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: «Основы алгоритмизации и программирования»

Тема: «Ассоциативные контейнеры библиотеки STL»

Семестр 2

Выполнил работу:

Студент группы ИВТ-22-2б

Цыбуцынин Фёдор Александрович

Проверил:

Доцент кафедры ИТАС

Полякова Ольга Андреевна

Г. Пермь-2023

**Постановка задачи**

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.

**Описание класса**

Задача 1

1. Контейнер - set

2. Тип элементов - float

Задача 2

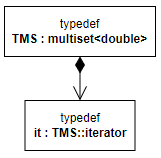
Тип элементов Time (см. лабораторную работу №3).

Задача 3

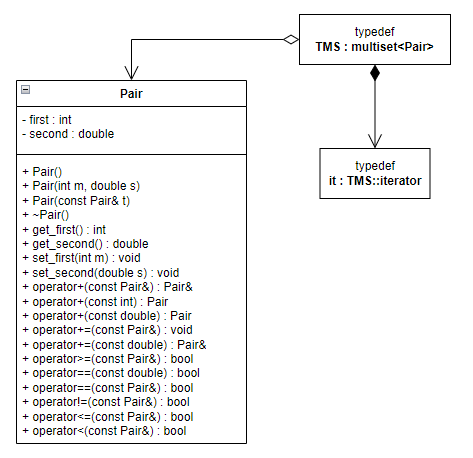
Параметризованный класс – Вектор (см. лабораторную работу №7)

**UML - Диаграмма**

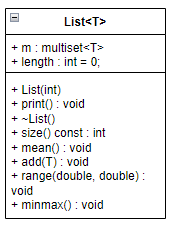
**Задача 1**



**Задача 2**



**Задача 3**

****

**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 отображённым значением (mapped value).

1. Перечислить ассоциативные контейнеры библиотеки STL.
2. map - ассоциативный массив, по ключу в контейнере хранится одно значение
3. multimap - ассоциативный массив с повторяющимися ключами
4. set - массив уникальных ключей без значений
5. multiset - массив с повторяющимися ключами без значений
6. Каким образом можно получить доступ к элементам ассоциативного контейнера?

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

1. Привести примеры методов, используемых в ассоциативных контейнерах.

|  |  |
| --- | --- |
| **Операция или метод** | **Пояснение** |
| bool empty() const size\_type size() const  size\_type max\_size() | Методы определения размеров |
| insert() | Добавляет один элемент или диапазон элементов |
| erase() | Удаляет один элемент или диапазон элементов |
| clear() | Удаляет все элементы |
| swap() | Обмен данными с контейнером того же типа |
| key\_comp() value\_comp() | Возвращают объекты-функторы для сравнения  ключей и значений |
| find() count() lower\_bound()  upper\_bound() | Методы поиска (метод count() вычисляет для мульти контейнеров количество элементов с заданным ключом) |

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

#include <iostream>

#include <map>

using namespace std;

int main()

{

map<int, float> m;

int n;

cout << "n: ";

cin >> n;

float a;

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

{

cout << " ?";

cin >> a;

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

}

return 0;

}

1. Каким образом упорядочены элементы в контейнере map по умолчанию?

Как изменить порядок на обратный?

Ассоциативный контейнер map  это последовательность пар (ключ, значение), которая обеспечивает быстрое получение значения по ключу. Контейнер map предоставляет двунаправленные итераторы.

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

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 Сomp.

1. Какие операции определены для контейнера map?

Определена операция присваивания:

map& operator=(const map&);

Определены следующие операции: ==, <, <=, !=, >, >=.

1. 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));

}

1. 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;

}

1. 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;

}

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

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

{

cout << m[i] << endl;

}

1. 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.

1. Что представляет собой контейнер set?

Множества set можно рассматривать как ассоциативные массивы, в которых значения не играют роли, так что мы отслеживаем только ключи. Множество, как и словарь, требует, чтобы для типа T существовала операция “меньше” (<). Оно хранит свои элементы отсортированными, так что перебор происходит по порядку.

1. 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.

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

#include <iostream> #include <set>

using namespace std;

typedef set<int, less<int>>tset;

tset ::iterator i;

void main()

{

int a[4] = { 1,3,5,7 };

tset s(a, a + 4);

s.insert(10);

s.insert(6); for (i = s.begin(); i != s.end(); i++)

cout << \*i << " "; cout << endl;

}

1. 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

1. 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.

1. Написать функцию для добавления элементов в контейнер set.

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

{

cout << " ? ";

cin >> a;

s.insert(a);

}

1. Написать функцию для печати контейнера set.

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

1. 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.