ФИО: Гольчанский Максим Дмитриевич

Группа: 3530904/10003

Лабораторная работа №5: Двумерные массивы. Указатели.

Вариант №6

Дата отправки задания: 24.11.2021

**Общая постановка задачи:**

Разработать детальные требования и тест план для следующей задачи:

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.

**Задача 1:**

Из файла  выводится количество элементов матрицы и сама матрица. Программе необходимо найти количество столбцов исходной матрицы, в которых все элементы положительные.

2.   Детальные требования

2.1 Путь к файлу должен быть задан корректно

2.1.1 Файл должен существовать по указанному пути.

2.1.1.1 Если файла не существует, то вывести сообщение “Ошибка: Не удалось открыть файл, завершение программы”.

          2.2  Путь к файлу для записи результата должен быть задан корректно

2.2.1 Информация должна поступать в файл для записи корректно

2.2.1.1 Если в файле находиться не целое число, то вывести сообщение “Ошибка: Не удалось открыть файл для записи, завершение программы”.

3.   Тест-план

| Требования | Детальные требования | Данные | Результат | |
| --- | --- | --- | --- | --- |
| 3.1 Путь к файлу должен быть задан корректно | | | | |
| 3.1.1 Путь к файлу должен быть задан корректно | 3.1.1.1 Если файла не существует, то вывести сообщение “Ошибка: Не удалось открыть файл, завершение программы” |  | Сообщение “Ошибка: Не удалось открыть файл, завершение программы”. | |
| 3.2 Путь к файлу для записи результата должен быть задан корректно | | | | |
| 3.2.1  Информация должна поступать в файл для записи корректно | 3.2.1.1 Если в файле находиться не целое число, то вывести сообщение “Ошибка: Не удалось открыть файл для записи, завершение программы”. |  | | Сообщение “Ошибка: Не удалось открыть файл для записи, завершение программы”. |

**Задача 2:**

В файл вводится количество элементов квадратной матрицы. Программе необходимо по спирали заполнить всю матрицу числами, начиная с единицы, а место отсчета — верхний левый угол. Спираль закручивается против часовой стрелки.

2.   Детальные требования

2.1 Путь к файлу должен быть задан корректно

2.1.1 Файл должен существовать по указанному пути.

2.1.1.1 Если файла не существует, то вывести сообщение “Ошибка: Не удалось открыть файл, завершение программы”.

          2.2 Данные в файле должны быть заданы корректно.

2.2.1 В файле должно находиться целое положительное число.

2.2.1.1 Если в файле находиться не целое число, то вывести сообщение “Ошибка: Не удалось считать значение, завершение программы”.

2.2.1.2 Если в файле целое число <1, то вывести сообщение “Ошибка: Некорректно задана размерность матрицы, завершение программы”.

2.3 Путь к файлу для записи результата и права доступа должны быть заданы корректно.

2.3.1 Путь к создаваемому должен существовать, и программа должна иметь разрешение на запись в указанный файл.

2.3.1.1 Если пути к файлу не существует или программа не имеет разрешения на запись в указанный файл, то вывести сообщение “Ошибка: Не удалось открыть файл для записи результата, завершение программы”

3.   Тест-план

| Требования | Детальные требования | | | Данные | Результат | |
| --- | --- | --- | --- | --- | --- | --- |
| 3.1Путь к файлу должен быть задан корректно | | | | | | |
| 3.1.1 Файл должен существовать по указанному пути | 3.1.1.1 Если файла не существует, то вывести сообщение “Ошибка: Не удалось открыть файл, завершение программы”. | | |  | Сообщение “Ошибка: Не удалось открыть файл, завершение программы”. | |
| 3.2 Данные в файле должны быть заданы корректно. | | | | | | |
| 3.2.1 В файле должно находиться целое положительное число | 3.2.1.1 Если в файле находиться не целое число, то вывести сообщение “Ошибка: Не удалось считать значение, завершение программы”. | | | Gamma | | Сообщение “Ошибка: Не удалось считать значение, завершение программы”. |
| 3.2.1 В файле должно находиться целое положительное число | 3.2.1.2 Если в файле целое число <1, то вывести сообщение “Ошибка: Некорректно задана размерность матрицы, завершение программы” | | | -3 | | Сообщение “Ошибка: Некорректно задана размерность матрицы, завершение программы”. |
| 3.3 Путь к файлу для записи результата и права доступа должны быть заданы корректно. | | | | | | |
| 3.3.1 Путь к создаваемому должен существовать, и программа должна иметь разрешение на запись в указанный файл. | | 3.3.1.1 Если пути к файлу не существует или программа не имеет разрешения на запись в указанный файл, то вывести сообщение “Ошибка: Не удалось открыть файл для записи результата, завершение программы” |  | | | Сообщение “Ошибка: Не удалось открыть файл для записи результата, завершение программы” |

**Реализация**:

// WorkWithArrays.h

#ifndef WORK\_WITH\_ARRAYS\_H

#define WORK\_WITH\_ARRAYS\_H

#include <iostream>

namespace WorkWithArrays {

int CountOfPositiveColumns(const int matrix[], int n, int m);

void fillTheMatrix(int matrix[], int n, int m, std::istream& fin);

void printTheMatrix(const int matrix[], int n, int m, std::ostream& fout);

void fillTheMatrixForSecondPart(int matrix[], int n);

}

#endif

// WorkWithArrays.cpp

#include " WorkWithArrays.h"

int WorkWithArrays::CountOfPositiveColumns(const int matrix[], int n, int m) {

int count = 0;

bool Logic = true;

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

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

if (matrix[n \* m + n] < 0) {

Logic = false;

break;

}

}

if (Logic) {

count++;

}

Logic = true;

}

return count;

}

void  WorkWithArrays::fillTheMatrix(int matrix[], int n, int m, std::istream& fin) {

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

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

fin >> matrix[i \* m + j];

if (fin.fail() && !fin.eof()) {

throw std::logic\_error("Элементы матрицы должны быть целыми числами");

}

}

}

}

void  WorkWithArrays::printTheMatrix(const int matrix[], int n, int m, std::ostream& fout) {

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

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

fout << matrix[i \* m + j] << '\t';

}

fout << '\n';

}

}

void WorkWithArrays::fillTheMatrixForSecondPart(int matrix[], int n) {

int row = 0;

int column = 0;

int Value = 1;

for (int i = 0; i < n \* n; i++) {

matrix[row \* n + column] = Value;

if (i < 4 \* (n - 1) + 1) {

if (column == 0 && row < n - 1) {

++row;

}

else if (row == n - 1 && column < n - 1) {

++column;

}

else if (column == n - 1 && row > 0) {

--row;

}

else if (row == 0 && column > 0) {

--column;

}

}

else {

if (matrix[(column - 1) \* n + row] == 0 && matrix[column \* n + row - 1] != 0) {

--column;

}

else if (matrix[row \* n + column + 1] == 0 && matrix[(column - 1) \* n + row] != 0) {

++row;

}

else if (matrix[(column + 1) \* n + row] == 0 && matrix[column \* n + row + 1] != 0) {

++column;

}

else if (matrix[column \* n + row - 1] == 0 && matrix[(column + 1) \* n + row] != 0) {

--row;

}

}

++Value;

}

}

// main.cpp

#include <iostream>

#include <fstream>

#include "processStaticArrays.h"

int main() {

setlocale(LC\_ALL, "rus");

std::ifstream FinForFirst("C:\\Папка\\Laba\_5\_1\_input.txt");

if (!FinForFirst.is\_open()) {

std::cout << "Не удалось открыть файл с данными" << std::endl;

return 1;

}

const int ROWS = 3;

const int COLUMNS = 4;

int matrix[ROWS \* COLUMNS] =

{ 0, -8, 6, 4,

7, -5, 1, 2,

6,  2, 9, 3 };

FinForFirst.close();

if (!FinForFirst) {

std::cerr << "Не закрылся поток вывода" << std::endl;

}

std::ofstream FoutForFirst("C:\\Папка\\Laba\_5\_1\_output.txt");

if (!FinForFirst.is\_open()) {

std::cout << "Не удалось открыть файл для вывода данных" << std::endl;

return 1;

}

WorkWithArrays::printTheMatrix(matrix, ROWS, COLUMNS, FoutForFirst);

FoutForFirst << WorkWithArrays::CountOfPositiveColumns(matrix, ROWS, COLUMNS);

FoutForFirst.close();

if (!FoutForFirst) {

std::cerr << "Не закрылся поток вывода" << std::endl;

}

std::ifstream FinForSecond("C:\\Папка\\Laba\_5\_2\_input.txt");

if (!FinForSecond.is\_open()) {

std::cout << "Не удалось открыть файл с данными" << std::endl;

return 1;

}

int size = 0;

FinForSecond >> size;

if (FinForSecond.fail() && !FinForSecond.eof())

{

std::cout << "Размер матрицы должен быть задан целым положительным числом" << std::endl;

return 2;

}

int\* matrix = new int[size \* size]{};

FinForSecond.close();

if (!FinForSecond) {

std::cerr << "Не закрылся поток вывода" << std::endl;

}

std::ofstream FoutForSecond("C:\\Папка\\Laba\_5\_2\_output.txt");

if (!FinForSecond.is\_open()) {

std::cout << "Не удалось открыть файл для вывода данных" << std::endl;

return 1;

}

WorkWithArrays::printTheMatrix(matrix, size, size, FoutForSecond);

FoutForFirst << WorkWithArrays::CountOfPositiveColumns(matrix, size, size);

delete[] matrix;

FoutForSecond.close();

if (!FoutForFirst) {

std::cerr << "Не закрылся поток вывода" << std::endl;

}

return 0;

}