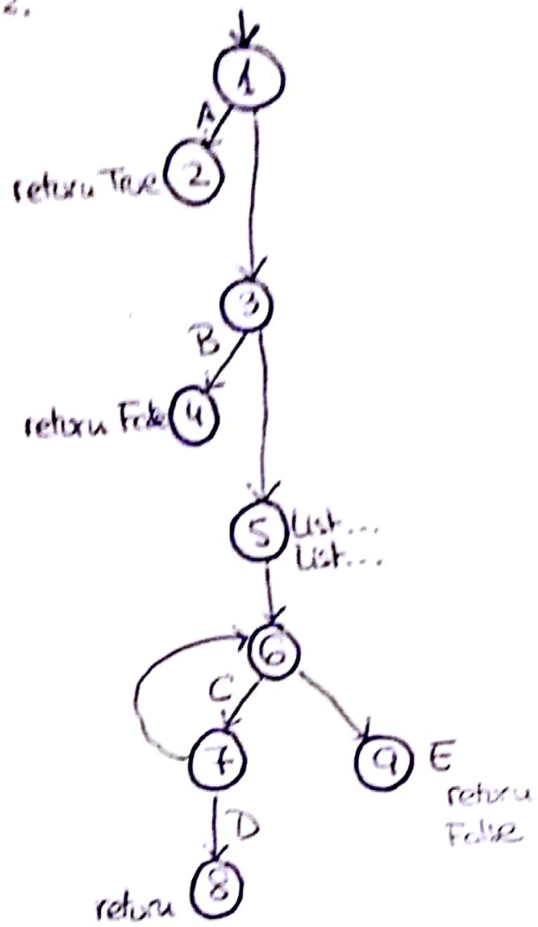
Ejercicio 2

1. equals()
2.



3. Cobertura de Nodos:

R= {1,2,3,4,5,6,7,8,9}

[1,2]
[1,3,4]
[1,3,5,6,9]
[1,3,5,6,7,8]

"Los 4 return":

```
// Test 1er return
List<String> list1 = new ArrayList<String>();
list1.add("foo");
assertTrue(list1.equals(list1));

// Test 2º return
"Ejercicio 4"

// Test 3er return
"Ejercicio 3"

// Test 4º return
List<String> list1 = new ArrayList<String>();
List<String> list2 = new ArrayList<String>();
assertFalse(list1.equals(list2));
```

4. Cobertura arcos

RT = {(1,2), (1,3), (3,4), (3,5), (5,6), (6,5), (6,7), (7,6), (7,8)}

[1,3,4] ⇒ "Ejercicio 4"

[1,2] ⇒ "1er return (apartado anterior)"

[1,3,5,6,9] ⇒ "4º return (apartado anterior)"

[6,7,6]

// Mínimo 1 vez entra en el while

```
List<String> list1 = new ArrayList<String>();
List<String> list2 = new ArrayList<String>();
list1.add("foo");
list1.add("bar");
list2.add("foo");
list2.add("bar");
assumeTrue(list1.equals(list2));
```

5. Cobertura de caminos principales:

RT = {[1,2], [1,3,4], [1,3,5,6,9], [1,3,5,6,7,6,9]

[1,3,5,6,7,8], [6,7,6,7,8]}

6 caminos ya casi todos están correspondidos con un test en los apartados anteriores, excepto:

[6,7,6,7,8] ⇒ Corresponde al test del apartado anterior.

[1,3,5,6,7,6,9] ⇒ // Entra 2 veces en el while y 3er return

```
List<String> list1 = new ArrayList<String>();
List<String> list2 = new ArrayList<String>();
list1.add("foo");
list1.add("bar");
list2.add("foo");
list2.add("ba");
assumeFalse(list1.equals(list2));
```