

## Chp7 三个修饰符

### Key Point

- static
- final
- abstract

### 练习

1. (static 属性) 有如下代码

```
class MyClass{
    static int a;
    int b;
}

public class TestMain{
    public static void main(String args[]) {
        MyClass mc1 = new MyClass();
        MyClass mc2 = new MyClass();
        mc1.a = 100;
        mc1.b = 200;
        mc2.a = 300;
        mc2.b = 400;
        System.out.println(mc1.a);
        System.out.println(mc1.b);
        System.out.println(mc2.a);
        System.out.println(mc2.b);
    }
}
```

请写出程序输出结果。

2. (静态成员) 有如下代码

```
class MyClass {
    int a;
    static int b;
    void fa() {}
    static void fb() {}
    public void m1() {
        System.out.println(a); //1
        System.out.println(b); //2

        fa(); //3
        fb(); //4
    }
    public static void m2() {
        System.out.println(a); //5
    }
}
```

```

        System.out.println(b); //6
        fa(); //7
        fb(); //8
    }
}

```

请问哪些行会编译出错？

- A. //1
- B. //2
- C. //3
- D. //4
- E. //5
- F. //6
- G. //7
- H. //8

3. （静态属性）有如下代码

```

class MyClass {
    static int count = 0;
    public MyClass() {
        count++;
        System.out.println(count);
    }
}

public class TestMain{
    public static void main(String args[]) {
        MyClass mc1 = new MyClass();
        MyClass mc2 = new MyClass();
        MyClass mc3 = new MyClass();
    }
}

```

请写出该程序运行时输出的结果。

4. \*（静态初始化代码块）有如下代码

```

class MyClass{
    static int i = 10;
    static {
        i = 20;
        System.out.println("In Static");
    }
}

```

```

        public MyClass() {
            System.out.println("MyClass()");
        }
        public MyClass(int i) {
            System.out.println("MyClass(int)");
            this.i = i;
        }
    }
}
public class TestMain{
    public static void main(String args[]) {
        MyClass mc1 = new MyClass();
        System.out.println(mc1.i);
        MyClass mc2 = new MyClass(10);
        System.out.println(mc2.i);
    }
}

```

请写出该程序运行的结果

5. （静态方法）有以下代码

```

class Super{
    public static void m1() {
        System.out.println("m1 in Super");
    }
    public void m2() {
        System.out.println("m2 in Super");
    }
}
class Sub extends Super{
    public static void m1() {
        System.out.println("m1 in Sub");
    }
    public void m2() {
        System.out.println("m2 in Sub");
    }
}
public class TestMain{
    public static void main(String args[]) {
        Super sup = new Sub();
        sup.m1();
        sup.m2();
        Sub sub = (Sub) sup;
        sub.m1();
        sub.m2();
    }
}

```

```
}
```

写出这个程序的运行结果。

6. (static) \*以下哪些论述是正确的
- A. 静态方法中不能调用非静态方法
  - B. 非静态方法中不能调用静态方法
  - C. 静态方法不能被覆盖
  - D. 静态方法能够用类名直接调用
  - E. 可以在不产生任何一个对象的情况下调用静态方法
  - F. 静态方法里可以使用 this

7. (final 属性的初始化) \*有如下代码

```
1) class MyClass{
2)     final int value;
3)     public MyClass() {}
4)     public MyClass(int value) {
5)         this.value = value;
6)     }
7) }
8) public class TestMain{
9)     public static void main(String args[]) {
10)         MyClass mc = new MyClass(10);
11)         System.out.println(mc.value);
12)     }
13)}
```

选择正确答案:

- A. 编译通过, 输出 10
- B. 编译不通过, 把第 2 行改为 final int value = 10;
- C. 编译不通过, 把第 3 行改为 public MyClass() { value = 10; }

8. (final 变量) \*有如下代码

```
class MyClass {
    public void printValue(final int value) {
        System.out.println(value);
    }
    public void changeValue(int value) {
        value = value * 2;
        System.out.println(value);
    }
}

public class TestMain{
    public static void main(String args[]) {
        MyClass mc = new MyClass();
    }
}
```

```

        int value = 5;
        final int fvalue = 10;
        mc.printValue(value); //1
        mc.printValue(fvalue); //2
        mc.changeValue(value); //3
        mc.changeValue(fvalue); //4
    }
}

```

选择正确答案

- A. 编译通过
- B. //1 出错
- C. //2 出错
- D. //3 出错
- E. //4 出错

9. (final 修饰引用) \*有如下代码

```

class MyValue{
    int value;
}

public class TestFinal{
    public static void main(String args[]){
        final MyValue mv = new MyValue();
        mv.value = 100;
        //1
        System.out.println(mv.value);
    }
}

```

下面说法正确的是:

- A. 编译不通过
- B. 编译通过。在//1 处加上: mv.value = 200; 则编译不通过
- C. 编译通过。如果在//1 处加上: mv = new MyValue(); 则编译不通过。

10. (final 方法, 方法覆盖) 有如下代码

```

class Super{
    public final void m1(){
        System.out.println("m1() in Super");
    }
    public void m1(int i){
        System.out.println("m1(int) in Super");
    }
}

class Sub extends Super{
    public void m1(int i){
        System.out.println("m1(int) in Sub");
    }
}

```

```

    }
    public void m1(double d) {
        System.out.println("m1(double) in Sub");
    }
}
public class TestMain{
    public static void main(String args[]) {
        Sub s = new Sub();
        s.m1();
        s.m1(10);
        s.m1(1.5);
    }
}

```

以上程序是否能编译通过？如果可以，输出运行的结果；如果不可以，应该怎样修改？

11. (abstract, 方法覆盖) \*有以下代码

```

abstract class MyAbstractClass{
    public abstract void m1(); //1
    abstract protected void m2() {} //2
}
class MySubClass extends MyAbstractClass{
    void m1() {} //3
    protected void m2() {} //4
}

```

问：这段代码哪些地方有错误？

- A. //1
  - B. //2
  - C. //3
  - D. //4
12. (abstract) 关于 abstract, 以下选项正确的是:
- A. abstract 类中可以没有 abstract 方法
  - B. abstract 类的子类也可以是 abstract 类
  - C. abstract 类不能创建对象, 但可以声明引用
  - D. abstract 方法不能有方法体
13. (修饰符综合) \*下列方法声明中正确的是:
- A. abstract final void m()
  - B. public void final m()
  - C. static abstract void m()
  - D. private final void m()
  - E. private abstract void m()
  - F. public static final void m()

14. (abstract) 把 Chp6 中的 Shape 类改为抽象类，并把其中的求周长和求面积的方法改为抽象方法。
15. (abstract) \*把 Chp6 中的 Role 类改为抽象类，并把其中的 attack 方法改为抽象方法。
16. (static) \*设计一个类 MyClass，为 MyClass 增加一个 count 属性，用来统计总共创建了多少个对象。
17. (static, final) \*\*修改 Chp6 中的 Account 类，增加两个功能
  - a) Id 属性做成 final 的
  - b) 自动分配 id 的功能。当创建多个 Account 对象时，要求 a) Id 自动分配，每个 Account 对象的 id 都不重复；b) 自动分配的 id 从 100001 开始。

例如：

```
Account a1 = new Account();
Account a2 = new Account();
System.out.println(a1.getId()); //输出 100001
System.out.println(a2.getId()); //输出 100002
```

18. (静态初始化代码块，对象创建的过程) \*\*\*有以下代码

```
class ClassA{
    static {
        System.out.println("In ClassA Static");
    }
    public ClassA() {
        System.out.println("ClassA()");
    }
}

class ClassB{
    static {
        System.out.println("In ClassB Static");
    }
    public ClassB() {
        System.out.println("ClassB()");
    }
}

class ClassC extends ClassB{
    static{
        System.out.println("In ClassC Static");
    }
    public ClassC() {
        System.out.println("ClassC()");
    }
}
```

```

    }
}
class MyClass {
    static ClassA ca = new ClassA();
    ClassC cc = new ClassC();
    static{
        System.out.println("In MyClass Static");
    }
    public MyClass() {
        System.out.println("MyClass()");
    }
}
public class TestMain{
    public static void main(String args[]) {
        MyClass mc1 = new MyClass();
        MyClass mc2 = new MyClass();
        System.out.println(mc1.cc == mc2.cc);
        System.out.println(mc1.ca == mc2.ca);
    }
}

```

写出这个程序运行的结果。



