Day08

- static
  - 静态变量
  - 静态方法
  - 静态代码块
- final
  - 常量
  - 最终方法
  - 最终类
- abstract
  - 抽象类
  - 抽象方法

一、选择题

1. 对于代码：

```
class A {
    private static void m(){
    }
}
public class B extends A {
    // 1
}
```

下列各项中添加到 // 1处不会出错的是：

A.

```
public static int m(){
return 0;
}
```

B.

```
private void m() throws Exception{
}
```

C.

```
void m(){
}
```

D.

```
public final double m(double i){
return i *= 3;
}
```

E.

```
public abstract void m();
```

2. 对于代码：

```
package cn.tedu.test1;
public class A {
    int i;  // 1
    protected void ma(){
        System.out.println("A ma");
    }
}
package cn.tedu.test2;
import cn.tedu.test1.A;
public class B extends A {
    public void mb(){
        super.i = 5;    // 2
    }
}
package cn.tedu.test2;
public class C {
    public static void main(String[] args){
        A a = new A();
        a.ma(); // 3
        B b = new B();
        b.mb(); // 4
    }
}
```

会在标注处出错的是：

A. // 1　**B.** // 2　**C.** // 3　**D.** // 4　**E.** 标注处无错

3. 对于代码：

```
public class Test {
    final int i; // 1
    public Test(){} // 2
    public Test(int i){
        this.i = i; // 3
    }
}
```

下列说法错误的是：

A. 编译通过

B. 编译报错，如果将// 1处改为final int i = 5; 则编译通过

C. 编译报错，如果将// 2处改为public Test(){i = 10;}则编译通过

D. 编译报错，如果将// 3处改为this.i = 10;则编译通过

E. 编译报错，如果在// 1处添加{i = 5;}则编译通过

4. 对于代码：

```
class TestSuper {
```

```
    public void m1(){
        m2();
    }
    public void m2(){
        System.out.println("Super");
    }
}
class TestSub extends TestSuper {
    public void m2(){
        System.out.println("Sub");
    }
}
public class Test {
    public static void main(String[] args){
        TestSuper ts = new TestSub();
        ts.m1();
    }
}
```

的运行结果是：

A. Super **B.** Sub **C.** Super Sub **D.** Sub Super **E.** 编译报错

5. 对于代码：

```
class TestSuper {
    public static void ma(){
        System.out.println("Super ma");
    }
    public void mb(){
        System.out.println("Super mb");
    }
}
class TestSub extends TestSuper {
    public static void ma(){
        System.out.println("Sub ma");
    }
    public void mb(){
        System.out.println("Sub mb");
    }
}
public class Test {
    public static void main(Stirng[] args){
        TestSuper ts = new TestSub();
        ts.ma();
        ts.mb();
    }
}
```

的运行结果是：

A. Super ma Super mb **B.** Super ma Sub mb **C.** Sub ma Super mb

D. Sub ma Sub mb **E.** 编译报错

6. 子类A继承了父类B，A a = new B();则①父类B静态代码块②父类B非静态代码块③父类B构造函数④子类A静态代码块⑤子类A非静态代码块⑥子类A构造函数的执行顺序是：

   A. ①->②->③->④->⑤->⑥

   B. ①->④->②->③->⑤->⑥

   C. ①->②->④->⑤->③->⑥

   D. ④->⑤->①->②->③->⑥

   E. ④->⑤->⑥->①->②->③

7. 对于代码：

```
class TestSuper {
    private void m(){}
}
public class TestSub extends TestSuper {
    // 1
}
```

   下列各项中可以填入//1位置处的是：

   A. public void m(){}

   B. public int m(){return 0;}

   C. public int m(int i){return i;}

   D. private double m(double d){return d;}

   E. 以上各项都不行

8. 对于代码：

```
public class Test {
    int a;
    static int b;
    void fa(){}
    static void fb(){}
    public static void m2(){
        System.out.println(a); //1
        System.out.println(b); //2
        fa(); //3
        fb(); //4
    }
}
```

   会报错的位置是：

   A. //1　**B.** //2　**C.** //3　**D.** //4　**E.** 标注处无错

9. 下列说法中正确的是：

   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

10. 下列关于super的说法正确的是：

    A. super代表当前对象的引用

    B. 可以通过super语句来实现构造函数的调用

    C. 可以通过super来调用私有的方法或者私有的属性

    D. 可以在主函数中使用super来调用其他的方法

    E. 可以使用super关键字来调用静态属性

二、简答题

1. 有如下代码

```
class MyClass{
    static int a;
    int b;
}
public class Test{
    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. 有如下代码

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

3. 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 Test {
    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

4. 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 Test {
    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？

5. 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 Test{
    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。

三、 编程题

1. 设计一个类MyClass，为MyClass增加一个count属性，用于统计MyClass类一共产生了多少个对象。