OOP 第五周作业文档

2019010175 孔瑞阳 土木 92

一、项目信息

1、功能说明

实现了一个复数类 CP_Complex。

这个复数类支持与复数类 CP_Complex、double、int 进行运算。

支持的运算包括：+、-、*、/、前++、后++、前--、后--、==、!=。

（其中，==、!= 只支持 CP_Complex 与 CP_Complex 的判断）。

2、软件构件介绍

文件	功能介绍
CP_Complex.h/cpp	实现的复数类
CP_ComplexTest.h/cpp	复数类的测试（包括自动/手动）
CP_ComplexMain.cpp	主程序

3、测试环境

CPU	Intel(R) Core(TM)i7-9750H CPU @ 2.6Ghz 6 核 12 线程
GPU	NVIDIA GeForce RTX2070
RAM	DDR4 16G+16G
Operating System	Microsoft Windows 版本 1809 (OS 内部版本 17763.1098)
Compiler	MSVC++ 14.24

二、模型

1、计算公式

设 $c_1 = a + bi$，$c_2 = c + di$，x

对于复数间的运算：

$$c_1 + c_2 = (a + c) + (b + d)i$$

$$c_1 - c_2 = (a - c) + (b - d)i$$

$$c_1 * c_2 = (ac - bd) + (bc + ad)i$$

$$c_1 / c_2 = \frac{a + bi}{c + di} = \frac{(a + bi)(c - di)}{(c + di)(c - di)} = \frac{ac + bd}{c^2 + d^2} + \frac{bc - ad}{c^2 + d^2}i$$

$$c_1 ++ = ++ c_1 = (a + 1) + bi$$

$$c_1 -- = -- c_1 = (a - 1) + bi$$

其中，前++、前--的返回值是运算之后的结果，后++、后--返回值是运算前的结果。

对于复数与整数/实数之间的运算。

$c_1 + x = x + c_1 = (a + x) + bi$

$c_1 - x = (a - x) + bi$，$x - c_1 = (x - a) - bi$

$c_1 * x = x * c_1 = xa + xbi$

$c_1 / x = (a / x) + (b / x)i$，$x / c_1 = \frac{xa}{a^2 + b^2} - \frac{xb}{a^2 + b^2}i$

均采用全局函数重载运算符。

2、复数的输出

实部不为 0 就输出实部，虚部不为 0 就输出虚部（加上 i），
如果实部和虚部都是 0 的情况，输出 0。
如果虚部 i 的系数是 1 或者-1 时，输出 i 和-i。
当实部不为 0，虚部>0，则用+号连接。

3、复数的等于比较

由于自动测试中需要进行对拍，所以要判断两个复数是否相等。
当 a、c,b、d 之间的相对误差或绝对误差都小于 10^-10 时，则判断 c1==c2。
具体的实现在程序的以下两个实现中：

```
bool operator == (CP_Complex& c1, CP_Complex& c2)
bool operator != (CP_Complex& c1, CP_Complex& c2)
```

三、单元测试

1、手动测试(complexManualTest)

等价类划分

类型	等价类	选取案例
复数	实数	114
	纯虚数	-514i
	零	0
	一般复数	-19.19+810i
实数	正数	114.514
	零	0
	负数	-1919.810
整数	正数	114
	零	0
	负数	-514

从上至下依次编号为 c1~c4，d1~d6。

加法

+	c1	c2	c3	c4	d1	d2	d3	d4	d5	d6
c1	228	114-514i	114	94.81+810i	228.514	114	-1805.81	228	114	-400
c2	114-514i	-1028i	-514i	-19.19+296i	114.514-514i	-514i	-1919.81-514i	114-514i	-514i	-514-514i
c3	114	-514i	0	-19.19+810i	114.514	0	-1919.81	114	0	-514
c4	94.81+810i	-19.19+296i	-19.19+810i	-38.38+1620i	95.324+810i	-19.19+810i	-1939+810i	94.81+810i	-19.19+810i	-533.19+810i
d1	228.514	114.514-514i	114.514	95.324+810i						
d2	114	-514i	0	-19.19+810i						
d3	-1805.81	-1919.81-514i	-1919.81	-1939+810i						
d4	228	114-514i	114	94.81+810i						
d5	114	-514i	0	-19.19+810i						
d6	-400	-514-514i	-514	-533.19+810i						

减法

-	c1	c2	c3	c4	d1	d2	d3	d4	d5	d6
c1	0	114+514i	114	133.19-810i	-0.514	114	2033.81	0	114	628
c2	-114-514i	0	-514i	19.19-1324i	-114.514-514i	-514i	1919.81-514i	-114-514i	-514i	514-514i
c3	-114	514i	0	19.19-810i	-114.514	0	1919.81	-114	0	514
c4	-133.19+810i	-19.19+1324i	-19.19+810i	0	-133.704+810i	-19.19+810i	1900.62+810i	-133.19+810i	-19.19+810i	494.81+810i
d1	0.514	114.514+514i	114.514	133.704-810i						
d2	-114	514i	0	19.19-810i						
d3	-2033.81	-1919.81+514i	-1919.81	-1900.62-810i						
d4	0	114+514i	114	133.19-810i						
d5	-114	514i	0	19.19-810i						
d6	-628	-514+514i	-514	-494.81-810i						

乘法

*	c1	c2	c3	c4	d1	d2	d3	d4	d5	d6
c1	12996	-58596i	0	-2187.66+92340i	13054.6	0	-218858	12996	0	-58596
c2	-58596i	-264196	0	416340+9863.66i	-58860.2i	0	986782i	-58596i	0	264196i
c3	0	0	0	0	0	0	0	0	0	0
c4	-2187.66+92340i	416340+9863.66i	0	-655732-31087.8i	-2197.52+92756.3i	0	36841.2-1.55505e+06i	-2187.66+92340i	0	9863.66-416340i
d1	13054.6	-58860.2i	0	-2197.52+92756.3i						
d2	0	0	0	0						
d3	-218858	986782i	0	36841.2-1.55505e+06i						
d4	12996	-58596i	0	-2187.66+92340i						
d5	0	0	0	0						
d6	-58596	264196i	0	9863.66-416340i						

除法

-	c1	c2	c3	c4	d1	d2	d3	d4	d5	d6
c1	1	0.22179i	0	-0.00333247-0.140662i	0.995511	inf	-0.0593809	1	inf	-0.22179
c2	-4.50877i	1	0	-0.634212+0.0150253i	-4.48853i	-infi	0.267735i	-4.50877i	-infi	i
c3	0	0	0	0	0	0	0	0	0	0
c4	-0.168333+7.10526i	-1.57588-0.0373346i	0	1	-0.167578+7.07337i	-inf+infi	-0.168333+7.10526i	-133.19+810i	-inf+infi	0.0373346-1.57588i
d1	1.00451	0.22279i	0	-0.00334749-0.141296i						
d2	0	0	0	0						
d3	-16.8404	-3.73504i	0	0.0561202+2.36881i						
d4	1	0.22179i	0	-0.00333247-0.140662i						
d5	0	0	0	0						
d6	-628	-i	0	0.0150253+0.634212i						

后++

	结果	返回值
c1	115	114
c2	1-514i	-514i
c3	1	0
c4	-18.19+810i	-19.19+810i

前++

	结果	返回值
c1	115	115
c2	1-514i	1-514i
c3	1	1
c4	-18.19+810i	-18.19+810i

后--

	结果	返回值
c1	113	114
c2	-1-514i	-514i
c3	-1	0
c4	-20.19+810i	-19.19+810i

前--

	结果	返回值
c1	113	113
c2	-1-514i	-1-514i
c3	-1	-1
c4	-20.19+810i	-20.19+810i

1、所有数据在手打的过程中都进行了验证。

2、在乘法、除法的验证中，用 cout 输出会有四舍五入、科学计数法的情况，经过验证，在误差范围之内。

3、其中，加法、乘法均满足对称性，减法满足反对称性、除法满足倒对称性，符合客观情况。

4、在除法的验证中，除以 0 的情况本来就是 undefined behavior，虽然可以不用解决，但也可以分析结果的原因。

首先，c++中除法有以下规则：

（1）0/x=0

（2）在不满足(1)的情况下，+x/0=inf，-x/0=-inf

在涉及 c3 的运算中，由于计算的过程要计算(ac+bd)/(c^2+d^2)，这时候是 0/0，属于情况(1)，所以输出是 0，其他情况输出就是 inf/-inf。

2、自动测试(complexAutoTest)

事实上，c++自带复数类 complex，#include <complex> 即可。
于是采用 stl 的 complex 类与手写的 CP_Complex 进行对拍。

每次随机生成两个 CP_Complex c1,c2，int d1，double d2。
对于它们之间的所有 24 种运算（+-*/各 5 种，++,--各 2 种）全部测试一遍，如果出现错误则输出错误的数据，否则一直进行循环。（当 d1 随机出 0 时，不进行 c1/d1 的测试。）

经过 10 分钟的对拍，没有出现错误。
根据估算，10 分钟大致可以进行千亿(10^11)次计算，基本可以验证程序的正确性。