Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«**Пермский национальный исследовательский политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЕТ**

Дисциплина: «Информатика»

Тема Ассоциативные контейнеры библиотеки STL

Семестр 2

Выполнил работу

Студент группы РИС-22-1Б

Тарасов C.В.

Проверил

Доцент кафедры ИТАС

Полякова О.А.

Г. Пермь-2023

**Постановка задачи (общая и конкретного варианта).**

### Задача 1.

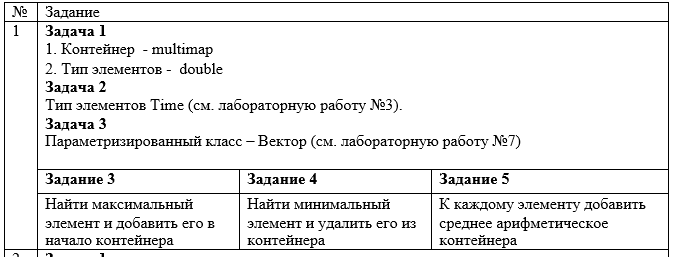
* + 1. Создать ассоциативный контейнер.
    2. Заполнить его элементами стандартного типа (тип указан в варианте).
    3. Добавить элементы в соответствии с заданием
    4. Удалить элементы в соответствии с заданием.
    5. Выполнить задание варианта для полученного контейнера.
    6. Выполнение всех заданий оформить в виде глобальных функций.

### Задача 2.

1. Создать ассоциативный контейнер.
2. Заполнить его элементами пользовательского типа (тип указан в варианте). Для пользовательского типа перегрузить необходимые операции.
3. Добавить элементы в соответствии с заданием
4. Удалить элементы в соответствии с заданием.
5. Выполнить задание варианта для полученного контейнера.
6. Выполнение всех заданий оформить в виде глобальных функций.

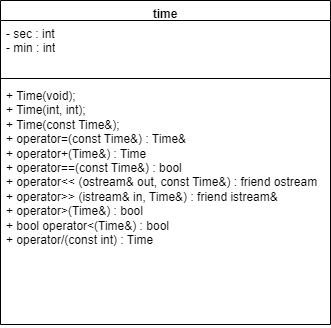
### Задача 3

1. Создать параметризированный класс, используя в качестве контейнера ассоциативный контейнер.
2. Заполнить его элементами.
3. Добавить элементы в соответствии с заданием
4. Удалить элементы в соответствии с заданием.
5. Выполнить задание варианта для полученного контейнера.
6. Выполнение всех заданий оформить в виде методов параметризированного класса.

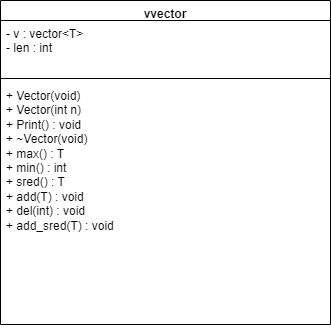


**UML**

**2**

****

**3**

****

**Функции для решения задачи 1.**

Tmap createAndFillMultimap(int n)

{

Tmap container;

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

{

double a;

cout << "?";

cin >> a;

//создаем пару и добавляем ее в словарь

container.insert(make\_pair(i, a));

}

return container;

}

double srednee(Tmap v)

{

double s = 0.0;

int n = v.size(); // количество элементов в мультимапе

// перебор мультимапы с использованием итератора

for (it it = v.begin(); it != v.end(); ++it)

{

s += it->second;

}

return s / n;

}

int Max(Tmap v)

{

it i = v.begin();

int nom = 0; // номер максимального элемента

int k = 0; // счетчик элементов

double m = i->second; // значение первого элемента

while (i != v.end())

{

if (m < i->second)

{

m = i->second;

nom = k;

}

++i; // итератор

++k; // счетчик элементов

}

return nom; // номер максимального элемента

}

int Min(Tmap v)

{

it i = v.begin();

int nom = 0; // номер максимального элемента

int k = 0; // счетчик элементов

double m = i->second; // значение первого элемента

while (i != v.end())

{

if (m > i->second)

{

m = i->second;

nom = k;

}

++i; // итератор

++k; // счетчик элементов

}

return nom; // номер минимального элемента

}

void del(Tmap& container)

{

int min = Min(container); // Находим номер минимального элемента

// Ищем элемент с соответствующим номером минимального элемента

auto it = container.begin();

std::advance(it, min);

container.erase(it); // Удаляем элемент из контейнера

}

void add(Tmap& container)

{

int max = Max(container); // Находим максимальный элемент

// Находим итератор для максимального элемента

it maxIterator = container.begin();

std::advance(maxIterator, max);

// Копируем максимальный элемент

auto maxElement = \*maxIterator;

// Вставляем максимальный элемент в начало контейнера

container.insert(container.begin(), maxElement);

}

void add\_sred(Tmap& container)

{

double sred = srednee(container);

for (auto& elem : container)

{

elem.second += sred;

}

}

void printMultimap(const Tmap & container)

{

for (const auto& element : container)

{

cout << "Номер: " << element.first << ", Значение: " << element.second << endl;

}

}

**Основная программа для решения задачи 1**

int main()

{

setlocale(LC\_ALL, "Russian");

int n;

cout << " введите кол-во элементов ";

cin >> n;

Tmap container = createAndFillMultimap(n);

printMultimap(container);

cout << endl;

add(container);

printMultimap(container);

cout << endl;

del(container);

printMultimap(container);

cout << endl;

add\_sred(container);

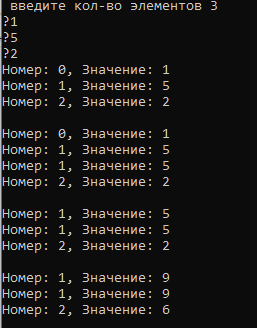
printMultimap(container);

cout << endl;

return 0;

}

**Объяснение результатов работы программы.**



**Описание пользовательского класса для решения задачи 2.**

class Time

{

private:

int min;

int sec;

public:

Time(void);

Time(int, int);

Time(const Time&);

Time& operator=(const Time&);

//перегруженные операции ввода-вывода

Time operator+(Time&);

bool operator==(const Time&);

friend ostream& operator<< (ostream& out, const Time&);

friend istream& operator>> (istream& in, Time&);

bool operator>(Time&);

bool operator<(Time&);

Time operator/(const int);

};

**Определение перегруженных операций для пользовательского класса.**

Time::Time(void)

{

min = sec = 0;

}

Time::Time(int M, int S)

{

if (S >= 60)

{

min = M + (S / 60);

sec = S % 60;

}

else

{

min = M;sec = S;

}

}

Time::Time(const Time& t)

{

min = t.min; sec = t.sec;

}

Time& Time::operator =(const Time& t)

{

min = t.min; sec = t.sec; return\*this;

}

ostream& operator<<(ostream& out, const Time& t)

{

out << t.min << " : " << t.sec; return out;

}

istream& operator>>(istream& in, Time& t)

{

cout << "\nmin? ";

in >> t.min;

cout << "\nsec? ";

in >> t.sec;

while (t.sec >= 60) {

cout << "Ошибка: количество секунд не может быть 60 или больше. Введите значение еще раз: " << endl;

in >> t.sec;

}

return in;

}

bool Time::operator==(const Time& t)

{

int temp = min \* 60 + sec;

int temp2 = t.min \* 60 + t.sec;

return temp == temp2;

}

Time Time::operator+(Time& t)

{

Time temp;

temp.min = ((min + t.min) \* 60 + (sec + t.sec)) / 60;

temp.sec = ((min + t.min) \* 60 + (sec + t.sec)) % 60;

return temp;

}

bool Time::operator>(Time& t)

{

int temp = min \* 60 + sec;

int temp2 = t.min \* 60 + t.sec;

return temp > temp2;

}

bool Time::operator<(Time& t)

{

int temp = min \* 60 + sec;

int temp2 = t.min \* 60 + t.sec;

return temp < temp2;

}

Time Time::operator/(const int a)

{

int temp = min \* 60 + sec;

int res = temp / a;

Time time;

time.min = res / 60;

time.sec = res % 60;

return time;

}

**Функции для решения задачи 2 .**

Tmap createAndFillMultimap(int n)

{

Tmap container;

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

{

Time a;

cout << "?";

cin >> a;

//создаем пару и добавляем ее в словарь

container.insert(make\_pair(i, a));

}

return container;

}

Time srednee(Tmap v)

{

Time s;

int n = v.size(); // количество элементов в мультимапе

// перебор мультимапы с использованием итератора

for (it it = v.begin(); it != v.end(); ++it)

{

s = s + it->second;

}

return s / n;

}

int Max(Tmap v)

{

it i = v.begin();

int nom = 0; // номер максимального элемента

int k = 0; // счетчик элементов

Time m = i->second; // значение первого элемента

while (i != v.end())

{

if (m < i->second)

{

m = i->second;

nom = k;

}

++i; // итератор

++k; // счетчик элементов

}

return nom; // номер максимального элемента

}

int Min(Tmap v)

{

it i = v.begin();

int nom = 0; // номер максимального элемента

int k = 0; // счетчик элементов

Time m = i->second; // значение первого элемента

while (i != v.end())

{

if (m > i->second)

{

m = i->second;

nom = k;

}

++i; // итератор

++k; // счетчик элементов

}

return nom; // номер минимального элемента

}

void del(Tmap& container)

{

int min = Min(container); // Находим номер минимального элемента

// Ищем элемент с соответствующим номером минимального элемента

auto it = container.begin();

std::advance(it, min);

container.erase(it); // Удаляем элемент из контейнера

}

void add(Tmap& container)

{

int max = Max(container); // Находим максимальный элемент

// Находим итератор для максимального элемента

it maxIterator = container.begin();

std::advance(maxIterator, max);

// Копируем максимальный элемент

auto maxElement = \*maxIterator;

// Вставляем максимальный элемент в начало контейнера

container.insert(container.begin(), maxElement);

}

void add\_sred(Tmap& container)

{

Time sred = srednee(container);

for (auto& elem : container)

{

elem.second = elem.second + sred;

}

}

void printMultimap(const Tmap& container)

{

for (const auto& element : container)

{

cout << "Номер: " << element.first << ", Значение: " << element.second << endl;

}

}

**Основная программа для решения задачи 2.**

int main()

{

setlocale(LC\_ALL, "Russian");

int n;

cout << " введите кол-во элементов ";

cin >> n;

Tmap container = createAndFillMultimap(n);

printMultimap(container);

cout << endl;

add(container);

printMultimap(container);

cout << endl;

del(container);

printMultimap(container);

cout << endl;

add\_sred(container);

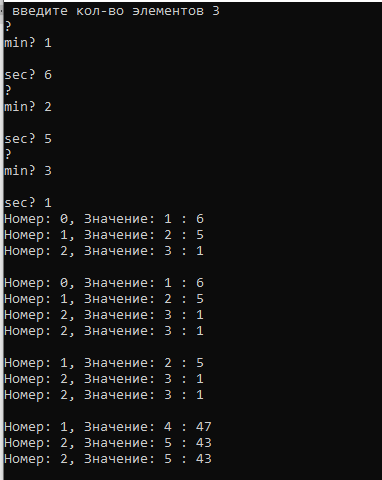
printMultimap(container);

cout << endl;

return 0;

}

**Объяснение результатов работы программы.**



**Описание параметризированного класса для решения задачи 3.**

template<class T> class Vector

{

vector <T> v;//последовательный контейнер для хранения элементов вектора

int len;

public:

Vector(void);//конструктор без параметра

Vector(int n);//конструктор с параметром

void Print();//печать

~Vector(void);//деструктор

T max();

int min();

T sred();

void add(T);

void del(int);

void add\_sred(T);

};

**Определение методов и операций для решения задачи 3.**

template <class T> Vector<T>::Vector()

{

len = 0;

}

//деструктор

template <class T>

Vector<T>::~Vector(void)

{

}

//конструктор с параметром

template <class T> Vector<T>::Vector(int n)

{

T a;

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

{

cin >> a; v.push\_back(a);

}

len = v.size();

}

//печать

template <class T> void Vector<T>::Print()

{

for (int i = 0;i < v.size();i++) cout << v[i] << " ";

cout << endl;

}

template <class T>

T Vector<T>::max()

{

T temp = v[0];

for (int i = 1; i < v.size(); i++)

{

if (v[i] > temp) temp = v[i];

}

return temp;

}

template <class T>

int Vector<T>::min()

{

T temp = v[0];

int n = 0;

for (int i = 1; i < v.size(); i++)

{

if (v[i] < temp)

{

temp = v[i]; n = i;

}

}

return n;

}

template <class T>

T Vector<T>::sred()

{

T temp = v[0];

for (int i = 1; i < v.size(); i++)

{

temp = temp + v[i];

}

int n = v.size();

return temp / n;

}

template <class T>

void Vector<T>::add(T a)

{

v.insert(v.begin(), a);

}

template <class T>

void Vector<T>::del(int a)

{

v.erase(v.begin() + a);

}

template <class T>

void Vector<T>::add\_sred(T a)

{

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

{

v[i] = v[i] + a;

}

}

**Основная программа для решения задачи 3**

int main()

{

Vector<Time> container(3);

setlocale(LC\_ALL, "Russian");

int n;

container.Print();

cout << endl;

Time a = container.max();

container.add(a);

container.Print();

cout << endl;

int b = container.min();

container.del(b);

container.Print();

cout << endl;

Time c = container.sred();

container.add\_sred(c);

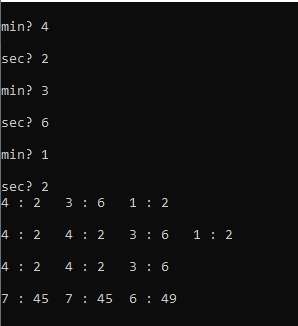
container.Print();

cout << endl;

return 0;

}

**Объяснение результатов работы программы.**



**Ответы на контрольные вопросы**

1. Что представляет собой ассоциативный контейнер?

Ассоциативный массив содержит пары значений. Зная одно значение,

называемое ключом (key), мы можем получить доступ к другому, называемому отображаемым значением (mapped value).

Ассоциативный массив можно представить как массив, для которого индекс не обязательно должен иметь целочисленный тип.

2. Перечислить ассоциативные контейнеры библиотеки STL.

map - ассоциативный массив, по ключу в контейнере хранится одно значение

multimap - ассоциативный массив с повторяющимися ключами

set - массив уникальных ключей без значений

multiset - массив с повторяющимися ключами без значений.

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

V& operator[](const K&) возвращает ссылку на элемент V, соответствующий значению K.

4. Привести примеры методов, используемых в ассоциативных контейнерах.

bool empty() const, size\_type size() const, size\_type max\_size(), insert(), erase(), clear(), swap(), key\_comp(), value\_comp(), find(), count(), lower\_bound(), upper\_bound().

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

#include <iostream>

#include <map>

using namespace std;

int main() {

map<int, float> m;//словарь\

int n;//количество элементов

cout << "n: ";

cin >> n;

float a;

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

cout <<" ?";

cin >> a; //создаем пару и добавляем ее в словарь

m.insert(make\_pair(i, a)); }

return 0;

}

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

Ассоциативный контейнер map требует, чтобы для типов ключа существовала операция “<”, то есть, элементы хранятся в порядке возрастания. Он хранит свои элементы отсортированными по ключу так, что перебор происходит по порядку.

Спецификация шаблона для класса map:

template <class Key, class T, class Comp = less <Key>, class Allocator = allocator <pair> >

class Comp = less <Key> - параметр, определяющий критерий

упорядочения, по умолчанию less (по возрастанию ключа).

Чтобы изменить критерий упорядочивания, нужно изменить параметр Comp.

7. Какие операции определены для контейнера map?

Определена операция присваивания: map& operator=(const map&);

Определены следующие операции: ==, <, <=, !=, >, >= и операция индексации ([]).

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

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

cout <<" ?";

cin >> a;

m.insert(make\_pair(i, a));

}

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

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

cout <<" ?";

cin >> a;

m[i] = a;

}

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

for (map<int, float>::iterator i = m.begin(); i != m.end(); ++i) {

cout << (\*i).first << " " << (\*i).second << endl;

}

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

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

cout << m[i] << endl;

}

12. Чем отличаются контейнеры map и multimap?

Словари с дубликатами (multimap) допускают хранение элементов с одинаковыми ключами. Поэтому для них не определена операция доступа по индексу. Элементы с одинаковыми ключами хранятся в словаре в порядке их занесения. При удалении по ключу функция erase возвращает количество удаленных элементов. В остальном они аналогичны обычным словарям.

13. Что представляет собой контейнер set?

set — это контейнер, который автоматически сортирует добавляемые элементы в порядке возрастания. Но при добавлении одинаковых

значений, set будет хранить только один его экземпляр. По другому его еще называют множеством.

14. Чем отличаются контейнеры map и set?

Тип set позволяет хранить уникальные объекты различных типов, эффективно добавлять, удалять объекты и выполнять поиск. Тип map позволяет хранить пары ключ-значение, причем ключи должны быть уникальными.

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

set<int> set1; // создается пустое множество

int а[5] = { 1. 2. 3. 4, 5};

set<int> set2(a, а + 5);// инициализация копированием

set<int> set3(set2); // инициализация другим множеством

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

Множество, как и словарь, требует, чтобы для типа T существовала операция “меньше” (<). Оно хранит свои элементы отсортированными, так что перебор происходит по порядку.

В множестве хранятся объекты, упорядоченные по некоторому ключу, являющемуся атрибутом самого объекта. Например, множество может хранить объекты класса Person, упорядоченные в алфавитном порядке по значению ключевого поля name. Если в множестве хранятся значения одного из встроенных типов, например int, то ключом является сам элемент.

set<int, greater<int>> set - изменение критерия упорядочения путем спецификации параметра компаратора

17. Какие операции определены для контейнера set?

Для вставки элементов в множество можно использовать метод insert(), для

удаления — метод erase(). Также к множествам применимы общие для всех контейнеров методы. Во всех ассоциативных контейнерах есть метод count(), возвращающий количество объектов с заданным ключом.

18. Написать функцию для добавления элементов в контейнер set.

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

cout <<" ? ";

cin >> a;

s.insert(a);

}

19. Написать функцию для печати контейнера set.

for(auto it = s.begin(); it != s.end(); ++i) cout<<\*i<<” ”;

20. Чем отличаются контейнеры set и multiset?

В множествах с дубликатами ключи могут повторяться. Элементы с одинаковыми ключами хранятся в множестве в порядке их занесения. Функция find() возвращает значение первого найденного элемента или end(), если ни одного элемента с заданным ключом не найдено.