ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ассистент |  |  |  | М. А. Мурашова |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ |
|  |
| по курсу: ОСНОВЫ ПРОГРАММИРОВАНИЯ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4134к |  |  |  | Н.А. Костяков |
|  |  |  | подпись, дата |  | инициалы, фамилия |

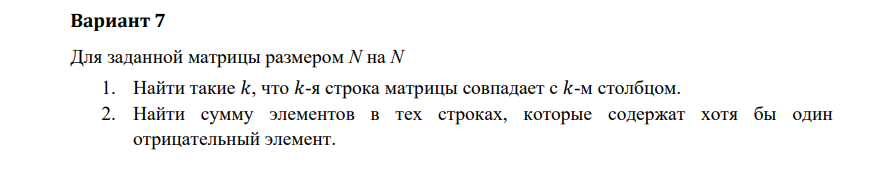
Санкт-Петербург 2022

***Цель работы***

Целью работы является изучение структуры данных двумерный массив.

***Задание на лабораторную работу***

Задания на лабораторную работу приводятся в каждом варианте. При написании программ можно использовать как динамические, так и нединамические массивы. Размерность последних задаѐтся именованной константой



***Функции использованные в программе***

***Имя*** serach()

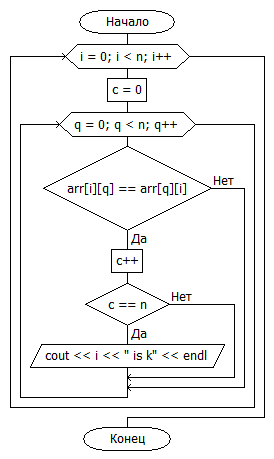
***Назначение*** Найти такие 𝑘, что 𝑘-я строка матрицы совпадает с 𝑘-м столбцом.

***Входные*** данные двумерный массив и его размер

***Выходные*** данные значение индекса строки

***Побочные эффекты нет***

***Блоксхема***

******

***Псевдокод***

***Для каждого I до n***

С= 0

Для каждого q до n

Если элемент строки совпадает с элементом столбца

С++

Если с==т

Вывод i

**Имя** fill()

**Назначение:** ввод с проверкой

**Входные данные :** запросы с клавиатуры

**Выходные данные:** число в double

**Побочные эффекты отсутствуют**

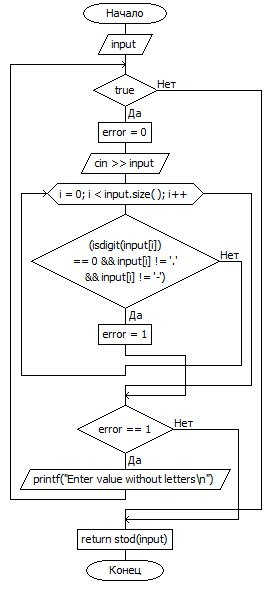
**Псевдокод**

Пока не введут значение без букв

Запрос ввода с клавиатуры

Вывод double

**Блок-схема**

****

***Имя*** sum\_m()

***Назначение*** Найти сумму элементов в тех строках, которые содержат хотя бы один отрицательный элемент.

***Входные данные*** двумерный массив и его размер

***Выходные данные*** double сумма

***Побочные эффекты нет***

***Псевдокод***

Для каждого I меньше n

Сумма= 0

Flag = 0

Для каждого q < n

Еасли элемент таблицы меньше нуля

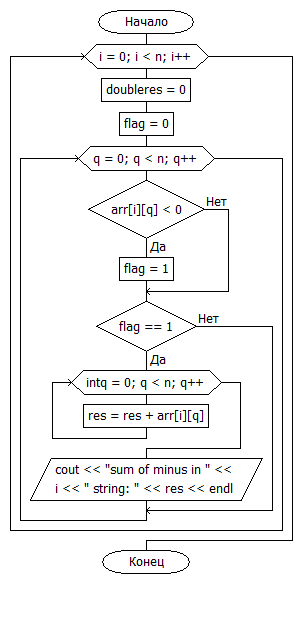
Flag =1

Если флаг = 1

Res = res + строка таблицы

Вывод резов

***Блоксхема***

******

***Имя main()***

***Назначение*** диалог с пользователем

***Входные*** данные массив

***Выходные –*** решение по варианту

***Побочных нет***

***Псевдокод***

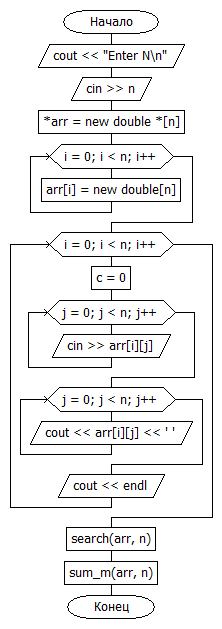
Ввод данных о массиве

Действия с массивом по варианту

Удаление массива

Проверка утечек памяти

***Блок схема***

******

***Листинг всей программы***

#define \_CRTDBG\_MAP\_ALLOC

#include <stdlib.h>

#include <crtdbg.h>

#ifdef \_DEBUG

#ifndef DBG\_NEW

#define DBG\_NEW new ( \_NORMAL\_BLOCK , \_\_FILE\_\_ , \_\_LINE\_\_ )

#define newDBG\_NEW

#endif

#endif

#include <iostream>

#include<string>

using namespace std;

void search(double\* arr[], int n) {

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

int c = 0;

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

if (arr[i][q] == arr[q][i]) {

c++;

if (c == n) cout << i <<" is k" << endl;

}

}

}

}

void sum\_m(double\* arr[], int n){

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

double res=0;

bool flag = 0;

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

if (arr[i][q] < 0) {

flag = 1;

}

if (flag == 1) {

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

res = res + arr[i][q];

}

cout <<"sum with minus in "<<i<<" string: " << res << endl;

res = 0;

flag = 0;

break;

}

}

}

}

double fill() {

std::string input;

while (true) //проверка введенного числа

{

bool error = 0;

std::cin >> input;

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

if ((isdigit(input[i]) == 0 && input[i] != '.' && input[i] != '-')) {

error = 1;

break;

}

}

if (error == 1) {

printf("Enter value without letters\n");

}

else

{

break;

}

}

return std::stod(input);

}

int main()

{

int n= -1;

while (n <= 0) {

cout << "Enter N\n";

n = fill();

}

double\*\* arr = new double\*[n];

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

arr[i] = new double[n];

}

cout << "Fill all your strings\n";

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

{

int c = 0;

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

{

cout << "enter [" << i << "] [" << j << "] element: ";

arr[i][j]=fill();

}

}

cout << endl;

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

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

cout << arr[i][q] << " ";

}

cout << endl;

}

cout << endl;

search(arr, n);

sum\_m(arr, n);

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

delete[] arr[i];

}

delete[] arr;

// Для обнаружения утечек памяти

\_CrtSetReportMode(\_CRT\_WARN, \_CRTDBG\_MODE\_FILE);

\_CrtSetReportFile(\_CRT\_WARN, \_CRTDBG\_FILE\_STDOUT);

\_CrtSetReportMode(\_CRT\_ERROR, \_CRTDBG\_MODE\_FILE);

\_CrtSetReportFile(\_CRT\_ERROR, \_CRTDBG\_FILE\_STDOUT);

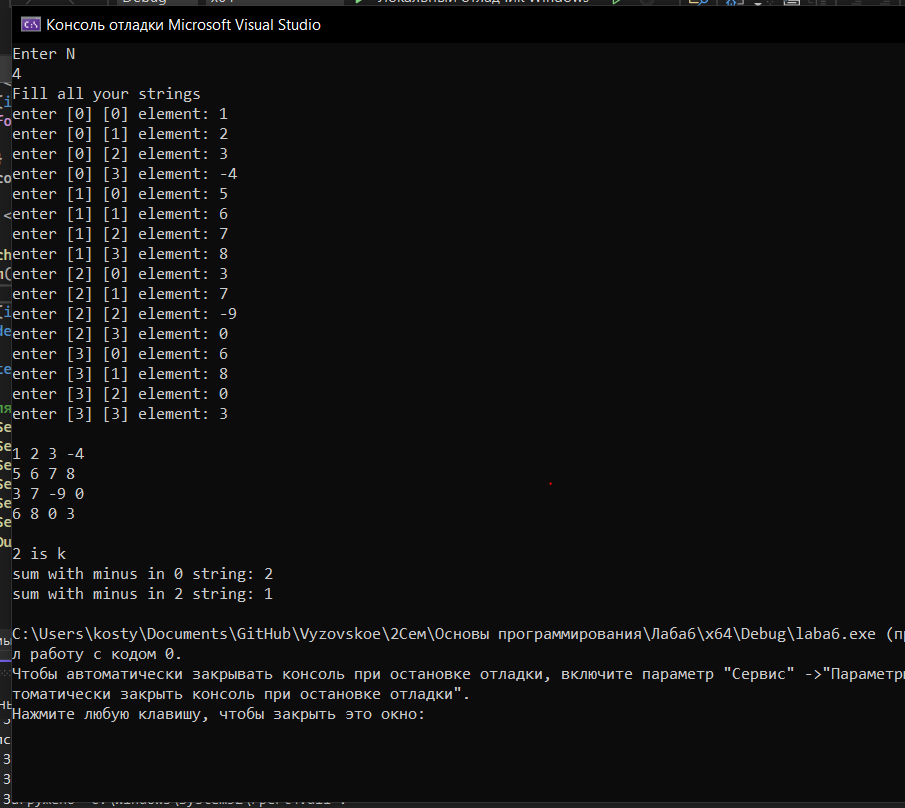
\_CrtSetReportMode(\_CRT\_ASSERT, \_CRTDBG\_MODE\_FILE);

\_CrtSetReportFile(\_CRT\_ASSERT, \_CRTDBG\_FILE\_STDOUT);

\_CrtDumpMemoryLeaks();

}

***Результаты работы программы***

.

***Выводы***

Я изучил работы с матрицами и решил поставленные задачи

Из достоинств могу отметить

Диалог с пользователем

Высокая скорость

Проверка входных данных

Минусов нет