

以 P. 324 - P. 325 例 10.10 为基础并进行适当扩展，讨论转换构造函数及类型转换函数的使用，完成下列文档

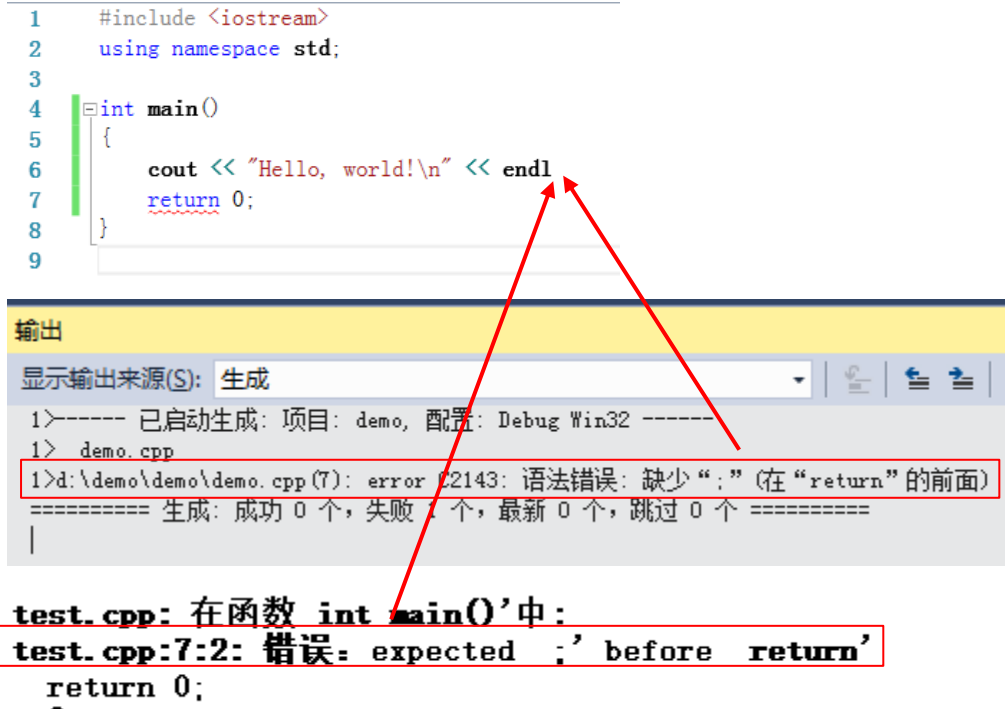
【填写方法：】

1、依次用 VS2017/Linux 编译指定的源程序文件

如果编译正确，则对应位置填写运行结果并给出得到此结果的原因解释

如果编译错误，则对应位置填写该行的编译错误提示及错误原因分析

2、如果编译器报多个错误，填写源程序文件对应的错误提示即可，示例如下，将红色框截图即可：



3、如果 main 函数中某一句错误，则将该句及下面的打印语句全部注释掉，继续观察其余正确语句的运行结果（示例如下）


<pre> c3 = c1 + Complex(2.5); //假设此句错误 c3.display(); </pre>	→	<pre> // c3 = c1 + Complex(2.5); // c3.display(); </pre>
---	---	--

4、用蓝色加粗字体填写

5、不需要填写的部分可以删除（例如：某句正确，则错误部分不填，可以删除）

【10-b1-1. cpp :】无转换构造函数、无类型转换函数、友元方式实现复数+

◆ `c3 = c1 + Complex(2.5)`


编译错误, VS2017 下:  (可删除横线后贴图)

Linux 下: `10-b1-1.cpp:60:23: 错误: 对'Complex::Complex(double)'的调用没有匹配的函数`
`c3 = c1 + Complex(2.5);`

(可删除横线后贴图)

错误原因分析: [Complex 类没有形参列表为一个 double 的构造函数](#)

◆ `c3 = c1 + 2.5`


编译错误, VS2017 下:  (可删除横线后贴图)

Linux 下: `10-b1-1.cpp:63:10: 错误: no match for 'operator+' (operand types are 'Complex' and 'double')`
`c3 = c1 + 2.5;` (可删除横线后贴图)

删除横线后贴图)

错误原因分析: [没有重载支持 Complex 和 double 相加的+号](#)

◆ `c3 = 2.5 + c1`

编译错误, VS2017 下:  (可删除横线后贴图)

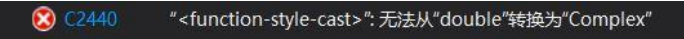
Linux 下: `10-b1-1.cpp:66:11: 错误: no match for 'operator+' (operand types are 'double' and 'Complex')`
`c3 = 2.5 + c1;` (可删除横线后贴图)

删除横线后贴图)

错误原因分析: [没有重载支持 double 和 Complex 相加的+号](#)

【10-b1-2. cpp :】无转换构造函数、无类型转换函数、成员方式实现复数+

◆ `c3 = c1 + Complex(2.5)`

编译错误, VS2017 下:  (可删除横线后贴图)

Linux 下: `10-b1-2.cpp:60:23: 错误: 对'Complex::Complex(double)'的调用没有匹配的函数`
`c3 = c1 + Complex(2.5);` (可删除横线后贴图)

删除横线后贴图)

错误原因分析: [Complex 类没有形参列表为一个 double 的构造函数](#)

◆ `c3 = c1 + 2.5`

编译错误, VS2017 下:  (可删除横线后贴图)

Linux 下: `10-b1-2.cpp:63:10: 错误: no match for 'operator+' (operand types are 'Complex' and 'double')`
`c3 = c1 + 2.5;` (可删除横线后贴图)

删除横线后贴图)

错误原因分析: [没有重载支持 Complex 和 double 相加的+号](#)

◆ `c3 = 2.5 + c1`

编译错误, VS2017 下:  (可删除横线后贴图)

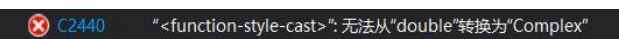
Linux 下: `10-b1-2.cpp:66:11: 错误: no match for 'operator+' (operand types are 'double' and 'Complex')`
`c3 = 2.5 + c1;` (可删除横线后贴图)

删除横线后贴图)

错误原因分析: [没有重载支持 double 和 Complex 相加的+号](#)

【10-b1-3. cpp :】无转换构造函数、有类型转换函数、友元方式实现复数+

◆ `c3 = c1 + Complex(2.5)`

编译错误, VS2017 下:  (可删除横线后贴图)

Linux 下: 10-b1-3.cpp:60:23: 错误: 对 'Complex::Complex(double)' 的调用没有匹配的函数
c3 = c1 + Complex(2.5);

(可删除横线后贴图)

错误原因分析: Complex 类没有形参列表为一个 double 的构造函数

◆ c3 = c1 + 2.5


编译错误, VS2017 下:  (可删除横线后贴图)

Linux 下: 10-b1-3.cpp:63:5: 错误: no match for 'operator=' (operand types are 'Complex' and 'double')
c3 = c1 + 2.5;

(可删除横线后贴图)

错误原因分析: 只允许 Complex 转化为 double 的类型转换 (等号右边), 没有允许 double 转化为 Complex 的类型转换 (等号)

◆ c3 = 2.5 + c1

编译错误, VS2017 下:  (可删除横线后贴图)

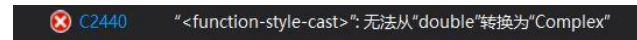
Linux 下: 10-b1-3.cpp:66:5: 错误: no match for 'operator=' (operand types are 'Complex' and 'double')
c3 = 2.5 + c1;

(可删除横线后贴图)

错误原因分析: 只允许 Complex 转化为 double 的类型转换 (等号右边), 没有允许 double 转化为 Complex 的类型转换 (等号)

【10-b1-4.cpp :】无转换构造函数、有类型转换函数、成员方式实现复数+

◆ c3 = c1 + Complex(2.5)

编译错误, VS2017 下:  (可删除横线后贴图)

Linux 下: 10-b1-4.cpp:60:23: 错误: 对 'Complex::Complex(double)' 的调用没有匹配的函数
c3 = c1 + Complex(2.5);

(可删除横线后贴图)

错误原因分析: Complex 类没有形参列表为一个 double 的构造函数

◆ c3 = c1 + 2.5

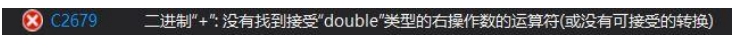
编译错误, VS2017 下:  (可删除横线后贴图)

Linux 下: 10-b1-4.cpp:63:5: 错误: no match for 'operator=' (operand types are 'Complex' and 'double')
c3 = c1 + 2.5;

(可删除横线后贴图)

错误原因分析: 只允许 Complex 转化为 double 的类型转换 (等号右边), 没有允许 double 转化为 Complex 的类型转换 (等号)

◆ c3 = 2.5 + c1

编译错误, VS2017 下:  (可删除横线后贴图)

Linux 下: 10-b1-4.cpp:66:5: 错误: no match for 'operator=' (operand types are 'Complex' and 'double')
c3 = 2.5 + c1;

(可删除横线后贴图)

错误原因分析: 只允许 Complex 转化为 double 的类型转换 (等号右边), 没有允许 double 转化为 Complex 的类型转换 (等号)

【10-b1-5.cpp :】有转换构造函数、无类型转换函数、友元方式实现复数+

◆ c3 = c1 + Complex(2.5)

编译正确, 运行结果: 5.5+4i, 能得到此结果的原因: Complex 类有了形参列表为一个 double 的构造函数

◆ c3 = c1 + 2.5

编译正确, 运行结果: 5.5+4i, 能得到此结果的原因: 2.5 被转换为了 Complex 对象, 与 c1 一起参与加法运算

◆ $c3 = 2.5 + c1$

编译正确，运行结果：5.5+4i，能得到此结果的原因：2.5 被转换为了 Complex 对象，与 c1 一起参与加法运算

【10-b1-6.cpp】有转换构造函数、无类型转换函数、成员方式实现复数+

◆ $c3 = c1 + \text{Complex}(2.5)$

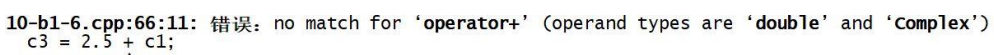
编译正确，运行结果：5.5+4i，能得到此结果的原因：Complex 类有了形参列表为一个 double 的构造函数

◆ $c3 = c1 + 2.5$

编译正确，运行结果：5.5+4i，能得到此结果的原因：2.5 被转换为了 Complex 对象，与 c1 一起参与加法运算

◆ $c3 = 2.5 + c1$

编译错误，VS2017 下：（可删除横线后贴图）

Linux 下：

（可删除横线后贴图）

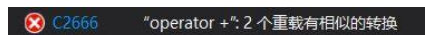
错误原因分析：加法运算符从左到右运算，2.5 不是 Complex 对象，无法像 10-b1-6 一样触发隐式类型转换变成 Complex 对象，重载+号的成员函数由于两边不都是 Complex 对象，无法发挥作用

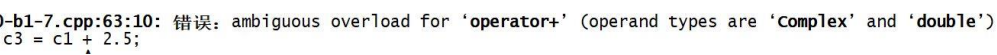
【10-b1-7.cpp】有转换构造函数、有类型转换函数、友元方式实现复数+

◆ $c3 = c1 + \text{Complex}(2.5)$

编译正确，运行结果：5.5+4i，能得到此结果的原因：Complex 类有了形参列表为一个 double 的构造函数

◆ $c3 = c1 + 2.5$

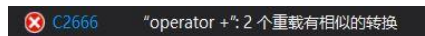
编译错误，VS2017 下：（可删除横线后贴图）

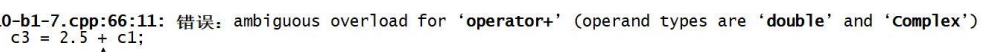
Linux 下：

（可删除横线后贴图）

错误原因分析：Complex 类的构造函数（返回 Complex 对象）与 Complex 类的类型转换函数（返回 double）在此处都可以执行，两者发生冲突，产生二义性

◆ $c3 = 2.5 + c1$

编译错误，VS2017 下：（可删除横线后贴图）

Linux 下：

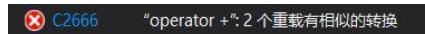
错误原因分析：Complex 类的构造函数（返回 Complex 对象）与 Complex 类的类型转换函数（返回 double）在此处都可以执行，两者发生冲突，产生二义性

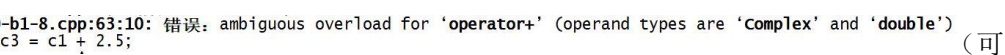
【10-b1-8.cpp】有转换构造函数、有类型转换函数、成员方式实现复数+

◆ $c3 = c1 + \text{Complex}(2.5)$

编译正确，运行结果：5.5+4i，能得到此结果的原因：Complex 类有了形参列表为一个 double 的构造函数

◆ $c3 = c1 + 2.5$

编译错误，VS2017 下：（可删除横线后贴图）

Linux 下：（可

删除横线后贴图）

错误原因分析：Complex 类的构造函数（返回 Complex 对象）与 Complex 类的类型转换函数（返回 double）在此处都可以执行，两者发生冲突，产生二义性

◆ $c3 = 2.5 + c1$

编译正确，运行结果：5.5+0i，能得到此结果的原因：Complex 类的类型转换函数（返回 double）无法执

行（与 10-b1-6 的第三个语句无法执行的原因相同），2.5 被转换为 Complex 对象，参与加法运算

【10-b1-9.cpp :】有转换构造函数、有类型转换函数、成员方式实现复数+、另有 double+Complex

◆ 仅讨论语句 `c3 = 2.5 + c1`，回答下列问题

1、为什么编译不错？

+号的重载函数中的 `Complex operator+(const double &d1, const Complex &c2)` 与该语句完美对应，不需要调用构造函数、类型转换函数等，不会产生临时变量，也就不会产生二义性。

2、运行结果是多少？

5.5+4i

3、为什么和 10-b1-8 结果不同？

10-b1-8 中，没有形参完全对应的函数，无法直接进行运算，必须要改变 2.5 或 c1 的类型才能进行后续的运算，但在究竟是改变 2.5 还是改变 c1 这一问题上发生了二义性。10-b1-9 中，恰好有一个对应的函数，就不必再进行类型转换，也就避免了二义性问题。

【10-b1-10.cpp :】单独讨论有类型转换的情况下，cout 重载的输出结果与期望值不同

◆ 目前 main 函数中第 4 个输出语句与期望值不同，原因是：`a+b` 产生了一个临时变量，临时变量被视为 `const`，而常变量不能作为引用类型的实参传入函数，因此流提取运算符 `<<` 没有找到对应的重载函数（重载函数没有发挥作用）而是隐式地调用了 Complex 类中定义的类型转换函数 `double`，输出了转化为 `double` 类型的 $\sqrt{3^2 + 3^2}$ 的值

◆ 仅允许改动两行，使程序输出与期望值相同：

改动第 33 行，原内容：`friend ostream & operator<<(ostream & stream, Complex &c);`

新内容：`friend ostream & operator<<(ostream & stream, const Complex &c);`

改动第 36 行，原内容：`ostream &operator<<(ostream &out, Complex &c)`

新内容：`ostream &operator<<(ostream &out, const Complex &c)`