# UNIT-4

1. What is testing? How is it different from debugging?

A: Software testing is a method of verification involving systematically executing software to detect defects. Glen Myers defines testing as "A process of executing a program with an intent of finding an error". Software testing plays a critical role in software quality. It is an ultimate review of specification, design and coding.

| aspects | Testing | Debugging |
|---|---|---|
| definition | Testing is the process to find bugs and errors. | Debugging is the process of correcting the bugs found during testing |
| purpose | The purpose of testing is to identify defects or errors in software system. | The purpose of debugging is to fix those defects or errors. |
| Timing | Testing is done before debugging | Debugging is done after testing. |

| aspects | Testing | Debugging |
|---|---|---|
| categorization basic | it is based on different testing levels i.e unit testing, System testing, integration testing. | debugging is based on different types of bugs |
| methodology | Testing is display of errors | debugging is the detective process |
| Team involve | Testing is done by the tester. | Debugging is done by either programmer ~~or~~ the developer. |

2. Explain various structural testing techniques with suitable examples.

A: Structural testing, also known as white-box testing or glass box testing involves testing the internal structures or working of an application. Here are several structural testing techniques along with examples-

1. Data flow testing-
It focuses on the points at which variables receive values and the points at which these values are used. It aims to test the lifecycle of data in the program-

Eg:-
```
def add (a,b);
    sum=a+b
    return sum
```

for 100% data flow testing,

- a=1, b=2 (definition of sum & its use)
- a=-1, b=-2 (definition of sum and its use with different values)

2. Statement coverage-

It ensures that every possible statement in the code is executed at least once.

Eg:

```
if x>0;
    y=x+1
else
    y=x-1
```

for 100% coverage we need to test-

- x=1 (ensuring y=x+1 is executed)
- x=-1 (ensuring y=x-1 is executed)

3. path coverage-

path coverage ensures that every possible path through the code is executed. It is more rigorous form of testing compared to branch coverage.

Eg:

```
if x>0;
    y=x+1
else:
    if x==0;
        y=0
    else:
```

for 100% path coverage we need to test

- path 1: $x=1$ (if $x>0$)
- path 2: $x=0$ (if $x==0$)
- path 3: $x=-1$ (else, $y=x-1$)

3. What is black box testing? Is it necessary to perform this? Explain various test activities.

A: Black box testing technique mainly deals with functional requirements of the software. Black testing is also known as behavioural testing, Software engineers acquire black-box testing to sets of input conditions which satisfies all functional requirements for a program.

Necessary to perform it -

Yes it is, this type of testing is conducted so as to ensure that the software satisfies its purpose of development. Hence, in this case, the software is exercised in all its functional aspects and is closed analyzed to conclude that its modules, functions to the expectations.

Test activities -

1. Requirement analysis -
- understanding requirements and specifications of the software.
- identifying testable functionalities and features

2. Test planning-
- defining the scope and objectives of testing
- preparing a test plan document.

3. Test case design-
- writing test cases based on requirements.
- ensuring test cases cover all possible scenarios, including positive and negative test cases.

4. Test case review-
- reviewing test cases with peers or stakeholders to ensure completeness and correctness.
- updating test cases based on feedback.

5. Test execution
- Running the test cases on the application.
- Recording the results of each test case, nothing any deviations from expected results.

4. What is difference between verification and validation?

| Verification | Validation |
|---|---|
| Verification means checking the documents, languages, design and other programming things. | Validation means testing the actual product. |
| Verification does not involve the execution of the code. | Validation involves the execution. |

| Verification | Validation |
|---|---|
| It is considered static testing. | It is considered dynamic testing. |
| It has the ability to detect errors quickly. | It can only detect errors that could not be determined by the verification method. |
| It includes checking documents delivered by humans. | It includes the execution of a program executed by a computer. |
| Verification uses methods such as walkthroughs, reviews, inspection. | Validation uses a method such as white box testing, blackbox testing etc. |

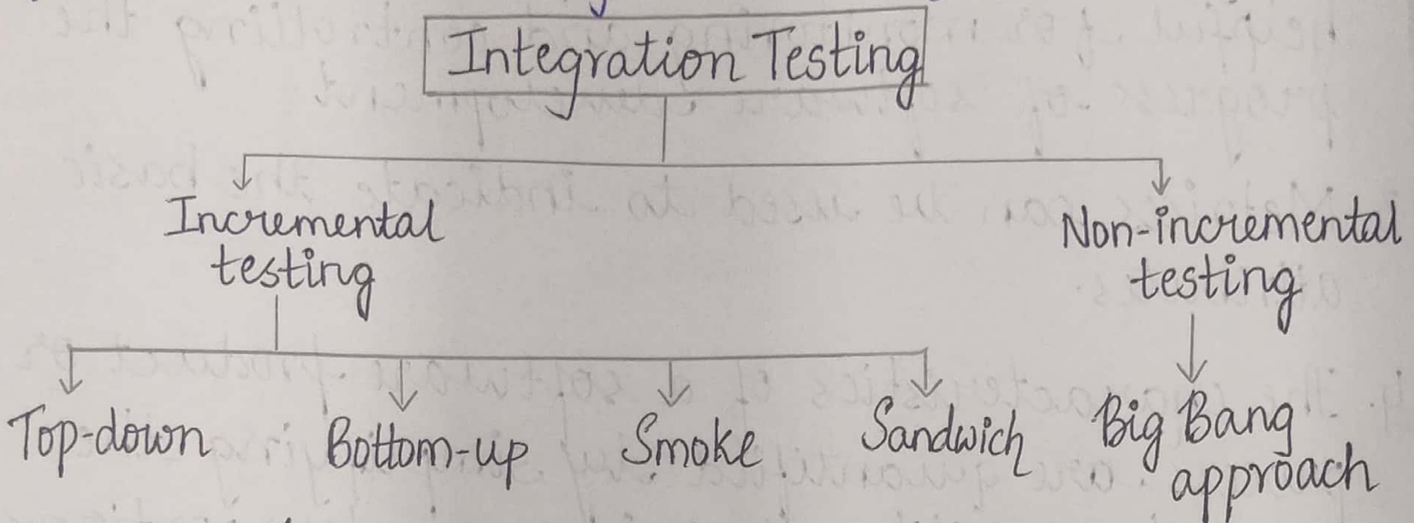
5. What is integration testing? Explain in detail.

A: Completion of unit testing marks the beginning of integration testing, integration testing is done after associating these units into one single architecture. Its main focus of integration testing remains on the collaboration of all units to form one single architecture and then implementing test cases to uncover bugs associated with the interfaces.

Modules when integrated may pose following-

1. data loses
2. increased imprecision errors
3. The purpose of the modules may not be fulfilled.

4. Global variables may worsen the situation etc



Need for Integration testing.
When individual components are already tested using unit testing and all errors are uncovered then the reason for integration testing is to ensure that units the work perfectly in isolation also works perfectly when integrated.

6. Discuss about metrics for testing in detail.

A: Software metrics are quantifiable indexes of the testing procedure, quality, productivity, and overall health. A software metric refers to any type of measurement to improve the process of developing software as well as to understand all aspects of the management of that software.

Importance of software metrics.

1. It is possible to measure all kinds of software in terms of its quantitative analysis.

2. They provide the techniques which are helpful for monitoring and controlling the progress of software development.

3. Metrices can be used to indicate the basic attributes.

4. The characteristics of a software product or process are quantified by simplifying the understandability of various system portions.

5. Metrices are useful to analyse and take decisions for a method and a product in order to accept refuse or develop a software.

7. Write a short note on White Box testing and system testing.

A: White box-testing.

It deals with internal logic and structure of the program code. It is also known as glass-box, structural or open-box testing. Here, test cases are derived based on knowledge of software structures and its implementation. Using white box testing, the tester can detect which module or unit is not functioning properly. This test is performed depending on knowledge of "how" the system is implemented. White box test can analyze the data flow, control flow, information flow, exception and error handling techniques. White box testing is done to check if the implementation is in accordance with the planned design. It is also check the implementation of security functions.

## System Testing.

In a computer based system, there is more than software i.e, hardware, operating system etc. Thus when software is developed it must be integrated with other elements and tested. This type of testing is called system testing, which is not a part of the development life cycle. The only responsibility of software engineer is to sit with system engineer and check if there are any errors due to software development. Even this can be avoided if the software engineer takes extreme care while designing, planning and testing the software.