# Homework 1

田野 2017329621125

1. findLast

(a)

The incorrect behavior is that in the for loop the method ignore the first element in the array. It could be revised to:

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

(b)

Test case: null pointer

When x is a null pointer, it will throw NullPointerException when int i=x.length-1. So in this case, it does not execute the fault.

(c)

If the method return before the end of the for loop, then the fault will be executed but does not result in an error state

test: x = [2, 3, 5]; y = 5

Expected = 2

Output = 2

(d)

If the intended value doesn’t exist in the array then the results will in an error but does not result a failure. One possible test case could be:

test: x = [2, 3, 5]; y = 1

Expected = -1

Output = -1

(e)

The first error state is when has the value 0, the for loop mistaken end at this point. In face, the for loop should continue, the judgement condition shoudl be “i>=0” instead of “i>0”. That is:

test: x = [2,3,5] y = 2

Expected = 0

Output = -1

First error state: i = 0

(f)

The fault could be revised to:

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

Since now the for loop will contain the first element in the array, so for the given test case will produce the expected output.

test: x = [2,3,5] y = 2

Expected = 0

Output = 0

2. lastZero

(a)

The incorrect behavior is that the method return the index of the first 0 instead of the last one. The fault exist in the for loop, one possible way to revise it would be:

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

( b)

All cases executes the for loop, so all cases will execute the fault even null input.

(c)

When x is [0], Since the return i statement will be executed, so it will not result in an error state.

test: x = [0]

Expected = 0

Output = 0

(d)

If the array only has one 0 and some other values, then the case may result in an error but not a failure. One possible solution would be:

test: x = [1,0,1]

Expected = 1

Output = 1

(e)

The first error is that when the index has the value 0 when it should have the value at the end of the array which is x.length -1. That is:

test: x=[0,1,0]

Expected = 2

Output = 0

First error state i = 0

(f)

The fault could be revised to:

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

Since now the for loop will reverse traversal the array, so it could find the last 0 in the array.

test: x = [0,1,0]

Expected = 2

Output = 2

3. countPositive

(a)

The incorrect behavior is that when then value is 0, the count will add. In fact 0 is not positive number. It could be revised to:

if (x[i] > 0)

(b)

If there is no element in the array, then the fault would not be executed. One possible test case would be:

test: x = []

Expected = 0

Output = 0

(c)

If there is no 0 in the array, then the fault would be executed, but not result in an error state. One possible test case would be:

test: x = [1, 2, 3]

Expected = 3

Output = 3

(d)

It’s impossible. Because all the test case result in an error will result a wrong count which cause a failure.

(e)

The first error state is that when the index i = 2, x[i] = 0, the count plus 1, but in fact 0 is not a positive number. That is:

test: x = [-4,2,0,2]

Expected = 2

Output = 3

First error state i = 2

(f)

The fault could be revised to:

if (x[i] > 0)

Since now 0 will no longer be regarded positive number, so the test case would produce the expected output.

test: x = [-4,2,0,2]

Expected = 2

Output = 2

4. oddOrPos

(a)

The incorrect behavior is that a negative odd number will not be taken into account. That’s because in java a negative odd number % 2 will generate -1. So one possible way to revise if statement would be:

if (x[i] % 2 == -1 || x[i] > 0)

positive odd number would be taken into account by the second test.

(b)

If the x is empty then the fault would not be execute.

test: x = []

Expected = 0

Output = 0

(c)

If there is no negative number in the array then the fault would be executed but does not result in an error state. One possible test cast would be:

test: x = [1, 2, 3]

Expected = 3

Output = 3

(d)

It is impossible, because in this case, every test case result in an error would cause the count plus, thus would led to a failure.

(e)

The first error state is when i = 0 x[i] = -3. Because in java a negative odd number % 2 will generate -1 so the first test in if is False. That is:

test x = [-3,-2,0,1,4]

Expected = 3

Output = 2

first error state i = 0

(f)

one possible way to revise if statement would be:

if (x[i] % 2 == -1 || x[i] > 0)

Since all the negative odd number % 2 result -1 in java and positive odd number would be taken into account by the second test. So the test case now produces the expected output.

test x = [-3,-2,0,1,4]

Expected = 3

Output = 3