# Тип данных: МАССИВЫ

**Одномерный массив (вектор) – пронумерованный, конечный набор элементов одного типа**

Описание массива:

```
const N=10;

Type ZZ=array[1..N] of integer;
Var a:ZZ;

Var a:array[1..N] of integer;
```

# СТРУКТУРА ПРОГРАММЫ С ИСПОЛЬЗОВАНИЕМ МАССИВА

1. Описание массива
2. Ввод элементов массива
3. Обработка элементов массива
4. Вывод результатов

# Ввод элементов массива

1. Ввод с клавиатуры (используется при обработке реальных данных):

```
For k:=1 to N do
  Readln(a[k]);
```

2. Ввод случайных данных (используется для проверки работы алгоритма):

```
Randomize;
For k:=1 to N do
 begin
   a[k]:=random(100)-50;
   write(a[k]:4);
 end;
```

# Ввод элементов массива

3. Формульный ввод:

```
For k:=1 to N do
  a[k]:=k*k-2+3*k;
```

# Операции  над элементами массива

## Вычисление суммы элементов:

```
S:=0;
For k:=1 to N do
  S:=S+a[k];
Writeln (S);
```

# СРЕДНЕЕ АРИФМЕТИЧЕСКОЕ ПОЛОЖИТЕЛЬНЫХ ЭЛЕМЕНТОВ МАССИВА

```
Sum:=0;  C:=0;
For i:=1 to N do
 If a[i]>0 then
                begin
                  Sum:=Sum+a[k];  {сумма
              положительных}      C:=C+1;     {счетчик
положительных}
                end;
  Writeln (Sum/C:0:2);
```

# Нахождение МИНИМАЛЬНОГО ЭЛЕМЕНТА И ЕГО ИНДЕКСА

```
Min := 37678;
For k:=1 to N do
  If a[k]<Min then
                    begin
                      Min:=a[k];
                      Y:=k;
                    end;
```

# Удаление элемента массива с номером К

```
Begin
 {Ввод элементов массива}
  readln (k);
 for i:=k to n-1 do {сдвиг массива справа налево}
   a[i]:=a[i+1];
 n:=n-1;
 { вывод массива}
End.
```

НАПИСАТЬ КОД СДВИГА СЛЕВА НАПРАВО

# Вставка элемента массива на место с номером К

```
Begin
 {Ввод элементов массива}
  readln (k);
 for i:=k+1 to n do {сдвиг массива вправо}
   a[i]:=a[i-1];
 readln (a[k]);
 { вывод массива}
End.
```

НАПИСАТЬ КОД СДВИГА ВЛЕВО

# Сортировка массива методом «пузырька»

**Алгоритм сортировки:**

При первом проходе по массиву элементы попарно сравниваются между собой: первый со вторым, затем второй с третьим, следом третий с четвертым и т.д. Если предшествующий элемент оказывается больше последующего, то их меняют местами.

Постепенно самое большое число оказывается последним. Остальная часть массива остается не отсортированной, хотя некоторое перемещение элементов с меньшим значением в начало массива наблюдается.

При втором проходе незачем сравнивать последний элемент с предпоследним. Последний элемент уже стоит на своем месте. Значит, число сравнений будет на одно меньше.

На третьем проходе уже не надо сравнивать предпоследний и третий элемент с конца. Поэтому число сравнений будет на два меньше, чем при первом проходе. В конце концов, при проходе по массиву, когда остаются только два элемента, которые надо сравнить, выполняется только одно сравнение. После этого первый элемент не с чем сравнивать, и, следовательно, последний проход по массиву не нужен.

Другими словами, количество проходов по массиву равно m-1, где m – это количество элементов массива. Количество сравнений в каждом проходе равно m-i, где i – это номер прохода по массиву (первый, второй, третий и т.д.). При обмене элементов массива обычно используется "буферная" (третья) переменная, куда временно помещается значение одного из элементов.

# Сортировка массива методом «пузырька»

```
{основной блок}
   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);
```

# Сортировка прямым выбором

```
for i:=1 to n-1 do
        begin k:=i;
            for j:=i+1 to n do
                if a[j]<a[k] then
                    k:=j;
                    x:=a[k];
                    a[k]:=a[i];
                    a[i]:=x;
         end;
```

ПРОВЕРИТЬ и, ЕСЛИ НУЖНО, дополнить!