**Лабораторная работа № 8**

**Задание № 3**

1. **Общая постановка задачи**
2. Напишите функцию (в соответствии с указанным вариантом) для работы со встроенным массивом. Функция, должна иметь три параметра – адрес матрицы, количество строк и количество столбцов и выполнять действия в соответствии с указанным вариантом. Элементы матрицы – целые числа. Ввод и вывод матрицы выполнять в отдельных функциях. Входные данные: для тестирования можно определить матрицы, как локальные переменные и сразу же проинициализировать или значения элементов матрицы ввести из файла. Выходные данные: в соответствии заданным с вариантом вывести в файл.

Количество строк, содержащих только положительные элементы.

2. Напишите функцию (в соответствии с указанным вариантом) для работы с двумерным массивом – квадратной матрицей в динамической памяти. Функция, должна иметь два параметра – адрес матрицы (целые числа), размерность матрицы и выполнять действия в соответствии с указанным вариантом. Элементы матрицы – целые числа. Ввод и вывод матрицы выполнять в отдельных функциях. Входные данные: размерность матрицы, значения элементов (если требуется) ввести из файла. Выходные данные: в соответствии заданным с вариантом вывести в файл.

Матрицу N\*N заполнить натуральными числами от 1 до N\*N по спирали, начинающейся в нижнем левом углу и закрученной по часовой стрелке.

1. **Таблица с детальными требованиями и тест план**

|  |  |  |  |
| --- | --- | --- | --- |
| **Требование** | **Детальные требования** | **Данные** | **Ожидаемый результат** |
| 1. **Данные должны быть введены корректно** | | | |
| 1.1  Размер матрицы – положительное, не равное нулю число | Если введено не положительное число или ноль, сообщение: "Введено неверное значение". | 0  -4 | Сообщение: «Введено неверное значение!» |
| 1. Матрицы | | | |
| 2.1 Вычисление полностью положительных строк в матрицу | Выводится количество полностью положительных строк | -1 2 3  -4 5 6  7 -8 9  1 2 3 | Выходной файл: «Количество строк только с положительными элементами: 1» |
| 2.2 Формирование спиральной матрицы по заданным размерам | Выводится спиральная матрица | 11 11 | 11 12 13 14 15 16 17 18 19 20 21  10 49 50 51 52 53 54 55 56 57 22  9 48 79 80 81 82 83 84 85 58 23  8 47 78 101 102 103 104 105 86 59 24  7 46 77 100 115 116 117 106 87 60 25  6 45 76 99 114 121 118 107 88 61 26  5 44 75 98 113 120 119 108 89 62 27  4 43 74 97 112 111 110 109 90 63 28  3 42 73 96 95 94 93 92 91 64 29  2 41 72 71 70 69 68 67 66 65 30  1 40 39 38 37 36 35 34 33 32 31 |

1. **Код программы**

#include <iostream>

#include <iomanip>

#include <locale>

#include <fstream>

const int ROWS = 4;

const int COLS = 3;

void outConstArray(int matrix[][COLS], std::fstream\* out)

{

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

{

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

{

\*out << std::setw(4) << matrix[i][j];

}

\*out << std::endl;

}

}

void outDynamicArray(int\*\* matrix, int rows, int cols, std::fstream\* out)

{

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

{

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

{

\*out << std::setw(4) << matrix[i][j];

}

\*out << std::endl;

}

}

int pozitiveRows(const int arr[ROWS][COLS], const int rows, const int cols)

{

int counter = 0;

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

{

bool flag = true;

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

{

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

{

flag = false;

}

}

if (flag == true)

{

counter++;

}

}

return counter;

}

int\*\* spiralMatrix(int\*\* matrix, const int rows, const int cols)

{

int number = 1;

int step = rows / 2;

for (int k = 1; k <= step + 1; k++)

{

for (int j = rows - k; j >= k; j--) matrix[j][k - 1] = number++;

for (int j = k - 1; j < rows - k + 1; j++) matrix[k - 1][j] = number++;

for (int j = k; j < rows - k + 1; j++) matrix[j][rows - k] = number++;

for (int j = rows - k - 1; j >= k; --j) matrix[rows - k][j] = number++;

}

return matrix;

}

void deleteMatrix(int\*\* matrix, int const rows, int const cols)

{

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

{

delete[] matrix[i];

}

delete[] matrix;

}

int main()

{

setlocale(LC\_ALL, "ru");

std::fstream fileOut;

fileOut.open("out.txt");

int constArr[ROWS][COLS] = { {-1, 2, 3}, {-4, 5, 6}, {7, -8, 9}, {1, 2, 3} };

fileOut << "Встроенная матрица: " << std::endl;

outConstArray(constArr, &fileOut);

fileOut << "Количество строк только с положительными элементами: ";

fileOut << pozitiveRows(constArr, ROWS, COLS) << std::endl;

int rowsD = 0;

int colsD = 0;

std::cout << "Введите количество строк: ";

std::cin >> rowsD;

if (std::cin.fail() || (rowsD <= 0))

{

std::cout << "Введено неверное значение!";

return -1;

}

std::cout << "Введите количество столбцов: ";

std::cin >> colsD;

if (std::cin.fail() || (colsD <= 0))

{

std::cout << "Введено неверное значение!";

return -1;

}

if (colsD != rowsD)

{

std::cout << "Матрица должна быть квадратной!";

return -1;

}

int\*\* dynamicMatrix = new int\* [colsD];

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

{

dynamicMatrix[i] = new int[rowsD];

}

fileOut << "Спиральная матрица: " << std::endl;

outDynamicArray(spiralMatrix(dynamicMatrix, rowsD, colsD), rowsD, colsD, &fileOut);

deleteMatrix(dynamicMatrix, rowsD, colsD);

fileOut.close();

return 0;

}