Тампио Илья Сергеевич 3530904/10002 Индивидуальное задание №5

1. Встроенные массивы

Общая постановка задачи

Напишите функцию, определяющую количество столбцов, содержащих хотя бы один нулевой элемент, для работы со **встроенным массивом**. Функция должна иметь три параметра – адрес матрицы, количество строк и количество столбцов.

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.**

Детальные требования

|  |  |  |  |
| --- | --- | --- | --- |
| 1. Ввод | | | |
| 1.1 Кол-во строк и кол-во столбцов | 1.1 Оба больше нуля, иначе исключение: «"Размер массива задан некорректно. Завершение программы...» и завершение программы. | 0 1  -1 1 | Сообщение: «"Размер массива задан некорректно. Завершение программы...» |
| 1.2 Файлы ввода \ выводы | 1.2 Файлы доступны программе, иначе соответствующее исключение | 1. Нет файла ввода: 2. Файл вывода недоступен для записи | 1. Сообщение: «"Файл ввода недоступен. Завершение программы..."» 2. «"Файл вывода недоступен. Завершение программы..."» |
| 1.3 Массив из целых чисел | Массив должен состоять из целых чисел. В противном случае, исключение: «"Ошибка ввода элемента массива. Завершение программы... » | 2 2  1 0  А Б | исключение: «"Ошибка ввода элемента массива. Завершение программы... » |
| 1. Реализация | | | |
| 2.1 Массивы из файла | Примечание:  Массивы из файла считываются в один большой встроенный массив, занимая лишь его часть. Таким образом мы добиваемся работы именно со встроенными массивами, а не с динамическими. |  |  |
| 1. Тесты | | | |
| 3.1 |  | 3 3  1 0 1  1 1 1  0 1 1 | 2 |
| 3.2 |  | 2 2  0 0  1 1 | 2 |
| 3.3 |  | 2 2  1 0  1 0 | 1 |

Код:

#include <iostream>

#include <fstream>

#include <iomanip>

#define FILE\_INPUT\_REGULAR "regular\_arrays\_in.txt"

#define FILE\_OUTPUT\_REGULAR "regular\_arrays\_out.txt"

int getCountZeroInColumnAny(const int\* array, int nRow, int nColumn) {

int counter = 0;

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

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

if (\*(array + i\*nRow + j) == 0) {

counter += 1;

break;

}

}

}

return counter;

}

void read2dArray(int\* destination, const int nRow, const int nColumn, std::ifstream& ifs) {

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

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

if (!ifs.good()) {

throw "Ошибка ввода элемента массива. Завершение программы...";

}

ifs >> \*(destination + nRow \* i + j);

}

}

}

void print2dArray(const int\* array, const int nRow, const int nColumn, std::ostream& os = std::cout) {

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

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

os << std::setw(4) << \*(array + nRow \* i + j);

}

os << '\n';

}

}

int main() {

setlocale(LC\_ALL, "Russian");

std::cout << "Встроенные массивы из файла: \n\n";

const int BIG\_SIZE = 100;

int matrix[BIG\_SIZE][BIG\_SIZE] = {};

try {

std::ifstream ifs(FILE\_INPUT\_REGULAR);

std::ofstream ofs(FILE\_OUTPUT\_REGULAR);

if (!ifs.good()) {

throw "Файл ввода недоступен. Завершение программы...";

}

if (!ofs.good()) {

throw "Файл вывода недоступен. Завершение программы...";

}

while (!ifs.eof()) {

int nRow = 0;

ifs >> nRow;

if (ifs.eof() || !ifs.good()) {

throw "Ошибка ввода размера массива. Завершение программы...";

}

else if (nRow <= 0) {

throw "Размер массива задан некорректно. Завершение программы...";

}

int nColumn = 0;

ifs >> nColumn;

if (ifs.eof() || !ifs.good()) {

throw "Ошибка ввода размера массива. Завершение программы...";

}

else if (nColumn <= 0) {

throw "Размер массива задан некорректно. Завершение программы...";

}

read2dArray(matrix[0], nRow, nColumn, ifs);

print2dArray(matrix[0], nRow, nColumn, std::cout);

int result = getCountZeroInColumnAny(matrix[0], nRow, nColumn);

std::cout << "result: " << result << "\n\n";

ofs << result << '\n';

}

}

catch (const char\* what) {

std::cerr << what;

return 1;

}

return 0;

}

1. Массивы в динамической памяти

Общая постановка задачи

Напишите функцию для работы с **двумерным массивом** – квадратной матрицей в динамической памяти. Функция, должна иметь два параметра – адрес матрицы (целые числа), размерность матрицы и заполнить эту натуральными числами от 1 до N\*N (N – размерность) по спирали, начинающейся в верхнем левом углу и закрученной по часовой стрелке.

Элементы матрицы – целые числа.

Ввод и вывод матрицы выполнять в отдельных функциях.

**Входные данные:** размерность матрицы, **ввести из файла**.

**Выходные данные**: итоговая матрица **вывести в файл**.

Детальные требования:

|  |  |  |  |
| --- | --- | --- | --- |
| 1. Ввод | | | |
| 1.1 Размерность | 1.1 Размерность больше нуля, иначе исключение. | 0  -1 | Сообщение: «"Размер массива задан некорректно. Завершение программы...» |
| 1.2 Файлы ввода \ выводы | Файлы доступны программе, иначе исключение | 1. Нет файла ввода: 2. Файл вывода недоступен для записи | 1. Сообщение: «"Файл ввода недоступен. Завершение программы..."» 2. «"Файл вывода недоступен. Завершение программы..."» |
| 1. Реализация | | | |
| 2.1 Динамические массивы | Примечание:  Динамический массив удаляется, как только исчезает необходимость его существования, в целях экономии памяти. При удалении стоит учитывать его двумерность. |  |  |
| 1. Тесты | | | |
| 3.1 |  | 1 | 1 |
| 3.2 |  | 2 | 1 2  4 3 |
| 3.3 |  | 3 | 1 2 3  8 9 4  7 6 5 |
| 3.4 |  | 4 | 1 2 3 4  12 13 14 5  11 16 15 6  10 9 8 7 |
| 3.5 |  | 5 | 1 2 3 4 5  16 17 18 19 6  15 24 25 20 7  14 23 22 21 8  13 12 11 10 9 |

Код:

#include <iostream>

#include <fstream>

#include <iomanip>

#define FILE\_INPUT\_DYNAMIC "dynamic\_arrays\_in.txt"

#define FILE\_OUTPUT\_DYNAMIC "dynamic\_arrays\_out.txt"

void fillSpiralClockwise(int\*\* destinationArray, int size) {

int nextN = 1;

int margin = 0;

int halfsize = size / 2;

while (margin < halfsize) {

for (int i = 0 + margin; i < size-1-margin; i++) {

destinationArray[0 + margin][i] = nextN;

nextN++;

}

for (int j = 0+ margin; j < size-1-margin; j++) {

destinationArray[j][size - 1 - margin] = nextN;

nextN++;

}

for (int i = size - 1 - margin; i > 0+margin; i--) {

destinationArray[size - 1 - margin][i] = nextN;

nextN++;

}

for (int j = size - 1 - margin; j >= 1+margin; j--) {

destinationArray[j][0 + margin] = nextN;

nextN++;

}

margin += 1;

}

if (halfsize\*2 != size){

destinationArray[margin][margin] = nextN; // middle

}

//std::clog << "final: " << nextN << "\n\n";

}

void printSquareMatrix(int \*\* array, const int size, std::ostream& os = std::cout) {

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

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

os << std::setw(4) << array[i][j];

}

os << '\n';

}

os << '\n';

}

int main() {

setlocale(LC\_ALL, "Russian");

std::cout << "Встроенные массивы из файла: \n\n";

try {

std::ifstream ifs(FILE\_INPUT\_DYNAMIC);

std::ofstream ofs(FILE\_OUTPUT\_DYNAMIC);

if (!ifs.good()) {

throw "Файл ввода недоступен. Завершение программы...";

}

if (!ofs.good()) {

throw "Файл вывода недоступен. Завершение программы...";

}

while (!ifs.eof()) {

int size = 0;

ifs >> size;

if (!ifs.good() && !ifs.eof()) {

throw "Ошибка ввода размера массива. Завершение программы...";

}

else if (size <= 0) {

throw "Размер массива задан некорректно. Завершение программы...";

}

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

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

{

matrix[i] = new int[size];

}

fillSpiralClockwise(matrix, size);

printSquareMatrix(matrix, size, std::cout);

printSquareMatrix(matrix, size, ofs);

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

delete[] matrix[i];

}

delete[] matrix;

}

}

catch (const char\* what) {

std::cerr << what;

return 1;

}

return 0;

}