

```

1 //Sources:complex.cpp
2 #include "complex.h"
3 #include <iostream>
4 #include <cmath>
5
6 bool complex::operator==(const complex & other)
7 {
8     /*
9     一般我们比较两个对象是否相等
10    会比较对象里的所有变量是否相等
11    若相等则认为两个对象相等
12    布尔值是c++关键字
13    */
14    return ( (real==other.real) && (imag==other.imag) );
15 }
16
17 complex operator+(float a, complex c)
18 {
19     /*
20     这是非类里的函数，不需要类解析符
21     但是访问类里私有变量因此要在类里加友元并声明
22     重载了加号+运算符
23     complex a=(float)c+(complex)b
24     相当于 a = c.operator+(b);
25     对一个复数实部进行加运算，虚部不变
26     */
27     complex temp;
28     temp.real = a + c.real;
29     temp.imag = c.imag;
30     return temp;
31 }
32
33 complex operator-(float a, complex c)
34 {
35     /*
36     这是非类里的函数，不需要类解析符
37     但是访问类里私有变量因此要在类里加友元并声明
38     重载了减号-运算符
39     complex a=(float)c-(complex)b
40     相当于 a = c.operator-(b);
41     对一个复数实部进行减运算，虚部不变
42     */
43     complex temp;
44     temp.real = a - c.real;
45     temp.imag = c.imag;
46     return temp;
47 }
48
49 complex operator*(float a, complex c)
50 {
51     /*
52     这是非类里的函数，不需要类解析符
53     但是访问类里私有变量因此要在类里加友元并声明
54     重载了乘号*运算符
55     complex a=(float)c*(complex)b
56     相当于 a = c.operator*(b);
57     对一个复数实部进行乘运算，虚部和实部分别相乘
58     */
59     complex temp;
60     temp.real = a * c.real;
61     temp.imag = a * c.imag;
62     return temp;
63 }
64
65 complex operator/(float a, complex c)
66 {
67     /*
68     这是非类里的函数，不需要类解析符
69     但是访问类里私有变量因此要在类里加友元并声明
70     重载了除号/运算符
71     complex a=(float)c/(complex)b
72     相当于 a = c.operator/(b);
73     对一个复数实部进行除运算，虚部和实部分别相除
74     */
75     if(c.real==0&& c.imag==0)
76     {
77         std::cout<<"Input Error!"<<std::endl;
78     }
79     else
80     {
81         complex temp;
82         temp.real = a / c.real;
83         temp.imag = a / c.imag;
84         return temp;

```

```

85     }
86 }
87
88
89 std::ostream &operator<<(std::ostream &output, complex c)
90 {
91     /*
92     重载了重定向<<运算符
93     重定向到标准输出流，由于要改变值，所以做引用传递
94     */
95     output<<"The complex is "<<c.real<<" + "<<c.imag<<"i"<<std::endl;
96     output<<"The complex 's real part is "<<c.real<<" ,its imag part is
97     "<<c.imag<<std::endl;
98     return output;
99 }
100
101 std::istream &operator>>(std::istream &input, complex &c)
102 {
103     /*
104     重载了重定向>>运算符
105     重定向到标准输入流，由于要改变值，所以做引用传递
106     */
107     std::cout << "Please enter the real part:";
108     input >> c.real;
109     std::cout << "please enter the imaginery part:";
110     input >> c.imag;
111     return input;
112 }
113
114 complex complex::operator+(complex c)
115 {
116     /*
117     重载了加号+运算符
118     complex a=(complex)b+(complex)c
119     相当于 a = b.operator+(c);
120     对两个复数进行加运算
121     */
122     complex temp;
123     temp.real = real + c.real;
124     temp.imag = imag + c.imag;
125     return temp;
126 }
127
128 complex complex::operator+(float c)
129 {
130     /*
131     重载了加号+运算符
132     complex a=(complex)b+(float)c
133     相当于 a = b.operator+(c);
134     对一个复数实部进行加运算，虚部不变
135     */
136     complex temp;
137     temp.real = real + c;
138     temp.imag = imag;
139     return temp;
140 }
141
142 complex complex::operator-(complex c)
143 {
144     /*
145     重载了减号-运算符
146     complex a=(complex)b-(complex)c
147     相当于 a = b.operator-(c);
148     对两个复数进行减运算
149     */
150     complex temp;
151     temp.real = real - c.real;
152     temp.imag = imag - c.imag;
153     return temp;
154 }
155
156 complex complex::operator-(float c)
157 {
158     /*
159     重载了减号-运算符
160     complex a=(complex)b-(float)c
161     相当于 a = b.operator-(c);
162     对一个复数实部进行减运算，虚部不变
163     */
164     complex temp;
165     temp.real = real - c;
166     temp.imag = imag;
167     return temp;
168 }
169
170 complex complex::operator*(complex c)

```

```

168 {
169 /*
170 重载了乘号*运算符
171 complex a=(complex)b*(complex)c
172 相当于 a = b.operator*(c);
173 对两个复数进行乘运算
174  $(a+bi) * (c+di) = (ac-bd) + (bc+ad)i$ ,
175 a=real;
176 b=imag;
177 c=c.real;
178 d=c.imag;
179 temp.real=(real*c.real-imag*c.imag)
180 temp.imag=(imag*c.real+real*c.imag)
181 */
182 if(c.real==0&& c.imag==0)
183 {
184     std::cout<<"Input Error!"<<std::endl;
185 }
186 else
187 {
188     complex temp;
189     temp.real = (real*c.real-imag*c.imag);
190     temp.imag = (imag*c.real+real*c.imag);
191     return temp;
192 }
193 }
194 complex complex::operator*(float c)
195 {
196 /*
197 重载了乘号*运算符
198 complex a=(complex)b*(float)c
199 相当于 a = b.operator*(c);
200 对一个复数实部进行乘运算，虚部和实部分别相乘
201 */
202 complex temp;
203 temp.real = real * c;
204 temp.imag = imag * c;
205 return temp;
206 }
207
208 complex complex::operator/(complex c)
209 {
210 /*
211 重载了除号/运算符
212 complex a=(complex)b/(complex)c
213 相当于 a = b.operator/(c);
214 对两个复数进行/运算
215  $(a+bi) / (c+di) = ((ac+bd) / (c^2+d^2)) + ((bc-ad) / (c^2+d^2)) i$ ,
216 a=real;
217 b=imag;
218 c=c.real;
219 d=c.imag;
220 temp.real=(real*c.real-imag*c.imag)/(c.real*c.real+c.imag*c.imag);
221 temp.imag=(imag*c.real+real*c.imag)/(c.real*c.real+c.imag*c.imag);
222 */
223 if(c.real==0&& c.imag==0)
224 {
225     std::cout<<"Input Error!"<<std::endl;
226 }
227 else
228 {
229     complex temp;
230     temp.real = (real*c.real-imag*c.imag)/(c.real*c.real+c.imag*c.imag);
231     temp.imag = (imag*c.real+real*c.imag)/(c.real*c.real+c.imag*c.imag);
232     return temp;
233 }
234 }
235 complex complex::operator/(float c)
236 {
237 /*
238 重载了除号/运算符
239 complex a=(complex)b/(float)c
240 相当于 a = b.operator/(c);
241 对一个复数实部进行除运算，虚部和实部分别相除
242 */
243 if(c==0)
244 {
245     std::cout<<"Input Error!"<<std::endl;
246 }
247 else
248 {
249     complex temp;
250     temp.real = real / c;
251     temp.imag = imag / c;

```

```

252         return temp;
253     }
254 }
255
256 float complex::abs()
257 {
258     return sqrt(real*real+imag*imag);
259 }
260
261 complex::complex()
262 {
263     real=0.0;
264     imag=0.0;
265 }
266
267 complex::complex(float r,float i)
268 {
269     real=r;
270     imag=i;
271 }
272
273 void complex::PrintComplex()
274 {
275     std::cout<<"The complex is ("<<real<<") + ("<<imag<<") i"<<std::endl;
276     std::cout<<"The complex 's real part is "<<real<<" ,its imag part is "<<imag<<std::endl;
277 }
278
279 void complex::setComplex(float r,float i)
280 {
281     real=r;
282     imag=i;
283 }
284
285 float complex::getImag()
286 {
287     return imag;
288 }
289
290 float complex::getReal()
291 {
292     return real;
293 }
294
295

```