Отчёт по лабораторной работе №_____01____ по курсу ___2

студента группы _____М80-208Б-18_____, № по списку _2

Адреса www, e-mail, jabber, skype _____alek.maria@yandex.ru

Работа выполнена: “29“ _____сентября_____ 2019г.

1. **Тема**: _____ Простые классы _____

2. **Цель работы**: ___Изучение системы сборки на языке C++, изучение систем контроля версии. Изучение основ работы с классами в C++._____

3. **Задание** (*вариант № 2*):

Комплексное число в тригонометрической форме представляются парой действительных чисел (r, φ), где r – радиус (модуль), φ – угол. Реализовать класс `Complex` для работы с комплексными числами. Обязательно должны быть присутствовать операции

- сложения `add`, (r_1, φ_1) + (r_2, φ_2);
- вычитания `sub`, (r_1, φ_1) - (r_2, φ_2);
- умножения `mul`, (r_1, φ_1) * (r_2, φ_2);
- деления `div`, (r_1, φ_1) / (r_2, φ_2);
- операции сравнения `equ`, (r_1, φ_1) = (r_2, φ_2), если ($r_1 = r_2$) и ($\varphi_1 = \varphi_2$);
- сопряженное число `conj`, `conj`(r, φ) = (r, $-\varphi$).

4. **Адрес репозитория на GitHub** https://github.com/PowerMasha/oop_exercise_01

5. **Код программы на C++**

main.cpp

```
#include <iostream>
#include <cmath>
#include "complex.h"

int main(){
      complex m1;
      complex m2;
printf("Введите первое комплексное число\n");
   m1.read(std::cin);
printf("Введите второе комплексное число\n");
   m2.read(std::cin);
printf("Первое комплексное число, модуль длины (r)  угол (u)\n");
  for(int i = 0; i < 2; ++i){
    std::cout << m1.get(i)<<' ';
    }
std::cout << '\n';
printf("Второе комплексное число, модуль длины (r)  угол (u)\n");
  for(int i = 0; i < 2; ++i){
    std::cout << m2.get(i)<<' ';
    }
std::cout << '\n';
std::cout <<'\n';

std::cout << "Длина и угол вектора суммы:\n";
complex sum = m1.add(m2);
for(int i = 0; i < 2; ++i){
 std::cout<<sum.get(i) << ' ';
```

```
  }
std::cout <<'\n';
std::cout << "Длина и угол вектора разности:\n";
complex sub = m1.sub(m2);
for(int i = 0; i < 2; ++i){

  std::cout <<sub.get(i) << ' ';
              }
std::cout <<'\n';

  std::cout << "Произведение: \n";
   m1.multiply(m2);
std::cout << '\n';

std::cout << "Деление :\n";
       m1.div(m2);
std::cout << '\n';

std::cout << "Сравнение комплексных чисел по длине вектора и углу:\n";
       m1.equ(m2);
std::cout << '\n';

std::cout << "sopr_m1:\n";
complex sm1=m1.sopr();
  for(int i = 0; i < 2; ++i){
 std::cout<<sm1.get(i)<<' ';
    }

std::cout << '\n';
std::cout << "sopr_m2:\n";
complex sm2=m2.sopr();
  for(int i = 0; i < 2; ++i){
      std::cout<<sm2.get(i)<<' ';
    }
std::cout << '\n';
}
```

complex.h

```
#ifndef D_COMPLEX_H
#define D_COMPLEX_H
#include <iostream>
#include <cmath>

struct complex {
double a;
double b;
complex(double a,double b);
complex ();
  complex add(const complex& rhs) const;
  complex multiply(const complex& rhs) const;
  complex sub(const complex& rhs) const;
 complex div(const complex& rhs) const;
 void equ(const complex& rhs) const;
complex sopr();

  void read(std::istream& is);
  void write(std::ostream& os) const;

double get(int i);
double cosi()const;
```

```
double sini()const;
void set(double x,double y);

private:
  double arr[2];
};
#endif // D_COMPLEX_H
```

complex.cpp

```
#include <cmath>
#include <iostream>
#include "complex.h"

complex::complex(): arr{0,0} {}

complex::complex(double a,double b): arr{a, b} {}
double PI=3.1415926535;
double complex::get(int i) {
  return arr[i];
}
void complex::set(double x,double y) {arr[0]=x;arr[1]=y;}
void complex::read(std::istream& is){
for (int i=0; i<2; ++i){
      is >> arr[i];}
}
double complex::cosi()const{
double k;
if (arr[1]==90 || arr[1]==270){ k=0;} else{ k=arr[0]*cos(arr[1]*PI/180);}
return k;
}
double complex::sini()const{
double s;
if (arr[1]==0 || arr[1]==180) {s=0;}else {s=arr[0]*sin(arr[1]*PI/180);}
return s;
}
complex complex::add(const complex& rhs) const{
complex sum{0,0};
double x1 = this->cosi();
double y1 = this->sini();
double x2 = rhs.cosi();
double y2 = rhs.cosi();
double x=x1+x2;
double y=y1+y2;

      sum.arr[0]=std::sqrt(x*x+y*y);
      sum.arr[1]=atan2(y,x);

return sum;
}

complex complex::sub(const complex& rhs) const{
complex raznost{0,0};
double x1 = this->cosi();
double y1 = this->sini();
double x2 = rhs.cosi();
double y2 = rhs.cosi();
double x=x1-x2;
double y=y1-y2;

       raznost.arr[0]=std::sqrt(x*x+y*y);
```

```
	raznost.arr[1]=atan2(y,x);

return raznost;
}
complex complex::multiply(const complex& rhs) const {
  complex result{0,0};
	result.arr[0] = arr[0]*rhs.arr[0];
	result.arr[1] = arr[1]+ rhs.arr[1];
std::cout << result.arr[0]<<"*(cos("<<result.arr[1]<<")+i*sin("<<result.arr[1]<<"))";
return result;
}

complex complex::div(const complex& rhs) const {
  complex result{0,0};
 if (rhs.arr[0]!=0) {result.arr[0] =(arr[0])/rhs.arr[0];}
	{result.arr[1] = arr[1] - rhs.arr[1];}
std::cout << result.arr[0]<<"*(cos("<<result.arr[1]<<")+i*sin("<<result.arr[1]<<"))";
return result;
}
void complex::equ(const complex& rhs) const {
int k=0;
int l=0;
if (arr[0]==rhs.arr[0]){k=1;} else {k=0;}
if (arr[1]==rhs.arr[1]){l=1;} else {l=0;}
if(k==1){std::cout << "Длины равны\n";} else {std::cout << "Длины не равны\n";}
if(l==1){std::cout << "Углы равны\n";} else {std::cout << "Углы не равны\n";}
}

complex complex::sopr(){
complex sop{0,0};
sop.arr[0]=arr[0];
sop.arr[1]=-arr[1];
return sop;
}
```

CMakeLists.txt

```
project(1lab)

add_executable(oop_exercise_01
	main.cpp
	complex.cpp)

set(CMAKE_CXX_FLAGS
	"${CMAKE_CXX_FLAGS} -Wall -Wextra")
```

6. Набор testcases

test_01.txt	Ожидаемое действие	Ожидаемый результат
1 30 3.4 45	add((1, 30) ,(3.4, 45))	4.37 0.72
	sub((1, 30) ,(3.4, 45))	2.45 -2.25
	multiply((1, 30) ,(3.4, 45))	3.4*(cos(75)+i*sin(75))

	div((1, 30) ,(3.4, 45))	0,29*(cos(-15)+i*sin(-15))
	sravn((1, 30) ,(3.4, 45))	Длины не равны Углы не равны
	Сопряженные числа	(1, -30) (3.4, -45)
test_02.txt	Ожидаемое действие	Ожидаемый результат
2 90 1 30	add((2,90),(1,30))	2.99 1. 27
	sub((2,90),(1,30))	1.43 2.22
	multiply ((2,90),(1,30))	2*(cos(120)+i*sin(120))
	div((2,90),(1,30))	2*(cos(60)+i*sin(60))
	sravn((2,90),(1,30))	Длины не равны Углы не равны
	Сопряженные числа	(2, -90), (1, -30)
test_03.txt	Ожидаемое действие	Ожидаемый результат
5 45 3 180	add((5,45),(3,180))	0.75 0.78
	sub((5,45),(3,180))	9.24 0.78
	multiply ((5,45),(3,180))	15*(cos(225)+i*sin(225))
	div((5,45),(3,180))	1.66*(cos(-135)+i*sin(-135))
	sravn((5,45),(3,180))	Длины не равны Углы не равны
	Сопряженные числа	(5, -45) (3,-180)

7. Результаты выполнения тестов

```
masha@masha-VirtualBox:~/2kurs/oop_exercise_01/tmp$ ./oop_exercise_01 < ~/2kurs/oop_exercise_01/test_01.txt
Введите первое комплексное число
Введите второе комплексное число
Первое комплексное число, модуль длины (r)  угол (u)
1 30
Второе комплексное число, модуль длины (r)  угол (u)
3.4 45

Длина и угол(в радианах) вектора суммы:
4.37359 0.726186
Длина и угол(в радианах) вектора разности:
2.4478 -2.25026
Произведение:
3.4*(cos(75)+i*sin(75))
Деление :
0.294118*(cos(-15)+i*sin(-15))
Сравнение комплексных чисел по длине вектора и углу:
Длины не равны
Углы не равны

sopr_m1:
1 -30
sopr_m2:
3.4 -45
masha@masha-VirtualBox:~/2kurs/oop_exercise_01/tmp$ ./oop_exercise_01 < ~/2kurs/oop_exercise_01/test_02.txt
Введите первое комплексное число
Введите второе комплексное число
Первое комплексное число, модуль длины (r)  угол (u)
2 90
Второе комплексное число, модуль длины (r)  угол (u)
1 30

Длина и угол(в радианах) вектора суммы:
2.99401 1.27735
Длина и угол(в радианах) вектора разности:
1.42685 2.22301
Произведение:
2*(cos(120)+i*sin(120))
Деление :
2*(cos(60)+i*sin(60))
Сравнение комплексных чисел по длине вектора и углу:
Длины не равны
Углы не равны

sopr_m1:
2 -90
sopr_m2:
1 -30
masha@masha-VirtualBox:~/2kurs/oop_exercise_01/tmp$ ./oop_exercise_01 < ~/2kurs/oop_exercise_01/test_03.txt
Введите первое комплексное число
Введите второе комплексное число
Первое комплексное число, модуль длины (r)  угол (u)
5 45
Второе комплексное число, модуль длины (r)  угол (u)
3 180

Длина и угол(в радианах) вектора суммы:
```

0.757359 0.785398
Длина и угол(в радианах) вектора разности:
9.24264 0.785398
Произведение:
15*(cos(225)+i*sin(225))
Деление :
1.66667*(cos(-135)+i*sin(-135))
Сравнение комплексных чисел по длине вектора и углу:
Длины не равны
Углы не равны

sopr_m1:
5 -45
sopr_m2:
3 -180

8. Объяснение результатов работы программы - вывод

В complex.h были заданы методы и свойства этого класса, а в fractions.cpp они были описаны. Описанные методы использовались в файле main.cpp .

Классы, описывают метода и свойства объектов, позволяют работать с этими объектами, не вдаваясь в подробности их реализации, что является примером абстракции данных. Такой подход незаменим при работе в групповых проектах.