедурах **Sub**, разбиваются на две категории: явно описанные внутри процедуры и не описанные внутри процедуры. Переменные, которые явно описаны в процедуре (с помощью ключевого слова **Dim** или эквивалентного ему **Private)**, всегда являются локальными для этой процедуры. Переменные, которые используются, но явно не описаны в процедуре, также являются локальными, если они явно не описаны на более высоком уровне.

Чтобы создать переменную с областью видимости на уровне модуля, она объявляется с помощью ключевого слова Dim внутри этого модуля за пределами всех процедур. Такой уровень видимости применяется в случаях, когда доступ к переменной необходим нескольким процедурам этого модуля.

# Пример для выполнения

Необходимо вычислить площадь параллелепипеда со сторонами a, b, c. Для этого нужно создать две процедуры – **Plosh** и **Argum**. В процедуре **Plosh** обеспечить ввод a, b, c и вывод результата **S**. А в **Argum** обеспечить расчет площади **S**. Причем, процедура **Plosh** должна получить результат **S** путем вызова процедуры **Argum**.

# Решение

Public a As Integer ' Описываются общие переменные в модуле за пределами всех процедур.

Public b As Integer

Public c As Integer

Public S As Integer

**Public Sub Plosh()**

a = Range("a1").Value

b = Range("a2").Value

c = Range("a3").Value

**Call Argum** ' Вызов процедуры Argum

Range("a5").Value = S

**End Sub**

**Public Sub Argum**()

S = a \* b \* c

**End Sub**

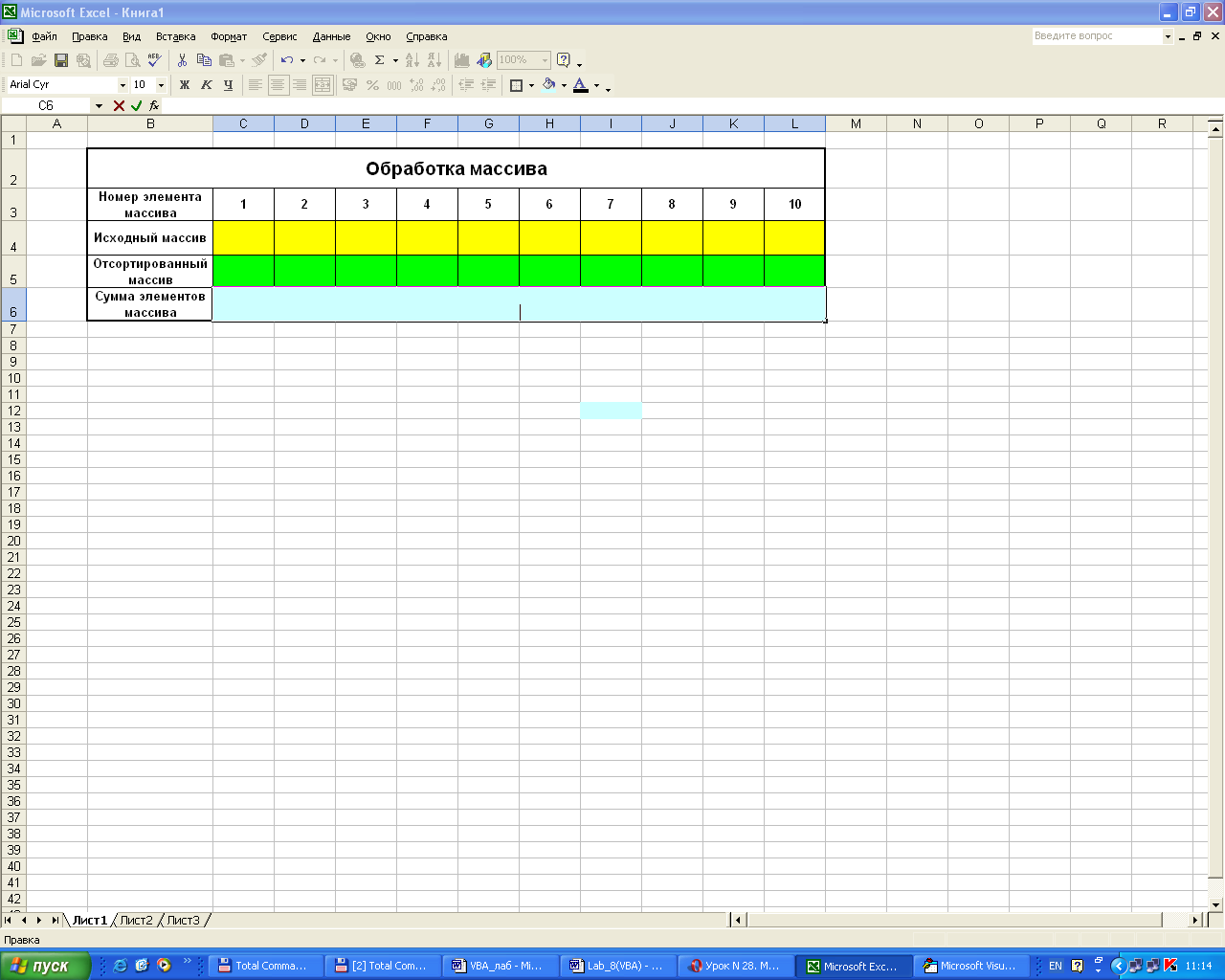
# Задания для выполнения:

Задание 1. Создать одномерный массив из 10 элементов. Его ввод обеспечить, используя InputBox .

Необходимо отсортировать массив по возрастанию, а так же найти сумму всех его элементов. Результаты вывести на рабочий лист Excel.

***Задачу решить при помощи создания:***

- **Макроса** построения на листе Excel шаблона таблицы для вывода результатов



- **Процедуры** ввода элементов массива и вывода исходного массива на рабочий лист в соответствующую строку таблицы.

- **Процедуры** сортировки элементов массива.

Пример алгоритма сортировки массива размерностью n

FOR i = 1 TO (n-1)

FOR j = i + 1 TO n

IF a(j) > a(i) THEN

b = a(i)

a(i) = a(j)

a(j) = b

END IF

NEXT j

NEXT i

- **Процедуры** нахождения суммы элементов массива.

- **Процедуры** вывода элементов отсортированного массива на рабочий лист в соответствующую строку таблицы.

- **Головной программы** (процедуры) для запуска в нужной последовательности процедур для решения задачи.

Задание 2. Создать одномерный массив из 10 элементов. Его ввод обеспечить, используя InputBox .

Найти максимальный и минимальный элементы, а так же найти среднее арифметическое массива. Задачу разрешить при помощи создания трех процедур. Результаты вывести на рабочий лист Excel.

Задание 3. Создать одномерный массив из 10 элементов. Его ввод обеспечить, используя InputBox .

Поменять местами соседние элементы массива и вывести произведение элементов, стоящих на чётных местах преобразованного массива. Результаты вывести на рабочий лист Excel.