МАССИВЫ

Задача: в одномерном массиве, состоящем из n целых чисел найти минимальный по модулю элемент и его номер

Задание: Ниже приведен фрагмент решения некоторой задачи. Внимательно рассмотрев решение, сформулируйте решаемую задачу

```
const n = 20;
var
        arrA, arrB: array[1..n] of real;
        i, j: byte;
        sum: real;

begin
        randomize;
        writeln('Array A:');
        for i:=1 to n do begin
                arrA[i] := random() * 10 - 5; // (-5;+5)
                write(arrA[i]:6:2);
                if i mod 10 = 0 then writeln;
        end;
```

```
const N = 10;
var arr: array[1..N] of integer;
i, k: byte; sum: integer;
avr: real;
begin
…….
  sum := 0; i := 1; k := 0;
  while i <= N do
    begin
       sum := sum + arr[i]; k := k + 1; i := i + 2
    end;
  writeln(sum);
  avr := sum / k;
  writeln(avr);
  readln;
end.
```

сформулировать условие задачи, которая решается в данном фрагменте программы:

Находится сумма элементов массива с нечетными индексами и их среднее арифметическое.

```
const N = 10;
var arr: array[1..N] of integer;
i, k: byte; sum: integer;
avr: real;
begin
.......
  sum := 0; i := 1; k := 0;
  while i <= N do
    begin
      if (arr[i] mod 2) = 0 then
        begin sum := sum + arr[i]; k := k + 1 end;
      i := i + 2
    end;
  writeln(sum);
  if k <> 0 then
    begin avr := sum / k; writeln(avr)
    end
 else writeln('No elements');
readln; end.
```

сформулировать условие задачи, которая решается в данном фрагменте программы:

Находится сумма четных элементов массива с нечетными индексами и их среднее арифметическое, при условии, что такие элементы существуют.

Точно и четко сформулирова
решается в данн

Вводится с клавиатуры количество элементов массива, сами элементы массива. Находится максимальный элемент, и каждый элемент массива увеличивается на значение максимального элемента. Полученный массив выводится на экран.

- ```
  Program Kr_2_3;
  Const NMax = 100;
  Type LinMass = Array[1..NMax] Of Integer;
  Var A : LinMass; N, I, M : Integer;
  Begin
    Write('Количество элементов массива? '); ReadLn(N);
    M := -32768;
    For I := 1 To N Do
    Begin
        Write('Введите A[', I, '] '); ReadLn(A[I]);
        If A[I] > M Then M := A[I]
    End;
    For I := 1 To N Do A[I] := A[I] + M;
    For I := 1 To N Do Write(A[I] : 6);
    WriteLn
  End.
  ```
  ```

Сортировка массивов

Метод «пузырька»

Метод пузырька

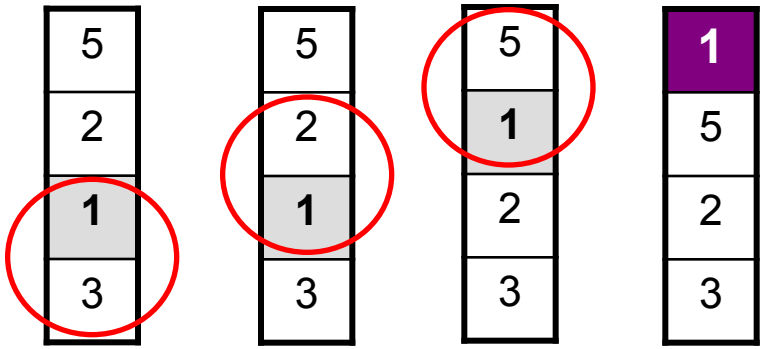
- Сортировка методом «пузырька» использует метод обменной сортировки и основана на выполнении в цикле операций сравнения и при необходимости обмена соседних элементов.

Метод пузырька. Идея

Идея – пузырек воздуха в стакане воды поднимается со дна вверх.

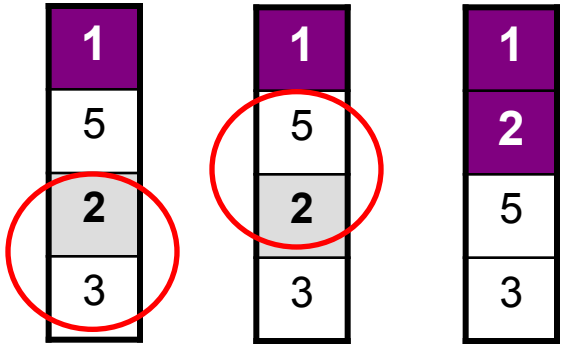
Для массивов – самый маленький ("легкий") элемент перемещается вверх ("всплывает").

1-ый проход

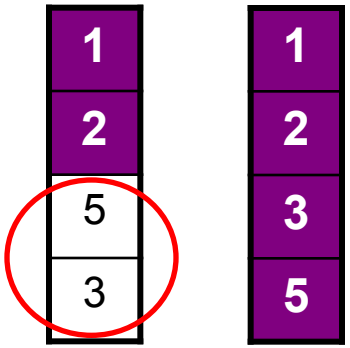


- начиная снизу, сравниваем два соседних элемента; если они стоят "неправильно", меняем их местами
- за 1 проход по массиву один элемент (самый маленький) становится на свое место

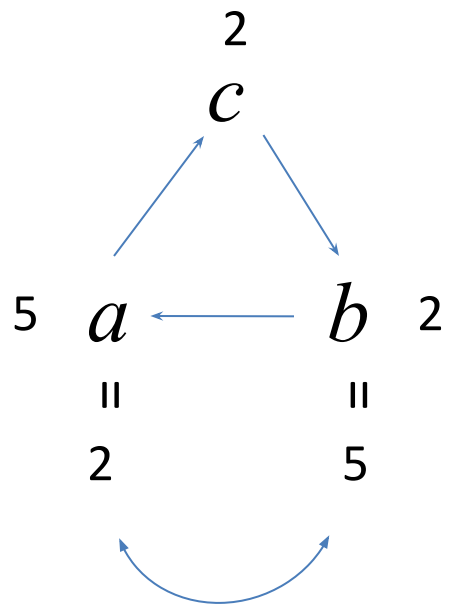
2-ый проход



3-ий проход



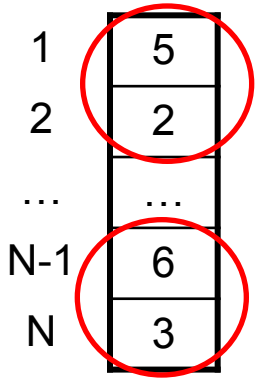
Для сортировки массива из N элементов нужен N-1 проход (достаточно поставить на свои места N-1 элементов).



Как поменять значения?

Метод пузырька. Программа

1-ый проход:



сравниваются пары

A[N-1] и A[N], A[N-2] и A[N-1]

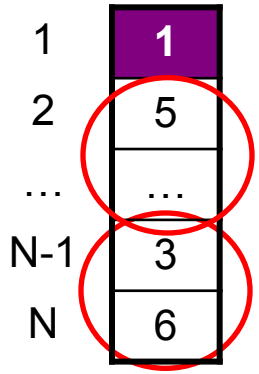
…

A[1] и A[2]

A[j] и A[j+1]

```
for j:=N-1 downto 1 do
  if A[j] > A[j+1] then begin
    c:=A[j]; A[j]:=A[j+1]; A[j+1]:=c;
  end;
```

2-ой проход



A[1] уже на своем месте!

```
for j:=N-1 downto 2 do
  if A[j] > A[j+1] then begin
    c:=A[j]; A[j]:=A[j+1]; A[j+1]:=c;
  end;
```

i-ый проход

```
for j:=N-1 downto i do
  ...
```

Метод пузырька. Программа

```
program qq;
const N = 10;
var A: array[1..N] of integer;
    i, j, c: integer;
begin
  { заполнить массив }
  { вывести исходный массив }
  for i:=1 to N-1 do begin
    for j:=N-1 downto i do
      if A[j] > A[j+1] then begin
        c := A[j];
        A[j] := A[j+1];
        A[j+1] := c;
      end;
   end;
  { вывести полученный массив }
end;
```

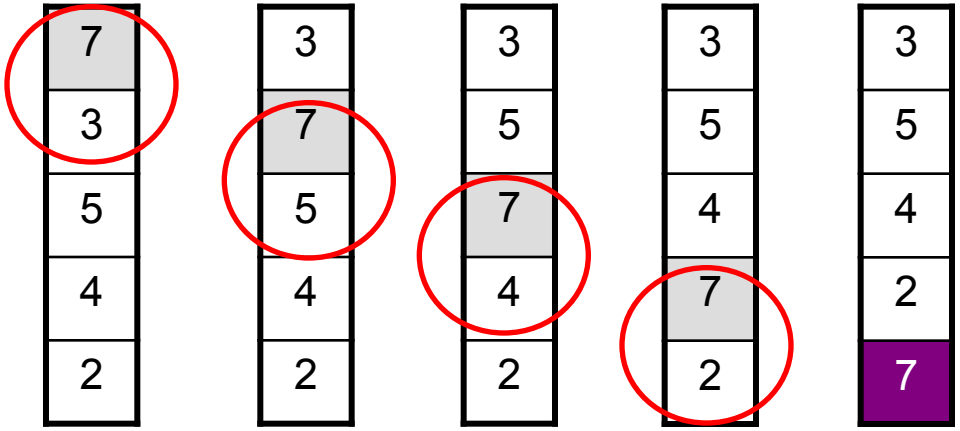
? Почему цикл по `i` до `N-1`?

элементы выше `A[i]` уже поставлены

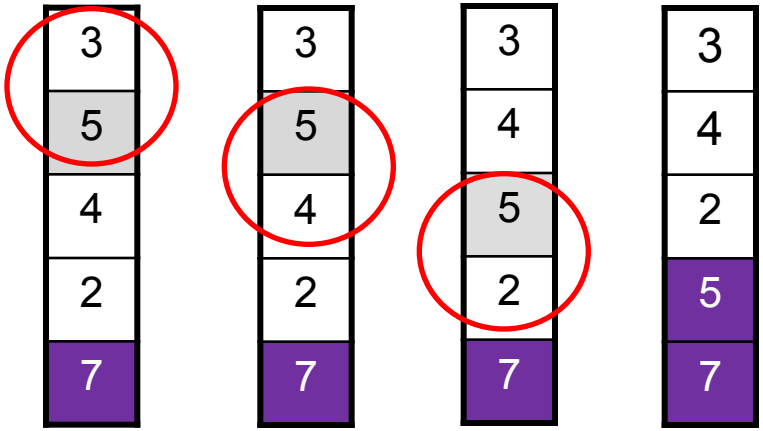
Метод пузырька. Идея

Или наоборот - самый большой ("тяжелый") элемент перемещается вниз ("тонет").

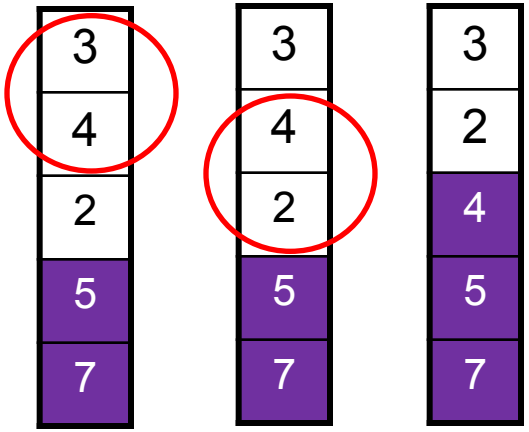
1-ый проход



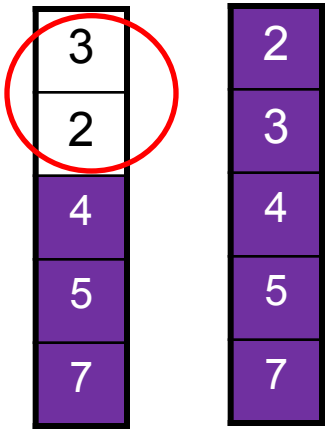
2-ый проход



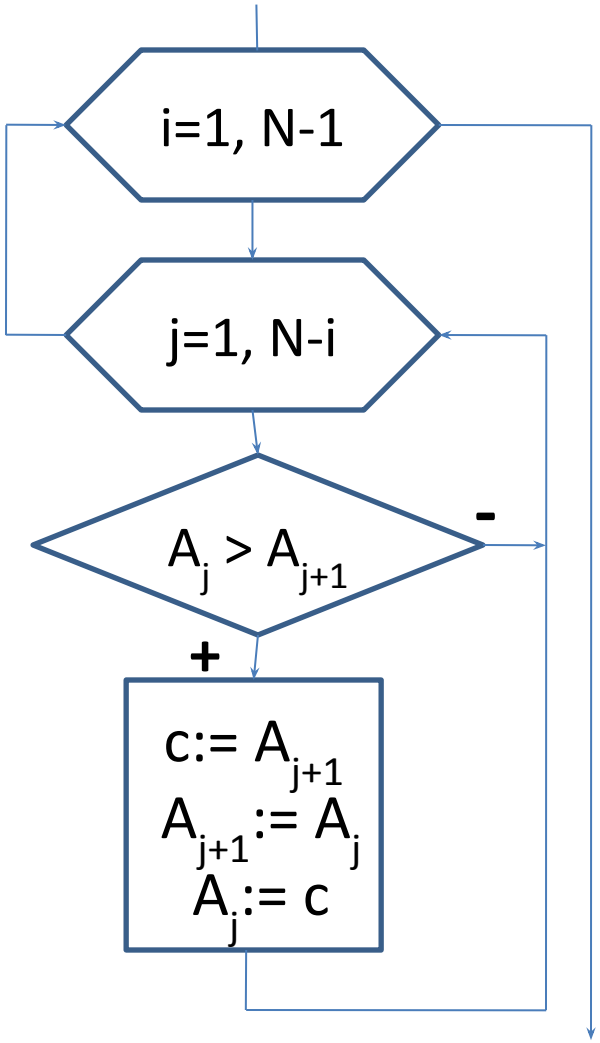
3-ый проход



4-ый проход



Метод пузырька. Алгоритм




```
program qq;
const N = 10;
var A: array[1..N] of integer;
    i, j, c: integer;
begin
  { заполнить массив }
  { вывести исходный массив }
  for i:=1 to N-1 do begin
    for j:=1 to N-i do
      if A[j] > A[j+1] then
begin
        c := A[j+1];
        A[j+1] := A[j];
        A[j] := c;
      end;
   end;
  { вывести полученный массив }
end;
```

Сортировка массивов

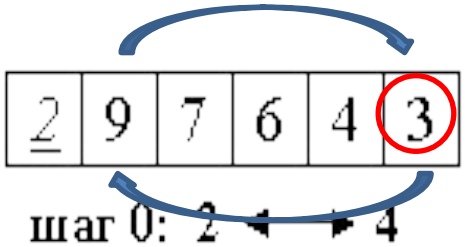
Сортировка выбором

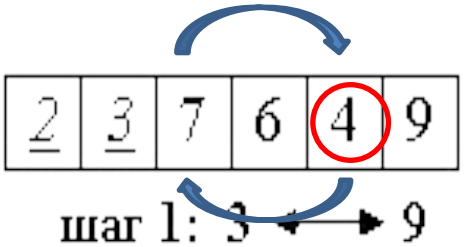
Сортировка выбором. Идея

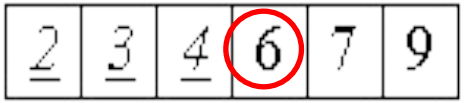
- При сортировке массива a[1], a[2], ..., a[n] методом простого выбора среди всех элементов находится элемент с наименьшим значением a[i], и a[1] и a[i] обмениваются значениями. Затем этот процесс повторяется для получаемых подмассивов a[2], a[3], ..., a[n], ... a[j], a[j+1], ..., a[n] до тех пор, пока мы не дойдем до подмассива a[n], содержащего к этому моменту наибольшее значение.

	8
	5
	2
	6
	9
	3
	1
	4
	0
	7

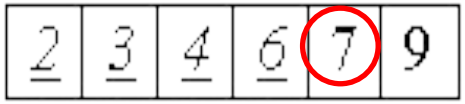
Сортировка выбором. Пример



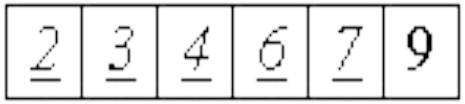




шаг 2: 4 ⟷ 7

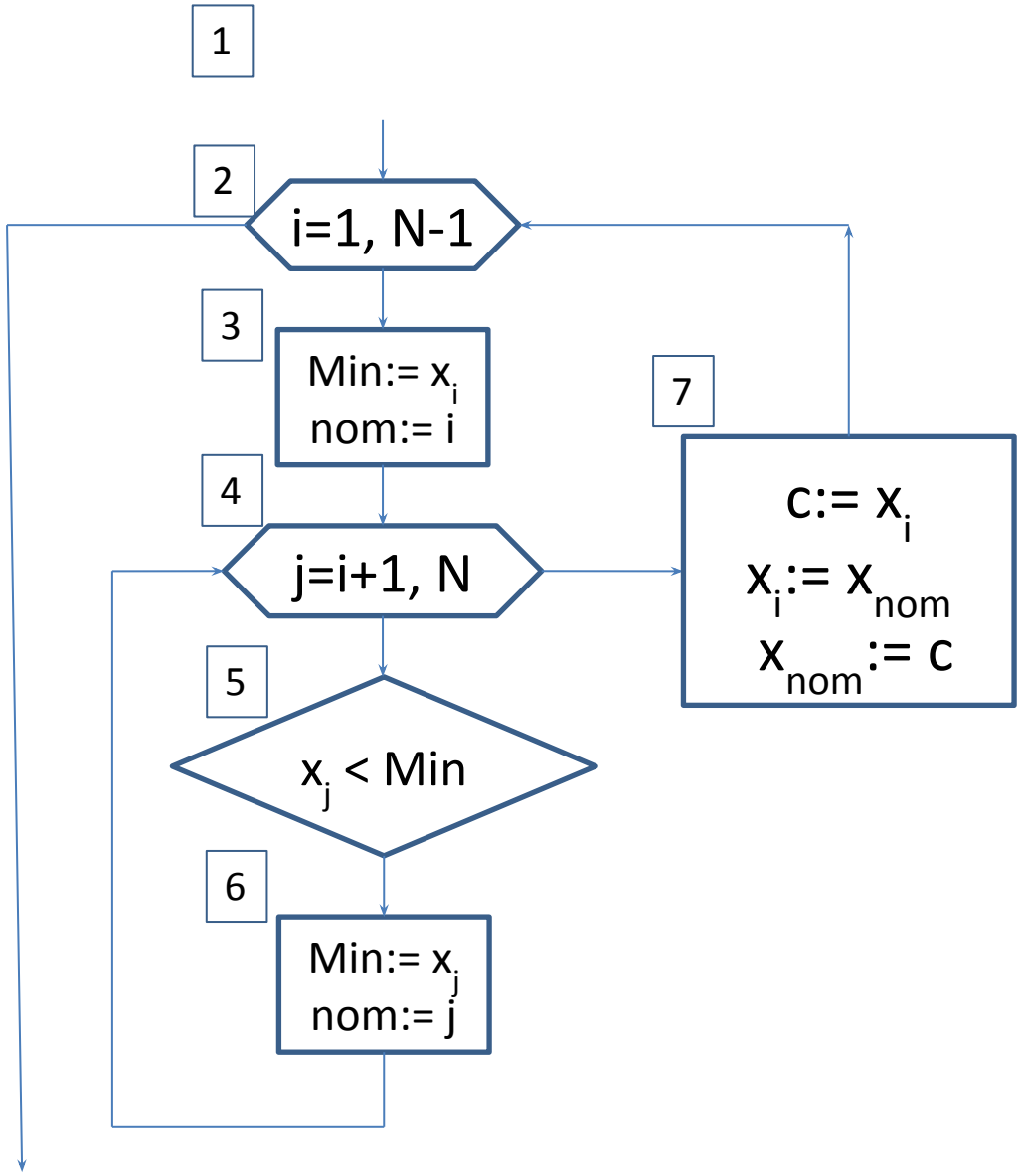


шаг 3: 6 ⟷ 6

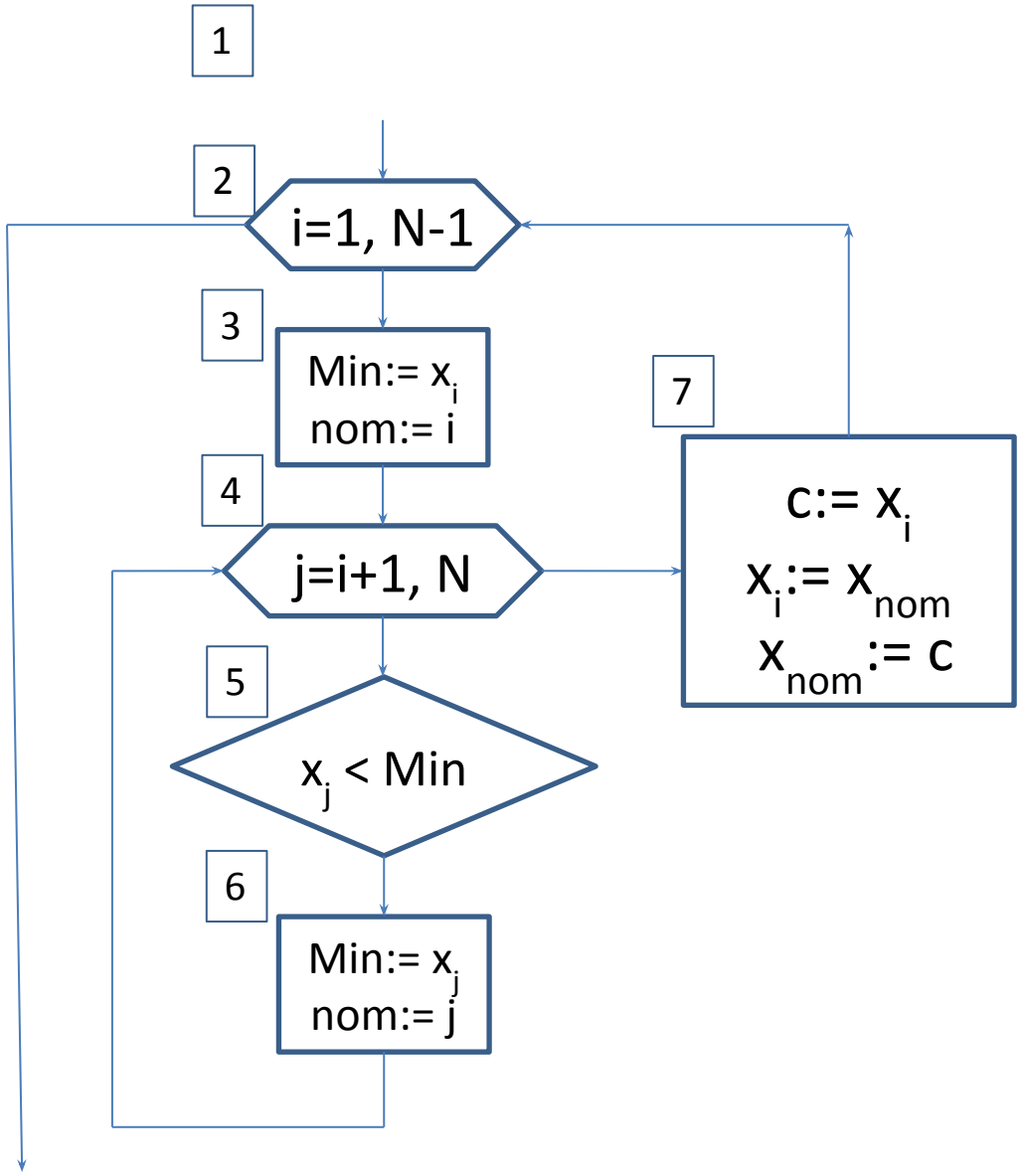


шаг 4: 7 ⟷ 7

Сортировка выбором. Алгоритм



Найдем в массиве самый маленький элемент (блоки 3–6) и поменяем его местами с первым элементом (блок 7). Повторим алгоритм поиска минимального элемента, начиная со второго, и поменяем его местами со вторым элементом (блоки 3–7). Описанную выше операцию поиска проводим до полного упорядочивания элементов в массиве.




```
…
for i := 1 to  N-1  do
begin
    Min:= x[ i ];
   nom := i ;
   for j:= i+1 to N do
     if x[j] < Min then
  begin
  Min:=x[j]; nom := j;
  end;
   if nom <> i then begin
     c:=x[i];
     x[i]:=x[nom];
     x[nom]:=c;
   end;
end;
```

Сортировка выбором. Программа

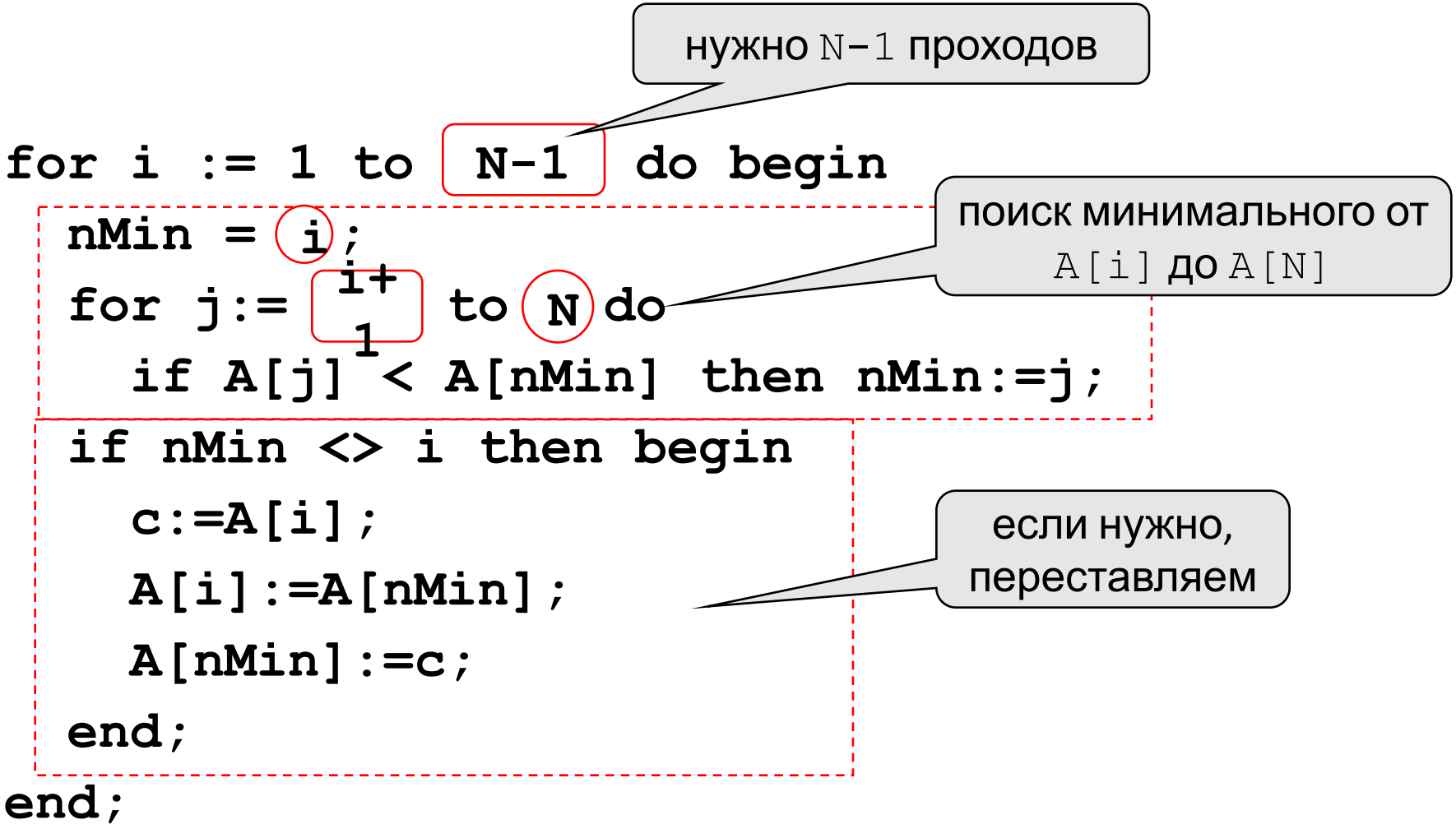
```
for i := 1 to N-1 do begin
   Min:= x[i];
   nom := i ;
   for j:= i+1 to N do
      if A[j] < A[nom] then
 begin
 Min:=A[j]; nom := j; end;
   if nom <> i then begin
      c:=A[i];
      A[i]:=A[nom];
      A[nom]:=c;
   end;
end;
```

нужно N-1 проходов

поиск минимального с A[i+1] до A[N]

если нужно, переставляем

Сортировка выбором. Программа



```
program Project1;
const n = 10;
var
  x: array[1..n] of integer; //объявляем массив
  i, j, nom: integer;
  Min, c: integer;
begin
 v
ra
fo
  for i:= 1 to  n  do
    begin
      Min:=x[i];
      nom:=i ;
      for j:=i+1 to n do //поиск миним. эл-та в неотсортированной части
       if x[j]<Min then
         begin
           Min:=x[j];
           nom:= j;
         end;
        c:=x[i];//меняем местами очередной минимум и найденный миним. эл
        x[i]:=x[nom];
        x[nom]:=c;
     end;
    writeln('Отсортированный массив'); //вывод отсортированного массива
  for i:=1 to n do write(x[i],' ');
  readln;
  end.
```

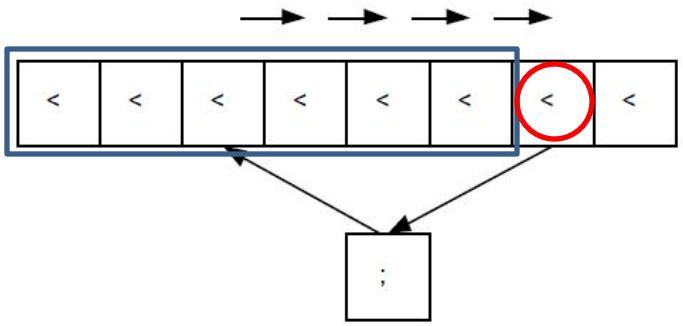
Сортировка массивов

Метод вставки

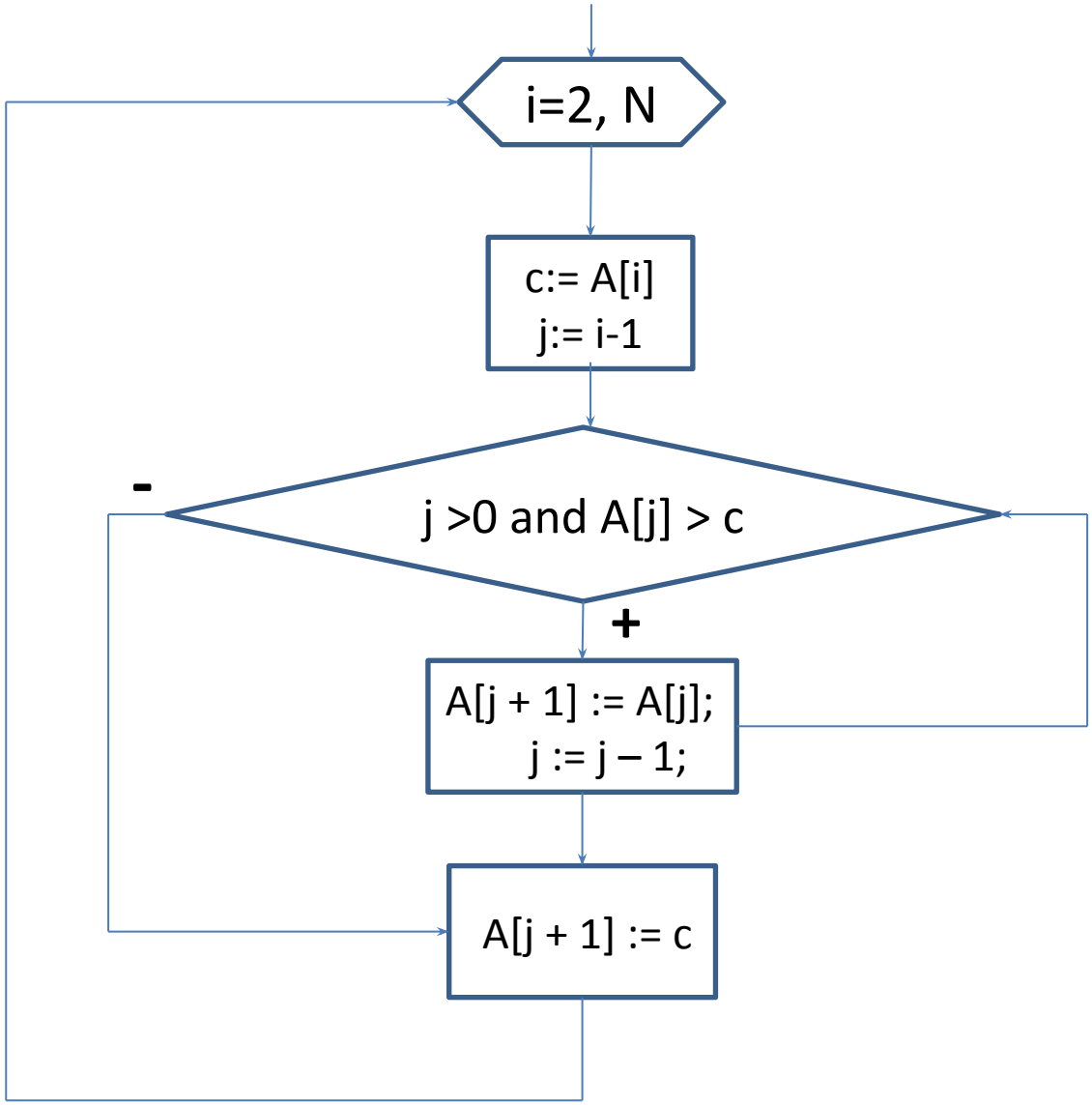
Метод вставки. Идея

На каждом шаге алгоритма мы выбираем один из элементов входных данных и вставляем его на нужную позицию в уже отсортированной части массива, до тех пор пока набор входных данных не будет исчерпан.

Метод выбора очередного элемента из исходного массива произволен; может использоваться практически любой алгоритм выбора. Обычно (и с целью получения устойчивого алгоритма сортировки), элементы вставляются по порядку их появления во входном массиве.



Метод вставки. Алгоритм



Метод вставки. Программа

```
for i = 2 to N do
begin
   c := A[i];
   j := i – 1;
   while (j > 0) and (A[j] > c) do
     begin
       A[j + 1] := A[j];
       j := j – 1;
     end;
   A[j + 1] := c;
end;
```

Двумерные массивы

Двумерный массив можно представить себе в виде таблицы (матрицы), в которой все строки и столбцы пронумерованы.

Каждый элемент такого массива имеет два индекса:

Первый индекс – это номер строки;

Второй индекс – номер столбца.

A[1,1]	A[1,2]	A[1,3]	A[1,4]	A[1,5]
A[2,1]	A[2,2]	A[2,3]	A[2,4]	A[2,5]
A[3,1]	A[3,2]	A[3,3]	A[3,4]	A[3,5]
A[4,1]	A[4,2]	A[4,3]	A[4,4]	A[4,5]

Описание двумерных массивов:

```
Const  n=4;
       m=5;
Var   A :array [1..n, 1..m] of integer;
```

Строки **Столбцы**

A [2,4]

A [4,2]

A[1,1]	A[1,2]	A[1,3]	A[1,4]	A[1,5]
A[2,1]	A[2,2]	A[2,3]	A[2,4]	A[2,5]
A[3,1]	A[3,2]	A[3,3]	A[3,4]	A[3,5]
A[4,1]	A[4,2]	A[4,3]	A[4,4]	A[4,5]

Описание двумерных массивов. Примеры

Пример 1.

```
const N = 3;
      M = 4;
var A: array[1..N,1..M] of integer;
    B: array[-3..0,-8..M] of integer;
    Q: array['a'..'d',False..True] of real;
```

Описание двумерных массивов. Примеры

- **Пример 2**. Массив можно описать как одномерный, элементами которого в свою очередь являются одномерные массивы.

- ```
  Const
    n=20; m=30;
  Type
    MyArray1 = array [1..m] of integer;
    MyArray2 = array [1..n] of MyArray1;
  Var
    V : MyArray1;
    A : MyArray2;
  ```
  ```

Описание двумерных массивов. Примеры

Пример 3.

```
Const
  n=20; m=30;
Type
  MyArray2 = array [1..n] of array [1..m] of integer;
Var
  A : MyArray2;
```

Описание двумерных массивов. Примеры

Пример 4.

```
Const
  n=20; m=30;
Type
  MyArray2 = array [1..n, 1..m] of integer;
Var
  A : MyArray2;
```

Создание двумерных массивов

Ввод с клавиатуры:

Если переставить циклы?

```
for j:=1 to M do
  for i:=1 to N do begin
    write('A[',i,',',j,']=');
    read ( A[i,j] );
  end;
```

i j

A[1,1] 2
A[2,1] 5
A[3,1] 4
=
A[3,4] 5
=

Заполнение случайными числами

```
for i:=1 to N do
  for j:=1 to M do
    A[i,j] := random(25) - 10;
```

цикл по строкам

цикл по столбцам

...тервал?

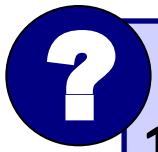
Создание двумерных массивов

Заполнение по некоторому правилу

```
For i:=1 to n do
     for j:=1 to m do
         a[i,j]:=i*j;
```

```
A[1,1]  1
A[1,2]  2
A[1,3]  3
   =
  ...
A[3,4]  1
   =    2
```

```
Program Vvod2;
 Var I, J : Integer;
    A : Array [1..20, 1..20] Of Integer;
 Begin
  FOR I := 1 TO 3 DO
   FOR J := 1 TO 2 DO A[I, J] := 456 + I
 End.
```



Назовите A[1, 1], A[1, 2], A[2, 1], A[2, 2], A[3, 1], A[3, 2].

Задание: Ниже приведен фрагмент решения некоторой задачи. Внимательно рассмотрев решение, сформулируйте решаемую задачу

```
for i := 1 to n do
begin
    for j := 1 to m do
      write(X[i, j]:5);
      writeln;
 end;
```

```
Begin
  Randomize;
    for i := 1 to n do
      for j := 1 to m do
        X[i, j]:= Random(50);
End;
```

Обработка всех элементов массива

Задача: заполнить матрицу из 3 строк и 4 столбцов случайными числами и вывести ее на экран. Найти сумму элементов матрицы.

```
program qq;
const N = 3; M = 4;
var A: array[1..N,1..M] of integer;
    i, j, S: integer;
begin
 S := 0;
  Randomize;
 for i:=1 to N do
  for j:=1 to M do
    begin
      A[i,j] := random(25) - 10;
       S := S + A[i,j];
    end;
writeln('Сумма элементов матрицы ', S);
end;
```

Обработка двумерных массивов

- Для обработки двумерных массивов могут применяться те же методы, что и для одномерных массивов.
- Поскольку положение элемента в двумерном массиве описывается двумя индексами, программы большинства задач строятся на основе вложенных циклов.

Стандартные задачи обработки массивов

- Нахождение элементов и количества элементов с данным свойством
- Определить, отвечает ли заданный массив некоторым требованиям
- Изменение значений некоторых элементов, удовлетворяющих заданному свойству
- Заполнение массива по правилу

НАХОЖДЕНИЕ ЭЛЕМЕНТОВ И КОЛИЧЕСТВА ЭЛЕМЕНТОВ С ДАННЫМ СВОЙСТВОМ

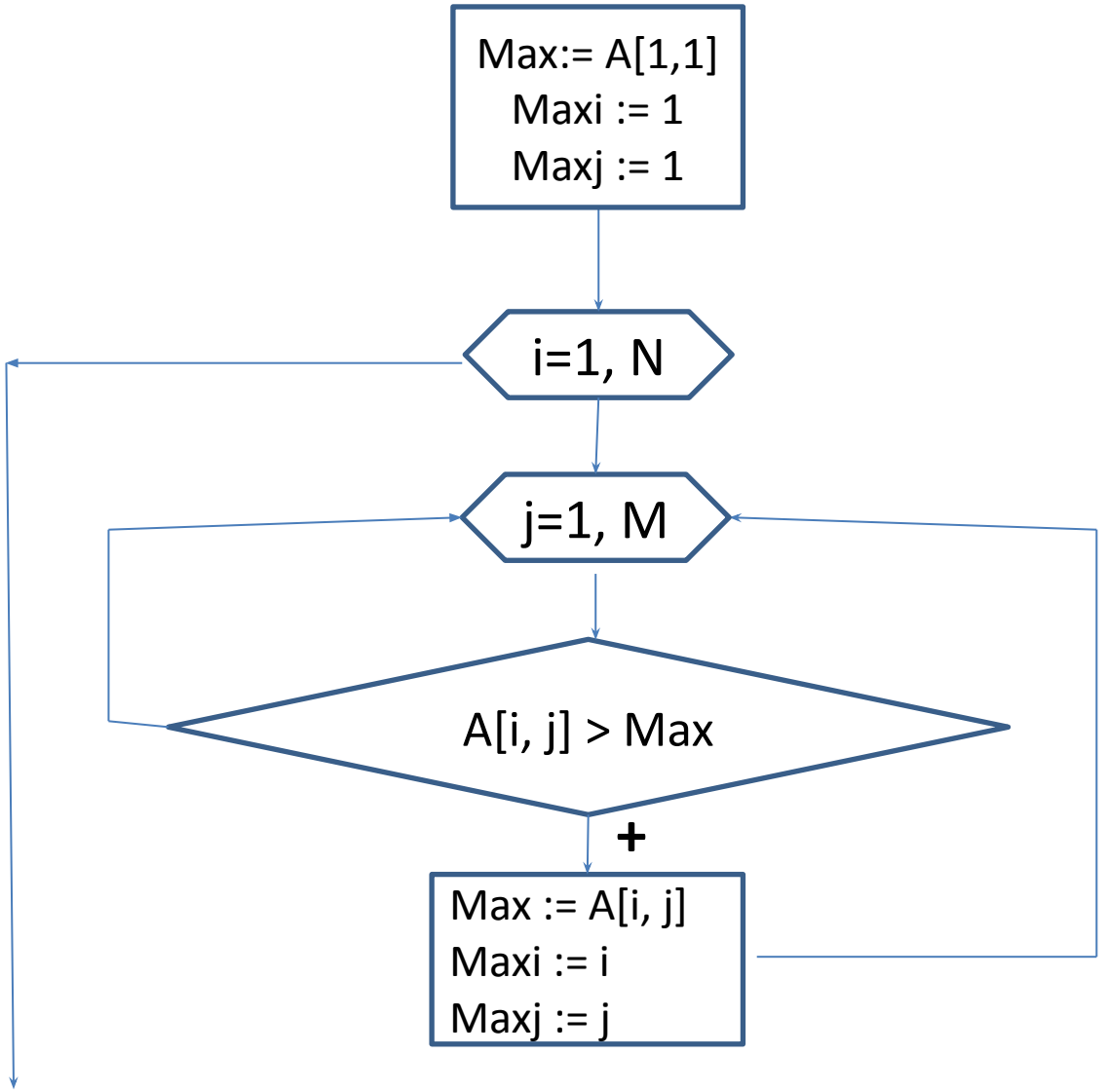
Задача 1. Найти максимальный элемент массива и его индексы.

- Идея:

1. Предположим, что максимумом является первый элемент → запомним первую строку и первый столбец
2. Пробегаем последовательно строки и столбцы массива
3. Проверяем: если среди элементов массива нашелся больший элемент, то внесем новое найденное значение в переменную Max и запомним новую строку и новый столбец

Задача 1. Найти максимальный элемент массива и его индексы.

- Алгоритм:



Задача 1. Найти максимальный элемент массива и его индексы.

- Программа:

```
Max := X[1, 1];
  Maxi := 1;
  Maxj := 1;
   for i := 1 to n do
     for j := 1 to m do
       if X[i, j] > Max
         then
           begin
             Max := X[i, j];
             Maxi := i;
             Maxj := j;
           end;
```

{Предположим, что максимумом является первый элемент}

{запомним первую строку и первый столбец}

{если среди элементов массива нашелся больший элемент, то}

{внесем новое найденное значение в переменную Max}

{запомним индексы строки и столбца }

Задача 2. Найти количество отрицательных элементов в массиве.

Задача 3. Найти количество отрицательных элементов в каждой строке.

- **Способ 1** - использовать счетчик, находить количество элементов строки и выводить значение на экран.

```
for i := 1 to n do
   begin
     k := 0;
      for j := 1 to m do
         if X[i, j] < 0     then          k:=k+1;
      writeln(i,' - ', k);
   end;
```

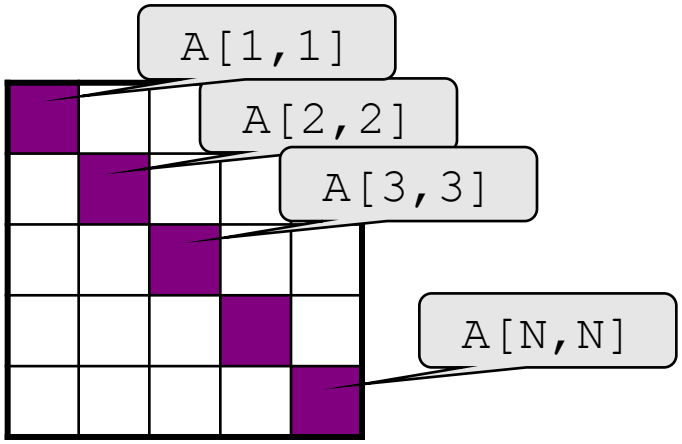
Задача 3. Найти количество отрицательных элементов в каждой строке.

- **Способ 2** - количество элементов каждой строки хранить в одномерном массиве (Y) соответствующей размерности.

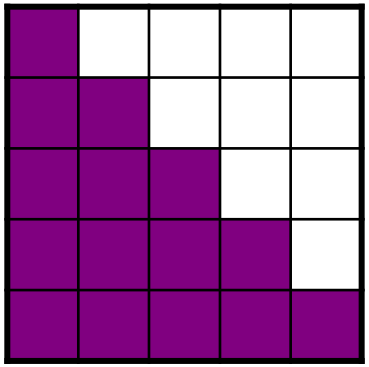
```
for i := 1 to n do
   begin
    Y[i] := 0; {записываем начальное значение количества элементов в соответствующую столбцу ячейку}
      for j := 1 to m do
       if X[i, j] < 0 {если отрицательный элемент найден}
         then
          Y[i] := Y[i]+1; {то увеличиваем текущее значение на единицу}
   end;
```

Задание: Ниже приведен фрагмент решения некоторой задачи. Внимательно рассмотрев решение, сформулируйте решаемую задачу

```
for i:=1 to N do
write(A[i,i]:5);
```



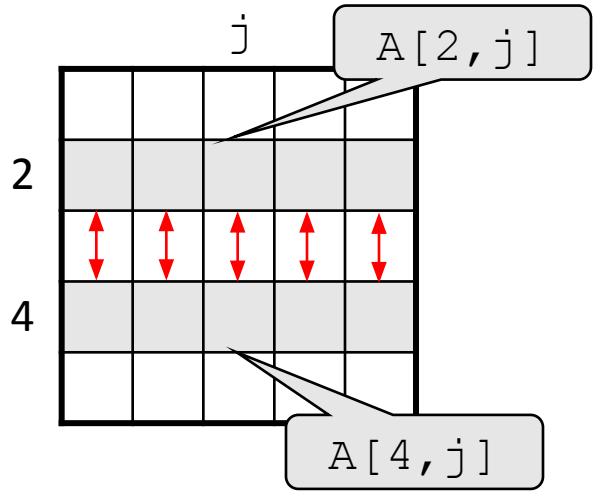

```
S := 0;
for i:=1 to N do
   for j:=1 to i do
      S := S + A[i,j];
```

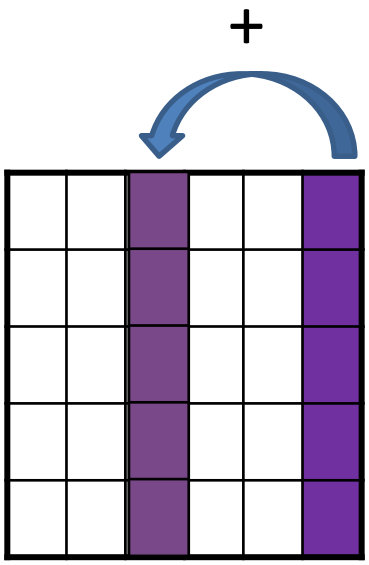


Задание: Ниже приведен фрагмент решения некоторой задачи. Внимательно рассмотрев решение, сформулируйте решаемую задачу

```
for j:=1 to M do
begin
   c := A[2,j];
   A[2,j] := A[4,j];
   A[4,j] := c;
end;
```

```
for i:=1 to N do
A[i,3]:=A[i,3]+A[i,6];
```





ОПРЕДЕЛИТЬ, ОТВЕЧАЕТ ЛИ ЗАДАННЫЙ МАССИВ НЕКОТОРЫМ ТРЕБОВАНИЯМ

Задача. Определить, является ли данный квадратный массив симметричным относительно своей главной диагонали.

Массив является симметричным, если для него выполняется равенство A[i, j]=A[j, i] для всех i=1, ..., n и j=1, ..., n.

Напишем **логическую функцию**, значение которой равно истине, если условие Flag выполняется, и ложь в противном случае. Будем сравнивать элементы и, если найдем неравные элементы, то присвоить функции значение False, иначе -True.

```
 Flag := True; {Предполагаем, что матрица симметрична}
 i := 2;
 while Flag and (i<n) do
   begin
     j := 1;
     while (j<i) and (X[i, j]=X[j, i]) do
       Inc(j);
     Flag := (j=i);
     Inc(i);
   end;
....
 End.
```

Зада

ом

масс

```
program Project1;
var i,j,k,n,m:integer;
A:array[1..255,1..255] of integer;
begin
k:=0;
randomize;
write('Введите количество строчек в массиве: '); read(n);
write('Введите количество столбцов в массиве: '); read(m);
for i := 1 to n do
begin
  for j := 1 to m do
    begin
      A[i,j]:=random(50)-25;
      write(' ', A[i,j]:3);
    end;
  writeln;
end;
for i := 1 to n do
  for j := 1 to m do
    begin
      if A[i,j]=0 then
        begin
          k:=k+1;
          write(' A[',i,',',j,']');
        end;
    end;
    writeln('Количество нулевых элментов: ',k);
 end.
```

ТЕСТ

1. Задан одномерный массив x[1..N]. Фрагмент алгоритма

```
s:=0; нц для k от 1 до N
| если (0<x[k])
| | то s:=s+x[k]
| все
кц
```

определяет:

1) максимальный элемент массива;
2) сумму положительных элементов;
3) количество положительных элементов;
4) индекс последнего положительного элемента;
5) индекс первого положительного элемента.

2. Для массива X[1..n] алгоритм

```
P:=0;
for k:=n downto 1 do
if X[k]<>T then P:=k;
```

определяет:

1) Номер последнего элемента массива, не равного T;
2) Количество элементов массива, не равных T;
3) Номер первого элемента массива, не равного T;
4) Номер последнего элемента, равного T;
5) Количество элементов, равных T;
6) Ни один из ответов 1-5 не верен.

3. Задан фрагмент алгоритма и три массива по шесть элементов в каждом. Определить, какой из данных массивов упорядочивается по возрастанию после обработки алгоритмом.

```
нц для k от 1 до 3
| если (x[k] > x[3+k])
| | то S:=x[k]; x[k]:= x[3+k];
x[3+k]:=S;
| все
кц
```

a) 26, 17, 35, 62, 53, 44
b) 17, 35, 44, 53, 26, 62
c) 62, 17, 44, 53, 26, 35

4. Задан двумерный массив A[1..n,1..n]. Фрагмент алгоритма

```
s:=0
нц для i от 1 до n
нц для j от 1 до n
если A[i,j]>0 то s:=s+A[i,j]* A[i,j]
кц
кц
```

вычисляет:

1) сумму положительных элементов массива
2) количество положительных элементов массива
3) сумму квадратов элементов массива
4) количество квадратов положительных элементов массива
5) сумму квадратов положительных элементов массива

5. Дан массив A[1,6], состоящий из чисел **1, -2, -3, 2, -4, 0.** Укажите, какой из предложенных массивов получен в результате выполнения алгоритма:

```
ib:=1; ifin:=6
нц для i от 1 до 6
если a[i]>0
то c[ib]:= a[i]; ib:=ib+1
иначе c[ifin]:=a[i]; ifin:=ifin-1
кц
```

1) 1, 2, 0, -2, -3, -4;
2) 1, 2, 0, -4, -3, -2;
3) 0, 1, 2, -4, -3, -2;
4) 0, 1, 2, -2, -3, -4;
5) 2, 1, 0, -4, -3, -2.

Критерии оценивания теста

Кол-во правильных ответов	Оценка
5	5
4	4
3	3
0, 1, 2	2

Ответы

№ вопроса	Ответ
1	2
2	3
3	В
4	5
5	2

САМОСТОЯТЕЛЬНАЯ РАБОТА

1 Вариант

1. Измените знак всех нечетных (четных) элементов массива, состоящего из L чисел
2. Найти и вывести на экран индексы четных элементов каждой строки массива (если их нет выдать соответствующее сообщение)

2 Вариант

1. Дан одномерный целочисленный массив A(1..N). Вывести на экран индексы равных элементов.
2. Найти сумму и количество отрицательных элементов каждого столбца, меньших заданного числа a;