



**Работа с многомерными
массивами**

Список литературы по курсу

- 1. Князев А.В. Основы языка C++. Учебное пособие. М.: Издательство МЭИ, 2013 – 80 с. ISBN 978-5-7046-1425-8.
- 2. Князев А.В. Работа со сложными структурами данных на языке C++. Учебное пособие. М.: Издательство МЭИ, 2015 – 48 с. ISBN 978-5-7046-1658-0
- 3. Программирование. Сборник задач. Учебное пособие. Санкт-Петербург: Лань, 2019 – 140 с. ISBN 978-5-8114-3857-0
URL: <https://e.lanbook.com/book/121485>

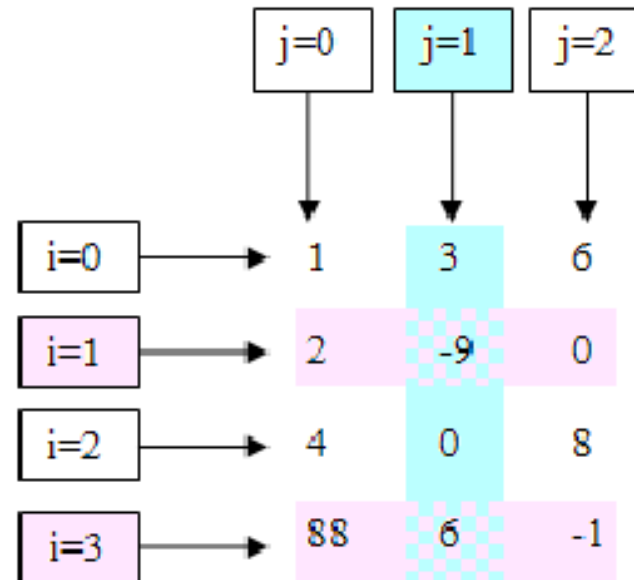
ТИПОВЫЕ АЛГОРИТМЫ

МНОГОМЕРНЫЕ МАССИВЫ

- В математике часто используются многомерные массивы.
- Основные ограничения и принципы работы с одномерными массивы справедливы и в многомерном случае.
- **Матрица** — двумерный массив, задаётся в виде прямоугольной таблицы, которая представляет собой совокупность строк и столбцов, на пересечении которых находятся её элементы.

У двумерного массива имеется два индекса:
 i – номер строки, j – номер столбца.

- **Размерность матрицы** — $n \times m$, количество строк n и столбцов m .



- Если размерность матрицы $n \times m$ и $n \neq m$, тогда матрица называется **прямоугольной**.

Если $n = m$, тогда матрица называется **квадратной**.

Объявление матрицы в программе

тип_данных имя_массива [кол-во строк][кол-во столбцов];

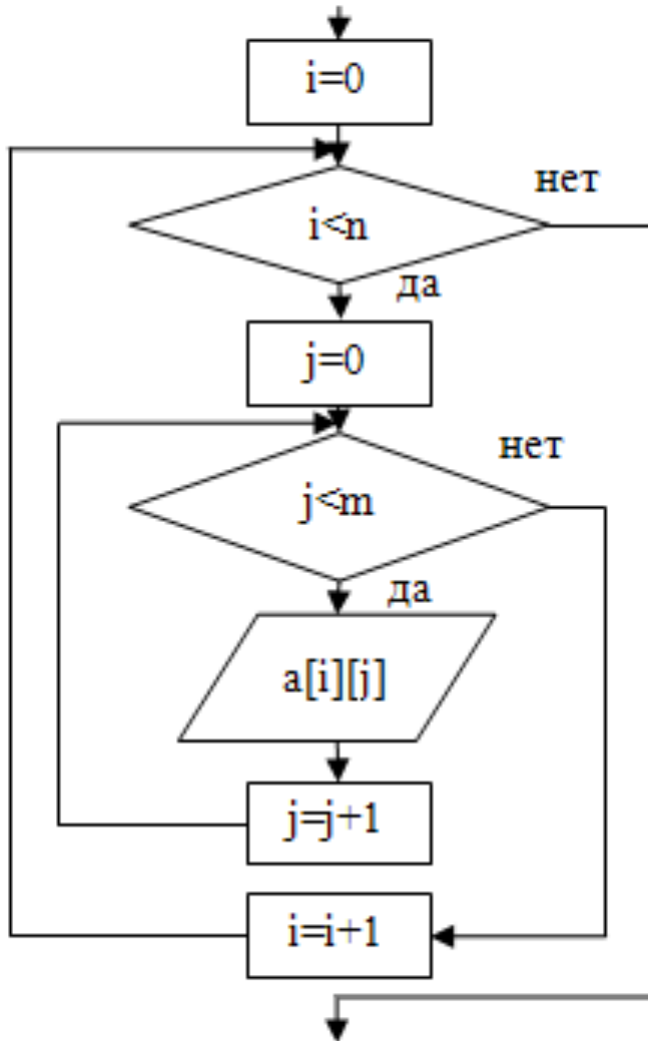
int z [4] [3] – задает матрицу (двумерный массив) из 12 целых чисел. Четыре строки, в каждой из которых по три числа.

z[0,0]	z[0,1]	z[0,2]
z[1,0]	z[1,1]	z[1,2]
z[2,0]	z[2,1]	z[2,2]
z[3,0]	z[3,1]	z[3,2]

1	3	6
2	-9	0
4	0	8
88	6	-1

- Для обращения к элементам матрицы необходимо указать имя матрицы и индексы по строке и столбцу в квадратных скобках **z[i][j]**.
- То есть при обращении к элементам многомерного массива, число индексов должно быть равно числу измерений массива.
- В качестве индекса может быть любое выражение, получающее значения типа индекса (целый тип).

Ввод\вывод матрицы a размерностью $n \times m$



```
#define n 4
#define m 3
int main () {
    double a[n][m];
    int i,j;
    for(i=0; i<n; i++)
    {
        for(j=0; j<m; j++)
        {
            cin>>a[i][j];
            // вывод матрицы cout<<a[i][j];
        }
    }
    return 0;
}
```


Примеры работы с матрицами

- Для обращения к элементам главной диагонали матрицы **a** можно указать **a[i][i]**, т.к. на главной диагонали матрицы **i=j**.
- Для обращения к элементам матрицы **a**, лежащим выше главной диагонали необходимо учитывать, что **j>i**.

```
for(i=0; i<n-1; i++)
{
    for(j=i+1; j<m; j++)
    {
        ...
    }
}
```

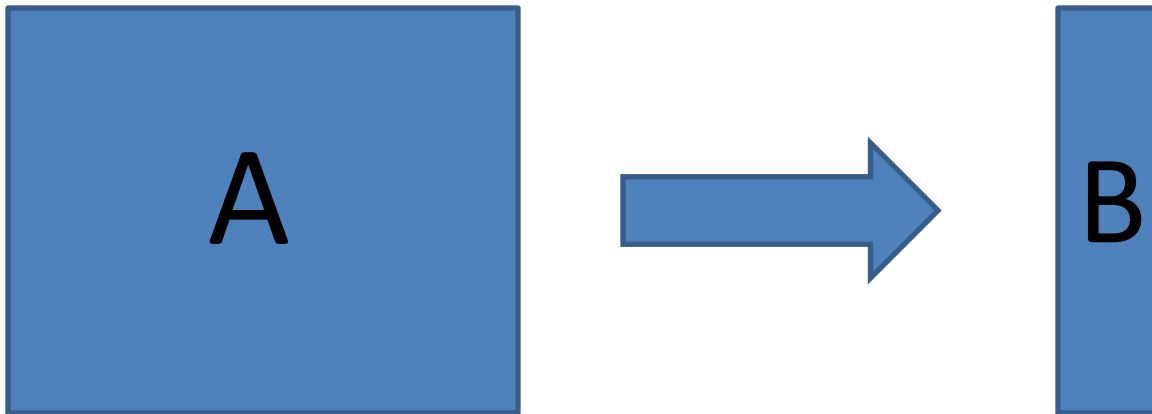
- Для обращения к элементам матрицы **a**, лежащим ниже главной диагонали необходимо учитывать, что **$i > j$** .

```
for(i=1; i<n; i++)  
{  
    for(j=0; j<i; j++)  
    {  
        ...  
    }  
}
```

Пример №1

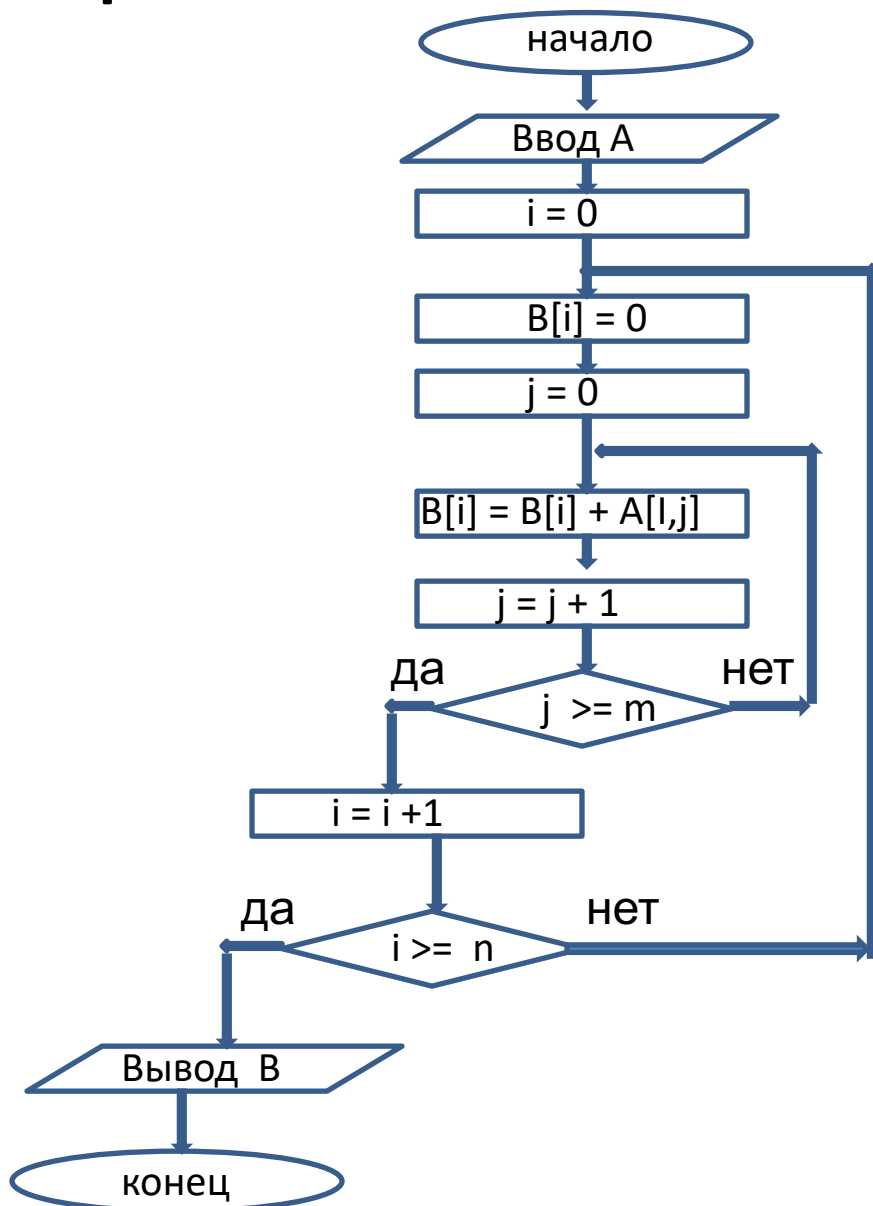
Задана матрица **A** размером $n \times m$.

Найти построчные суммы всех строк матрицы.



$$B_i = \sum_{j=1}^m A_{i,j}$$

Кратный цикл



```
#define n 5
#define m 5
int main () {
float A [n][m]; //матрица
float B [n]; //массив сумм
int i,j; //номера строки и столбца
for (i=0; i<n; i++) // ввод матрицы
{
for (j=0;j<m; j++) {
scanf(“%f”,&A[i][j]); }
printf(“\n”);
}
```

```
for (i= 0; i<n; i++) // цикл для перебора строк
{
    B[i]=0;
    for(j=0; j<m; j++) // суммирование строки
        B[i]=B[i] + A[i][j];
}
for (i=0;i<n;i++) // вывод массива сумм
    printf("%f:5:1",B[i]);
printf("\n");
return 0;
}
```

Пример №2

Задана матрица X из целых чисел.

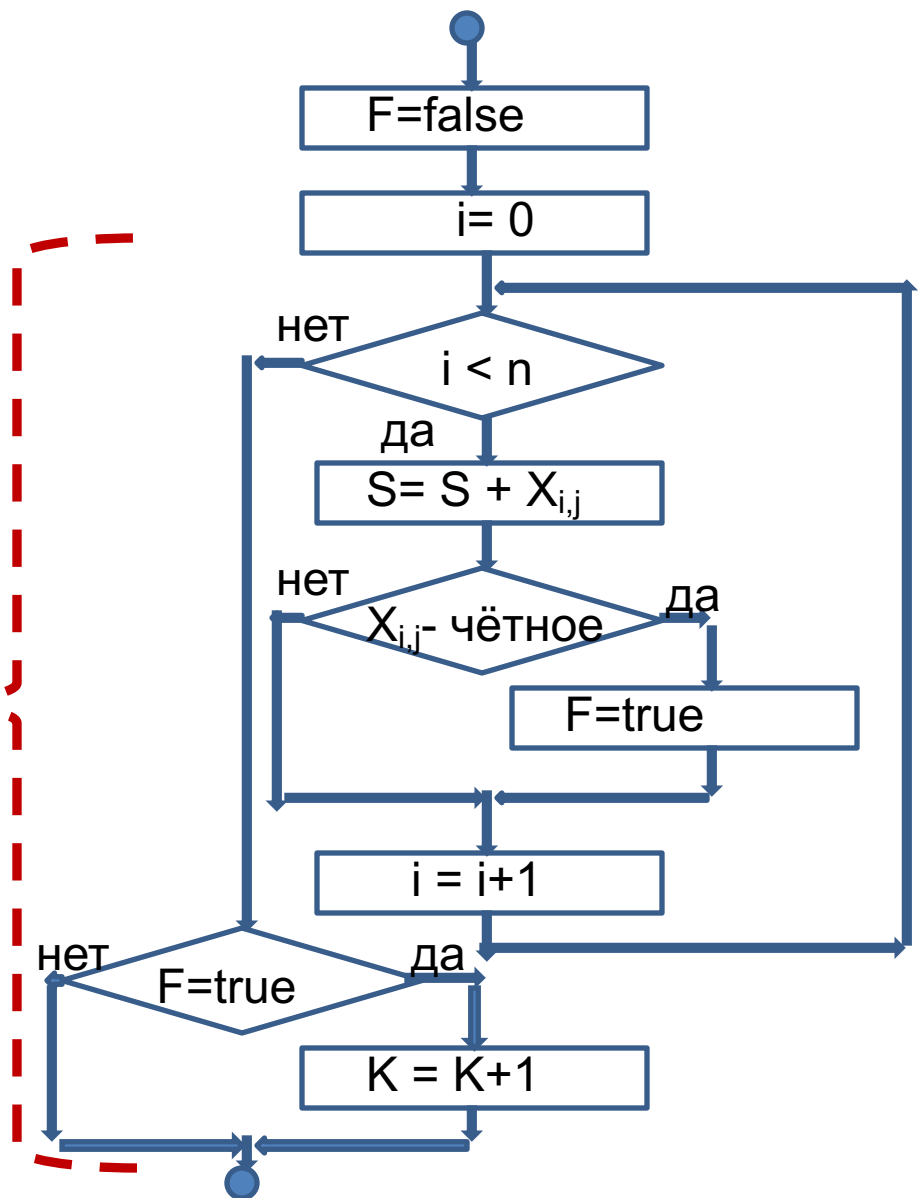
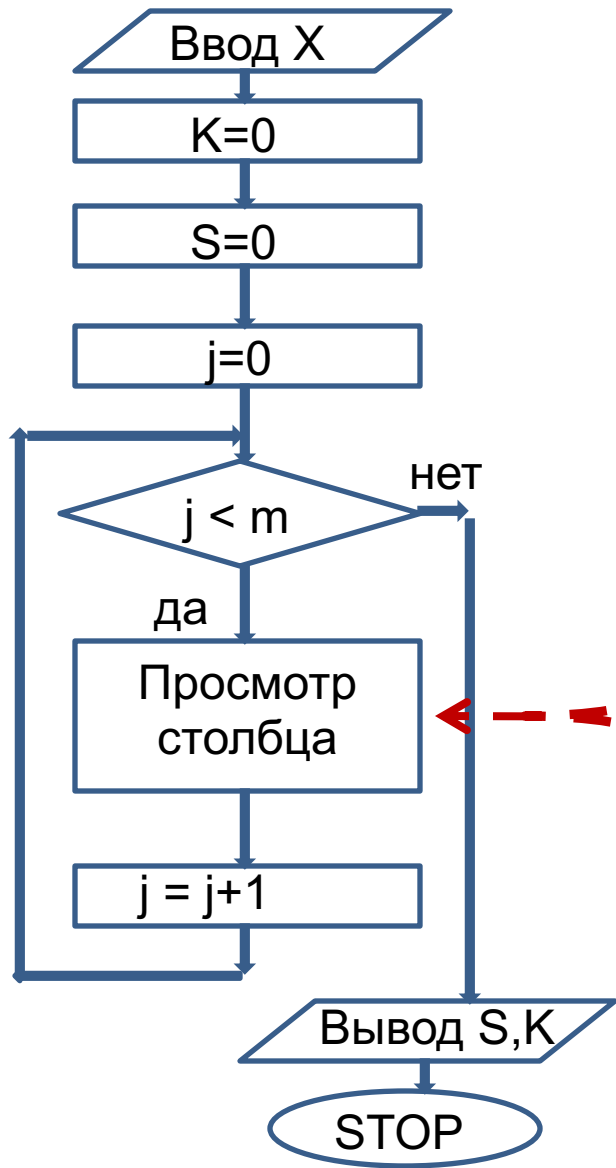
Определить в скольких столбцах матрицы встречаются чётные числа и найти сумму элементов матрицы.

Метод решения

Будем просматривать матрицу по столбцам и суммировать элементы.

Для подсчёта количества столбцов, имеющих чётные элементы, введём логическую переменную **F**, которой будем присваивать значение **TRUE**, если в столбце есть чётный элемент, и значение **FALSE**, если таких элементов нет.

Обозначим сумму элементов матрицы через **S**, а количество столбцов с чётными элементами **K**.



```
#define n 5 // количество строк
#define m 5 // количество столбцов
int main () {
int X [n][m];
int S,K,i,j;
bool F; // флажок
for (i=0; i<n; i++) // ввод матрицы X
{
for (j=0;j<m; j++) {
printf("\n Введите элемент X[%d][%d]: ", i, j);
cin>>X[i][j]; }
}
```

K=0; *// количество строк с чётными эл-ми*

S=0; *// сумма элементов матрицы*

for (j=0;j<m; j++) {

 F=false; *// начальное значение флажка*

 for (i=0; i<n; i++) {

 S+=X[i][j];

 if (X[i][j] % 2 == 0) F=true;

 }

 if (F==true) K++;

}

 cout<<"s="<<S<<" k="<<K;

return 0;

}

Пример №3

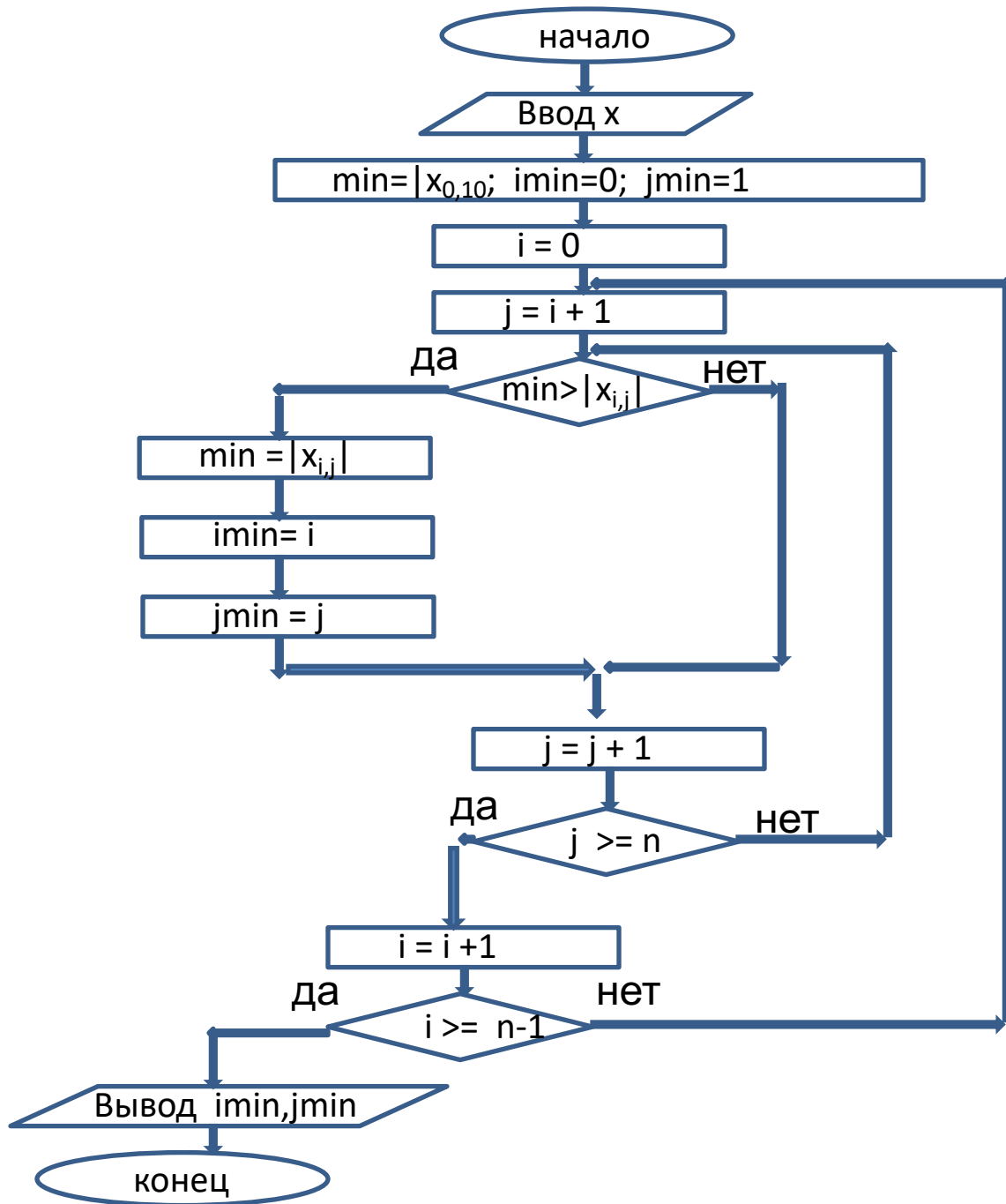
Задана матрица x размерностью $n \times n$.

Найти индексы минимального по модулю элемента матрицы выше главной диагонали.

Для решения задачи будет использоваться кратный цикл.

Обозначим через i_{\min} , j_{\min} индексы минимального по модулю элемента матрицы.

Величину минимума по модулю обозначим \min .



```
#define n 10 // количество строк
int main () {
float x [n][n];
float min;
int imin, jmin, i, j;
for (i=0; i<n; i++) // ввод матрицы
{
for (j=0;j<n; j++) {
scanf("%f",&x[i][j]); }
printf("\n");
}
```

```
min = fabs(x[0][1]);
imin = 0; jmin = 1;
for (i=0; i<n-1; i++) {
    for (j=i+1; j<n; j++) {
        if (min > fabs(x[i][j]))
        {
            min = fabs(x[i][j]);
            imin = i;
            jmin = j;
        }
        cout<<"imin="<<imin<<" jmin="<<jmin;
return 0;
}
```