**1 ЦЕЛЬ И ЗАДАЧИ ЛАБОРАТОРНОЙ РАБОТЫ**

**1.1** Цель лабораторной работы – научиться разрабатывать алгоритмы для работы с одномерными массивами и писать код на языке Си по составленному алгоритму.

**1.2** Для достижения поставленной задачи необходимо изучить лекционный материал по теме «Многомерные массивы».

**1.3** Выполнить задания по лабораторной в соответствии с вариантом №2, разработав алгоритмы их реализации, запрограммировав их с использованием языка «Си», отладив и представив результаты работы компьютерных программ.

**2 Результаты выполнения лабораторной работы**

**2.1 Задание 1.**Элемент матрицы назовем седловой точкой, если он наименьший в своей строке и наибольший (одновременно) в своем столбце (или наоборот, наибольший в своей строке и наименьший в своем столбце). Для заданной целой матрицы размером 10 x 12 напечатать индексы всех ее седловых точек.

**2.1.1** На рисунке 2.1.1 приведена блок-схема алгоритма для выполнения задания №1.

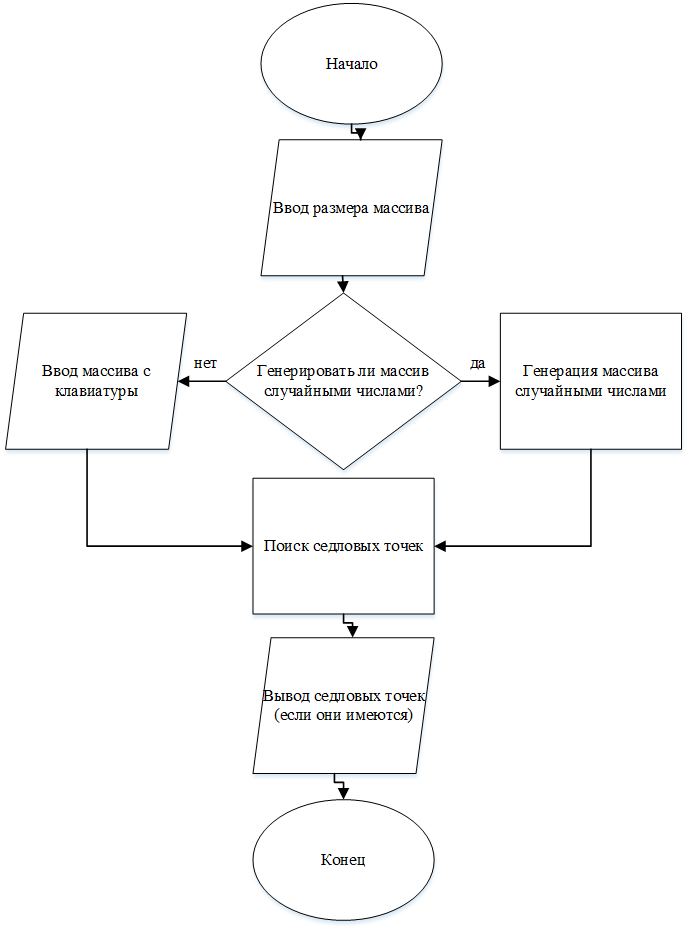


Рисунок 2.1.1

**2.1.2** Листинг компьютерной программы по заданию №1.

#include <stdio.h>

#include <stdlib.h>

#include <process.h>

int resh, col, row = 0;

int i,j=0;

int array[100][100] = { 0 };

//int min = array[0][0];

void input\_numb()

{

printf("Enter the dimension of rows\n");

while (scanf\_s("%d", &row) != 1 || row > 100 || row < 0 || getchar() != '\n')

{

printf("Wrong input,try again\n");

rewind(stdin);

}

printf("Enter the dimension of columns\n");

while (scanf\_s("%d", &col) != 1 || col > 100 || col < 0 || getchar() != '\n')

{

printf("Wrong input,try again\n");

rewind(stdin);

}

}

void rnd ()//random func.

{

srand(\_getpid());

int oper;

printf("Choose an operation:\n1: Random\n2: Keyboard\nSelected operation: ");

while (scanf\_s("%d", &oper) != 1 || oper > 2 || oper < 1 || getchar() != '\n')

{

printf("\nError.\tInput number: ");

rewind(stdin);

}

switch (oper)

{

case 1:

{

for (int i = 0; i <= row; i++)

{

for (int j = 0; j <=col; j++)

{

array[i][j] = rand() % 150-50;

}

}

}

break;

case 2:

printf("Enter the array elements:\n");

{

for (int i = 0; i <= row; i++)

{

for (int j = 0; j <= col; j++)

{

while (scanf\_s("%d", &array[i][j]) != 1 || getchar() != '\n')

{

printf("\nError.\tInput number: ");

rewind(stdin);

}

}

}

}

break;

}

}

void conclusion()//exit array

{

for (int i = 0; i <= row; i++)

{

for (int j = 0; j <= col; j++) {

printf("[%d] [%d] = %d \t", i, j, array[i][j]);

}

printf("\n");

}

}

void min\_max()

{

int k = 0;

int max1,min1,f1 = 0;

int max2, min2, f2 = 0;

printf("\n");

for (int i = 0; i <= row; i++)

{

min1 = array[i][0];

for (int x = 1; x <=col; x++)

{

if (array[i][x] < min1)

{

min1 = array[i][x];

}

}

for (int j = 0; j <=col; j++)

{

if (array[i][j] == min1)

{

max1 = array[0][j];

for (int y = 1; y <=row; y++){

if (array[y][j] > max1){

max1 = array[y][j];

}

}

if (array[i][j] == max1)

{

printf("Saddle point:[%d][%d]=%d\n", i, j,array[i][j]);

f1 = 1;

}

}

}

}

if (!f1){

printf("The matrix has no saddle points!(1;max)\n");

}

for (int i = 0; i <= row; i++){

max1 = array[i][0];

for (int x = 1; x <= col; x++)

{

if (array[i][x] >max1)

{

max1 = array[i][x];

}

}

for (int j = 0; j <= col; j++){

if (array[i][j] == max1)

{

min1 = array[0][j];

for (int y = 1; y <= row; y++)

{

if (array[y][j] < min1)

{

min1 = array[y][j];

}

}

if (array[i][j] == min1)

{

printf("Saddle point:[%d][%d]=%d\n", i, j, array[i][j]);

f2 = 1;

}

}

}

}

if (!f2){

printf("The matrix has no saddle points!(2;min)\n");

}

}

int main()

{

input\_numb();

rnd();

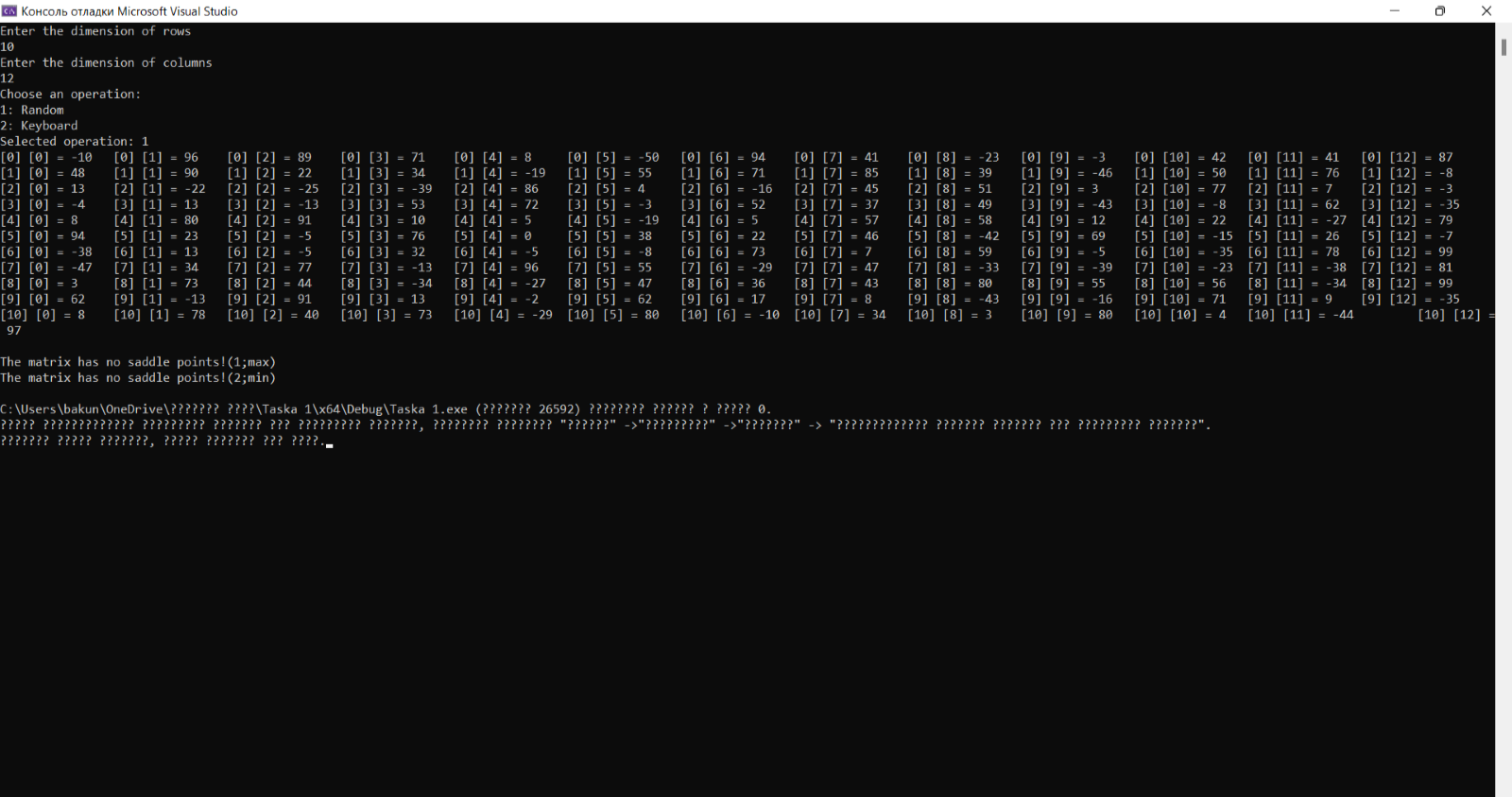
conclusion();

min\_max();

return 0;

}

**2.1.3** Результат выполнения компьютерной программы на экран в виде «скриншота» изображения на мониторе представлены на рисунках 2.1.2:

  
  
Рисунок 2.1.2

**2.2 Задание 2.** Найти в матрице первую строку, все элементы которой отрицательны. Увеличить все элементы матрицы на значение первого элемента найденной строки.

**2.2.1** На рисунке 2.2.1 приведена блок-схема алгоритма для выполнения задания №2.

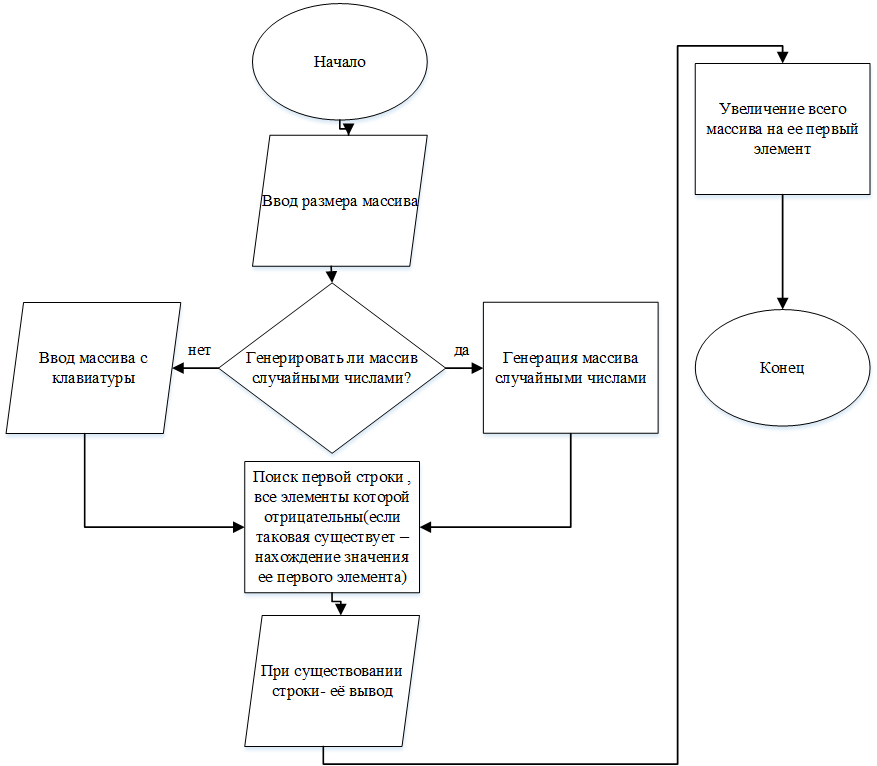


Рисунок 2.2.1

**2.2.2** Листинг компьютерной программы по заданию №2.

#include <stdio.h>

#include <stdlib.h>

#include <process.h>

int resh, col, row = 0;

int i, j = 0;

int array[100][100] = { 0 };

//int min = array[0][0];

void input\_numb()

{

printf("Enter the dimension of rows\n");

while (scanf\_s("%d", &row) != 1 || row > 100 || row < 0 || getchar() != '\n')

{

printf("Wrong input,try again\n");

rewind(stdin);

}

printf("Enter the dimension of columns\n");

while (scanf\_s("%d", &col) != 1 || col > 100 || col < 0 || getchar() != '\n')

{

printf("Wrong input,try again\n");

rewind(stdin);

}

}

void rnd()//random func.

{

srand(\_getpid());

int oper;

printf("Choose an operation:\n1: Random\n2: Keyboard\nSelected operation: ");

while (scanf\_s("%d", &oper) != 1 || oper > 2 || oper < 1 || getchar() != '\n')

{

printf("\nError.\tInput number: ");

rewind(stdin);

}

switch (oper)

{

case 1:

{

for (int i = 0; i <= col; i++)

{

for (int j = 0; j <= row; j++)

{

array[i][j] = rand() % 1000 - 650;

}

}

}

break;

case 2:

{

for (int i = 0; i <= col; i++)

{

for (int j = 0; j <= row; j++)

{

while (scanf\_s("%d", &array[i][j]) != 1 || getchar() != '\n')

{

printf("\nError.\tInput number: ");

rewind(stdin);

}

}

}

}

break;

}

}

void conclusion()//exit array

{

printf("\nMatrix:\n");

for (int i = 0; i <= col; i++)

{

for (int j = 0; j <= row; j++) {

printf("[%d] [%d] = %d \t", i, j, array[i][j]);

}

printf("\n");

}

}

void search()

{

int h=row;

int j,k = 0;

int frst = 0;

int cntr = 0;

for (int i = 0; i <= col; i++)

{

if (array[i][0] < 0)

{

for (int j = 0; j <= row; j++)

{

if (array[i][j] < 0)

{

cntr++;

if ( k == 0)

{

if (cntr == row + 1)

{

printf("\n");

printf("The row with all negative elements is under the number = %d\n", i);

k++;

if (frst == 0)

{

frst= array[i][0];

printf("\n");

printf("First negative element is [%d] [%d]=%d\n",i,j, frst);

}

if (frst != 0)

{

printf("\nEnlarged array:\n");

for (int i = 0; i <= col; i++)

{

for (int j = 0; j <= row; j++) {

printf("[%d] [%d] = %d \t", i, j, array[i][j]+frst);

}

printf("\n");

}

}

}

}

}

}

}

cntr = 0;

}

}

int main()

{

input\_numb();

rnd();

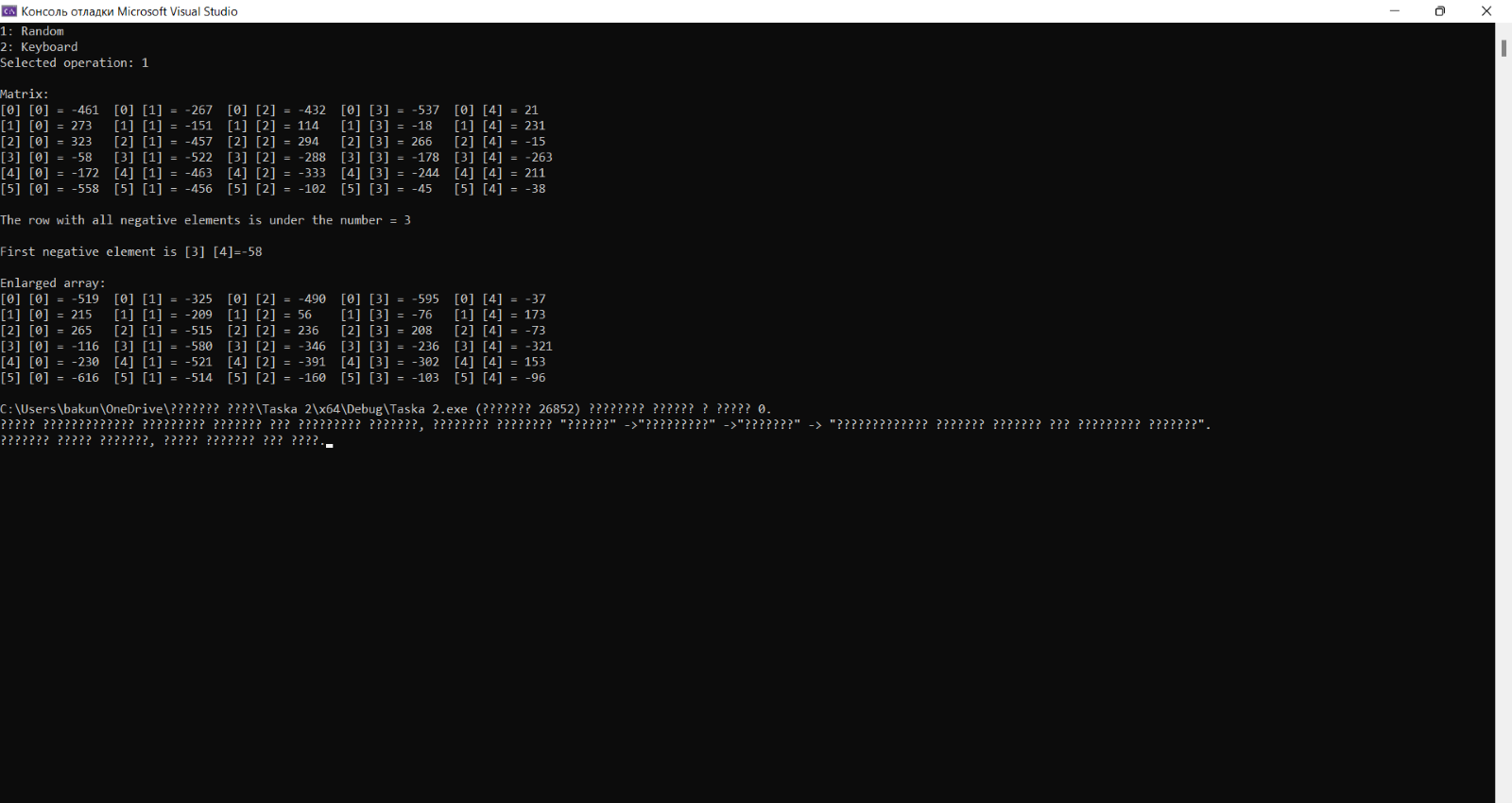
conclusion();

search();

return 0;

}

**2.2.3** Результат выполнения компьютерной программы на экран в виде «скриншота» изображения на мониторе представлены на рисунках 2.2.2:

  
  
Рисунок 2.2.2

**2.3 Задание 3.**В квадратной матрице размером NxN найти сумму элементов в 2-ой области. Область представлена на рисунке 2.3.1:

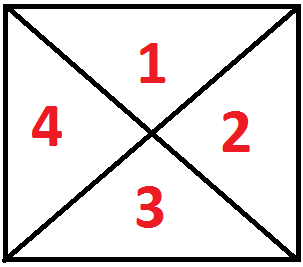


Рисунок 2.3.1

**2.3.1** На рисунке 2.3.2 приведена блок-схема алгоритма для выполнения задания №3.

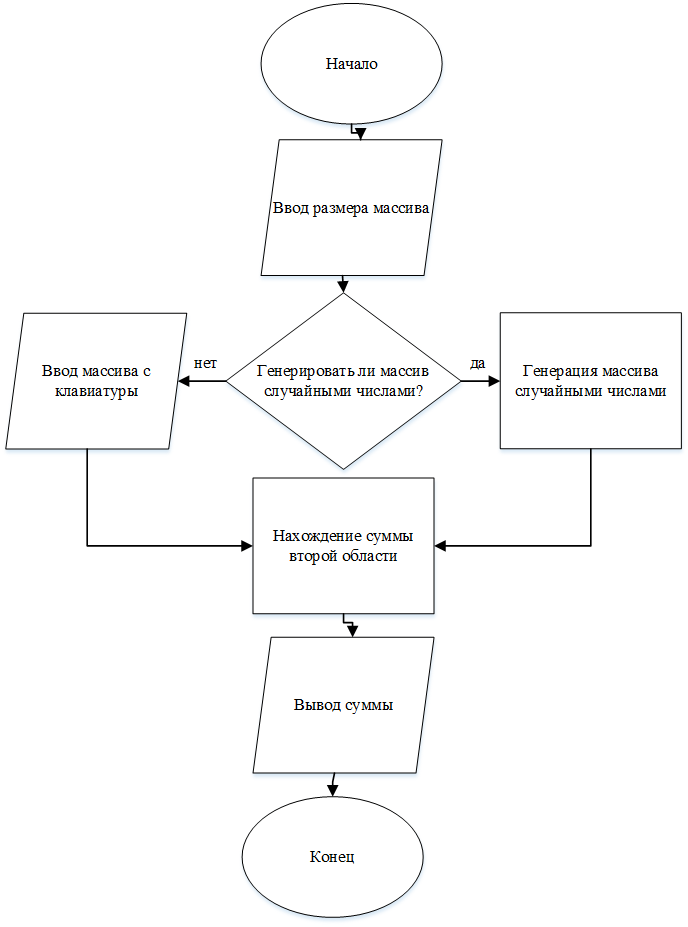


Рисунок 2.3.2

**2.3.2** Листинг компьютерной программы по заданию №3.

#include <stdio.h>

#include <stdlib.h>

#include <process.h>

int size = 0;

int i, j = 0;

int array[100][100] = { 0 };

void input\_numb()

{

printf("Enter the matrix size\n");

while (scanf\_s("%d", &size) != 1 || size > 100 || size<0 || getchar() != '\n')

{

printf("Wrong input,try again\n");

rewind(stdin);

}

}

void rnd()//random func.

{

srand(\_getpid());

int oper;

printf("Choose an operation:\n1: Random\n2: Keyboard\nSelected operation: ");

while (scanf\_s("%d", &oper) != 1 || oper > 2 || oper < 1 || getchar() != '\n')

{

printf("\nError.\tInput number: ");

rewind(stdin);

}

switch (oper)

{

case 1:

{

for (int i = 0; i <= size; i++)

{

for (int j = 0; j <= size; j++)

{

array[i][j] = rand() % 10;

}

}

}

break;

case 2:

printf("Enter the array elements:\n");

{

for (int i = 0; i <= size; i++)

{

for (int j = 0; j <= size; j++)

{

while (scanf\_s("%d", &array[i][j]) != 1 || getchar() != '\n')

{

printf("\nError.\tInput number: ");

rewind(stdin);

}

}

}

}

break;

}

}

void sumator()

{

int k = { size }; ;

int j = 0;

//int i = size;

int half = (size) / 2;

int sum = 0;

for (int i = 0; i<= size; i++)

{

if (i <= size / 2 )

{

for (int j = size ; j >= size- i; j--)

{

sum += array[i][j];

//printf("\nSum1: [%d] [%d]=%d ", i, j, sum);

}

}

else

{

for (int j = size; j >=i; j--)

{

sum += array[i][j];

//printf("\nSum2: [%d] [%d]=%d ", i, j, sum);

}

}

}

printf("\nSum: = %d ", sum);

}

void conclusion()//exit array

{

printf("\nMatrix:\n");

for (int i = 0; i <= size; i++)

{

for (int j = 0; j <= size; j++) {

printf("[%d] [%d] = %d \t", i, j,array[i][j]);

}

printf("\n");

}

}

int main()

{

input\_numb();

rnd();

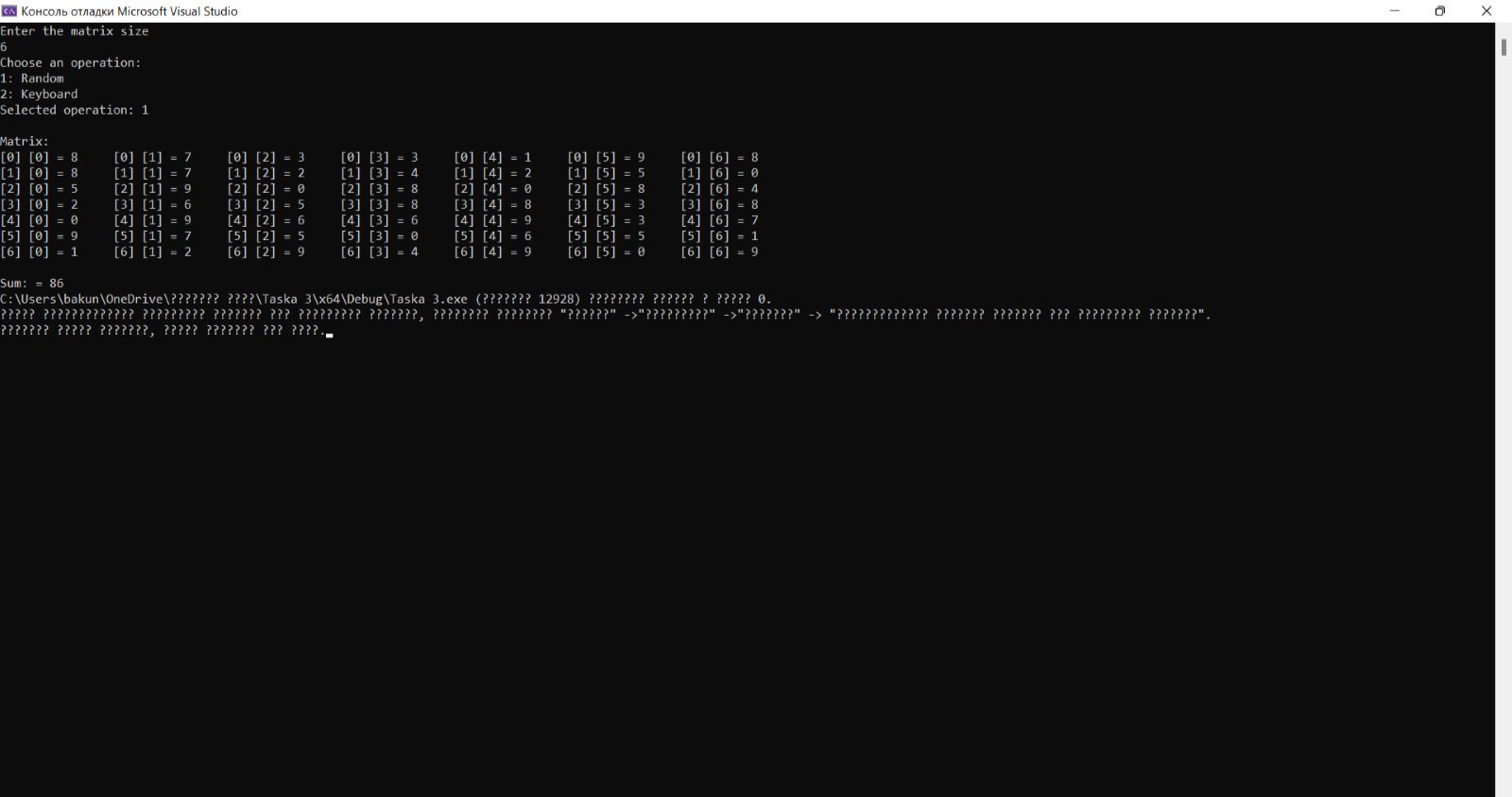
conclusion();

sumator();

return 0;

}

**2.3.3** Результат выполнения компьютерной программы на экран в виде «скриншота» изображения на мониторе представлены на рисунках 2.3.3:

  
  
  
Рисунок 2.3.3

**4 ВЫВОДЫ ПО РЕЗУЛЬТАТАМ ВЫПОЛНЕНИЯ ЛР**

В результате выполнения лабораторной работы, я научился разрабатывать алгоритмы для работы с многомерными массивами и писать код на языке Си по составленному алгоритму.