## 第12章 反射

一、填空题

1．反射机制的优点是可以实现\_\_\_\_\_\_\_\_创建对象和编译。

2．如果想通过Class类实例化其他类的对象，则可以使用\_\_\_\_\_\_\_\_方法，但是必须要保证被实例化的类中存在一个无参构造方法。

3．通过反射可以得到一个类的完整结构，包括类的构造方法、类的属性、类的方法，这就需要使用到java.lang.reflect包中的以下几个类，分别是\_\_\_\_\_\_\_\_、\_\_\_\_\_\_\_\_和\_\_\_\_\_\_\_\_。

4．要取得一个类所实现的全部接口，必须使用Class中的\_\_\_\_\_\_\_\_方法。

5．JVM在加载.class文件时，会产生一个\_\_\_\_\_\_\_\_对象代表该.class字节码文件，从该Class对象中可以获得类的信息。

二、判断题

1．将Class对象实例化为本类对象时，可以通过无参构造完成，也可以通过有参构造完成。

2．要取得一个类中的全部方法，可以使用Class类中的getMethods()方法。

3．在反射机制中，把类中的成员（构造方法、成员方法、成员变量）都封装成了对应的类进行表示。

4．Class类的对象用于表示当前运行的 Java 应用程序中的类和接口，Class类是一个未继承Object类的特殊类。

5．JAVA 反射机制是在运行状态中，对于任意一个类，都能够知道这个类的所有属性和方法；对于任意一个对象，都能够调用它的任意一个方法；这种动态获取的信息以及动态调用对象的方法的功能称为java 语言的反射机制。

三、选择题

1.如何获取一个类的字节码文件对象。（）

A、对象名.class

B、类名.getClass()

C、Object类中的forName()方法

D、以上说法都不正确

2.关于反射机制下列说法错误的是（）

A、反射可以获取类中所有的属性和方法

B、反射可以构造类的对象，并获取其私有属性的值

C、反射机制指的是在程序编译期间，通过.class文件加载并使用一个类的过程

D、暴力反射可以获取类中私有的属性和方法

3.下列关于通过反射方式获取方法并执行的过程说法正确的是 （ ）

A、通过对象名.方法名(参数列表)的方式调用该方法

B、通过Class.getMethod(方法名，参数类型列表)的方式获取该方法

C、通过Class.getDeclaredMethod(方法名，参数类型列表)获取私有方法

D、通过invoke(对象名,参数列表)方法来执行一个方法

4. 以下哪些方法在Class类中定义？（ ）（多选）

A．getConstructors()

B．getPrivateMethods()

C．getDeclaredFields()

D．getImports()

E．setField()

5．假定Tester 类有如下test()方法：

public int test(int p1, Integer p2)

以下哪段代码能正确地动态调用一个Tester 对象的test方法？（）

A、Class classType=Tester.class;

Object tester=classType.newInstance();

Method addMethod=classType.getMethod("test",new Class[]{int.class,int.class});

Object result=addMethod.invoke(tester,new Object[]{new Integer(100),new Integer(200)});

B、Class classType=Tester.class;

Object tester=classType.newInstance();

Method addMethod=classType.getMethod("test",new Class[]{int.class,int.class});

int result=addMethod.invoke(tester,new Object[]{new Integer(100),new Integer(200)});

C、Class classType=Tester.class;

Object tester=classType.newInstance();

Method addMethod=classType.getMethod("test",new Class[]{int.class,Integer.class});

Object result=addMethod.invoke(tester,new Object[]{new Integer(100),new Integer(200)});

D、Class classType=Tester.class;

Object tester=classType.newInstance();

Method addMethod=classType.getMethod("test",new Class[]{int.class,Integer.class});

Integer result=addMethod.invoke(tester,new Object[]{new Integer(100),new Integer(200)});

四、简答题

1．简述一下反射机制。

2．简述实例化Class对象的三种方式。