北京师范大学2018-2019学年第一学期 “C++程序设计”

课堂练习四

姓 名	学 号	日 期
袁宇昊	201611130126	2018.12.5

Group 1 exercises

You have to finish all the problems in this group.

Exercise 1:

Fill in the blanks in each of the following statements:

a) Treating a base-class object as a(n) ***derived-class object*** can cause errors.

b) Polymorphism helps eliminate ***similar*** logic.

c) If a class contains at least one pure virtual function, it’s a(n) ***virtual*** class.

d) Classes from which objects can be instantiated are called ***concrete*** classes.

e) The destructors of any derived classes are also virtual if ***base-class*** destructor is declared virtual.

f) Constructors cannot be declared ***virtual***.

g) Dynamic binding with virtual functions occurs only off ***reference*** and ***pointer***.

h) Overridable functions are declared using keyword ***virtual*** .

i) To call a function appropriately at execution time is known as ***Dynamic binding*** .

Exercise 2:

State whether each of the following is true or false. If false, explain why.

a) Polymorphism enables “programming in the specific.”

True slide 4

b) With polymorphism, one function call can cause different actions to occur.

True especially the virtual function

c) Polymorphism is implemented only via dynamic binding.

False also statics binding

d) After the derived-class destructor runs, the destructors for all of that class’s base classes run all the way up the hierarchy.

False it happens only with a base-class pointer points a derived-class with virtual destructors .But if a derived-class runs its destructor,it only runs its own.

e) Objects of an abstract class may be instantiated. (Deitel)

False an abstract class contains at least one pure virtual function.So the virtual functions need to be defined by its derived-class.

Group 2 exercises

Please select **one** topic from this group and discuss it with the members in your group, then write your opinion below.

Topic 2:

A class Rational with a lot of operator overloading member functions.

Weakness:

Denominator cannot be 0 . The program ignores this situation.

no rational simplification(like fractional).

cannot divided by 0.

Topic 3:

```
cout << 14 << " + " << b << " = " << 14 + b << endl;
```

Reporting error messages when compiling this new line.

```
C:\Users\HASEE\Desktop\rational.cpp|82|error: no match for 'operator+' (operand types
are 'double' and 'Rational')|//error massage
```

the massage says that there is no match for 'operator+' with 'double' and 'Rational'. Obviously, type 'double' means '14' , and type 'Rational' means 'b', because there is only one '+' operator in this line.

So why no match ? Simply we did not define operator '+' between 'double' and 'Rational'. The operator '+' overloading member functions we defined in class Rational only support 'Rational' and 'Rational'.

So how do we solve it ? Two ways.

One is trans '14' double into '(14,1)' Rational.

```
cout << 14 << " + " << b << " = " << Rational(14) + b << endl;
```

Or we define another overloading operator '+' function as non-member.

```
friend Rational operator +(const Rational& a,const Rational &b);//in class

Rational operator +(const Rational& a,const Rational &b){//out class
    return Rational((a._n * b._d) + (a._d * b._n), a._d * b._d);
}
```

when this function deal with '14+b', it trans 14 into Rational a.

Here is the problem:

why this line

```
cout<< b + 14 <<endl;
```

is ok , while 14 + b is not.

Group 3 exercises

Please select **one** exercise from this group and attempt to program it， then write here the questions you want ask during the seminar time.

Exercise 1:

Group3Exercise1.cpp

Question:

```
cout << p1 += p2 <<endl;//error
cout << (p1+=p2) <<endl;//ok

cout << p1 + p2 <<endl;//ok
cout << (p1+p2) <<endl;//ok
```

error massage :

```
error: no match for 'operator<<' (operand types are 'Polynomial' and '<unresolved
overloaded function type>')|
```