Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет телекоммуникаций и информатики» (СибГУТИ)

Кафедра ПМиК

**Лабораторная работа № 13**

**по дисциплине «Современные технологии программирования»**

**«Абстрактный тип данных (ADT) Множество (на шаблоне)»**

Выполнил: студент 4 курса ф. ИВТ, гр. ИП-813

Пещеров Вячеслав Александрович

Проверил: ассистент

к. ПМиК Агалаков А.А.

Новосибирск, 2021

Оглавление

[**Задание** 3](#_Toc90053601)

[**Реализация** 7](#_Toc90053602)

[**Демонстрация работы** 8](#_Toc90053603)

[**Вывод** 9](#_Toc90053604)

[**Список литературы** 10](#_Toc90053605)

[**Приложение** 11](#_Toc90053606)

[Листинг 1. UnitTests.cpp 11](#_Toc90053607)

[Листинг 2. TSet.h 14](#_Toc90053608)

[Листинг 3. TSet.cpp 15](#_Toc90053609)

# **Задание**

1. В соответствии с приведенной ниже спецификацией реализуйте шаблон классов «множество». Для тестирования в качестве параметра шаблона T выберите типы:
   * int;
   * TFrac (простая дробь), разработанный вами ранее.
2. Протестировать каждую операцию, определенную на типе данных, используя средства модульного тестирования.
3. Если необходимо, предусмотрите возбуждение исключительных ситуаций.

Спецификация типа данных «множество»

**ADT** tset

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

Операции могут вызываться только объектом «множество» (тип tset), указатель на который передаётся в них по умолчанию. При описании операций этот объект в разделе «Вход» не указывается.

Таблица 1. Описание операций на ADT tset.

|  |  |
| --- | --- |
| Наименование  Операции | Описание |
| ***Конструктор*** |  |
| Начальные | Нет. |

|  |  |
| --- | --- |
| значения: |  |
| Процесс: | Создаёт пустое множество элементов типа  T. |
|  | |
| ***Опустошить*** |  |
| Вход: | Нет. |
| Предусловия: | Нет. |
| Процесс: | Удаляет из множества все элементы. |
| Выход: | Нет. |
| Постусловия: | Множество - пусто. |
|  | |
| ***Добавить*** |  |
| Вход: | d – элемент типа Т. |
| Предусловия: | Нет. |
| Процесс: | Добавляет d во множество, если в нем нет  такого элемента. |
| Выход: | Нет. |
| Постусловия: | Множество содержит элемент d. |
|  | |
| ***Удалить*** |  |
| Вход: | d – элемент типа Т. |
| Предусловия: | Нет. |
| Процесс: | Удаляет элемент d из множества, если d  принадлежит множеству. |
| Выход: | Нет. |
| Постусловия: | Множество не содержит элемент d. |
|  | |
| ***Пусто*** |  |
| Вход: | Нет. |
| Предусловия: | Нет. |
| Процесс: | Определяет, содержит ли множество элементы. Возвращает значение True, если множество не пусто, False – в противном  случае. |
| Выход: | Булевское значение. |
| Постусловия: | Нет. |
|  | |
| ***Принадлежит*** |  |
| Вход: | d – элемент типа Т. |
| Предусловия: | Нет. |
| Процесс: | Определяет, принадлежит ли элемент d множеству. Возвращает True, если d принадлежит множеству, False - в  противном случае. |

|  |  |
| --- | --- |
| Выход: | Булевское значение. |
| Постусловия: | Нет. |
|  | |
| ***Объединить*** |  |
| Вход: | Множество q. |
| Предусловия: | Нет |
| Процесс: | Создаёт множество, полученное в результате объединения множества с  множеством q. |
| Выход: | Множество. |
| Постусловия: | Нет. |
|  | |
| ***Вычесть*** |  |
| Вход: | Множество q. |
| Предусловия: | Нет. |
| Процесс: | Создаёт множество, полученное в результате вычитания из множества  множество q. |
| Выход: | Множество. |
| Постусловия: | Нет. |
|  | |
| ***Умножить*** |  |
| Вход: | Множество q. |
| Предусловия: | Нет. |
| Процесс: | Создаёт множество, являющееся  пересечением множества с множеством q. |
| Выход: | Множество. |
| Постусловия: | Нет. |
|  | |
| ***Элементов*** |  |
| Вход: | Нет. |
| Предусловия: | Нет. |
| Процесс: | Подсчитывает и возвращает количество  элементов во множестве, если множество пустое - ноль |
| Выход: | Целое - количество элементов во  множестве. |
| Постусловия: | Нет. |
|  | |
| ***Элемент*** |  |
| Вход: | j - номер элемента множества. |
| Предусловия: | Нет. |
| Процесс: | Обеспечивает доступ к элементу множества  для чтения по индексу j так, что если |

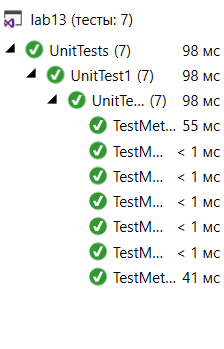
|  |  |
| --- | --- |
|  | изменять j от 1 до количества элементов во множестве, то можно просмотреть все  элементы множества. |
| Выход: | Элемент множества типа Т. |
| Постусловия: | Множество не модифицируется |

end tset

# **Реализация**

В данной работе мы создали абстрактный тип данных множество и все необходимые для работы с ним операции, такие как: конструктор, удаление из множества всех элементов, добавление во множество элементов, удалить элемент, проверка содержания множества, принадлежат ли элементы, объединение элементов, вычитание элементов, умножение элементов, доступ к элементу.

# **Демонстрация работы**



# **Вывод**

Мы научились работать в среде Visual Studio, а именно разрабатывать в ней модульные тесты для тестирования наших функции и классов на языке C#. Данная среда отлично подходит для выполнения модульного тестирования и автоматизации.

# **Список литературы**

1. Подбельский В.В., Фомин С.С.инт Курс программирования на языке Си: учебник. – М.:ДМК Пресс, 2012 – 384 с.
2. Павловская Т.А. C#. Программирование на языке высокого уровня: Учебник для вузов. - СПб. : Питер, 2014 - 432 с. : ил. - (Серия "Учебник для вузов").
3. Рихтер Дж. CLR via C#. Программирование на платформе Microsoft.NET Framework 4 на языке C# . 3-е изд.: - СПб.:Питер, 2012 - 928 с. : ил.
4. Котляров, В. П. Основы тестирования программного обеспечения : учебное пособие для СПО / В. П. Котляров. — Саратов : Профобразование, 2019 — 335 c. — ISBN 978-5-4488-0364-2. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/86202.html (дата обращения: 21.08.2020). — Режим доступа: для авторизир. пользователей

# **Приложение**

## Листинг 1. UnitTests.cpp

#include "pch.h"

#include "CppUnitTest.h"

#include <algorithm>

#include "../lab13/TSet.h"

//#include "../lab13/TSet1.h"

#include "../lab13/TSet.cpp"

//#include "../lab13/TSet1.cpp"

#include "../lab13/TFrac.h"

#include "../lab13/TFrac.cpp"

using namespace Microsoft::VisualStudio::CppUnitTestFramework;

namespace UnitTest1

{

TEST\_CLASS(UnitTest1)

{

public:

TSet<int> t0, t1, res;

string str = "", strRes;

TEST\_METHOD(TestMethod1)

{

for (int i = 1; i < 10; i++) {

t0.add(i);

t1.add(2 + i);

}

for (int i = 0; i < t0.count(); i++)

str += to\_string(t0.element(i)) + " ";

strRes = "1 2 3 4 5 6 7 8 9 ";

Assert::AreEqual(strRes, str);

}

TEST\_METHOD(TestMethod2)

{

for (int i = 1; i < 10; i++) {

//t0.add(i);

t1.add(2 + i);

}

str.clear();

for (int i = 0; i < t1.count(); i++)

str += to\_string(t1.element(i)) + " ";

strRes = "3 4 5 6 7 8 9 10 11 ";

Assert::AreEqual(strRes, str);

}

TEST\_METHOD(TestMethod3)

{

for (int i = 1; i < 10; i++) {

t0.add(i);

t1.add(2 + i);

}

res = t0.concat(t1);

str.clear();

for (int i = 0; i < res.count(); i++)

str += to\_string(res.element(i)) + " ";

strRes = "1 2 3 4 5 6 7 8 9 10 11 ";

Assert::AreEqual(strRes, str);

}

TEST\_METHOD(TestMethod4)

{

for (int i = 1; i < 10; i++) {

t0.add(i);

t1.add(2 + i);

}

res = t0.subtract(t1);

str.clear();

for (int i = 0; i < res.count(); i++)

str += to\_string(res.element(i)) + " ";

strRes = "1 2 ";

Assert::AreEqual(strRes, str);

}

TEST\_METHOD(TestMethod5)

{

for (int i = 1; i < 10; i++) {

t0.add(i);

t1.add(2 + i);

}

res = t0.multiply(t1);

str.clear();

for (int i = 0; i < res.count(); i++)

str += to\_string(res.element(i)) + " ";

strRes = "3 4 5 6 7 8 9 ";

Assert::AreEqual(strRes, str);

}

TEST\_METHOD(TestMethod6)

{

for (int i = 1; i < 10; i++) {

res.add(i);

}

res.remove(4);

str.clear();

for (int i = 0; i < res.count(); i++)

str += to\_string(res.element(i)) + " ";

strRes = "1 2 3 5 6 7 8 9 ";

Assert::AreEqual(strRes, str);

}

TEST\_METHOD(TestMethod7)

{

res.clear();

bool check = res.empty();

Assert::AreEqual(true, check);

}

};

}

## Листинг 2. TSet.h

#pragma once

#include <set>

using namespace std;

template <class T>

class TSet {

set<T> tset;

public:

TSet();

TSet(const TSet& q);

void clear(void);

void add(const T& d);

void remove(const T& d);

bool empty(void);

bool contains(const T& d);

TSet<T> concat(const TSet<T>& q);

TSet<T> subtract(const TSet<T>& q);

TSet<T> multiply(const TSet<T>& q);

int count(void);

T element(const int& j);

~TSet();

};

## Листинг 3. TSet.cpp

#include "TSet.h"

#include <iterator>

template<class T>

TSet<T>::TSet() {}//конструктор

template<class T>

TSet<T>::TSet(const TSet& q)//сэтер

{

tset = q.tset;

}

template<class T>

void TSet<T>::clear(void)//очищение

{

tset.clear();

}

template<class T>

void TSet<T>::add(const T& d)//добавление числа

{

tset.insert(d);

}

template<class T>

void TSet<T>::remove(const T& d)// убираем число

{

if (tset.count(d))

tset.erase(d);

}

template<class T>

bool TSet<T>::empty(void)// проверяем на пустоту

{

return tset.empty();

}

template<class T>

bool TSet<T>::contains(const T& d)//принадлежит ли

{

return tset.count(d);

}

template<class T>

TSet<T> TSet<T>::concat(const TSet<T>& q)//объединение множест

{

TSet<T> result;

set\_union(tset.begin(), tset.end(), q.tset.begin(), q.tset.end(), inserter(result.tset, result.tset.begin()));

return result;

}

template<class T>

TSet<T> TSet<T>::subtract(const TSet<T>& q)//вычетание множеств

{

TSet<T> result;

set\_difference(tset.begin(), tset.end(), q.tset.begin(), q.tset.end(), inserter(result.tset, result.tset.begin()));

return result;

}

template<class T>

TSet<T> TSet<T>::multiply(const TSet<T>& q)//умножение множеств

{

TSet<T> result;

set\_intersection(tset.begin(), tset.end(), q.tset.begin(), q.tset.end(), inserter(result.tset, result.tset.begin()));

for (const T& elem : q.tset)

if (tset.count(elem))

result.add(elem);

return result;

}

template<class T>

int TSet<T>::count(void)//количество элементов

{

return tset.size();

}

template<class T>

T TSet<T>::element(const int& j)//посмотреть до какого-то элемента

{

if (j < 0 || j >= tset.size())

return T();

T it = \*next(tset.begin(), j);

return it;

}

template<class T>

TSet<T>::~TSet() {

tset.clear();

}