FPP Quiz 0

1.
```
class MyClass{
   System.out.println("hello");
}
```

When you compile/run this program the result is:

a. Outputs hello to the console
b. Compiler error
c. Runtime exception

2.
```
class MyClass {
   public static void main(String[] args) {
      myMethod();
   }

   public void myMethod() {
      System.out.println("hello");
   }
}
```

When you compile/run this program the result is:

a. Outputs `"hello"` to the console
b. Compiler error
c. Runtime exception

3.
```
class MyClass {
   public static void main(String[] args) {
      MyClass m = new MyClass();
      m.myMethod();
   }
   private void myMethod() {
      System.out.println("hello");
   }
}
```

When you compile/run this program the result is:

a. Outputs `"hello"` to the console
b. Compiler error
c. Runtime exception

4.
```
class MyClass {
   public static void main(String[] args) {
      AnotherClass a = new AnotherClass(new MyClass());

   }
   private void myMethod() {
```

```
      System.out.println("hello");
   }
}

class AnotherClass {
   AnotherClass(MyClass m) {
      m.myMethod();
   }
}
```

When you compile/run this program the result is:

a. Outputs `"hello"` to the console
b. Compiler error
c. Runtime exception

5.
```
class MyClass {
   public static void main(String[] args) {
      AnotherClass a = new AnotherClass(new MyClass());
   }
   private void myMethod() {
      System.out.println("hello");
   }
}

class AnotherClass {
   AnotherClass(MyClass m) {
      myMethod();
   }
}
```

When you compile/run this program the result is:

a. Outputs `"hello"` to the console
b. Compiler error
c. Runtime exception

6.
```
class MyClass {
   public static void main(String[] args) {
      AnotherClass a = new AnotherClass(new MyClass());
      a.anotherMethod();
   }
   void myMethod() {
      System.out.println("hello");
   }
}

class AnotherClass {
   MyClass m;
   AnotherClass(MyClass m) {
```

```
      this.m = m;
      anotherMethod();
   }
   void anotherMethod() {
      m.myMethod();
   }

}
```

When you compile/run this program the result is:

a. Outputs `"hello"` to the console
b. Outputs `"hello"` *twice* to the console
c. Compiler error
d. Runtime exception

7.
```
class MyClass {
   public static void main(String[] args) {
      AnotherClass a = new AnotherClass(new MyClass());
      a.anotherMethod();
   }
   void myMethod() {
      System.out.println("hello");
      a.anotherMethod();
   }
}

class AnotherClass {
   MyClass m;
   AnotherClass(MyClass m) {
      this.m = m;
   }
   void anotherMethod() {
      System.out.println("hello");
      m.myMethod();
   }

}
```

When you compile/run this program the result is:

a. Continuously outputs `"hello"` to the console
b. Compiler error
c. Runtime exception

8.

```
class MyClass2 {
   AnotherClass a;

   public static void main(String[] args) {
      int n = 0;
      if(args[0] != null) {
         n = Integer.parseInt(args[0]);
      }
      MyClass2 m = new MyClass2();
      m.a = new AnotherClass(m);
      m.a.anotherMethod(n);
   }
   void myMethod(int k) {
      if(k == 0 || k == 1) {
         System.out.println("hello");
         return;
      }
      a.anotherMethod(--k);
   }
}

class AnotherClass {
   MyClass2 m;
   AnotherClass(MyClass2 m) {
      this.m = m;
   }
   void anotherMethod(int k){
      m.myMethod(--k);
   }

}
```

When you run this program like this:

```
java MyClass2 n
```

where n is any positive integer, the result is:

a. Outputs `"hello"` to the console
b. Stack overflow exception
c. Code will not compile because of a "cyclic reference"

9. When the main method below is run (to the right), what happens?
a. "Times through the loop: 0" is printed to the console
b. "Times through the loop:1" is printed to the console.
c. "Times through the loop: n" is printed to the console for some n > 1.
d. The while loop does not terminate.

```
public class Test {
    public static void main(String[] args) {
        Test t = new Test();
        t.run();
    }
    public void run() {
        int loopcount = 0;
        double scalar = 0.1;
        double x = 0.9;
        double y = 0.7;
        while (x - scalar != y + scalar) {
            x *= scalar;
            y *= scalar;
            scalar *= scalar;
            ++loopcount;
        }
        System.out.println("Times through the loop: "
            + loopcount);
    }
}
```