Пользовательские процедуры и функции языка Паскаль.

В языке Паскаль, как и в большинстве языков программирования, предусмотрены средства, позволяющие оформлять последовательность операторов как подпрограмму. Это бывает необходимо, когда такая последовательность неоднократно встречается в программе или когда имеется возможность использовать некоторые фрагменты уже написанных ранее программ. В Паскале такую возможность предоставляют процедуры и функции, определенные пользователем.

Описание процедур и функций располагается в программе в разделе описаний. Сами описания не означают выполнения действий. Действия, предусмотренные процедурой или функцией, выполняются при вызове процедуры или функции из основного блока программы.

1. Функции.

function <имя> (<список параметров>):<тип>;
<разделы локальных описаний>;
<раздел операторов (тело функции)>;

где <имя> - идентификатор функции, не совпадающий с ключевыми словами и не содержащий пробелов и других специальных символов (кроме знака подчеркивания);

<список параметров> - список аргументов, с которыми работает функция, с указанием их типов;

<тип> - тип значения, возвращаемого функцией.

Аргументы, передаваемые в функцию, в описании функции называются ***формальными параметрам***. В точке вызова функции вслед за ее именем в скобках указываются ***фактические параметры***, т.е. те конкретные значения, которые передаются в функцию для обработки.

В точке вызова функции происходит неявное присваивание фактических значений формальным переменным, имена которых перечислены в заголовке функции при ее описании.

В разделе операторов функции обязательно должен присутствовать оператор присваивания вида:

<имя функции>:=<выражение>;

Этот оператор определяет значение, которое будет возвращено функцией в точку ее вызова.

Пример.

Найти максимальное из 4-х чисел.

```
Program max_4;
  uses crt;
  var a, b, c, d, m: integer;
  function max_2(x,y:integer):integer;
    var max:integer;
  begin
    if x>y then
```

```
        max:=x
      else
        max:=y;
      max_2:=max;
  end;
Begin
  clrscr;
  readln(a, b, c, d);
  m:=max_2(a,b);
  m:=max_2(m,c);
  m:=max_2(m,d);
(или m:=max_2(max_2(max_2(a,b), c), d);)
  writeln(m);
End.
```

2. Процедуры.

procedure <имя> [(<список параметров>[; var <параметр>])];
<разделы локальных описаний>;
<раздел операторов (тело процедуры>;

где <имя> - идентификатор процедуры (те же ограничения, что и для функции);

<список параметров> - список аргументов процедуры с указанием их типов (список параметров-значений);

var <параметр> - имя параметра-переменной, т.е. той переменной, значение которой будет возвращено в точку вызова процедуры.

Процедура может не содержать никаких параметров. В этом случае выполняются операторы, стоящие в теле процедуры. Примером такой процедуры может служить процедура вычерчивания рамки на текстовом экране:

```
procedure ramka;
  var i,j:integer;
begin
  for i:=1 to 80 do
     begin
        gotoxy(i, 1);
        write('*');
        gotoxy(i, 25);
        write('*');
     end;
  for j:=1 to 25 do
     begin
        gotoxy(1, j);
        write('*');
        gotoxy(80, j);
        write('*');
```

```
        end;
    end;
```

Если же в заголовке процедуры указаны параметры, то в точке вызова процедуры происходит неявное присваивание фактических значений параметрам-значениям, а по окончании действия процедуры в точку вызова возвращаются значения параметров-переменных.

Пример.

Найти максимальное из 4-х чисел.

```
Program max_4;
  uses crt;
  var a, b, c, d, m: integer;
  procedure max_2(x,y:integer; var max:integer);
  begin
    if x>y then
     max:=x
    else
     max:=y;
  end;
Begin
  clrscr;
  readln(a, b, c, d);
  max_2(a,b,m);
  max_2(m,c,m);
  max_2(m,d,m);
  writeln(m);
End.
```

3. Локальные и глобальные переменные.

Переменные описываются в программе в разделе описания переменных. Если же программа содержит описание процедуры или функции, то некоторые переменные могут быть описаны в этой процедуре или функции в разделе локальных описаний.

Переменные, описанные в главной программе, называются ***глобальными***. Ими можно пользоваться и в главной программе, и в процедуре или функции.

Переменные, описанные в процедуре или функции, называются ***локальными***. Ими можно пользоваться только в той процедуре или функции, где они описаны. Вне тела процедуры или функции значения таких переменных не определены.

Замечание.

При использовании локальных переменных необходимо следить за тем, чтобы их идентификаторы не совпадали с именами глобальных переменных.

Если процедура или функция изменяет значение глобальной переменной, то говорят, что она имеет побочный эффект.

Пример.

Пользовательские процедуры и функции

1)

```
Program summa;
    var a, b, s: integer;
    procedure newval;
        begin
            a:=1;
            b:=1;
         end;
Begin
    a:=0;
    b:=0;
    newval;
    s:=a+b;
    writeln(s);
End.
```

2)

```
Program summa;
    var a, b, s: integer;
    procedure newval;
        var b: integer;
        begin
            a:=1;
            b:=1;
         end;
Begin
    a:=0;
    b:=0;
    newval;
    s:=a+b;
    writeln(s);
End.
```

4.Параметры процедур и функций.

Имена переменных, перечисленных в заголовке процедуры (функции), носят название *формальных параметров.* Они служат для обозначения тех данных, с которыми процедура (функция) будет работать после вызова.

Процедура (функция) вызывается главной программой или другой процедурой (функцией). Обращение к процедуре или функции называется ***точкой вызова процедуры (функции).*** При вызове необходимо задать значения параметров, с которыми будет работать процедура (функция). Эти значения называются *фактическими параметрами.*

Соответствующие друг другу формальные и фактические параметры должны принадлежать одному и тому же типу.

При вызове процедуры необходимо знать способ подстановки фактических параметров:

- подстановка значения;
- подстановка переменной.

Пример 1.

```
Program sum;
  var p,q,s: integer;
  procedure newval (r,t:integer);
  begin
    r:=r+1;
    t:=t+2;
  end;
Begin
  p:=0;
  q:=0;
  newval(p,q);
  s:=p+q;
  writeln(s);
End.
```

Здесь в описании процедуры формальные параметры описаны как параметры значения. Это означает, что при вызове процедуры вместо них будут подставлены значения фактических параметров. В главной программе вызов процедуры осуществляется оператором ***newval(p,q)***. При выполнении этого оператора сначала вместо формальных параметров ***r*** и ***t*** будут подставлены значения фактических параметров, т.е. неявно будут выполнены операторы:

```
r:=p; t:=q;
```

В результате выполнения процедуры ***r*** и ***t*** получат значения: r=1; t=2. Однако, никаких связей между ***r*** и ***p***, ***t*** и ***q*** в теле процедуры не установлено. Поэтому после выполнения процедуры ***p*** и ***q*** будут иметь такие же значения, что и до ее выполнения. В результате ***s*** получит значение 0.

Пример 2.

```
Program sum;
  var p,q,s: integer;
  procedure newval (r:integer; var t:integer);
  begin
    r:=r+1;
    t:=t+2;
  end;
Begin
  p:=0;
  q:=0;
  newval(p,q);
  s:=p+q;
  writeln(s);
```

```
End.
```

Здесь формальный параметр ***t*** описан как параметр-переменная, а ***r*** – как параметр-значение. В главной программе при выполнении оператора ***newval(p,q)*** вместо формального параметра ***r*** будет подставлено значение ***p,*** т.е. неявно будет выполнен оператор присваивания r:=p. Вместо же параметра ***t*** будут подставлена переменная ***q***, т.е. переменные ***t*** и ***q*** будут считаться синонимами.

В результате после выполнения процедуры ***p*** будет иметь значение 0, а ***q*** получит значение 2. Следовательно, ***s*** будет равно 2.

Пример 3.

```
Program sum;
   var p,q,s: integer;
   procedure newval (var r:integer; var t:integer);
   begin
      r:=r+1;
      t:=t+2;
   end;
Begin
   p:=0;
   q:=0;
   newval(p,q);
   s:=p+q;
   writeln(s);
End.
```

В этом случае оба параметра процедуры описаны как параметры-переменные. Следовательно, в результате выполнения процедуры ***s*** получит значение 3.

Вывод.

Если параметр процедуры является ее аргументом, предпочтительнее подстановка значения. Если же параметр процедуры есть ее результат, то необходима подстановка переменной.

Правила пунктуации.

Точка с запятой: 1) является разграничителем операторов;
2) наличие между операторами нескольких точек с запятой не является ошибкой;
3) не ставится после type, const, var;
4) ставится после завершения каждого описания;
5) не ставится после begin и перед end;
6) не ставится после then и перед else;
7) не ставится после while, repeat, do и перед until.