**Параметры процедур и функций**

Глобальные переменные (описанные в главной программе) доступны из любого места программы или подпрограммы, кроме тех случаев, когда в подпрограмме используются локальные переменные с такими же именами. Значит, такие переменные можно использовать для обмена данными между основной программой и подпрограммой. Однако такой способ создает неудобства при отладке и сужает возможность использования [процедур](http://informatics-lesson.ru/pascal/procedure-user.php) и [функций](http://informatics-lesson.ru/pascal/user-function.php) в других программах.

* параметр, перед которым отсутствует ключевое слово **var** или **const** называется **параметром-значением**, например:

**function AddSquare(a, b: integer): integer;**

* параметр, перед которыми стоит ключевое слово **var** или **const** называется **параметром-переменной**, например:

**procedure Mult2(var a: integer);**

*Формальный параметр-значение* при вызове подпрограммы получает, специально созданную в области памяти, копию значения соответственного фактического параметра. В подпрограмме выполняются операции с копией без доступа к переменной вызывающей программы. После окончания работы подпрограммы выделенная под параметр-значение память освобождается. Этот способ называется *передачей по значению*.

*Формальный параметр-переменная* при вызове подпрограммы получает адрес области памяти, в которой хранится соответствующий фактический параметр. Поэтому любое изменение формального параметра-переменной в подпрограмме приводит к изменению переменной в вызывающей программе. Такой способ называется *передачей по ссылке (или по адресу)*.

*Пример. В программе используется процедура «Удвоитель».*

* **procedure Mult2(a: integer; var b: integer);**
* **begin**
* **a \*= 2;**
* **b \*= 2;**
* **end;**
* **begin**
* **var num1: integer := 10;**
* **var num2: integer := 10;**
* **Mult2(num1, num2);**
* **writeln(num1);**{вывод 10}
* **writeln(num2);**{вывод 20}
* **end;**
* Переменная **num1** передается по значению, поэтому ее значение не меняется.
* Значение переменной **num2** изменяется, т. к. переменная передается по ссылке (о чем свидетельствует наличие ключевого слова **var** перед соответственным формальным параметром **b**).

**Формальными параметрами** называются переменные описанные в заголовке программы и используемые только внутри процедуры (функции).

|  |
| --- |
| **Procedure** <имя процедуры>{(формальные параметры: их тип)}; {Var   (локальные переменные)} begin   . . . end; |

        Формальные параметры указывают, с какими параметрами следует обращаться к этой процедуре (функции), т.е. их количество, последовательность, тип. Они задаются в заголовке процедуры (функции) в виде списка параметров, разбитого на группы (секции) однотипных параметров, разделенных ‘;’. Порядок следования групп (секций) произвольный. Количество параметров м.б. любым.  
        Для каждого формального параметра следует указать имя и тип. Имена параметров могут быть любыми, они могут совпадать с глобальными именами программы. Необходимо лишь помнить, что в этом случае параметр основной программы с таким именем становится недоступным для непосредственного использования подпрограммой. Но формальные имена не могут совпадать с локальными переменными!!!!, т.к. они располагаются в одном стековом сегменте памяти. Тип формального параметра может быть практически любым, однако в заголовке процедуры (функции) нельзя вводить новый тип.

 Формальные параметры можно разбить на 2 категории:

1. параметры - значения (эти параметры в основной программе процедурой (функцией) не меняются );

2. параметры - переменные (эти параметры процедура (функция) может изменить в основной программе );

**Параметрами - значениями** называются переменные, которые могут изменяться внутри процедуры (функции), но не передают свои значения в программу.

**Параметрами - переменными** называются переменные, которые изменяются в процедуре (функции) и передают свое значение в программу.  
        Перед параметрами-переменными ставится служебное слово **Var.**        Список формальных параметров обеспечивает связь процедуры (функции) с вызывающей программой. Через него в процедуру (функцию) передаются исходные данные и возвращается результат.

        При вызове процедуры (функции) из программы указываются фактические параметры.

**Фактическими параметрами** называются параметры, которые задаются при вызове процедуры (функции).

**Процедура вызывается по имени:**

|  |
| --- |
| <имя процедуры> {(фактические параметры)}; |

**Правила задания фактических параметров:**

1. Фактические параметры должны совпадать с формальными по количеству, по порядку следования, по типу.

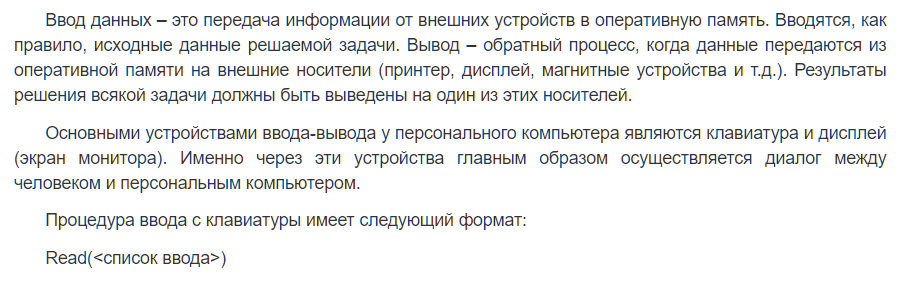
2. Если формальный параметр – параметр-переменная, то фактический параметр при вызове процедуры задается только идентификатором (вызов по ссылке).

3. Если формальный параметр – параметр- значение, то фактический параметр при вызове процедуры задается:

* идентификатором
* константой
* выражением

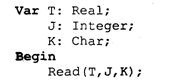
(вызов по значению).

4.  Для параметров-переменных (вызов по ссылке) должно быть точное совпадение по типу фактических и формальных параметров, а для параметров-значений (вызов по значению) допустима совместимость.



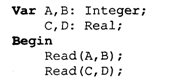
где <список ввода> **–**это последовательность имен переменных, разделенных запятыми. (Read **–**это оператор обращения к стандартной процедуре ввода.)

Пример:



Если в программе имеется несколько операторов Read, то данные для них вводятся потоком, т. е. после считывания значений переменных для одного оператора Read данные для следующего оператора читаются из той же строки на экране, что и для предыдущего до окончания строки, затем происходит переход на следующую строку.

Пример:



Другой вариант оператора ввода с клавиатуры имеет вид:

https://3ys.ru/images/lib/yazyki-i-sistemy-programmirovaniya/184b982418cd8eb419eaa3f62a7561f7/9102468dd62a0fdb74aa321125633424.gif

Здесь слово ReadLn означает read line **–**читать строку. Этот оператор отличается от Read только тем, что после считывания последнего в списке значения для одного оператора ReadLn данные для следующего оператора будут считываться с начала новой строки. Если в предыдущем примере заменить операторы Read на ReadLn :

https://3ys.ru/images/lib/yazyki-i-sistemy-programmirovaniya/184b982418cd8eb419eaa3f62a7561f7/07f02838b329e3bebd581e9582142f4e.gif

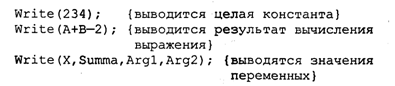
то ввод значений будет происходить из двух строк.

Оператор вывода на экран (обращение к стандартной процедуре вывода) имеет следующий формат:

https://3ys.ru/images/lib/yazyki-i-sistemy-programmirovaniya/184b982418cd8eb419eaa3f62a7561f7/8d58d32af2fbd727e33ed6cd6aab3a68.gif

Здесь элементами списка вывода могут быть выражения различных типов (в частности, константы и переменные).

Пример:



При выводе на экран нескольких чисел в строку они не отделяются друг от друга пробелами. Программист сам должен позаботиться о таком разделении. Пусть, например, I = 1; J = 2, К = 3. Тогда, написав в программе

https://3ys.ru/images/lib/yazyki-i-sistemy-programmirovaniya/184b982418cd8eb419eaa3f62a7561f7/3c109e1ce61f2944e54ccc0a44615da8.gif

получим на экране строку: 1 2 3. После вывода последнего символа курсор остается в той же строке. Следующий вывод на экран будет начинаться с этой позиции курсора.

Второй вариант процедуры вывода на экран:

https://3ys.ru/images/lib/yazyki-i-sistemy-programmirovaniya/184b982418cd8eb419eaa3f62a7561f7/54162c35e0ffe350c3b35b710b28e2e5.gif

Слово WriteLn **–**write line **–**означает писать строку. Его действие отличается от оператора Write тем, что после вывода последнего в списке значения происходит перевод курсора к началу следующей строки. Оператор WriteLn, записанный без параметров, вызывает перевод строки.

В списке вывода могут присутствовать указатели форматов вывода (форматы). Формат определяет представление выводимого значения на экране. Он отделяется от соответствующего ему элемента двоеточием. Если указатель формата отсутствует, то машина выводит значение по определенному правилу, предусмотренному по умолчанию.

Ниже кратко, в справочной форме, приводятся правила и примеры бесформатного и формати­рованного вывода величин различных типов. Для представления списка вывода здесь будут использованы следующие обозначения:

I, Р, Q **–**целочисленные выражения;

R **–**выражение вещественного типа;

В **–**выражение булевского типа;

Ch **–**символьная величина;

S **–**строковое выражение;

# **–**цифра;

\* **–**знак «+» или «—»;

\_ **–**пробел.

**Управление выводом на экран**

Использование для вывода на экран только процедур Write и WriteLn дает программисту очень слабые возможности для управления расположением на экране выводимого текста. Печать текста может производиться только сверху вниз, слева направо. Невозможны возврат к предыдущим строкам, стирание напечатанного текста, изменение цвета символов и т.д.

Дополнительные возможности управления выводом на экран дают процедуры и функции модуля CRT.

Для установления связи пользовательской программы с модулем перед разделами описаний должна быть поставлена строка

Uses CRT