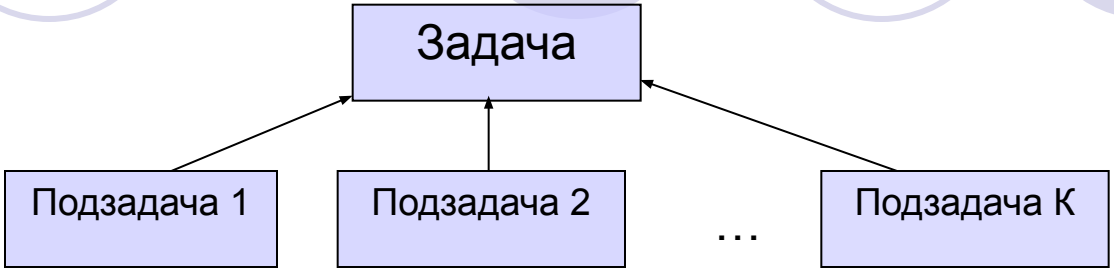
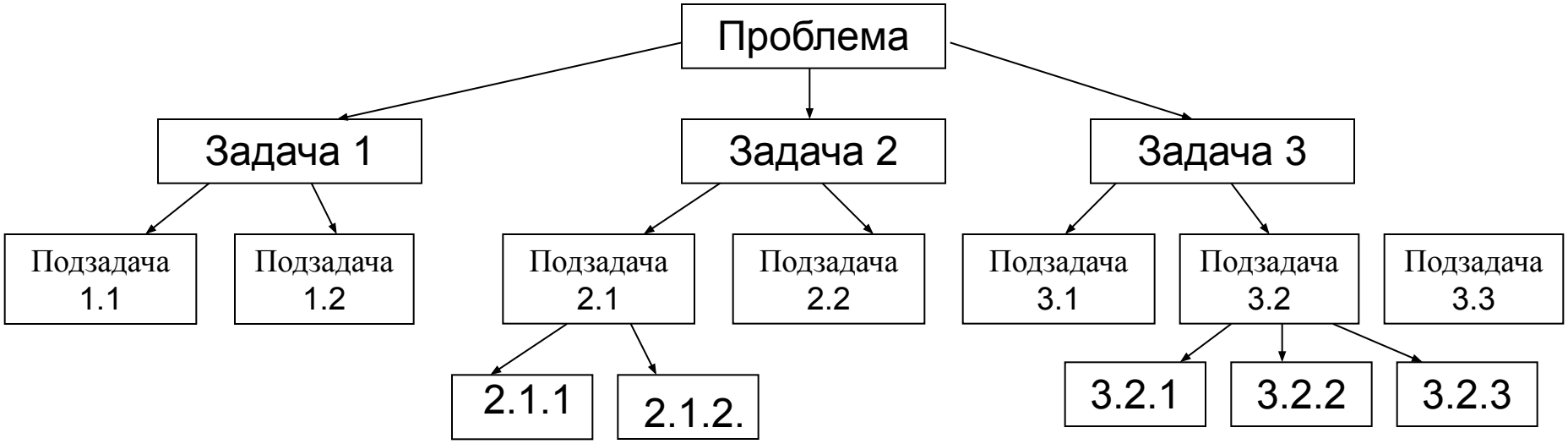
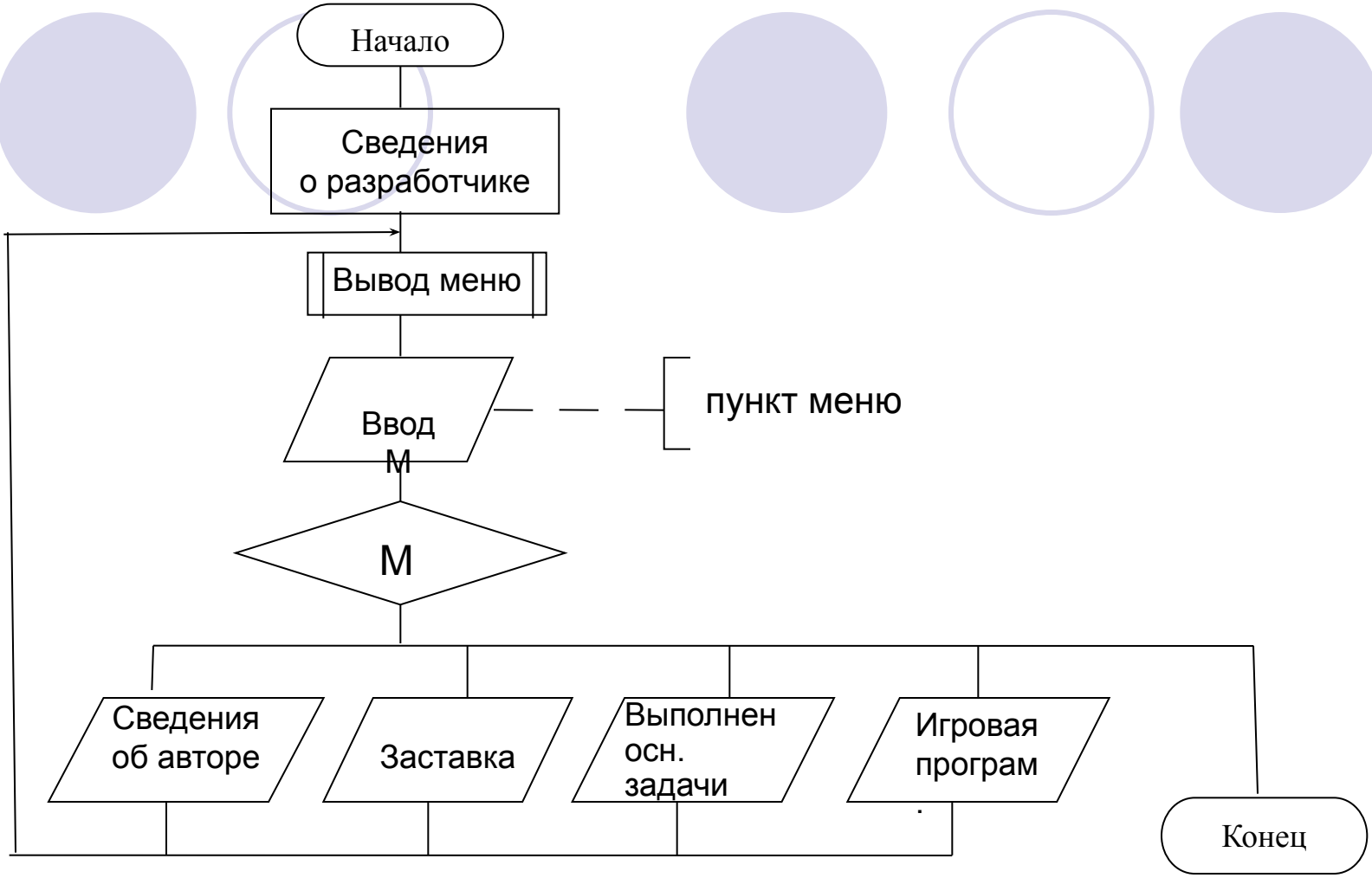
Структуризация

Восходящее проектирование алгоритмов и программ



Нисходящее программирование («сверху - вниз)





Укрупненная схема алгоритма для курсовой работы

Подпрограммы

процедуры и функции

ЗАГОЛОВОК;
Директива;
LABEL
{Разделы} **TYPE**
{описаний} **CONST**
VAR
{Описания процедур и функций}
BEGIN
{Раздел операторов}
END;

Заголовки:

PROCEDURE <**имя**>(список формальных параметров);
FUNCTION <**имя**>(список формальных параметров):<тип>;

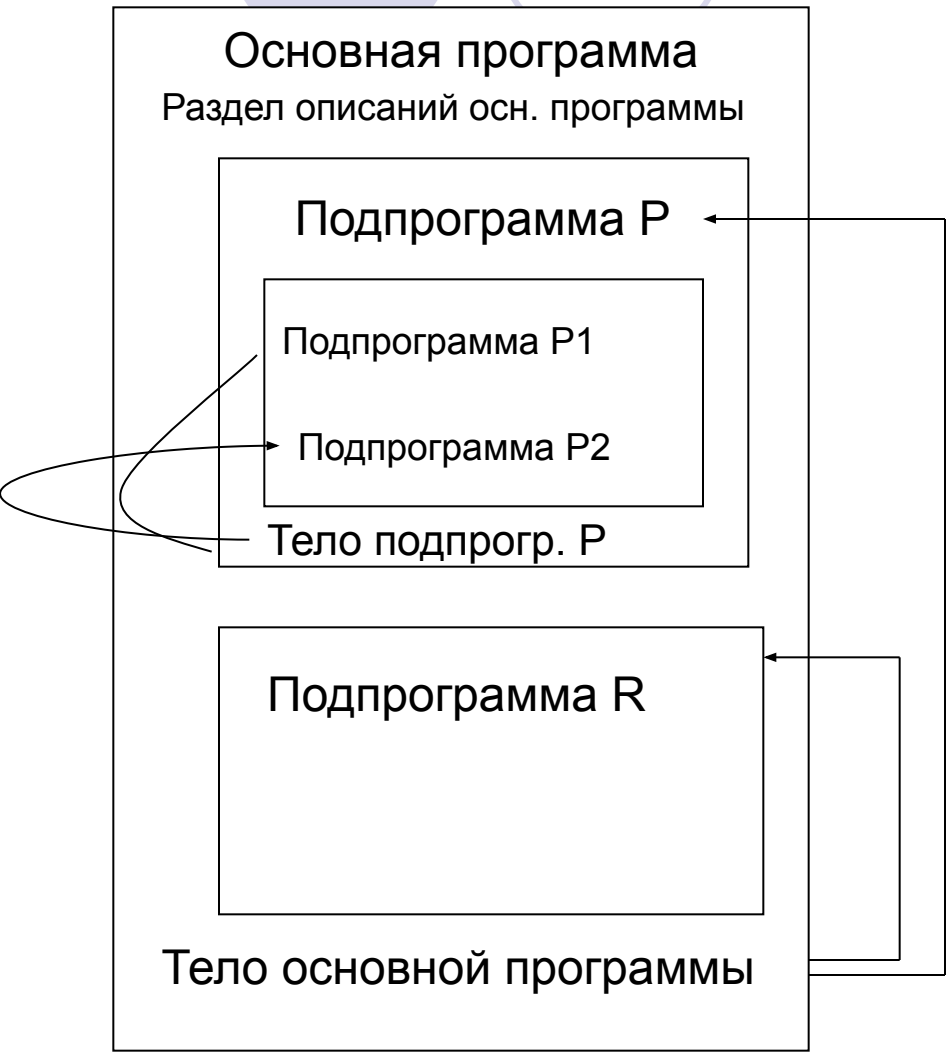
Формальные параметры - параметры, описанные в заголовке и используемые при записи операторов внутри подпрограммы, при выполнении программы они заменяются фактическими переменными или значениями.

Примеры заголовков: PROCEDURE AB (a:real; b:integer; VAR c:char)

FUNCTION FF(x1,x2:real):real;

ASSEMBLER;
EXTERNAL;
FAR;
FORWARD;
INLINE;
INTERRUPT;

Вложенные подпрограммы.
Принцип локализации имен переменных



(P1 и P2 вложены в P).

```
Program OSN;
VAR A,B,C: real; {глобальные перем}
 Procedure P(var v1,v2:byte);
 Var  s, d: byte; {локальные для P}
  Procedure P1(var h1,h2:real);
  Var  m: byte; {локальная дляP1}
  Begin
    {тело подпрограммы P1}
    {действуют A,B,C,h1,h2,s, d,m}
  End;
  Procedure P2( f : real);
  Var  x: real; {локальная дляP2}
  Begin
        {тело подпрограммы P2}
       {действуют A,B,C,f,x,s, d}
  End;
Begin
   {тело подпрограммы P}
   {действуют A,B,C,v1,v2,s, d}
End;
Procedure R( y1,y2,y3:integer);
```

Константы, переменные, типы, описанные в разделе описаний основной программы называются *глобальными*.

В основной программе можно использовать только переменные A,B,C.

Внутри подпрограммы – локальные. Область действия

```
Procedure R( y1,y2,y3:integer);
Var  z: integer; {локальная для R}
       b:byte; ; {локальная для R  закрывает B глобальную}
Begin
   {тело подпрограммы R}
   {действуют A,C-глобальные,y1,y2,y3-форм,z, b -локальные}
End;

BEGIN

      {тело основной программы}
      {действуют A, B,C}
      {вызываются п/п P и R}

END.
```

Вызов подпрограмм

<имя> (список фактических параметров через запятую);
<имя>;

Количество, тип и порядок перечисления ***фактических*** параметров должен обязательно соответствовать количеству, типу, порядку перечисления формальных параметров в описании этой подпрограммы.

Вложенные п/п могут вызываться только внутри охватывающей п/п.

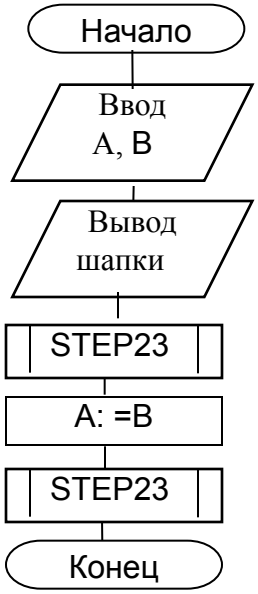
Процедуры

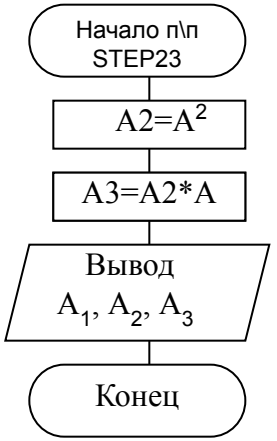
```
PROCEDURE <имя>(список формальных параметров);      {Заголовок}
LABEL                  {раздел описаний (метки,          }
TYPE                   {          типы,                  }
CONST            {   константы,                }
VAR         {    переменные )                 }
{описания вложенных процедур и функций}
BEGIN
   {операторы подпрограммы}
END;
```

Процедуры без параметров

Пример: по заданным A и B вычислить их квадраты и кубы

```
PROGRAM PROC;
VAR A,B:INTEGER;    {глобальные}
PROCEDURE STEP23;
  BEGIN
     A2:=A*A; A3:=A2*A;
     WRITELN(A:5; A2:5; A3:6);
 END;
BEGIN
      READ(A,B);
      WRITELN('число квадрат куб');
      STEP23;
      A:=B;
      STEP23;
END.
```





Обмен данными между основным блоком и подпрограммой производится через глобальные переменные.

Процедуры с параметрами

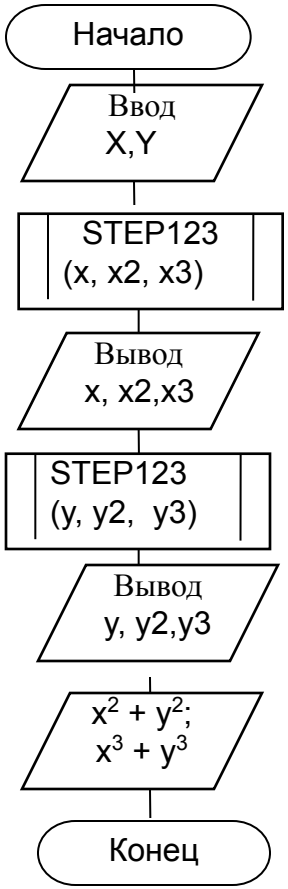
Любой из формальных параметров процедуры может быть либо **параметром - значением**, либо параметром переменной.

Пример:

```
PROCEDURE P(A:REAL; VAR A2,A3:REAL);
{A - параметр-значение, A2,A3 -  параметры-переменные}
```

Пример: Вычислить вторую и третью степени для заданных чисел X и У и их суммы

```
PROGRAM PROC;
VAR x,y,x2,y2,x3,y3:word;   {глобальные}
PROCEDURE STEP123(A:word;Var A2,A3:word);
            BEGIN
              A2:=A*A; A3:=A2*A;
            END;
BEGIN
      READ(x,y);
      WRITELN('число квадрат куб');
      STEP123(x,x2,x3); WRITELN('x=',x:3; x2:5; x3:6);
      STEP123(y,y2,y3); WRITELN('y=',y:3; y2:5; y3:6);
      WRITELN('сумма квадратов=', x2+y2);
      WRITELN('сумма кубов=', x3+y3);
END.
```



Функции

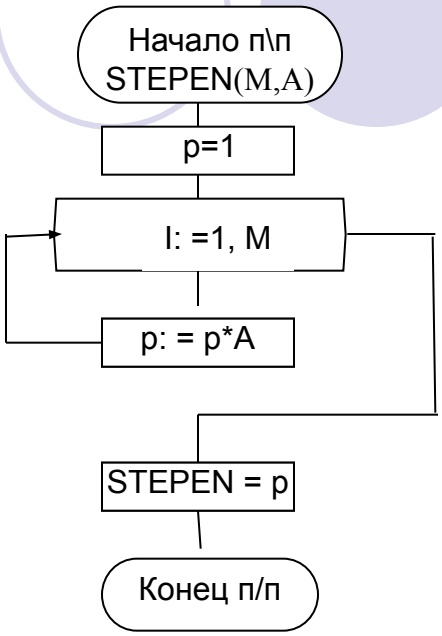
FUNCTION <имя>:<тип>; {без параметров}

FUNCTION <имя>(формальные параметры):<тип>; {с параметрами}
LABEL {раздел описаний меток}
CONST {константы}
TYPE { типы}
VAR { переменные}
{описание вложенных процедур и функций}
BEGIN
{операторы функции}
<имя>: =<выражение> {обязательный оператор}
END;

Функция передает в вызывающую программу единственное значение, которое хранится под ее именем.

Пример: $y=x^2+3^N$

Вычислим произвольную целую степень с помощью функции




```
PROGRAM FUNC;
VAR Y,X :REAL; N:INTEGER;

FUNCTION STEPEN (M:INTEGER; A:REAL): REAL;
VAR P: REAL; I:INTEGER;
  BEGIN
    P:=1;
    FOR I:=1 TO M DO P:=P*A;
    STEPEN:=P;
  END;

BEGIN
  READ (X,N);
  Y:= stepen(2,x)+stepen(N,3);
  WRITELN ('y=', y);
END.
```

Пример 2. Подсчитать общее количество цифр в трех числах X,Y,Z.

```
Program FUN_kol2;  {Возврат дополнительного значения k из функции}
VARX,Y,Z:LongInt;
        xk, yk, zk:byte;    - количество цифр для каждого

FUNCTION KOL(A:LongInt; Var k:byte):byte;
Begin
    k:=0;
    Repeat
        A:=A div 10;
        k:=K+1
    Until A=0;
  KOL:=k
 end;

BEGIN
   Writeln(' Введи три целых числа '); Read(x,y,z);
   Writeln('всего в 3 числах ', KOL(x,xk)+KOL(y,yk)+KOL(z,zk),' - цифр ');
   Writeln('В  числе X ', xk,' - цифр ');
   Writeln('В  числе Y ', yk,' - цифр ');
   Writeln('В числе Z ', zk,' - цифр ');
   readln
END.
```

Передача в подпрограмму параметров-массивов и параметров-строк

Пример: TYPE **ATYPE= ARRAY[1..10] OF REAL;**
PROCEDURE R(A:ATYPE);

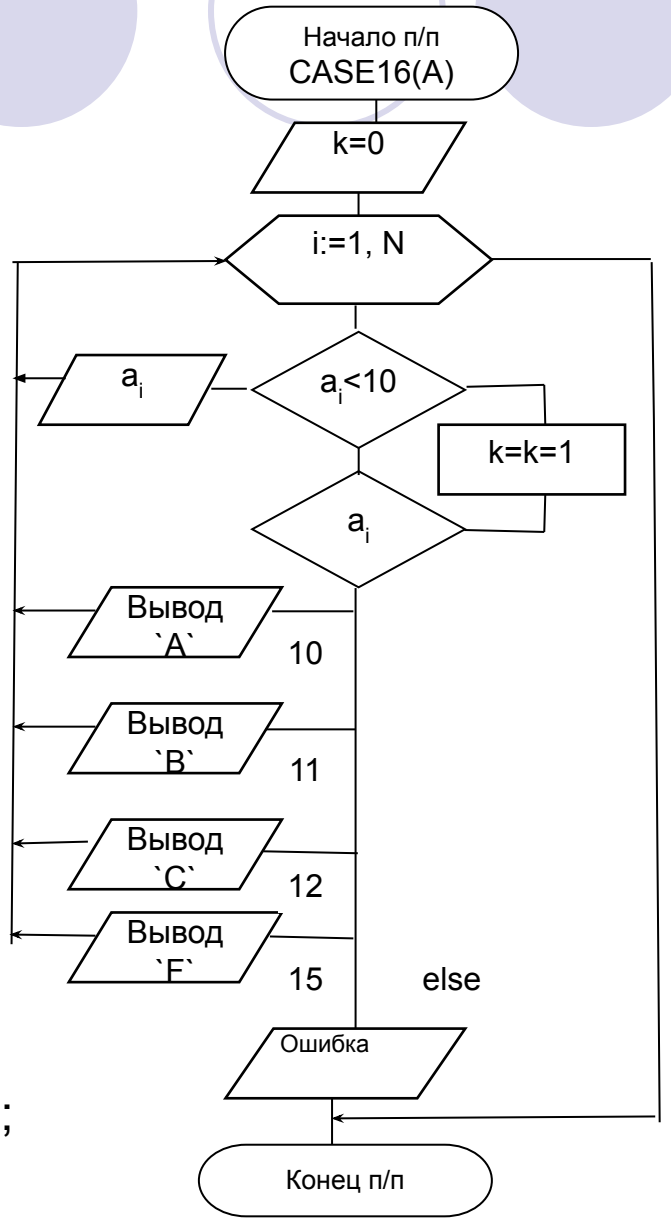
Пример:

```
TYPE  INTYPE=STRING[15];
      ONTYPE=STRING[30];

FUNCTION R(A:INTYPE): ONTYPE; {строка как один элемент}
```

Пример 2. В трех одномерных массивах A,B,C целых чисел от 0 до 16 заменить числа шестнадцатеричными цифрами. Подсчитать в каждом массиве сколько цифр обозначены буквами.

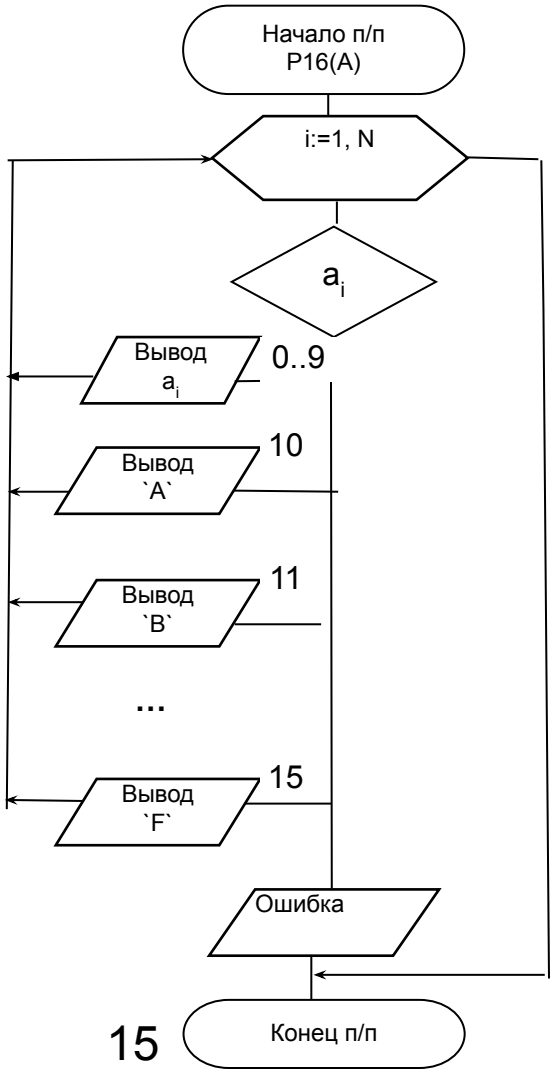
```
Program FUN16;
Const N=10;  число элементов в массиве
TYPE       mass=array[1..N] of 0..16;
VAR A,B,C:mass;
            i:byte;
FUNCTION CASE16(A:mass):byte;
VAR k:byte;
BEGIN
       k:=0;
       For i:=1 to N do
             if a[i]<10 then Write(a[i]:2,' ')
                    else
                        Begin
                              k:=k+1;
                              CASE a[i] of
                                10:   Write(‘ A   ');
                                11:   Write(‘ B   ');
                                12:   Write(‘ C   ');
                                13:   Write(‘ D   ');
                                14:   Write(‘ E   ');
                                15: Write(‘ F   ');
                                      else Writeln('Ошибка ');
                                     end;
                 end;
```



```
    Writeln;
    CASE16:=k имени функции присвоим значение
 end;
BEGIN
 randomize;
      For i:=1 to N do a[i]:=random(15); {можно ввод и печать тоже}
      For i:=1 to N do b[i]:=random(15); {оформить подпрограммой}
      For i:=1 to N do c[i]:=random(15);
      Writeln('В массивах ');
      Write('A - ');
      For i:=1 to N do Write(a[i]:2,' ');
           Writeln;
           Writeln('   ',case16(A),' - букв ');
      Write('B - ');
      For i:=1 to N do Write(b[i]:2,' ');
           Writeln;
           Writeln('   ',case16(B),' - букв ');
      Write('C - ');
      For i:=1 to N do Write(c[i]:2,' ');
           Writeln;
           Writeln('   ',case16(C),' - букв ');
  readln
END.
```

Пример 3. В трех одномерных массивах A,B,C целых чисел от 0 до 15 заменить числа шестнадцатеричными цифрами.

```
Program proc16ma;
Const N=10;  {число элементов в массиве}
TYPE     mass=array[1..N] of 0..16;
VAR      A,B,C:mass;
         i:byte;
PROCEDURE P16(A:mass);
VAR k:byte;
BEGIN
   k:=0;
Write('   в 16 СС - ');
   For i:=1 to N do
            CASE a[i] of
              0..9: Write(a[i]:2,' ');
                10: Write(' A ');
                11: Write(' B ');
                12: Write(' C ');
                13: Write(' D ');
                14: Write(' E ');
                15: Write(' F ');
                     else Writeln('Ошибка ');
             end;
   Writeln;
end;
```



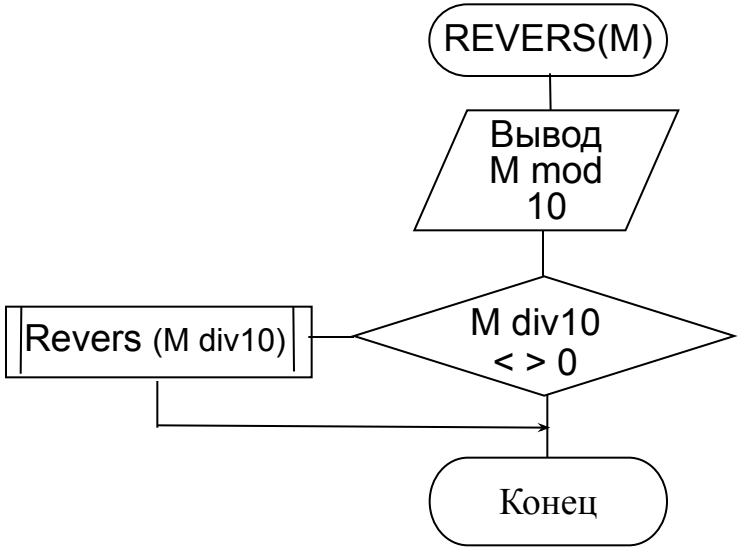
```
BEGIN
   randomize;
   For i:=1 to N do a[i]:=random(15); {можно ввод и печать тоже}
   For i:=1 to N do b[i]:=random(15); {оформить подпрограммой}
   For i:=1 to N do c[i]:=random(15);
   Writeln('Массивы ');
   Write('A исходный - ');
   For i:=1 to N do Write(a[i]:2,' ');
      Writeln;
      P16(A);
   Write('B исходный - ');
   For i:=1 to N do Write(b[i]:2,' ');
      Writeln;
      P16(B);
   Write('C исходный - ');
   For i:=1 to N do Write(c[i]:2,' ');
      Writeln;
      P16(C);
  readln
END.
```

Рекурсия

Пример: Пусть задано целое положительное число, выдать цифры в обратном порядке

```
PROGRAM REKURS;
VAR N:INTEGER;
PROCEDURE REVERS(M:INTEGER);
BEGIN
    WRITE(M MOD 10);
    IF (M DIV 10)<>0 THEN revers (M DIV 10);
END;

BEGIN
     WRITE('Введи целое число без знака');
    READ (N);
    REVERS (N);
END.
```



Условие полного окончания работы рекурсивной процедура должно находиться в самой процедуре .

Рекурсия может быть прямой или косвенной. (A→ B → A).

Опережающее описание FORWARD (вперед).

```
PROCEDURE B(I:BYTE);  FORWARD;

  PROCEDURE A(J:BYTE);
  BEGIN
  ...
  B(J);
  END;
  PROCEDURE B; {параметры опускаются}
  BEGIN
  ...
  A(I);
  END.
```

При повторном описании тело процедуры B начинается заголовком, в котором уже не указываются описанные ранее формальные параметры.