1) **In the below Class X, is 'method' properly overloaded?**

```
class X
{
    int method(int i, int d)
    {
        return i+d;
    }

    static int method(int i, double d)
    {
        return (int)(i+d);
    }

    double method(double i, int d)
    {
        return i+d;
    }

    static double method(double i, double d)
    {
        return i+d;
    }
}
```

2) **What will be the output of the following program?**

```
class X
{
    void method(int a)
    {
        System.out.println("ONE");
    }

    void method(double d)
    {
        System.out.println("TWO");
    }
}

class Y extends X
{
    @Override
    void method(double d)
    {
        System.out.println("THREE");
    }
}

public class MainClass
{
    public static void main(String[] args)
    {
        new Y().method(100);
    }
}
```

3) Will you find out the error in the below code?

```
class X
{
    static void methodOfX()
    {
        System.out.println("Class X");
    }
}

class Y extends X
{
    @Override
    static void methodOfX()
    {
        System.out.println("Class X");
    }
}
```

4) What will be the output of the following program?

```
class ClassOne
{
    void method(String s1)
    {
        method(s1, s1+s1);
    }

    void method(String s1, String s2)
    {
        method(s1, s2, s1+s2);
    }

    void method(String s1, String s2, String s3)
    {
        System.out.println(s1+s2+s3);
    }
}

public class MainClass
{
    public static void main(String[] args)
    {
        ClassOne one = new ClassOne();

        one.method("JAVA");
    }
}
```

5) What will be the outcome of the below program?

```
public class MainClass
{
    double overloadedMethod(double d)
    {
        return d *= d;
    }

    int overloadedMethod(int i)
    {
        return overloadedMethod(i *= i);
    }

    float overloadedMethod(float f)
    {
        return overloadedMethod(f *= f);
    }

    public static void main(String[] args)
    {
        MainClass main = new MainClass();

        System.out.println(main.overloadedMethod(100));
    }
}
```

6) Can we override protected method as private?