

# Процедуры и функции

# Содержание

**Процедуры**

**Глобальные и  
локальные переменные**

**Функции**

**Формальные и  
фактические параметры**

**Рекурсия**

**Механизм передачи  
параметров в функции и  
процедуры**

## Подпрограммы: глобальные и локальные переменные

Все подпрограммы должны быть описаны в разделе описаний. Каждая подпрограмма должна иметь имя.

Информация между основной программой и подпрограммами передается **глобальными параметрами (переменными)**, действующими в любой части программы, имеющими имя, описанное в основной программе.

Внутри подпрограммы могут быть использованы **локальные параметры (переменные)**, – их имена и значения имеют смысл только в пределах границ данной подпрограммы и недоступны вызывающей программе



# Формальные и фактические параметры

В описании подпрограмм параметры обозначены только именами, поэтому их называют **формальными**. До вызова подпрограммы они не имеют значений. Они лишь резервируют место для **фактических** параметров, фиксируя их число и тип данных.



## Типы фактических параметров:

Параметры-значения показывают, какое значение надо присвоить определенному параметру подпрограммы. После завершения подпрограммы они принимают прежние значения, даже если были изменены в подпрограмме.

Параметры-переменные в подпрограмме становятся на место формальных, могут в ходе исполнения подпрограммы изменить свое значение и сохраняют изменения при выходе из подпрограммы (перед параметрами-переменными стоит ключевое слово Var).

# Процедуры

Решение задач

# Описание процедуры

Program Pr1;

Const ...

Type ...

Var ...

**Procedure <имя процедуры>( <список формальных параметров>);**

**Описательная часть**

**Begin**

**Тело процедуры**

**End;**

Begin

...

**<имя процедуры>( <список фактических параметров>);**

...

**end.**

При вызове процедуры  
формальные параметры  
заменяются фактическими.



# Процедура вычисления суммы двух чисел

```
program pr1;  
Uses crt;  
Var a,b,s:real;
```

**a,b,s** – глобальные переменные

**x,y,z** – формальные параметры, локальные переменные

```
procedure summa(x,y:real;var z:real);
```

```
begin
```

```
z:=x+y;
```

```
end;
```

```
begin
```

```
clrscr;
```

```
writeln('введите a,b'); readln(a,b);
```

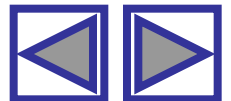
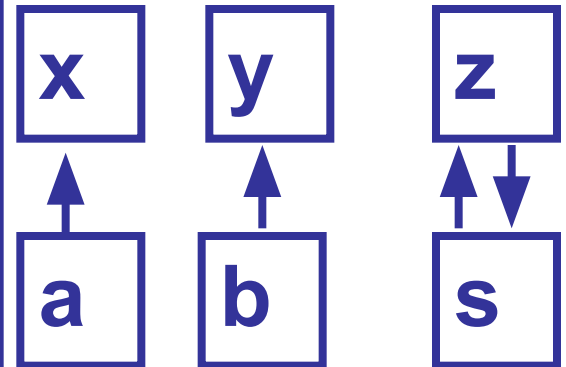
```
summa(a,b,s);
```

**a,b,s** – фактические параметры

```
writeln(' сумма чисел ',a:3:1,' и ',b:3:1,' =  
,s:3:1);
```

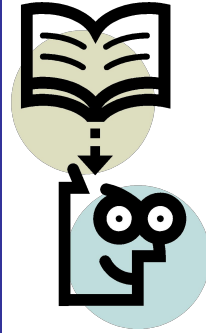
```
readln; end.
```

Параметры-значения    Параметр-переменная

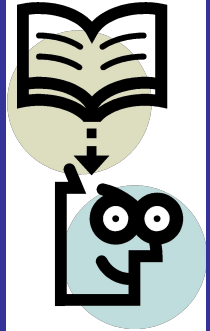


# Программа перестановки значений переменных $a, b, c$ в порядке возрастания ( $a < b < c$ )

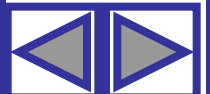
```
program perest;  
var a,b,c: integer;  
procedure swap(var x,y: integer);  
var t: integer;  
begin  
t:=x; x:=y; y:=t;  
end;  
begin  
writeln('Введите три числа ');  
readln(a,b,c);  
if a>b then swap(a,b);  
if b>c then swap(b,c);  
if a>c then swap(a,c);  
writeln(a, ' ', b, ' ', c);  
readln; end.
```



Найдите ошибку в этом решении. Для этого составьте полную систему тестов.



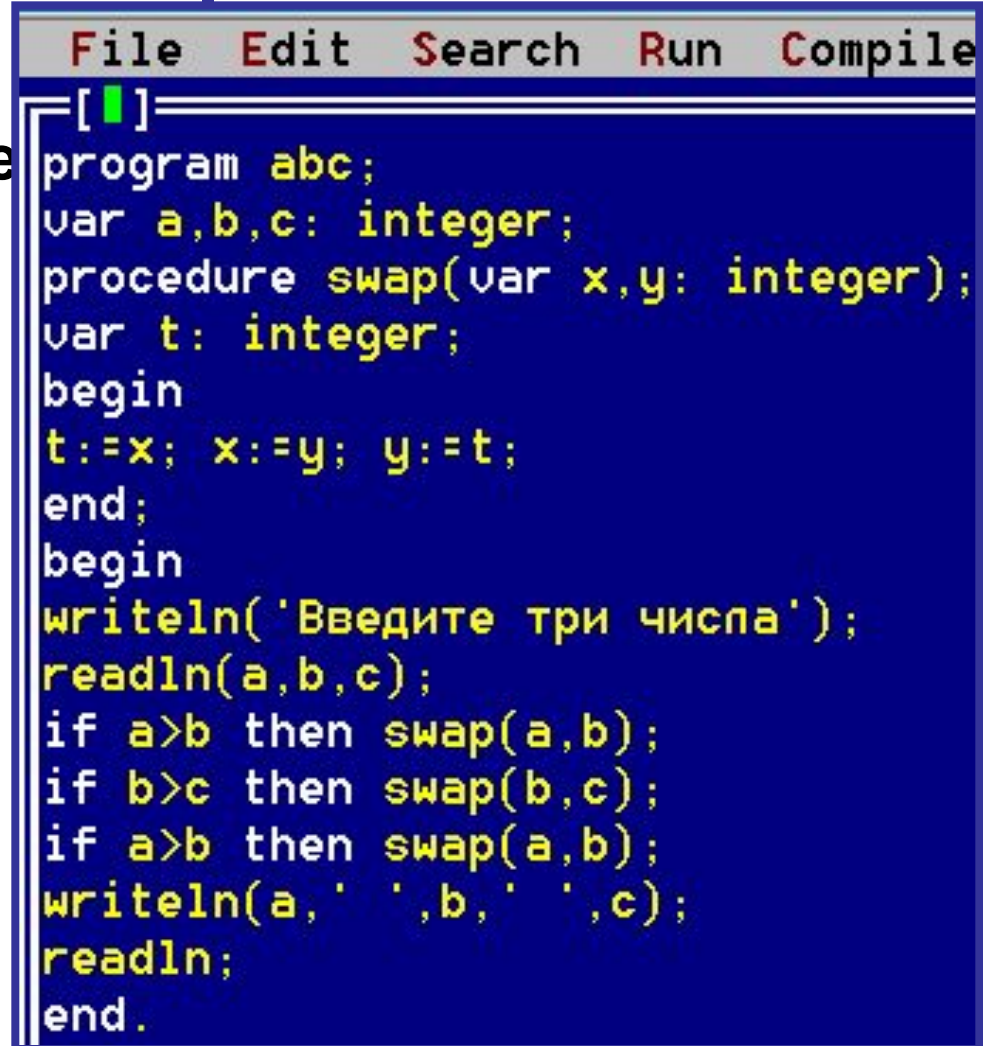
Измените программу так, чтобы аналогичная задача решалась для четырех переменных





# Программа перестановки значений переменных a,b,c в порядке возрастания( $a < b < c$ )

```
program perest;  
var a,b,c: integer;  
procedure swap(var x,y: integer;  
var t: integer;  
begin  
t:=x; x:=y; y:=t;  
end;  
begin  
writeln('Введите три числа ');  
readln(a,b,c);  
if a>b then swap(a,b);  
if b>c then swap(b,c);  
if a>c then swap(a,c);  
writeln(a,' ',b,' ',c);  
readln; end.
```



The screenshot shows a window with a menu bar containing 'File', 'Edit', 'Search', 'Run', and 'Compile'. The main area contains the following Pascal code:

```
[ ]  
program abc;  
var a,b,c: integer;  
procedure swap(var x,y: integer);  
var t: integer;  
begin  
t:=x; x:=y; y:=t;  
end;  
begin  
writeln('Введите три числа');  
readln(a,b,c);  
if a>b then swap(a,b);  
if b>c then swap(b,c);  
if a>c then swap(a,b);  
writeln(a,' ',b,' ',c);  
readln;  
end.
```



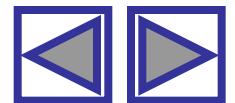
# Вычислить значение выражения

$$a:=(3n!+2m!)/(m+n)!$$

```
program pr2;
Uses crt;
Var m,n,x,y,z:integer; a:real;
procedure fact(d:integer;var q:integer);
  var i:integer;
  begin
    q:=1;
    for i:=1 to d do
      q:=q*i;
  end;
begin
clrscr;
writeln('введите значения n, m '); readln(n,m);
fact(n,x); fact(m,y); fact(m+n,z);
  a:=(3*x+2*y)/z;
writeln('значение выражения при m= ',m:4,' и n= ',n:4,'равно',a:8:3);
readln; end.
```

$$N!=1\cdot 2\cdot 3\cdot \dots \cdot N$$

Для нахождения факториала  
какой тип переменных  
целесообразно использовать?



## Ввод вывод элементов одномерного массива

**Задание:** Оформить ввод элементов одномерного массива с помощью генератора случайных чисел (диапазон значений от -10 до 20) и вывод элементов как процедуры.

Функция **Random(X)** формирует случайное число от 0 до X целого или вещественного типа (перед обращением к функции ее необходимо инициализировать, используя процедуру **Randomize**). Если параметр X не задан, результат будет типа Real в пределах от 0.0 до 1.0.

Для получения массива целых случайных чисел из диапазона [A,B] **random(B-A+1)+A**

Для **A=-10 B=20 random(20-(-10)+1)+(-10)**



# Ввод и вывод элементов одномерного массива

```
[█] PROC2.PAS
program proc2;
const n=8; l=-10; h=20;
type mas=array[1..n] of integer;
var a: mas;
procedure init(t,v,w:integer;var x:mas);
  var i: integer;
  begin
    randomize;
    for i:=1 to t do x[i]:=v+random(w-v+1);
  end;
procedure print(t:integer;var x:mas);
  var i: integer;
  begin
    for i:=1 to t do write(x[i]:5);
  end;
begin
writeln('формирование значений
init(n,l,h,a);
writeln('Вывод ');
print(n,a);
readln;      end.
```



```
формирование значений элементов массива A
Вывод
  10  17  11  -2   2  -6  -8  -2
формирование значений элементов массива A
Вывод
  12  -6 -10  20  20  20  -1   3
формирование значений элементов массива A
Вывод
   8  -4  -9  -5   7  16  10   6
```

# Функции

Решение задач

# Описание функции

Функции предназначены для того, чтобы вычислять только одно значение,

1. поэтому ее первое отличие состоит в том, что процедура может иметь новые значения у нескольких параметров, а функция только одно (оно и будет результатом).
2. Второе отличие заключается в заголовке функции. Он состоит из слова FUNCTION, за которым идет имя функции, затем в круглых скобках идет список формальных параметров, после чего через двоеточие записывается тип результата функции.
3. В теле функции обязательно должен быть хотя бы один оператор присвоения, где в левой части стоит имя функции, а в правой – ее значение.

```
Function <имя>(<список формальных параметров>):<тип результата>
```

```
  Описательная часть
```

```
  Begin
```

```
    Тело функции
```

```
    <имя>:=<значение>;
```

```
  End;
```



# Вычислить значение выражения

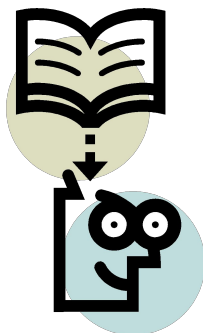
$$a := (3n! + 2m!) / (m+n)!$$

```
program fn2;  
Uses crt;  
Var m,n:integer; a:real;  
function fact(d:integer) :longint;  
  var i:integer; q:longint;  
  begin  
    q:=1;  
    for i:=1 to d do  
      q:=q*i;  
      fact:=q;  
  end;  
  
begin  
clrscr;  
writeln('введите значения n, m '); readln(n,m);  
  a:=(3*fact(n)+2*fact(m))/fact(m+n);;  
writeln('значение выражения при m= ',m:4,' и n= ',n:4,'равно',a:8:3);  
readln; end.
```



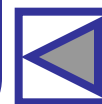
Составить программу, которая будет находить  $a^b$ , то есть  $b$ -ю степень числа  $A$ , где  $A$  и  $B$  – это целые числа и  $B > 0$ , вводимые с клавиатуры.

```
program pr2;  
Uses crt;  
Var a,b:integer;  
    c:longint;  
Function stepen(x,y:integer):longint;  
var i:integer; s:longint;  
begin  
    s:=1;  
    for i:=1 to y do  
        s:=s*x;  
    Stepen:=s;  
end;
```



```
begin  
clrscr;  
writeln('введите  
значения a, b');  
readln(a,b);  
C:=stepen(a,b);  
writeln('s=',s);  
readln;  
end.
```

Составьте программу,  
заменив функцию  
процедурой





# **Механизм передачи параметров в функции и процедуры**

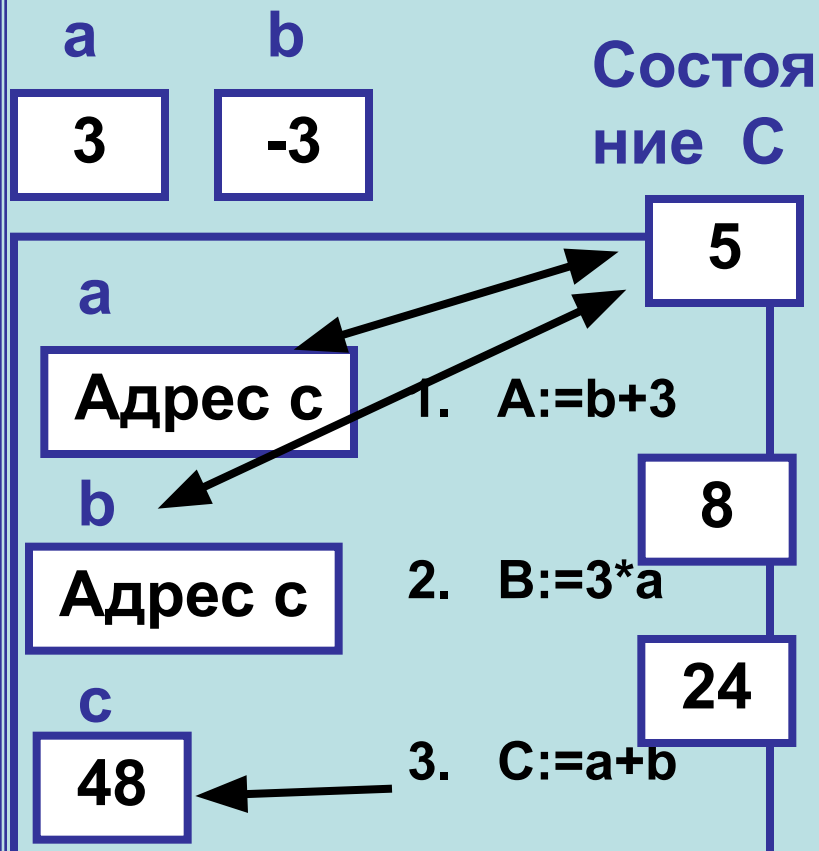
**Разбор заданий**

# Механизм передачи параметров в функции и процедуры

Что будет напечатано процедурой, а что программой?

```
[ ]
program proc4;
var
  a:integer;b,c:integer;
procedure cc(var a,b:integer);
var c:integer;
begin
  a:=b+3; b:=3*a; c:=a+b; {1}
  writeln(c)
end;
begin
  a:=3; b:=-3; c:=5;
  cc(c,c); {2}
  writeln(c)
end.
```

Глобальные переменные



Локальные переменные

Ответ

48  
24

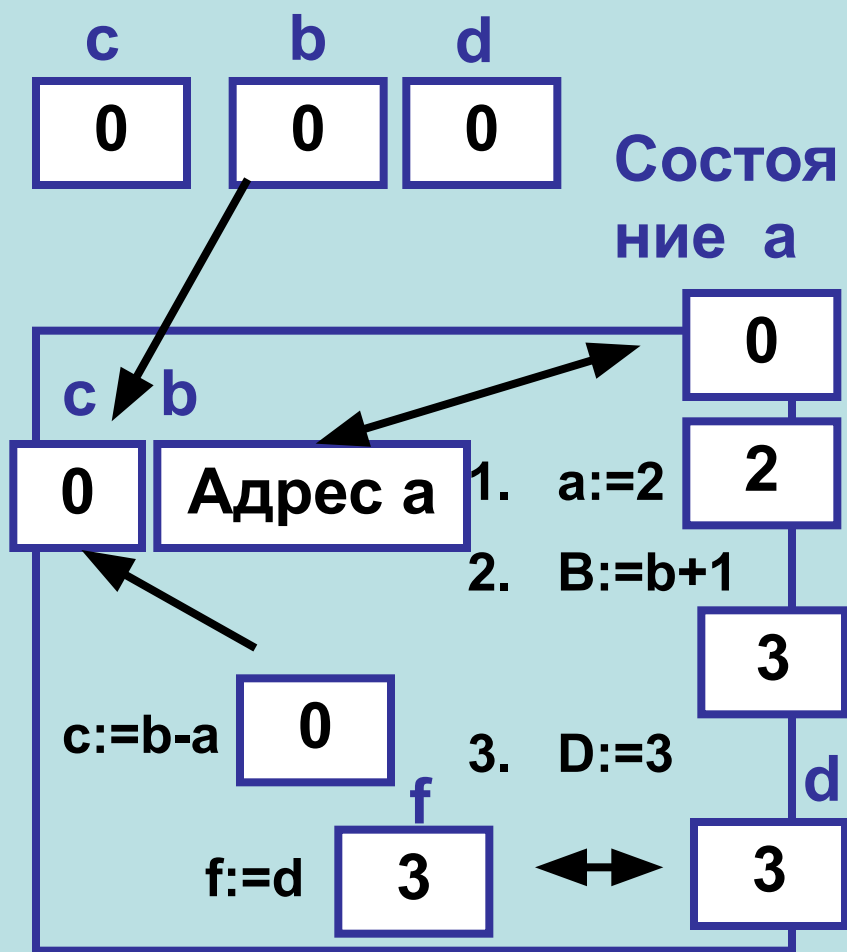
# Механизм передачи параметров в функции и процедуры

Что будет напечатано функцией, а что программой?

```

[1] PROC3.P
program fun3;
var
  a,b,c,d:integer;
function f(var b:integer; c:integer):integer;
var d:integer;
begin
  a:=2; b:=b+1; d:=3; c:=b-a;
  writeln(a:3,b:3,c:3,d:3); f:=d
end;
begin
  a:=0; b:=0; c:=0; d:=0;
  d:=f(a,b);
  writeln(a:3,b:3,c:3,d:3)
end.
    
```

## Глобальные переменные



Ответ

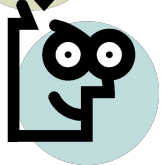
```

3 3 0 3
3 0 0 3
    
```

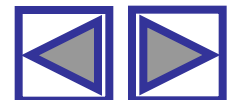
# Механизм передачи параметров в функции и процедуры

```
NONAME00.PA
program c1;
var
  a:integer;b,c:integer;
procedure Proc(var a:integer; b,c:integer);
begin
  a:=7; b:=8; c:=9;
  writeln(a:3,b:3,c:3)
end;
begin
  a:=1; b:=2; c:=3;
  writeln(a:3,b:3,c:3);
  proc(a,b,4);
  writeln(a:3,b:3,c:3)
end.
```

Ответ



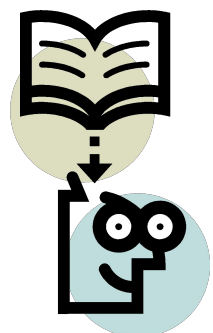
Определите результат выполнения предложенной программы.



# Механизм передачи параметров в функции и процедуры

```
[ ] PROC5.  
program proc5;  
var  
  a:integer;b,n:integer;  
procedure aa(var a:integer; b:integer);  
  var n:integer;  
  begin  
    a:=b+3; b:=3*a; n:=a+b; {1}  
    writeln(n)  
  end;  
begin  
  a:=0;_n:=5;  
  aa(n,n); {2}  
  writeln(n)  
end.
```

Ответ



Определите результат выполнения предложенной программы.





# Механизм передачи параметров в функции и процедуры

```
[ ]
program proc6;
var
  b,n:integer;
procedure bb(x,y:integer);
  var n:integer;
  begin
    x:=y+3; y:=3*x; n:=x+y; {1}
    writeln(n)
  end;
begin
  b:=0; n:=12;
  bb(n,n); {2}
  writeln(n)
end.
```

Ответ



Определите результат выполнения предложенной программы.



# Рекурсия

Примеры задач

# Понятие рекурсии

Подпрограммы в Turbo Pascal могут обращаться к самим себе. Такое обращение называется рекурсией. Объект, который частично определяется через самого себя, называется рекурсивным. Рекурсивные определения как мощный аналитический аппарат используются во многих областях науки, особенно в математике. Для того, чтобы не было бесконечного обращения подпрограммы к самой себе, требуется наличие некоторого условия (условного оператора) в тексте программы, по достижении которого дальнейшее обращение не происходит. Таким образом, рекурсивное программирование может включаться только в одну из ветвей условного оператора, присутствующего в подпрограмме.

**Подпрограмма <имя>(<список формальных параметров>):**

Описательная часть

Begin

...

if <условие> then < обращение к подпрограмме <имя>>

else <операторы>;

...

End;

if <условие> then <операторы>

else < обращение к подпрограмме <имя>>;





# Вычисление факториала натурального числа

```
program fn2;  
Uses crt;  
Var n:integer; a:longint;;  
function factorial(n:integer) :longint;  
  begin  
    if n=1 then factorial:=1  
    else factorial:=n*factorial(n-1);  
  end;  
  
begin  
clrscr;  
writeln('введите значение n:'); readln(n);  
  a:=factorial(n); writeln('значение факториала ',n,'!=',a:8);  
readln; end.
```

