Министерство образования и науки Российской Федерации

Севастопольский государственный университет

Кафедра ИС

Отчет

по лабораторной работе № 7

«Обработка двумерных массивов

с помощью функций»

Выполнил

ст. гр. И12д

Серегин А.В.

Проверил:

асс. Забаштанский А.К.

Севастополь

2015

1. Цель работы

Изучить основные принципы работы с массивами в языках С/С++.

Исследовать способы передачи параметров в функции.

2.Варинты заданий

Каждый пункт нижеприведенного задания оформить в виде функции. Все необходимые данные для функций должны передаваться им в качестве параметров. Ввод-вывод данных и результатов также организовать с помощью соответствующих функций.

Вариант 20:

Дана целочисленная квадратная матрица. Определить:

1) количество отрицательных элементов в тех строках, которые содержат хотя бы один нулевой элемент;

2) номера строк и столбцов всех седловых точек матрицы.

3. Структурная схема

На рисунках ниже представлена структурная схема алгоритма программы.



Рисунок 1 – Функция Print\_Negative.



Рисунок 2 – Функция Init\_matrix.



Рисунок 3 – Функция Free\_mem.



Рисунок 4 – Функция MinInStr.



Рисунок 5 – Функция Print\_Saddles.



Рисунок 6 – Функция main.

5. Код Программы

#include <iostream>

#include <ctime>

using namespace std;

//Инициализация и ввод элементов матрицы

int\*\* Init\_matrix(const int& n, const int& m)

{

int \*\*arr = new int\*[n]; // выделение памяти под массив

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

{

arr[i] = new int[m];

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

cin >> arr[i][j];

}

return arr;

}

//удаление матрицы из памяти

void Free\_mem(int \*\*arr, const int& n)

{

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

delete[] arr[i];

delete[] arr;

}

//Вывод количества отрицательных элементов в нулевых строках

void Print\_Negative(int\*\* arr, const int& n, const int& m)

{

bool Enable = false; //наличие нулевых строк

for (int i(0); i < n; i++) //поиск нулевых строк

{

bool flag = false;

unsigned int counter = 0; //счетчик отр. эл-тов

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

{

if (arr[i][j] < 0) counter++;

if (arr[i][j] == 0) flag = true;

}

if (flag)

{

Enable = true;

cout << "В нулевой строке " << i+1 << ' '

<< counter << " отрицательных чисел." << endl;

}

}

if (!Enable)

cout << "Нулевых строк нет." << endl;

}

//нахождение минимального элемента в несортированной строке

int MinInStr(int\* arr, const int& n)

{

int min = arr[0];

for (int i(1); i < n; i++)

if (min>arr[i]) min = arr[i];

return min;

}

//вывод седловых точек матрицы

void Print\_Saddles(int\*\* arr, const int& n, const int& m)

{

bool Enable = false; //наличие седловых точек

int\* max\_in\_col = new int[n]; //массив макс. эл-тов столбцов

int\* min\_in\_str = new int[m]; //массив мин. эл-тов строк

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

min\_in\_str[i] = MinInStr(arr[i], m);

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

{

max\_in\_col[j] = arr[0][j];

for (int i(1); i < n; i++)

if (max\_in\_col[j] < arr[i][j]) max\_in\_col[j] = arr[i][j];

}

for (int i(0); i < n; i++) //поиск седловых точек

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

if (min\_in\_str[i] == max\_in\_col[j])

{

Enable = true;

cout << "Седловая точка: " << i+1 << ' ' << j+1 << endl;

}

if (!Enable)

cout << "Седловых точек нет" << endl;

delete[] max\_in\_col; //удаление массивов

delete[] min\_in\_str;

}

// функция main

int main()

{

srand(time(NULL));

setlocale(LC\_ALL, "Russian"); //отображение кириллицы

int n, m; // размеры матрицы

cout << "Введите размеры матрицы:" << endl;

cin >> m >> n;

cout << "Введите матрицу NxM:" << endl;

int \*\*arr=Init\_matrix(n,m); // двумерная матрица

Print\_Negative(arr, n, m);

Print\_Saddles(arr, n, m);

Free\_mem(arr,n);

system("pause");

return 0;

}

6. Тестовые примеры

На рисунке изображен тестовый пример работы программы.

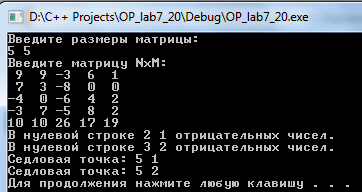


Рисунок 7 – Тест программы.

Выводы

Благодаря функциям можно легко организовать свою программу, разбив её на множество функцию. Каждую функцию в отдельности легче отладить, в случае ошибок в них, их легко исправить или даже заменить саму функцию. Также функции дают возможность уменьшить число повторений одной и той же последовательности операторов, что значительно уменьшает размеры и упрощает процесс написание кода.