**Московский авиационный институт**

**(Национальный исследовательский университет)**

Факультет: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Объектно-ориентированное программирование»

**Лабораторная работа № 1**

Тема: Простые классы на языке С++

Студент: Ляшун Дмитрий Сергеевич

Группа: 80-207

Преподаватель: Чернышов Л.Н.

Дата:

Оценка:

Москва, 2020

1. **Постановка задачи**

Разработать программу на языке C++ согласно варианту задания. Программа на C++ должна собираться с помощью системы сборки CMake. Программа должна получать данные из стандартного ввода и выводить данные в стандартный вывод.

**Вариант №2:**

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

- сложения add, (r1, j1) + (r2, j2);

- вычитания sub, (r1, j1) – (r2, j2);

- умножения mul, (r1, j1) ´ (r2, j2);

- деления div, (r1, j1) / (r2, j2) ;

- сравнение equ, (r1, j1) = (r2, j2), если (r1 = r2) и (j1 = j2);

- сопряженное число conj, conj(r, j) = (r, –j).

Реализовать операции сравнения по действительной части.

1. **Решение задачи**

Для реализации комплексного числа в тригонометрической форме опишем класс Complex со следующими полями:

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| r | double | Хранит значение радиуса комплексного числа. |
| phi | double | Хранит значение угла (в радианах) комплексного числа. |

В классе Complex будут реализованы следующие методы:

|  |  |
| --- | --- |
| **Метод** | **Назначение** |
| double get\_r() | Возвращает значение радиуса комплексного числа. |
| double get\_phi() | Возвращает значение угла комплексного числа. |
| void write() | Выводит значение радиуса и угла комплексного числа в стандартный поток вывода. |

Также в программе будут реализованы следующие функции работы с классом Complex:

|  |  |
| --- | --- |
| **Функция** | **Назначение** |
| void Convert\_to\_norm(Complex& arg, double& a, double& b) | Находит значение действительной части a и мнимой части b комплексного числа arg |
| Complex Convert\_to\_trig(const double a, const double b) | Создает комплексное число в тригонометрической форме по значениям a и b. |
| Complex Add(Complex &first, Complex & second) | Возвращает сумму комплексных чисел first и second. |
| Complex Sub(Complex &first, Complex & second) | Возвращает разность комплексных чисел first и second. |
| Complex Mul(Complex& first, Complex& second) | Возвращает произведение комплексных чисел first и second. |
| Complex Div(Complex& first, Complex& second) | Возвращает частное комплексных чисел first и second. |
| bool Equ(Complex& first, Complex& second) | Возвращает true, если комплексные числа first и second равны. |
| Complex Conj(Complex& arg) | Возвращает сопряженное комплексное число для arg. |
| bool Equal\_valid(Complex& first, Complex& second) | Возвращает true, если действительные части комплексных чисел first и second равны. |
| bool Less\_valid(Complex& first, Complex& second) | Возвращает true, если действительная часть комплексного числа first меньше second. |

Для работы с программой необходимо вводить данные в следующем формате: <имя операции> <радиус числа1> <угол числа 1> (<радиус числа2> <угол числа2>). Ответ на каждый запрос выводится в стандартный поток вывода.

1. **Руководство по использованию программы**

**oop\_exercise\_01** - исполняемый файл программы.

**tests1.txt** - первый набор тестов для программы.

**tests2.txt** - второй набор тестов для программы.

**Входные данные**

Для работы с программой необходимо вводить данные в следующем формате: name r1 phi1 r2 phi2, где name - название операции:

add - сложения,

sub - вычитания,

mul - умножения,

div - деления,

equ - проверки на равенство,

conj - нахождение сопряженного числа,

comp - сравнение по действительной части.

Значение углов phi1 и phi2 указываются в радианах.

**Выходные данные**

Для каждого запроса на выполнение операции выводится ответ.

**Тестовые примеры**

Ввод: add 1 1.57 3 1.57

Вывод: (r = 1, phi = 1.57) + (r = 3, phi = 1.57) = (r = 4, phi = 1.57)

Ввод: sub 8 3.14 7 9.42

Вывод: (r = 8, phi = 3.14) - (r = 7, phi = 9.42) = (r = 1.00028, phi = 3.12089)

Ввод: mul 4 0 2 3.14

Вывод: (r = 4, phi = 0) \* (r = 2, phi = 3.14) = (r = 8, phi = 3.14)

Ввод: div 20 0.78 1 3.14

Вывод: (r = 20, phi = 0.78) / (r = 1, phi = 3.14) = (r = 20, phi = -2.36)

Ввод: equ 1 1.575 3 1.42

Вывод: (r = 1, phi = 1.575) <> (r = 3, phi = 1.42)

Ввод: equ 1 1.575 1 1.575

Вывод: (r = 1, phi = 1.575) = (r = 1, phi = 1.575)

Ввод: conj 2 -1.345

Вывод: Сопряженное число для (r = 2, phi = -1.345) это (r = 2, phi = 1.345)

Ввод: comp 5 0.785 5 0.785

Вывод: Действительная часть (r = 5, phi = 0.785) = 3.53694 равна действительной части (r = 5, phi = 0.785) = 3.53694

Ввод: comp 5 0.785 3 0

Вывод: Действительная часть (r = 5, phi = 0.785) = 3.53694 больше чем действительная часть (r = 3, phi = 0) = 3

1. **Листинг программы**

Исходный программы состоит из двух файлов: **complex.cpp** - содержит объявление и реализацию класса Complex, а также все функции, которые с ним работают; и **main.cpp** - основная программа. **CMakeLists.txt** - содержит код для утилиты cmake, которая проводит сборку программы из исходного кода.

**Содержимое файла complex.cpp**

/\* Ляшун Дмитрий Сергеевич, М8О-207Б-19 \*/

#include <cmath>

#include <iostream>

const double EPS = 1e-7;

class Complex {

public:

Complex(double r1, double phi1): r(r1), phi(phi1) {}

double get\_r() { return r; }

double get\_phi() { return phi; }

void write() { std::cout << "(r = " << r << " , phi = " << phi << ")"; }

private:

double r;

double phi;

};

void Convert\_to\_norm(Complex& arg, double& a, double& b) {

a = arg.get\_r() \* cos(arg.get\_phi());

b = arg.get\_r() \* sin(arg.get\_phi());

}

Complex Convert\_to\_trig(const double a, const double b) {

double r, phi;

double cosn = a / sqrt(a\*a+b\*b);

double sinn = b / sqrt(a\*a+b\*b);

if (abs(cosn) > EPS) {

r = a / cosn;

}

else {

r = b / sinn;

}

phi = acos(cosn);

return Complex(r, phi);

}

Complex Add(Complex &first, Complex &second) {

double a1, b1, a2,b2, a3, b3;

Convert\_to\_norm(first, a1, b1);

Convert\_to\_norm(second, a2, b2);

a3 = a1 + a2;

b3 = b1 + b2;

return Convert\_to\_trig(a3, b3);

}

Complex Sub(Complex &first, Complex &second) {

double a1, b1, a2, b2, a3, b3;

Convert\_to\_norm(first, a1, b1);

Convert\_to\_norm(second, a2, b2);

a3 = a1 - a2;

b3 = b1 - b2;

return Convert\_to\_trig(a3, b3);

}

Complex Mul(Complex& first, Complex& second) {

double r = first.get\_r() \* second.get\_r(); // Для комплексных чисел в тригоном. записи умножение можно представить как: сложение углов и произведение радиусов

double phi = first.get\_phi() + second.get\_phi();

return Complex(r, phi);

}

Complex Div(Complex& first, Complex& second) {

double r = first.get\_r() / second.get\_r();

double phi = first.get\_phi() - second.get\_phi(); // Для комплексных чисел в тригоном. записи деление можно представить как: вычитание углов и деление радиусов

return Complex(r, phi);

}

bool Equ(Complex& first, Complex& second) {

return (abs(first.get\_r() - second.get\_r()) < EPS && abs(first.get\_phi() - second.get\_phi()) < EPS); // При сравнении double чисел важно учитывать их погрешность

}

Complex Conj(Complex& arg) {

return Complex(arg.get\_r(), -arg.get\_phi());

}

bool Equal\_valid(Complex& first, Complex& second) {

double a1 = first.get\_r()\*cos(first.get\_phi());

double a2 = second.get\_r()\*cos(second.get\_phi());

return (abs(a1-a2) < EPS);

}

bool Less\_valid(Complex& first, Complex& second) {

double a1 = first.get\_r()\*cos(first.get\_phi());

double a2 = second.get\_r()\*cos(second.get\_phi());

return (a1 < a2);

}

**Содержимое файла main.cpp**

/\* Ляшун Дмитрий Сергеевич, группа М8О-207Б-19 \*/

#include <string>

#include "complex.cpp"

using namespace std;

int main() {

double r1, phi1, r2, phi2;

string oper;

cout << "Для работы с комплексными числами в тригонометрической записи введите: " << endl;

cout << "\t\tadd - сложение двух чисел." << endl;

cout << "\t\tsub - вычитание двух чисел." << endl;

cout << "\t\tmul - умножение двух чисел." << endl;

cout << "\t\tdiv - деление двух чисел." << endl;

cout << "\t\tequ - проверки на равенство двух чисел." << endl;

cout << "\t\tconj - найти сопряженное число." << endl;

cout << "\t\tcomp - сравнение двух чисел по действительной части." << endl;

cout << "Введите exit - для выхода из программы." << endl;

while (true) {

cout << "> ";

cin >> oper;

if (oper == "exit") break;

cout << "Введите 1-ое число (r1, phi1): ";

cin >> r1 >> phi1;

if (oper != "conj") {

cout << "Введите 2-ое число (r2, phi2): ";

cin >> r2 >> phi2;

}

Complex first(r1, phi1), second(r2, phi2);

if (oper == "add") {

first.write();

cout << " + ";

second.write();

cout << " = ";

Add(first, second).write();

cout << endl;

}

else if (oper == "sub") {

first.write();

cout << " - ";

second.write();

cout << " = ";

Sub(first, second).write();

cout << endl;

}

else if (oper == "mul") {

first.write();

cout << " \* ";

second.write();

cout << " = ";

Mul(first,second).write();

cout << endl;

}

else if (oper == "div") {

first.write();

cout << " / ";

second.write();

cout << " = ";

Div(first,second).write();

cout << endl;

}

else if (oper == "equ") {

first.write();

cout << ( Equ(first, second) ? " = " : " <> ");

second.write();

cout << endl;

}

else if (oper == "conj") {

cout << "Сопряженное число для ";

first.write(); cout << " это ";

Conj(first).write();

cout << endl;

}

else if (oper == "comp") {

double a1, b1, a2, b2;

Convert\_to\_norm(first, a1, b1);

Convert\_to\_norm(second, a2, b2);

cout << "Действительная часть "; first.write(); cout << " = " << a1;

if (Equal\_valid(first, second)) cout << " равна ";

else if (Less\_valid(first, second)) cout << " меньше ";

else cout << " больше ";

cout << "действительной части "; second.write(); cout << " = " << a2 << endl;

}

else cout << "Ошибка! Неправильный ввод!" << endl;

}

}

**Содержимое файла CMakeLists.txt**

cmake\_minimum\_required(VERSION 2.8) # Указание необходимой версии CMake

set(SOURCES main.cpp) # присвоение переменной SOURCES значения main.cpp

add\_executable(oop\_exercise\_01 ${SOURCES}) # Компиляция исполняемого файла oop\_exercise\_01

1. **Выводы**

В результате выполнения лабораторной работы я познакомился с основами создания классов на C++, а также изучил систему сборки CMake для данного языка.

**Список литературы**

1. Справочник по языку C++ [Электронный ресурс]. URL: [http://www.cplusplus.com/](about:blank) (дата обращения: 23.09.2020)