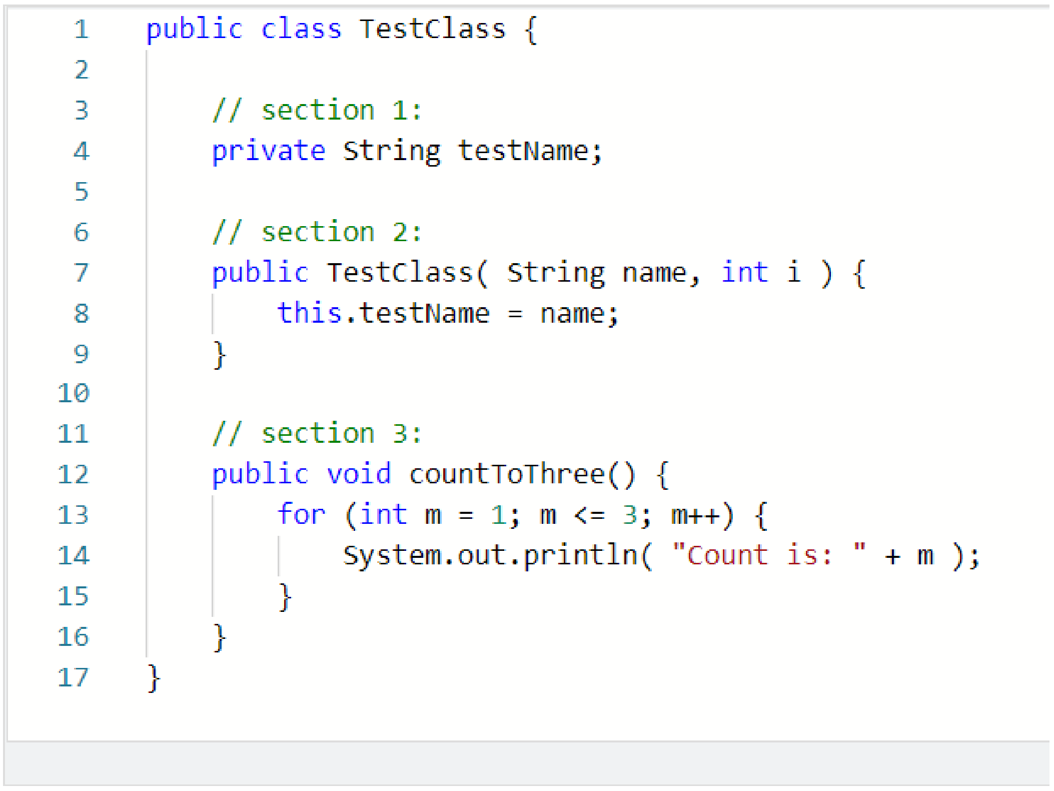
1.Look at the following code:



```java
public class TestClass {

    // section 1:
    private String testName;

    // section 2:
    public TestClass( String name, int i ) {
        this.testName = name;
    }

    // section 3:
    public void countToThree() {
        for (int m = 1; m <= 3; m++) {
            System.out.println( "Count is: " + m );
        }
    }
}
```

What is defined in the denoted sections of this class?

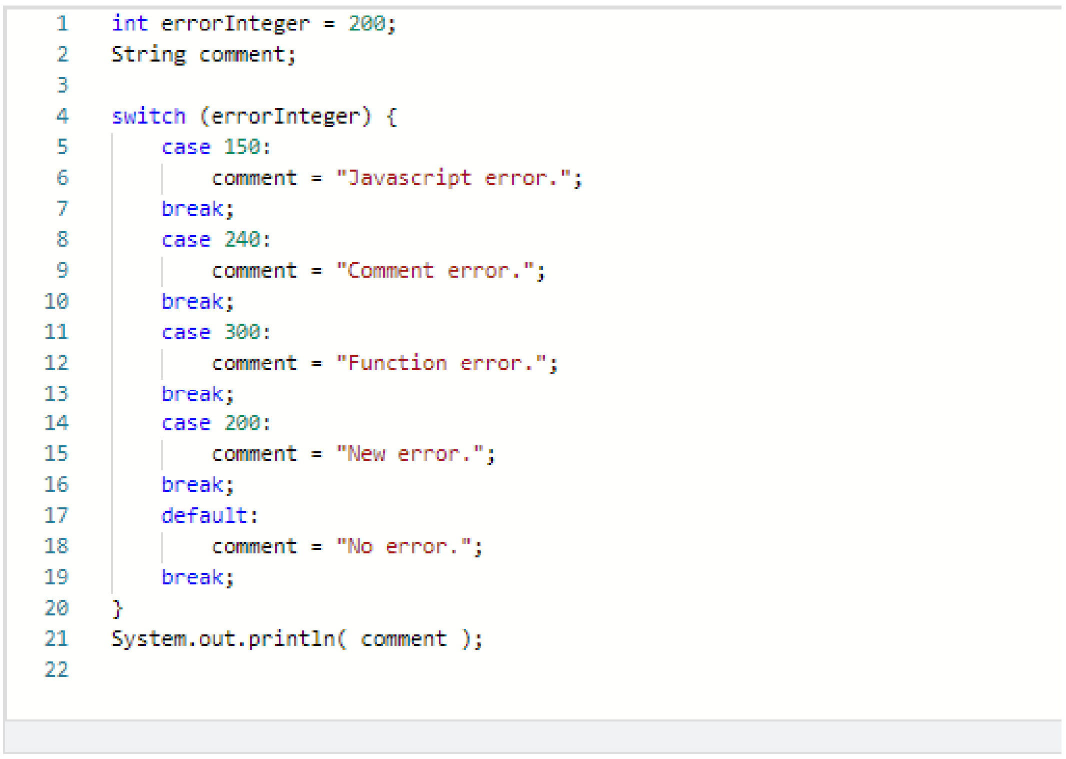
What is defined in the denoted sections of this class?

- ◯ **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 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;
String comment;

switch (errorInteger) {
    case 150:
        comment = "Javascript error.";
    break;
    case 240:
        comment = "Comment error.";
    break;
    case 300:
        comment = "Function error.";
    break;
    case 200:
        comment = "New error.";
    break;
    default:
        comment = "No error.";
    break;
}
System.out.println( comment );
```

What would be the resulting output from this code?

- ○ Comment error.
- [✓ New error.]
- ○ Javascript error.
- ○ Function error.

4.Look at the following class:

```java
public class Test {
    private String testName;

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

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

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

- ○
```java
Test testName = "old";
testName = "new";
```

- ○
```java
Test testObj = new Test( "old" );
testObj.testName = "new";
```

- ○
```java
Test testObj = new Test( "old" );
testObj[testName] = "new";
```

- [✓]
```java
Test testObj = new Test( "old" );
testObj.setTestName( "new" );
```