Глава 3 Обработка массивов

Массив – это упорядоченная совокупность *однотипных данных*. Каждому элементу массива соответствует один или несколько *индексов*, определяющих положение элемента в массиве.

a

-5	0	12	54	-8
0	1	2	3	4

c

	0	1	2
0	-5	0	13
1	46	83	-8
2	54	0	93

b

A	N	D		O	R	...	T
0	1	2	3	4	5	...	255

Объявление массива:

<Тип элемента> <Имя>[<Размер1>] [<Размер2>] ...[= {<Список значений >}];

Количество индексов задает ***размерность*** массива.

Тип индекса – порядковый – определяет доступ к элементу.

Нумерация индексов **ВСЕГДА** начинается с 0.

Размер – определяет количество элементов по данному индексу.

Тип элемента – любой кроме файла, в том числе, другой массив.

Массив в памяти не может занимать более 2 Гб.

3.1 Одномерные массивы

Одномерными называются массивы, в котором положение элемента в массиве определяется одним индексом.

Объявление одномерных массивов

Примеры определения одномерных массивов;

int a[10]; - массив на 10 целых чисел;// индекс меняется 0 - 9

float mas[20] – массив на 20 вещественных чисел;

char sim[8] – массив на 8 символов;

double massiv[30] – массив на 30 вещественных чисел двойной точности;

unsigned int koord[10] – массив целых беззнаковых чисел .

Индекс меняется от 0 до величины, на 1 меньшей указанной в размере

Инициализация массива при объявлении

```
int a[5]={0,-36,78,3789,50};
float b[10]={0,-3.6,7.8,3.789,5.0,6.1,0,-6.5,8.9,3.0};
long double c[4]={7.89L,6.98L,0.5L,56.8L};
```

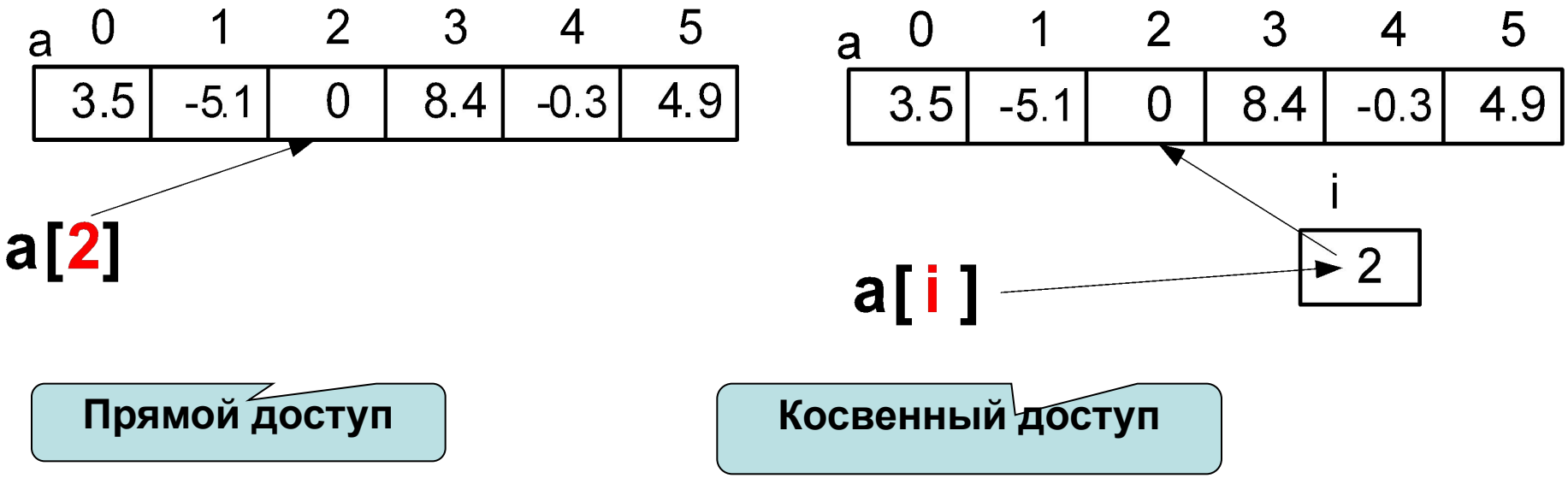
Операции над одномерными массивами

1. Доступ к элементу массива:

Пример:

```
int a[5],I;
    ...
    a[0]=51; {прямой доступ}
    ...
    I=3;
    a[I]=3; {косвенный доступ: значения индексов
             находятся в переменных}
```

Косвенный доступ к элементам массива



Косвенный доступ позволяет реализовать **последовательную обработку элементов массивов**:

```
for(i=0;i<6;i++)a[i]=i*i;
```

или

```
for(i=5;i>=0;i--) a[i]=i*i;
```

Операции над массивами (2)

2. Ввод массивов.

Осуществляется поэлементно:

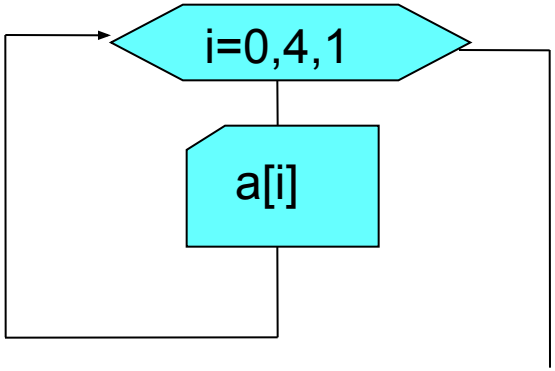
Пример 1. Ввод элементов одномерного массива

```
int a[5]; //массив на 5 целых чисел
   ...
for(i=0;i<5;i++)scanf("%d",&a[i]);
printf("\n"); // вводит последнее Enter и очищает буфер
```

Значения вводятся через пробел, Tab(→) или Enter(↵):

а) 2 -6 8 56 34 ↵

б) 2 ↵
-6 → 8 ↵
56 ↵
34 ↵



Операции над массивами (3)

3. Вывод массива

Также осуществляется поэлементно.

0 6

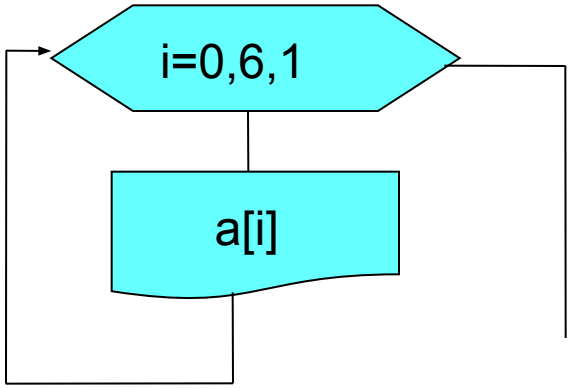
```
int b[7]={-3,5,8,-45,0,-1,8};
...
for(j=0;j<7;j++)
  printf("%4d"a[j]);
```

{ a_0 a_1 a_2 a_3 a_4 a_5 a_6}

```
  printf("\n"); {переходим на следующую строку}
...
```

На экране:

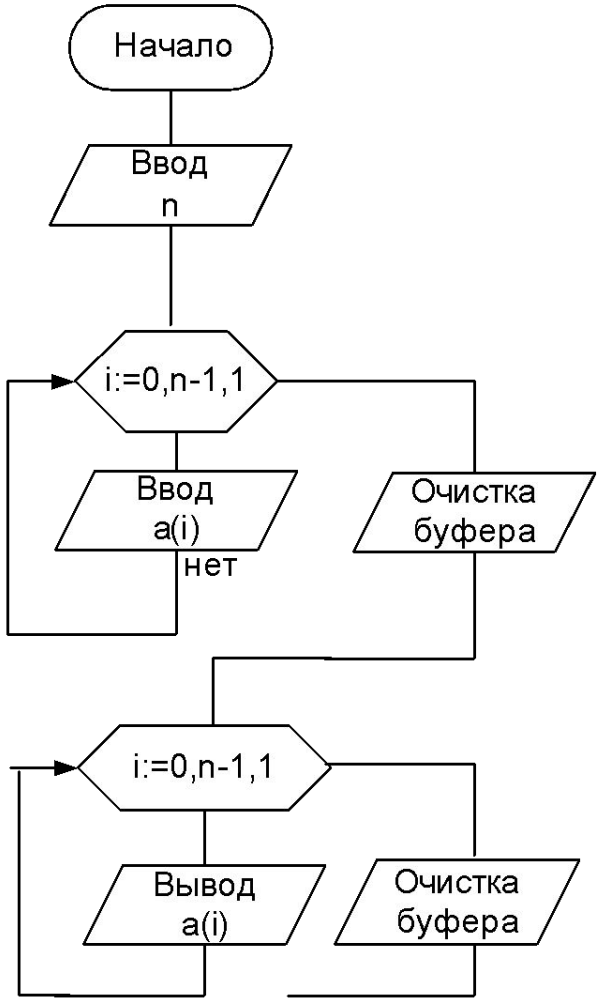
UU-3UUU5UUU8U-45UUU0UU-1UUU8



Пример программы с вводом выводом

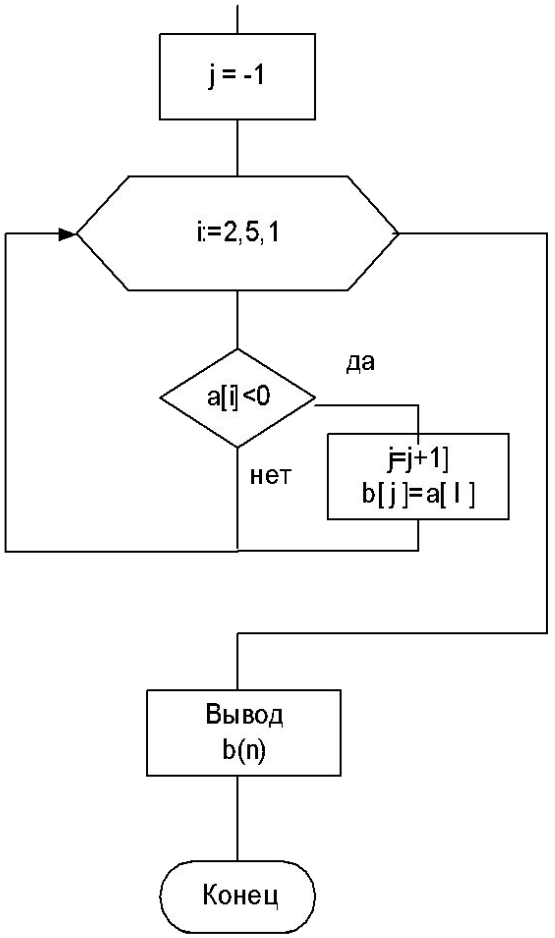
Написать программу формирования массива b из отрицательных элементов массива a

```
// Ex3_1
#include "stdafx.h"
#include <stdio.h>
int a[8],b[8];
int i,j,n;
int main(int argc, char* argv[])
{
puts("Input n<=8");
 scanf("%d",&n);
 printf("input %3d  elementov \n",n);
 for(i=0;i<n;i++) scanf("%d",&a[i]);
    printf("\n");
    puts("Inputed Massiv");
for(i=0;i<n;i++)printf("%4d",a[i]);
    printf("\n");
```



Пример программы с вводом выводом(2)

```
j=-1;
for(i=0;i<n;i++)
   if(a[i]<0)
    { j++;
      b[j]=a[i];
    }
 if (j==-1) puts("Massiv  b  empty");
 else
  {
  puts("New  Massiv");
      for(i=0;i<=j;i++)printf("%4d",b[i]);
      printf("\n");
 }
    return 0;
}
```



3.2 Основные приемы программирования обработки одномерных массивов

Все задачи по работе с массивами можно разбить на следующие группы:

1. Однотипная обработка массивов.
2. Переформирование массивов.
3. Одновременная обработка нескольких массивов и/или подмассивов.
4. Поисковые задачи.

3.2.1 Однотипная обработка массивов

а) **Поэлементная** (нахождение суммы элементов, произведения элементов, среднего арифметического, среднего геометрического, подсчет количества элементов, отвечающих определенному условию или обладающих некоторыми признаками, а также их суммы, произведения и т.д.).

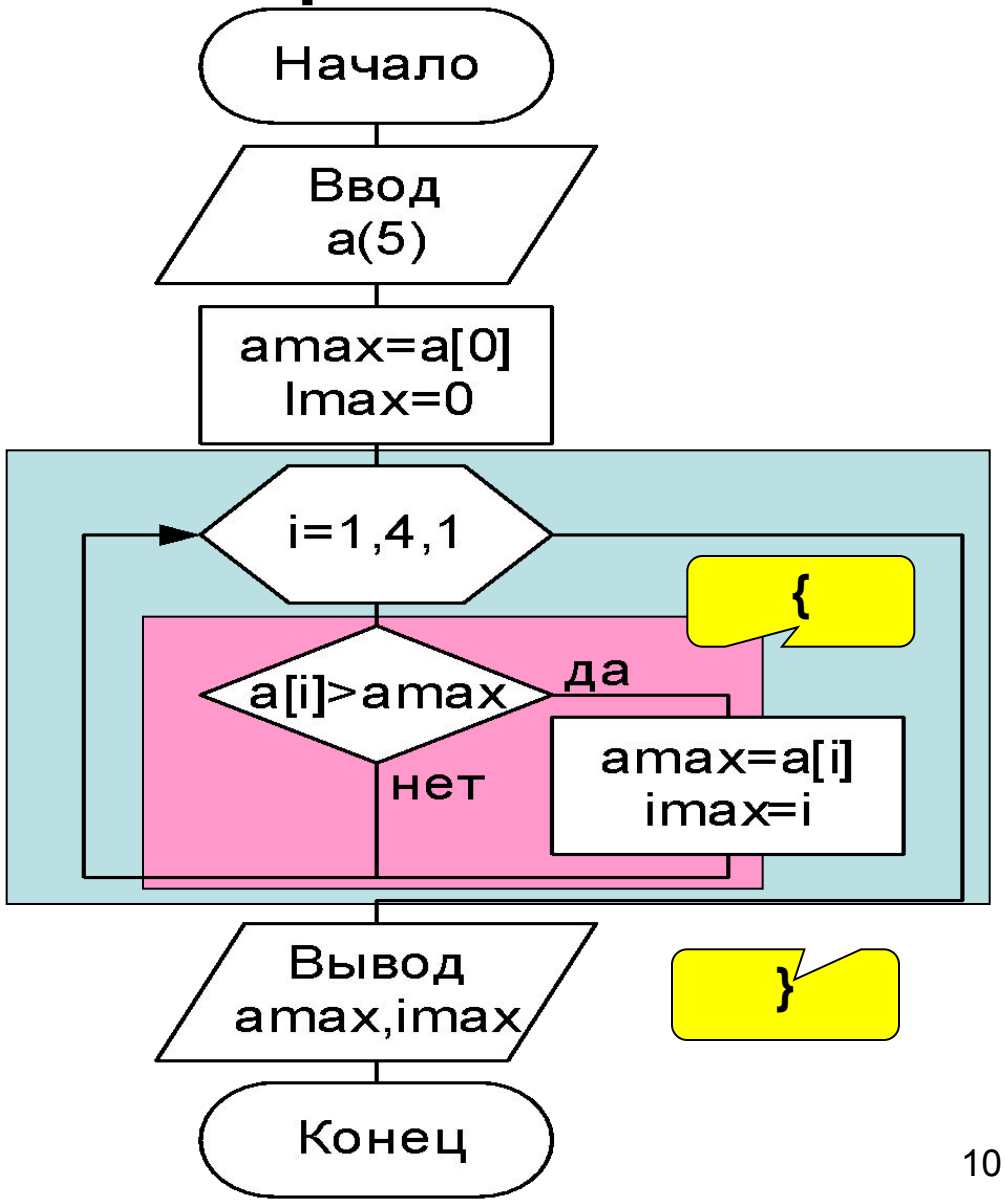
Пример. Написать программу определения максимального элемента массива и его положения в массиве.

Определить максимальный элемент массива и его номер

A

45	34	56	2	-3

AMAX	IMAX	i
56	2	4



Программа определения максимального элемента массива и его номера

```
//Ex3_2;
#include “stdafx.h”
#include  <stdio.h>
int main(int argc, char* argv[])
{float a[5], amax; int i, imax;
 puts(“Input 5 values:”);
    for(i=0;i<5;i++)scanf(“%f “,&a[i]);printf(“\n”);
    amax=a[0];
    imax=0;
    for(i=1;i<5;i++)
        if(a[i]>amax)
            { amax=a[i];  imax=i;}
    puts(“Values:“);
    for(i=0;i<5;i++)printf(“%7.2f ”,a[i]);printf(“\n”);
    printf(“Max = %7.2f  number = %5d\n”,amax, imax);
 return 0; }
```

Однотипная обработка массивов(2)

b) **Выборочная** (задачи по формулировке сходные с задачами предыдущего типа, но операция выполняется не надо всеми элементами массива, а только теми, которые имеют вполне определенное значение индексов).

Особенностью таких задач является наличие определенного закона изменения индексов рассматриваемых элементов**.**

Пример .

Написать программу, определяющую количество отрицательных элементов среди элементов одномерного массива целых чисел, стоящих на четных местах.

int A[7]

45	-34	12	-25	16	11	10
0	1	2	3	4	5	6

1.Счетный цикл

а) i = 1 – n/2 с шагом 1

N_{el}=i*2-1

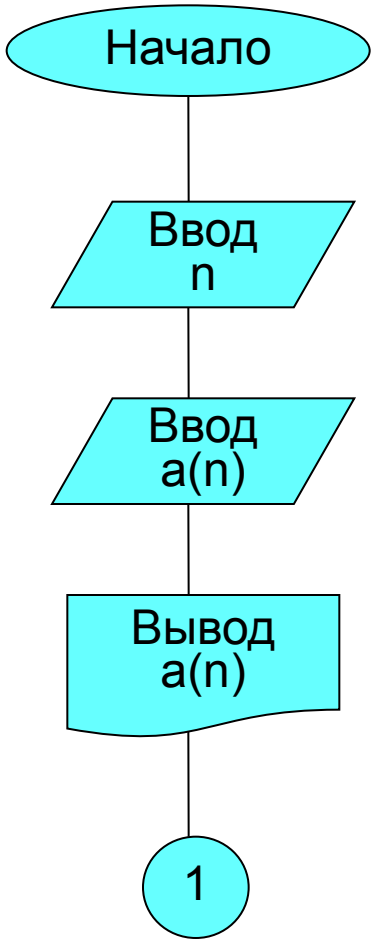
б) i = 1 – n с шагом 2

N_{el}=i

2. Цикл с постусловием

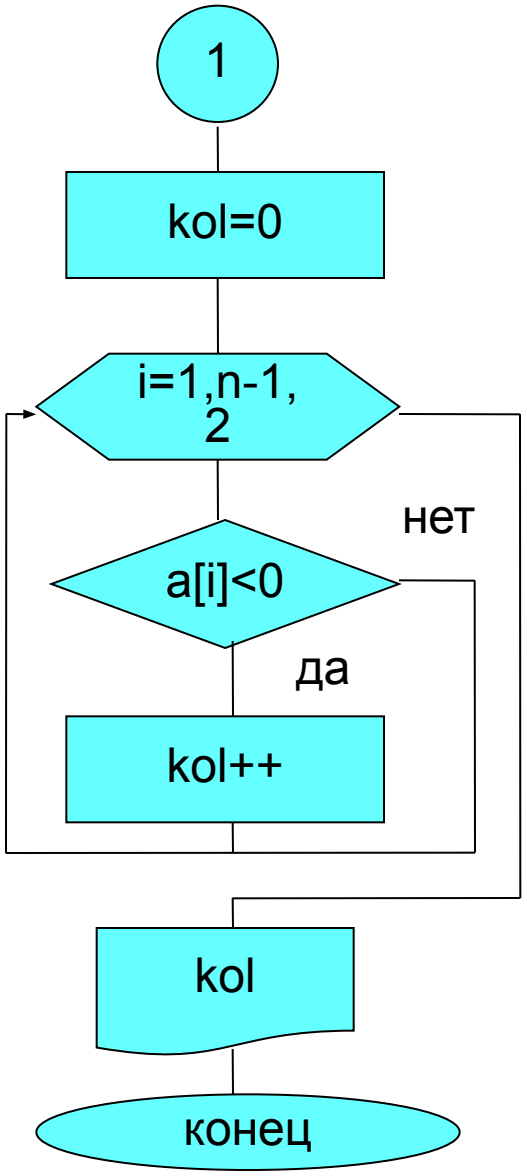
Программа определения количества отрицательных элементов, стоящих на четных местах

```
// Ex3_3.cpp
#include "stdafx.h"
#include <stdio.h>
int a[7];
int i,kol,n;
int main(int argc, char* argv[])
{ puts("Input n<=7");
 scanf("%d",&n);
 printf("input %3d  elementov massiva\n",n);
 for(i=0;i<n;i++)scanf("%d",&a[i]);
    printf("\n");
puts("Inputed Massiv");
for(i=0;i<n;i++)printf("%4d",a[i]);
    printf("\n");
```



Продолжение программы

```
kol=0;
for(i=1;i<n;i=i+2)
  if(a[i]<0) kol++;
printf("V massive  %4d otricatelnyx",kol);
printf("elementov na chetnyx mectax\n");
    return 0;
}
```

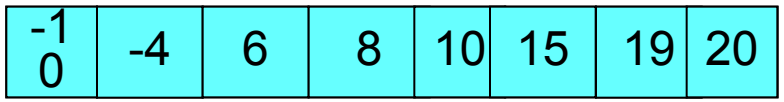


3.2.2 Переформирование массива.

– ***Переформирование массива без изменения его размеров*** (перестановки элементов различного характера и сортировки).

Пример. Написать программу упорядочивания массива по возрастанию его элементов.

int A[8]



0 1 2 3 4 5 6 7

1. Находим минимальный элемент и его номер
2. Меняем его местами с 0
3. В оставшейся чисти массива находим минимальный элемент
4. Меняем его с 1

Далее действие 3 и 4 повторяем, пока не закончится массив

Переформирование массива (2)

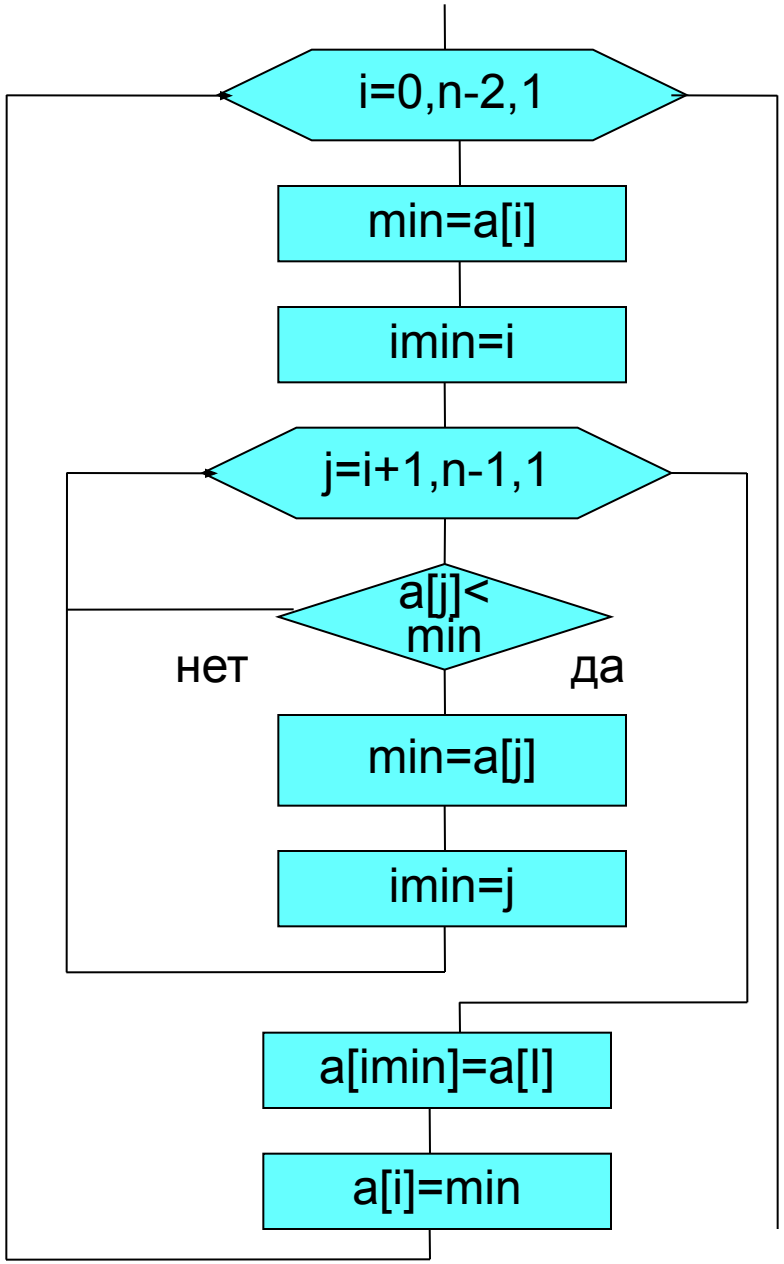
```
// Ex3_4.cpp

#include "stdafx.h"
#include <stdio.h>

int a[8];
int i,j,imin,min,n;
int main(int argc, char* argv[])
{puts("Input n<=8");
 scanf("%d",&n);
 printf("input %3d  elem. massiva\n",n);
 for(i=0;i<n;i++)scanf("%d",&a[i]);
   printf("\n");
 puts("Inputed Massiv");
 for(i=0;i<n;i++)printf("%4d",a[i]);
    printf("\n");
```

Переформирование массивов (3)

```
for(i=0;i<n-1;i++)
 {   min=a[i];
     imin=i;
     for(j=i+1;j<n;j++)
      if (a[j]<min)
      {min=a[j];
       imin=j;
      }
      a[imin]=a[i];
      a[i]=min;
 }
  puts("Sorted Massiv");
for(i=0;i<n;i++)printf("%4d",a[i]);
    printf("\n");
    return 0;
}
```



Переформирование массивов (4)

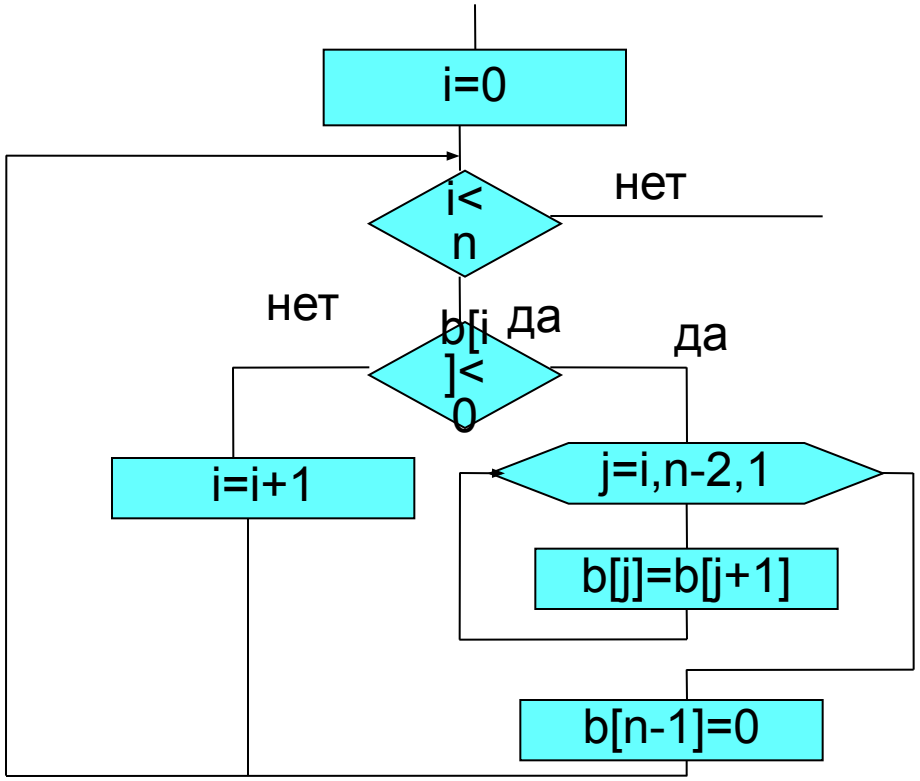
b) Переформирование массива с изменением его размеров *(*вычеркивание и вставка элементов, отвечающих определенным условиям или обладающих заданными признаками).

Пример. Дан одномерный массив. Вычеркнуть из него все отрицательные элементы. Есть 2 варианта решения.

1.

int B[6]

10	6	2	0	0	0



Переформирование массивов (5)

2.

i=0 - 5

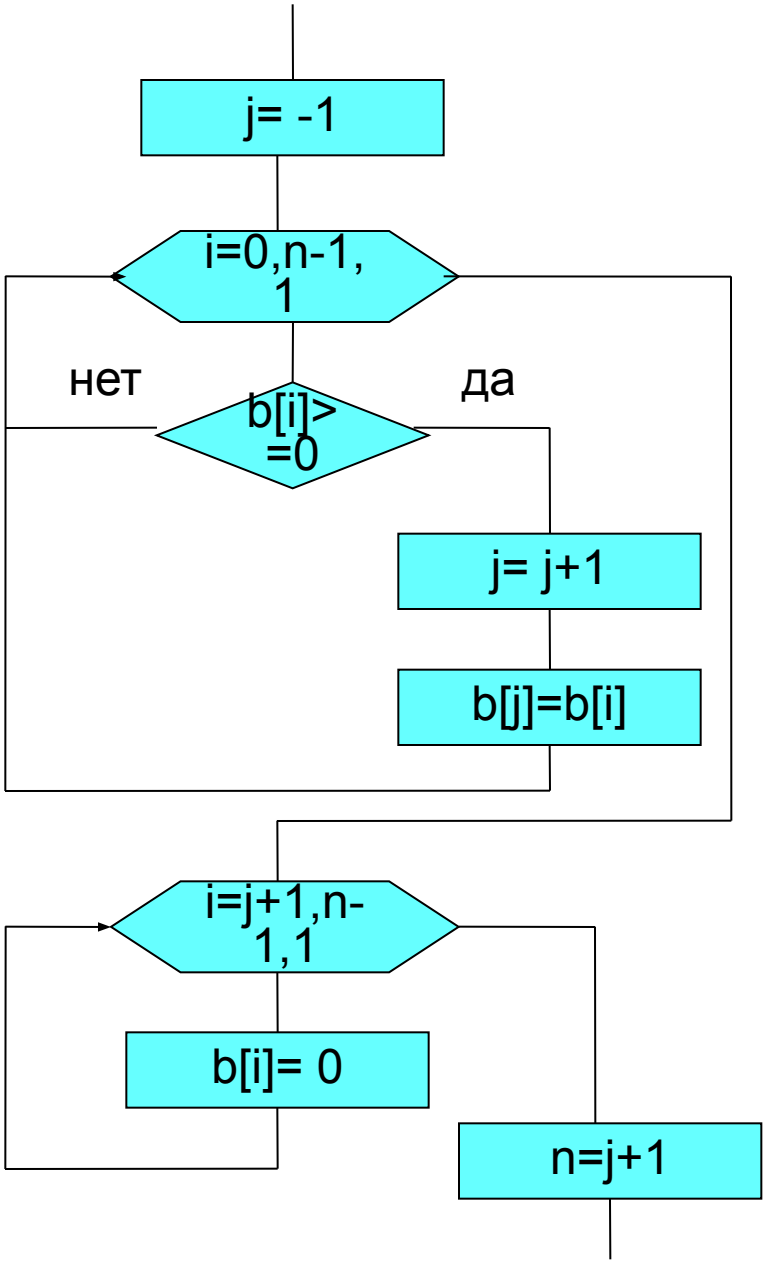
i=5

10	6	2	0	0	0

j= 3 -5

n=3

```
Ex3_5
#include "stdafx.h"
#include <stdio.h>
int b[6];
int i,j,n;
int main(int argc, char* argv[])
 {puts("Input n<=6");
 scanf("%d",&n);
 printf("input %3d  elem. massiva\n",n);
 for(i=0;i<n;i++)scanf("%d",&b[i]);
    printf("\n");
```



Переформирование массивов (6)

```
puts("Inputed Massiv");
 for(i=0;i<n;i++)printf("%4d",b[i]);
    printf("\n");
j=-1;
for(i=0;i<n;i++)  // перезапись неотрицательных элементов
    if (b[i]>=0)
    { j=j+1;
      b[j]=b[i];
    }
 for(i=j+1;i<n;i++)  //  обнуление оставшихся элементов
        b[i]=0;
     n=j+1;    // новый размер массива

  puts("New  Massiv");
  for(i=0;i<n;i++)printf("%4d",b[i]);
    printf("\n");
    return 0;
}
```

3.2.3 Одновременная обработка массивов

а) Синхронная обработка нескольких массивов или подмассивов

Пример. Дан массив целых чисел, содержащий четное количество элементов. Определить, является ли вторая половина массива, копией первой.

int s[8]

key=true

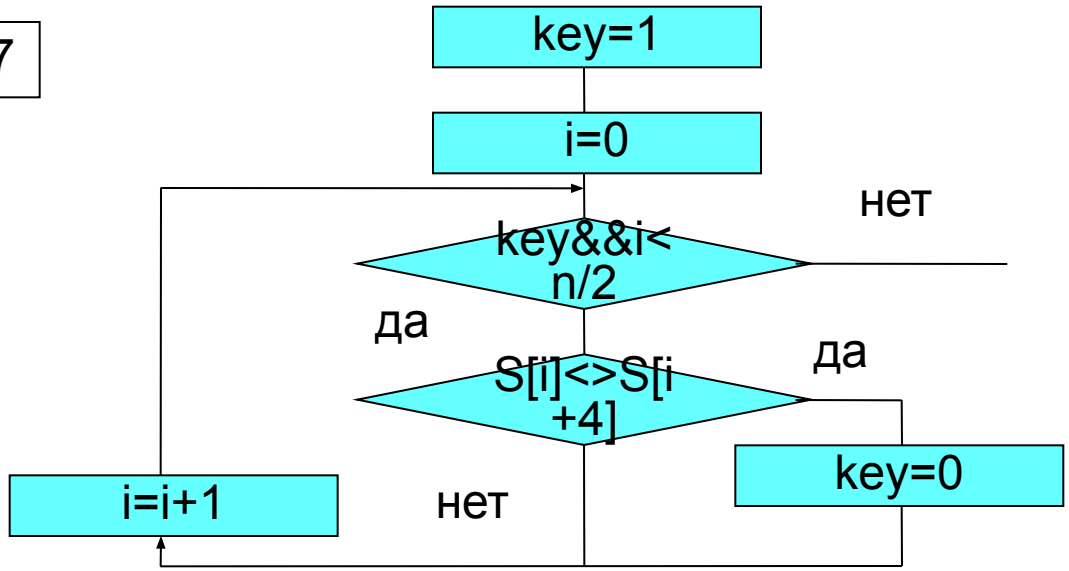
1	8	3	9	1	8	3	9
0	1	2	3	4	5	6	7

i= 0 - 3 j= 4 - 7

j=i+4

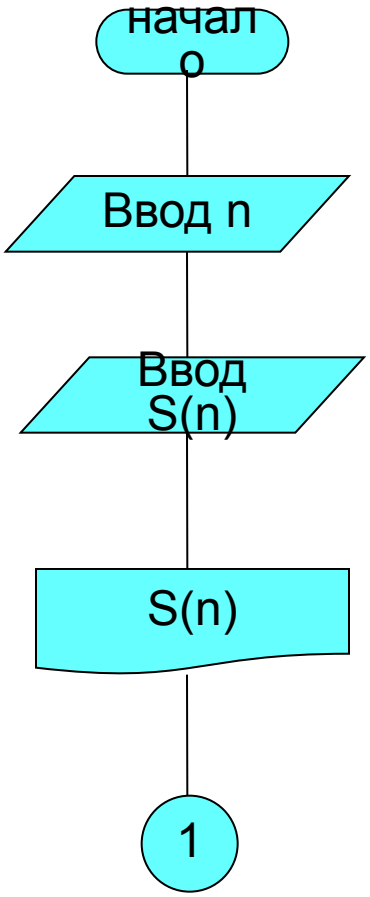
key=fals e

5	6	4	2	6	4



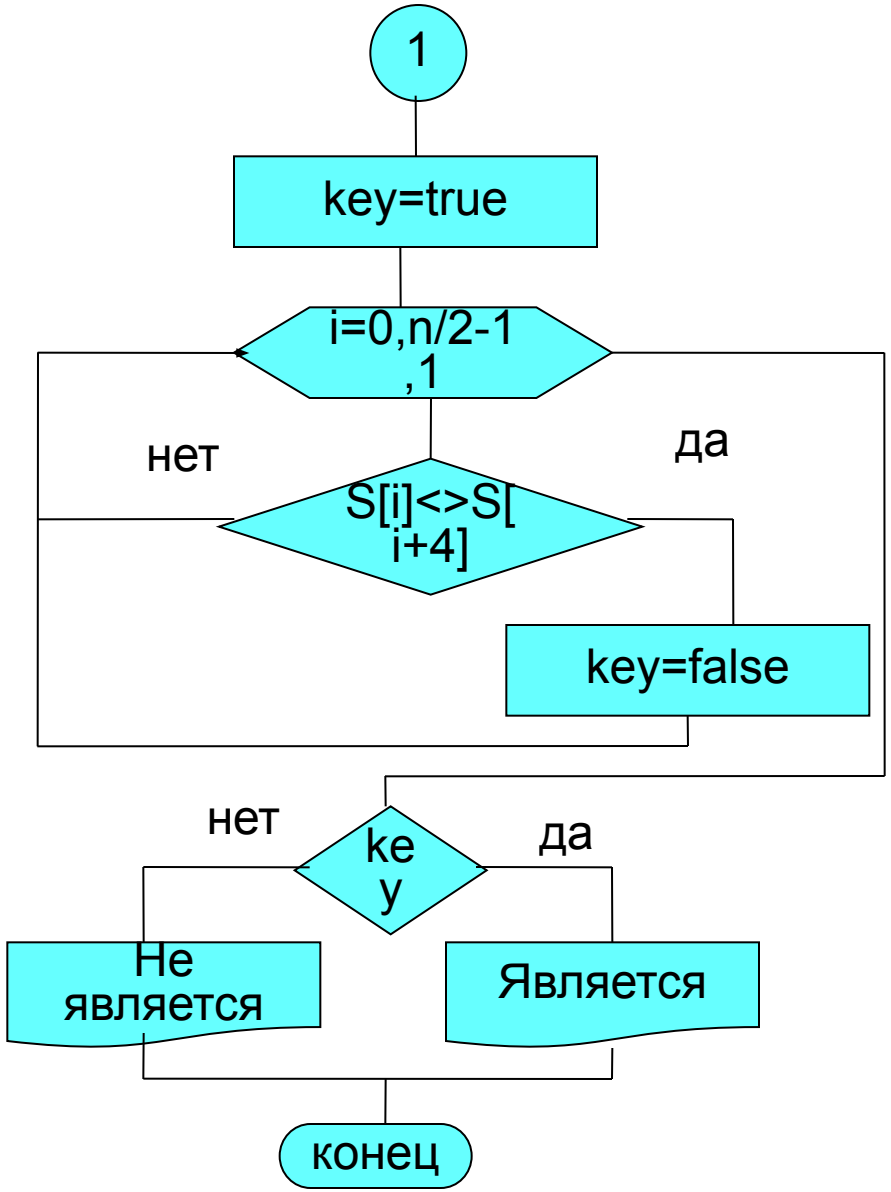
Синхронная обработка нескольких массивов или подмассивов

```
//Ex3_6
#include "stdafx.h"
#include <stdio.h>
int main(int argc, char* argv[])
{int s[8];
 int i,n,key;
 puts("Input kol.elementov chetnoe n<=8");
 scanf("%d",&n);
 printf("Input %4d elem \n",n);
 for(i=0;i<n;i++)
    scanf("%d",&s[i]);
 puts("Inputed Massiv");
 for(i=0;i<n;i++)
     printf("%3d",s[i]);
 printf("\n");
```



Продолжение программы

```
key=1;
 for(i=0;i<n/2;i++)
     if(s[i]!=s[i+4])key=0;
 if (key)
puts("Podmassivy ravny");
else
 puts("Podmassivy not ravny");
    return 0;
}
```

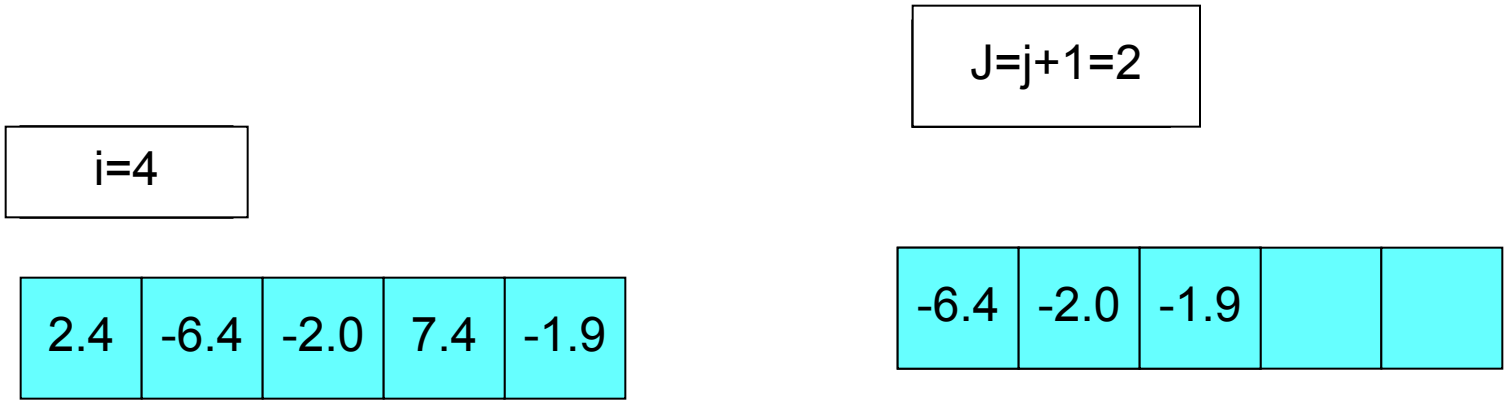


Одновременная обработка массивов

б) **Асинхронная обработка массивов**

Пример. Дан массив A вещественных чисел. Переписать в массив C все отрицательные элементы массива A.

float A[5],C[5]



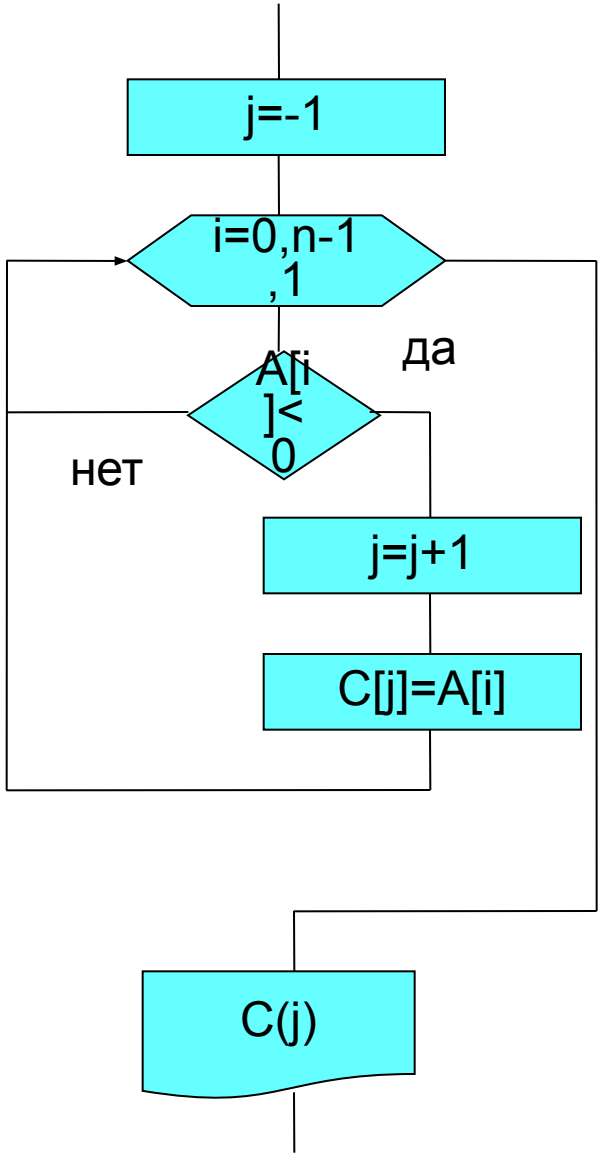
Асинхронная обработка массивов(2)

```
// Ex3_7.cpp
#include "stdafx.h"
#include <stdio.h>

int main(int argc, char* argv[])
{ float A[5],C[5];
 int i,j;
  printf("Input 5 elem \n");
 for(i=0;i<5;i++)
   scanf("%f",&A[i]);
 puts("Inputed Massiv  A");
 for(i=0;i<5;i++)
    printf("%5.2f",A[i]);
 printf("\n");
```

Асинхронная обработка массивов(3)

```
 j=-1;
 for(i=0;i<5;i++)
   if(A[i]<0)
    {j++;
     C[j]=A[i];}
 puts("New  Massiv  C");
 for(i=0;i<=j;i++)
   printf("%5.2f",C[i]);
 printf("\n");
 return 0;
}
```



3.2.4 Поисковые задачи

Пример:

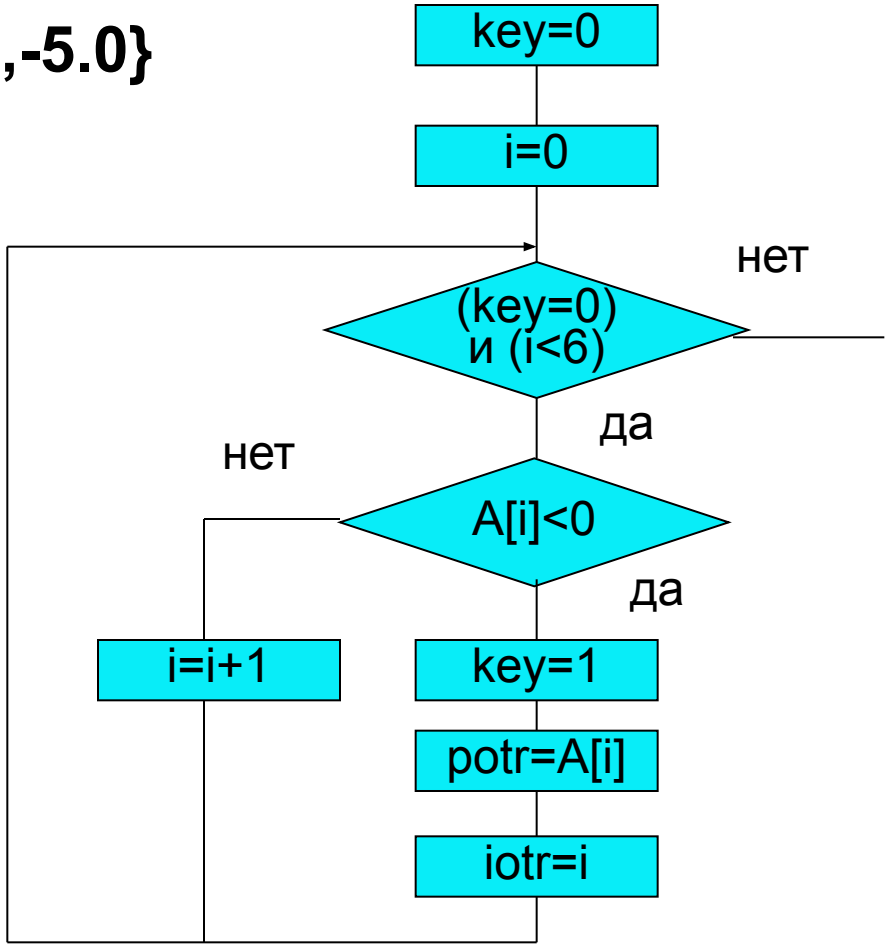
Дан массив A вещественных чисел. Определить первый отрицательный элемент массива и его номер.

float A[6] ={2.6,4.7,-5.8,-4.0,7.1,-5.0}

i=2

2.6	4.7	-5.8	-4.7	7.1	-5.0

key=1 iotr=2 potr=-5.6



Поисковые задачи (2)

```
// Ex3_8.cpp
#include "stdafx.h"
#include <stdio.h>

int main(int argc, char* argv[])
{ float potr,A[6]={2.6,4.7,-5.8,-4.0,7.1,-5.0};
  int i,iotr,key;
 puts("Inputed Massiv  A");
 for(i=0;i<6;i++)
     printf("%6.2f",A[i]);
 printf("\n");
  i=0;
  key=0;
  while((key==0)&&(i<6))
     if(A[i]<0)
          {key=1;potr=A[i];iotr=i;}
     else i++;
  printf("potr=  %6.2f  iotr= %4d\n",potr,iotr);
    return 0;
}
```

3.3 Обработка матриц

Двумерными называются массивы, имеющие два индекса. По аналогии с математикой, иногда такие массивы называют матрицами.

Для простоты изложения в дальнейшем будем придерживаться именно этой терминологии.

Описание матриц

int a[4][5] – матрица целого типа из 4 строк и 5 столбцов

индексы меняются первый от 0 до 3, второй от 0 до 4

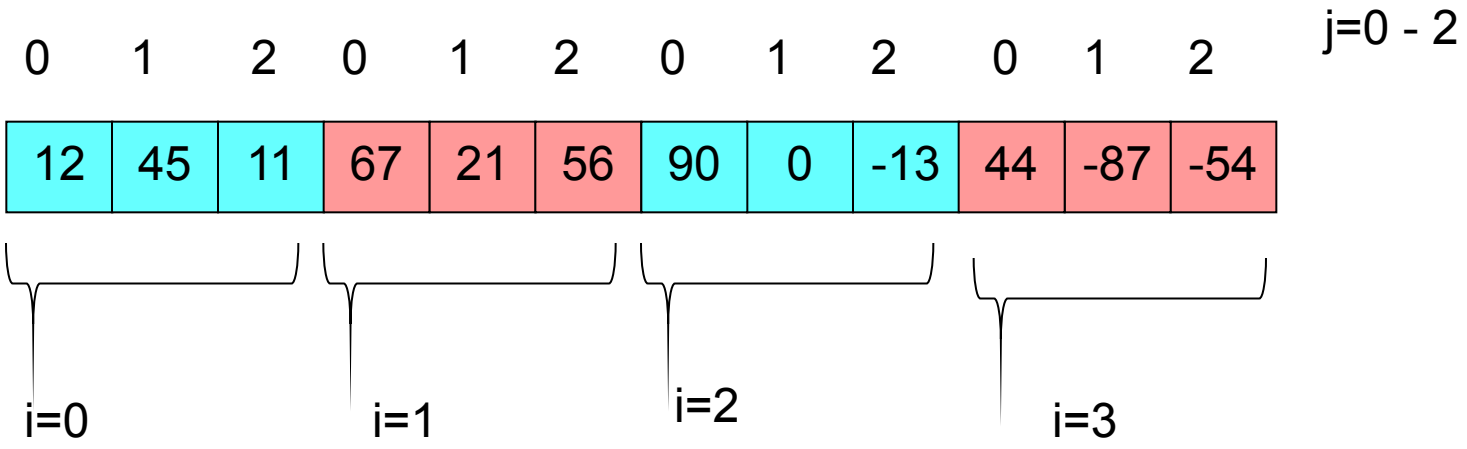
float matr[10][20] – матрица вещественного типа из 10 строк и 20 столбцов

double x[10][10] - матрица вещественного типа с двойной точностью из 10 строк и 10 столбцов

В памяти матрицы располагаются по строкам. Быстрее изменяется второй индекс

Расположение матрицы в памяти

int A[4][3]



Инициализация матриц при объявлении

short x[3][4] ={{9,6,-56,0}, {10,3,-4,78}, {-6,8,45,7}};

int A[4][3]={{12,45,11},{67,21,56},{90,0,-13},{44,-87,-54}};

Операции над матрицами

1. **Доступ к элементам матрицы**

Пример:

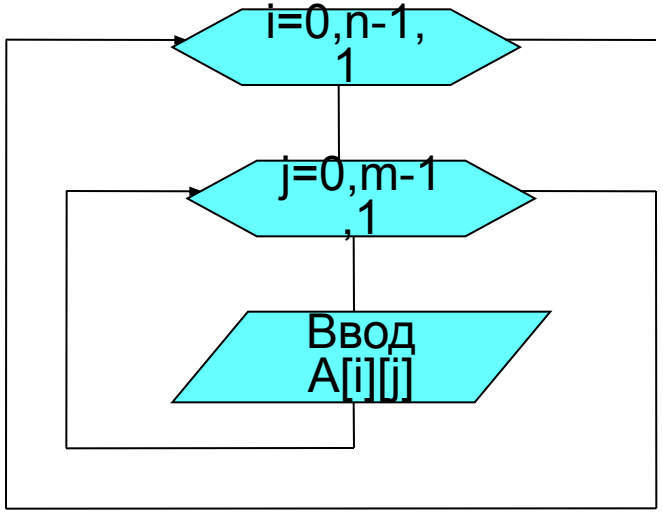
```
int a[5][4],i,j;
    ...
    a[0][1]=5.1; {прямой доступ}
    ...
    i=3;j=3
    a[i][j]:=23; {косвенный доступ: значения индексов
                  находятся в переменных}
```

Операции над матрицами (2)

2. **Ввод матриц**

Ouществляется поэлементно, по строкам:

Пример 1. Ввод элементов матрицы




```
const int n=3;
const int m=4;
float A[n][m];
printf("Input %3d  strok po %3d elem.\n");
for(int i=0;i<n;i++)
  for(int j=0;j<m;j++)
     scanf("%f",&A[i][j]);
```

Значения вводятся через пробел, Tab(->) или Enter(↵):

2 -6 8 23↵
56 9 0 -7↵
6 12 -56 -8↵

Операции над матрицами (3)

3. Вывод матриц

Осуществляется поэлементно, по строкам или по столбцам, в зависимости от требования программы

Пример 1. Вывод элементов матрицы

….

puts(“ MASSIV”);

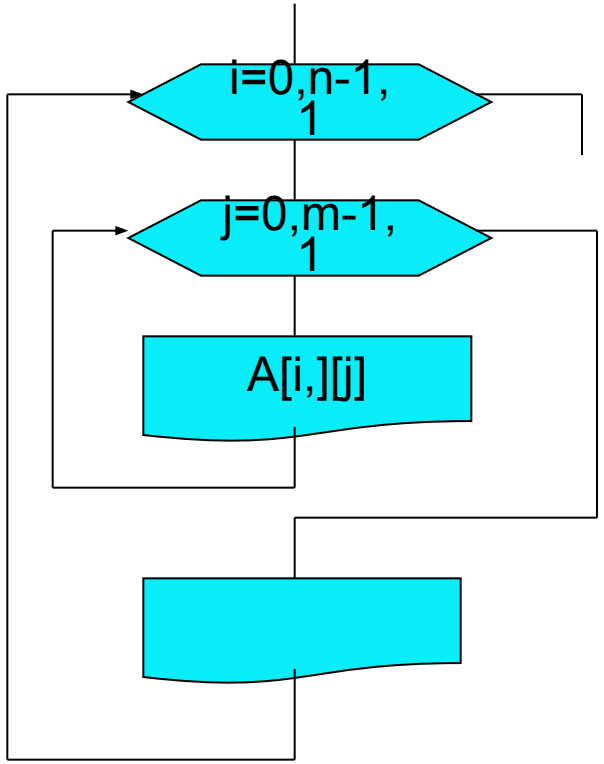
```
for(int i=0;i<n;i++){
  for(int j=0;j<m;j++)
     printf(“%7.2f”,A[i][j]);
 printf(“\n”);}
```

Информация на экране:

UUU**2.00**UU**−6.00**UUU**8.00** U**U23.00**

UU**56.00**UUU**9.00**UUU**0.00**UU**−7.00**

UUU**6.00**UU**12.00**U**−56.00**UU**−8.00**



3.3.1 Особенности программирования обработки матриц

1. При обработке матриц используются вложенные циклы.
2. Обработка матриц может производиться как по строкам, так и по столбцам.
3. Так как матрица расположена в памяти по строкам, второй индекс меняется быстрее. Поэтому при обработке по строкам, внешний цикл индексирует строки, а внутренний столбцы.

```
for(i=0;i<n:i++)
   for(j=0;j<m;j++)
      {…обработка элемента A[ i ] [ j ]...}
```

4. При необходимости обойти матрицу по столбцам, достаточно изменить последовательность выполнения циклов.

```
for(j=0;j<m:j++)
   for(i=0;i<n;i++)
      {…обработка элемента A[ i ] [ j ]..}
```

В этих примерах i – номер элемента в строке

j - номер элемента в столбце

Поэлементная обработка матрицы

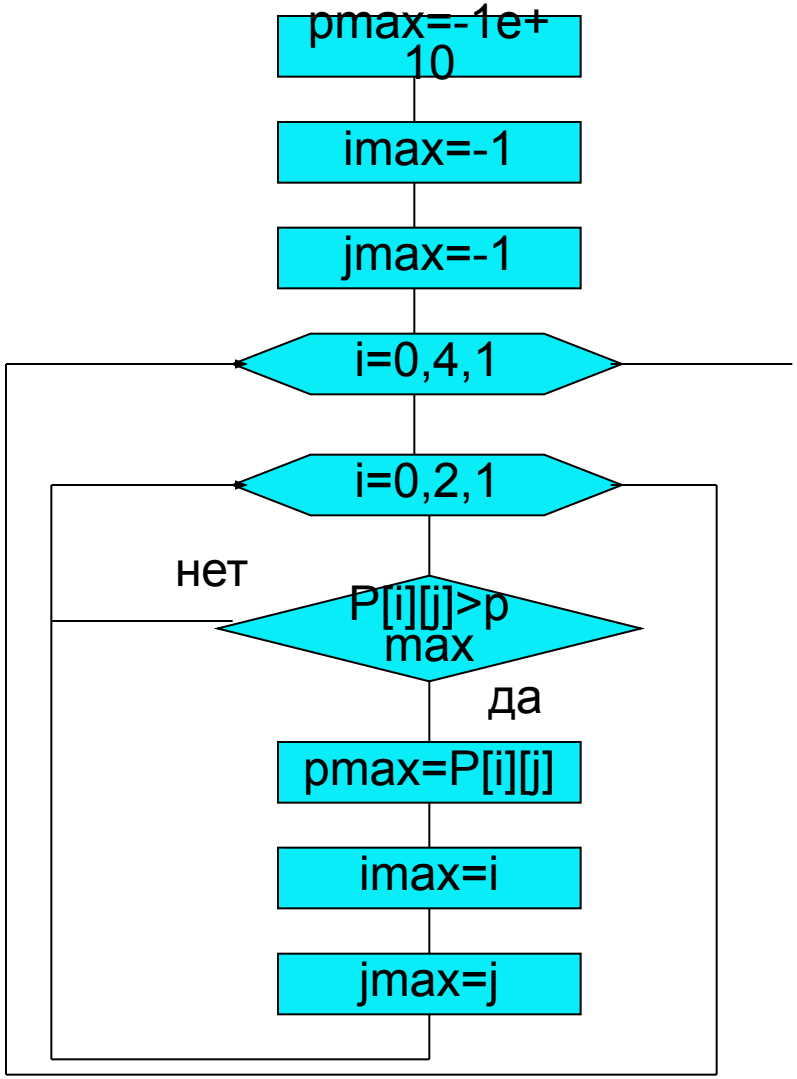
Пример.

Дана матрица вещественного типа. Определить максимальный элемент матрицы и его координаты в матрице.

float P[3][5];

	0	1	2	3	4
0	2.45	17.5	-12.4	20.25	-0.45
1	45.0	-55.1	12.4	21.5	72.0
2	2.45	2.45	2.45	2.45	2.45

pmax=72.0	imax=1	jmax=4



Поэлементная обработка матрицы (2)

```
// Ex3_10.cpp
#include "stdafx.h"
#include <stdio.h>
#include <conio.h>
int main(int argc, char* argv[])
{   float P[3][5],pmax;
int i,j,imax,jmax;
    for(i=0;i<3;i++)
     { printf("Input numbers of %2d
     string:\n",i);
       for (j=0;j<5;j++)
     scanf("%f",&P[i][j]);
     }
       puts("      MATRICA  ");
     for(i=0;i<3;i++)
     { for (j=0;j<5;j++)
     printf("%7.2f ",P[i][j]);
       printf("\n");
     }
```

```
pmax=-1e+6;
imax=-1;
jmax=-1;
      for(i=0;i<3;i++)
        for(j=0;j<5;j++)
            if (P[i][j]>pmax)
            { pmax=P[i][j];
              imax=i;
              jmax=j;
            }

printf("Max Eem. = %7.2f",pmax);
printf("  imax=  %4d”,imax+1);
printf(“  jmax=%4d\n",jmax+1);

getch();

      return 0;
}
```

Поэлементная обработка матрицы (2)

Пример.

Дана вещественная матрица. Определить номер строки, содержащей самую большую сумму элементов.

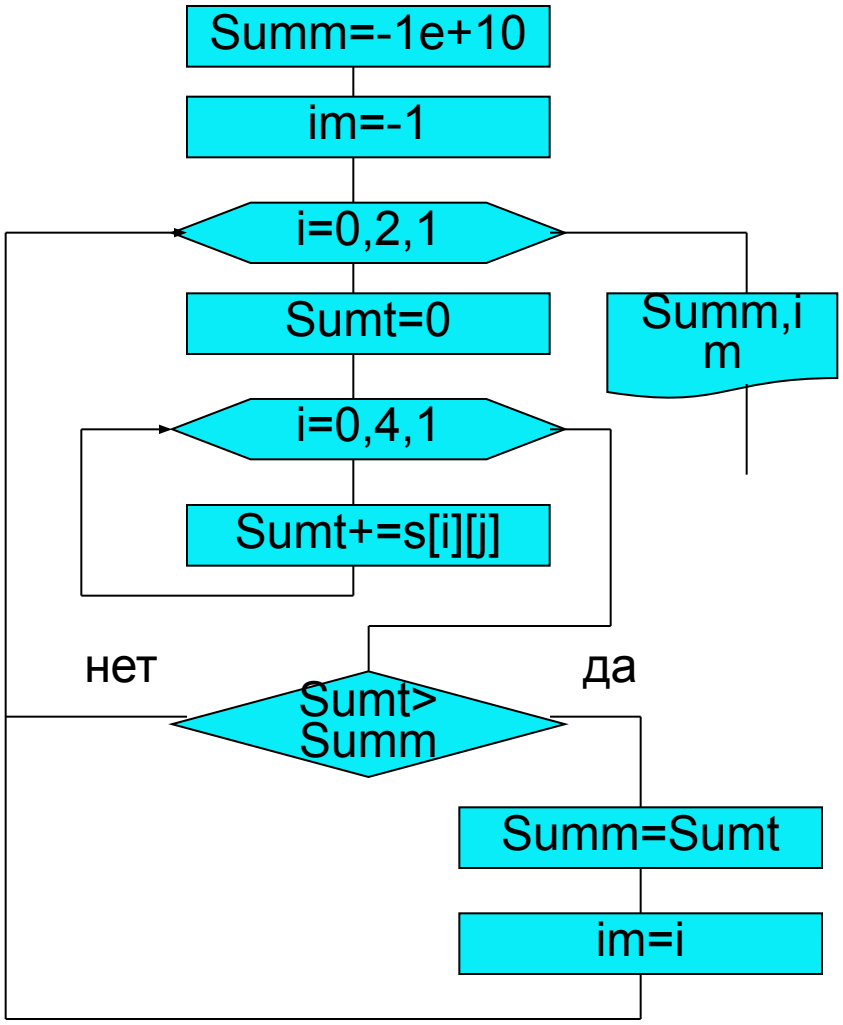
float s[3][5] ;

Summ=95.8

im=1

	0	1	2	3	4
0	2.45	17.5	-2.4	20.25	-0.45
1	45.0	-55.1	12.4	21.5	72.0
2	2.45	2.45	2.45	2.45	2.45

Sumt=12.25



Поэлементная обработка матрицы(3)

```
// Ex3_11.cpp
#include "stdafx.h"
#include <stdio.h>
#include <conio.h>
int main(int argc, char* argv[])
{   float s[3][5],Summ,Sumt;
int i,j,im;
for(i=0;i<3;i++)
 { printf("Input numbers of %2d string:\n",i+1);
   for (j=0;j<5;j++) scanf("%f",&s[i][j]);
  }
puts("      MATRICA  ");
for(i=0;i<3;i++)
  { for (j=0;j<5;j++)
     printf("%7.2f ",s[i][j]);
     printf("\n");
  }
```

```
Summ=-1e+6;
im=-1;
for(i=0;i<3;i++)
 {  Sumt=0;
    for(j=0;j<5;j++)
      Sumt+=s[i][j];
    if (Sumt>Summ)
      { Summ=Sumt;
        im=i;
       }
  }
printf("Max Sum Elem. =“);
printf(“%7.2f",Summ);
printf("  im=  %4d \n",im+1);

      getch();

      return 0;
}
```

Выборочная обработка матрицы

Пример. Дана целочисленная матрица. Определить среди четных строк, строку, имеющую наибольшее среднее арифметическое ее элементов.

int C[4][3]

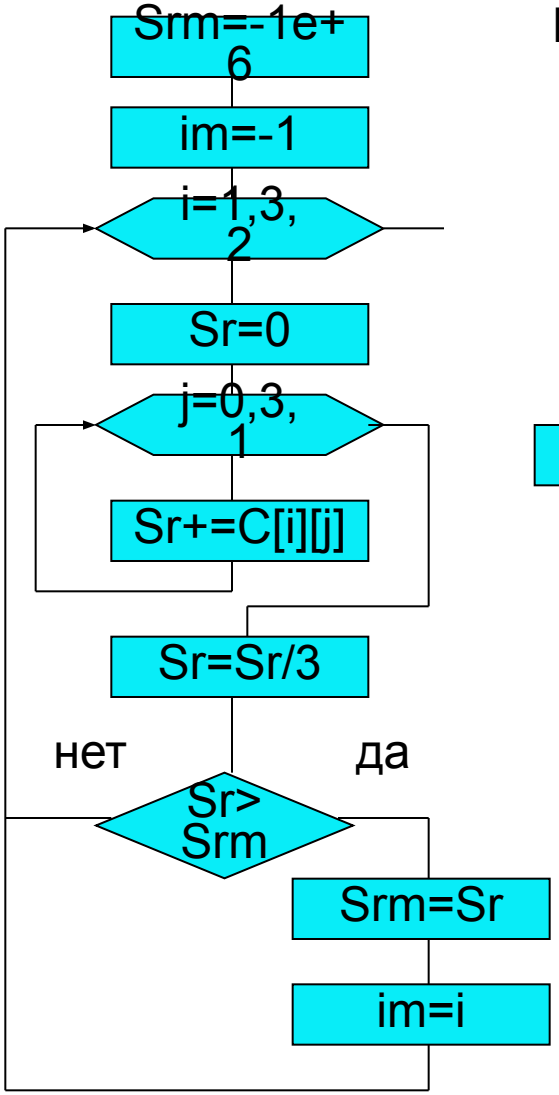
Srm=6

im=3

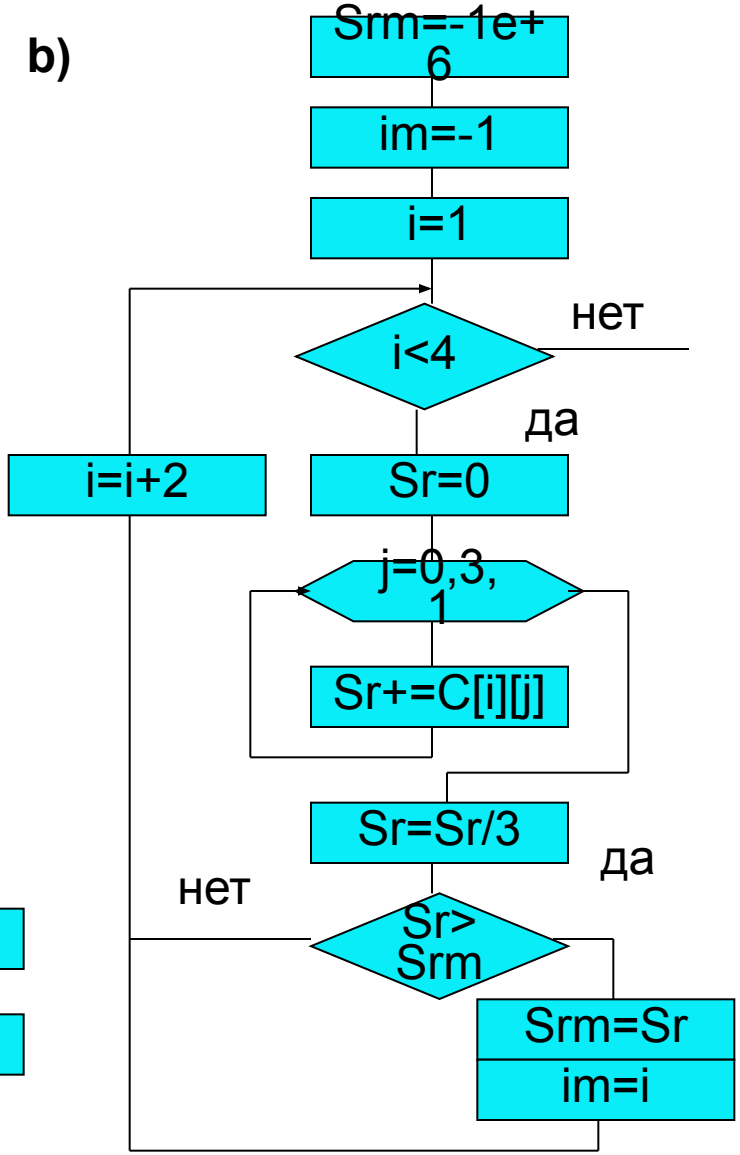
	0	1	2
0	2	11	-41
1	-12	21	6
2	17	-8	32
3	43	-31	6

Sr=6

a)



b)



Выборочная обработка матрицы(2)

```
// Ex3_12.cpp
#include "stdafx.h"
#include <stdio.h>
#include <conio.h>
int main(int argc, char*
  argv[])
{   int C[4][3]; float Srm,Sr;
int i,j,im;
for(i=0;i<4;i++)
{printf("Input numbers of”);
  printf(“ %2d string:\n",i+1);
  for (j=0;j<3;j++)
  scanf("%d",&C[i][j]);
}
 puts("      MATRICA  ");
 for(i=0;i<4;i++)
{ for (j=0;j<3;j++)
     printf("%4d ",C[i][j]);
  printf("\n");
}
```

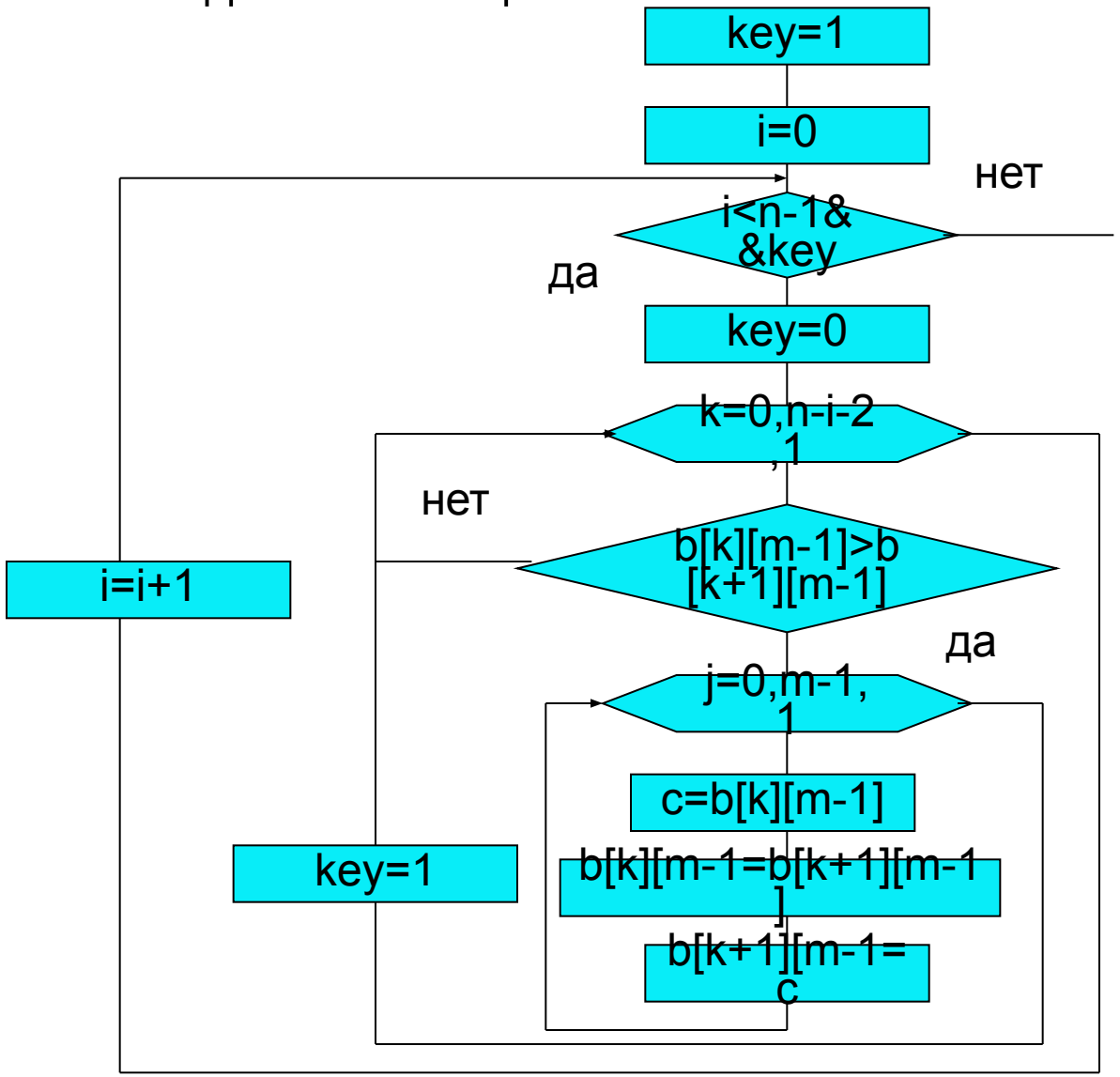
```
Srm=-1e+6;
im=-1;
for(i=1;i<4;i=i+2)
{  Sr=0;
    for(j=0;j<3;j++)
      Sr+=C[i][j];
    Sr=Sr/3;
    if (Sr>Srm)
    { Srm=Sr;
      im=i;
      }
}
printf("Max SR Sum Elem.= ");
 printf(" %7.3f",Srm);
 printf("  im=  %4d
   \n",im+1);
 getch();
 return 0;
}
```

Переформирование матрицы

Пример. Дана целочисленная матрица. Отсортировать ее по возрастанию элементов последнего столбца.

```
int  b[3][4]
```

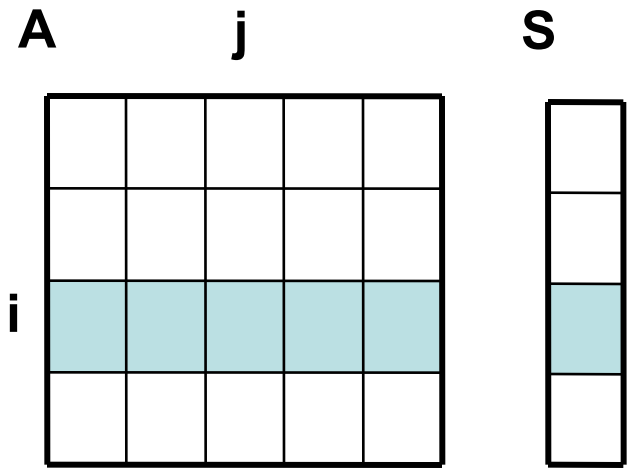
	0	1	2	3
0	15	9	11	5
1	4	6	22	12
2	10	23	11	34



Переформирование матрицы(2)

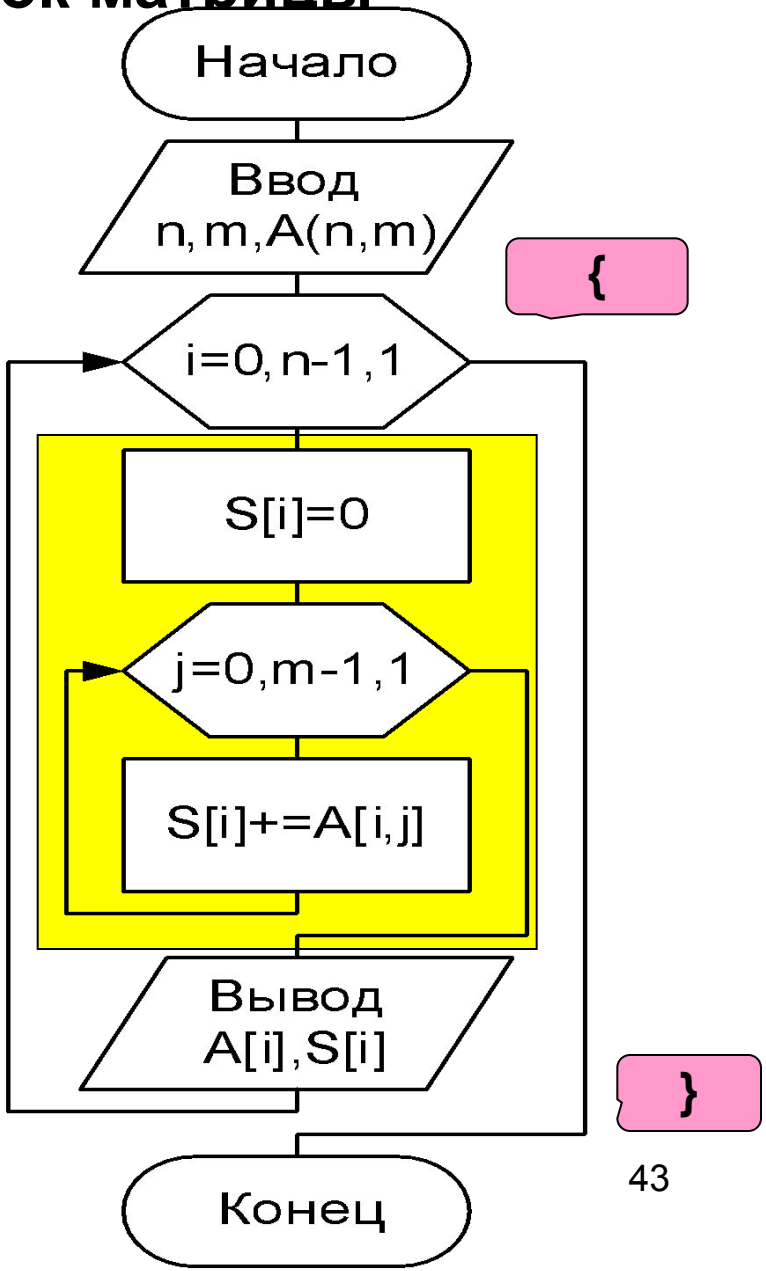
```
// Ex3_13.cpp
#include "stdafx.h"
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <conio.h>
int main(int argc, char* argv[])
{int n,mat[10][10],i,j,m,k,key,b;
srand( (unsigned)time( NULL ));
printf(" Inputed size of massiv n,m<=10  \n");
scanf("%d %d",&n,&m);
 for (i=0;i<n;i++)
  for (j=0;j<m;j++)
   mat[i][j]=rand()/1000-rand()/1000;
 printf("\n Inputed Massiv \n");
  for(i=0;i<n;i++)
   for(j=0;j<m;j++)
    printf("%4d%c",mat[i][j],(j==m-1)?'\n':' ');
key=1;
i=0;
while((i<n-1)&&(key==1))
{key=0;
 for(k=0;k<n-i-1;k++)
  if(mat[k][m-1]>mat[k+1][m-1])
  {  for(j=0;j<m;j++)
    { b=mat[k][j];
      mat[k][j]=mat[k+1][j];
      mat[k+1][j]=b;}
      key=1;
    }
 }
 printf("\n  Sorted massiv \n");
 for(i=0;i<n;i++)
  for(j=0;j<m;j++)
printf("%4d%c",mat[i][j],(j==m-
1)?'\n':' ');
 getch();
return 0;}
```

Одновременная обработка массивов и подмассивов.
Сумма элементов строк матрицы



Подсчет суммы элементов i-ой строки

Дана матрица A(5,5) целого типа. Определить сумму элементов каждой строки и записать ее в новый массив S.



Программа определения суммы строк марицы (Ex3_9)

```
#include "stdafx.h"
#include <stdio.h>
#include <conio.h>
const int N=5;
int main(int argc, char* argv[])
{   int a[N][N],i,j,s[N];
    for(i=0; i<N; i++)
     { printf("Input numbers of %2d string:\n",i);
       for (j=0; j<N; j++) scanf("%d",&a[i][j]);
     }
    puts("    MATRICA          SUMMA ");
    for(i=0;i<N;i++)
       for (j=0,s[i]=0;j<N;j++) s[i]+=a[i][j];
    for(i=0; i<N; i++)
      { for (j=0; j<N; j++) printf("%3d ",a[i][j]);
        printf("%10d\n",s[i]);
      }
    getch();
    return 0; }
```

Обработка матрицы по частям

Кроме задач, по приемам похожих на обработку одномерных массивов, есть большая группа задач, связанных с различными вариантами обхода матриц, и задач обработки разных групп элементов.

int c[5][5]

i=j

i=0 – 4

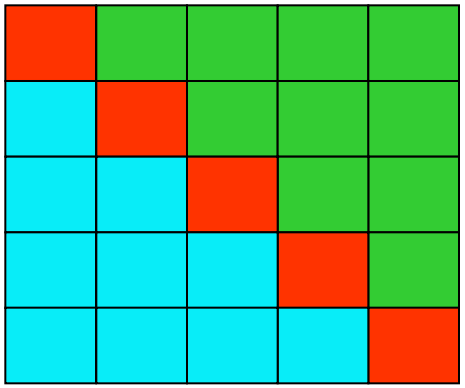
j=i+1- 4

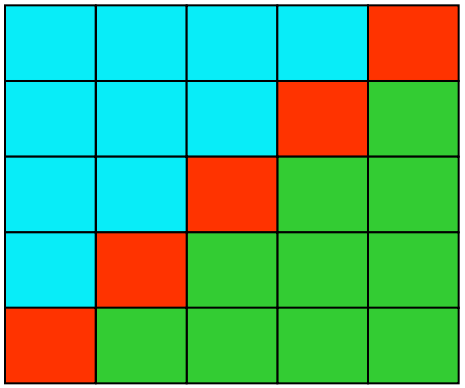
i= 0 – 3

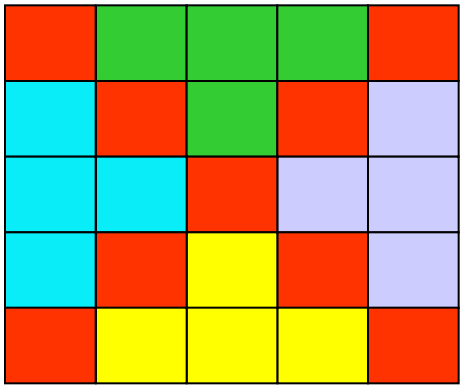
j=0 – 4-i

i+j=4

j=4-i







i=1 – 4

j=0 – i-1

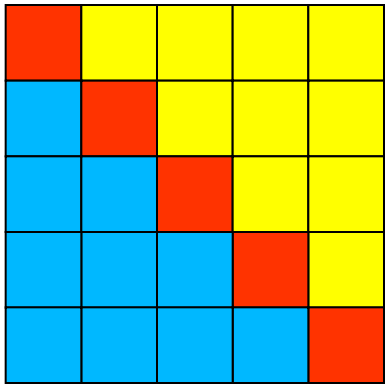
i=1 -4

j=4-i+1 - 4

Пример обработки элементов матрицы, лежащих в определенной области

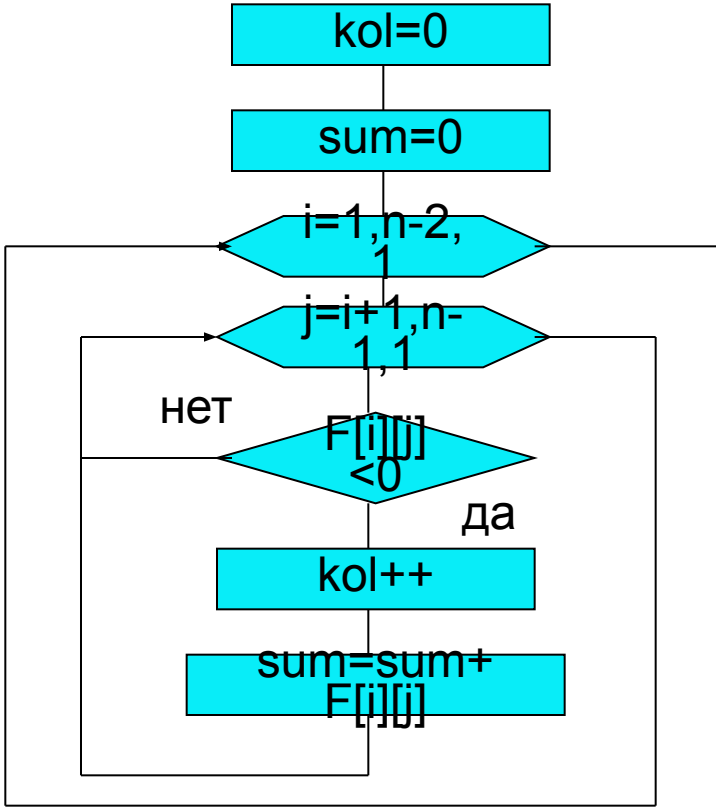
Пример. Дана целочисленная матрица F(10,10). Определить сумму отрицательных элементов матрицы и их количество, среди элементов, лежащих выше главной диагонали.

int F[10][10] n=5



i=3

j=4



Пример программы

```
// Ex3_14.cpp
#include "stdafx.h"
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <conio.h>
int main(int argc, char* argv[])
{int n,mat[10][10],i,j,sum,kol;
srand( (unsigned)time( NULL ));
 printf(" Inputed size of massiv n<=10  \n");
 scanf("%d",&n);
for (i=0;i<n;i++)
 for (j=0;j<n;j++)
    mat[i][j]=rand()/1000-rand()/1000;
printf("\n Inputed Massiv \n");
for(i=0;i<n;i++)
 for(j=0;j<n;j++)
 printf("%4d%c",mat[i][j],(j==n-1)?'\n':' ');
 sum=0;
  kol=0;
 for(i=0;i<n-1;i++)
  for(j=i+1;j<n;j++)
        if(mat[i][j]<0)
        { sum=sum+mat[i][j];
          kol=kol+1;
        }
printf("Summa  “);
printf(“%7d otricatelnyx”,kol);
printf(“elementov");
printf("  =   %8d\n",sum);

 getch();
return 0;
}
```