刘京宗 3019213043

```
public int findLast (int[] x, int y) {
    //Effects: If x==null throw NullPointerException
    // else return the index of the last element
    // in x that equals y.
    // If no such element exists, return -1
    for (int i=x.length-1; i > 0; i--) {
        if (x[i] == y) {
            return i;
        }
    } return -1;
}
// test: x=[2, 3, 5]; y = 2
// Expected = 0
```

- **The fault**: The loop condition should be `i >= 0` instead of `i > 0`.
- **A test case that does not execute the fault**: `x=null; y=2`. This test case will throw NullPointerException.
- **A test case that executes the fault but does not result in an error state**: `x=[1,2,3]; y=2`. This test case will execute the fault because the loop will not check the first element of the array. However, it will not result in an error state because the method will return the correct index of the last element that equals `y=2`, which is `1`.
- **A test case that results in an error but not a failure**:`x=[1,2,3]; y=4` . This test case will return correct answer `-1`. But `i` should be `-1` after the loop rather than `0`.

```
public static int lastZero (int[] x) {
    //Effects: if x==null throw NullPointerException
    // else return the index of the LAST 0 in x.
    // Return -1 if 0 does not occur in x
    for (int i = 0; i < x.length; i++) {
        if (x[i] == 0) {
            return i;
        }
    }
     return -1;
}
// test: x=[0, 1, 0]
// Expected = 2
```

- **The fault**: The loop condition should be `int i=x.length;i>=0;i--` instead of `int i = 0; i < x.length; i++`.
- **A test case that does not execute the fault**: `x=null`. This test case will throw NullPointerException.

- **A test case that executes the fault but does not result in an error state**: `x=[1,2,0]`. This test case will execute the fault because the loop starts with `i=0`. However, it will not result in an error state because the method will return the correct index of the last element that equals `0`, which is `2`.
- **A test case that results in an error but not a failure**: `x=[1,2,3];` . This test case will return correct answer `-1`. But `i` should be `-1` after the loop rather than `3`.