**САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО**

**ИНСТИТУТ КОМПЬЮТЕРНЫХ НАУК И ТЕХНОЛОГИЙ**

**ВЫСШАЯ ШКОЛА ПРОГРАММНОЙ ИНЖЕНЕРИИ**

**Контрольная работа №2**

**Вариант 11**

**по предмету «Алгоритмизация и программирование»**

**Выполнил: студент гр. 5130904/30002 Севостьянова А.В.**

**Руководитель Череповский Д.К.**

**Санкт-Петербург**

**23 октября 2023 г.**

1. **Общая постановка задачи**

Реализовать функции для встроенного и динамического массива, подсчитывающих количество строк, содержащих повторяющиеся элементы.

Входные данные: количество строк и столбцов динамического массива, элементы динамического массива, элементы статического массива

Выходные данные: динамический массив, статический массив, количество строк с повторяющимися элементами с в динамическом массиве, количество строк с повторяющимися элементами в статическом массиве.

1. **Код программы**

**Main.cpp**

#include"functions.h"

int main()

{

setlocale(LC\_ALL, "ru");

int\*\* arr = nullptr;

bool test = false;

int rows = 0;

int cols = 0;

string value = "";

int arr\_static[ROWS][COLS] = {0};

while (!test)

{

try

{

test = true;

cout << "Введите количество строк динамического масства:\t";

cin >> value;

rows = checkInput(value);

if (rows <= 0) throw runtime\_error("Количество строк динамического массива должно принимать положительное значение");

cout << "Введите количество столбцов динамического масства:\t";

cin >> value;

cols = checkInput(value);

if (cols <= 0) throw runtime\_error("Количество столбцов динамического массива должно принимать положительное значение");

}

catch (runtime\_error& ex)

{

cerr << "\n" << ex.what() << "\n";

test = false;

}

}

try

{

arr = allocateMemory(arr, rows, cols);

cout << "\nЗаполнение динамического массива\n ";

inArray(arr, rows, cols);

cout << "\nЗаполнение статического массива\n ";

inArrayForStatic(arr\_static,ROWS, COLS);

cout << "\nПолученный динамический массив:\n";

outArray(arr, rows, cols);

cout << "\n\nПолученный статический массив массив:\n";

outArrayForStatic(arr\_static, ROWS, COLS);

cout << "\n\nКоличество сторок с повторяющимися элементами в динамическом массиве =\t"<<countOfSameElements(arr, rows, cols)<<"\n";

cout << "\nКоличество сторок с повторяющимися элементами в статическом массиве =\t" << countOfSameElementsForStatic(arr\_static, ROWS, COLS) << "\n";

}

catch (bad\_alloc& ex)

{

cerr << "\n" << ex.what() << "\n";

deleteMemory(arr, rows, cols);

exit(EXIT\_FAILURE);

}

arr = deleteMemory(arr, rows, cols);

return 0;

}

**functions.h**

#pragma once

#include<iostream>

#include<regex>

#include<string>

using namespace std;

#define ROWS 2

# define COLS 3

const string ERROR = "Непредвиденный символ";

int checkInput(const string &value);

void inArray(int\*\* arr, int rows, int cols);

void outArray(int\*\* arr, int rows, int cols);

int\*\* allocateMemory(int\*\* arr, int rows, int cols);

int\*\* deleteMemory(int\*\* arr, int rows, int cols);

int countOfSameElements(int\*\* arr, int rows, int cols);

void inArrayForStatic(int(&arr)[ROWS][COLS], int rows, int cols);

void outArrayForStatic(int(&arr)[ROWS][COLS], int rows, int cols);

int countOfSameElementsForStatic(int (&arr)[ROWS][COLS], int rows, int cols);

**functions.cpp**

#include "functions.h"

int checkInput(const string& value)

{

regex regular("-?\\d+");

if (regex\_match(value, regular)) return stoi(value);

else throw runtime\_error(ERROR);

}

void inArray(int\*\* arr, int rows, int cols)

{

bool test = true;

string value = "";

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

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

{

do

{

cout << "Введите элемент массива\t";

cin >> value;

try

{

arr[i][j] = checkInput(value);

test = true;

}

catch (runtime\_error& ex)

{

cerr << ex.what() << "\n";

test = false;

}

} while (!test);

}

}

void inArrayForStatic(int(&arr)[ROWS][COLS], int rows, int cols)

{

bool test = true;

string value = "";

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

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

{

do

{

cout << "Введите элемент массива\t";

cin >> value;

try

{

arr[i][j] = checkInput(value);

test = true;

}

catch (runtime\_error& ex)

{

cerr << ex.what() << "\n";

test = false;

}

} while (!test);

}

}

void outArray(int\*\* arr, int rows, int cols)

{

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

{

cout << "\n" << i << " строка\t";

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

cout << arr[i][j] << "\t";

}

}

void outArrayForStatic(int(&arr)[ROWS][COLS], int rows, int cols)

{

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

{

cout <<"\n" << i << " строка\t";

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

cout<< arr[i][j] << "\t";

}

}

int\*\* allocateMemory(int\*\* arr, int rows, int cols)

{

arr = new int\* [rows];

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

arr[i] = new int[cols] {0};

return arr;

}

int\*\* deleteMemory(int\*\* arr, int rows, int cols)

{

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

delete arr[i];

delete[] arr;

return arr;

}

int countOfSameElements(int\*\* arr, int rows, int cols)

{

int count = 0;

bool test = false;

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

{

test = false;

for (int j = 0; j < cols-1; j++)

for (int k = j + 1; k < cols; k++)

{

if (arr[i][j] == arr[i][k])

test = true;

if (test) break;

}

if (test) count++;

}

return count;

}

int countOfSameElementsForStatic(int(&arr)[ROWS][COLS], int rows, int cols)

{

int count = 0;

bool test = false;

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

{

test = false;

for (int j = 0; j < cols - 1; j++)

for (int k = j + 1; k < cols; k++)

{

if (arr[i][j] == arr[i][k])

test = true;

if (test) break;

}

if (test) count++;

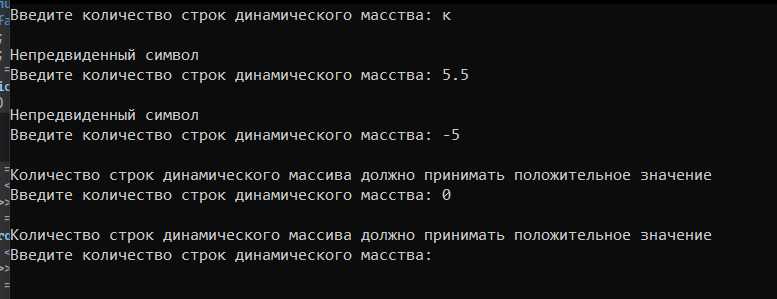
}

return count;

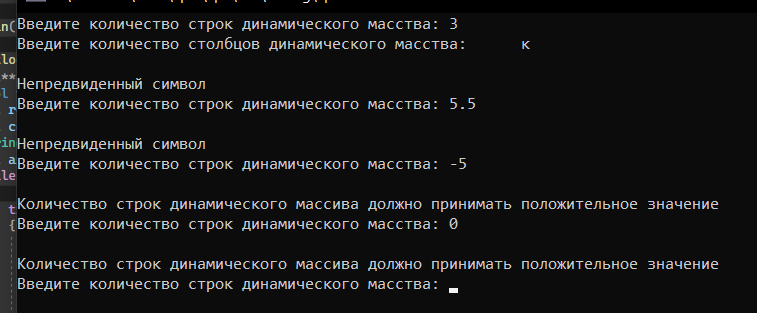
}

1. **Работа программы**

**Количество строк динамического массива принимает неверное значение**

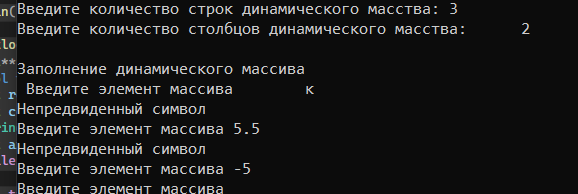
****

**Количество столбцов динамического массива принимает неверное значение**

****

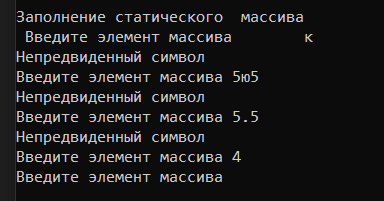
**При верном определении количества строк и столбцов динамического массива происходит его заполнение пользователем**

**Ввод неверных элементов массива**

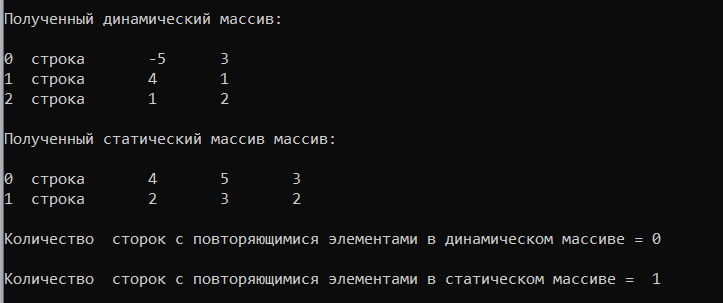
****

**После заполнения динамического массива происходит заполнение статического массива пользователем**

**Ввод неверных элементов массива**

****

**После заполнения оба массива и количество строк с повторяющимися элементам в каждом из них выводятся на экран выводятся на экран**

****