Sample Questions and Answers

Chapter 1: Why do we test?

Two faulty programs are given below. Each includes test inputs that result in failure. For each of these two programs, answer the following questions **a — f**. For **"f"** copy and paste your corrected code in this document.

`numZero` method

```
/**
 * Counts zeroes in an array
 *
 * @param x   array to count zeroes in
 * @return    number of occurrences of 0 in x
 * @throws    NullPointerException if x is null
 */
// As example in the book points out, this loop should start at 0.
// Better yet, it should be a foreach loop,
// which eliminates the possibility of the fencepost fault:
// for (int i:x) { if (x==0) count++; }
public static int numZero (int[] x)
{
    int count = 0;
    for (int i = 1; i < x.length; i++) {
        if (x[i] == 0){
            count++;
        }
    }
    return count;
}
// Test x = [0, 4, 6], Expected = 1
```

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

Answer:

The fault is located in the initialization part of the for loop (int i = 1). i should start searching at 0, not 1.

fault

```
for (int i = 1; i < x.length; i++) {
```

To eliminate the fault, the for loop should be modified as follows:

```
for (int i = 0; i < x.length; i++) {
```

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

Answer:

No such test case. All inputs start the loop, so all inputs execute the fault–even the null input.

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.

Answer:

Error occurs when the for loop initializes i to 1, not 0, on the first iteration. This causes the program state to be incorrect since the correct program would initialize i to 0, on the first iteration. Thus, no such test case exists because all inputs execute the faults and result in error state.

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

Answer:

If the array does not contain any 0 in the first index, the error state does not propagate to the output (i.e., no failure). One possible test case is [4, 6, 0]. In other words, the value of the count variable coincidentally correct (count = 1), the error state does not propagate to the output, and hence this test case does not result in a failure.

e) For the given test case below, describe the first error state. Be sure to describe the complete state. Hint: Don't forget about the program counter.

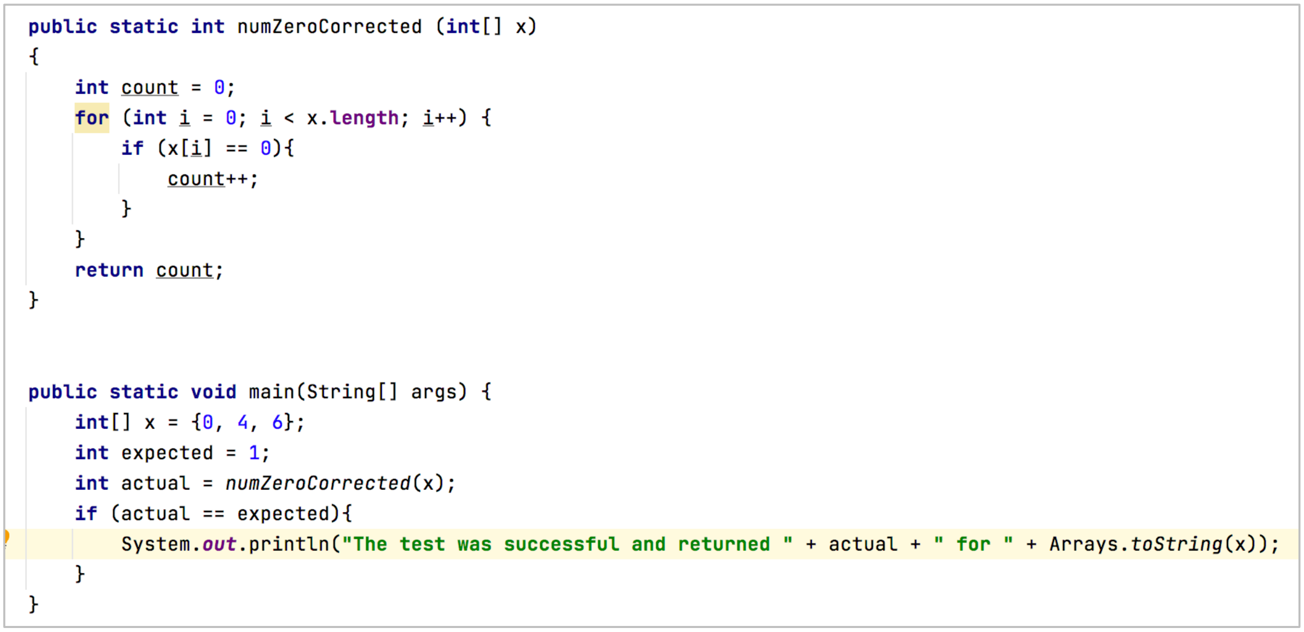
```
x =[0,4,6], Expected = 1
```

Answer:

The first error state occurs when index i has the value 1 when it should have 0.

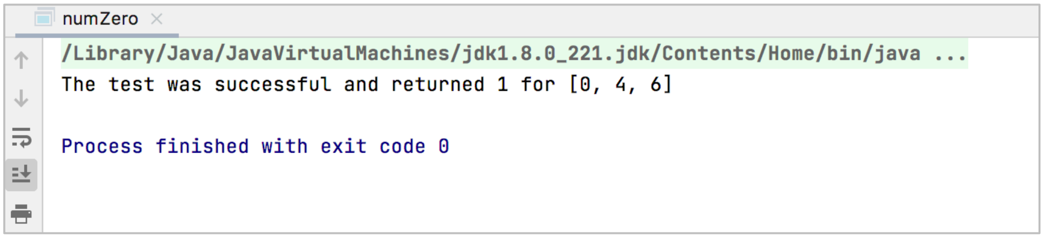
```
Input: x = [0, 4, 6]
Expected Output: 1
Actual Output: 0
```
First Error State: `x = [0, 4, 6]  i = 1, PC = just after i=1;`

f) Implement your repair and verify that the given test now produces the expected output. Submit a screenshot demonstrating your new program works.



```
public static int numZeroCorrected (int[] x)
{
    int count = 0;
    for (int i = 0; i < x.length; i++) {
        if (x[i] == 0){
            count++;
        }
    }
    return count;
}


public static void main(String[] args) {
    int[] x = {0, 4, 6};
    int expected = 1;
    int actual = numZeroCorrected(x);
    if (actual == expected){
        System.out.println("The test was successful and returned " + actual + " for " + Arrays.toString(x));
    }
}
```



```
/Library/Java/JavaVirtualMachines/jdk1.8.0_221.jdk/Contents/Home/bin/java ...
The test was successful and returned 1 for [0, 4, 6]

Process finished with exit code 0
```

`lastZero` method

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

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

Answer:

The method returns the index of the first occurrence of 0, not the last. There are many ways to fix this issue, one of which could be to search the item in the array from high to low, instead of low to high as shown below.

```
for (int i = x.length-1; i >= 0; i--)
```

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

Answer:

No such test case. All inputs start the loop, so all inputs execute the fault–even the null input.

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.

Answer:

For **`x = [0]`**, the incorrect and correct programs will have the same program states as shown below. Thus, though this test case executes the fault, it does not result in an error state.

The execution of `lastZero([0])` **would be as follows for the code with a for loop that searches items from low to high (i.e., incorrect program) vs. high to low (i.e., correct program) .**		
	Incorrect Program	Correct Program
1	`< x=[0], PC=[int i = 0 (L1)] >`	`< x=[0], PC=[int i=x.length-1 (L1)]>`
2	`< x=[0], i=0, PC=[i < x.length (L1)] >`	`< x=[0], i=0, PC=[i >= 0 (L1)] >`
3	`< x=[0], i=0, PC=[ if(x[i] ==0) (L2)] >`	`< x=[0], i=0, PC=[ if(x[i] ==0) (L2)] >`
4	`< x=[0], i=0, PC=[ return i (L3)] >`	`< x=[0], i=0, PC=[ return i (L3)] >`

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

Answer:

Though (almost) all of the executions contain error states, if there is a single zero in the array as shown below, the method will return the correct answer (i.e., no failure).

Input: x = [5, 0, 6]
Expected Output: 1
Actual Output: 1

e) For the given test case below, describe the first error state. Be sure to describe the complete state. Hint: Don't forget about the program counter.

```
x = [0, 1, 0], Expected = 2
```

Answer:

The first error state occurs when index i has the value 0 when it should have a value at the end of the array, namely x.length-1.

Input: x = [0, 1, 0]

Expected Output: 2

Actual Output: 0

First Error State: x = [0, 1, 0] i = 0 PC = just after i= 0;