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 № 10**

**по дисциплине «Современные технологии программирования»**

**«Редактор комплексных чисел»**

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

Пещеров Вячеслав Александрович

Проверил: ассистент

к. ПМиК Агалаков А.А.

Новосибирск, 2021

Оглавление

[**Задание** 3](#_Toc90049904)

[**Реализация** 4](#_Toc90049905)

[**Демонстрация работы** 5](#_Toc90049906)

[**Вывод** 10](#_Toc90049907)

[**Список литературы** 11](#_Toc90049908)

[**Приложение** 12](#_Toc90049909)

[Листинг 1. main.cpp 12](#_Toc90049910)

[Листинг 2. TEditor.h 13](#_Toc90049911)

[Листинг 3. TEditor.cpp 14](#_Toc90049912)

# **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 Visual Studio по критерию

С2.

3. Если необходимо, предусмотрите возбуждение исключительных

ситуаций.

# **Реализация**

В данной работе мы создали редактор комплексного числа и все необходимые для работы с ним операции, такие как: добавить ноль, добавить числа, умножить на минус 1, удалить символ, очистить число, показать число, ввести число как строку, смена режима на левую и правую часть, установить режим на реальную и мнимую часть.

# **Демонстрация работы**

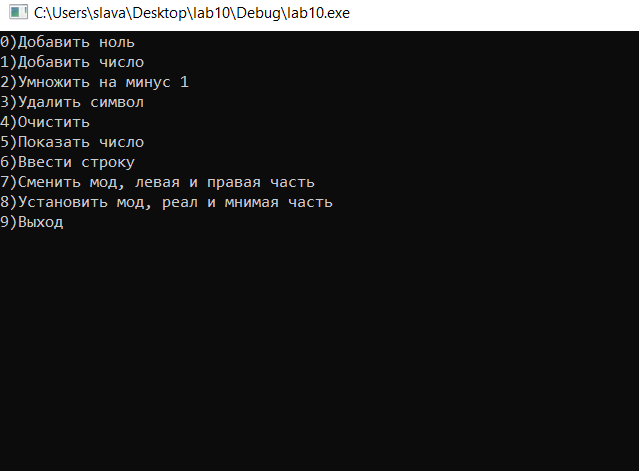


Рисунок 1. Меню

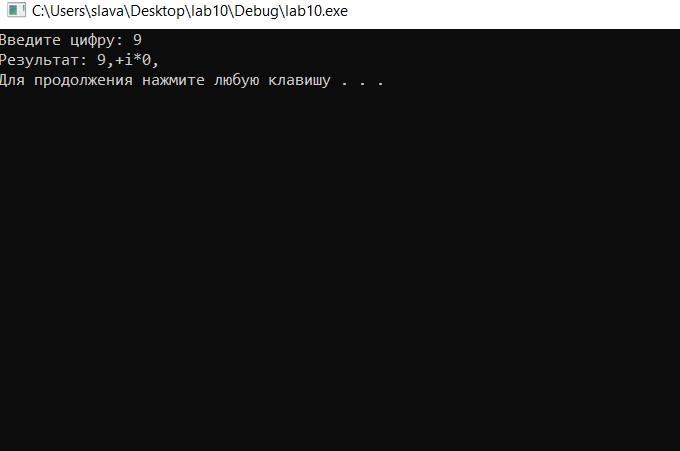


Рисунок 2. Добавляем число

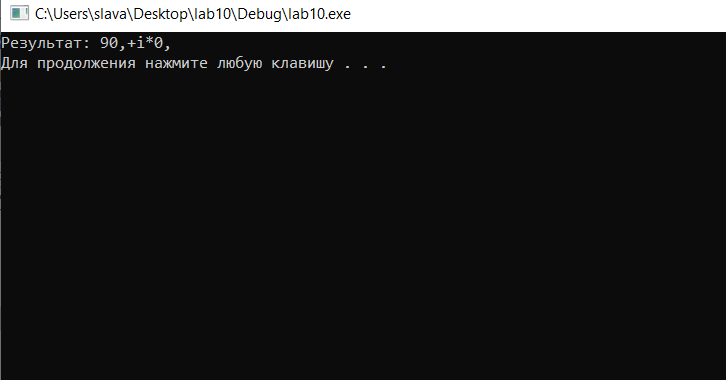


Рисунок 3. Добавляем 0

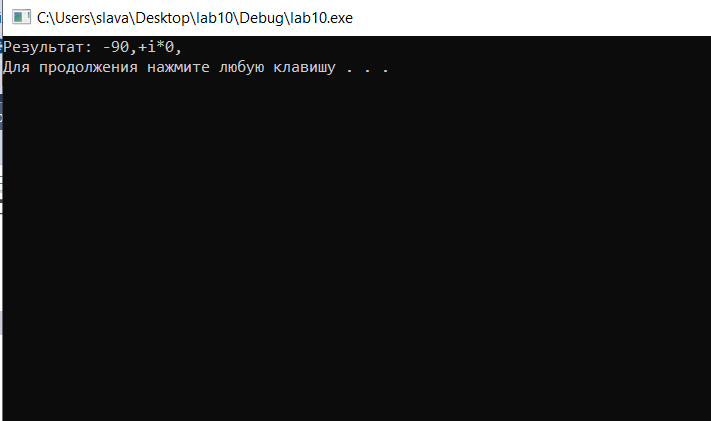


Рисунок 4. Добавляем минус

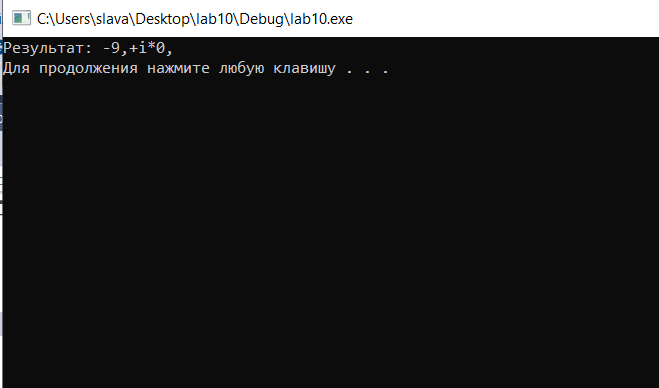


Рисунок 5. Убираем одно число

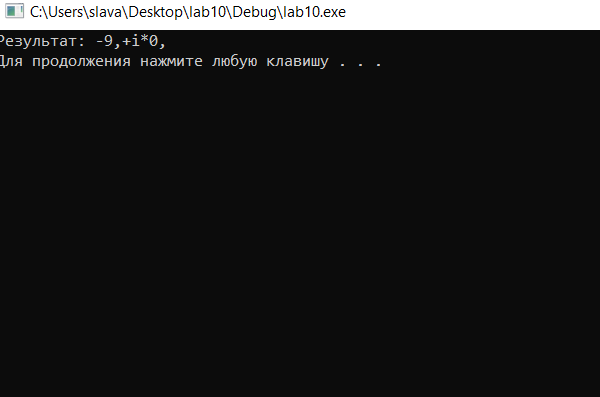


Рисунок 6. Вывод числа

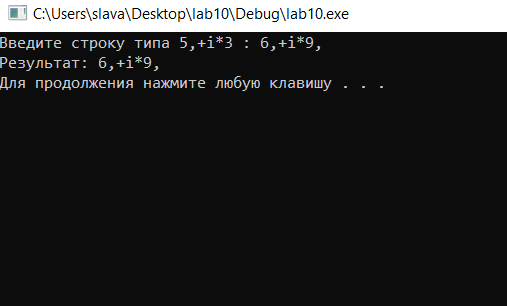


Рисунок 7. Вводим число как строку

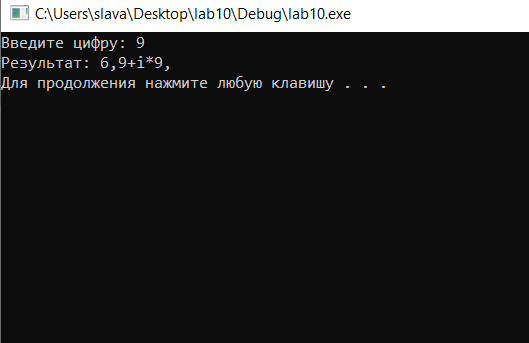


Рисунок 8. Перешли в правый режим и ввели число в правую часть

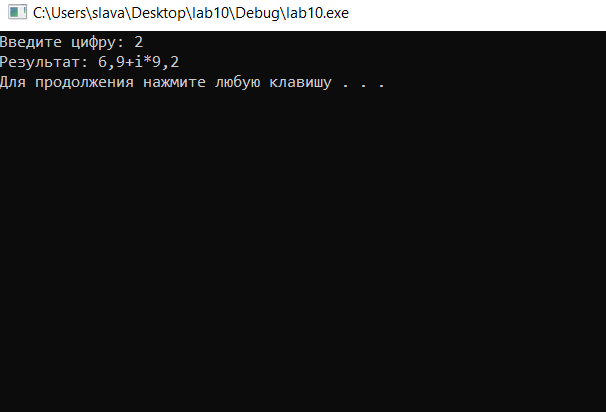


Рисунок 9. Перешли в режим мнимой части и ввели число

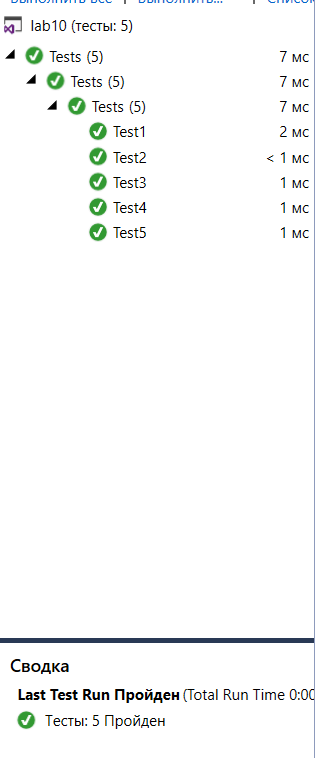


Рисунок 10. Тесты

# **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. main.cpp

#include <iostream>

#include "TEditor.h"

using namespace std;

int main()

{

setlocale(LC\_ALL, "RUS");

TEditor t;

int c;

while (1) {

cout << "0)Добавить ноль" << endl;

cout << "1)Добавить число" << endl;

cout << "2)Умножить на минус 1" << endl;

cout << "3)Удалить символ" << endl;

cout << "4)Очистить" << endl;

cout << "5)Показать число" << endl;

cout << "6)Ввести строку" << endl;

cout << "7)Сменить мод, левая и правая часть " << endl;

cout << "8)Установить мод, реал и мнимая часть " << endl;

cout << "9)Выход" << endl;

cin >> c;

if (c == 9) break;

system("cls");

cout << "Результат: " << t.edit((commandType)c) << endl;

system("pause");

system("cls");

}

return 0;

}

// Запуск программы: CTRL+F5 или меню "Отладка" > "Запуск без отладки"

// Отладка программы: F5 или меню "Отладка" > "Запустить отладку"

## Листинг 2. TEditor.h

#pragma once

#include <iostream>

#include <string>

using namespace std;

enum partToEdit { REAL, IMAG };

enum numberPartToEdit { LEFT, RIGHT };

enum commandType { ADD\_ZERO, ADD\_NUMBER, ADD\_SIGN, DEL\_CHAR, CLEAR, READ, WRITE, MODE\_NUMBER, MODE };

class TEditor

{

private:

const string strZero = "0,+i\*0,";

const string strSeparatorNumber = ",";

const string strSeparatorParts = "i\*";

partToEdit mode;

numberPartToEdit numberMode;

string strComplex;

public:

TEditor();

~TEditor();

bool isComplexNumberZero();

string addSign();

string addNumber(int a);

string addZero();

string deleteCharacter();

string clear();

string readString();

void writeString(string a);

string edit(commandType a);

void setModeNumber(numberPartToEdit a);

void setMode(partToEdit a);

};

## Листинг 3. TEditor.cpp

#include "TEditor.h"

#include <regex>

TEditor::TEditor() {

clear();

}

TEditor::~TEditor() {}

bool TEditor::isComplexNumberZero()//проверка на 0

{

string tmp = strComplex;

if (tmp.at(0) == '-')

tmp.erase(tmp.begin());

replace(tmp.begin(), tmp.end(), '-', '+');

return tmp == strZero;

}

string TEditor::addSign()//изменить знак

{

if (mode == REAL) {

if (strComplex[0] == '-')

strComplex = strComplex.substr(1, strComplex.size());

else strComplex = '-' + strComplex;

}

else {

int index = strComplex.find\_last\_of('-');

if (index == 0 || index == -1)

strComplex[strComplex.find(strSeparatorParts) - 1] = '-';

else strComplex[index] = '+';

}

return strComplex;

}

string TEditor::addNumber(int a)//добавить число

{

if (a < 0 || a > 9)

return strComplex;

int indexSep = strComplex.find(strSeparatorParts);

if (mode == REAL) {

if (numberMode == LEFT) {

if (strComplex[0] == '0')

strComplex[0] = '0' + a;

else if (strComplex[0] == '-' && strComplex[1] == '0')

strComplex[0] = '0' + a;

else

strComplex.insert(strComplex.find\_first\_of(strSeparatorNumber), 1, ('0' + a));

}

else {

if (indexSep > 0)

strComplex.insert(indexSep - 1, 1, ('0' + a));

else strComplex += '0' + a;

}

}

else {

if (numberMode == LEFT) {

indexSep += 2;

if (strComplex[indexSep] == '0')

strComplex[indexSep] = '0' + a;

else

strComplex.insert(strComplex.find\_last\_of(strSeparatorNumber), 1, ('0' + a));

}

else strComplex += '0' + a;

}

return strComplex;

}

string TEditor::addZero()//добавить 0

{

return addNumber(0);

}

string TEditor::deleteCharacter()//убрать символ

{

int indexSep = strComplex.find(strSeparatorParts);

if (mode == REAL) {

if (numberMode == LEFT) {

if (strComplex[0] == '0' || (strComplex[0] == '-' && strComplex[1] == '0'))

return strComplex;

else {

strComplex.erase(strComplex.find\_first\_of(strSeparatorNumber) - 1, 1);

if (strComplex[0] == ',' || (strComplex[0] == '-' && strComplex[1] == ','))

strComplex.insert(strComplex.find\_first\_of(strSeparatorNumber), 1, '0');

}

}

else {

if (!isdigit(strComplex[indexSep - 2]))

return strComplex;

strComplex.erase(indexSep - 2, 1);

}

}

else {

if (numberMode == LEFT) {

indexSep += 2;

if (strComplex[indexSep] == '0')

return strComplex;

else {

strComplex.erase(strComplex.find\_last\_of(strSeparatorNumber) - 1, 1);

if (strComplex[indexSep] == ',')

strComplex.insert(strComplex.find\_last\_of(strSeparatorParts) + 1, 1, '0');

}

}

else {

if (strComplex[strComplex.size() - 1] == ',')

return strComplex;

else strComplex.erase(strComplex.size() - 1, 1);

}

}

return strComplex;

}

string TEditor::clear()//очистить число

{

strComplex = strZero;

mode = REAL;

numberMode = LEFT;

return strComplex;

}

string TEditor::readString()//считать число

{

return strComplex;

}

void TEditor::writeString(string a)//записать число

{

regex regex("-?([0-9]+),([0-9]\*)(\\+|-)i\\\*([0-9]+),([0-9]\*)");

if (regex\_match(a, regex))

strComplex = a;

}

void TEditor::setModeNumber(numberPartToEdit a)//сменить режим

{

numberMode = a;

}

void TEditor::setMode(partToEdit a)//сменить режим между мнимой частью

{

mode = a;

}

string TEditor::edit(commandType a)//редактор

{

string res, f;

switch (a) {

case ADD\_ZERO:

res = addZero();

break;

case ADD\_NUMBER:

int number;

cout << "Введите цифру: ";

cin >> number;

res = addNumber(number);

break;

case ADD\_SIGN:

res = addSign();

break;

case DEL\_CHAR:

res = deleteCharacter();

break;

case CLEAR:

res = clear();

break;

case READ:

res = readString();

break;

case WRITE:

cout << "Введите строку типа 5,+i\*3 : ";

cin >> f;

writeString(f);

res = strComplex;

break;

case MODE\_NUMBER:

cout << "0 - для перехода в левую часть \n1 - для правой части: ";

int modeNumber;

cin >> modeNumber;

modeNumber ? setModeNumber(RIGHT) : setModeNumber(LEFT);

break;

case MODE:

cout << "0 для реал \n1-мнимая часть: ";

int mode;

cin >> mode;

mode ? setMode(IMAG) : setMode(REAL);

break;

default:

res = strComplex;

break;

}

return res;

}