

**Тема: Процедуры и функции.
Заголовок и тело процедур и
функций, классификация
параметров. Вызов процедур и
функций. Особенности их
использования**

Подпрограммы на языке Паскаль

- Если какой то участок основной программы повторяется в тексте несколько раз и в разных местах, то этот участок можно оформить в виде подпрограммы.
- Подпрограммы могут быть оформлены в виде процедур и в виде функций.



Нарисовать елку с помощью

Program Elka;

*{Нарисовать елку с помощью *}*

*Procedure Treugolnik; {имя процедуры
(подпрограммы)}*

begin {начало подпрограммы}

*Writeln(' *');*

*Writeln(' * *'); { тело подпрограммы
(процедуры)}*

*Writeln(' * *');*

*Writeln('*****');*

end; {конец подпрограммы (процедуры)}

Begin {начало основной программы}

Treugolnik;

*Treugolnik; {команда вызова процедуры
(подпрограммы)}*

Treugolnik;

Treugolnik;

End. {конец программы}

Оформление

- Любую подпрограмму можно оформить в виде **процедуры**. Обычно из основной программы в процедуру некоторые параметры передаются (аргументы процедуры), а другие параметры после обработки передаются из процедуры в основную программу (результаты процедуры).
- Если из процедуры в программу **передается** только **один** параметр (результат), то такую процедуру можно оформить в виде **функции**.
- Возможны процедуры, которые вообще не связаны с передачей параметров из основной программы в основную программу.

Процедуры и функции имеют такую же структуру разделов, как и основная программа на языке Паскаль.

- Тексты процедур и функций записываются в конце раздела описаний основной программы (перед служебным словом `begin`).*

ПРОЦЕДУРЫ

Формат:

```
Procedure <имя > (формальные  
    параметры: тип);  
<разделы описаний>;  
    begin  
        <раздел операторов>  
    end;
```

Пример 1

procedure Min (i, j : integer; x, y: real);

- Заданы только вводимые параметры (аргументы): i, j – целые,
- x, y – вещественные величины.

Пример 2

procedure Max (i, j: integer; var u: real);

- вводимые параметры (аргументы):
 - i, j – целые числа,
- выводимые параметры (результат):
 - u – вещественное число.
- Выводимые параметры (результаты) записываются с использованием служебного слова **var**.
- Если выводимые параметры разных типов, то слово var записывается перед **каждым** типом данных.

Пример 3

procedure Min (i: integer; x: real; var j: integer; var u, v: real);

В тексте процедуры описывать величины, которые введены в заголовке, ещё раз не надо.

Вызов процедуры

- *В основной программе для вызова процедуры используется **оператор вызова**.*
- *Оператор вызова имеет вид имени процедуры, после которого в круглых скобках записываются величины, которые используются в основной программе, а также конкретные числа, текстовые строки и т.п.*

Пример вызова процедуры

procedure Min (i, j : integer; x, y: real);

Min (k, 10, z, 2.35);

i получает значение величины k
основной программы,

j значение 10,

x значение z,

y значение 2,35.

procedure Max (i, j: integer; var u: real);

Max (l, 35, x);

Главное при записи оператора вызова процедуры

- это совпадение типов величин и значений, которые записаны в операторе вызова с типами, которые определены в заголовке процедуры.
 - Формальные параметры – это величины, которые введены в заголовке процедуры, в скобках.
 - Фактические параметры – это величины, которые используются в основной программе.
 - Между фактическими и формальными параметрами должно существовать соответствие:
 - По количеству параметров;
 - Порядку их следования;
 - Типу данных.

Глобальные и локальные переменные

- **Глобальные переменные** описываются в **основной** программе и действуют как в основной программе, так и **во всех** ее подпрограммах.
- **Локальные переменные** описываются в **процедуре** и действуют **только** в пределах **этой** подпрограммы.

Задача: Задача: Дан 2-х мерный массив целых чисел $A(m \times n)$, где $m \leq 10$, $n \leq 10$. Изменить массив таким образом, чтобы в чётных строках отрицательные элементы заменить 0. Вывести изменённый массив. Ввод, вывод и изменение оформить в виде процедур.

Program Proc;

**{В четных строках
отрицательные заменить на 0}**

var A: array [1..10, 1..10] of integer;

l,k,i,j: integer;

procedure vvod ;

begin

for i:= 1 to l do

for j:= 1 to k do read (A[i,j]);

writeln

end;

procedure zam;

begin

for i:= 1 to l do

for j:= 1 to k do

if i mod 2 = 0 then

if A [i,j] < 0 then A [i, j] := 0

end;

procedure out ;

begin

for i:= 1 to l do

begin

for j:= 1 to k do write (A[i, j], ' ');

writeln;

end;

end;

Begin

readln (l,k);

vvod ;

zam;

out ;

End.

ФУНКЦИИ

Формат:

function <имя функции>(список формальных параметров):тип результата функции;

В качестве выводимого параметра используется имя функции.

Пример описания функции

*Нахождение максимального из двух
целых чисел*

```
Function max(a,b:integer):integer;  
begin  
    if a>b then max:=a  
    else max:=b;  
end;
```


Вызов функции

Function max(a,b:integer):integer;

- *В основной программе имя функции используется в выражениях как величина.*
- *После имени функции в скобках вместо формальных параметров через запятую записываются фактические величины или значения.*

z:=0.5+max(4,y);

Пример: Найти площадь 4-х угольника, используя формулу Герона. Вычисление площади треугольника оформить в виде функции.

```
Program func;  
{Найти площадь 4-х угольника, используя  
формулу Герона.}  
Var a, b, c, d, e, s: real;  
Function pl_tr(x1, x2, x3:real): real;  
Var p: real;  
begin  
    P:=(x1+x2+x3)/2;  
    pl_tr:=sqrt(p*(p-x1)*(p-x2)*(p-x3  
));  
end;
```

```
Begin  
    writeln('Введите стороны четырехугольника');  
    Readln(a, b, c, d, e);  
    S:=pl_tr(a, b, c)+pl_tr(c, d, e);  
    writeln('S=', s:4:2);  
End.
```

Задания(использовать процедуры и функции)

1. Дан двумерный массив. Положительные элементы заменить на 2, а отрицательные – на -2.
2. Ввести два целых числа. Найти их сумму, разность, частное и произведение.
3. Треугольник задан координатами своих вершин. Найти периметр треугольника. Вычисление длины отрезка оформить в виде подпрограммы.