**Московский авиационный институт**

**(Национальный исследовательский университет)**

Институт: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Объектно-ориентированное программирование»

**Лабораторная работа № 2**

Тема: Перегрузка операторов в С++

Студент: Киреев Александр Константинович

Группа: 80-206

Преподаватель: Чернышов Л.Н.

Дата:

Оценка:

Москва, 2020

1. Постановка задачи

**Разработать класс Complex** для работы с комплексными числами. Обязательно должны присутствовать операции: сложения **add**, вычитания **sub**, умножения **mul**, деления **div**, сравнения **equ**, сопряженное число **conj**, операция сравнения модулей чисел. Так же операции сложения, вычитания, умножения, деления, сравнения на равенство должны быть выполнены в виде перегрузки операторов. Необходимо реализовать пользовательский литерал для работы с константами типа **Complex**.

1. Описание программы

Все операции, указанные в задании выполняются при помощи перегруженных арифметических операторов, которые использую внутри себя ранее созданные методы класса Complex, такие как: add, sub, div, mul, equ. Только методы conj и сравнение по модулю вызываются напрямую, т.к. для них не перегружен никакой оператор. Так же в своей программе я перегрузил операторы ввода и вывода в поток для простоты написания более общего кода. В программе также реализован пользовательский литерал, который заданную константу вида “x + yi” преобразует в объект класса Complex.

1. Набор тестов

**Тест 1**:

1 + 2i

1 - 2i

add

sub

mul

div

equ

conj 1

conj 2

module

lit

end

**Тест 2**:

1.90 - 2.28i

0.05 - 10i

add

sub

mul

div

equ

conj 1

conj 2

module

lit

end

1. Результаты выполнения тестов

Вывод на тест 1:

2 + 0i

0 + 4i

5 + 0i

-0.6 + 0.8i

0

1 - 2i

1 + 2i

Eq

12.3 + 6.78i

Вывод на тест 2:

1.95 - 12.28i

1.85 + 7.72i

-22.705 - 19.114i

0.228944 + 0.188855i

0

1.9 + 2.28i

0.05 + 10i

First number is less than second

12.3 + 6.78i

1. Листинг программы

// Киреев Александр Константинович 206

#include <iostream>

#include <string>

#include <sstream>

#include <cmath>

enum class Comp {

Less,

More,

Eq

};

class Complex {

public:

// конструкторы

Complex();

Complex(const double& r, const double& i);

Complex(const Complex& val);

// методы

static Complex add(const Complex& lhs, const Complex& rhs);

static Complex sub(const Complex& lhs, const Complex& rhs);

static Complex mul(const Complex& lhs, const Complex& rhs);

static Complex div(const Complex& lhs, const Complex& rhs);

static bool equ(const Complex& lhs, const Complex& rhs);

static Complex conj(const Complex& val);

// метод сравнения по модулю

static Comp moduleComp(const Complex& lhs, const Complex& rhs);

double getReal() const;

double getImag() const;

private:

double real\_;

double imaginary\_;

};

Complex::Complex() {

real\_ = 0.0;

imaginary\_ = 0.0;

}

Complex::Complex(const double& r, const double& i) {

real\_ = r;

imaginary\_ = i;

}

// конструктор копирования

Complex::Complex(const Complex& val) {

real\_ = val.real\_;

imaginary\_ = val.imaginary\_;

}

// Методы - реализация

Complex Complex::add(const Complex& lhs, const Complex& rhs) {

return Complex(lhs.getReal() + rhs.getReal(), lhs.getImag() + rhs.getImag());

}

Complex Complex::sub(const Complex& lhs, const Complex& rhs) {

return Complex(lhs.getReal() - rhs.getReal(), lhs.getImag() - rhs.getImag());

}

Complex Complex::mul(const Complex& lhs, const Complex& rhs) {

return Complex(lhs.getReal() \* rhs.getReal() - lhs.getImag() \* rhs.getImag(),

lhs.getReal() \* rhs.getImag() + lhs.getImag() \* rhs.getReal());

}

Complex Complex::div(const Complex& lhs, const Complex& rhs) {

double d = rhs.getReal() \* rhs.getReal() + rhs.getImag() \* rhs.getImag();

assert(d != 0.0);

return Complex((lhs.getReal() \* rhs.getReal() + lhs.getImag() \* rhs.getImag()) / d,

(lhs.getImag() \* rhs.getReal() - lhs.getReal() \* rhs.getImag()) / d);

}

bool Complex::equ(const Complex& lhs, const Complex& rhs) {

return (lhs.getReal() == rhs.getReal()) && (lhs.getImag() == rhs.getImag());

}

Complex Complex::conj(const Complex& val) {

return Complex(val.real\_, -val.imaginary\_);

}

Comp Complex::moduleComp(const Complex& lhs, const Complex& rhs) {

double mL, mR;

mL = sqrt(lhs.getReal() \* lhs.getReal() + lhs.getImag() \* lhs.getImag());

mR = sqrt(rhs.getReal() \* rhs.getReal() + rhs.getImag() \* rhs.getImag());

if (mL == mR)

return Comp::Eq;

else {

return (mL > mR ? Comp::More : Comp::Less);

}

}

double Complex::getReal() const { return real\_; }

double Complex::getImag() const { return imaginary\_; }

// Перегрузка операторов

Complex operator+(const Complex& lhs, const Complex& rhs) {

return Complex::add(lhs, rhs);

}

Complex operator-(const Complex& lhs, const Complex& rhs) {

return Complex::sub(lhs, rhs);

}

Complex operator\*(const Complex& lhs, const Complex& rhs) {

return Complex::mul(lhs, rhs);

}

Complex operator/(const Complex& lhs, const Complex& rhs) {

return Complex::div(lhs, rhs);

}

bool operator==(const Complex& lhs, const Complex& rhs) {

return Complex::equ(lhs, rhs);

}

// перегрузим оператор << и >> для более удобного вывода/ввода прямоугольника в поток

std::ostream& operator<<(std::ostream& stream, const Complex& v) {

if (v.getImag() < 0)

stream << v.getReal() << " - " << abs(v.getImag()) << "i";

else

stream << v.getReal() << " + " << v.getImag() << "i";

return stream;

}

std::istream& operator>>(std::istream& stream, Complex& v) {

std::string r, i, plus;

getline(stream, r, ' ');

getline(stream, plus, ' ');

getline(stream, i, 'i');

std::stringstream sr(r);

std::stringstream si(i);

double real, imag;

sr >> real;

si >> imag;

if (plus == "-")

imag \*= -1.0;

v = Complex{real, imag};

return stream;

}

// Пользовательский литерал

Complex operator "" \_c(const char\* str, long unsigned int size) {

std::stringstream ss(str);

Complex res;

ss >> res;

return res;

}

// Вспомогательные функции

void helper() {

std::cout << "description | command \n";

std::cout << "-------------------------|---------------------\n";

std::cout << "сложить два числа | add \n";

std::cout << "вычисть из левого правое | sub \n";

std::cout << "умножить два числа | mul \n";

std::cout << "поделить левое на правое | div \n";

std::cout << "сравнить два числа | equ \n";

std::cout << "получить сопряженное | conj [1, 2] \n";

std::cout << "сравнить числа по модулю | module \n";

std::cout << "пользовательский литерал | lit \n";

std::cout << "( 12.3 + 6.78i )| \n";

std::cout << "вывести число на экран | print [1, 2] \n";

std::cout << "end | end \n";

}

// главный цикл интерактивной программы

void mainLoop(Complex& f, Complex& s) {

std::string cmd;

int which = 1;

while (cmd != "end") {

std::cin >> cmd;

// вызываем соответ-ую команду, проверяем ее на валидность

if (cmd == "help")

helper();

else if (cmd == "add") {

Complex tmp = f + s;

std::cout << tmp << std::endl;

} else if (cmd == "sub") {

Complex tmp = f - s;

std::cout << tmp << std::endl;

} else if (cmd == "mul") {

Complex tmp = f \* s;

std::cout << tmp << std::endl;

} else if (cmd == "div") {

Complex tmp = f / s;

std::cout << tmp << std::endl;

} else if (cmd == "equ") {

std::cout << (f == s) << std::endl;

} else if (cmd == "conj") {

std::cin >> which;

if (which == 1) {

std::cout << Complex::conj(f) << std::endl;

} else if (which == 2) {

std::cout << Complex::conj(s) << std::endl;

} else {

std::cout << "Wrong args!" << std::endl;

}

} else if (cmd == "module") {

Comp res = Complex::moduleComp(f, s);

if (res == Comp::Eq)

std::cout << "Eq" << std::endl;

else if (res == Comp::Less)

std::cout << "First number is less than second" << std::endl;

else

std::cout << "Second number is less than first" << std::endl;

} else if (cmd == "lit") {

std::cout << "12.3 + 6.78i"\_c << std::endl;

} else if (cmd == "print") {

std::cin >> which;

if (which == 1) {

std::cout << f << std::endl;

} else if (which == 2) {

std::cout << s << std::endl;

} else {

std::cout << "Wrong args!" << std::endl;

}

} else if (cmd == "end")

continue;

else

std::cout << "Wrong comand!" << std::endl;

}

}

int main() {

// создаем два числа

Complex f, s;

std::cout << "First complex (x + yi): " << std::endl;

std::cin >> f;

std::cout << "Second complex: " << std::endl;

std::cin >> s;

helper();

// запускаем цикл

mainLoop(f, s);

return 0;

}

1. Выводы

В C++ есть множество инструментов для обобщения работы над пользовательскими объектами, например перегрузка операторов, которая позволяет писать одинаковый код, как для пользовательских объектов, так и для уже существующих, применяя к ним одинаковые операции. Задача по идентификации оператора для конкретного класса ложится на плечи компилятора. Также в С++ можно написать свой собственный литерал для работы с константами, что также может упростить работу с ними.

1. Источники

Б. Страуструп “Язык программирования С++”.