Министерство образования и науки РФ
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования

Тульский государственный университет

КАФЕДРА АВТОМАТИКИ И ТЕЛЕМЕХАНИКИ

ПЕРЕГРУЗКА ОПЕРАТОРОВ. ПЕРЕГРУЗКА ФУНКЦИЙ

Лабораторная работа № 5
по курсу «Объектно-ориентированное программирование»

Вариант № 11

Выполнил: студент группы 220601 _____ Белым А.А.
(подпись)
Проверил: к. ф.-м. н., доцент каф. АТМ _____ Середин О.С.
(подпись)

Тула 2013

Цель работы

Изучить принцип перегрузки операторов и функций. Написать программу на C++ с использованием этого принципа.

Задание

Задан класс:

```
class CMPLX
 {
   public:
      CMPLX();  /*1  Инициализация как (0,0) */
      CMPLX (float real, float imag);  /*2  Инициализация как (real,imag) */
      void Setcompl(float, float);  /*3  Изменение числа */
      void Inc();  /*4  Инкремент: |z|=|z|+1  */
      void Dec();   /*5  Декремент: |z|=|z|-1  */
      CMPLX Add(CMPLX what);  /*6  Сложение */
      CMPLX Sub(CMPLX what);  /*7  Вычитание */
      CMPLX Mul(CMPLX what);  /*8  Умножение */
      CMPLX Cmul(float what);  /*9  Умножение на действительное число
      */
      CMPLX Div(CMPLX what);  /*10  Деление */
      void Print();  /*11  Вывод в виде "("re"," im")" */
    private:
          float re;
          float im;
  }
```

Реализовать следующие методы класса и написать демонстрационную программу, инициализирующую несколько переменных посредством конструктора №2, выводящую их, производящую над ними некоторые действия с помощью методов №№4,7,11; и выводящую результаты.

Используя данный класс, написать демонстрационную программу, в которой необходимо перегрузить функцию, соответствующую номеру варианта, а так же перегрузить два стандартных оператора по выбору.

Теоретическая справка

Реализация класса

Ниже представлено определение класса, файл CMPLX.h:

```
#ifndef CMPLX_HPP
#define CMPLX_HPP
#include <iostream>
/*!Класс комплексных чисел.*/
class CMPLX {
    public:
        /*!Конструктор по умолчанию.*/
        CMPLX();
        /*!Конструктор, задающий вещественную и мнимую части.
         * \param real вещественная часть
         * \param imag мнимая часть
         */
        CMPLX(float real,float imag);
        /*!Установка значения.
         * \param real вещественная часть
         * \param imag мнимая часть
         */
        void Setcompl(float real,float imag);
        /*!Инкремент - увеличение модуля на единицу.
         */
        void Inc();
        /*!Декремент - уменьшение модуля на единицу.*/
        void Dec();
        /*!Операция сложения комплексных чисел.
         * \param what второе слагаемое
         * \return сумму комплексных чисел
         */
        CMPLX Add(CMPLX what);
        /*!Операция вычитания комплексных чисел.
         * \param what вычитаемое
         * \return разность комплексных чисел
         */
        CMPLX Sub(CMPLX what);
        /*!Операция вычитания вещественного числа.
         * \param what вычитаемое
         * \return разность чисел
         */
        CMPLX Sub(double what);
        /*!Операция умножения комплексных чисел.
         * \param what множитель
         * \return результат умножения
         */
        CMPLX Mul(CMPLX what);
        /*!Операция умножения на вещественное число.
         * \param what множитель
         * \return результат умножения
         */
        CMPLX Cmul(float what);
        /*!Операция деления комплексных чисел.
         * \param what делитель
         * \return результат деления
         */
        CMPLX Div(CMPLX what);
        /*!Вывод числа на экран в формате (real,imag).*/
        void Print();
        /*!Оператор преинкремента.
         * \return ссылку на данное число после выполнение инкремента
         */
        CMPLX& operator++();
        /*!Оператор постинкремента.
         * \return значение числа перед выполнением инкремента.
         */
```

```
        CMPLX operator++(int);
        /*!Оператор вычитания комплексного числа.
         * \param what вычитаемое
         * \return разность чисел
         */
        CMPLX operator-(CMPLX what);
        /*!Оператор вычитания вещественного числа.
         * \param what вычитаемое
         * \return разность чисел
         */
        CMPLX operator-(double what);
        /*!Оператор вычитания из вещественного числа.
         * \param a вещественное уменьшаемое
         * \param b комплексное вычитаемое
         * \return разность чисел
         */
        friend CMPLX operator-(double a,CMPLX b);
        /*!Оператор вывода в поток.
         * \param os поток вывода
         * \param what выводимое число
         * \return ссылку на поток вывода
         */
        friend std::ostream& operator<<(std::ostream& os,
                                        const CMPLX& what);
        /*!Оператор ввода из потока.
         * \param is поток ввода
         * \param what считываемое число
         * \return ссылку на поток ввода
         */
        friend std::istream& operator>>(std::istream& is,
                                        CMPLX& what);
    private:
        //!Вещественная часть
        float re;
        //!Мнимая часть
        float im;
};
#endif //CMPLX_HPP
```

Ниже представлена реализация класса, файл CMPLX.cpp:

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

CMPLX::CMPLX():re(0),im(0){
}

CMPLX::CMPLX(float real,float imag):re(real),im(imag){
}

void CMPLX::Setcompl(float r,float i){
    re=r; im=i;
}

void CMPLX::Inc(){
    float r=sqrt(re*re+im*im)+1,phi=atan2(im,re);
    re=r*cos(phi); im=r*sin(phi);
}

void CMPLX::Dec(){
    float r=sqrt(re*re+im*im)-1,phi=atan2(im,re);
    re=r*cos(phi); im=r*sin(phi);
}

CMPLX CMPLX::Add(CMPLX what){
```

```
    CMPLX temp;
    temp.re=re+what.re;
    temp.im=im+what.im;
    return temp;
}

CMPLX CMPLX::Sub(CMPLX what){
    CMPLX temp;
    temp.re=re-what.re;
    temp.im=im-what.im;
    return temp;
}

CMPLX CMPLX::Sub(double what){
    CMPLX temp;
    temp.re=re-what;
    temp.im=im;
    return temp;
}

CMPLX CMPLX::Mul(CMPLX what){
    CMPLX temp;
    temp.re=re*what.re-im*what.im;
    temp.im=im*what.re+re*what.im;
    return temp;
}

CMPLX CMPLX::Cmul(float what){
    CMPLX temp;
    temp.re=re*what;
    temp.im=im*what;
    return temp;
}

CMPLX CMPLX::Div(CMPLX what){
    CMPLX temp;
    float denom=what.re*what.re+what.im*what.im;
    temp.re=(re*what.re+im*what.im)/denom;
    temp.im=(im*what.re-re*what.im)/denom;
    return temp;
}

void CMPLX::Print(){
    std::cout<<"("<<re<<","<<im<<")"<<std::endl;
}
CMPLX& CMPLX::operator++(){
    Inc();
    return *this;
}

CMPLX CMPLX::operator++(int){
    CMPLX c=*this;
    Inc();
    return c;
}

CMPLX CMPLX::operator-(CMPLX what){
    return Sub(what);
}

CMPLX CMPLX::operator-(double what){
    return Sub(what);
}
CMPLX operator-(double a,CMPLX b){
    return CMPLX(a,0)-b;
}
```

```
std::ostream& operator<<(std::ostream& os,
                         const CMPLX& what){
    return os<<"("<<what.re<<","<<what.im<<")";
}
std::istream& operator>>(std::istream& is,
                         CMPLX& what){
   return is>>what.re>>what.im;
}
```

Демонстрационная программа

Далее приводится демонстрационная программа, файл lab1.cpp:

```
#include "CMPLX.h"
#include <iostream>
using namespace std;
int main()
{
    CMPLX a,b;
    double r;
    cout<<"Введите комплексное число a."<<endl;
    cin>>a;
    cout<<"Введите комплексное число b."<<endl;
    cin>>b;
    cout<<"Введите вещественное число r."<<endl;
    cin>>r;
    cout<<"a="<<a<<endl;
    cout<<"b="<<b<<endl;
    cout<<"r="<<r<<endl;

    cout<<"a.Sub(b)="<<a.Sub(b)<<endl;
    cout<<"a-b="<<a-b<<endl;

    cout<<"a.Sub(r)="<<a.Sub(r)<<endl;
    cout<<"a-r="<<a-r<<endl;

    cout<<"r-a="<<r-a<<endl;

    cout<<"b++="<<b++<<endl;
    cout<<"b="<<b<<endl;
    cout<<"++b="<<++b<<endl;
    cout<<"b="<<b<<endl;

    return 0;
}
```

Инструкция программисту

Далее приводится описание функций, методов, типов данных и классов.

Класс CMPLX

Класс комплексных чисел.

```
#include <CMPLX.h>
```

Открытые члены

- **CMPLX** ()
- **CMPLX** (float real, float imag)
- void **Setcompl** (float real, float imag)
- void **Inc** ()
- void **Dec** ()
- **CMPLX Add** (**CMPLX** what)
- **CMPLX Sub** (**CMPLX** what)
- **CMPLX Sub** (double what)
- **CMPLX Mul** (**CMPLX** what)
- **CMPLX Cmul** (float what)
- **CMPLX Div** (**CMPLX** what)
- void **Print** ()
- **CMPLX** & **operator++** ()
- **CMPLX operator++** (int)
- **CMPLX operator-** (**CMPLX** what)
- **CMPLX operator-** (double what)

Закрытые данные

- float **re**

Вещественная часть

- float **im**

Мнимая часть

Друзья

- **CMPLX operator-** (double a, **CMPLX** b)
- std::ostream & **operator<<** (std::ostream &os, const **CMPLX** &what)
- std::istream & **operator>>** (std::istream &is, **CMPLX** &what)

Конструкторы

CMPLX::CMPLX()

Конструктор по умолчанию.

CMPLX::CMPLX(float *real*, float *imag*)

Конструктор, задающий вещественную и мнимую части.

Аргументы конструктора представлены в таблице 1.

Аргументы:

Таблица 1 – Аргументы конструктора (float,float)

real	вещественная часть
imag	мнимая часть

Методы

CMPLX CMPLX::Add (CMPLX *what*)

Операция сложения комплексных чисел.

Аргументы метода представлены в таблице 2.

Аргументы:

Таблица 2 – Аргументы метода сложения

what	второе слагаемое

Возвращает:

сумму комплексных чисел.

CMPLX CMPLX::Cmul (float *what*)

Операция умножения на вещественное число.

Аргументы метода представлены в таблице 3.

Аргументы:

Таблица 3 – Аргументы метода умножения на вещественное

what	множитель

Возвращает:

результат умножения.

void CMPLX::Dec ()

Декремент - уменьшение модуля на единицу.

CMPLX CMPLX::Div (CMPLX *what*)

Операция деления комплексных чисел.

Аргументы метода представлены в таблице 4.

Аргументы:

Таблица 4 – Аргументы метода деления

what	делитель

Возвращает:

результат деления.

void CMPLX::Inc ()

Инкремент - увеличение модуля на единицу.

CMPLX CMPLX::Mul (CMPLX *what*)

Операция умножения комплексных чисел.

Аргументы метода представлены в таблице 5.

Аргументы:

Таблица 5 – Аргументы метода умножения

what	множитель

Возвращает:

результат умножения.

CMPLX & CMPLX::operator++ ()

Оператор преинкремента.

Возвращает:

ссылку на данное число после выполнение инкремента

CMPLX CMPLX::operator++ (int)

Оператор постинкремента.

Возвращает:

значение числа перед выполнением инкремента.

CMPLX CMPLX::operator- (CMPLX *what*)

Оператор вычитания комплексного числа.

Аргументы оператора представлены в таблице 6.

Аргументы:

Таблица 6 – Аргументы оператора вычитания комплексного числа

what	вычитаемое

Возвращает:

разность чисел

CMPLX CMPLX::operator- (double *what*)

Оператор вычитания вещественного числа.

Аргументы оператора представлены в таблице 7.

Аргументы:

Таблица 7 – Аргументы оператора вычитания вещественного числа

what	вычитаемое

Возвращает:

разность чисел

void CMPLX::Print ()

Вывод числа на экран в формате (real,imag).

void CMPLX::Setcompl (float *real*, float *imag*)

Установка значения.

Аргументы метода представлены в таблице 8.

Аргументы:

Таблица 8 – Аргументы метода установки значения

real	вещественная часть
imag	мнимая часть

CMPLX CMPLX::Sub (CMPLX *what*)

Операция вычитания комплексных чисел.

Аргументы метода представлены в таблице 9.

Аргументы:

Таблица 9 – Аргументы метода вычитания

what	вычитаемое

CMPLX CMPLX::Sub (double *what*)

Операция вычитания вещественного числа.

Аргументы метода представлены в таблице 10.

Аргументы:

Таблица 10 – Аргументы метода вычитания вещественного числа

what	вычитаемое

Возвращает:

разность чисел

Документация по друзьям класса и функциям, относящимся к классу

CMPLX operator- (double *a*, CMPLX *b*) `[friend]`

Оператор вычитания из вещественного числа.

Аргументы оператора представлены в таблице 11.

Аргументы:

Таблица 11 – Аргументы оператора вычитания из вещественного числа

a	вещественное уменьшаемое
b	комплексное вычитаемое

Возвращает:

разность чисел

std::ostream& operator<< (std::ostream & *os*, const CMPLX & *what*) `[friend]`

Оператор вывода в поток.

Аргументы оператора представлены в таблице 12.

Аргументы:

Таблица 12 – Аргументы оператора вывода в поток

os	поток вывода
what	выводимое число

Возвращает:

ссылку на поток вывода

std::istream& operator>> (std::istream & *is*, CMPLX & *what*) `[friend]`

Оператор ввода из потока.

Аргументы оператора представлены в таблице 13.

Аргументы:

Таблица 13 – Аргументы оператора ввода из потока

is	поток ввода
what	считываемое число

Возвращает:

ссылку на поток ввода

Инструкция пользователю

Данная программа производит инкремент и вычитание комплексных и вещественных чисел.

Для работы введите два комплексных числа, вводя вещественную и мнимую части через пробел, а после этого вещественное число.

После этого программа выведет значения этих чисел, результат их инкремента и разности.

Контрольный пример

Ниже на рисунке 1 представлен пример работы программы, использующей класс комплексных чисел.

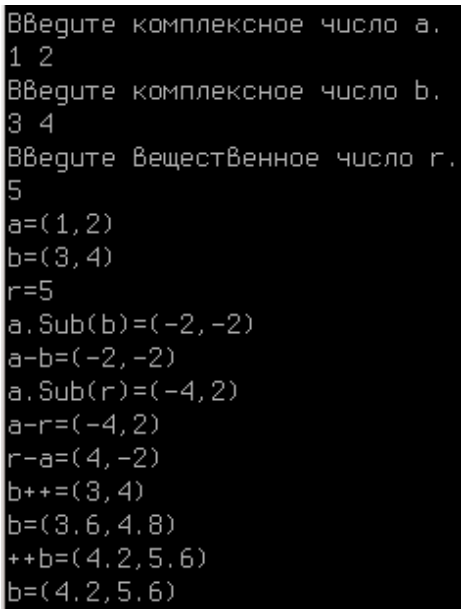

Рисунок 1— Пример работы программы, оперирующей комплексными числами

Вывод

В данной лабораторной работе я познакомился с перегрузкой методов и операторов в языке Си++. Был доработан класс комплексных чисел, в который была добавлена возможность вычитания вещественного числа, и поддержка операторов инкремента и вычитания.