**Московский авиационный институт**

**(Национальный исследовательский университет)**

Институт: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Объектно-ориентированное программирование»

**Лабораторная работа № 1**

Тема: Простые классы

Студент: Бутырев Даниил Вячеславович

Группа: 80-206

Преподаватель: Чернышов Л.Н.

Дата:

Оценка:

Москва, 2021

<https://github.com/SteepturN/oop_exercise_01>

1. Постановка задачи:

Комплексное число в алгебраической форме представляются парой действительных чисел (a, b), где a – действительная часть, b – мнимая часть. Реализовать класс Complex для работы с комплексными числами. Обязательно должны быть присутствовать операции

- сложения add, (a, b) + (c, d) = (a + c, b + d);

- вычитания sub, (a, b) – (c, d) = (a – c, b – d);

- умножения mul, (a, b) ´ (c, d) = (ac – bd, ad + bc);

- деления div, (a, b) / (c, d) = (ac + bd, bc – ad) / (c2 + d2);

- сравнение equ, (a, b) = (c, d), если (a = c) и (b = d);

- сопряженное число conj, conj(a, b) = (a, –b).

Реализовать операции сравнения модулей.

1. Описание программы

Программа состоит из 3 файлов:

1. main.cpp - здесь находится пользовательский интерпретатор, происходят манипуляции с объектами класса Complex и их результаты отправляются на стандартный поток вывода,
2. Read\_input.cpp - здесь происходит ввод данных,
3. Complex.cpp - здесь описан необходимый класс и его методы.

3. Набор тестов

a 1 2 3 4

a 1 3 -3 5

s 0 9 1 2

s 1.123 -4.341 5.134 3.4355

m 1 2 3 4

m 1.234 5.123 -3.1234 5.2134

d 1 2 3 4

d -4 1.123 3.4 3.4

e 5 6 7 6

e -5.12 4.4 -5.12 4.4

bth 123 123 5 6

bth 43 43 43 43

exit

a 1.235 5.234 1.134 45.543

a 1 343 -35 5

s 0 9135 11.135 2

s 1.123 341 -5.134 -53.4355

m 1 2 3 -46

m -1.234 5.123 -3.1234 5.2134

d -1 -2 -3 -4

d -4 1.123 -3.4 3.4

e 5 6 7 -6

e -5.12 -4.4 -5.12 4.4

bth -123 -123 -5 6

bth 43 -43 -43 43

exit

4. Результаты выполнения тестов

(1, 2) a (3, 4) = (4, 6)

(1, 3) a (-3, 5) = (-2, 8)

(0, 9) s (1, 2) = (-1, 7)

(1.123, -4.341) s (5.134, 3.4355) = (-4.011, -7.7765)

(1, 2) m (3, 4) = (-5, 10)

(1.234, 5.123) m (-3.1234, 5.2134)=(-30.5625, -9.56784)

(1, 2) d (3, 4) = (0.44, 0.08)

(-4, 1.123) d (3.4, 3.4) = (-0.423088, 0.753382)

(5, 6) e (7, 6) = 0

(-5.12, 4.4) e (-5.12, 4.4) = 1

(123, 123) bth (5, 6) = 1

(43, 43) bth (43, 43) = 0

(1.235, 5.234) a (1.134, 45.543) = (2.369, 50.777)

(1, 343) a (-35, 5) = (-34, 348)

(0, 9135) s (11.135, 2) = (-11.135, 9133)

(1.123, 341) s (-5.134, -53.4355) = (6.257, 394.435)

(1, 2) m (3, -46) = (95, -40)

(-1.234, 5.123) m (-3.1234, 5.2134) = (-22.854, -22.4345)

(-1, -2) d (-3, -4) = (0.44, 0.08)

(-4, 1.123) d (-3.4, 3.4) = (0.753382, 0.423088)

(5, 6) e (7, -6) = 0

(-5.12, -4.4) e (-5.12, 4.4) = 0

(-123, -123) bth (-5, 6) = 1

(43, -43) bth (-43, 43) = 0

5. Листинг программы

main.cpp:

#include <iostream>

#include <string>

#include "Complex.cpp"

#include <cstdio>

#include <set>

#include "Read\_input.cpp"

int main(){

char a[] = "You can use\n\

--add:\t a (real part of num1) (imagin part of num1) (real part of num2) (imagin part of num2)\n\

--subtraction:\t s (real part of num1) (imagin part of num1) (real part of num2) (imagin part of num2)\n\

--conjuction:\t c (real part of num) (imagin part of num)\n\

--multiply:\t m (real part of num1) (imagin part of num1) (real part of num2) (imagin part of num2)\n\

--division:\t d (real part of num1) (imagin part of num1) (real part of num2) (imagin part of num2)\n\

--equivalent:\t e (real part of num1) (imagin part of num1) (real part of num2) (imagin part of num2)\n\

--if num1 bigger than num2:\t bth (real part of num1) (imagin part of num1) (real part of num2) (imagin part of num2)\n\

--exit\n";

Complex c1(0,0), c2(0,0);

char ch(' ');

char command[20] = {0};

std::set<std::string> valid\_commands = {"c", "a", "s", "m", "d", "e", "bth", "exit"};

std::cout << a;

do {

bool exit = false;

do{

read\_return\_t answer = get\_command(valid\_commands, command);

switch(answer) {

case END\_OF\_FILE: return 0;

case END\_OF\_LINE: continue;

case VALID\_INPUT: exit = true; break;

case INVALID\_INPUT:

do ch=getchar(); while((ch != EOF) && (ch != '\n'));

std::cout << "wrong input" << std::endl;

if(ch == EOF) return 0;

else break;

}

} while(!exit);

if(static\_cast<std::string>(command) == "exit") return 0;

if((get\_value<double>(c1.a) != VALID\_INPUT) || (get\_value<double>(c1.b) != VALID\_INPUT)) {

do ch = getchar(); while((ch != EOF) && (ch != '\n'));

std::cout << "wrong input" << std::endl;

if(ch == EOF) return 0;

continue;

}

if(static\_cast<std::string>(command) != "c") {

if((get\_value<double>(c2.a) != VALID\_INPUT) || (get\_value<double>(c2.b) != VALID\_INPUT)) {

do ch=getchar(); while((ch != EOF) && (ch != '\n'));

std::cout << "wrong input" << std::endl;

if(ch == EOF) return 0;

continue;

}

}

c1.print();

std::cout << '\t' << command << '\t';

if(static\_cast<std::string>(command) != "c") {

c2.print();

std::cout << '\t';

}

std::cout << '\t' << '=' << '\t';

if(static\_cast<std::string>(command) == "c") {

c1.conj().print();

} else if (static\_cast<std::string>(command) == "bth") {

std::cout << c1.bth(c2) << std::endl;

} else if(static\_cast<std::string>(command) == "e") {

std::cout << c1.equ(c2) << std::endl;

} else {

c1.exec(command, c2).print();

}

std::cout << std::endl;

} while(true);

return 0;

}

Complex.cpp:

//var 1

#include <iostream>

#include <cstring>

class Complex{

public:

Complex(double a, double b)

: a(a), b(b) {};

double a;

double b;

Complex add(const Complex& c){

return Complex(a+c.a, b+c.b);

}

Complex sub(const Complex& c){

return Complex(a-c.a, b-c.b);

}

Complex mul(const Complex& c){

return Complex(a\*c.a-b\*c.b, a\*c.b+b\*c.a);

}

Complex div(const Complex& c){

return Complex((a\*c.a+b\*c.b)/(c.a\*c.a+c.b\*c.b), (b\*c.a-a\*c.b)/(c.a\*c.a+c.b\*c.b));

}

bool equ(const Complex& c){

return (a==c.a) && (b==c.b);

}

Complex conj(){

return Complex(a, -b);

}

bool bth(const Complex& c){

return (a\*a+b\*b) > (c.a\*c.a+c.b\*c.b);

}

void print(){

std::cout << '(' << a << ", " << b << ')';

}

Complex exec(const char\* command, Complex c) {

Complex answ(0, 0);

if(strcmp(command,"a")==0){

answ = this->add(Complex(c.a, c.b));

} else if(strcmp(command,"s")==0){

answ = this->sub(Complex(c.a, c.b));

} else if(strcmp(command,"m")==0){

answ = this->mul(Complex(c.a, c.b));

} else if(strcmp(command,"d")==0){

answ = this->div(Complex(c.a, c.b));

} else if(strcmp(command,"c")==0){

answ = this->conj();

}

return answ;

}

};

//var 1

#include <iostream>

#include <cstring>

class Complex{

public:

Complex(double a, double b)

: a(a), b(b) {};

double a;

double b;

Complex add(const Complex& c){

return Complex(a+c.a, b+c.b);

}

Complex sub(const Complex& c){

return Complex(a-c.a, b-c.b);

}

Complex mul(const Complex& c){

return Complex(a\*c.a-b\*c.b, a\*c.b+b\*c.a);

}

Complex div(const Complex& c){

return Complex((a\*c.a+b\*c.b)/(c.a\*c.a+c.b\*c.b), (b\*c.a-a\*c.b)/(c.a\*c.a+c.b\*c.b));

}

bool equ(const Complex& c){

return (a==c.a) && (b==c.b);

}

Complex conj(){

return Complex(a, -b);

}

bool bth(const Complex& c){

return (a\*a+b\*b) > (c.a\*c.a+c.b\*c.b);

}

void print(){

std::cout << '(' << a << ", " << b << ')';

}

Complex exec(const char\* command, Complex c) {

Complex answ(0, 0);

if(strcmp(command,"a")==0){

answ = this->add(Complex(c.a, c.b));

} else if(strcmp(command,"s")==0){

answ = this->sub(Complex(c.a, c.b));

} else if(strcmp(command,"m")==0){

answ = this->mul(Complex(c.a, c.b));

} else if(strcmp(command,"d")==0){

answ = this->div(Complex(c.a, c.b));

} else if(strcmp(command,"c")==0){

answ = this->conj();

}

return answ;

}

};

Read\_input.cpp:

#include <iostream>

#include <string>

#include <cstdio>

#include <set>

typedef enum {

END\_OF\_FILE,

VALID\_INPUT,

INVALID\_INPUT,

END\_OF\_LINE,

} read\_return\_t;

read\_return\_t get\_command(std::set<std::string>& valid\_commands, char\* command)

{

char ch(' ');

while((ch == '\t') || (ch == ' ')) {

std::cin >> ch;

if(std::cin.eof()) return END\_OF\_FILE;

if(ch == '\n') return END\_OF\_LINE;

}

std::cin.unget();

std::cin >> command;

if(std::cin.fail()) {

std::cin.clear();

return INVALID\_INPUT;

}

if(valid\_commands.count(command)) return VALID\_INPUT;

else return INVALID\_INPUT;

}

template <typename T>

read\_return\_t get\_value(T& d)

{

char ch(' ');

while((ch == '\t') || (ch == ' ')) {

std::cin >> ch;

if(std::cin.eof()) return END\_OF\_FILE;

if(ch == '\n') return END\_OF\_LINE;

}

std::cin.unget();

std::cin >> d;

if(std::cin.fail()) {

std::cin.clear();

return INVALID\_INPUT;

}

return VALID\_INPUT;

}

6. Вывод

Классы в C++ являются хорошим средством для организации программы и для реализации ООП.

ЛИТЕРАТУРА

1. Б. Страуструп "Язык программирования С++", четвёртое издание, 2013.