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 № 6**

**по дисциплине «Современные технологии программирования»**

**«Абстрактный тип данных “комплексное число”»**

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

Пещеров Вячеслав Александрович

Проверил: ассистент

к. ПМиК Агалаков А.А.

Новосибирск, 2021

Оглавление

[**Задание** 3](#_Toc90045463)

[**Реализация** 12](#_Toc90045464)

[**Демонстрация работы** 13](#_Toc90045465)

[**Вывод** 14](#_Toc90045466)

[**Список литературы** 15](#_Toc90045467)

[**Приложение** 16](#_Toc90045468)

[Листинг 1. TComplex.h 16](#_Toc90045469)

[Листинг 2. TComplex.cpp 17](#_Toc90045470)

# **Задание**

1. Реализовать абстрактный тип данных «комплексное число», используя класс С++, в соответствии с приведенной ниже спецификацией.
2. Протестировать каждую операцию, определенную на типе данных, используя средства модульного тестирования.
3. Если необходимо, предусмотрите возбуждение исключительных ситуаций.

**Данные**

Комплексное число TComplex - это неизменяемая пара вещественных чисел, представляющие действительную и мнимую части комплексного числа (a + i\*b).

Операции

Операции могут вызываться только объектом комплексное число (тип TComplex), указатель на который в них передаётся по умолчанию. При описании операций этот объект называется this «само число».

|  |  |
| --- | --- |
| ***КонструкторЧисло*** |  |
| Вход: | Пара вещественных чисел a и b. |
| Предусловия: | Нет. |
| Процесс: | Устанавливает значения a, b в поля экземпляра класса  комплексное число (тип TComplex) this.a - |

|  |  |
| --- | --- |
|  | действительной частью и this.b мнимая часть.  Например:  *КонструкторЧисло* (6,3)=6 + i\*3  *КонструкторЧисло* (3,0)=3 + i\*0  *КонструкторЧисло* (0,0)=0 + i\*0 |
| Постусловия: | Поля объекта this инициализированы входными  данными. |
| Выход: | Нет. |
|  | |
| ***КонструкторСтрока*** |  |
| Вход: | Строка f, представляющая комплексное число. |
| Предусловия: | Изображение комплексного числа во входной строке f  должно быть представлено в заданном формате. |
| Процесс: | Выделяет из строки f = ’a + i\*b’, действительную частью (a) и комплексную часть (b) и преобразует их в число. Устанавливает значения a, b в поля экземпляра класса комплексное число (тип TComplex) this.a - действительной частью и this.b мнимая часть.  Например:  *КонструкторСтрока*(‘6+i\*3’) = 6+i\*3  *КонструкторСтрока*(‘0+i\*3’) = 0+i\*3 |
| Постусловия: | Поля объекта this инициализированы входными  данными. |
| Выход: | Нет. |
|  | |
| **Копировать:** |  |
| Вход: | Нет. |
| Предусловия: | Нет. |

|  |  |
| --- | --- |
| Процесс | Создаёт и возвращает собственную копию - комплексное число (тип TComplex) с действительной и мнимой частями такими же, как у самого числа this. |
| Выход: | Комплексное число (тип TComplex).  Например:  c = 6+i3, Копировать(c) = 6+i3 |
| Постусловия: | Нет. |
|  | |
| ***Сложить*** |  |
| Вход: | Комплексное число d (тип TComplex). |
| Предусловия: | Нет. |
| Процесс | Создаёт и возвращает комплексное число, полученное сложением самого числа this = a1+i\*b1 с числом d = a2+i\*b2: ((a1+i\*b1)+(a2+i\*b2)=(a1+a2)+i\*(b1+b2)).  Например:  q = (2 +i\*1), d = (2 +i\*1), q.Сложить(d) = (4 +i\*2). |
| Выход: | Комплексное число (тип TComplex). |
| Постусловия: | Нет. |
|  | |
| ***Умножить*** |  |
| Вход: | Комплексное число d (тип TComplex). |
| Предусловия: | Нет. |
| Процесс | Создаёт и возвращает комплексное число, полученное умножением самого числа this = a1+i\*b1 на число d = a2+i\*b2: ((a1+i\*b1)\*(a2+i\*b2)=(a1\*a2 -  b1\*b2)+i\*(a1\*b2+ a2\*b1)). |
| Выход: | Комплексное число (тип TComplex). |

|  |  |
| --- | --- |
| Постусловия: | Нет. |
|  | |
| ***Квадрат*** |  |
| Вход: | Нет. |
| Предусловия: | Нет. |
| Процесс | Создаёт и возвращает комплексное число (тип TComplex), полученное умножением числа this на самого себя: ((a1+i\*b1)\*(a1+i\*b1)=(a1\*a1 -  b1\*b1)+i\*(a1\*b1+ a1\*b1)). |
| Выход: | Комплексное число (тип TComplex). |
| Постусловия: | Нет. |
|  | |
| ***Обратное*** |  |
| Вход: | Нет. |
| Предусловия: | Нет. |
| Процесс | Создаёт и возвращает комплексное число (тип TComplex), полученное делением единицы на само число 1/((a1+i\*b1) = a1/(a1\*\*2 + b1\*\*2) - i\* b1/( a1\*\*2 +  b1\*\*2 )). |
| Выход: | Комплексное число (тип TComplex). |
| Постусловия: | Нет. |
|  | |
| ***Вычесть*** |  |
| Вход: | Комплексное число d (тип TComplex).. |
| Предусловия: | Нет. |
| Процесс | Создаёт и возвращает комплексное число (тип TComplex), полученное вычитанием d = a2 + i b2 из самого числа this = (a1+i\*b1): (a1+i\*b1)-(a2+i\*b2)=(a1-  a2)+i\*(b1-b2). |

|  |  |
| --- | --- |
|  | Например:  q = (2 +i\*1), d = (2 +i\*1)) q.Вычесть(d) = (0 + i0). |
| Выход: | Комплексное число (тип TComplex). |
| Постусловия: | Нет. |
|  | |
| ***Делить*** |  |
| Вход: | Комплексное число d. |
| Предусловия: | Нет. |
| Процесс | Создаёт и возвращает комплексное число (тип TComplex), полученное делением самого числа this на число (d) ((a1+i\*b1)/(a2+i\*b2)=(a1\*a2 + b1\*b2)/(a2\*\*2 +  b2\*\*2)+i\*(a2\*b1 – a1\*b2)/(a2\*\*2 + b2\*\*2)). |
| Выход: | Комплексное число (тип TComplex). |
| Постусловия: | Нет. |
|  | |
| ***Минус*** |  |
| Вход: | Нет. |
| Предусловия: | Нет. |
| Процесс | Создаёт и возвращает комплексное число (тип TComplex), являющееся разностью комплексных чисел z и и самого числа this, где z – комплексное число  (0+i0). |
| Выход: | Комплексное число (тип TComplex). |
| Постусловия: | Нет. |
|  | |
| ***Модуль*** |  |
| Вход: | Нет. |
| Предусловия: | Нет. |

|  |  |
| --- | --- |
| Процесс | Вычисляет и возвращает модуль самого комплексного числа this. Например:  q = (2 +i\*1), q. Модуль = (2\*2+1\*1). q = (i\*17), q. Модуль = (0\*0+17\*17). |
| Выход: | Вещественное число. |
| Постусловия: | Нет. |
|  | |
| ***УголРад*** |  |
| Вход: | Нет. |
| Предусловия: | Нет. |
| Процесс | Возвращает аргумент fi самого комплексного числа this (в радианах). fi = (arcTg(b/a), a>0; pi/2, a = 0, b > 0; arcTg(b/a) + pi, a < 0; -pi/2, a = 0, b <0 ).  Например:  q = (1 +i\*1), q. УголРад = 0,79. |
| Выход: | Вещественное число. |
| Постусловия: | Нет. |
|  | |
| ***УголГрад*** |  |
| Вход: | Нет. |
| Предусловия: | Нет. |
| Процесс | Возвращает аргумент самого комплексного числа this (в градусах).  Например:  q = (1 +i\*1), q. Град = 45. |
| Выход: | Вещественное число. |
| Постусловия: | Нет. |
|  | |

|  |  |
| --- | --- |
| ***Степень*** |  |
| Вход: | Целое n. |
| Предусловия: | Нет. |
| Процесс | Возвращает целую положительную степень n самого комплексного числа this. thisn = rn(cos (n\*fi)+ i\* sin  (n\*fi)). |
| Выход: | Комплексное число (тип TComplex). |
| Постусловия: | Нет. |
|  | |
| ***Корень*** |  |
| Вход: | Целое n, целое i. |
| Предусловия: | Нет. |
| Процесс | Возвращает i-ый корень целой положительной степени n самого комплексного числа this. n(this) = n(r)\* (cos ((fi + 2\*k\*pi)/n)+ i\* sin ((fi + 2\*k\*pi)/n)). При этом коофициенту k придается последовательно n значений: k = 0,1,2…, n - 1 и получают n значений корня, т.е.  ровно столько, каков показатель корня. |
| Выход: | Комплексное число (тип TComplex). |
| Постусловия: | Нет. |
|  | |
| ***Равно*** |  |
| Вход: | Комплексное число d. |
| Предусловия: | Нет. |
| Процесс | Сравнивает само комплексное число this с числом d. Возвращает значение True, если они - тождественные комплексные числа, и значение False - в противном  случае. |
| Выход: | Булевское значение. |

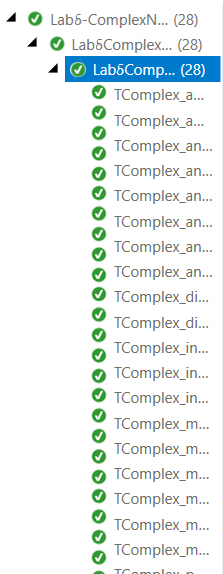
|  |  |
| --- | --- |
| Постусловия: | Нет. |
|  | |
| ***НеРавно*** |  |
| Вход: | Комплексное число d. |
| Предусловия: | Нет. |
| Процесс | Сравнивает само комплексное число this с числом d. Возвращает значение True, если само число <> d, -  значение False - в противном случае. |
| Выход: | Булевское значение. |
| Постусловия: | Нет. |
|  | |
| ***ВзятьReЧисло*** |  |
| Вход: | Нет |
| Предусловия: | Нет. |
| Процесс | Возвращает значение действительной части самого  комплексного числа this в числовом формате. |
| Выход: | Вещественное значение. |
| Постусловия: | Нет. |
|  | |
| ***ВзятьImЧисло*** |  |
| Вход: | Нет. |
| Предусловия: | Нет. |
| Процесс | Возвращает значение мнимой части самого  комплексного числа this в числовом формате. |
| Выход: | Вещественное значение. |
| Постусловия: | Нет. |
|  | |
| ***ВзятьReСтрока*** |  |
| Вход: | Нет. |

|  |  |
| --- | --- |
| Предусловия: | Нет. |
| Процесс | Возвращает значение вещественной части самого  комплексного числа this в строковом формате. |
| Выход: | Строка. |
| Постусловия: | Нет. |
|  | |
| ***ВзятьImСтрока*** |  |
| Вход: | Нет. |
| Предусловия: | Нет. |
| Процесс | Возвращает значение мнимой части самого  комплексного числа this в строковом формате. |
| Выход: | Строка. |
| Постусловия: | Нет. |
|  | |
| ***ВзятьКомплексноеСтрока*** |  |
| Вход: | Нет. |
| Предусловия: | Нет. |
| Процесс | Возвращает значение самого комплексного числа this в  строковом формате. |
| Выход: | Строка. |
| Постусловия: | Нет. |

# **Реализация**

В данной работе мы создали класс данных комплексных чисел и все необходимые для работы с ним операции, такие как: конструктор число, конструктор строка, копировать, сложить, умножить, квадрат, обратное, вычесть, делить, минус, модуль, нахождение аргумента в радианах, нахождение аргумента в градусах, найти степень, найти корень, проверки равенства, проверки неравенства, взять Re числа, взять Im числа, взять Re в качестве строки, взять Im в качестве строки, взять комплексное число как строку.

# **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. TComplex.h

#pragma once

#include <cmath>

#include <string>

#include <iostream>

#include <stdexcept>

using namespace std;

#define COMPL\_PI 3.14159265358979323846

class TComplex

{

private:

double re;

double im;

int is\_equal(double x) const noexcept;

public:

class Zero : exception

{

public:

explicit Zero(const char\* message) : exception(message)

{

}

};

explicit TComplex(double real, double image);

explicit TComplex(const string &f);

TComplex copy() const noexcept;

TComplex add(const TComplex &d) const noexcept;

TComplex mul(const TComplex &d) const noexcept;

TComplex square() const noexcept;

TComplex inverse() const;

TComplex sub(const TComplex &d) const noexcept;

TComplex div(const TComplex &d) const;

TComplex minus() const noexcept;

double module() const noexcept;

double angle\_rad() const;

double angle\_deg() const noexcept;

TComplex pow(int n) const noexcept;

TComplex root(int n, int i) const;

bool equals(const TComplex &d) const noexcept;

bool non\_equals(const TComplex &d) const noexcept;

double get\_re() const noexcept;

string get\_re\_as\_string() const noexcept;

double get\_im() const noexcept;

string get\_im\_as\_string() const noexcept;

string get\_complex\_as\_string() const noexcept;

};

## Листинг 2. TComplex.cpp

#include "pch.h"

#include "TComplex.h"

TComplex::TComplex(double real, double image)//конструктор

{

this->re = round(real \* 1000000) / 1000000;

this->im = round(image \* 1000000) / 1000000;

}

TComplex::TComplex(const string &f) //конструктор для строки

{

size\_t pos = f.find("+i\*");

int k = 1;

if (pos == string::npos)

{

pos = f.find("-i\*");

k = -1;

}

if (pos == string::npos)

{

this->re = round(stod(f) \* 1000000) / 1000000;

this->im = 0;

}

else

{

this->re = round(stod(f.substr(0, pos)) \* 1000000) / 1000000;

this->im = round(stod(f.substr(pos + 3)) \* k \* 1000000) / 1000000;

}

}

TComplex TComplex::copy() const noexcept//копирование

{

return TComplex(this->re, this->im);

}

TComplex TComplex::add(const TComplex& d) const noexcept//добавить число

{

return TComplex(this->re + d.re, this->im + d.im);

}

TComplex TComplex::mul(const TComplex &d) const noexcept//умножить

{

return TComplex(this->re \* d.re - this->im \* d.im, this->re \* d.im + this->im \* d.re);

}

TComplex TComplex::square() const noexcept//квадрат

{

return this->mul(\*this);

}

TComplex TComplex::inverse() const//инверсия числа

{

return TComplex(1, 0).div(\*this);

}

TComplex TComplex::sub(const TComplex &d) const noexcept//вычитание

{

return TComplex(this->re - d.re, this->im - d.im);

}

TComplex TComplex::div(const TComplex &d) const//деление числа

{

if (d.re == 0 && d.im == 0)

{

throw Zero("Division by zero");

}

return TComplex

(

((this->re \* d.re + this->im \* d.im) / (d.re \* d.re + d.im \* d.im)),

((this->im \* d.re - this->re \* d.im) / (d.re \* d.re + d.im \* d.im))

);

}

TComplex TComplex::minus() const noexcept//делаем число отрицательным

{

return TComplex(-this->re, -this->im);

}

double TComplex::module() const noexcept//модуль числа

{

return sqrt(this->re \* this->re + this->im \* this->im);

}

double TComplex::angle\_rad() const//угол в радианах

{

switch (is\_equal(this->re))

{

case 1:

return atan(this->im / this->re);

case 0:

switch (is\_equal(this->im))

{

case 1:

return COMPL\_PI / 2;

case 0:

throw Zero("Number is zero!");

case -1:

return -COMPL\_PI / 2;

}

case -1:

return atan(this->im / this->re) + COMPL\_PI;

}

}

double TComplex::angle\_deg() const noexcept//угол обычный

{

return this->angle\_rad() \* 180 / COMPL\_PI;

}

TComplex TComplex::pow(int n) const noexcept//возведение в степень

{

double mod = std::pow(this->module(), n);

double arg = this->angle\_rad() \* n;

return TComplex(mod \* cos(arg), mod \* sin(arg));

}

TComplex TComplex::root(int n, int i) const

{

if (n == 0)

{

throw Zero("Division by zero");

}

if (i >= n)

{

throw Zero("Index must be less than exponent");

}

double mod = std::pow(this->module(), 1.0 / n);

double arg = (this->angle\_rad() + 2 \* COMPL\_PI \* i) / n;

return TComplex(mod \* cos(arg), mod \* sin(arg));

}

bool TComplex::equals(const TComplex &d) const noexcept//проверка равенства

{

if (this->re == d.re && this->im == d.im)

{

return true;

}

else

{

return false;

}

}

bool TComplex::non\_equals(const TComplex &d) const noexcept//проверка неравенства

{

return !(this->equals(d));

}

double TComplex::get\_re() const noexcept//получить re часть

{

return this->re;

}

string TComplex::get\_re\_as\_string() const noexcept//получить re часть как строку

{

return to\_string(this->re);

}

double TComplex::get\_im() const noexcept

{

return this->im;

}

string TComplex::get\_im\_as\_string() const noexcept//получить im часть как строку

{

return to\_string(this->im);

}

string TComplex::get\_complex\_as\_string() const noexcept// получить комплексное число как строку

{

string re = get\_re\_as\_string();

string im;

string sign;

switch (is\_equal(this->im))

{

case -1:

sign = "-i\*";

im = std::to\_string(fabs(this->im));

break;

case 0:

sign = im = "";

break;

case 1:

sign = "+i\*";

im = get\_im\_as\_string();

break;

}

return re + sign + im;

}

int TComplex::is\_equal(double x) const noexcept// обработка исключений

{

if (x > 0)

{

return 1;

}

else if (x == 0)

{

return 0;

}

return -1;

}