ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯНАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ «МЭИ»

Лабораторная работа #5

ООП C++. ШАБЛОНЫ

Черкасов Александр
А-08-19
1 вариант

Содержание

Задание (5.1)

В одномерном массиве, состоящем из n элементов, вычислить сумму отрицательных элементов массива;

Задание (5.2)

В одномерном массиве, состоящем из n элементов, вычислить произведение элементов массива, расположенных между максимальным и минимальным элементами;

Задание (5.3)

В одномерном массиве, состоящем из n элементов упорядочить элементы массива по возрастанию (сортировка методом обмена).

1. Постановка задачи

Разработать объектно-ориентрованную программу в соответствии с заданиями 5.1, 5.2 и 5.3
Входные данные: одномерный массив из n элементов

5.1
Функции: вычисление суммы отрицательных элементов массива
Выходные данные: сумма отрицательных элементов

5.2
Функции: вычисление произведения элементов между минимум и максимумом
Выходные данные: произведение элементов между первым минимум и первым максимумом
Ограничения: Минимум и максимум не совпадают, между ними есть хотя бы один элемент, не пустой массив

5.3
Функции: Сортировка массива по возрастанию
Выходные данные: отсортированный массив
Ограничения: массив не пуст

Вид приложения - консольное приложение на языке C++.
Среда разработки – CLion

2. Разработка программы

2.1 Разработка структуры приложения

5.1 Создать шаблонную функцию sumOfNegative, вычисляющий и возвращающий сумму отрицательных элементов массива. Если таких не будет – вернуть 0.

5.2 Создать шаблонную функцию mulFromMinToMax, вычисляющий произведение элементов между первым минимум и первым максимумом. Необходима проверка на наличие элементов, на совпадение минимума и максимума, на наличие элементов между ними. Предусмотреть 2 случая:

индекс минимума < индекса максимума

индекс минимума > индекса максимума

5.3 Создать шаблонную функцию bubbleSort, сортирующую массив по возрастанию. Сортировку производить методом обмена(пузырьком). Необходима проверка на пустоту массива

2.2 Разработка схемы алгоритма

Алгоритм для задания 5.2 представлен на рис 2.1

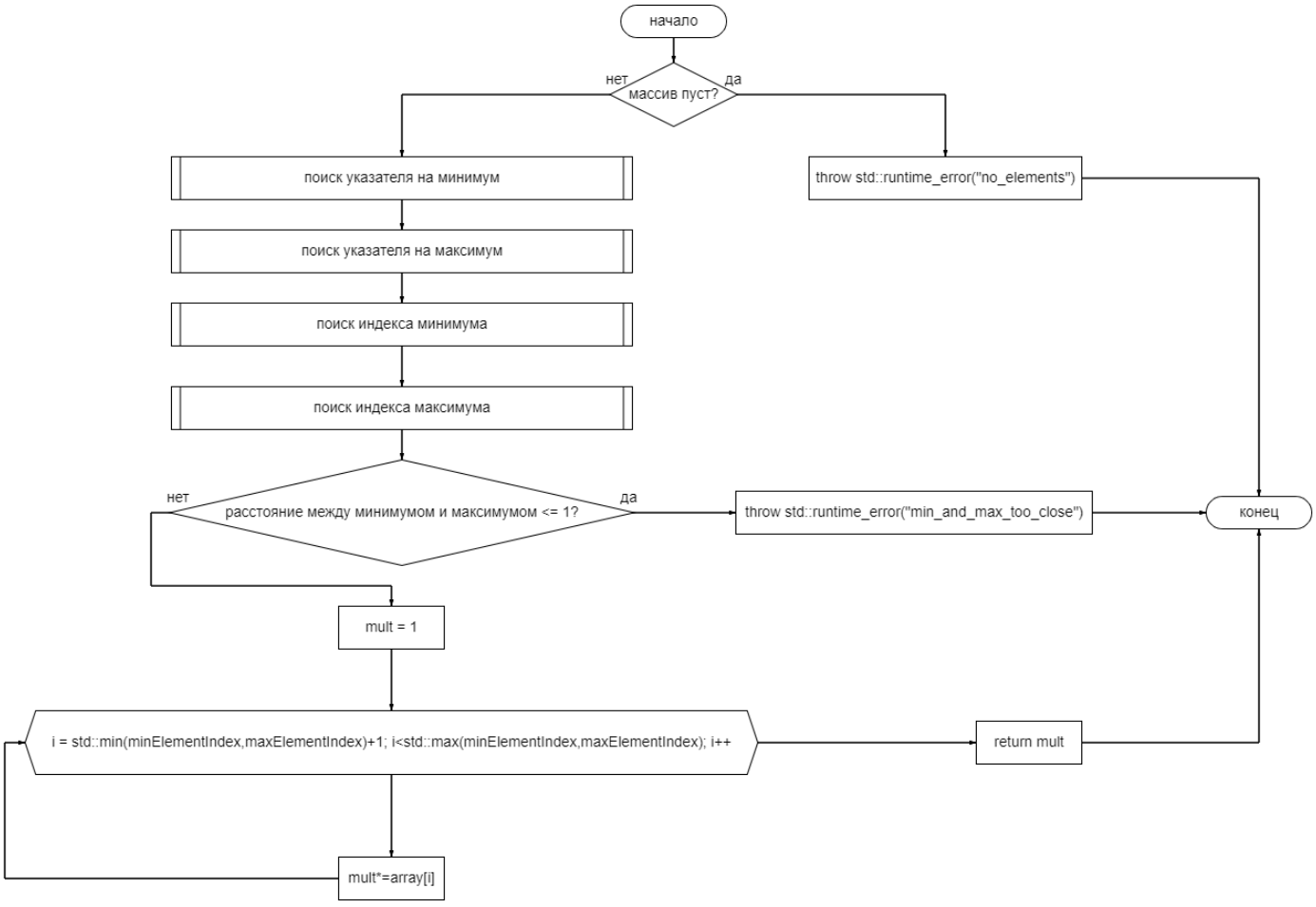

рис 2.1

2.3 Разработка интерфейса

Необходимо предоставить пользователю диалоговое меню, с помощью которого он сможет воспользоваться разработанными функциями. В меню будут отображаться тип используемого массива, его содержимое. Будет дана возможность переключаться между массивами типов int, float, double, вводить массив вручную или сгенерировать системой. Пример меню на рис. 2.2

```
Current Vector: int
{ }
sum - Get sum of negative elements of array
mult - Multiply elements between minimum and maximum
sort - sort array using shaking(modified bubble) sort
type - Write new array
generate - Get new auto generated array
switch - Switch between array types
exit - Close menu
```

рис 2.2

3. Реализация и тестирование программы

3.1 Описание разработанной программы

sumOfNegative принимает одномерный массив из n элементов и возвращает сумму отрицательных элементов. Если таких не будет – возвращает 0

mulFromMinToMax принимает одномерный массив из n элементов и возвращает произведение элементов между первым минимум и первым максимумом. Если минимум и максимум совпадают или являются соседними элементами, выбрасывается ошибка std::runtime_error("min_and_max_too_close"), если массив пуст std::runtime_error("no_elements"). Предусмотрены различные расположения минимума и максимума относительно друг друга: минимум раньше максимума и наоборот

bubbleSort принимает одномерный массив из n элементов по ссылке и сортирует его «улучшенным» методом пузырька – shaking (coctail) sort. При пустом массиве выбрасывается ошибка std::invalid_argument("empty_array")

3.2 Тестирование программы

Объект тестирования – разработанные функции

Цель тестирования – проверить их работоспособность

Средства испытаний – Среда CLion, компилятор MinGW, стандарты языка - 17

Порядок испытаний – функции тестируются для типов данных в порядке int, float, double

Методы испытаний – функциональное тестирование (тестирование по входу)

Тестирование проводилось на нескольких наборах данных – в нормальных, граничных и исключительных условиях. Результаты тестирования приведены в таблицах 3.1,3.2,3.3:

табл 3.1

№ теста	Смысл теста	Входные данные	Результат
1	Проверка работоспособности разработанных функций	1. type 5 4 2 -1 5 9 2. generate 3. sum 4. mult 5. sort	1. Current array { 4 2 -1 5 9 } 2. Current array { 10 1 -8 -8 1 7 9 9 2 } 3. Sum of negative elements in array:-16 4. Result of multiplication of elements between min and max:1 5. Current array { -8 -8 1 1 2 7 9 9 10 }

2	Реакция на отсутствие отрицательных элементов	1.type 4 4 2 5 9 2.sum	1. Current array { 4 2 5 9 } 2. Sum of negative elements in array:0
3	Реакция на расположение минимальный и максимальный элементов рядом	1. type 6 4 2 -1 -5 9 -8 2. mult	1. Current array { 4 2 -1 -5 9 -8 } 2. Result of multiplication of elements between min and max: min_and_max_too_close
4	Несколько максимумов и минимумов	1. type 5 -9 -9 5 5 5 2. mult	1. Current array { -9 -9 5 5 5 } 2. Result of multiplication of elements between min and max:-9
5	Массив заранее отсортирован	1. type 5 1 2 3 4 5 2. sort	1. Current array { 1 2 3 4 5 } 2. Current array { 1 2 3 4 5 }

табл 3.2

№ теста	**Смысл теста**	**Входные данные**	**Результат**
1	Проверка работоспособности разработанных функций	1. type 5 4.1 1.3 4.5 3.1 0.3 2. generate 3. sum 4. mult 5. sort	1. Current array { 4.1 1.3 4.5 3.1 0.3 } 2. Current array { -3.69243 -8.4959 -9.11344 -12.5752 -7.3162 14.615 -7.85043 8.1695 } 3. Sum of negative elements in array:-49.0436 4. Result of multiplication of elements between min and max:-7.3162 5. Current array { -12.5752 -9.11344 -8.4959 -7.85043 -7.3162 -3.69243 8.1695 14.615 }
2	Реакция на отсутствие отрицательных элементов	1. type 5 4.1 1.3 4.5 3.1 0.3 2. sum	1. Current array { 4.1 1.3 4.5 3.1 0.3 } 2. Sum of negative elements in array:0
3	Реакция на расположение минимальный и максимальный элементов рядом	1. type 5 1 8 -5.5 4 4 2. mult	1. Current array { 1 8 -5.5 4 4 } 2. Result of multiplication of elements between min and max: min_and_max_too_close
4	Несколько максимумов и минимумов	1. type 5 -5.5 -5.5 -5.5 4 4 2. mult	1. Current array { -5.5 -5.5 -5.5 4 4 } 2. Result of multiplication of elements between min and max:30.25
5	Массив заранее отсортирован	1. type 5 -1.6 -0.46 -0.1 5 16 2. sort	1. Current array { -1.6 -0.46 -0.1 5 16 } 2. Current array { -1.6 -0.46 -0.1 5 16 }

Табл 3.3

№ теста	**Смысл теста**	**Входные данные**	**Результат**
1	Проверка работоспособности разработанных функций	1. type 5 -5.5 1.3 8.1235 3.1 -5.37777 2. generate 3. sum 4. mult 5. sort	1. Current array { -5.5 1.3 8.1235 3.1 -5.37777 } 2. Current array { 7.12119 3.30332 -4.59059 6.59566 9.73006 -5.51988 -6.84561 5.68239 } 3. Sum of negative elements in array:-16.9561

			4. Result of multiplication of elements between min and max:-5.51988 5. Current array { -6.84561 -5.51988 -4.59059 3.30332 5.68239 6.59566 7.12119 9.73006 }
2	Реакция на отсутствие отрицательных элементов	1. type 5 1.345678 2.1234567 31.34567 5.4567 5.76223 2. sum	1. Current array { 1.34568 2.12346 31.3457 5.4567 5.76223 } 2. Sum of negative elements in array:0
3	Реакция на расположение минимальный и максимальный элементов рядом	1. type 5 5.46 6.12 1.545 41.4 6.23 2. mult	1. Current array { 5.46 6.12 1.545 41.4 6.23 } 2. Result of multiplication of elements between min and max:min_and_max_too_close
4	Несколько максимумов и минимумов	1. type 6 12.555 12.555 12.555 -6.1 -6.1 -6.1 2. mult	1. Current array { 12.555 12.555 12.555 -6.1 -6.1 -6.1 } 2. Result of multiplication of elements between min and max:157.628
5	Массив заранее отсортирован	1. type 5 -1.6123 -0.464444 -0.1645453 5.123 16.562221 2. sort	1. Current array { -1.6123 -0.464444 -0.164545 5.123 16.5622 } 2. Current array { -1.6123 -0.464444 -0.164545 5.123 16.5622 }

Вывод

Результаты тестов показали, что программа работает, как и задумано.

Научился создавать шаблонные функции.

Приложение. Код программы

main.cpp

```
#include <iostream>
#include "lab_ui.hpp"
#include <ctime>

int main() {
    lab_ui ui;
    ui.showUI();
    return 0;
}
```

lab_ui.hpp

```
#ifndef LAB5_LAB_UI_HPP
#define LAB5_LAB_UI_HPP

#include <ctime>
#include <map>
#include <iostream>
#include "task_functions.hpp"

//Вывод массива
template<class T>
std::ostream& operator<<(std::ostream& os,const std::vector<T>& v){
    os << "{ ";
    for (const auto& item :v) os << item << " ";
    os << "}";
    return os;
}

class lab_ui {
    std::vector<int>* vInt;
    std::vector<float>* vFloat;
    std::vector<double>* vDouble;

    std::map<std::string,short>* commands;
```

```
    int vType;

public:
    lab_ui();
    ~lab_ui() = default;

    void showUI();
    template<class T>
    static void generate_vector(std::vector<T>&);
private:
    short get_command();
    bool event_handler(short);

    template<class T>
    static void get_vector(std::vector<T>&);
    template<class T>
    static void show_vector(std::vector<T>&);
    template<class T>
    static void show_sum(std::vector<T>&);
    template<class T>
    static void show_mult(std::vector<T>&);

    static double get_num();
// «Удобный способ» вызова команд
#define execute(function) if (vType==1) function(*vFloat); else if (vType==2) function(*vDouble);
else function(*vInt)
};



//Генерация случайного массива
template<class T>
void lab_ui::generate_vector(std::vector<T>& v) {
    const double max_number = 15;

    size_t vector_size = rand()%20;
    for (size_t i=0;i<vector_size;i++){
        double value = (double)rand()/RAND_MAX*max_number * (rand()%2 ? 1 : -1);
        v.push_back(static_cast<T>(value));
    }
}

lab_ui::lab_ui(): vInt(nullptr), vDouble(nullptr), vFloat(nullptr), vType(0), commands(nullptr){}
//Основной цикл меню
void lab_ui::showUI() {
    srand(time(0));
    vInt = new std::vector<int>;
    commands = new std::map<std::string,short>;
    *commands =
{{"generate",1},{"type",2},{"sum",3},{"mult",4},{"sort",5},{"switch",6},{"exit",7}};
    while(event_handler(get_command()));

    delete vInt;
    delete vFloat;
    delete vDouble;
    delete commands;
}
//Обработчик команд
bool lab_ui::event_handler(short command) {
    switch (command) {
        default:
            std::cout << "Unknown command\n";
            break;
        case(1):
            execute(generate_vector);
            break;
        case(2):
            execute(get_vector);
            break;
        case(3):
            execute(show_sum);
            break;
        case(4):
            try {
                execute(show_mult);
            } catch (std::exception& error) {
                std::cout << error.what() << std::endl;
            }
            break;
        case(5):
            try {
                execute(bubbleSort);
            } catch (std::exception& error) {
                std::cout << error.what() << std::endl;
            }
            break;
        case(6): {
```

```
            std::cout << "Which type to use? int float double?"<<std::endl;
            std::string newType;
            std::cin>> newType;
            for (auto& item: newType) item=tolower(item);
            if (newType=="int"){
                delete vDouble;
                delete vFloat;
                vInt = new std::vector<int>;
                vType=0;
            }
            if (newType=="float"){
                delete vDouble;
                delete vInt;
                vFloat = new std::vector<float>;
                vType=1;
            }
            if (newType=="double"){
                delete vInt;
                delete vFloat;
                vDouble = new std::vector<double>;
                vType=2;
            }
            else std::cout << "Wrong command\n";
        }
            break;
        case(7): return false;
    }
    return true;
}
//Получение команд от пользователя
short lab_ui::get_command() {
    std::cout << "\nCurrent Vector: "<< (vType ? vType==1 ? "float" : "double" : "int") <<"\n";
    execute(show_vector);
    std::cout<<"\nsum - Get sum of negative elements of array"<< std::endl <<
                "mult - Multiply elements between minimum and maximum"<< std::endl <<
                "sort - sort array using shaking(modified bubble) sort"<< std::endl <<
                "type - Write new array"<< std::endl <<
                "generate - Get new auto generated array"<< std::endl <<
                "switch - Switch between array types"<< std::endl <<
                "exit - Close menu"<< std::endl;
    std::string command;
    std::cin>>command;
    for (auto& item: command) item=tolower(item);
    try{
        return commands->at(command);
    }
    catch (...) {
        return 0;
    }
}
//Ввод массива с клавиатуры
template<class T>
void lab_ui::get_vector(std::vector<T>& v) {
    std::cout << "Specify size" << std::endl;
    auto vector_size = static_cast<size_t>(get_num());
    std::cout << "Fill in elements" << std::endl;
    std::vector<T> new_vector;
    for (size_t i=0;i<vector_size;i++)
        new_vector.push_back(static_cast<T>(get_num()));
    v = new_vector;
}
// «Обложка» для вывода массива
template<class T>
void lab_ui::show_vector(std::vector<T> & v) {
    std::cout << v;
}
//Ввод числа из потока с обработкой входных данных
double lab_ui::get_num() {
    double num;
    std::string sNum;
    bool isCorrect = false;
    while(!isCorrect) {
        std::cin>>sNum;
        try {
            num= std::stod(sNum);
            isCorrect = true;
        }
        catch (...) {
            std::cout << "Wrong argument. Try again.\n";
        }
    }
    return num;
}
// «Обложка» для вывода суммы отриц. элементов
template<class T>
void lab_ui::show_sum(std::vector<T>& v) {
    std::cout << "Sum of negative elements in array:" << sumOfNegative(v) << std::endl;
```

```
}
// «Обложка» для вывода произведения между мин и макс
template<class T>
void lab_ui::show_mult(std::vector<T>& v) {
    std::cout << "Result of multiplication of elements between min and max:"  <<
mulFromMinToMax(v) << std::endl;
}


#endif //LAB5_LAB_UI_HPP
```

task_functions.hpp

```
#ifndef LAB5_TASK_FUNCTIONS_HPP
#define LAB5_TASK_FUNCTIONS_HPP
#include <vector>
#include <algorithm>
#include <cmath>


template<class T>
T sumOfNegative(const std::vector<T>&);

template<class T>
T mulFromMinToMax(const std::vector<T>&);

template<class T>
void bubbleSort(std::vector<T>&);

//Просто проход по всему массиву с поиском отриц элементов
template<class T>
T sumOfNegative(const std::vector<T>& array){
    T sum = 0;
    for (const T& item:array) if (item<0) sum+=item;
    return sum;
}


template<class T>
T mulFromMinToMax(const std::vector<T>& array){
    if (array.empty()) throw std::runtime_error("no_elements");

//Получение указателей
    auto minElementIterator = std::min_element(array.begin(),array.end());
    auto maxElementIterator = std::max_element(array.begin(),array.end());

//Через них поиск индексов
    size_t minElementIndex = std::distance(array.begin(),minElementIterator);
    size_t maxElementIndex = std::distance(array.begin(),maxElementIterator);

//Проверка на искл. ситуацию
    if(std::abs((int)minElementIndex-(int)maxElementIndex)<=1) throw
std::runtime_error("min_and_max_too_close");

    T mult = 1;

//От меньшего индекса до большего
    for (size_t i =
std::min(minElementIndex,maxElementIndex)+1;i<std::max(minElementIndex,maxElementIndex);i++)
        mult*=array[i];
    return mult;
}

template<class T>
void bubbleSort(std::vector<T>& array){
//Проверка на пусто массив
    if (!array.empty()) {
        size_t start = 0, end = array.size();
         //Если ничего не изменилось - массив пуст
        bool isChanged =true;

        while (isChanged && start != end) {
            isChanged = false;
         //Сама сортировка
            for (size_t i = start++; i < end-1; i++) {
                if (array[i]>array[i+1]) {
                    T temp = array[i];
                    array[i]=array[i+1];
                    array[i+1] = temp;
                    isChanged=true;
                }
            }

            for (size_t i = --end-1; i >= start && isChanged; i--) {
                if (array[i-1]>array[i]) {
                    T temp = array[i];
                    array[i]=array[i-1];
```

```
                    array[i-1] = temp;
                    isChanged= true;
                }
            }
        }
    }
    else{
        throw std::invalid_argument("empty_array");
    }
}

#endif //LAB5_TASK_FUNCTIONS_HPP
```