ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

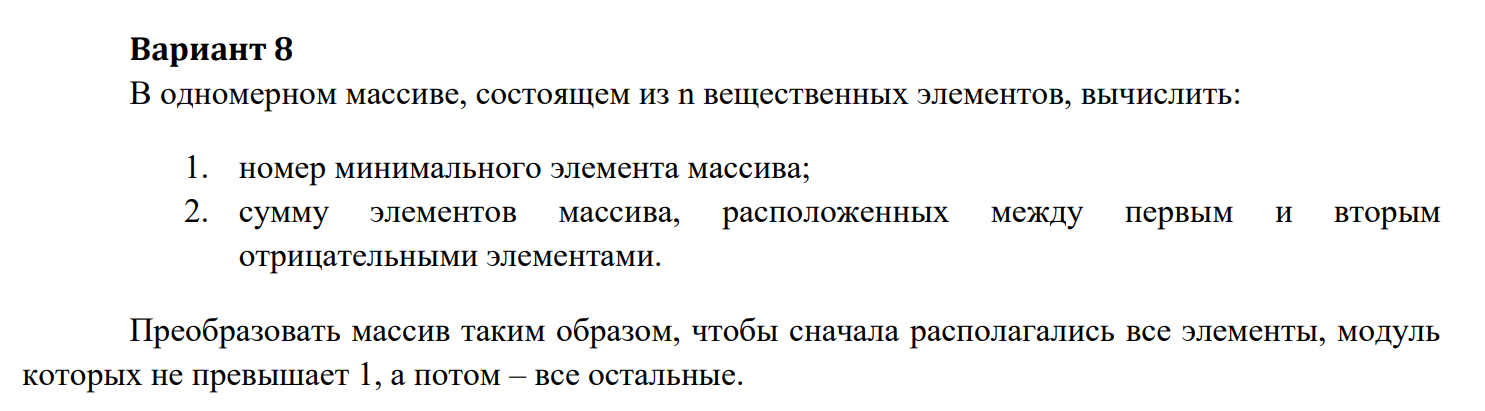
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ассистент |  |  |  | М. А. Мурашова |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

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

РАБОТУ ВЫПОЛНИЛ

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

Санкт-Петербург 2022

**1.Цель работы:** Целью работы является изучение структуры данных одномерный массив.  
**2.Задачи работы:** Задания на лабораторную работу приводятся в каждом варианте. При написании программ можно использовать как динамические, так и нединамические массивы. Размерность последних задаѐтся именованной константой.  
  
**3.Описание функций:  
1)Имя: main**   
**Назначение:** Нахождение номера минимального элемента массива, суммы элементов расположенных между первым и вторым отрицательными элементами и преобразование массива  
**Входные данные:** нет.   
**Выходные данные:** нет.  
**Побочный эффект:** отсутствует.   
**Тестовые данные:**

|  |  |
| --- | --- |
| Массив | Ответ |
| 2 -2 3 -6 3 2 0 4 3 5 | * Номер первого минимального элемента массива: 3 * Cумма элементов массива, расположенных между   первым и вторым отрицательными элементами: 3   * 0 2 -2 3 -6 3 2 4 3 -5 |

**Прототип:** int main()

|  |  |
| --- | --- |
| **Псевдокод** | **Блок-схема** |
| * Ввод размера массива * Ввод массива * Нахождение и вывод суммы элементов расположенных между первым и вторым отрицательными элементами. * Нахождение и вывод номера минимального элемента массива * Преобразование массива * Вывод массива | **C:\Users\nikit\AppData\Local\Microsoft\Windows\INetCache\Content.Word\main.png** |

**2)Имя: check**  
**Назначение:** Проверка ввода  
**Входные данные:** i(int).  
**Выходные данные:** num.  
**Побочный эффект:** отсутствует.

**Прототип:** double check()

|  |  |
| --- | --- |
| **Псевдокод** | **Блок-схема** |
| Ввод числа  Если число некорректно, то очищаем поток ввода и повторяем ввод заново | C:\Users\nikit\AppData\Local\Microsoft\Windows\INetCache\Content.Word\check.png |

**2)Имя: minimum**  
**Назначение:** Нахождение минимального элемента массива  
**Входные данные:** array, n.   
**Выходные данные:** p.  
**Побочный эффект:** отсутствует.

**Прототип:** int minimum()

|  |  |
| --- | --- |
| **Псевдокод** | **Блок-схема** |
| * Создаём буферную переменную и записываем в неё 0 * Цикл по всему массиву * Если элемент массива меньше буфера, то записываем в буфер этот элемент * После цикла возвращаем буфер | C:\Users\nikit\AppData\Local\Microsoft\Windows\INetCache\Content.Word\minimum.png |

**2)Имя: sum**  
**Назначение:** Сумма элементов массива, расположенных между первым и вторым отрицательными элементами.  
**Входные данные:** array, n, q.   
**Выходные данные:** s.  
**Побочный эффект:** отсутствует.

**Прототип:** double sum()

|  |  |
| --- | --- |
| **Псевдокод** | **Блок-схема** |
| * Находим первый отрицательный элемент массива * Находим второй отрицательный элемент массива * Суммируем все элементы находящиеся между первым и вторым отрицательным элементом * Возвращаем сумму | C:\Users\nikit\AppData\Local\Microsoft\Windows\INetCache\Content.Word\sum.png |

**2)Имя: transform**  
**Назначение:** преобразование массива  
**Входные данные:** array, n, array2.   
**Выходные данные:** нет.   
**Побочный эффект:** отсутствует.

**Прототип:** void transform ()

|  |  |
| --- | --- |
| **Псевдокод** | **Блок-схема** |
| * Цикл по массиву * Если модуль элемента меньше либо равно 1, то записываем его в массив * После цикла запускаем ещё цикл по массиву * Если модуль элемента больше 1, то записываем его в массив |  |

**5.ЛИСТИНГ КОДА:**

// Для обнаружения утечек памяти

#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 <limits>

#include <cmath>

using namespace std;

double check(int i);

int minimum(double \*array, int n);

double sum(double \*array, int n, bool \*q);

void transform(double \*array, int n, double \*array2);

int main() {

//setlocale(LC\_ALL, "Rus");

system("chcp 65001");

int n;

cout << "Введите длину массива: "; cin >> n;

while (cin.fail() || n <= 0) {

cin.clear();

cin.ignore(INT\_MAX, '\n');

cout << "Ой, кажется вы пытаетесь создать несуществующий массив.\nВведите длину массива ещё раз: ";

cin >> n;

cout << endl;

}

cout << "Вводите элементы массива по порядку: \n";

bool q = false;

double\* array = new double[n];

double\* array2 = new double[n];

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

array[i] = check(i);

}

sum(array, n, &q);

cout << "\n\nНомер первого минимального элемента массива: " << minimum(array, n);

if (q) {

cout << "\n\nCумма элементов массива, расположенных между \nпервым и вторым отрицательными элементами: " << sum(array, n, &q);

}

else {

cout << "\n\nCумма элементов массива, расположенных между \nпервым и вторым отрицательными элементами: нет элементов";

}

transform(array, n, array2);

cout << "\n\nПреобразованный массив: ";

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

cout << array2[i] << '\t';

}

cout << endl << endl;

delete[] array;

delete[] array2;

// 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();

return 0;

}

double check(int i) {

double num;

cout << "Arr[" << i << "] = "; cin >> num;

while (cin.fail()) {

cin.clear();

cin.ignore(INT\_MAX, '\n');

cout << "Ой, кажется вы ошиблись. Введите число ещё раз." << endl;

cout << "Array[" << i << "] = "; cin >> num;

}

return num;

}

int minimum(double \*array, int n){

int p = 0;

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

if (array[p] > array[i]) p = i;

}

return p;

}

double sum(double \*array, int n, bool \*q) {

int min = 0, p, max = 0;

double s = 0;

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

if (array[i] < 0) { min = i; break; }

}

p = min + 2;

for (p; p < n; p++) {

if (array[p] < 0) { \*q = true; max = p; break; }

}

for (int i = min + 1; i < max; i++) {

s += array[i];

}

return s;

}

void transform(double\* array, int n, double\* array2){

int i2 = 0;

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

if (abs(array[i]) <= 1) {

array2[i2] = array[i];

i2++;

}

}

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

if (abs(array[i]) > 1) {

array2[i2] = array[i];

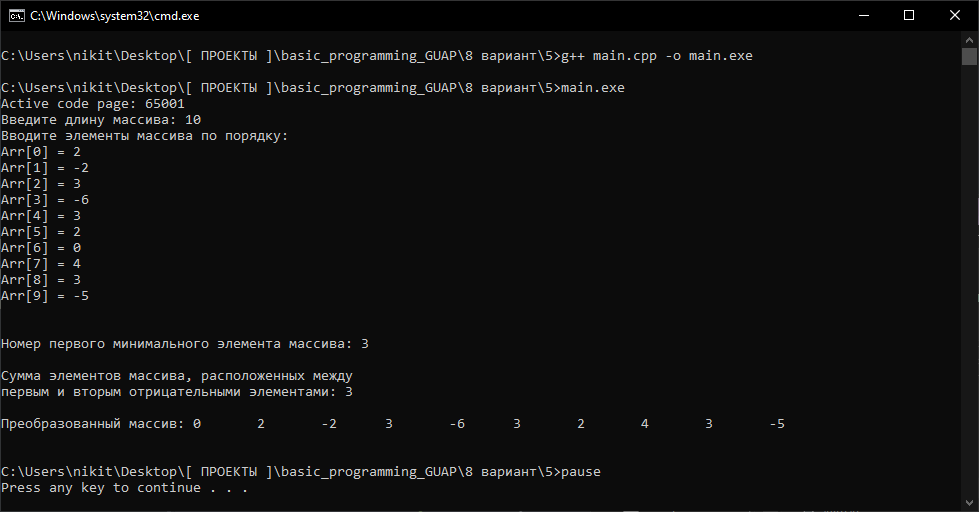
i2++;

}

}

}

**10.Пример выполнения программы:**



Видно, что результаты расчётов совпадают с тестовыми данными.

**11.Анализ результатов и выводы:**

В ходе этой лабораторной работы мы научились работать с одномерными массивами.

Недостатков нет.

К плюсам можно отнести:

• Правильное выполнение поставленной задачи

• Отсутствие ошибок

• Присутствие интерфейса