Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет телекоммуникаций и информатики»

(СибГУТИ)

Кафедра прикладной математики и кибернетики

Лабораторная работа № 9

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](#_Toc53942379)

[ТЕСТОВЫЕ НАБОРЫ ДАННЫХ 3](#_Toc53942380)

[ДЕМОНСТРАЦИЯ РАБОТЫ ПРОГРАММЫ 4](#_Toc53942381)

[ВЫВОД 5](#_Toc53942382)

[ПРИЛОЖЕНИЕ 6](#_Toc53942383)

[Листинг 1. TEditorForComplexNumbers.cs 6](#_Toc53942384)

[Листинг 2. TEditorForComplexNumbersTests.cs 7](#_Toc53942385)

# ЗАДАНИЕ

1. Разработать и реализовать класс «Ввод и редактирование комплексных чисел» (TEditor), 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

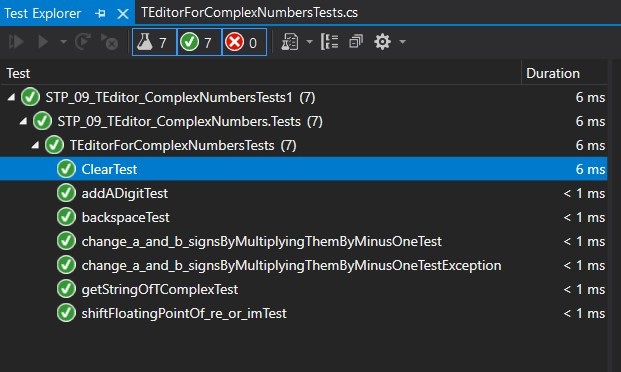
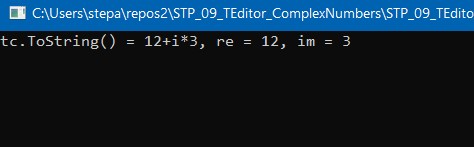
TEditorForComplexNumbers te = new TEditorForComplexNumbers();

TComplex tc = new TComplex(-55, -11);

string str = te.getStringOfTComplex(tc);

Assert.AreEqual(str, "-55-i\*11");

# ДЕМОНСТРАЦИЯ РАБОТЫ ПРОГРАММЫ



# ВЫВОД

Научился включать проекты друг в друга, автоматизировать создание тестов. Убедился в удобстве наличия автоматических тестов при разработке. Разработал методы в соответствии с заданием, интегрировал проекты для тестирования, создания комплексных чисел и их редактирования в один проект.

# ПРИЛОЖЕНИЕ

## Листинг 1. TEditorForComplexNumbers.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using STP\_05\_ComplexNumber;

namespace STP\_09\_TEditor\_ComplexNumbers

{

public class TEditorForComplexNumbers

{

static void Main(string[] args)

{

TComplex tc = new TComplex("12+i\*3");

string im = tc.getImaginaryString();

string re = tc.getRealString();

Console.WriteLine("tc.ToString() = " + tc.ToString() + ", re = " + re + ", im = " + im);

Console.ReadLine();

}

public TComplex addADigit(TComplex tc, int index, int digit)

{

string str = tc.ToString();

str = str.Insert(index, digit.ToString());

return new TComplex(str);

}

public TComplex change\_a\_and\_b\_signsByMultiplyingThemByMinusOne(TComplex tc, int forA, int forB)//forA and forB have to be either 1 or -1

{

if ((forA != 1 && forA != -1) || (forB != 1 && forB != -1))

{

throw new WrongInputException();

}

double a = tc.getRealDouble() \* (double)forA;

double b = tc.getImaginaryDouble() \* (double)forB;

return new TComplex(a, b);

}

public TComplex shiftFloatingPointOf\_re\_or\_im(TComplex tc, int newIndexOfCommaforA, int newIndexOfCommaForB)

{

string a = tc.getRealString();

string b = tc.getImaginaryString();

if (a.Contains(","))

{

int ind = a.IndexOf(",");

a = a.Remove(ind, 1);

}

a = a.Insert(newIndexOfCommaforA, ",");

if (b.Contains(","))

{

int ind = b.IndexOf(",");

b = b.Remove(ind, 1);

}

b = b.Insert(newIndexOfCommaForB, ",");

double aDouble = Double.Parse(a);

double bDouble = Double.Parse(b);

return new TComplex(aDouble, bDouble);

}

public string backspace(TComplex tc)

{

return tc.ToString().Substring(0, tc.ToString().Length - 1);

}

public TComplex Clear(TComplex tc)

{

return new TComplex(0, 0);

}

public string getStringOfTComplex(TComplex tc)

{

return tc.ToString();

}

}

public class WrongInputException : Exception

{

public WrongInputException()

{

Console.WriteLine("wrong input");

}

}

}

## Листинг 2. TEditorForComplexNumbersTests.cs

using Microsoft.VisualStudio.TestTools.UnitTesting;

using STP\_09\_TEditor\_ComplexNumbers;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using STP\_05\_ComplexNumber;

namespace STP\_09\_TEditor\_ComplexNumbers.Tests

{

[TestClass()]

public class TEditorForComplexNumbersTests

{

[TestMethod()]

public void addADigitTest()

{

TEditorForComplexNumbers te = new TEditorForComplexNumbers();

TComplex tc = new TComplex(-55, 10);

tc = te.addADigit(tc, 2, 4);

Assert.AreEqual(tc.ToString(), "-545+i\*10");

}

[TestMethod()]

public void change\_a\_and\_b\_signsByMultiplyingThemByMinusOneTest()

{

TEditorForComplexNumbers te = new TEditorForComplexNumbers();

TComplex tc = new TComplex(-55, 10);

tc = te.change\_a\_and\_b\_signsByMultiplyingThemByMinusOne(tc, -1, -1);

Assert.AreEqual(tc.ToString(), "55-i\*10");

}

[TestMethod()]

public void change\_a\_and\_b\_signsByMultiplyingThemByMinusOneTestException()

{

bool exceptionWasThrown = false;

try

{

TEditorForComplexNumbers te = new TEditorForComplexNumbers();

TComplex tc = new TComplex(-55, 10);

tc = te.change\_a\_and\_b\_signsByMultiplyingThemByMinusOne(tc, -10, -1);

}

catch (WrongInputException)

{

exceptionWasThrown = true;

}

Assert.IsTrue(exceptionWasThrown);

}

[TestMethod()]

public void shiftFloatingPointOf\_re\_or\_imTest()

{

TEditorForComplexNumbers te = new TEditorForComplexNumbers();

TComplex tc = new TComplex(-55, 11);

tc = te.shiftFloatingPointOf\_re\_or\_im(tc, 2, 1);

Assert.AreEqual(tc.ToString(), "-5,5+i\*1,1");

}

[TestMethod()]

public void backspaceTest()

{

TEditorForComplexNumbers te = new TEditorForComplexNumbers();

TComplex tc = new TComplex(-55, 11);

string str = te.backspace(tc);

Assert.AreEqual(str, "-55+i\*1");

}

[TestMethod()]

public void ClearTest()

{

TEditorForComplexNumbers te = new TEditorForComplexNumbers();

TComplex tc = new TComplex(-55, 11);

tc = te.Clear(tc);

Assert.AreEqual(tc.ToString(), "0");

}

[TestMethod()]

public void getStringOfTComplexTest()

{

TEditorForComplexNumbers te = new TEditorForComplexNumbers();

TComplex tc = new TComplex(-55, -11);

string str = te.getStringOfTComplex(tc);

Assert.AreEqual(str, "-55-i\*11");

}

}

}