**Московский авиационный институт**

**(Национальный исследовательский университет)**

Факультет: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Объектно-ориентированное программирование»

**Лабораторная работа № 2**

Тема: Перегрузка операторов в С++

Студент: Трофимов Максим

Группа: 80-201

Преподаватель: Чернышов Л.Н.

Дата:

Оценка:

Москва, 2019

1. Постановка задачи

. Реализовать класс комплексных чисел в тригонометрической форме , где - радиус (модуль), а -угол. Реализовать также функции сложения, вычитания, умножения, деления, проверки на равенство и нахождения сопряжённого числа, а также пользовательский литерал. Операции реализовать через перегрузку операторов.

1. Описание программы

Программа выводит меню команд для работы с двумя комплексными числами. По умолчанию эти два числа и , которые создаются пользовательским литералом. Радиус может быть любым неотрицательным действительным числом. В случае, если происходит попытка изменить модуль на отрицательное число, то всё число становится и выводится ошибка. Если модуль нулевой, то и угол равен нулю. Угол приводится к диапазону . Если происходит деление на ноль, то выводится ошибка (но возвращается значение равное ).

ссылка на github: https://github.com/student31415/oop\_exercise\_02

1. Набор testcases

|  |  |
| --- | --- |
| Тест | Примичание |
| 2  5  6  7  8  9  10  11  12  13  0 | простейшая демонстрация сравнения двух чисел, их сложение, вычитание одного из другого и наоборот, их умножение, деление одного на другого и наоборот и нахождение их сопряжённых. |
| 2  4  -1  2  3  3.1415  2  2  11  10  9  0 | Попытка изменить модуль второго числа на отрицательное число (ошибка), изменение первого, деление второго на первое (производится без ошибок), деление первого на второе (ошибка), их умножение. |

1. Результаты выполнения тестов.

test\_01.txt:

1. Печать меню.

2. Печать двух комплексных чисел.

3. Изменение первого комплексного числа.

4. Изменение второго комплексного числа.

5. Проверка их на равенство.

6. Сумма двух комплексных чисел.

7. Вычитание второого из первого.

8. Вычитание первого из второго.

9. Произведение двух комплексных чисел.

10. Деление первого комплексного числа на второе.

11. Деление второго комплексного числа на первое.

12. Сопряжённое первого числа.

13. Сопряжённое второго числа.

0.Выход.

ПрИгЛаШеНиЕ>>5

Числа различны.

ПрИгЛаШеНиЕ>>6

( 1 , 0 ) + ( 1 , 1.5708 ) = (1.41421 , 0.785398 )

ПрИгЛаШеНиЕ>>7

( 1 , 0 ) - ( 1 , 1.5708 ) = (1.41421 , 5.49779 )

ПрИгЛаШеНиЕ>>8

( 1 , 1.5708 ) - ( 1 , 0 ) = (1.41421 , 2.35619 )

ПрИгЛаШеНиЕ>>9

( 1 , 0 ) \* ( 1 , 1.5708 ) = (1 , 1.5708 )

ПрИгЛаШеНиЕ>>10

( 1 , 0 ) / ( 1 , 1.5708 ) = (1 , 4.71239 )

ПрИгЛаШеНиЕ>>11

( 1 , 1.5708 ) / ( 1 , 0 ) = (1 , 1.5708 )

ПрИгЛаШеНиЕ>>12

( 1 , 0 )

ПрИгЛаШеНиЕ>>13

( 1 , 4.71239 )

ПрИгЛаШеНиЕ>>0

test\_02.txt:

1. Печать меню.

2. Печать двух комплексных чисел.

3. Изменение первого комплексного числа.

4. Изменение второго комплексного числа.

5. Проверка их на равенство.

6. Сумма двух комплексных чисел.

7. Вычитание второого из первого.

8. Вычитание первого из второго.

9. Произведение двух комплексных чисел.

10. Деление первого комплексного числа на второе.

11. Деление второго комплексного числа на первое.

12. Сопряжённое первого числа.

13. Сопряжённое второго числа.

0.Выход.

ПрИгЛаШеНиЕ>>2

Первое=( 1 , 0 )

Второе=( 1 , 1.5708 )

ПрИгЛаШеНиЕ>>4

Введите модуль комплексного числа.

-1

Модлуь числа меньше нуля.

ПрИгЛаШеНиЕ>>2

Первое=( 1 , 0 )

Второе=( 0 , 0 )

ПрИгЛаШеНиЕ>>3

Введите модуль комплексного числа.

3.1415

Введите угол комлексного числа.

2

ПрИгЛаШеНиЕ>>2

Первое=( 3.1415 , 2 )

Второе=( 0 , 0 )

ПрИгЛаШеНиЕ>>11

( 0 , 0 ) / ( 3.1415 , 2 ) = (0 , 0 )

ПрИгЛаШеНиЕ>>10

Деление на ноль. Деление не произведенно.

ПрИгЛаШеНиЕ>>9

( 3.1415 , 2 ) \* ( 0 , 0 ) = (0 , 0 )

ПрИгЛаШеНиЕ>>0

1. Листинг программы

## **complex\_trig.cpp:**

#include<iostream>

#include"complex\_trig.h"

#include<cmath>

Complex\_Trig::Complex\_Trig(double R, double Fi){

if(R>=0) \_r=R; else std::cerr<<"Модлуь числа меньше нуля."<<std::endl;

while(Fi>=2\*M\_PI) Fi-=2\*M\_PI;

while(Fi<0) Fi+=2\*M\_PI;

if(R==0)\_fi=0; else \_fi=Fi;

}

Complex\_Trig::Complex\_Trig(){

\_r=0;

\_fi=0;

}

Complex\_Trig operator + (const Complex\_Trig& a, const Complex\_Trig& b){

Complex\_Trig c;

c.\_r=(double) sqrt((a.\_r\*cos(a.\_fi)+b.\_r\*cos(b.\_fi))\*(a.\_r\*cos(a.\_fi)+b.\_r\*cos(b.\_fi)) + (a.\_r\*sin(a.\_fi)+b.\_r\*sin(b.\_fi))\*(a.\_r\*sin(a.\_fi)+b.\_r\*sin(b.\_fi)));

if(c.\_r>0){

c.\_fi=acos((a.\_r\*cos(a.\_fi)+b.\_r\*cos(b.\_fi))/c.\_r);

if(a.\_r\*sin(a.\_fi)+b.\_r\*sin(b.\_fi)<0) c.\_fi=2\*M\_PI-c.\_fi;

}else{ c.\_r=0; c.\_fi=0; }

return c;

}

Complex\_Trig operator - (const Complex\_Trig& a, const Complex\_Trig& b){

Complex\_Trig c;

c.\_r=(double) sqrt((a.\_r\*cos(a.\_fi)-b.\_r\*cos(b.\_fi))\*(a.\_r\*cos(a.\_fi)-b.\_r\*cos(b.\_fi)) + (a.\_r\*sin(a.\_fi)-b.\_r\*sin(b.\_fi))\*(a.\_r\*sin(a.\_fi)-b.\_r\*sin(b.\_fi)));

if(c.\_r>0){

c.\_fi=acos((a.\_r\*cos(a.\_fi)-b.\_r\*cos(b.\_fi))/c.\_r);

if(a.\_r\*sin(a.\_fi)-b.\_r\*sin(b.\_fi)<0) c.\_fi=2\*M\_PI-c.\_fi;

}else{c.\_r=0; c.\_fi=0;}

return c;

}

Complex\_Trig operator \* (const Complex\_Trig& a, const Complex\_Trig& b){

Complex\_Trig c;

c.\_r=a.\_r\*b.\_r;

if(a.\_fi+b.\_fi>2\*M\_PI){

c.\_fi=a.\_fi+b.\_fi-2\*M\_PI;

}else{

c.\_fi=a.\_fi+b.\_fi;

}

if(c.\_r==0) c.\_fi=0;

return c;

}

Complex\_Trig operator / (const Complex\_Trig & a, const Complex\_Trig& b){

Complex\_Trig c(0, 0);

if(b.\_r==0){

std::cout<<"Деление на ноль. Деление не произведенно."<<std::endl;

return c;

}

c.\_r=a.\_r/b.\_r;

if(a.\_fi-b.\_fi>=0) c.\_fi=a.\_fi-b.\_fi; else c.\_fi=2\*M\_PI+(a.\_fi-b.\_fi);

if(c.\_r==0) c.\_fi=0;

return c;

}

double Complex\_Trig::radius(){

return \_r;

}

double Complex\_Trig::angle(){

return \_fi;

}

void Complex\_Trig::ch\_radius(double R){

if(R>0) \_r=R; else{

if(R<0) std::cerr<<"Модлуь числа меньше нуля."<<std::endl;

\_r=0; \_fi=0;

}

return;

}

void Complex\_Trig::ch\_angle(double Fi){

if(\_r==0) return;

while(Fi>=2\*M\_PI) Fi-=2\*M\_PI;

while(Fi<0) Fi+=2\*M\_PI;

\_fi=Fi;

return;

}

bool operator == (const Complex\_Trig& a, const Complex\_Trig& b){

return a.\_fi==b.\_fi && a.\_r==b.\_r;

}

std::ostream& operator << (std::ostream& stream, const Complex\_Trig& a){

stream << "( "<< a.\_r<<" , "<<a.\_fi<<" )";

return stream;

}

Complex\_Trig Complex\_Trig::conj(){

Complex\_Trig a;

a.\_r=\_r;

if(\_fi!=0) a.\_fi=2\*M\_PI-\_fi; else a.\_fi=0;

return a;

}

## **complex\_trig.h:**

#ifndef \_COMPLEX\_TRIG\_

#define \_COMPLEX\_TRIG\_

class Complex\_Trig{

public:

Complex\_Trig(double R, double Fi); // создать комплексное число на основе радиуса и углa от 0 до 2\*пи

Complex\_Trig();

friend Complex\_Trig operator + (const Complex\_Trig& a, const Complex\_Trig& b);

friend Complex\_Trig operator - (const Complex\_Trig& a, const Complex\_Trig& b);

friend Complex\_Trig operator \* (const Complex\_Trig& a, const Complex\_Trig& b);

friend Complex\_Trig operator / (const Complex\_Trig& a, const Complex\_Trig& b);

friend bool operator == (const Complex\_Trig& a, const Complex\_Trig& b); //пара поэлементно равна

Complex\_Trig conj(); //conj(r, fi)= (r, -fi)

//friend Complex\_Trig operator "" \_tocmplx(const char\* str);

friend std::ostream &operator << (std::ostream& stream, const Complex\_Trig& a);

double radius();

double angle();

void ch\_radius(double R);

void ch\_angle(double Fi);

private:

double \_r; //radius

double \_fi; //angle

};

#endif

## **main\_menu.cpp:**

#include<iostream>

#include<cmath>

#include"complex\_trig.h"

#include<cstdlib>

/\*

Трофимво Максим М8О-201б-18

---------------------------

вариант номер 2:

создать класс для хранения и работы комплексных чисел в комплексной форме.

Реализовать сложениие, вычитание, умножение, деление, проверку на равенство и нахождение споряжённого.

\*/

Complex\_Trig operator ""\_tocmplx(const char\* str, size\_t size){

char\* buffer = new char[15];

int key=0;

int j=0;

double r, fi;

//std::string str(str\_1);

for(size\_t i=0; i<size; ++i){

if(((str[i]>='0' && str[i]<='9') || str[i]=='-'|| str[i]=='.') && j<14)

buffer[j++]=str[i];

if(str[i]==',' ){

key=1;

buffer[j]='\0';

j=0;

r=std::atof(buffer);

for(int k=0; k<15; ++k) buffer[k]='\0';

}

}

fi= std::atof(buffer);

if(r<0) r=0;

while(fi>=2\*M\_PI) fi-=2\*M\_PI;

while(fi<0) fi+=2\*M\_PI;

if(r==0) fi=0;

delete [] buffer;

return Complex\_Trig(r, fi);

}

void print\_menu(){

std::cout<<"1. Печать меню.\n"

<<"2. Печать двух комплексных чисел.\n"

<<"3. Изменение первого комплексного числа.\n"

<<"4. Изменение второго комплексного числа.\n"

<<"5. Проверка их на равенство.\n"

<<"6. Сумма двух комплексных чисел.\n"

<<"7. Вычитание второого из первого.\n"

<<"8. Вычитание первого из второго.\n"

<<"9. Произведение двух комплексных чисел.\n"

<<"10. Деление первого комплексного числа на второе.\n"

<<"11. Деление второго комплексного числа на первое.\n"

<<"12. Сопряжённое первого числа.\n"

<<"13. Сопряжённое второго числа.\n"

<<"0.Выход.\n";

return;

}

int main(){

Complex\_Trig lhs="1, 0"\_tocmplx, rhs="1,1.5708 "\_tocmplx, ans= "0,0"\_tocmplx;

int status=1;

std::cout<<"Программа для работы с комплексными числами в тригонометрической форме"<<std::endl;

std::cout<<"Первое= "<<lhs<<"\n";

std::cout<<"Второе= "<<rhs<<std::endl;

while(1){

switch(status){

case 0 :

return 0;

break;

case 1 :

print\_menu();

break;

case 2 :

std::cout<<"Первое= "<<lhs<<"\n";

std::cout<<"Второе= "<<rhs<<"\n";

break;

case 3 :{

double r, fi;

std::cout<<"Введите модуль комплексного числа."<<std::endl;

std::cin>>r;

lhs.ch\_radius(r);

if(r<0) break;

std::cout<<"Введите угол комлексного числа."<< std::endl;

std::cin>>fi;

lhs.ch\_angle(fi);

break;

}

case 4 :{

double r, fi;

std::cout<<"Введите модуль комплексного числа."<<std::endl;

std::cin>>r;

rhs.ch\_radius(r);

if(r<0) break;

std::cout<<"Введите угол комлексного числа."<< std::endl;

std::cin>>fi;

rhs.ch\_angle(fi);

break;

}

case 5 :

if(lhs == rhs){

std::cout<<"Числа равны."<<std::endl;

}else{

std::cout<<"Числа различны."<<std::endl;

}

break;

case 6 :

ans= lhs + rhs;

std::cout<<lhs<<" + "<<rhs<<" = "<<ans<<std::endl;

break;

case 7 :

ans=lhs - rhs;

std::cout<<lhs<<" - "<<rhs<<" = "<<ans<<std::endl;

break;

case 8 :

ans= rhs - lhs;

std::cout<<lhs<<" - "<<rhs<<" = "<<ans<<std::endl;

break;

case 9 :

ans=lhs \* rhs;

std::cout<<lhs<<" \* "<<rhs<<" = "<<ans<<std::endl;

break;

case 10 :

ans=lhs / rhs;

if(rhs.radius()!=0) std::cout<<lhs<<" / "<<rhs<<" = "<<ans<<std::endl;

break;

case 11 :

ans=rhs / lhs;

if(lhs.radius()!=0) std::cout<<lhs<<" / "<<rhs<<" = "<<ans<<std::endl;

break;

case 12 :

ans=lhs.conj();

std::cout<<ans<<std::endl;

break;

case 13 :

ans=rhs.conj();

std::cout<<ans<<std::endl;

break;

default:

std::cout<<"Неверно введено действие."<<std::endl;

}

std::cout<<"ПрИгЛаШеНиЕ>>";

std::cin>>status;

}

}

## 

1. Вывод  
   Научился реализовывать класс с помощью перегрузки операторов и дружественных функций. Так же добавил возможность создания константных выражений с помощью литералов. Корректность программы проверил на двух тестах.

**Список литературы**

1. Шилдт, Герберт. С++: базовый курс, 3-е изд. : Пер. с англ. - М. : ООО “И.Д. Вильямс”, 2018. - 624 с. : ил. - Парал. тит. англ.

2. Справочник по языку С++ [Электронный ресурс]. URL:<http://www.cplusplus.com/reference/deque/> (дата обращения: 17.09.2019).