1.Look at the following code:

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

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

What would be the resulting output from this code?

Comment error.



- Javascript error.
- Function error.

4.Look at the following class:

```
public class Test {
    private String testName;

public Test( String name ) {
    this.testName = name;
    }

public setTestName( String name ) {
    this.testName = name;
    }

public setTestName = name;
}

public setTestName = name;
}
```

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" );
```