

1. Look at the following code:

```
1  public class TestClass {
2
3      // section 1:
4      private String testName;
5
6      // section 2:
7      public TestClass( String name, int i ) {
8          this.testName = name;
9      }
10
11     // section 3:
12     public void countToThree() {
13         for (int m = 1; m <= 3; m++) {
14             System.out.println( "Count is: " + m );
15         }
16     }
17 }
```

What is defined in the denoted sections of this class?

**What is defined in the denoted sections of this class?**

☐ **section 1:** member variable

**section 2:** constructor

**section 3:** class method

☒ **section 1:** member variable

**section 2:** class method

**section 3:** method

☐ **section 1:** method

**section 2:** constructor

**section 3:** member variable

☒ **section 1:** member variable

**section 2:** constructor

**section 3:** method

---

2. As an established Java convention, what would it mean if the name of a variable was spelled in all uppercase?

☐ Nothing. There is no such convention, and such a variable is like any other.

☒ The variable is a constant, whose value should not change.

☐ The variable contains a string that has all capital letters.

☐ The variable is reserved for use by the Java environment, and you should not refer to it.

3. Look at the following code:

```
1  int errorInteger = 200;
2  String comment;
3
4  switch (errorInteger) {
5      case 150:
6          comment = "Javascript error.";
7          break;
8      case 240:
9          comment = "Comment error.";
10         break;
11     case 300:
12         comment = "Function error.";
13         break;
14     case 200:
15         comment = "New error.";
16         break;
17     default:
18         comment = "No error.";
19         break;
20 }
21 System.out.println( comment );
22
```

What would be the resulting output from this code?

- ☐ Comment error.
- ☒ New error.
- ☐ Javascript error.
- ☐ Function error.

4. Look at the following class:


```
1 public class Test {  
2     private String testName;  
3  
4     public Test( String name ) {  
5         this.testName = name;  
6     }  
7  
8     public setTestName( String name ) {  
9         this.testName = name;  
10    }  
11 }
```

What would be the proper way to construct a Test object with member variable testName initially being "old", then later changed to "new"

- ☐

```
1 Test testName = "old";  
2 testName = "new";
```
- ☐

```
1 Test testObj = new Test( "old" );  
2 testObj.testName = "new";
```
- ☐

```
1 Test testObj = new Test( "old" );  
2 testObj[testName] = "new";
```
-  ☒

```
1 Test testObj = new Test( "old" );  
2 testObj.setTestName( "new" );
```