ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

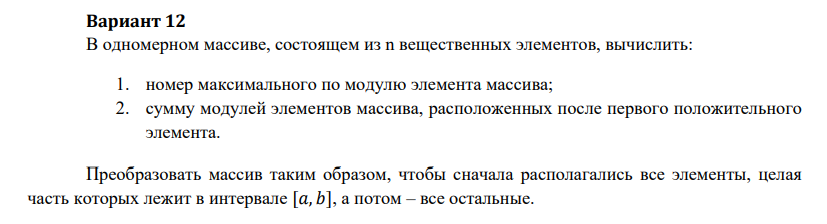
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ассистент |  |  |  | М. А. Мурашова |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

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

РАБОТУ ВЫПОЛНИЛ

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

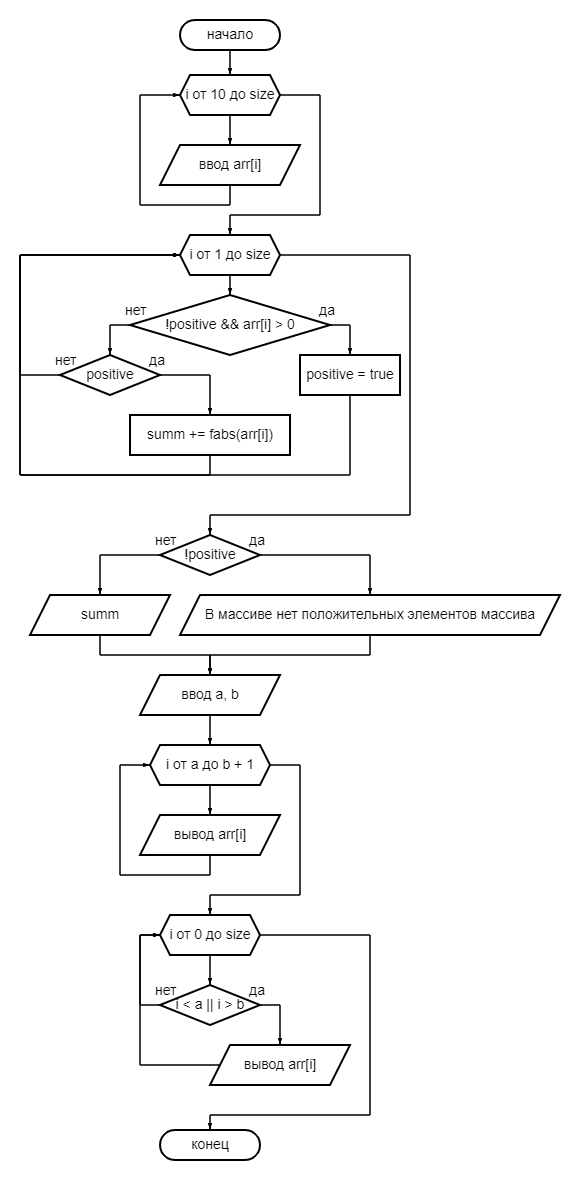
Санкт-Петербург 2022

**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:** Вычисление номера максимального по модулю элемента, суммы модулей элементов массива и преобразование массива.  
**Входные данные:** нет.   
**Выходные данные:** нет.  
**Побочный эффект:** отсутствует.   
**Тестовые данные:**

|  |  |  |  |
| --- | --- | --- | --- |
| Массив | A | B | Ответ |
| 1 2 3 4 5 6 7 8 9 0 | 2 | 5 | 1 2 3 4 5 6 7 8 9 0 3 4 5 6 1 2 7 8 9 0 |

**Прототип:** int main()  
**Псевдокод:**  
Ввод массива arr из 10 элементов  
Вывод массива  
Определение максимального по модулю элемента массива  
Определение суммы элементов после 1 положительного элемента  
Ввод диапазона  
Вывод элементов массива в этом диапазоне  
Вывод остального массива

**Блок-схема:**



**4. Имя: read\_value**  
**Назначение:** проверка переменной a на корректность ввода  
**Входные данные:** Отсутствуют.   
**Выходные данные:** x(double).  
**Побочный эффект:** отсутствует.   
**Прототип:** double read\_value()

|  |  |
| --- | --- |
| **Псевдокод:** | **Блок-схема:** |
| Бесконечный цикл  ввод строки  проверка на пробелы и удаление их  проверка на запятые и замена их на точки  проверка на минус  проверка на лишние символы  если не было проблем, то останавливаем бесконечный цикл­  конец бесконечного цикла  перевод строки в число  вывод числа |  |

**5.ЛИСТИНГ КОДА:**

**main.cpp**/\*

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 12

В одномерном массиве, состоящем из n вещественных элементов, вычислить:

1. номер максимального по модулю элемента массива;

2. сумму модулей элементов массива, расположенных после первого положительного

элемента.

Преобразовать массив таким образом, чтобы сначала располагались все элементы, целая

часть которых лежит в интервале 𝑎, 𝑏 , а потом – все остальные.

\*/

#include <iostream>

using namespace std;

#include "libs/lib.h"

#include <cmath>

// проверка ввода

#include "libs/simple\_char.h"

#include "libs/input\_validation.h"

int main() {

// смена кодировки

system("chcp 65001");

draw\_line(20);

// ввод элементов массива

int size = read\_value("Введите размер массива: ", false, false, false);

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

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

cout << "array [" << i << "] = ";

arr[i] = read\_value("", true, true, false);

}

draw\_line(20);

// выводим массив

draw\_float\_array(arr, size);

draw\_line(20);

// Номер максимального по модулю элемента массива

int max\_id = 0;

for (int i = 1; i < size; i++) {

if (fabs(arr[i]) > fabs(arr[max\_id]))

max\_id = i;

}

cout << "Номер максимального по модулю элемента массива: " << max\_id << endl;

// Сумма модулей после позитивного элемента

bool positive = false;

float summ = 0;

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

if (!positive && arr[i] > 0) {

positive = true;

} else if (positive){

summ += fabs(arr[i]);

}

}

if (!positive)

cout << "В массиве нет положительных элементов массива\n";

else

cout << "Сумма элементов массива после 1 положительного элемента: " << summ << endl;

draw\_line(20);

// ввод диапазона

int a, b, k;

cout << "Введите диапазон [a, b]\n";

while (true) {

a = read\_value("A = ", false, false, false);

b = read\_value("B = ", false, false, false);

// меняем местами числа, если пользователь их перепутал

if (a > b) {

k = a;

a = b;

b = k;

}

// если числа одинаковые, то вводим заново

if (a == b) {

cout << "A и B не могут быть одинаковыми." << endl;

draw\_line(20);

}else if (a >= size || b >= size) {

cout << "A и B не могут быть больше массива." << endl;

draw\_line(20);

}

else {

// если всё ок выходим из цикла

break;

}

}

// пересобираем массив

double \*new\_arr = (double\*) malloc(size \* sizeof(double));

int j = 0;

for (int i = a; i < b + 1; i++){

new\_arr[i] = arr[j++];

}

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

if (i < a || i > b)

new\_arr[i] = arr[j++];

}

// выводим новый массив

draw\_line(20);

draw\_float\_array(new\_arr, size);

draw\_line(20);

// удаление массивов

delete(arr);

delete(new\_arr);

return 0;

}

**lib.h**

#include <iostream>

#include <cmath>

using namespace std;

void draw\_line(int size) {

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

cout << '-';

cout << endl;

}

int random\_int(int a, int b) {

return a + rand() % b;

}

int get\_N(int r, int n) {

int out = r % (n + 1);

if (out < 1)

return out + 1;

else

return out;

}

double read\_double(){

double x;

while ( (scanf("%lf",&x) ) != 1 ) {

printf("Неверное введенное значение, попробуйте еще: ");

while(getchar() != '\n');

}

return x;

}

int read\_int(){

int x;

while ( (scanf("%d",&x) ) != 1 ) {

printf("Неверное введенное значение, попробуйте еще: ");

while(getchar() != '\n');

}

return x;

}

void draw\_float\_array(double \*arr, int size) {

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

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

cout << endl;

}

**input\_validation.h**

/\*

\*/

#include <iostream>

#define SYMB\_LEN\_DOUBLE 11

using namespace std;

double read\_value(

const char \*promt = "",

bool check\_dot = true,

bool check\_minus = true,

bool check\_space = true

) {

// строка введённая пользователем

char \*char\_str;

int len;

// очищенная строка (без пробелов)

char \*clear\_char\_str;

int len\_clear;

// счётсчик для динамического массива

int capacity;

// словарь

char symb[SYMB\_LEN\_DOUBLE] = "1234567890";

int i, j;

// флаг состояния

bool ok = false;

// флаги ошибок

bool error\_dot = false;

bool error\_minus = false;

// флаг на существование точки

bool dot = false;

bool minus = false;

while (true) {

// возвращаем все флаги в исходное состояние

ok = false;

dot = false;

minus = false;

error\_dot = false;

error\_minus = false;

// читаем строку

cout << promt;

char\_str = get\_string(&len);

capacity = 1;

clear\_char\_str = (char\*) malloc(sizeof(char));

// очистка строки от пробелов

if (check\_space){

len\_clear = 0;

for (i = 0; i < len; i++) {

if (char\_str[i] != ' ') {

// замента запятой на точку

if (char\_str[i] == ',')

clear\_char\_str[len\_clear++] = '.';

else

clear\_char\_str[len\_clear++] = char\_str[i];

capacity \*= 2;

clear\_char\_str = (char\*) realloc(clear\_char\_str, capacity \* sizeof(char));

}

}

} else {

len\_clear = len;

clear\_char\_str = char\_str;

}

// проверка на знак минуса

if (char\_str[0] == '-' && check\_minus) {

minus = true;

}

// проходимся по каждому символу строки

for (i = ((minus)? 1: 0); i < len\_clear; i++) {

switch (clear\_char\_str[i]) {

// проверка на точку

case('.'):

if (check\_dot)

if (dot) {

ok = false;

error\_dot = true;

} else {

ok = true;

dot = true;

}

else {

ok = false;

error\_dot = true;

}

break;

// проверка на знак минуса

case('-'):

ok = false;

error\_minus = true;

break;

// проверка на остальные символы

default:

ok = false;

// проходимся по каждому символу словаря

for (j = 0; j < SYMB\_LEN\_DOUBLE; j++) {

// стравниваем символ со словарём

if (clear\_char\_str[i] == symb[j]){

// если нашли символ в словаре,

// то останавливаем цикл со словарём

ok = true;

break;

}

}

break;

}

// если мы не нашли символ, пишем ошибку

if (!ok) {

cout << " [ Ошибка ввода ]: ";

// ошибки связанные с точками

if (error\_dot && !check\_dot) {

cout << "Число не должно содержать точки.";

} else if (error\_dot) {

cout << "Слишком много точек.";

}

// ошибки связанные со знаком минус

if (error\_minus && !check\_minus) {

cout << "Число не может быть отрицательным.";

} else if (error\_minus) {

cout << "Число не может содержать несколько знаков минус.";

}

// остальные ошибки

if (!error\_dot && !error\_minus) {

cout << "Число не может содержать в себе посторонних символов.";

}

cout << endl;

break;

}

}

// если небыло ошибок, то останавливаем бесконечный цикл

if (ok)

break;

}

// переводим и возвращаем значение

return atof(clear\_char\_str);

}

**simple\_char.h**

#include <iostream>

char \*get\_string(int \*len) {

\*len = 0;

int capacity = 1;

char \*s = (char\*) malloc(sizeof(char));

char c = getchar();

while (c != '\n') {

s[(\*len)++] = c;

if (\*len >= capacity) {

capacity \*= 2;

s = (char\*) realloc(s, capacity \* sizeof(char));

}

c = getchar();

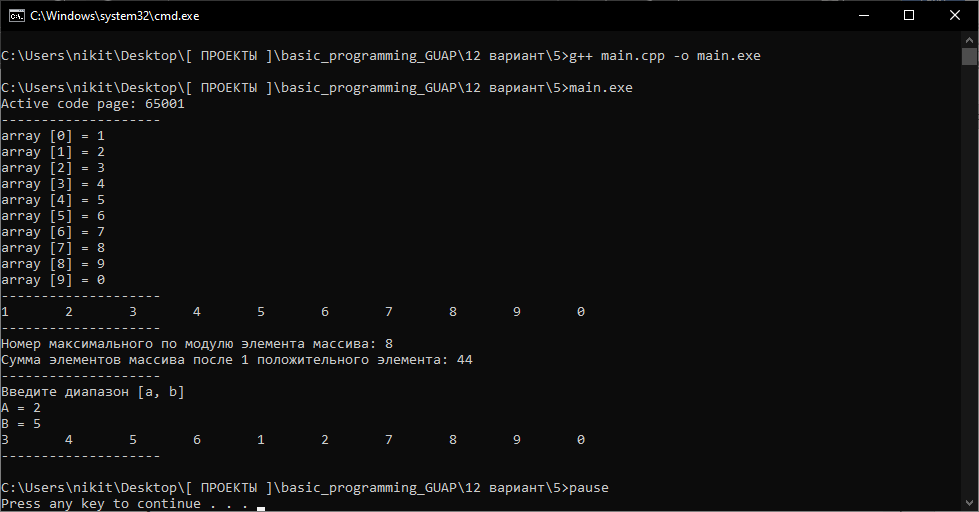
}

s[\*len] = '\0';

return s;

}

**10.Пример выполнения программы:**



Видно, что результаты расчётов совпадают с тестовыми данными.

**11.Анализ результатов и выводы:**

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

Из недостатков можно отметить то что многие части кода не разбиты на разные функции.

Из плюсов можно отметить то что оно работает.