ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

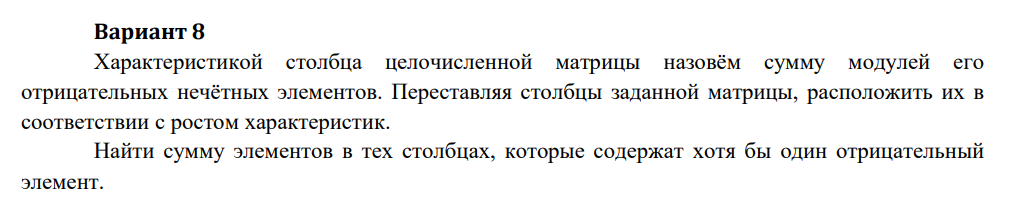
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ассистент |  |  |  | М. А. Мурашова |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ |
| Обработка числовых матриц |
| по курсу: ОСНОВЫ ПРОГРАММИРОВАНИЯ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4135К |  |  |  | Столяров Н.С. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2022

**Цель работы:** Целью работы является изучение структуры данных двумерный массив.  
**Задачи работы:**  
  
**Описание функций:  
1)Имя: main**   
**Назначение:** Вычисление номера первой строки содержащей положительный элемент. Удаление столбцов и строк заполненными нулями.  
**Входные данные:** Высота, ширина и массив.   
**Выходные данные:** Индекс и массив.  
**Побочный эффект:** отсутствует.   
**Тестовые данные:**

|  |  |  |  |
| --- | --- | --- | --- |
| Ширина | Высота | Массив | Ответ |
| 4 | 4 | 1 2 3 4  4 3 2 1  -4 -1 -9 54  2 -3 4 -7 | Сумма элементов в тех столбцах, которые содержат хотя бы один отрицательный элемент: 56 1 2 4 3  4 3 1 2  -4 -1 54 -9  2 -3 -7 4 |

**Прототип:** int main()

|  |  |
| --- | --- |
| **Псевдокод:** | **Блок-схема:** |
| Ввод ширины и высоты массивами Ввод массива получение характеристики  Вывод массива с характеристикой  Сортировка массива по характеристике  Вывод сортированного массива | C:\Users\nikit\Desktop\main.png |

**Имя: draw\_float\_double\_array**  
**Назначение:** Вывод двумерного массива в консоль.  
**Входные данные:** arr, size\_x, size\_y.   
**Выходные данные:** массив.  
**Побочный эффект:** отсутствует.   
**Прототип:** draw\_float\_double\_array ()

|  |  |
| --- | --- |
| **Псевдокод:** | **Блок-схема:** |
| Вывод массива в консоль | C:\Users\nikit\AppData\Local\Microsoft\Windows\INetCache\Content.Word\diagram.png |

**Имя: read\_size\_arr**  
**Назначение:** Проверка на размер массива.  
**Входные данные:** promt(комментарий).   
**Выходные данные:** size(размер массива).  
**Побочный эффект:** отсутствует.   
**Прототип:** read\_size\_arr ()

|  |  |
| --- | --- |
| **Псевдокод:** | **Блок-схема:** |
| Бесконечный цикл  Ввод значения  Если значение больше 1, то  Останавливаем цикл  Если нет  Заново вводим | C:\Users\nikit\Desktop\diagram (1).png |

**Имя: get\_info\_arr**  
**Назначение:** Вычисление характеристики столбцов массива.  
**Входные данные:** arr, size\_x, size\_y   
**Выходные данные:** info  
**Побочный эффект:** отсутствует.   
**Прототип:** double \*get\_info\_arr ()

|  |  |
| --- | --- |
| **Псевдокод:** | **Блок-схема:** |
| * Создаём новый массив, который будет хранить характеристику столбцов * Цикл по всему массиву * Если в столбце есть отрицательный и нечётный элемент, то прибавляем его модуль к характеристике его столбца * После массива возвращаем массив с характеристикой |  |

**Имя: get\_summ**  
**Назначение:** Вычисление суммы в тех столбцах, которые содержат хотя бы один отрицательный элемент.  
**Входные данные:** arr, size\_x, size\_y.   
**Выходные данные:** summ.  
**Побочный эффект:** отсутствует.   
**Прототип:** double get\_summ ()

|  |  |
| --- | --- |
| **Псевдокод:** | **Блок-схема:** |
| * Цикл по всему массиву * Если в столбце нашли отрицательный массив, то суммируем весь этот столбец * После цикла возвращаем сумму |  |

**ЛИСТИНГ КОДА:  
/\***

**Задания на лабораторную работу приводятся в каждом варианте. При написании программ**

**можно использовать как динамические, так и нединамические массивы. Размерность последних**

**задаѐтся именованной константой**

**Вариант 8**

**Характеристикой столбца целочисленной матрицы назовѐм сумму модулей его**

**отрицательных нечѐтных элементов. Переставляя столбцы заданной матрицы, расположить их в**

**соответствии с ростом характеристик.**

**Найти сумму элементов в тех столбцах, которые содержат хотя бы один отрицательный**

**элемент.**

**\*/**

**#define RANDOM\_NUMS false**

**#include <iostream>**

**using namespace std;**

**#include "libs/lib.h"**

**#include <cmath>**

**#include <time.h>**

**// работа с массивами**

**#include "libs/array.h"**

**#include "arr.h"**

**int read\_size\_arr(const char \*promt = "") {**

**int size;**

**while (true) {**

**cout << promt;**

**size = read\_int();**

**if (size > 0) {**

**break;**

**} else {**

**cout << "Размер должен быть больше 1." << endl;**

**}**

**}**

**return size;**

**}**

**int main() {**

**// смена кодировки**

**system("chcp 65001");**

**// очистка терминала**

**//clear\_scr();**

**// рандом**

**srand(time(NULL));**

**int x, y;**

**int size\_x, size\_y;**

**// ввод размеров массива**

**while (true) {**

**cout << "Ширина массива: ";**

**size\_x = read\_size\_arr();**

**cout << "Высота массива: ";**

**size\_y = read\_size\_arr();**

**if (size\_x < 1 || size\_y < 1) {**

**cout << "Массив не может содержать 0 элементов." << endl;**

**} else break;**

**}**

**// создаём новый массив**

**double \*\*arr = (double\*\*)malloc(size\_y \* sizeof(double\*));**

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

**arr[i] = (double\*)malloc(size\_x \* sizeof(double));**

**}**

**// ввод значений массива**

**arr = read\_double\_arr(arr, size\_x, size\_y, RANDOM\_NUMS);**

**draw\_line(20);**

**cout << "Сумма элементов в тех столбцах, которые содержат хотя бы один отрицательный элемент: " << get\_summ(arr, size\_x, size\_y) << endl;**

**// получение характеристики**

**double \*info = get\_info\_arr(arr, size\_x, size\_y);**

**// вывод массива и иформации о нём**

**draw\_float\_array(info, size\_x);**

**draw\_line(20);**

**draw\_float\_double\_array(arr, size\_x, size\_y);**

**draw\_line(20);**

**double temp;**

**// Сортировка массива пузырьком**

**for (int i = 0; i < size\_x - 1; i++) {**

**for (int j = 0; j < size\_x - i - 1; j++) {**

**if (info[j] > info[j + 1]) {**

**// меняем элементы местами**

**temp = info[j];**

**info[j] = info[j + 1];**

**info[j + 1] = temp;**

**// меняем столбцы местами**

**for (int y = 0; y < size\_y; y++) {**

**temp = arr[y][j];**

**arr[y][j] = arr[y][j + 1];**

**arr[y][j + 1] = temp;**

**}**

**}**

**}**

**}**

**draw\_line(20);**

**draw\_float\_array(info, size\_x);**

**draw\_line(20);**

**draw\_float\_double\_array(arr, size\_x, size\_y);**

**draw\_line(20);**

**// очистка памяти**

**for(y = 0; y < size\_y; y++) {**

**free(arr[y]);**

**}**

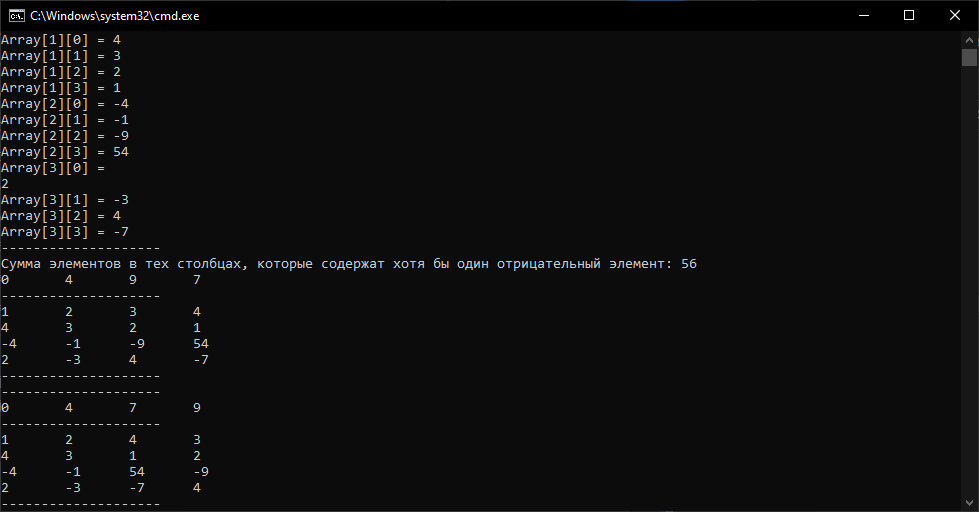
**free(arr);**

**free(info);**

**return 0;**

**}**

**Пример выполнения программы:**



Видно, что результаты расчётов совпадают с тестовыми данными.

**Анализ результатов и выводы:**

В ходе этой лабораторной работы я устал перебирать динамические массивы.  
Из плюсов можно отметить её работоспособность.

Из минусов ужасный ввод элементов массива.