

# ЛИНЕЙНЫЕ МАССИВЫ

Турбо Паскаль

- Описание типа «массив»
- Действия над элементами массива

# Описание типа «массив»

- *Массив* – это структурированный тип данных, состоящий из фиксированного числа элементов, имеющих один и тот же тип.
- Название *регулярный тип* (или ряды) массивы получили за то, что в них объединены однотипные (логически однородные) элементы, упорядоченные по индексам, определяющим положение каждого элемента в массиве.
- Тип элементов массива называется *базовым*.
- В ЯПВУ Паскаль число элементов массива фиксируется при описании и в процессе выполнения программы не меняется.

# Общий вид описания массива

Описание массива определяет *имя, размер массива и базовый тип*.

1. массив является *переменной* величиной:

**Var** <имя массива>: **array** [<тип индекса>] **of** <базовый тип>;

2. массив является *постоянной* величиной:

**Const n=10** ; {размер массива}

<имя массива>: **array** [<тип индекса>] **of** <базовый тип>=(значения элементов массива);

Чаще всего в качестве типа индекса используется *интервальный* целый тип.

Массивы могут быть *одномерными, двумерными* и *многомерными*.

# Примеры описания массивов:

1) **Var** b: **array** [0..5] **of** real;

r: **array** [1..34] **of** char;

n: **array** ['A'..'Z'] **of** integer;

2) **Type** Klass=(k1,k2,k3,k4);

Znak= **array** [1..255] **of** char;

**Var** Mas1: Znak;

M: **array** [1..4] **of** Klass;

3) **Const** n=5; M: **array** [1..n] **of** byte = (10, 5, 2, 56,198);

тип элементов  
имя массива  
массива

тип элементов  
массива

имя массива

размер массива

значения элементов  
массива

# Примеры описания массивов:

Если в качестве базового типа взят другой массив, образуется *многомерный* массив.

## Примеры:

- **Type** Vector = **array** [1..4] **of** integer;  
  Massiv = **array** [1..4] **of** Vector ;  
**Var** Matr: Massiv ;

Ту же структуру можно получить, используя другую форму записи:

- **Var** Matr: **array** [1..4, 1..4] **of** integer;

В данном случае формируется массив, состоящий из 4 строк и 4 столбцов.

# Линейный (одномерный) массив

- это массив, у которого элементами являются простые переменные. В одномерных массивах хранятся значения линейных таблиц.

Индекс (номер элемента)	1	2	3	4
Элемент массива	123	-568	0	-56

i	1	2	3	4
a[i]	a[1]	a[2]	a[3]	a[4]

**Var a: array [1..4] of integer;**

# Действия над элементами массива

1. Ввод и вывод;
2. Копирование массива;
3. Поиск элементов по некоторому условию;
4. Подсчет количества, суммы и произведения;
5. Нахождение максимума (минимума) и их индексов;
6. Сдвиг, вставка Сдвиг, вставка и удаление;
7. Сортировка.



# I. Ввод и вывод массива

происходит *поэлементно*

## Способы заполнения массива:

- с клавиатуры;
- случайными числами;
- по правилу.





# Заполнение массива с клавиатуры

- С использованием процедур ввода *Read*, *Readln*.

```
program zapln2;  
const n=5; ← Реальный размер массива  
var i:integer;  
a:array[1..100]of integer; ← Максимальный размер массива  
begin  
  
    for i:=1 to n do begin ← Номера элементов массива  
        write('Введите a[' ,i,'] element ');readln(a[i]);  
    end;  
    for i:=1 to n do begin  
        write (a[i]:3)  
    end;  
    writeln;  
    for i:=1 to n do begin  
        write('a[' ,i,'] =');  
        writeln (a[i]:3) ← вывод массива в строку  
    end; ← вывод массива с пояснением  
end.
```

ввод каждого элемента массива с новой строки

# Вид пользовательского экрана после выполнения программы

```
Uvedite a[1] element -4
Uvedite a[2] element 9
Uvedite a[3] element 21
Uvedite a[4] element 0
Uvedite a[5] element 67
-4 9 21 0 67
a[1]= -4
a[2]= 9
a[3]= 21
a[4]= 0
a[5]= 67
```





# Вид пользовательского экрана после выполнения программы

```
vvedite kol-vo elementov -> 15  
9 20 6 73 33 98 73 82 52 32 30 64 13 7 41
```

Заполнение массива случайными числами из промежутка от **A** до **B**. Тип элементов объявляется при описании массива.

$$\text{Mas}[i] := \text{random}(\mathbf{B-A}) + \mathbf{A}$$

**Примеры:**

- $\text{random}(\mathbf{60+1}) - \mathbf{20}$  случайные числа от **-20** до **40**
- $\text{random}(\mathbf{10+1}) + \mathbf{70}$  случайные числа от **70** до **80**



# Заполнение массива с помощью оператора присваивания

(по правилу)

Задача. Заполнить массив А числами в 3 раза больше их порядкового номера.

```
program zapoln1;  
var n,i:integer;  
a:array[1..100]of integer;  
begin  
    write('vvedite kol-vo elementov -> '); readln(n);  
    for i:=1 to n do begin  
        a[i]:=i*3; ← элемент массива в  
        write (a[i]:3) 3 раза больше  
        end;           своего номера  
        writeln;  
    end.  
end.
```

# Вид пользовательского экрана после выполнения программы

```
vvedite kol-vo elementov -> 10  
3 6 9 12 15 18 21 24 27 30
```

В памяти формируется таблица A  
(количество элементов  $n=10$ )

<b>i</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b>7</b>	<b>8</b>	<b>9</b>	<b>10</b>
<b>a[i]</b>	3	6	9	12	15	18	21	24	27	30



## II. Копирование элементов массива

*Копированием* массивов называется присваивание всех элементов одного массива всем соответствующим элементам другого массива.

```
program zapln1;  
var n,i:integer;  
a,b:array[1..100]of integer;  
begin randomize;  
  writeln;  
  write('vvedite kol-vo elementov ->'); readln(n);  
  writeln('Massiv A:');  
  for i:=1 to n do begin  
    a[i]:=random(10)-15;  
    write (a[i]:3)  
  end;  
  
  writeln;  
  writeln('Massiv B:');  
  for i:=1 to n do begin  
    b[i]:=a[i];  
    write (b[i]:3)  
  end;  
  
end.
```

← массив A копируется  
в массив B

# Вид пользовательского экрана после выполнения программы

Массив **B** является копией массива **A**:

```
Введите кол-во элементов ->5  
Массив A:  
-6-11-14-13-11  
Массив B:  
-6-11-14-13-11
```





# III. Поиск элементов по некоторому условию

Задача. Дан массив **A**, заполненный случайными числами из промежутка от **10** до **20**.  
Сформировать новый массив **B** из номеров элементов массива **A**, меньших некоторого числа **M**.

```
program poisk_po_usloviu;
const n=10;
var i:integer; k:byte;
    a:array[1..100]of integer; b:array[1..n]of integer;
    m:integer;
    f:boolean;
begin
  randomize;
  for i:=1 to n do b[i]:=0;
  for i:=1 to n do begin
    a[i]:=random(10)+10;
    write (a[i]:3)
  end;

  writeln;
  write('Введите число M-критерий отбора элементов массива A-> ');
  readln(m);
  k:=0;
  f:=false;
  for i:=1 to n do
    if a[i]<M then begin
      k:=k+1;
      b[k]:=i;
      f:=true;
    end;

    if f then begin
      write('Элементы, меньшие ',m,' имеют номера: ');
      for i:=1 to k do write (b[i]:3);
      writeln;
    end
    else writeln('чисел, меньших ',m,' в массиве A нет');
  readln
end.
```

Реальный размер массива A

массив B не более массива A

Исходный массив

Конечный массив

Число критерия отбора элементов массива A

Обнуление элементов массива B

Заполнение массива A случайными числами от 10 до 20

Начальное значение количества элементов, меньших M

Предположим, что элементов, меньше M в массиве A нет

Переход к следующему номеру

Элемент, меньше M обнаружен

Заполнение массива B номерами элементов массива A, если они меньше числа M

Если числа, меньше M обнаружены, то их номера выводятся на экран

В противном случае выводится сообщение об отсутствии

31:5

# Вид пользовательского экрана после выполнения программы

1 запуск.

```
18 18 10 15 10 19 13 18 14 13
Введите число M-критерий отбора элементов массива A-> 15
Элементы, меньшие 15 имеют номера: 3 5 7 9 10
```

2 запуск.

```
14 18 17 11 18 15 17 12 11 12
Введите число M-критерий отбора элементов массива A-> 11
чисел, меньших 11 в массиве A нет
```



# IV. Подсчёт суммы, произведения и количества

Задача. В одномерном массиве найти:

- сумму положительных элементов
- произведение отрицательных
- количество четных элементов

```

program sum_proiz_kol;
var n,i,k:integer; s,p:longint;
a:array[1..100]of integer;
begin randomize;
  write('Введите количество элементов -> '); readln(n);
  for i:=1 to n do begin
    a[i]:=random(100)-50;
    write (a[i]:3)
  end; writeln;
  s:=0; p:=1; k:=0;
  for i:=1 to n do begin
    if a[i]>0 then s:=s+a[i];
    if a[i]<0 then p:=p*a[i];
    if a[i] mod 2=0 then k:=k+1;
  end;
  writeln('Сумма положительных элементов ',s);
  writeln('Произведение отрицательных элементов ',p);
  writeln('Количество четных элементов ',k);
end.

```

Заполнение массива случайными целыми числами от -50 до 50 → a[i]:=random(100)-50;  
 write (a[i]:3)  
 ← Начальные значения суммы, произведения и количества  
 s:=0; p:=1; k:=0;  
 Подсчёт суммы положительных элементов → if a[i]>0 then s:=s+a[i];  
 Подсчёт произведения отрицательных элементов → if a[i]<0 then p:=p\*a[i];  
 Подсчёт количества чётных элементов → if a[i] mod 2=0 then k:=k+1;  
 → writeln('Сумма положительных элементов ',s);  
 writeln('Произведение отрицательных элементов ',p);  
 writeln('Количество четных элементов ',k);  
 Вывод результатов на экран  
 end.

# Вид пользовательского экрана после выполнения программы

Два запуска программы:

```
Введите количество элементов -> 10
 25 14  2 -6-11  7-32  1 -9 14
Сумма положительных элементов 63
Произведение отрицательных элементов 19008
Количество четных элементов 5
Введите количество элементов -> 6
 -1 34 18  5  8-30
Сумма положительных элементов 65
Произведение отрицательных элементов 30
Количество четных элементов 4
```



# V. Нахождение максимального элемента и его номера

```
uses crt;
var n,i,max,imax:integer;
a:array[1..10]of integer;
begin clrscr;randomize;writeln;
  write('Введите количество элементов (n<=13) -> '); readln(n);
  writeln;writeln;
  write('№':5,' |');
  for i:=1 to n do begin
    write (i:3,' |')
  end;
  writeln;
  for i:=1 to 6*n do begin
    write ('-')
  end;
  writeln;
  write('a[i]':5,' |');
  for i:=1 to n do begin
    a[i]:=random<100>-50;
    write (a[i]:3,' |')
  end;
  writeln;writeln;
  max:=a[1]; imax:=1;
  for i:=2 to n do if max<a[i] then begin
    max:=a[i];
    imax:=i
  end;
  writeln;
  writeln('Максимальный элемент ',max,' находится на ',imax,' позиции');
  readln;
end.
```

Оформление массива на экране

Предположим, что максимальный элемент находится на 1 позиции

Сравниваем текущее значение max элемента с остальными элементами массива (со 2 до n)

Если значение максимального элемента оказалось меньше значения текущего элемента, то его значение заменяется значением просматриваемого элемента и номер фиксируется в величине imax

# Вид пользовательского экрана после выполнения программы

```
Введите количество элементов (n<=13) -> 12
```

```
№ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
```

```
-----  
a[i] | -37 | 2 | -6 | 45 | 49 | 23 | -11 | 28 | 20 | 2 | -13 | -46 |
```

```
Максимальный элемент 49 находится на 5 позиции
```



# VI. Сдвиг и вставка числа в одномерный массив

```
USES Crt;
CONST n=10;
VAR a:ARRAY[1..n+1] OF Integer;
    i,k: Byte;
    c: Integer;
BEGIN ClrScr;
      Randomize;
      FOR i:=1 TO n DO BEGIN a[i]:=Random(50)-10;Write(a[i]:4);END;
      Writeln;
      write('Введите элемент, который нужно вставить в массив: ');
      Readln(c);
      write('Введите номер позиции, на которую будет вставлен элемент ',c,' ( 1<=k<=',n,' ): ');
      Readln(k);
      FOR i:=n+1 DOWNTO k DO a[i]:=a[i-1];
      a[k]:=c;
      FOR i:=1 TO n+1 DO Write(a[i]:4);
      Readln;
END.
```

Начальный размер массива

Конечный размер массива

Режим очистки экрана

Вставляем число

Номер позиции, на которую вставляется элемент

Вставляем число в позицию массива k

последнего до k-го на 1 позицию вправо

Вывод конечного массива на экран



# Вид пользовательского экрана после выполнения программы

```
11 -4 14 -10 25 12 5 28 8 31
Введите элемент, который нужно вставить в массив: 555
Введите номер позиции, на которую будет вставлен элемент 555 ( 1<=k<=10 ): 7
11 -4 14 -10 25 12 555 5 28 8 31
```



## VI. Удаление элементов массива

- с указанным номером
- с указанным значением



Задача. В одномерном массиве удалить элемент с номером  $C$ .

Решение. Удаление указанного элемента происходит в результате *сдвига* элементов массива, начиная с номера  $C$  на 1 позицию *влево*.

Для  $C=2$  сдвиг происходит следующим образом:

$i$	1	2	3	4
$a[i]$	$a[1]$	$a[2]$	$a[3]$	$a[4]$



В итоге массив, уменьшается на 1 элемент

$i$	1	2	3
$a[i]$	$a[1]$	$a[2]$	$a[3]$

## Удаление элемента

```
CONST n=5;
VAR a:ARRAY[1..n] OF Integer;
    i: Byte;
    c: Integer;

BEGIN
  Randomize; FOR i:=1 TO n DO BEGIN a[i]:=Random(3)-2; Write(a[i]:4); END;
  Writeln;
  Write('Введите номер позиции элемента, который нужно удалить из массива: ');
  Readln(c);
  FOR i:=c TO n-1 DO a[i]:=a[i+1];
  Writeln('-----');
  FOR i:=1 TO n-1 DO Write(a[i]:4);
  Readln;
END.
```

Описание, заполнение и вывод на экран элементов массива

Сдвиг влево на 1 позицию

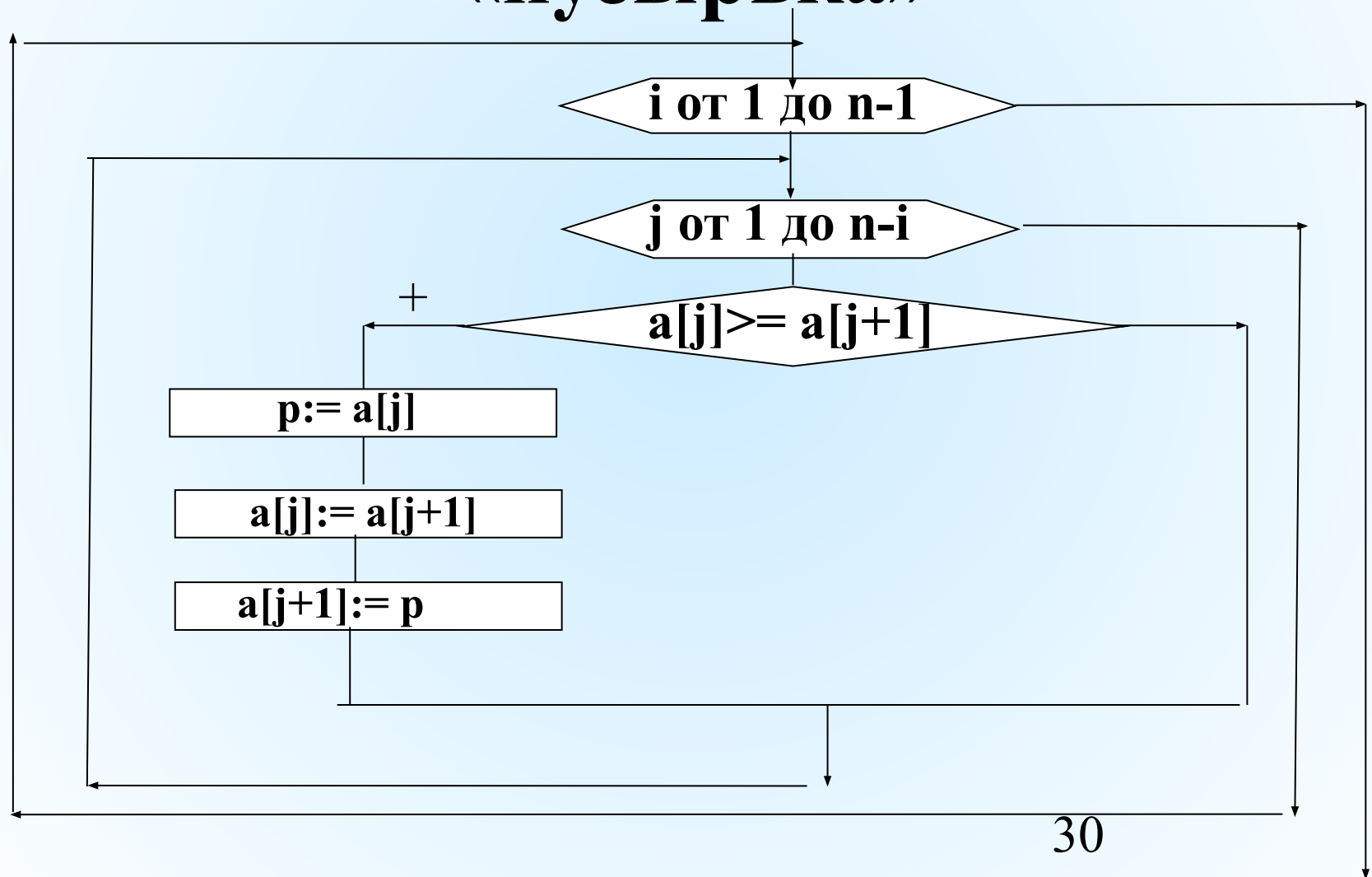
Вывод итогового массива на экран



# IV. Сортировка элементов одномерного массива

- *Сортировка* – распределение элементов множества по группам в соответствии с определенными правилами. Часто используют сортировку элементов массива по возрастанию или по убыванию.
- Сортировка *методом «пузырька»* предусматривает сравнение соседних элементов массива. Если они не упорядочены, то выполняется обмен значениями этих элементов. Таким образом упорядоченные элементы постепенно «всплывают».

# Блок-схема сортировки методом «пузырька»



# Метод «пузырька»

```
program sortirovka;  
const n=5;a:array [1..n] of integer=<15,10,13,27,12>;  
var i,j,p :integer;  
begin  
  for i:=1 to n-1 do  
    i элементов  
    упорядочено, граница  
    цикла подвижная  
    begin  
      for j:=1 to n-i do  
        сравниваем соседние  
        элементы  
        если предыдущий элемент больше следующего, то  
        их значения меняют местами  
        if a[j] > a[j+1] then  
          begin  
            p:=a[j];  
            a[j]:=a[j+1];  
            a[j+1]:=p  
            P - буфер обмена  
          end  
        end  
      end  
    end  
  for i:=1 to n do write (a[i], ' ' )  
end.
```

просматриваем элементы с 1 до предпоследнего

сравниваем соседние элементы

если предыдущий элемент больше следующего, то их значения меняют местами

P - буфер обмена

# Вид пользовательского экрана после выполнения программы

Массив после сортировки

```
10 12 13 15 27
```



[Посмотреть трассировку](#)