

# Использование подпрограмм в Паскале

1. Структура программы на языке Паскаль
2. Описание и вызов процедур
3. Описание функции
4. Формальные и фактические параметры
5. Область действия имен
6. Рекурсивные процедуры и функции
7. Предварительно-определенные процедуры
8. Модули

# Структура программы

**PROGRAM** <имя программы> [ (<список файлов>) ] ;

**LABEL** <описание меток>;

**CONST** <описание констант>;

**TYPE** <описание типов>;

**VAR** <описание переменных>;

**PROCEDURE** <описание процедуры> ;

**FUNCTION** <описание функции>;

**BEGIN**

    <исполнительная часть программы>

**END .**

# Пример программы

```
begin  
    writeln('Лекция по информатике')  
end.
```

```
Borland Pascal Version 7.0 Copyright  
(c) 1983,92 Borland International
```

```
Лекция по информатике
```

# Директива include

```
{ $I <имя файла> }
```

```
PROGRAM A1;  
VAR ...  
{ $I B1.PAS }  
BEGIN  
    ...  
END.
```

Файл B1.PAS может  
иметь вид:

```
PROCEDURE PP;  
VAR ...  
BEGIN  
    ...  
END;
```

# Процедура

***Процедура*** - это поименованное сложное действие, которое представляет собой совокупность операторов, вычисляющих некоторое число результатов в зависимости от некоторого числа аргументов.

# Описание и вызов процедур

**Заголовок процедуры:**

**PROCEDURE** < имя процедуры >

[( <список формальных параметров >)];

**Вызов процедуры:**

<имя процедуры > [( <список фактических параметров >)];

# Пример заголовка и вызова

```
Type t=array[1..50] of integer;  
Procedure sort(n:integer; a:t; var b:t);  
Var ...  
Begin ... { алгоритм решения задачи }  
End;  
  
Var a,b:t; n:integer;  
Begin  
... Sort(n,a,b); ...  
End.
```

# Пример процедуры

```
VAR X, Y, E: REAL;
```

```
PROCEDURE SINX (X, E: REAL; VAR Y: REAL) ;
```

```
VAR U, Z: REAL;
```

```
    K: INTEGER;
```

```
BEGIN
```

```
    K:=0;
```

```
    Y:=0;
```

```
    U:=X;
```



# Продолжение

```
Z := SQR (X) ;
```

```
WHILE ABS (U) > E DO
```

```
BEGIN
```

```
    Y := Y + U ;
```

```
    K := K + 2 ;
```

```
    U := -U * Z / (K * (K + 1)) ;
```

```
END ;
```

```
END ;
```

# Продолжение

```
BEGIN
```

```
    READLN (X, E) ;
```

```
    SINX (X, E, Y) ;
```

```
    WRITELN (' SIN=' , SIN (X) , ' Y=' , Y) ;
```

```
END .
```

# Пример процедуры

Вводится целое число. Вывести сообщение - число простое или составное.

```
var x:integer;  
    f:boolean;  
procedure prost(x:integer;  
                var f:boolean);  
var d:integer;  
begin  
    f:=x>1;  
    for d:=2 to x div 2 do  
        if x mod d =0 then  
            f:=false;  
end;
```

# Пример процедуры

```
begin
  write('Введите число x= ');
  readln(x);
  prost(x, f);
  if f then
    writeln('Число ', x, ' простое')
  else
    writeln('Число ', x, ' составное')
end.
```

# Отличия функции от

## процедуры

- \* **результатом** обращения к функции может быть **одно** единственное значение;
- \* идентификатор результата **не** указывается в списке формальных параметров;
- \* в выполняемой части функции, хотя бы один раз, **имени функции** должно быть присвоено **значение результата** (чаще всего перед выходом из функции);
- \* после списка формальных параметров задается **тип результата**;
- \* после обращения к функции управление передается на выполнение следующей **операции** данного выражения (в соответствии с приоритетом).

# Описание функции

**FUNCTION** < имя функции >

[(<список формальных параметров>)] :

<тип результата>;

**FUNCTION PRF (A,B,C: INTEGER) : REAL;**

...

# Примеры вызова функций

```
Var S:real;
```

```
. . .
```

```
S:=PRF (A,B,C);
```

```
Writeln ( PRF ( A,B,C) );
```

```
If PRF ( A,B,C) > 20 then K=K+1;
```

```
. . .
```

# Пример функции

```
VAR X, E, Y: REAL;  
FUNCTION SINX (X, E: REAL) : REAL;  
VAR U, Z, Y: REAL;  
    K: INTEGER;  
BEGIN  
    K := 0;  
    Y := 0;  
    U := X;
```



# Продолжение

```
Z := SQR (X) ;
```

```
WHILE ABS (U) > E DO
```

```
BEGIN
```

```
    Y := Y + U ;
```

```
    K := K + 2 ;
```

```
    U := -U * Z / (K * (K + 1)) ;
```

```
END ;
```

```
SINX := Y
```

```
END ;
```

# Продолжение

```
BEGIN
```

```
    READLN (X, E) ;
```

```
    Y := SINX (X, E) ;
```

```
    WRITELN ( ' SIN=' , SIN (X) , ' Y=' , Y) ;
```

```
END .
```

**ИЛИ**

```
WRITELN ( ' SIN=' , SIN (X) , ' Y=' , SINX (X, E) ) ;
```

# Пример функции

{проверка простого числа}

```
var x:integer;
```

```
function prost(x:integer):boolean;
```

```
var d:integer;
```

```
begin
```

```
    prost:=x>1;
```

```
    for d:=2 to x div 2 do
```

```
        if x mod d =0 then
```

```
            prost:=false;
```

```
end;
```

# Пример функции

```
begin
  write('Введите число x= ');
  readln(x);
  if prost(x) then
    writeln('Число ', x, ' простое')
  else
    writeln('Число ', x, ' составное')
end.
```

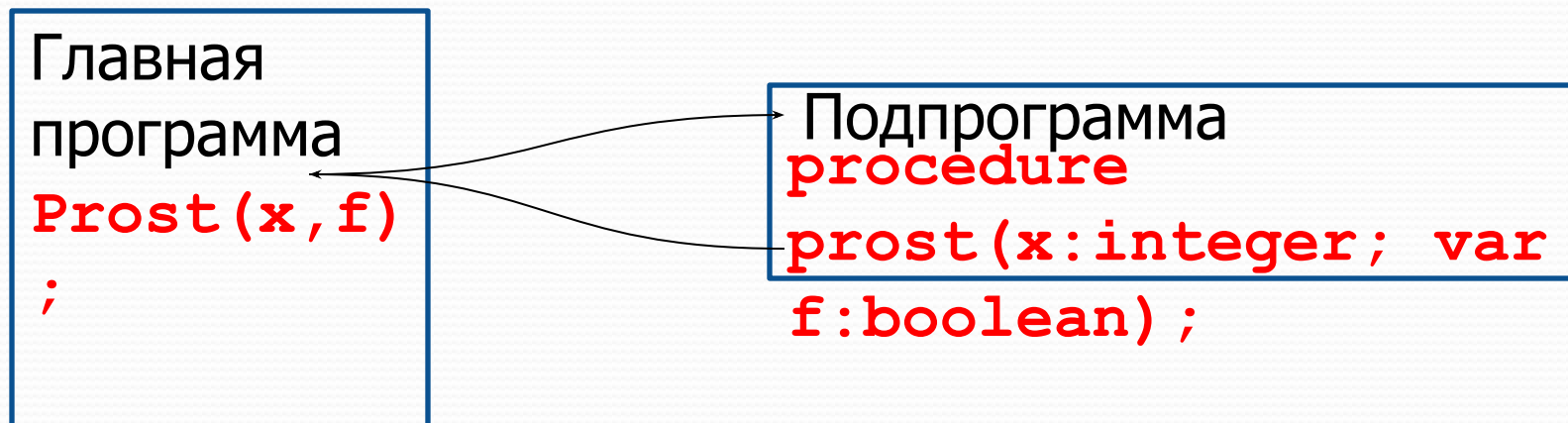
# Типы параметров

**При описании процедуры (функции) в ее заголовке могут быть указаны параметры следующих видов:**

- параметры-значения;**
- параметры-переменные;**
- параметры-константы;**
- параметры-процедуры;**
- параметры-функции.**

# Параметры подпрограммы

Список параметров, задаваемых в заголовке процедуры или функции, обеспечивает **связь подпрограммы с вызывающей программой**. Через него в подпрограмму передаются исходные данные и возвращается результат (в процедуре).



# Правила записи параметров

- **число** формальных и фактических параметров должно быть **одинаково**;
- **порядок** следования и **тип** фактических параметров должен совпадать с порядком и типом соответствующих формальных параметров;
- идентификаторы формальных и фактических параметров могут совпадать;
- формальные параметры в языке Турбо Паскаль в заголовке находятся вместе **с описаниями** и объявлять их в разделе описаний процедуры(функции) не требуется;
- формальные параметры должны иметь **простые** или **ранее определенные типы**.

# Параметры массивы

```
TYPE TV=ARRAY [1..30] OF INTEGER;  
      TM=ARRAY [1..20,1..20] OF REAL;
```

...

```
PROCEDURE TOP ( A:TM; VAR B: TV ;  
                N: INTEGER);
```

...



# Параметры-значения

...

```
BEGIN
```

```
    READLN (X) ;
```

```
    WRITELN ( ' SIN=' , SIN (X) , ' Y=' ,
```

```
                SINX (X, 1/10000) ) ;
```

```
END .
```

# Параметры-переменные

```
FUNCTION SUM (VAR A: ARRAY OF INTEGER)
                : INTEGER;
VAR S, I : INTEGER;
BEGIN
    S := 0;
    FOR I := 0 TO HIGH (A) DO
        S := S + A [I];
    SUM := S;
END;
```

# Пример программы

```
USES CRT;  
TYPE TMAS=ARRAY[1..100,1..100] OF WORD;  
      TVECT=ARRAY[1..100] OF WORD;  
VAR A:TMAS;  
    V:TVECT;  
    N,M,K:BYTE;  
    I,J:BYTE;
```

# Продолжение

```
PROCEDURE FORM (VAR X : TMAS ;  
                N , M : BYTE ;  
                VAR R : TVECT ;  
                VAR K : BYTE) ;  
  
VAR I , J , Z , S : BYTE ;  
    F : BOOLEAN ;  
  
FUNCTION PROS (B : WORD) : BOOLEAN ;  
VAR I : WORD ;
```

# Продолжение

```
BEGIN
```

```
    PROS := B > 1 ;
```

```
    FOR I := 2 TO B DIV 2 DO
```

```
        IF B MOD I = 0 THEN
```

```
            PROS := FALSE ;
```

```
END ;
```

```
BEGIN
```

```
    K := 0 ;
```

# Продолжение

```
FOR J:=1 TO M DO
```

```
BEGIN
```

```
  Z:=0; S:=0; F:=TRUE;
```

```
  FOR I:=1 TO N-1 DO
```

```
  BEGIN
```

```
    IF X[I,J]>X[I+1,J] THEN Z:=Z+1;
```

```
    IF X[I,J]<X[I+1,J] THEN S:=S+1
```

```
  END;
```

```
  IF (Z = N-1) OR (S = N-1) THEN
```

# Продолжение

```
BEGIN
```

```
    FOR I:=1 TO N DO
```

```
        IF NOT (PROS (X[I,J])) THEN F:=FALSE;
```

```
        IF F THEN
```

```
            BEGIN
```

```
                K:=K+1; R[K]:=J
```

```
            END;
```

```
        END;
```

```
    END;
```

```
END;
```

# Продолжение

```
BEGIN
```

```
  WRITELN ( ' Введите N и M: ' ) ;
```

```
  READLN ( N , M ) ;
```

```
  WRITELN ( ' Введите матрицу: ' ) ;
```

```
  FOR I:=1 TO N DO
```

```
    FOR J:=1 TO M DO
```

```
      READLN ( A [ I , J ] ) ;
```

```
  FORM ( A , N , M , V , K ) ;
```

```
  WRITELN ( ' Результат: ' ) ;
```

```
  FOR I:=1 TO K DO WRITE ( V [ I ] , '   ' ) ;
```

```
  READKEY
```

```
END .
```



# Область действия имен

B, T, D A A' X X' C Y K, L

**Program PR;** **БЛОК 1**

**Type T = array[1..50] of integer;**

**Var A, B, X: real;**

**D: T;**

**Procedure PR1;** **БЛОК 2**

**Var K, L : integer;**

**Begin**

**...**

**End;**

**Procedure PR2;** **БЛОК 3**

**Var A, C : integer;**

**Procedure PR3;** **БЛОК 4**

**Var X, Y : byte;**

**Begin**

**...**

**End;**

**Begin**

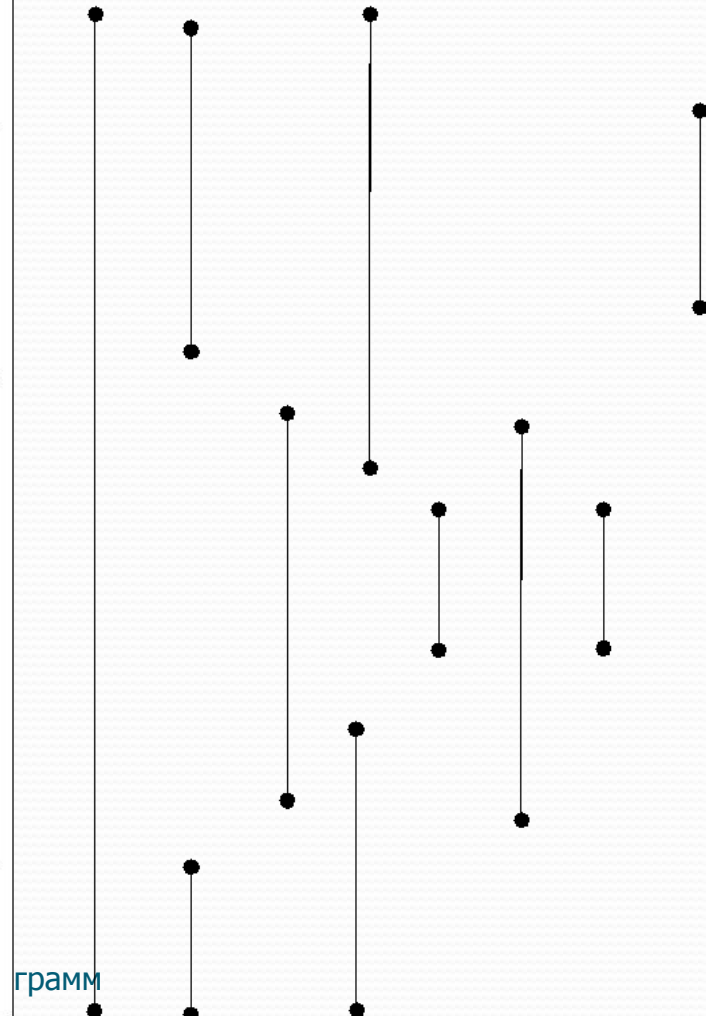
**...**

**End;**

**Begin**

**...**

**End;**



# Рекурсия

*Рекурсия* – это способ организации вычислительного процесса, при котором процедура или функция в процессе выполнения входящих в ее состав операторов обращается сама к себе непосредственно, либо через другие процедуры и функции.

Пример:  $F=M!$

При  $M=1$   $F=1$

При  $M>1$   $F=M!=M*(M-1)!$

# Рекурсивная функция

```
PROGRAM MAIN;  
VAR M: INTEGER;  
    F: REAL;  
FUNCTION FACT (N: INTEGER) : REAL;  
BEGIN  
    IF N=1 THEN FACT:=1 ELSE  
        FACT:= N* FACT (N-1) ;  
END ;  
BEGIN  
    READLN (M) ;  
    F:= FACT ( M ) ;  
    WRITELN ( ' M!=', F ) ;  
END .
```

# Рекурсивная процедура

```
VAR M: INTEGER;  
    F: REAL;  
PROCEDURE FACT (N: INTEGER; VAR F: REAL) ;  
VAR Q : REAL;  
BEGIN  
    IF N=1 THEN  
        Q:=1  
    ELSE  
        FACT (N-1 , Q) ;  
        F:=N*Q;  
    END ;  
END ;
```

# Продолжение

```
BEGIN  
    READLN (M) ;  
    FACT ( M, F ) ;  
    WRITELN ( ' M!=', F ) ;  
END.
```

# Forward

```
PROGRAM KOSV_R;  
VAR X,Y: INTEGER;  
PROCEDURE AA (A: INTEGER) ; FORWARD ;  
PROCEDURE BB (B: INTEGER) ;  
    ... BEGIN ...  
        AA (...);  
        ...  
    END ;
```

# Продолжение

```
PROCEDURE AA ;
```

```
... BEGIN ...
```

```
    BB (... ) ;
```

```
    ...
```

```
    END ;
```

```
BEGIN
```

```
... BB (X) ; AA (Y) ; ...
```

```
END .
```

# Модуль

```
UNIT <имя модуля>;
```

```
INTERFACE
```

```
IMPLEMENTATION
```

```
[ BEGIN
```

```
    <Иницилирующая часть модуля> ]
```

```
END .
```



# Пример интерфейсной части

```
UNIT MOD1;
```

```
INTERFACE
```

```
TYPE
```

```
    TMAS= array [1..50,1..50] of real;
```

```
PROCEDURE PR1 (var A:TMAS; M,N: byte) ;
```

```
...
```

```
Uses MOD1;
```

# Пример модуля

```
USES CRT,MODSORT;
```

```
VAR A:MAS;
```

```
    I:BYTE;
```

```
    N:BYTE;
```

```
BEGIN
```

```
    WRITELN('ВВОД ИСХОДНЫХ ДАННЫХ: ');
```

```
    READLN(N);
```

```
    FOR I:=1 TO N DO
```

```
        READLN(A[I]);
```

# Продолжение

```
SORT (A , N) ;  
FOR I :=1 TO N DO  
    WRITELN (A [ I ] ) ;  
READKEY  
END .
```

# Продолжение

```
UNIT MODSORT;
```

```
INTERFACE
```

```
    TYPE MAS=ARRAY[1..100] OF INTEGER;
```

```
    PROCEDURE SORT (VAR A:MAS; N:BYTE);
```

```
IMPLEMENTATION
```

```
    PROCEDURE SORT;
```

```
    VAR I, J:BYTE;
```

```
        X:INTEGER;
```

# Продолжение

**BEGIN**

**FOR J:=1 TO N-1 DO**

**FOR I:=1 TO N-J DO**

**IF A[I]>A[I+1] THEN**

**BEGIN**

**X:=A[I] ; A[I]:=A[I+1] ; A[I+1]:=X**

**END ;**

**END ;**

**END .**