Chapter 07 三个修饰符

Key Point：

- ●static
- ●abstract
- ●final

答案：

1. 输出结果：

   300

   200

   300

   400

2. E、G

   静态方法中不能访问非静态成员，即不能访问非静态变量和调用非静态函数。

3. 输出结果：1、2、3

   用这种方式可以用来统计总共创建了多少个对象。

4. 输出结果：

   In Static

   MyClass()

   20

MyClass(int)

10

5.输出结果：

m1 in Super

m2 in Sub

m1 in Sub

m2 in Sub

注意：静态方法没有多态。

6. ACDE

B 选项错误，非静态方法中可以调用静态方法

F 选项错误，this 代表当前实例，为对象。静态方法中不能访问直接对象。

7. C

I. 这个程序选择了在构造方法时对 final 属性赋值。在原代码中，如果创建对象时调用了无参构造方法，在整个创建对象的过程中都没有为 final 属性赋值，这样会造成编译错误。

II. 为了保证创建对象时，无论调用哪一个构造方法，final 属性都会被正确赋值，要求必须在每个重载的构造函数中，都加上对 final 属性赋值的语句。

8. A

I. //1, //2 均正确，因为在 printValue 方法中，没有修改 final 的形参。

II. //3, //4 均正确，因为在 changeValue 中，修改的是形参的值，而没有涉及到实参。因此不过实参是否是 final 的，都不影响形参能否被修改。

9. C

final 修饰引用类型，表示的是引用指向的对象不能改变。

在本题中，final 修饰 mv 引用。mv = new MyValue(); 表示让 mv 引用指向一 个 MyValue 对象。这样就对 mv 引用进行了一次赋值，之后，mv 引用就不能被改变。要注意的是，所谓 mv 引用不能改变，指的是 mv 引用中保存的地址不能改变， 也就是说，mv 引用不能指向别的对象。然而，mv 所指 向的对象，其属性是可 以修改的。因此，对于本题来说

final MyValuemv = new MyValue(); mv.value = 100;

//1

System.out.println(mv.value);

在//1 处写上 mv.value = 200 可以编译通过，因为这修改了 mv 引用所指向对象 的属性,而不是让 mv 指向别的对象。如果在//1 处写上 mv = new MyValue() 则不能编译通过，因为这修改了 mv 引用的值，让 mv 引用指向了不同的对象。

10.可以编译通过，输出结果：

m1() in Super

m1(int) in Sub

m1(double) in Sub

注意：父类有 m1 方法，并且是 final 的，子类也有 m1 方法，但是子类的 m1 方法和父类的 m1 方法不构成方法覆盖。

11.BC

B 选项：抽象方法不能有方法体；

C 选项：子类的覆盖方法访问权限修饰符相同或更宽。

12.ABCD

13.DF

A 错误，abstract 不能与 final 连用，abstract 方法必须被子类覆盖，而 final 方法不能被覆盖，矛盾。

B 错误，应该写成 public final void ...

C 错误，abstract 不能与 static 连用。abstract 方法被子类覆盖 之后，会多态的进行调用，而 static 方法没有多态，矛盾。

E 错误，abstract 不能与 private 连用。private 方法不能被继承，因此子类就无法覆盖父类的 private 方法。而 abstract 方法要求一定要被子类覆盖，矛盾。特别的，abstract 方法的访问修饰符也不能是(default)，abstract 修饰方法时，只能与访问 修饰符 public 或 protected 连用。

14.参考：Shape.java

15.参考：MyClass.java