6.1

Read the document with the notes on Software Testing. (sw\_testing\_notes)

1. Tests that are performed over the entire system are a special case of the so-called integration testing.
2. In which category would you include the so-called unit testing? Component testing(unit🡪individual program component)
3. What is Input Equivalence Partitioning? A technique to divide the tests based on categories of the input values.

A technique that requires various tests in order to execute all possible code blocks within a function. – Program flow graph/path testing

1. What is a pre-condition when describing a function? A condition that must be true before the execution of the function for it to produce the correct result.
2. What is the essential feature of “white box testing”? The tests are derived by exploring the structure of the code (all program statements).

The input values are divided into partitions so that tests are chosen for each partition. – Equivalence partition (tries to define test cases that uncover classes of errors, derived from input attributes in the requirements specification—black box testing)

1. A function does not need to be executed in order to obtain the program flow graph
2. Two functions that implement the same functionality, may not have identical program flow graphs, since they may be implemented with different algorithm.
3. The cyclomatic complexity of a function is used to know the number of tests so that all control statements are tested once.

cyclomatic complexity is based on the evaluation of each branch derived from control statements, and derive the equal number of tests to test those condition statements, at same time, the complexity is derived from the graph. However, it only test one path at each case, the combination of paths are tested.

6.2 Review the content of the Software Testing notes (sw\_testing\_notes) with special attention to:

Component vs Integration testing.

Structure of a test case: data, result, reports.

Black box testing.

Equivalence Partitioning

Structural testing: program flow graphs.

6.3

1. What type of code portions are more likely tested by each method?

WBT would be focus on testing the all the program block/component/units as much as possible, and data structures or inner workings inside the system, with known knowledge of system.

BBT would be cover attributes of system, including Correctness, Usability, and Performance etc. It is to perform over the entire system, ensure the behavior of system meet the system specifications. It does not use knowledge of internal structure, code or mechanism. It generally use for higher levels of testing.

2. When is each method more used during the development time of a product?

WBT is more likely to be used during the earlier implementation of features, to ensure the proper function of each single block of codes.

BBT is more used during the later stage system test and integration test, which involves the interaction with interface from other modules.

(White box testing can be thought of as black box testing at a much smaller granularity, since tests are really designed from different level abstract, sometimes distinction are not quite important.)

3. What type of defects to they focus on

WBT would focus on any deep logically/branch bugs, and language specific errors which are related to internal executions. (Try to cover tests for all possible execution paths and their combinations in the code)

BBT would focus on failure or faults of the system, such as improper function of interface and interactions, UI error and any performance issues, which are related to input-output relations of functions for software requirements(functional and non-functional).

However, WBT may miss some design errors which could be identified with help of BBT.

4. Which of these methods detects defects that are easier to fix

The defects detected from WBT may be much easier to fix, because it is mainly in the small block of the system with less interfering with other blocks, and also the inputs of test cases are simple, makes origin of those defects be easier to track down.

The defects from BBT are for entire system, usually involve interactions with other modules or dependency systems, the inputs of test cases are more complicated and hard to find out the root of errors(normally require help of log and traverse relevant components in the complex interactions).

5. What type of individual is more likely to perform each of the tests?

WBT is usually done by people with implementation knowledge of certain features/components of the system.

BBT is usually carried out by test engineers or with knowing entire specifications of the system, and people intend to use the interface of the target system or modules, they usually do not need to know implementation detail (usually software quality department).

Both tests should be run regularly on the CI server.

6.4

public static void sort(int[] array)

{

int i = 0;int n = array.length;--1

while (i < n - 1)--2

{ int j = i + 1;--3

while (j < n)--4

{if (array[i] < array[j])--5

{ int tmp;tmp = array[i];--6

array[i] = array[j];array[j] = tmp;

}

j = j + 1;--7

}

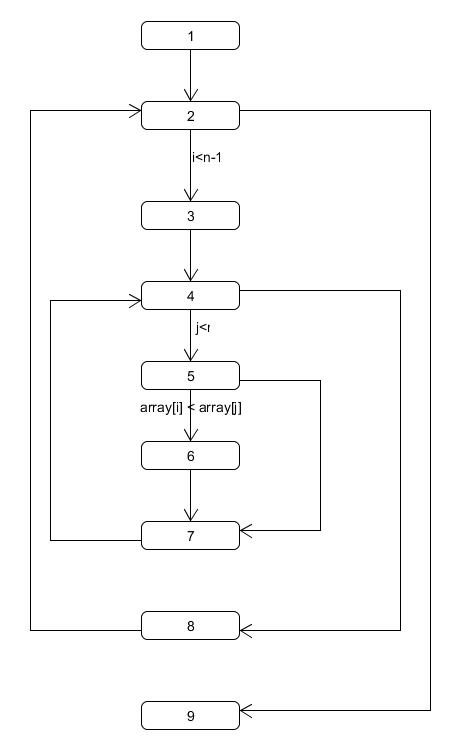
i = i + 1;--8

}

--9

}

1. Analyze the code of the function sort and draw the program flow graph

**

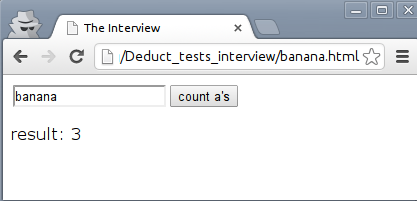
1. Calculate the cyclomatic complexity of the function.

Number of edges=11, number of node=9, complexity=11-9+2=4

1. Write the set of independent paths so that all statement are executed.

1,2,9; 1,2,3,4,8; 1,2,3,4,5,7,4,8; 1,2,3,4,5,6,7,4,8

6.5

**

Together with your team mates, create a one page document with the most interesting black box tests to apply to the JavaScript behind this page. Each test needs to be succinctly described with the input to write in the form followed by a colon, the expected output shown when the button is pressed, and a single sentence in parenthesis explaining the reason. For example: “this is a test”: 1 (Regular test case)

"": 0 (check empty)

"ewqewqewqewq": 0 (check not exist)

"^^&\*": 0 (check special character)

"哈哈哈": 0 (check not english character)

"aaaa": 4 (check all a)

"a ththth":1 (check first a)

"ththth a":1 (check last a)

"tht a hth":1 (check middle a)

"a a":2 (check space between a)

"A a":1 (check capital)

“@“: 0 (check character seems like ‘a’)

“<a>”:1 (check with escape character)

“å”:0

"👌", 0 (check with emoji)

“àáâäæãāå”:0 (check with character seems like ‘a’)

“àáâäæãāåa”:1 (check with character seems like ‘a’)

“àáâäæãāåA”:1 (check with character seems like ‘a’)

"\n\t\": 0 (check escape)

"select \* from dual;": 1 (check sql)

"<script type='text/javascript'>": 2 (inject script code)