

# Homework 1

Below are four faulty programs. Each includes test inputs that result in failure. Answer the following questions about each program.

<pre> /**  * Find last index of element  *  * @param x array to search  * @param y value to look for  * @return last index of y in x; -1 if absent  * @throws NullPointerException if x is null  */ public int findLast (int[] x, int y) {     for (int i=x.length-1; i &gt; 0; i--)     {         if (x[i] == y)         {             return i;         }     }     return -1; } // test: x = [2, 3, 5]; y = 2; Expected = 0 // Book website: FindLast.java // Book website: FindLastTest.java </pre>	<pre> /**  * Find last index of zero  *  * @param x array to search  *  * @return last index of 0 in x; -1 if absent  * @throws NullPointerException if x is null  */ public static int lastZero (int[] x) {     for (int i = 0; i &lt; x.length; i++)     {         if (x[i] == 0)         {             return i;         }     }     return -1; } // test: x = [0, 1, 0]; Expected = 2 // Book website: LastZero.java // Book website: LastZeroTest.java </pre>
<pre> /**  * Count positive elements  *  * @param x array to search  * @return count of positive elements in x  * @throws NullPointerException if x is null  */ public int countPositive (int[] x) {     int count = 0;     for (int i=0; i &lt; x.length; i++)     {         if (x[i] &gt;= 0)         {             count++;         }     }     return count; } // test: x = [-4, 2, 0, 2]; Expcted = 2 // Book website: CountPositive.java // Book website: CountPositiveTest.java </pre>	<pre> /**  * Count odd or postive elements  *  * @param x array to search  * @return count of odd/positive values in x  * @throws NullPointerException if x is null  */ public static int oddOrPos(int[] x) {     int count = 0;     for (int i = 0; i &lt; x.length; i++)     {         if (x[i]%2 == 1    x[i] &gt; 0)         {             count++;         }     }     return count; } // test: x = [-3, -2, 0, 1, 4]; Expected = 3 // Book website: OddOrPos.java // Book website: OddOrPosTest.java </pre>

(a) Explain what is wrong with the given code. Describe the fault precisely by proposing a modification to the code.

i.

<pre> /**  * Find last index of element  *  * @param x array to search  * @param y value to look for  * @return last index of y in x; -1 if absent  * @throws NullPointerException if x is null  */ public int findLast (int[] x, int y) {     for (int i=x.length-1; i &gt;= 0; i--) </pre>
--

```

    {
        if (x[i] == y)
        {
            return i;
        }
    }
    return -1;
}
// test: x = [2, 3, 5]; y = 2; Expected = 0
// Book website: FindLast.java
// Book website: FindLastTest.java

```

- 原程式漏檢查index為0的element，因此需修改迴圈的條件式為

$i \geq 0$ 。

ii.

```

/**
 * Find last index of zero
 *
 * @param x array to search
 *
 * @return last index of 0 in x; -1 if absent
 * @throws NullPointerException if x is null
 */
public static int lastZero (int[] x)
{
    for (int i = x.length-1; i >= 0; i--)
    {
        if (x[i] == 0)
        {
            return i;
        }
    }
    return -1;
}
// test: x = [0, 1, 0]; Expected = 2
// Book website: LastZero.java
// Book website: LastZeroTest.java

```

- 原程式的return結果為第一個element值為0的index，須從array x中最

後一個元素往前檢查才符合需求。

iii.

```

/**
 * Count positive elements
 *
 * @param x array to search
 * @return count of positive elements in x
 * @throws NullPointerException if x is null
 */
public int countPositive (int[] x)
{
    int count = 0;
    for (int i=0; i < x.length; i++)
    {
        if (x[i] > 0)

```

```

        {
            count++;
        }
    }
    return count;
}
// test: x = [-4, 2, 0, 2]; Expcted = 2
// Book website: CountPositive.java
// Book website: CountPositiveTest.java

```

- 0非正數，須將判斷條件中的等號移除。

iv.

```

/**
 * Count odd or postive elements
 *
 * @param x array to search
 * @return count of odd/positive values in x
 * @throws NullPointerException if x is null
 */
public static int oddOrPos(int[] x)
{
    int count = 0;
    for (int i = 0; i < x.length; i++)
    {
        if (x[i]%2 == -1 || x[i] > 0)
        {
            count++;
        }
    }
    return count;
}
// test: x = [-3, -2, 0, 1, 4]; Expected = 3
// Book website: OddOrPos.java
// Book website: OddOrPosTest.java

```

- 負奇數mod 2的結果為-1，因此須將判斷式 $x[i]\%2 == 1$ 修改為 $x[i]\%2 == -1$  (因為 $x[i]\%2 == 1$ 已包含於 $x[i] > 0$ )。

(b) If possible, give a test case that does not execute the fault. If not, briefly explain why not.

在所有code中，只要x為null則不會執行到fault，而是進入NullPointerException。

(c) If possible, give a test case that executes the fault, but does not result in an error state. If not, briefly explain why not.

- $x = [2, 3, 5]$  ;  $y = 3$ , Result = 1
- 必定發生error state。Error: i的初始值應為 $x.length-1$ ，其值卻為0
- $x = [-4, 2, -1, 2]$ , Result = 2
- $x = [-2, 2, 0, 1, 4]$ , Result = 3

(d) If possible, give a test case that results in an error state, but not a failure. Hint: Don't forget about the program counter. If not, briefly explain why not.

- $x = [2, 3, 5]$ ;  $y = 0$ , Result = -1 ◦ Error: 出迴圈時 $i$ 的值應為-1，其值卻為0
- $x = [1, 1, 0]$ , Result = 2 ◦ Error:  $i$ 的初始值應為 $x.length-1$
- error state必定發生在count，而count為return結果，因此必定造成failure ◦
- error state必定發生在count，而count為return結果，因此必定造成failure ◦

(e) For the given test case, describe the first error state. Be sure to describe the complete state.

- 出迴圈時 $i$ 的值應為-1，其值卻為0
- $i$ 的初始值應為 $x.length-1$ ，其值卻為0
- 執行完 $x[2]$ 的判斷時，count應為1，其值卻為2
- 執行完 $x[0]$ 的判斷時，count應為1，其值卻為0

(f) Implement your repair and verify that the given test now produces the expected output. Submit a screen printout or other evidence that your new program works.

The screenshot shows an IDE with two main windows. The left window displays the `MainTest.java` file, which contains the following code:

```

public static int findLast(int[] x, int y) {
    for (int i = x.length - 1; i >= 0; i--) {
        if (x[i] == y) {
            return i;
        }
    }
    return -1;
}

/**
 * Find last index of zero
 * @param x array to search
 * @return last index of 0 in x; -1 if absent
 * @throws NullPointerException if x is null
 */
public static int lastZero(int[] x) {
    for (int i = x.length - 1; i >= 0; i--) {
        if (x[i] == 0) {
            return i;
        }
    }
    return -1;
}

/**
 * Count positive elements
 * @param x array to search
 * @return count of positive elements in x
 * @throws NullPointerException if x is null
 */
public static int countPositive(int[] x) {

```

The right window displays the `MainTest` class, which contains the following code:

```

class MainTest {

    @test
    void findLast() {
        int[] x = {0, 0, 0};
        int y = 2;
        assertEquals("expected 0, Main.findLast(x, y)", 0, Main.findLast(x, y));
    }

    @test
    void lastZero() {
        int[] x = {0, 1, 0};
        assertEquals("expected 2, Main.lastZero(x)", 2, Main.lastZero(x));
    }

    @test
    void countPositive() {
        int[] x = {-4, 2, 0, 2};
        assertEquals("expected 2, Main.countPositive(x)", 2, Main.countPositive(x));
    }

    @test
    void oddOrPos() {
        int[] x = {-3, -2, 0, 1, 4};
        assertEquals("expected 3, Main.oddOrPos(x)", 3, Main.oddOrPos(x));
    }
}

```

At the bottom, a "Run" window shows the test results for `MainTest`. The results indicate that all tests passed:

Test	Time	Result
findLast	46ms	Passed
lastZero	46ms	Passed
countPositive	2ms	Passed
oddOrPos	2ms	Passed

The overall test results show "Tests passed: 4 of 4 tests - 46ms".