SSC0130 - Trabalho 3

- **Gustavo Siqueira Barbosa - 10728122**
- **Luiz Fernando Santos - 10892680**

1. (ENADE 2011) Uma equipe está realizando testes com o código-fonte de um sistema. Os testes envolvem a verificação de diversos componentes individualmente, bem como das interfaces entre eles. Essa equipe está realizando testes de:

- **a) unidade**
- **b) aceitação**
- **c) sistema e aceitação**
- **d) integração e sistema**
- **e) unidade e integração**

Resposta:

- Letra **(e) unidade e integração**.

4. Reescreva o seguinte teste, que verifica o levantamento de uma exceção EmptyStackException, para que ele fique mais simples e fácil de entender.

```
@Test
public void testEmptyStackException() {
  boolean sucesso = false;
  try {
    Stack s<Integer> = new Stack<Integer>();
    s.push(10);
    int r = stack.pop();
    r = stack.pop();
  } catch (EmptyStackException e) {
    sucesso = true;
  }
  assertTrue(sucesso);
}
```

Resposta:

- Podemos simplificar o teste sem inserir um elemento na lista antes.

```
@Test(expected = java.util.EmptyStackException.class)
public void testEmptyStackException() {
    Stack s<Integer> = new Stack<Integer>();
    s.pop();
}
```

5. Suponha que um programador escreveu o teste a seguir para a classe `ArrayList` de Java. Como você irá perceber, no código são usados diversos `System.out.println`. Ou seja, no fundo, ele é um teste manual, pois o desenvolvedor tem que conferir o seu resultado manualmente. Reescreva então cada um dos testes (de 1 a 6) como um teste de unidade, usando a sintaxe e os comandos do JUnit. Observação: se quiser executar o código, ele está disponível neste link.

```
import java.util.List;
import java.util.ArrayList;

public class Main {

  public static void main(String[] args) {

    // teste 1
    List<Integer> s = new ArrayList<Integer>();
    System.out.println(s.isEmpty());

    // teste 2
    s = new ArrayList<Integer>();
    s.add(1);
    System.out.println(s.isEmpty());

    // teste 3
    s = new ArrayList<Integer>();
    s.add(1);
    s.add(2);
    s.add(3);
    System.out.println(s.size());
    System.out.println(s.get(0));
    System.out.println(s.get(1));
    System.out.println(s.get(2));

    // teste 4
    s = new ArrayList<Integer>();
    s.add(1);
    s.add(2);
    s.add(3);
    int elem = s.remove(2);
    System.out.println(elem);
    System.out.println(s.get(0));
    System.out.println(s.get(1));

    // teste 5
    s = new ArrayList<Integer>();
```

```
    s.add(1);
    s.remove(0);
    System.out.println(s.size());
    System.out.println(s.isEmpty());

    // teste 6
    try {
      s = new ArrayList<Integer>();
      s.add(1);
      s.add(2);
      s.remove(2);
    }

    catch (IndexOutOfBoundsException e) {
      System.out.println("IndexOutOfBound");
    }

  }

}
```

Resposta:

```
import java.util.List;
import java.util.ArrayList;

public class Test {

    @Before
    public void init(){
        s = new ArrayList<Integer>();
    }

    // teste 1
    @Test
    public void testEmptyList() {
        assertTrue(s.isEmpty());
    }

    // teste 2
    @Test
    public void testAddElem() {
        s.add(1);
        assertFalse(s.isEmpty())
    }
```

```
// teste 3
@Test
public void testSizeAndGet() {

    s.add(1);
    s.add(2);
    s.add(3);

    // testa size
    assertEquals(3, s.size());

    // testa get
    assertEquals(1, s.get(0));
    assertEquals(2, s.get(1));
    assertEquals(3, s.get(2));
}

// teste 4
@Test
public void testRemove() {

    s.add(1);
    s.add(2);
    s.add(3);

    assertEquals(3, s.remove(2));
    assertEquals(1, s.get(0));
    assertEquals(2, s.get(1));
}

// teste 5
public void testRemoveAndIsEmpty() {

    s.add(1);
    s.remove(0);

    assertEquals(0, s.size());
    assertTrue(s.isEmpty());
}

// teste 6
@Test(expected = java.util.IndexOutOfBoundsException.class)
public void testExceptionOutOfBounds() {
    s.add(1);
    s.add(2);
    s.remove(2);
```

```
    }
}
```

7. Suponha o seguinte requisito: alunos recebem conceito A em uma disciplina se tiverem nota maior ou igual a 90. Seja então a seguinte função que implementa esse requisito:

```
boolean isConceitoA(int nota) {
  if (nota > 90)
    return true;
  else return false;
}
```

O código dessa função possui três comandos, sendo um deles um if; logo, ela possui dois branches. Responda agora às seguintes perguntas.

- **a. A implementação dessa função possui um bug? Se sim, quando esse bug resulta em falha?**
  - **Resposta:** Sim. O método não cobre quando a nota é igual à 90, retornando `false` quando deveria retornar `true`.
- **b. Suponha que essa função — exatamente como ela está implementada — seja testada com duas notas: 85 e 95. Qual a cobertura de comandos desse teste? E a cobertura de branches?**
  - **Resposta:**
    - *Cobertura de comandos*: 100%
    - *Cobertura de branches*: 100% (as duas branches foram testadas)
- **c. Seja a seguinte afirmação: se um programa possui 100% de cobertura de testes, em nível de comandos, ele está livre de bugs. Ela é verdadeira ou falsa? Justifique.**
  - **Resposta:** Falsa. Testes não podem garantir que o programa está livre de quaisquer bugs, apenas dos bugs previstos e testados. Lembrando sempre: testes de software mostram a presença de bugs, não sua ausência.

8. Complete os comandos assert nos trechos indicados.

```
public void test1() {
   LinkedList list = mock(LinkedList.class);
   when(list.size()).thenReturn(10);
   assertEquals(___________, ___________);
}

public void test2() {
   LinkedList list = mock(LinkedList.class);
   when(list.get(0)).thenReturn("Engenharia");
   when(list.get(1)).thenReturn("Software");
   String result = list.get(0) + " " + list.get(1);
```

```
   assertEquals(__________, __________);
}
```

Resposta:

```
public void test1() {
   LinkedList list = mock(LinkedList.class);
   when(list.size()).thenReturn(10);
   assertEquals(10, list.size());
}

public void test2() {
   LinkedList list = mock(LinkedList.class);
   when(list.get(0)).thenReturn("Engenharia");
   when(list.get(1)).thenReturn("Software");
   String result = list.get(0) + " " + list.get(1);
   assertEquals("Engenharia Software", result);
}
```