Массивы в Паскале.

Одномерные массивы

Массив *– это именованная группа однотипных данных, хранящихся в последовательных ячейках памяти*.

Каждая ячейка содержит элемент массива. Элементы нумеруются по порядку.

Порядковый номер элемента массива называется **индексом** этого элемента.

Способы создания массивов

В разделе TYPE:

Type

Имя типа = Array [диапазон] Of тип элементов;

Var

Имя переменной: имя типа;

В разделе Var:

Var Имя переменной: array [тип индекса] Of тип элементов;

Здесь

Array – служебное слово (в переводе с английского означает «массив»);

Of – служебное слово (в переводе с английского означает «из»).

Тип индекса – любой порядковый тип, кроме типов *integer, longint.* Кроме *файлового* типа.

Например:

Type

mas = array [1..20] of real;

Var X: mas;

Например:

Var X: array [1..20] of real;

Массив *X* – одномерный, состоящий из двадцати элементов вещественного типа. Элементы массива хранятся в памяти компьютера последовательно друг за другом.

Ввод и вывод одномерных массивов

Введем одномерный массив *X*, состоящий из 30 элементов.

Т.е. необходимо ввести некую последовательность элементов $X_1, X_2, \ldots, X_{30}$.

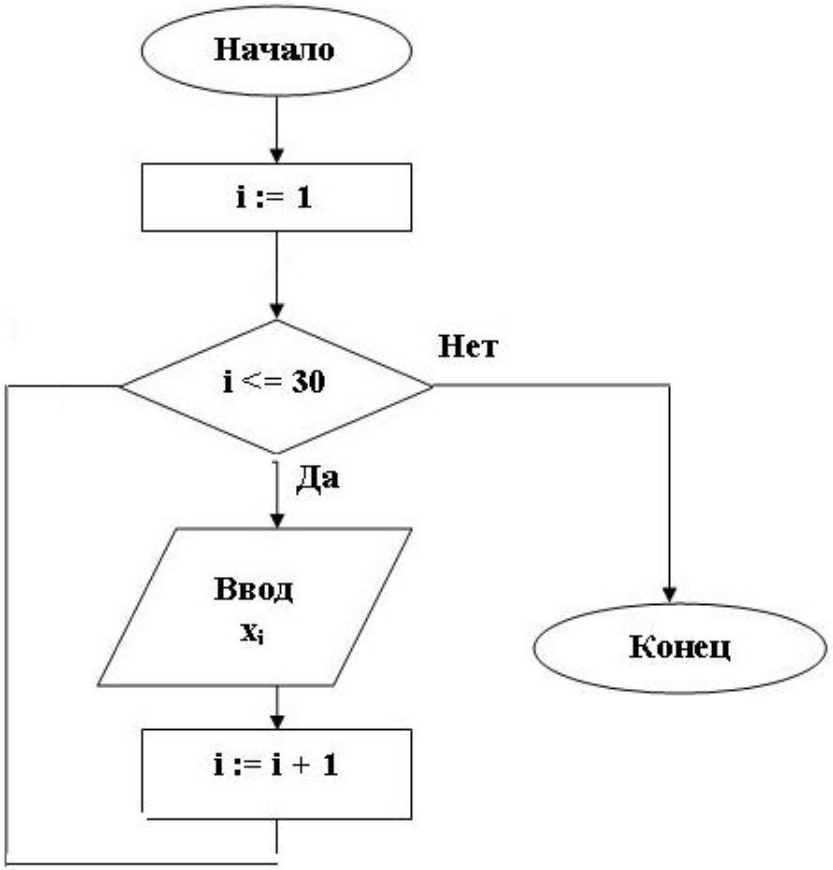
Пусть i – индекс (порядковый номер) элемента в массиве *X*. Тогда

X_i – i-й элемент массива *X*, где $i = 1, 2, \ldots, 30$.

Для ввода массива можно использовать любой цикл.

1 - цикл с предусловием

```
Var i: byte;
    X: array [1..30] of Integer;
Begin
    i := 1;
    While i <= 30 Do
       Begin
         Read (X[i]);
         i := i + 1
        End;
     End.
```

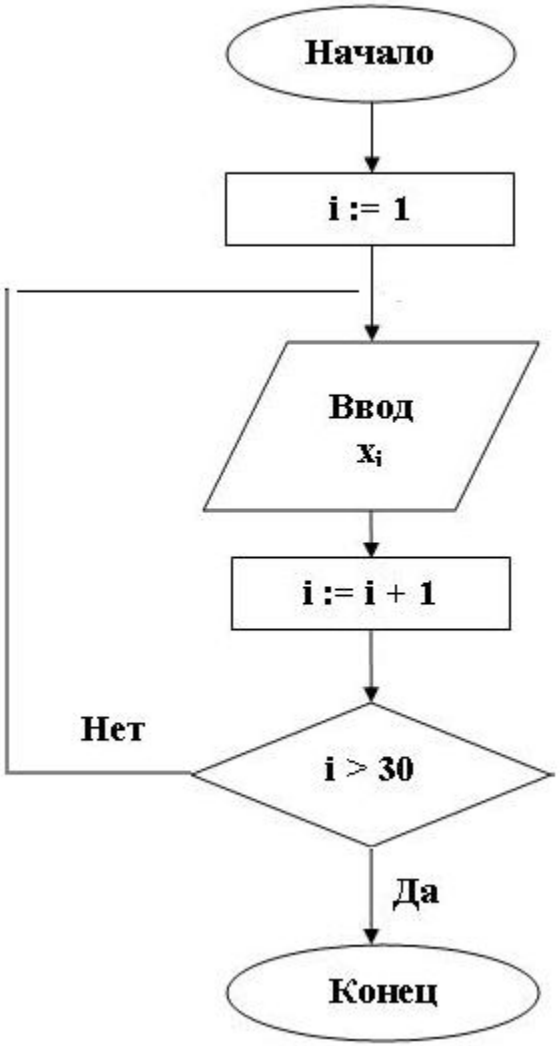


Можно ввести с клавиатуры элементы следующим образом:
-7 _ 4 _ -2 _ 0 _ 12 _ -1 _ -5 _ 9 _ 11 _ -3 _ -5 _ … _15,
то есть через пробел ввести в строчку и нажать клавишу Enter.

Можно было ввести элементы в столбец, отделяя элементы кла
строки:
-7 Enter
4 Enter
-2 Enter
…
…
…
15 Enter

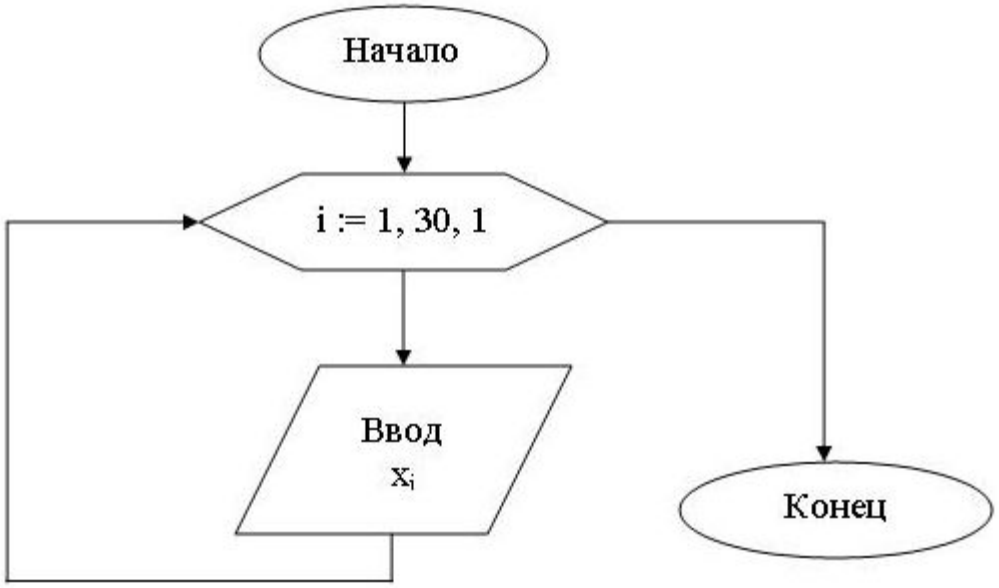
2 - цикл с постусловием

```
Program Primer_2;
Var i: integer;
    X: array [1..30] of Integer;
Begin
    i := 1;
    Repeat
        Read (X[i]);
        i := i + 1
    Until i > 30;
     Readln
End.
```



3 - цикл с параметром

```
Var i: integer;
    X: array [1..30] of Integer;
Begin
    For i := 1 To 30 Do Read (X[i]);
 End.
```



Задача 1. Дан целочисленный одномерный массив, состоящий из n элементов. Найти сумму и произведение нечетных элементов, кратных 3.

Введем обозначения:

n – количество элементов в массиве;

A – имя массива; i – индекс элемента массива;

A_i – i-й элемент массива A;

s – сумма нечетных элементов массива, кратных 3;

p – произведение нечетных элементов массива, кратных 3.

Входные данные: n, A.

Выходные данные: s, p.

```
Var A: Array[1..20] Of Integer;
     i, n, s, p: Integer;
Begin
    Write (‘n=’); Readln (n);
    For i:=1 To n Do Readln (A[i]);   {ввод массива}
    s:= 0;  p:=1;
    For i:=1 To n Do   {обработка массива}
     If (A[i] mod 2 <>0) and (A[i] mod 3 = 0) Then
     begin
        s:=s+A[i];
        p:= p*A[i]
     end;
     Writeln (‘s=’, s, ‘p=’, p);
     Readln
End.
```

Задача 2. Найти номер последнего отрицательного элемента массива.

Введем обозначения: n – количество элементов в массиве;

A – имя массива; i – индекс элемента массива;

A_i – i-й элемент массива A;

m – номер последнего отрицательного элемента массива.

Входные данные: n, A.

Выходные данные: m.

Последний отрицательный элемент массива – это первый отрицательный элемент, который встретится при просмотре массива с конца.

```
Const n=10;
Var A: Array[1..n] Of Integer;
    i, m: Integer;
Begin
    For i:=1 To n Do Readln (A[i]);   {ввод массива}
    m := 0;  i:=n;
    While  (i >= 1) and (A[i] >=0) Do
            i:=i-1;
   m:=i;
   Writeln ('m=', m);
   Readln
   End.
```

Пример:

A=(2 -1 3 -5 -4 0 1 6 -3 7)

i =	1	2	3	4	5	6	7	8	9	10
№ элемента										

Сортировка методом пузырька

```
const m = 10;
var A: array[1..m] of integer;
i, j, k: integer;
begin
randomize;
write ('Исходный массив: ');
for i := 1 to m do begin
                A[i] := random(256);
                write (A[i]:4);
                  end;
writeln;
for i := 1 to m-1 do
for j := 1 to m-i do
if A[j] > A[j+1] then begin
                k := A[j];
                A[j] := A[j+1];
                A[j+1] := k
                end;
write ('Отсортированный массив: ');
for i := 1 to m do
write (A[i]:4);
 end.
```

Двумерные массивы

Их можно описать как таблицу, в ячейках которой располагаются значения. Для обращения к данным массива указывается номера их строк и столбцов. Часто табличные массивы называют матрицами.

1 вариант – описание массива через раздел type:

```
const M = 10;
      N = 5;
type matrix = array [1..M, 1..N] of integer;
var a: matrix;
```

2 вариант – описание массива в разделе переменных:

```
Const M = 10;
      N = 5;
var a: array [1..M, 1..N] of integer;
```

Обращение к элементам

Для обращения к элементу двухмерного массива необходимо указать имя массива и в квадратных скобках через запятую – значения двух индексов (первый указывает номер строки, а второй – номер столбца), на пересечение которых стоит элемент (например, a[i,2]:=6).

допустимо разделение индексов с помощью квадратных скобок (например, a[i][5]:= 7).

Пример: Ввести массив (3x5), а затем вывести элементы на экран в виде таблицы.

```
var
      matrix: array[1..3,1..5] of integer;
      i, j: integer;
begin
      writeln ('Введите 15 чисел: ');
      for i := 1 to 3 do
          for j := 1 to 5 do
               read (matrix[i,j]);

      for i := 1 to 3 do begin
           for j := 1 to 5 do
                write (matrix[i,j], ' ');
           writeln
      end;

readln
end.
```