Name: **Sajjad Younas**

ID : **13850**

Instructor: **Sir Zain Shaukat**

Course**: Software Verification And Validation**

Department**: BS (SE)**

Date: 25/6/2020

****

" Software verification and validation"

Q1. MCQS (10)

1. When should company stop the testing of a particular software?

a. After system testing done
b. *It depends on the risks for the system being tested*
c. After smoke testing done
d. None of the above

2. White-Box Testing is also known as _____ .

a. Structural testing
b. Code-Based Testing

c. *Clear box testing*
d. All of the above

3. _____ refers to a different set of tasks ensures that the software that has been built is traceable to Customer Requirements.

a. Verification
b. Requirement engineering
c. *Validation*
d. None of the above

4. _____ verifies that all elements mesh properly and overall system functions/performance is achieved.

a. Integration testing
b. Validation testing
c. Unit testing
d. *System Testing*

5. What do you verify in White Box Testing?
- Published on 03 Aug 15

a. Testing of each statement, object and function on an individual basis.
b. Expected output.
c. The flow of specific inputs through the code.
d. *All of the above.*

6. _____ refers to the set of tasks that ensures the software correctly implements a specific function.
- Published on 03 Aug 15

a. *Verification*
b. Validation
c. Modularity
d. None of the above.

7. Who performs the Acceptance Testing?
- Published on 03 Aug 15

a. Software Developer
b. End users
c. Testing team
d. Systems engineers

8. Which of the following is not a part of Performance Testing?
- Published on 30 Jul 15

a. Measuring Transaction Rate.
b. Measuring Response Time.
c. *Measuring the LOC.*
d. None of the above.

9. Which of the following can be found using Static Testing Techniques?
- Published on 29 Jul 15

a. *Defect*
b. Failure
c. Both A & B

10. Testing of individual components by the developers are comes under which type of testing?

- Published on 29 Jul 15

a. Integration testing
b. Validation testing
c. Unit testing
d. None of the above.

****

Q 02:- Explain Black Box testing and White Box testing in detail. (10)

Ans:- **Black box testing:-**

Black box testing is a method of software testing that examines the functionality of an application without peering into its internal structures or working.

Black box testing focuses on the functional requirements of the software.

It enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements for a program

Input **output**

Black box testing (BBT) attempts to find errors in the following categories

➔Incorrect or missing functions

➔Interface errors

➔Errors in data structures

➔Behavior or performance errors

➔Initialization and termination errors

Tests are designed to answer the following questions

➔How is functional validity tested?

➔How is system behavior and performance tested?

➔What classes of input will make good test cases?

As BBT purposely disregards control structure, attention is focused on the information domain.

By applying BBT, we derive a set of test cases that satisfy the following criteria

➔Test cases that reduce the number of additional test cases that must be designed to achieve reasonable testing.

➔Test cases that tell us something about the presence or absence of classes of errors, rather than an error associated only with the specific test at hand.

The **technique** used in BBT is given below:

(i)Equivalence partitioning

(ii)Boundary value analysis

Equivalence partitioning:- Equivalence partitioning is a black-box testing method that divides the input domain of a program into classes of data from which test cases can be derived.

Equivalence partitioning strives to define a test case that uncovers classes of errors, thereby reducing the total number of test cases that must be developed.

An equivalence class represents a set of valid or invalid states for input conditions

Boundary value analysis:

Boundary value analysis (BVA) is a testing technique, which leads to a selection of test cases that exercise bounding values

This is because that for reasons not clearly completely clear, a greater number of errors tends to occur at the boundaries of the input domain rather than in the center.

Boundary value analysis:

1)**User queries 2**)**mouse picks 3**)out**put format 4**)p**rompts 5**)**FK input 6**)**data**

This all of the above is the input domain and its give us output domain.

State testing are the uses equivalence partitioning techniques.

Tools for bbt:-

➔QTP, Selenium, Loadrunner, jmeter

<u>White box testing:-</u>

White box testing(WBT) is a type of testing in which software internal structure, design, and coding is tested. It is also known as Clear Box testing, Open Box testing, Structural testing, Transparent Box testig. It is usually performed by developers.

In this type of testing, the code is visible to the tester.

White box testing involves the testing of the software code for the following:

➔Internal security holes

➔poorly structured paths in the coding processes

➔Expected output

➔The functionality of conditional loops

➔Testing of each statement, object, and function on an individual basis

White box testing technique:

A major White box testing technique is Code Coverage analysis

Code coverage:-

Code Coverage analysis eliminates gaps in a Test Case suite. It identifies areas of a program that are not exercised by a set of test cases. Once gaps are identified, you create test cases to verify untested parts of the code, thereby increasing the quality of the software product.

There are automated tools available to perform Code coverage analysis. Below are a few coverage analysis techniques a box tester can use to perform tests:

1) **Statement Coverage**: This technique requires every possible statement in the code to be tested at least once during the testing process of software engineering.
2) **Branch coverage**:- This technique checks every possible path (if-else and other conditional loops) of a software application.

Following are some other techniques I mention only its names and taken from internet

- Decision Coverage
- Condition Coverage
- Multiple Condition Coverage
- Finite State Machine Coverage
- Path Coverage
- Control flow testing
- Data flow testing

White box testing tools:

some of the white box testing tools is given below:

- Parasoft Jtest
- EclEmma
- NUnit
- PyUnit
- HTMLUnit

Example:

```
Printtme   (int a, intb)

 {

  int result = a+ b;

  If  (result > 0)

  Print(“positive”, result)


Else


  Print(“negative”, result)

}
```

****

Q3. Find the cyclomatic Complexity and draw the Graph of this code. (15)

Program-*X*:

```
sumcal(int maxint, int value)
{
    int result=0, i=0;
    if (value <0)
    {
      value = -value;
    }
    while((i<value) AND (result
<= maxint))
    {
       i=i+1;
      result = result + 1;
    }
    if(result <= maxint)
    {
       printf(result);
    }
    else
    {
      printf("large");
    }
    printf("end of program");
}
```

Ans:- **Cyclomatic complexity:-** Cyclomatic complexity is a software metric used to measure the complexity of a program.

These metric, measures independent paths through program source code.

Cyclomatic complexity can be calculated with respect to functions, modules, methods or classes within a program.

Cyclomatic complexity of the code is:

The complexity M is define as.

$M = E - N + 2P$

Where

E = the number of edges of the graph.

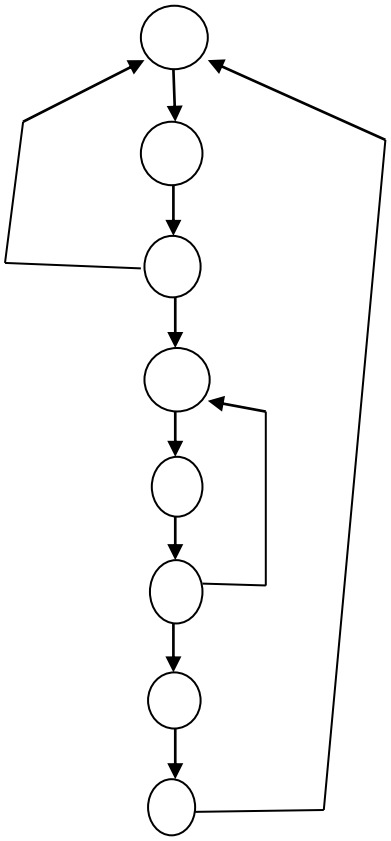
N = the number of nodes of the graph.

P = the number of connected components.

So using the above formula:

For first program X, E = 11, N = 9, P = 1, So **M = 11-9+2*1 = 4**

Graph of the code is given below:-



****

Q4. What is Z specification and why its is used for, also give some example this code written in Z specification. (15)

Ans 04:- **Z specification:-** **"**Describe the behavior of system in the language o f modern mathematics**".**

the Z specification used for describing and modeling computer systems. It is targeted at the clear specification of computer program and computer based systems in general.

Z contains a standardized catalogue (called the *mathematical toolkit*) of commonly used mathematical functions and predicates, defined using Z itself.

Although Z notation (just like the APL language, long before it) uses many non-ASCII symbols, the specification includes suggestions for rendering the Z notation symbols in ASCII and in LaTeX. There are also Unicode encodings for all standard Z symbols.

➔**WHY Z specification:-**

Following are the main reason why we used Z specification:

>expressive power.

>precise formalism.

>can be used to model a broad range of system.

>Accuracy important for safety-critical systems.

➔**Example:-**

Banking system:

WithdrawMoney

BankAccount----------->BankAccount’

Dollars : N	Dollars’ : N
Cent: N	Cent’ : N
Dollars ≥ 0	Dollars’ ≥ 0
Cent ≥ 0	Cent’ ≥ 0

△BankAccount

Dollarmount? : N

centAmount? : N

dollarAmount? ≤dollars

DollarAmount? = dollar ==>centAmount? ≤ cents

centAmount? > cents

⇨ (dollars' = dollars - dollarAmount? – 1

^ cents' = cents - centAmount> + 100)

centAmount? ≤ cents

==>(dollars' = dollars - dollarAmount?

^ cents' = cents - centAmount?)

Example 02:-

Delete entry

Delete_ok

△Datadictionary

Name?:NAME

name? ἐ dom ddict

ddict' {name ?}◁ ddict

THE END