Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет телекоммуникаций и информатики»

(СибГУТИ)

Кафедра прикладной математики и кибернетики

Лабораторная работа № 7

по дисциплине «Современные технологии программирования»

Выполнил:

студент группы ИП-712

Алексеев Степан Владимирович

ФИО студента

Работу проверил:

ассистент кафедры Агалаков А.А.

ФИО преподавателя

Новосибирск 2020 г.

Оглавление

[ЗАДАНИЕ 2](#_Toc53920531)

[ТЕСТОВЫЕ НАБОРЫ ДАННЫХ 3](#_Toc53920532)

[ДЕМОНСТРАЦИЯ РАБОТЫ ПРОГРАММЫ 4](#_Toc53920533)

[ВЫВОД 5](#_Toc53920534)

[ПРИЛОЖЕНИЕ 6](#_Toc53920535)

[Листинг 1. TEditor.cs 6](#_Toc53920536)

[Листинг 2. TEditorTests.cs 7](#_Toc53920537)

# ЗАДАНИЕ

1. Разработать и реализовать класс TEditor «Редактор р-ичных чисел», используя класс С++.

2. Протестировать каждую операцию, определенную на типе данных, используя средства модульного тестирования.

3. Если необходимо, предусмотрите возбуждение исключительных ситуаций.

# ТЕСТОВЫЕ НАБОРЫ ДАННЫХ

Для тестирования в числа вводились неверные символы: te.addADigitOrALetter(tp, "Z", 4); для проверки выброса исключения.

Для проверки работы функции обнуления вводилось:

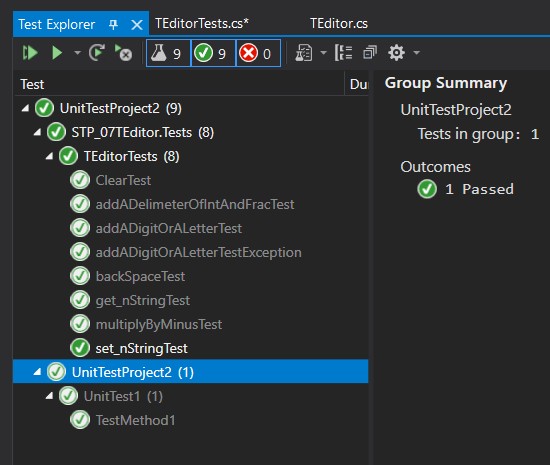
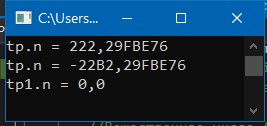
TPNumber tp = new TPNumber("AC4D,A5", 16, 7);

TEditor te = new TEditor();

te.Clear(tp);

Assert.AreEqual("0,0", tp.n);

# ДЕМОНСТРАЦИЯ РАБОТЫ ПРОГРАММЫ



# ВЫВОД

Научился генерировать тесты автоматически. Задумался о необходимости хранения чисел в неизменяемом виде. Исправил предыдущую лабораторную для большего удобства использоания с помощью нового класса.

# ПРИЛОЖЕНИЕ

## Листинг 1. TEditor.cs

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using STP\_06\_TPNumber;

namespace STP\_07TEditor

{

public class TEditor

{

public TEditor()

{

}

static void Main(string[] args)

{

TPNumber tp = new TPNumber(546.164, 16, 7);

Console.WriteLine("tp.n = " + tp.n);

TEditor te = new TEditor();

te.multiplyByMinus(tp);

te.addADigitOrALetter(tp, "B", 3);

Console.WriteLine("tp.n = " + tp.n);

TPNumber tp1 = new TPNumber("0,0", 3, 7);

Console.WriteLine("tp1.n = " + tp1.n);

Console.ReadLine();

}

public void addADigitOrALetter(TPNumber tp, string newElement, int position)

{

if (tp.alphabet.Contains(newElement))

tp.n = tp.n.Insert(position, newElement);

else

{

Console.WriteLine("The newElement is not in the alphabet of the number");

throw new WrongInput();

}

}

public void multiplyByMinus(TPNumber tp)

{

if (tp.n.Substring(0, 1) == "-")

tp.n = tp.n.Substring(1);//если уже стоит минус, то убираем его

else

{

tp.n = "-" + tp.n;

}

}

public void addADelimeterOfIntAndFrac(TPNumber tp, int index)

{

int ind = tp.n.IndexOf(",");

tp.n = tp.n.Remove(ind, 1);

tp.n = tp.n.Insert(index, ",");

}

public void backSpace(TPNumber tp)

{

tp.n = tp.n.Remove(tp.n.Length - 1, 1);

}

public void Clear(TPNumber tp)

{

tp.n = "0,0";

}

public string get\_nString(TPNumber tp)

{

return tp.getn();

}

public void set\_nString(TPNumber tp, string newN)

{

tp.n = newN;

}

}

public class WrongInput : Exception

{

public WrongInput()

{

Console.WriteLine("wrong input in constructor exception");

}

}

}

## Листинг 2. TEditorTests.cs

using Microsoft.VisualStudio.TestTools.UnitTesting;

using STP\_07TEditor;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using STP\_06\_TPNumber;

namespace STP\_07TEditor.Tests

{

[TestClass()]

public class TEditorTests

{

[TestMethod()]

public void addADigitOrALetterTestException()

{

bool exceptionThrown = false;

TPNumber tp = new TPNumber(546.164, 16, 7);

TEditor te = new TEditor();

try

{

te.addADigitOrALetter(tp, "Z", 4);

}

catch (Exception)

{

exceptionThrown = true;

}

Assert.IsTrue(exceptionThrown);

}

[TestMethod()]

public void addADigitOrALetterTest()

{

bool exceptionThrown = false;

TPNumber tp = new TPNumber("AC4D,A5", 16, 7);

TEditor te = new TEditor();

try

{

te.addADigitOrALetter(tp, "B", 4);

}

catch (Exception)

{

exceptionThrown = true;

}

Assert.AreEqual("AC4DB,A5", tp.n);

}

[TestMethod()]

public void multiplyByMinusTest()

{

TPNumber tp = new TPNumber("AC4D,A5", 16, 7);

TEditor te = new TEditor();

te.multiplyByMinus(tp);

Assert.AreEqual("-AC4D,A5", tp.n);

}

[TestMethod()]

public void addADelimeterOfIntAndFracTest()

{

TPNumber tp = new TPNumber("AC4D,A5", 16, 7);

TEditor te = new TEditor();

te.addADelimeterOfIntAndFrac(tp, 2);

Assert.AreEqual("AC,4DA5", tp.n);

}

[TestMethod()]

public void backSpaceTest()

{

TPNumber tp = new TPNumber("AC4D,A5", 16, 7);

TEditor te = new TEditor();

te.backSpace(tp);

Assert.AreEqual("AC4D,A", tp.n);

}

[TestMethod()]

public void ClearTest()

{

TPNumber tp = new TPNumber("AC4D,A5", 16, 7);

TEditor te = new TEditor();

te.Clear(tp);

Assert.AreEqual("0,0", tp.n);

}

TPNumber tpGeneral = new TPNumber("AC4D,A5", 16, 7);

TEditor teGeneral = new TEditor();

[TestMethod()]

public void get\_nStringTest()

{

string str = teGeneral.get\_nString(tpGeneral);

Assert.AreEqual(str, "AC4D,A5");

}

[TestMethod()]

public void set\_nStringTest()

{

teGeneral.set\_nString(tpGeneral, "123D");

Assert.AreEqual(tpGeneral.n, "123D");

}

}

}