Министерство цифрового развития, связи и массовых коммуникаций

Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования «Сибирский государственный университет

телекоммуникаций и информатики» (СибГУТИ)

Кафедра прикладной математики и кибернетики

Лабораторная работа

«АДТ Множество»

Выполнил:

Студент группы ИП-017

Костин А.В.

Работу проверил:

ассистент кафедры ПМиК

Агалаков А.А.

Новосибирск 2023 г.

Содержание

[1. Задание 3](#_Toc153307169)

[2. Исходный код программы 6](#_Toc153307170)

[2.1. Код программы 6](#_Toc153307171)

[2.2. Код тестов 10](#_Toc153307172)

[3. Результаты модульных тестов 14](#_Toc153307173)

[4. Вывод 15](#_Toc153307174)

1. **Задание**
2. В соответствии с приведенной ниже спецификацией реализуйте шаблон классов «множество». Для тестирования в качестве параметра шаблона T выберите типы:

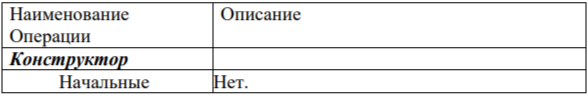
• int;

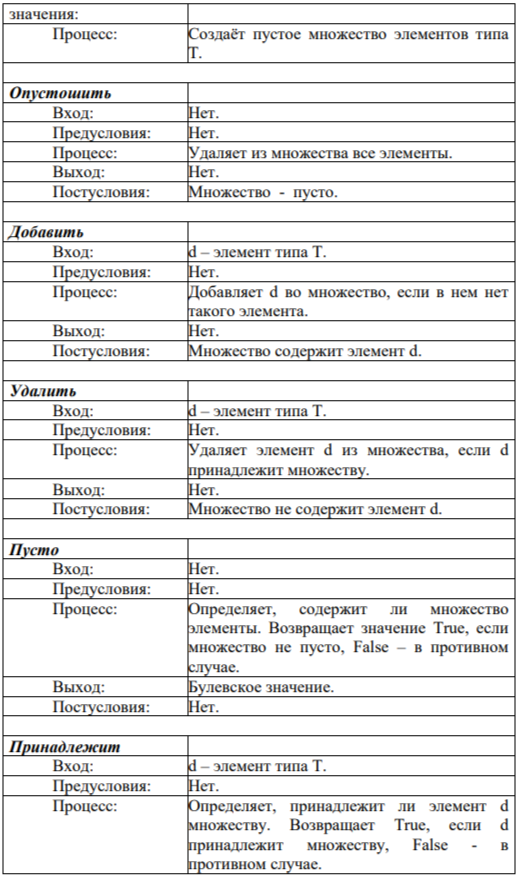
1. Протестировать каждую операцию, определенную на типе данных, используя средства модульного тестирования.
2. Если необходимо, предусмотрите возбуждение исключительных ситуаций.

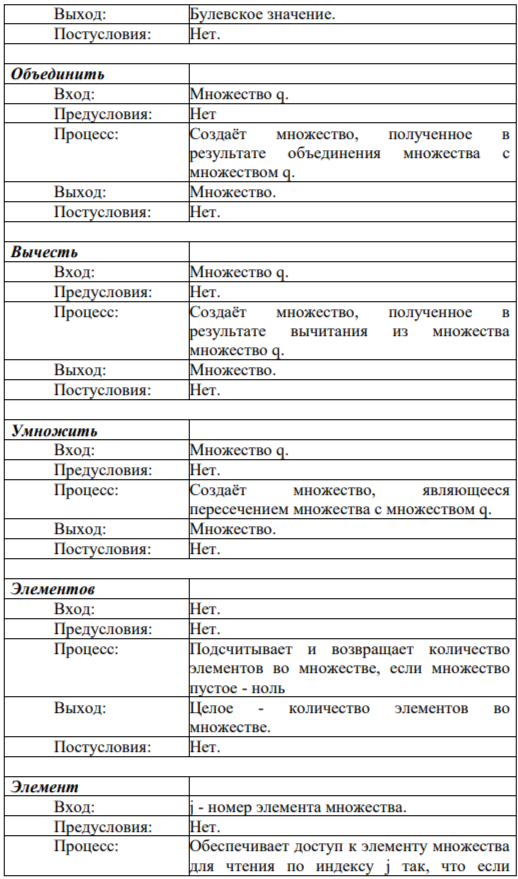
**Данные**

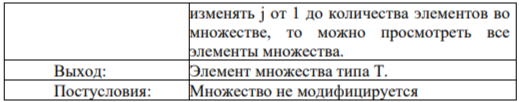
Множества - это изменяемые неограниченные наборы элементов типа T. Содержимое множества изменяется следующими операциями: • Опустошить (опустошение множества); • Добавить (добавление элемента во множество); • Удалить (извлечение элемента из множества). Множество поддерживает следующую дисциплину записи и извлечения элементов: элемент может присутствовать во множестве только в одном экземпляре, при извлечении выбирается заданный элемент множества и удаляется из множества.

**Операции**









1. **Исходный код программы**
   1. **Код программы**

**Set.h**

#pragma once

#include <set>

using namespace std;

template<class T>

class MegaSet

{

private:

set<T> data;

public:

MegaSet() {};

MegaSet(set<T> newData);

void insert(T a);

void remove(T a);

void clear();

bool isEmpty();

bool isCount(T a);

MegaSet<T> add(MegaSet<T> a);

MegaSet<T> dif(MegaSet<T> a);

MegaSet<T> mul(MegaSet<T> a);

T operator[](int i);

int size();

};

template<class T>

inline int MegaSet<T>::size()

{

return data.size();

}

template<class T>

inline MegaSet<T>::MegaSet(set<T> newData):data(newData)

{

}

template<class T>

inline void MegaSet<T>::insert(T a)

{

data.insert(a);

}

template<class T>

inline void MegaSet<T>::remove(T a)

{

data.erase(a);

}

template<class T>

inline void MegaSet<T>::clear()

{

data.clear();

}

template<class T>

inline MegaSet<T> MegaSet<T>::add(MegaSet<T> a)

{

MegaSet<T> res(data);

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

{

res.insert(a[i]);

}

return res;

}

template<class T>

inline MegaSet<T> MegaSet<T>::dif(MegaSet<T> a)

{

MegaSet<T> res;

for (auto i : data)

{

if (!a.isCount(i))

res.insert(i);

}

return res;

}

template<class T>

inline MegaSet<T> MegaSet<T>::mul(MegaSet<T> a)

{

MegaSet<T> res;

for (auto i : data)

{

if (a.isCount(i))

res.insert(i);

}

return res;

}

template<class T>

inline T MegaSet<T>::operator[](int id)

{

int i = 0;

for (auto t : data)

{

if (id == i)

return t;

++i;

}

throw "index out of range";

}

template<class T>

inline bool MegaSet<T>::isEmpty()

{

return data.empty();

}

template<class T>

inline bool MegaSet<T>::isCount(T a)

{

return data.count(a);

}

* 1. **Код тестов**

**UnitTest.cpp**

#include "pch.h"

#include "CppUnitTest.h"

#include "../ModernCodingMnoshestvo/MegaSet.h"

using namespace Microsoft::VisualStudio::CppUnitTestFramework;

namespace UnitTest

{

TEST\_CLASS(UnitTest)

{

public:

TEST\_METHOD(TestOperator)

{

MegaSet<int> a;

a.insert(1);

a.insert(2);

int res = a[0];

int expected = 1;

Assert::AreEqual(expected, res);

}

TEST\_METHOD(TestRemove)

{

MegaSet<int> a;

a.insert(1);

a.insert(2);

a.insert(3);

a.remove(2);

int res = a[1];

int expected = 3;

Assert::AreEqual(expected, res);

}

TEST\_METHOD(TestClear)

{

MegaSet<int> a;

a.insert(1);

a.insert(2);

a.insert(3);

a.clear();

int res = a.size();

int expected = 0;

Assert::AreEqual(expected, res);

}

TEST\_METHOD(TestAdd)

{

MegaSet<int> a;

a.insert(1);

a.insert(2);

a.insert(3);

MegaSet<int> b;

b.insert(4);

MegaSet<int> c;

c = a.add(b);

MegaSet<int> res = c;

int expected[] = {1, 2, 3, 4};

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

{

Assert::AreEqual(expected[i], res[i]);

}

}

TEST\_METHOD(TestDif)

{

MegaSet<int> a;

a.insert(1);

a.insert(2);

a.insert(3);

MegaSet<int> b;

b.insert(3);

MegaSet<int> c;

c = a.dif(b);

MegaSet<int> res = c;

int expected[] = { 1, 2 };

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

{

Assert::AreEqual(expected[i], res[i]);

}

}

TEST\_METHOD(TestMul)

{

MegaSet<int> a;

a.insert(1);

a.insert(2);

a.insert(3);

MegaSet<int> b;

b.insert(3);

b.insert(4);

b.insert(2);

MegaSet<int> c;

c = a.mul(b);

MegaSet<int> res = c;

int expected[] = { 2, 3 };

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

{

Assert::AreEqual(expected[i], res[i]);

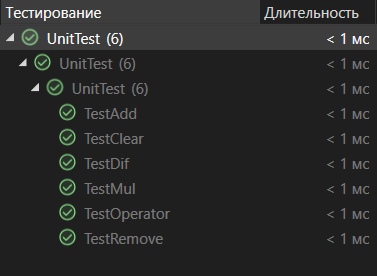
}

}

};

}

1. **Результаты модульных тестов**



1. **Вывод**

По итогам данной лабораторной работе были сформированы практические навыки реализации шаблона класса множество с помощью классов С++ и их модульного тестирования.