

Wednesday Sep. 19
Lecture 5

Lab Test I: Oct. I

Slides:

Classes & Objects

Exceptions

JUnit

Monday Sep. 24

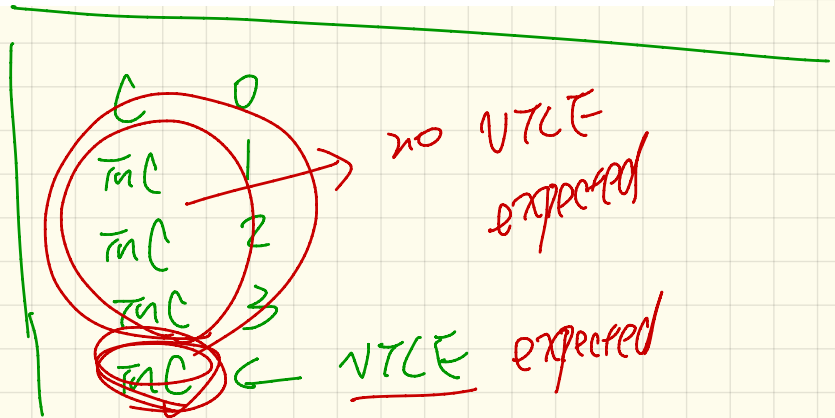
Programming:

- Lab 1 (20 arrays, nested loops)
- Practice Problem

Testing from Console (VI): Test I

```
public class CounterTester1 {  
    public static void main(String[] args) {  
        ✓ Counter c = new Counter();  
        System.out.println("Init val: " + c.getValue());  
        try {  
            c.decrement();  
            System.out.println("ValueTooSmallException NOT thrown as expected.");  
        } catch (ValueTooSmallException e) {  
            System.out.println("ValueTooSmallException thrown as expected.");  
        }  
    }  
}
```

VTSE did not occur



Testing from Console (V1) : Test 2

assume:
correct
imp.

```
public class CounterTester2 {  
    public static void main(String[] args) {  
        Counter c = new Counter();  
        System.out.println("Current val: " + c.getValue());  
        try {  
            c.increment();  
            c.increment();  
            c.increment();  
        } catch (ValueTooLargeException e) {  
            System.out.println("ValueTooLargeException was thrown unexpectedly.");  
        }  
        System.out.println("Current val: " + c.getValue());  
        try {  
            c.increment();  
            System.out.println("ValueTooLargeException was NOT thrown as expected.");  
        } catch (ValueTooLargeException e) {  
            System.out.println("ValueTooLargeException thrown as expected.");  
        }  
    }  
}
```

for (int i=0; i<3; i++)
 c.increment();

we don't expect NICE to occur; but it did.

↓ c.getValue()
3

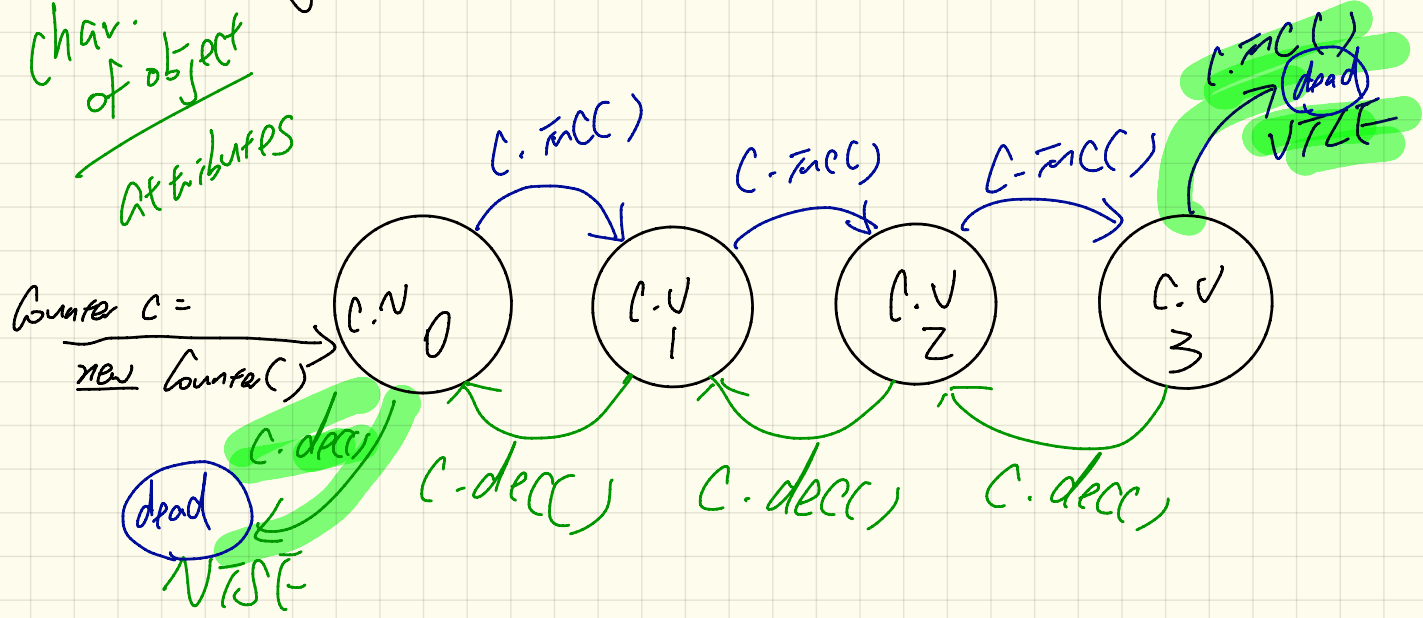
Testing from Console (v2)

```
public class CounterTester3 {  
    public static void main(String[] args) {  
        Scanner input = new Scanner(System.in);  
        String cmd = null;  
        Counter c = new Counter();  
        boolean userWantsToContinue = true;  
        while (userWantsToContinue) {  
            System.out.println("Enter \"inc\", \"dec\", or \"val\":");  
            cmd = input.nextLine();  
            try {  
                if (cmd.equals("inc")) {  
                    c.increment();  
                } else if (cmd.equals("dec")) {  
                    c.decrement();  
                } else if (cmd.equals("val")) {  
                    System.out.println(c.getValue());  
                } else {  
                    userWantsToContinue = false;  
                    System.out.println("Bye!");  
                }  
            } catch (ValueTooLargeException e) {  
                System.out.println("Value too big!");  
            } catch (ValueTooSmallException e) {  
                System.out.println("Value too small!");  
            }  
        }  
        input.close();  
    }  
}
```

VTSE

State Diagram for Counter Object

char. of object
Attributes



JUnit Test Case 1

@Test

```
public void testIncAfterCreation() {  
    Counter c = new Counter();  
    assertTrue(Counter.MIN_VALUE == c.getValue());  
    assertEquals(Counter.MIN_VALUE, c.getValue());  
    assertEquals("Initial counter value is Counter.MIN_VALUE", Counter.MIN_VALUE, c.getValue());  
    try {  
        c.increment();  
        assertEquals(1, c.getValue());  
    }  
    catch (ValueTooLargeException e) {  
        fail("ValueTooLargeException thrown unexpectedly.");  
    }  
}
```

assertEquals (expect , actual)

myfaEv

JUnit Test Case 2

```
@Test
public void testDecFromMinValue() {
    /*
     * This test automates what's done in CounterTester1
     */
    Counter c = new Counter();
    assertEquals(Counter.MIN_VALUE, c.getValue());
    try {
        c.decrement();
        /* reaching this line means that c.decrement() did not throw an exception */
        fail("ValueTooSmallException was NOT thrown as expected.");
    } catch (ValueTooSmallException e) {
        /*
         * Do nothing - ValueTooSmallException thrown as expected.
         */
    }
}
```

Handwritten annotations:

- Red circles around `@Test` and `void testDecFromMinValue()`.
- Green arrow pointing to `c.decrement();` with text: *Case 1: VTSE thrown*
- Pink arrow pointing to `fail("ValueTooSmallException was NOT thrown as expected.");` with text: *Case 2: VTSE NOT thrown*
- Green box around the `catch` block.
- Green arrow pointing from the `catch` block to the `fail` line.

JUnit Test Case 3

```
@Test
public void testIncFromMaxValue() {
    /*
     * This test automates what's done in CounterTester2
     */
    Counter c = new Counter();
    try {
        c.increment();
        c.increment();
        c.increment();
    } catch (ValueTooLargeException e) {
        fail("ValueTooLargeException was thrown unexpectedly.");
    }
    assertEquals("Counter reaches max", Counter.MAX_VALUE, c.getValue());
    try {
        c.increment();
        fail("ValueTooLargeException was NOT thrown as expected.");
    } catch (ValueTooLargeException e) {
        /*
         * Do nothing - ValueTooLargeException thrown as expected.
         */
    }
}
```

Question: Is this alternative version appropriate?

```
1  @Test
2  public void testIncFromMaxValue() {
3      Counter c = new Counter();
4      try {
5          c.increment();
6          c.increment();
7          c.increment();
8          assertEquals(Counter.MAX_VALUE, c.getValue());
9          c.increment();
10         fail("ValueTooLargeException was NOT thrown as expected.");
11     } catch (ValueTooLargeException e) {
12     }
```

Handwritten annotations on the code:

- A blue vertical bar highlights lines 4 through 11.
- A green circle highlights the `c.increment()` call on line 5, with an arrow pointing to the text "VTLCE (unexpectedly)".
- A green circle highlights the `c.increment()` call on line 9, with an arrow pointing to the text "VTLCE (expected)".
- A green circle highlights the `catch` block on line 11, with an arrow pointing to the text "VTLCE (expected)".
- A blue circle highlights the `fail` call on line 10.
- A blue circle highlights the `ValueTooLargeException e` parameter in the `catch` block on line 11.
- A blue circle highlights the empty body of the `catch` block on line 12.