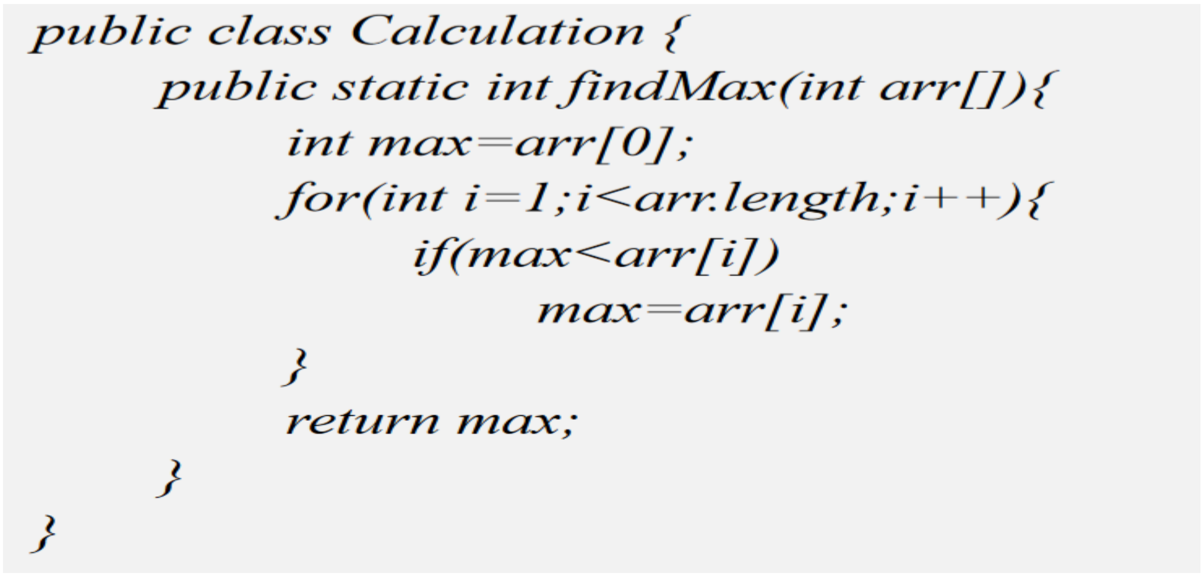
CPT203 Software Engineering 1

Tutorial: testing 2

Suggested answers

Q1. Suppose you have the following class which you want to test:

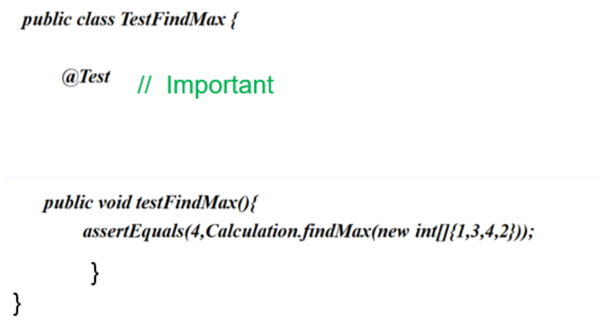


```
public class Calculation {
        public static int findMax(int arr[]){
                int max=arr[0];
                for(int i=1;i<arr.length;i++){
                        if(max<arr[i])
                                max=arr[i];
                }
                return max;
        }
}
```

Here findMax() is a function that accepts an array of integers as its input parameter. It returns the largest value among the values in the input array. Write a Junit unit test for the findMax() function.

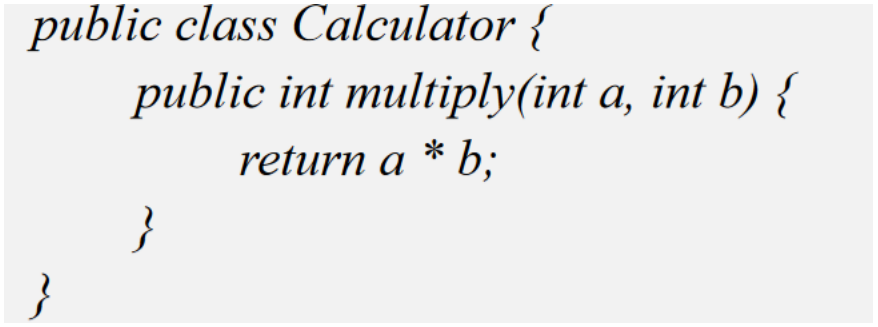
The requirement is: Use an array with values 1, 3, 4, 2 as the test case. Here the date type of 1 is int, of 3 is int, of 4 is int, and of 2 is int.

Example answer:



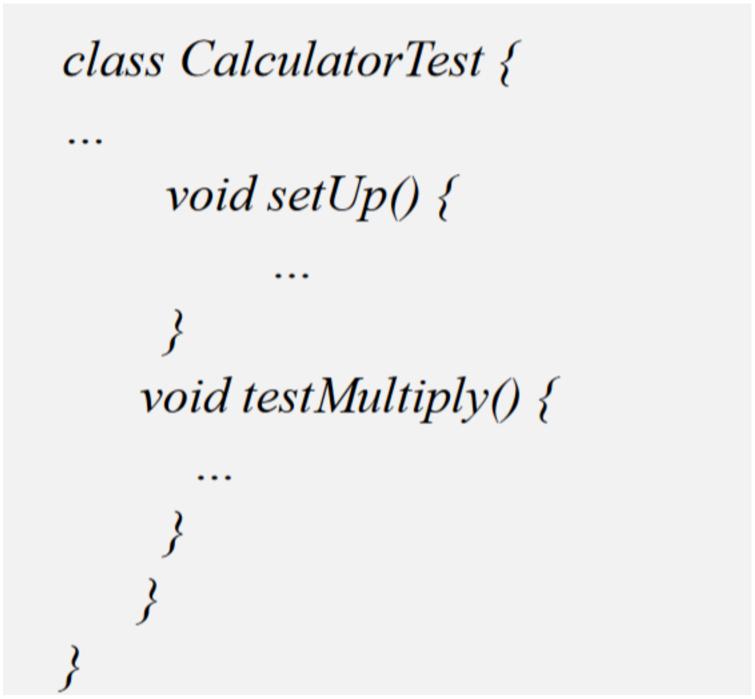
```
public class TestFindMax {

    @Test   // Important


    public void testFindMax(){
        assertEquals(4,Calculation.findMax(new int[]{1,3,4,2}));
    }
}
```

Q2. Suppose you have the following Calculator class which you want to test.



```
public class Calculator {
        public int multiply(int a, int b) {
                return a * b;
        }
}
```

Fill up the following test class to complete the testing for the above Calculator class.



```
class CalculatorTest {
...
        void setUp() {
                ...
        }
      void testMultiply() {
          ...
        }
      }
}
```

Additional requirements are:
• Use 4* 5 = 20 as the test case
• The test should run 5 times
• to import only the junit packages that are needed for this question.

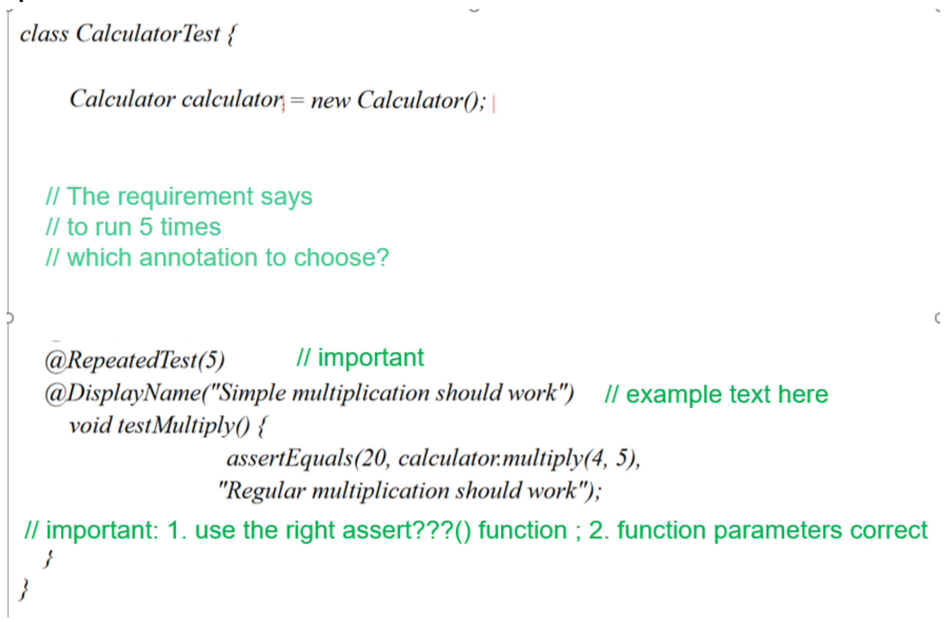
Example answers:

1. To import junit packages for using :
   assertEquals()
   @RepeatedTest
   @DisplayName

```
 import org.junit.jupiter.api.DisplayName;
 import org.junit.jupiter.api.RepeatedTest;
import static org.junit.jupiter.api.Assertions.assertEquals; // lecture page 23
```

(https://junit.org/junit5/docs/current/user-guide/ Section 2. Writing tests, Section 2.4 display names, and Section 2.16 repeated tests)

2. To complete the class



```
class CalculatorTest {

    Calculator calculator = new Calculator();


  // The requirement says
  // to run 5 times
  // which annotation to choose?


  @RepeatedTest(5)          // important
  @DisplayName("Simple multiplication should work")    // example text here
    void testMultiply() {
                    assertEquals(20, calculator.multiply(4, 5),
                   "Regular multiplication should work");
// important: 1. use the right assert???() function ; 2. function parameters correct
  }
}
```