Министерство высшего образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«Пермский национальный исследовательский политехнический университет» (ПНИПУ)**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

ОТЧЁТ

по лабораторной работе №12

Тема: «Ассоциативные контейнеры библиотеки STL»

Выполнил

Студент группы РИС-22-2б

Арапов М.В.

Проверил доц. Кафедры ИТАС

Полякова О.А.

Пермь 2023

# Постановка задачи

# Задача 1.

# 1. Создать последовательный контейнер.

# 2. Заполнить его элементами пользовательского типа (тип указан в варианте). Для пользовательского типа перегрузить необходимые операции.

# 3. Заменить элементы в соответствии с заданием (использовать алгоритмы replace\_if(), replace\_copy(), replace\_copy\_if(), fill()).

# 4. Удалить элементы в соответствии с заданием (использовать алгоритмы remove(),remove\_if(), remove\_copy\_if(),remove\_copy())

# 5. Отсортировать контейнер по убыванию и по возрастанию ключевого поля (использовать алгоритм sort()).

# 6. Найти в контейнере заданный элемент (использовать алгоритмы find(), find\_if(), count(), count\_if()).

# 7. Выполнить задание варианта для полученного контейнера (использовать алгоритм for\_each()) .

# 8. Для выполнения всех заданий использовать стандартные алгоритмы библиотеки STL.

# Задача 2.

# 1. Создать адаптер контейнера.

# 2. Заполнить его элементами пользовательского типа (тип указан в варианте). Для пользовательского типа перегрузить необходимые операции.

# 3. Заменить элементы в соответствии с заданием (использовать алгоритмы replace\_if(), replace\_copy(), replace\_copy\_if(), fill()).

# 4. Удалить элементы в соответствии с заданием (использовать алгоритмы remove(),remove\_if(), remove\_copy\_if(),remove\_copy())

# 5. Отсортировать контейнер по убыванию и по возрастанию ключевого поля (использовать алгоритм sort()).

# 6. Найти в контейнере элемент с заданным ключевым полем (использовать алгоритмы find(), find\_if(), count(), count\_if()).

# 7. Выполнить задание варианта для полученного контейнера (использовать алгоритм for\_each()) .

# 8. Для выполнения всех заданий использовать стандартные алгоритмы библиотеки STL.

# Задача 3

# 1. Создать ассоциативный контейнер.

# 2. Заполнить его элементами пользовательского типа (тип указан в варианте). Для пользовательского типа перегрузить необходимые операции.

# 3. Заменить элементы в соответствии с заданием (использовать алгоритмы replace\_if(), replace\_copy(), replace\_copy\_if(), fill()).

# 4. Удалить элементы в соответствии с заданием (использовать алгоритмы remove(),remove\_if(), remove\_copy\_if(),remove\_copy())

# 5. Отсортировать контейнер по убыванию и по возрастанию ключевого поля (использовать алгоритм sort()).

# 6. Найти в контейнере элемент с заданным ключевым полем (использовать алгоритмы find(), find\_if(), count(), count\_if()).

# 7. Выполнить задание варианта для полученного контейнера (использовать алгоритм for\_each()) .

# 8. Для выполнения всех заданий использовать стандартные алгоритмы библиотеки STL.

Задача 1

1. Контейнер - multiset

2. Тип элементов - double

Задача 2 Тип элементов Pair (см. лабораторную работу №3).

Задача 3 Параметризированный класс – Список (см. лабораторную работу №7)

Задание 3

Найти среднее арифметическое и добавить его в конец контейнера

Задание 4

Найти элементы ключами из заданного диапазона и удалить их из контейнера

Задание 5

К каждому элементу добавить сумму минимального и максимального элементов контейнера.

Контрольные вопросы

1. Что представляет собой ассоциативный контейнер?

Ассоциативный контейнер - это контейнер, в котором элементы хранятся в отсортированном порядке и доступ к ним осуществляется с помощью ключей.

2. Перечислить ассоциативные контейнеры библиотеки STL.

Ассоциативные контейнеры библиотеки STL: map, set, multimap, multiset.

3. Каким образом можно получить доступ к элементам ассоциативного контейнера?

Доступ к элементам ассоциативного контейнера осуществляется с помощью ключей.

4. Привести примеры методов, используемых в ассоциативных контейнерах.

Примеры методов, используемых в ассоциативных контейнерах: insert, erase, clear, find, begin, end, size.

5. Каким образом можно создать контейнер map? Привести примеры.

Контейнер map можно создать с помощью конструктора без параметров: map<Key, T> m;

6. Каким образом упорядочены элементы в контейнере map по умолчанию? Как изменить порядок на обратный?

Элементы в контейнере map упорядочены по ключам. Для изменения порядка на обратный можно создать контейнер с использованием структуры std::greater в качестве компаратора: map<Key, T, std::greater<Key>> m;

7. Какие операции определены для контейнера map?

Операции, определенные для контейнера map: size, empty, insert, erase, clear, find, count, begin, end, operator[].

8. Написать функцию для добавления элементов в контейнер map с помощью функции make\_pair().

Функция для добавления элементов в контейнер map с помощью функции make\_pair():

void insert\_to\_map(std::map<int, std::string>& m, int key, std::string value) {

m.insert(std::make\_pair(key, value));

}

9. Написать функцию для добавления элементов в контейнер map с помощью функции операции прямого доступа [].

Функция для добавления элементов в контейнер map с помощью функции операции прямого доступа []:

void insert\_to\_map(std::map<int, std::string>& m, int key, std::string value) {

m[key] = value;

}

10. Написать функцию для печати контейнера map с помощью итератора.

Функция для печати контейнера map с помощью итератора:

void print\_map(std::map<int, std::string>& m) {

for (auto it = m.begin(); it != m.end(); ++it) {

std::cout << it->first << " => " << it->second << ", ";

}

}

11. Написать функцию для печати контейнера map с помощью функции операции прямого доступа [].

Функция для печати контейнера map с помощью функции операции прямого доступа []:

void print\_map(std::map<int, std::string>& m) {

for (auto& [key, value] : m) {

std::cout << key << " => " << value << ", ";

}

}

12. Чем отличаются контейнеры map и multimap?

Контейнер map хранит только уникальные ключи и соответствующие им значения, в то время как контейнер multimap может хранить несколько значений для одного и того же ключа.

13. Что представляет собой контейнер set?

Контейнер set - это контейнер, содержащий уникальные элементы, отсортированные в порядке возрастания.

14. Чем отличаются контейнеры map и set?

Контейнер map содержит ключи и соответствующие им значения, в то время как контейнер set содержит только уникальные элементы.

15. Каким образом можно создать контейнер set? Привести примеры.

Контейнер set можно создать с помощью конструктора без параметров: set<T> s;

16. Каким образом упорядочены элементы в контейнере set по умолчанию? Как изменить порядок на обратный?

Элементы в контейнере set упорядочены по возрастанию. Для изменения порядка на обратный можно создать контейнер с использованием структуры std::greater в качестве компаратора: set<T, std::greater<T>> s;

17. Какие операции определены для контейнера set?

Операции, определенные для контейнера set: size, empty, insert, erase, clear, find, count, begin, end.

18. Написать функцию для добавления элементов в контейнер set.

Функция для добавления элементов в контейнер set:

void insert\_to\_set(std::set<int>& s, int value) {

s.insert(value);

}

19. Написать функцию для печати контейнера set.

void print\_set(std::set<int>& s) {

for (auto it = s.begin(); it != s.end(); ++it) {

std::cout << \*it << ", ";

}

}

20. Чем отличаются контейнеры set и multiset?

Контейнер set хранит только уникальные элементы, в то время как контейнер multiset может хранить несколько значений для одного и того же элемента.

UML – диаграмма



Код программы

**Pair.h**

#pragma once

#include <iostream>

using namespace std;

class Pair

{

public:

int a;

double b;

friend istream& operator>>(istream& in, Pair& p);

friend ostream& operator<<(ostream& out, const Pair& p);

Pair(int a, double b);

Pair();

Pair(const Pair& p);

~Pair();

Pair& operator=(const Pair& p);

Pair operator-(const Pair& p);

Pair& operator+(int a);

Pair& operator+(double b);

Pair& operator++();

Pair operator++(int);

Pair operator+(const Pair& p);

bool operator!=(const Pair& p) { return !((this->a == p.a) && (this->b == p.b)); }

bool operator==(const Pair& p) { return ((this->a == p.a) && (this->b == p.b)); }

bool operator>(const Pair& p) { return (this->a > p.a) || (this->b > p.b); }

bool operator>=(const Pair& p) { return (this->a >= p.a) || (this->b >= p.b); }

bool operator<=(const Pair& p) { return (this->a <= p.a) || (this->b <= p.b); }

bool operator<(const Pair& p) { return (this->a < p.a) || (this->b < p.b); }

Pair operator \* (Pair& pair)

{

Pair new\_pair;

new\_pair.a = this->a \* pair.a;

new\_pair.b = this->b \* pair.b;

return new\_pair;

}

};

**main.cpp**

#include <iostream>

#include <set>

#include "Pair.h"

using namespace std;

bool operator<(const Pair& pr1, const Pair& pr2)

{

return ((pr1.a < pr2.a) && (pr1.b < pr2.b));

}

void ex\_1()

{

setlocale(0, "rus");

multiset <double> mst;

multiset <double> tmp;

mst.insert(1.5);

mst.insert(-1.5);

mst.insert(2.3);

mst.insert(2.3);

double sr = 0;

cout << "Изначальное множество" << endl;

for (auto it = mst.begin(); it != mst.end(); it++)

{

sr += \*it;

cout << \*it << endl;

}

//1

cout << "Среднее, добавленное в конец: " << sr / mst.size() << endl;

mst.insert(sr / mst.size());

//2

double max = \*mst.begin();

double min = \*mst.begin();

tmp = mst;

mst.clear();

int c = 0;

int i, j;

double res;

cout << "Задайте диапозон: "; cin >> i >> j; cout << "Задайте ключ: "; cin >> res;

for (auto it = tmp.begin(); it != tmp.end(); it++, c++)

{

if (!(\*it == res && (c > i && c < j)))

{

if (max < \*it) max = \*it;

if (min > \*it) min = \*it;

mst.insert(\*it);

cout << \*it << endl;

}

}

cout << "Добавление суммы мин и макс: " << endl;

tmp = mst;

mst.clear();

for (auto it = tmp.begin(); it != tmp.end(); it++)

{

mst.insert(max + min + \*it);

cout << \*it + max + min << endl;

}

}

void ex\_2()

{

setlocale(0, "rus");

multiset <Pair> mst;

multiset <Pair> tmp;

mst.insert(Pair(1, 1.5));

mst.insert(Pair(1, -1.5));

mst.insert(Pair(2.3, 2.3));

mst.insert(Pair(2.3, 2.3));

Pair sr(0, 0);

cout << "Изначальное множество" << endl;

for (auto it = mst.begin(); it != mst.end(); it++)

{

sr = sr + \*it;

cout << \*it << endl;

}

//1

sr.a /= mst.size();

sr.b /= mst.size();

cout << "Среднее, добавленное в конец: " << sr << endl;

mst.insert(sr);

//2

Pair max = \*mst.begin();

Pair min = \*mst.begin();

tmp = mst;

mst.clear();

int c = 0;

int i, j;

Pair res;

cout << "Задайте диапозон: "; cin >> i >> j; cout << "Задайте ключ: "; cin >> res;

for (auto it = tmp.begin(); it != tmp.end(); it++, c++)

{

if ((c < i) || (c > j) || (res.a != (\*it).a || res.b != (\*it).b))

{

if (max < \*it) max = \*it;

if (min > \*it) min = \*it;

mst.insert(\*it);

cout << \*it << endl;

}

}

cout << "Добавление суммы мин и макс: " << endl;

tmp = mst;

mst.clear();

for (auto it = tmp.begin(); it != tmp.end(); it++)

{

mst.insert(max + min + \*it);

cout << max + min + \*it << endl;

}

}

void ex\_3()

{

setlocale(0, "rus");

multiset <Pair> mst;

multiset <Pair> tmp;

mst.insert(Pair(1, 1.5));

mst.insert(Pair(1, -1.5));

mst.insert(Pair(2.3, 2.3));

mst.insert(Pair(2.3, 2.3));

Pair sr(0, 0);

cout << "Изначальное множество" << endl;

for (auto it = mst.begin(); it != mst.end(); it++)

{

sr = sr + \*it;

cout << \*it << endl;

}

//1

sr.a /= mst.size();

sr.b /= mst.size();

cout << "Среднее, добавленное в конец: " << sr << endl;

mst.insert(sr);

//2

Pair max = \*mst.begin();

Pair min = \*mst.begin();

tmp = mst;

mst.clear();

int c = 0;

int i, j;

Pair res;

cout << "Задайте диапозон: "; cin >> i >> j; cout << "Задайте ключ: "; cin >> res;

for (auto it = tmp.begin(); it != tmp.end(); it++, c++)

{

if ((c < i) || (c > j) || (res.a != (\*it).a || res.b != (\*it).b))

{

if (max < \*it) max = \*it;

if (min > \*it) min = \*it;

mst.insert(\*it);

cout << \*it << endl;

}

}

cout << "Добавление суммы мин и макс: " << endl;

tmp = mst;

mst.clear();

for (auto it = tmp.begin(); it != tmp.end(); it++)

{

mst.insert(max + min + \*it);

cout << max + min + \*it << endl;

}

}

int main()

{

ex\_1();

ex\_2();

ex\_3();

}