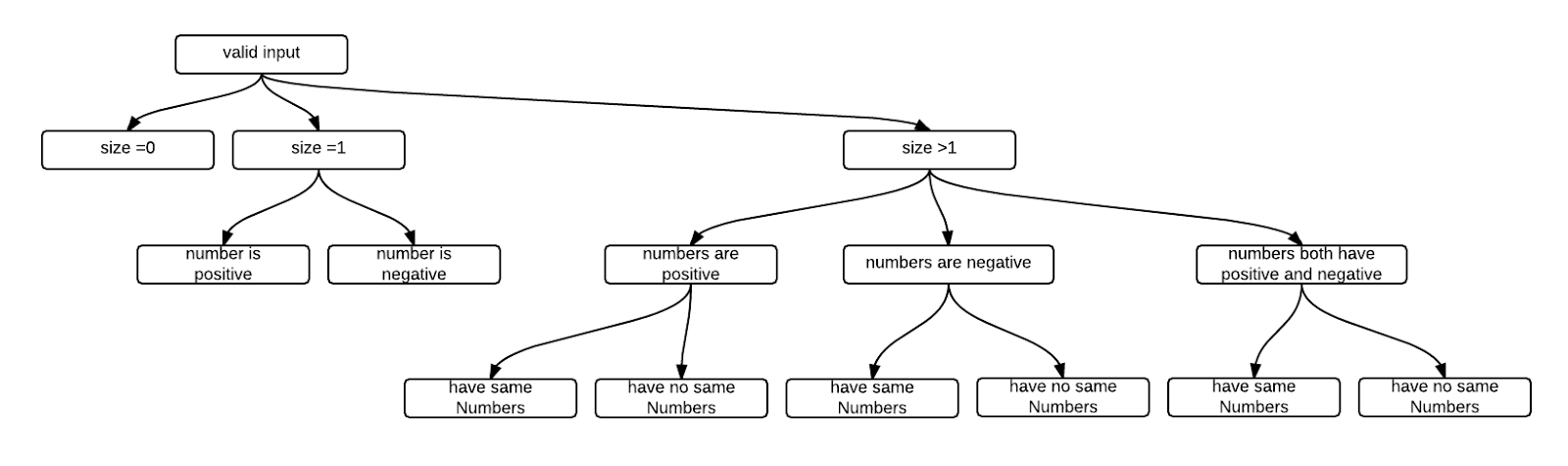
Program: original program NonAbstractedBubbleSort

Input domain: int[]

Input partition are shown in the following graph:



test cases:

(1) {} =>{}

(2) {2} => {2}

(3) {-3} => {-3}

(4) {4,1,2,6,3} => {1,2,3,4,6}

(5) {4,3,2,6,2} => {2,2,3,4,6}

(6) {-1,-10,-2,-5,-3} => { -10,-5,-3,-2,-1}

(7) {-1,-4,-2,-10,-4} => { -10, -4, -4, -2, -1}

(8) {2,4,-1,3,-10} => { -10, -1, 2, 3, 4}

(9) {6,-1,10,3,-1} => { -1, -1, 3, 6, 10}

No faults.

mutant 1:

line 7 : m = n => m = n-1

No faults

mutant 2:

line 7 : m >= 0 => m >= 1

No faults

mutant 3:

line 8 : i = 0 => i = 1

6 failures

mutant 4:

line 8 : i < n-1 => i< n

8 failures;

mutant 5:

line 9 k =i+1 => k = i

6 failures;

mutant 6:

line 7 : m > = 0 => m>= 2

no faults;

mutant 7:

line 7 : m > = 0 => m > = 5

6 failures

mutant 8:

line 7 m = n => m = n -2

no faults

mutant 9:

line 8 m = n => m = n-10

6 failures

mutant 10:

i => i +1

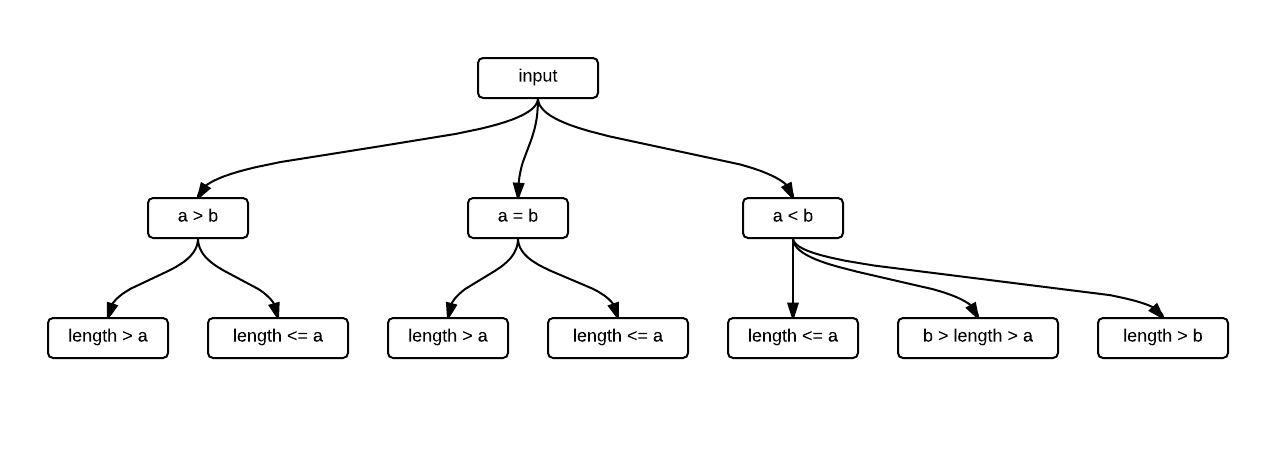
6 failures

**Tested program : Abstracted BubbleSort**

Function: swapNumbers

input domain : int a\*int b\*int[]

input partition shown in the following graph



test cases:

1 , 2, {2,3,6,1,5} => {2,6,3,1,5}

4 , 3, {5,23,1,6,1} => {5,23,1,1,6}

3 , 3, {2,5,1,4} => {2,5,1,5}

1 , 2, {1} => exception

2 , 5, {1,4,5,2} => exception

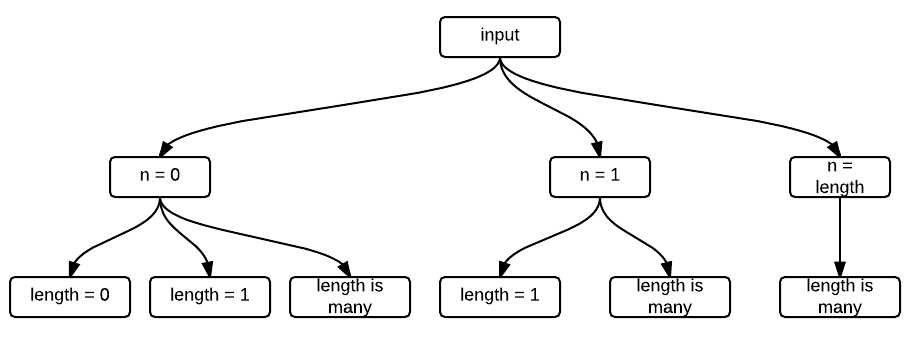
4 , 3, {1,4,2} => exception

3 , 3, {1} => expection

Function: iterateArray

Input domain int n \* int[]

input partition shown in the following graph



test cases:

0, {} => exception

0, {2} => {2}

0, {2,3,54,1,4} => {2,3,54,1,4}

1, {2} => exception

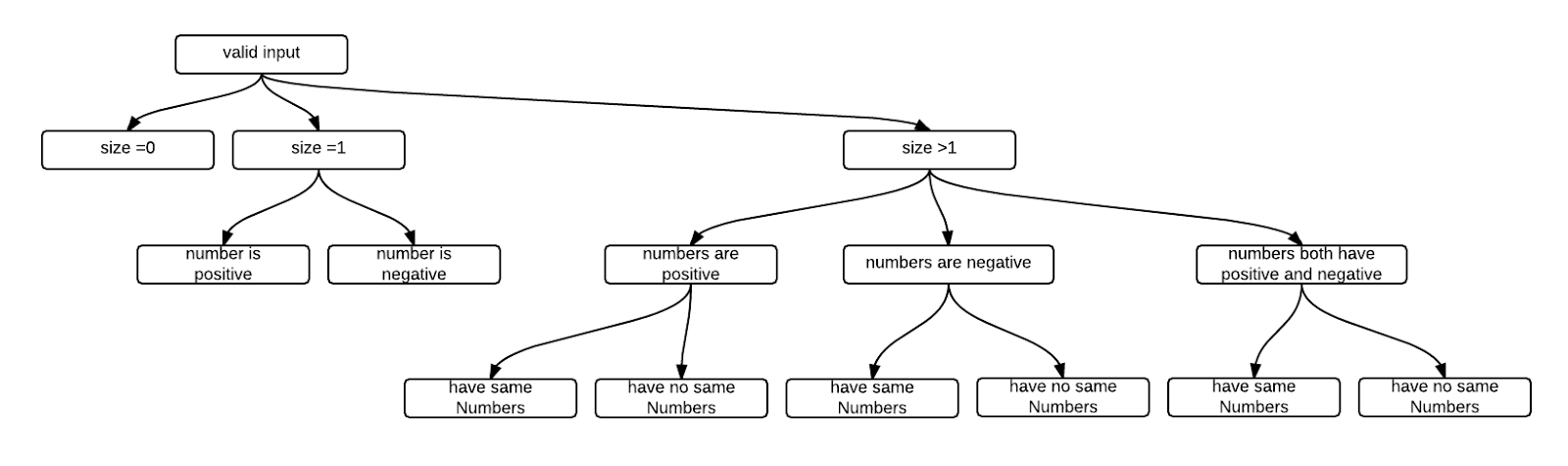
1, {34,3,5,1,45} => {3,34,5,1,45}

3, {2,54,12,5,6,1} => {2,12,5,54,6,1}

Function: Bubble\_srt

Input domain: int[]

Input partition are shown in the following graph:



test cases:

(1) {} =>{}

(2) {2} => {2}

(3) {-3} => {-3}

(4) {4,1,2,6,3} => {1,2,3,4,6}

(5) {4,3,2,6,2} => {2,2,3,4,6}

(6) {-1,-10,-2,-5,-3} => { -10,-5,-3,-2,-1}

(7) {-1,-4,-2,-10,-4} => { -10, -4, -4, -2, -1}

(8) {2,4,-1,3,-10} => { -10, -1, 2, 3, 4}

(9) {6,-1,10,3,-1} => { -1, -1, 3, 6, 10}

original program:

no fault

mutant 1:

line 6 : m = n => m =n-1

no fault

mutant 2:

line 6 : m >= 0 => m >= 1

no fault

mutant 3:

line 16 : i = 0 => i = 1

7 failure

mutant 4:

line 16 : i < n-1 => i< n

9 errors and 2 failure

mutant 5:

line 17 k =i+1 => k = i

7 failures;

mutant 6:

line 6 : m > = 0 => m>= 2

no fault

mutant 7:

line 6 : m > = 0 => m>= 5

6 failures

mutant 8:

line 6 m = n => m = n -2

no faults

mutant 9:

line 8 m = n => m = n-10

6 failures

mutant 10:

i => i +1

6 failures