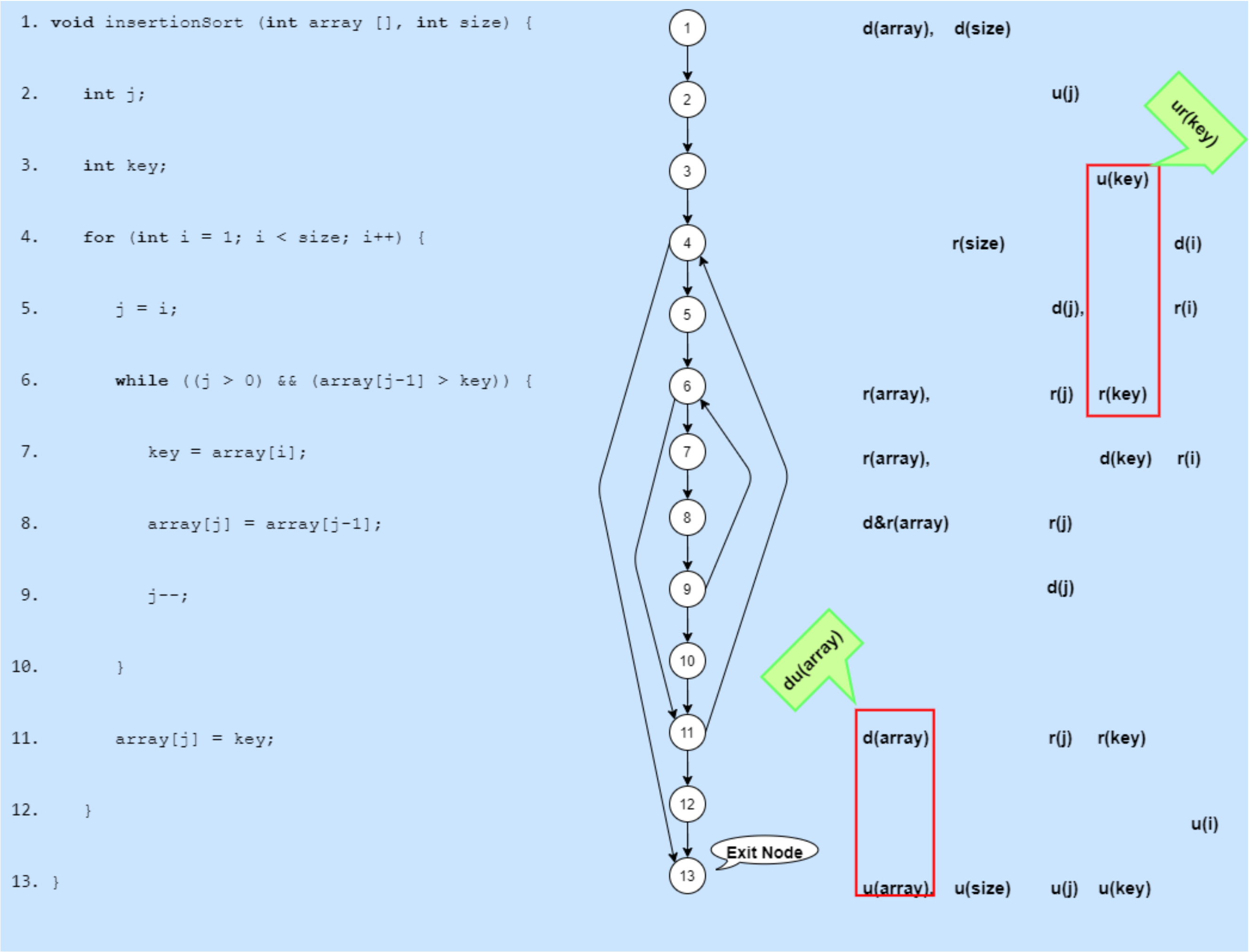
Alexander Guðmundsson
alg35@hi.is

Software Testing assignment 4

1 2 and 3)



```
1. void insertionSort (int array [], int size) {
2.     int j;
3.     int key;
4.     for (int i = 1; i < size; i++) {
5.         j = i;
6.         while ((j > 0) && (array[j-1] > key)) {
7.             key = array[i];
8.             array[j] = array[j-1];
9.             j--;
10.        }
11.        array[j] = key;
12.    }
13. }
```

Note: there should be an Starting note at note 1.

There could be a problem with the ur(key) anomaly because you are reading a value that is undefined, there is not a problem with du(array) since the u(array) is after the function has finished, unless you plan to use the array outside of the insertion sort function, then there could be a problem.

Alexander Guðmundsson
alg35@hi.is

4)

v(G) := e - n + p

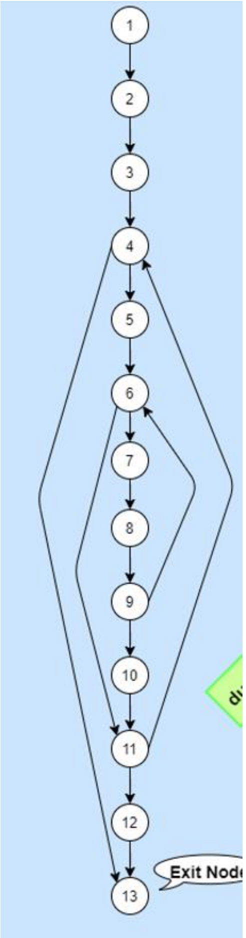
e = 16

n = 13

p = 2

v(G) = 16-13+2 = 5

5)

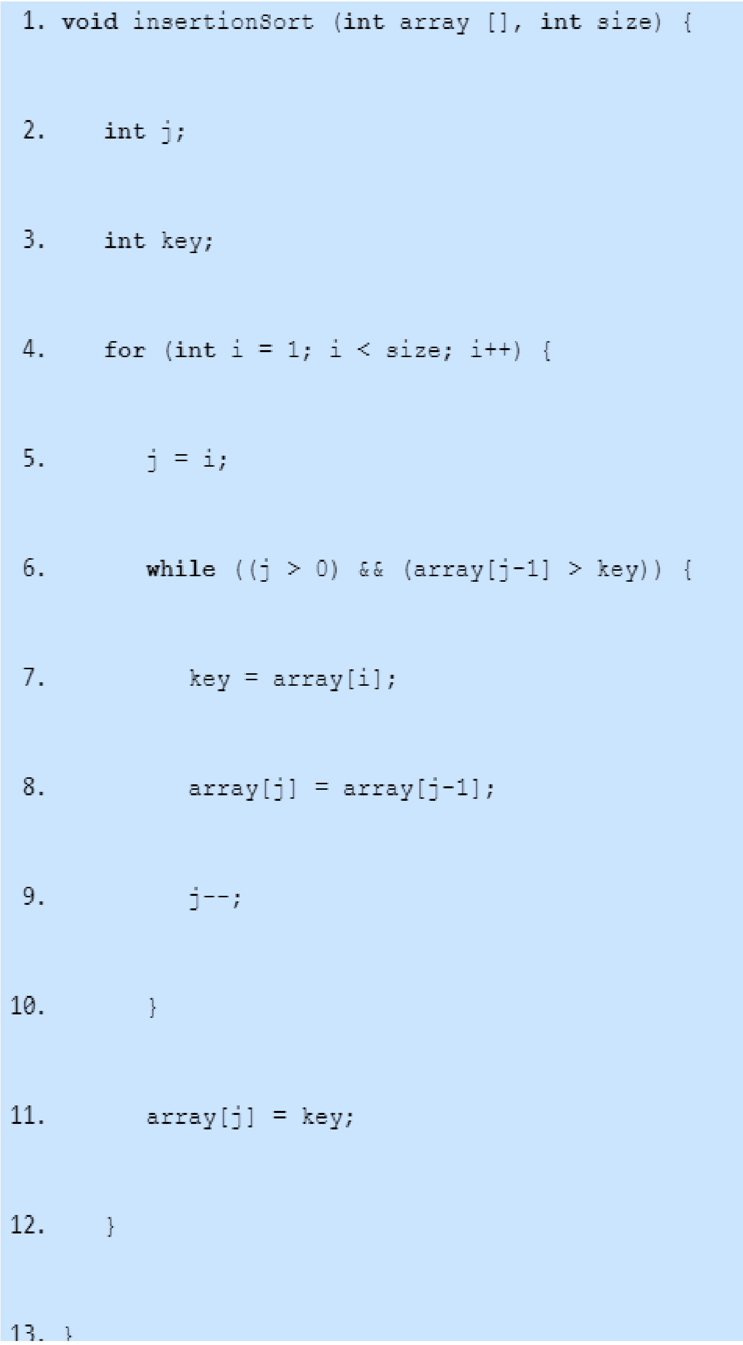


Let’s look at each path:
path 1 = {1,2,3,4,5,6,7,8,9,10,11,12,13}
path 2 = {1,2,3,4,13}
path 3 = {1,2,3,4,5,6,11…*}
path 4 = {1,2,3,4,5,6,7,8,9,6…*}
path 5 = {1,2,3,4,5,6,7,8,9,10,11,4…*}

Just like in McCabe Cyclomatic above, there are 5 linerally independent paths.

Alexander Guðmundsson
alg35@hi.is

6)



```
 1. void insertionSort (int array [], int size) {
 2.    int j;
 3.    int key;
 4.    for (int i = 1; i < size; i++) {
 5.       j = i;
 6.       while ((j > 0) && (array[j-1] > key)) {
 7.          key = array[i];
 8.          array[j] = array[j-1];
 9.          j--;
10.       }
11.       array[j] = key;
12.    }
13. }
```

1	nesting = 0
2	nesting = 1
3	nesting = 1
4	nesting = 1
5	nesting = 2
6	nesting = 2
7	**nesting = 3**
8	**nesting = 3**
9	**nesting = 3**
10	nesting = 2
11	nesting = 2
12	nesting = 1
13	nesting = 0

Maximum level og nesting = **3**