**MODUAL 2**

1. What is Exploratory Testing

**Exploratory Testing** is a type of software testing where Test cases are not created in advance but testers check system on the fly. They may note down ideas about what to test before test execution. The focus of exploratory testing is more on testing as a “thinking” activity.

1. What is traceability matrix

A Traceability Matrix is a document that co-relates any two-baseline documents that require a many-to-many relationship to check the completeness of the relationship.

It is used to track the requirements and to check the current project requirements are met.

1. What is Boundary value testing

 Boundary value testing is based on testing the boundary values of valid and invalid partitions. The behavior at the edge of the equivalence partition is more likely to be incorrect than the behavior within the partition, so boundaries are an area where testing is likely to yield defects.

It checks for the input values near the boundary that have a higher chance of error. Every partition has its maximum and minimum values and these maximum and minimum values are the boundary values of a partition.

1. What is Equivalence partitioning testing

**Equivalence Partitioning Method** is also known as Equivalence class partitioning (ECP). It is a [software testing](https://www.geeksforgeeks.org/software-testing-basics/) technique or [black-box testing](https://www.geeksforgeeks.org/software-engineering-black-box-testing/) that divides input domain into classes of data, and with the help of these classes of data, test cases can be derived. An ideal test case identifies class of error that might require many arbitrary test cases to be executed before general error is observed.

n equivalence partitioning, equivalence classes are evaluated for given input conditions. Whenever any input is given, then type of input condition is checked, then for this input conditions, Equivalence class represents or describes set of valid or invalid states.

1. What is Integration testing

**Integration Testing** is defined as a type of testing where software modules are integrated logically and tested as a group. A typical software project consists of multiple software modules, coded by different programmers. The purpose of this level of testing is to expose defects in the interaction between these software modules when they are integrated

Integration Testing focuses on checking data communication amongst these modules. Hence it is also termed as **‘I & T’** (Integration and Testing), **‘String Testing’** and sometimes **‘Thread Testing’**.

1. What determines the level of risk

Based on these two dimensions, we determine the level of risk.

***Level of Risk in Software = Probability of Risk Occurring  X  Impact if risk occurred***

We can calculate the probability of risk between 0 - 1 with 0 depicting 0% occurrence and 1 depicting 100% occurrence. In this case, the classification of the impact is Low, Medium, and High. Some folks also classify it as Minimal, Minor, Moderate, Significant, and Severe. For the formula to calculate the level of risk, we can show the impact on a scale of 1 - 10 with 1 being the lowest impact and 10 being the highest impact. We can also use a range of 1-5, but irrespective of that, the core concept remains the same.

1. What is Alpha testing

It is done by internal testers of the organization.

It is an internal test, performed within the organization.

Alpha Testing uses both black box and white box testing techniques

Identifies possible errors.

Developers start fixing bugs as soon as they are identified.

Long execution cycles.

1. What is beta testing

Beta testing is a type of user acceptance testing where the product team gives a nearly finished product to a group of target users to evaluate product performance in the real world.

There is no standard for what a beta test should look like and how to set up beta testing. The actual testing procedure should be relevant to your testing goals.:

1. What is component testing

Component testing, also known as program or module testing, is done after unit testing. In this type of testing those test objects can be tested independently as a component without integrating with other components e.g. modules, classes, objects, and programs. This testing is done by the development team.

1. What is functional system testing

Functional testing is a type of testing that seeks to establish whether each application feature works as per the software requirements. Each function is compared to the corresponding requirement to ascertain whether its output is consistent with the end user’s expectations. The testing is done by providing sample inputs, capturing resulting outputs, and verifying that actual outputs are the same as expected outputs.

1. What is Non-Functional Testing

**Non-Functional Testing** is defined as a type of Software testing to check non-functional aspects (performance, usability, reliability, etc) of a software application. It is designed to test the readiness of a system as per nonfunctional parameters which are never addressed by functional testing.

1. What is GUI Testing

There are two types of interfaces for a computer application. Command Line Interface is where you type text and computer responds to that command. GUI stands for Graphical User Interface where you interact with the computer using images rather than text.

**GUI Testing** is a software testing type that checks the Graphical User Interface of the Software. The purpose of Graphical User Interface (GUI) Testing is to ensure the functionalities of software application work as per specifications by checking screens and controls like menus, buttons, icons, etc.

1. What is Adhoc testing

**Ad hoc Testing** is an informal or unstructured software testing type that aims to break the testing process in order to find possible defects or errors at an early possible stage. Ad hoc testing is done randomly and it is usually an unplanned activity which does not follow any documentation and test design techniques to create test cases.

1. What is load testing

When your software development project is nearing completion, there’s one test that’s essential to understanding its readiness for deployment: load testing. This type of [performance testing](https://www.microfocus.com/en-us/what-is/performance-testing) allows you to determine how your web application will behave during normal and peak load conditions, as well as its breaking point (should it occur below the peak load condition). At its core, load testing is used to confirm that your web application meets your intended performance goals or objectives, which are frequently identified in a service level agreement (SLA).

With load testing, development teams can easily measure and analyze things like:

* Throughout rates, especially those required to support peak load conditions.
* Resource utilization levels.
* Hardware environment performance, such as CPU and RAM.
* Load balancer performance.
* Concurrency issues.
* Software functionality errors under different levels of load.
* Software design flaws.
* How many users the application can handle before breaking.

1. What is stress Testing

**Stress Testing** is a type of software testing that verifies stability & reliability of software application. The goal of Stress testing is measuring software on its robustness and error handling capabilities under extremely heavy load conditions and ensuring that software doesn’t crash under crunch situations. It even tests beyond normal operating points and evaluates how software works under extreme conditions.

1. What is white box testing and list the types of white box testing

**White Box Testing** is a testing technique in which software’s internal structure, design, and coding are tested to verify input-output flow and improve design, usability, and security. In white box testing, code is visible to testers, so it is also called Clear box testing, Open box testing, Transparent box testing, Code-based testing, and Glass box testing.

following are important WhiteBox Testing Techniques:

1. Statement Coverage
2. Decision Coverage
3. Branch Coverage
4. Condition Coverage
5. Multiple Condition Coverage
6. Finite State Machine Coverage
7. Path Coverage
8. Control flow testing
9. Data flow testing

17 .What is black box testing? What are the different black box testing techniques

Black box testing involves testing a system with no prior knowledge of its internal workings. A tester provides an input, and observes the output generated by the system under test. This makes it possible to identify how the system responds to expected and unexpected user actions, its response time, usability issues and reliability issues.

Black box testing is a powerful testing technique because it exercises a system end-to-end. Just like end-users “don’t care” how a system is coded or architected, and expect to receive an appropriate response to their requests, a tester can simulate user activity and see if the system delivers on its promises.

1. Mention what are the categories of defects

Following are the common types of defects that occur during development:

* Arithmetic Defects
* Logical Defects
* Syntax Defects
* Multithreading Defects
* Interface Defects
* Performance Defects

1. Mention what bigbang testing is

Big Bang Integration Testing is an integration testing strategy wherein all units are linked at once, resulting in a complete system. When this type of testing strategy is adopted, it is difficult to isolate any errors found, because attention is not paid to verifying the interfaces across individual units.

1. What is the purpose of exit criteria

Exit criterion is used to determine whether a given test activity has been completed or NOT. Exit criteria can be defined for all of the test activities right from planning, specification and execution.

1. When should "Regression Testing" be performed

Regression testing is done **after functional testing has concluded**, to verify that the other functionalities are working. In the corporate world, regression testing has traditionally been performed by a software quality assurance team after the development team has completed work.

1. What is 7 key principles? Explain in detail
2. Testing shows presence of defects
3. Exhaustive testing is not possible
4. Early testing
5. Defect clustering
6. Pesticide paradox
7. Testing is context dependent
8. Absence of errors fallacy

## 1) Exhaustive testing is not possible

Yes! Exhaustive testing is not possible. Instead, we need the optimal amount of testing based on the risk assessment of the application.

And the million dollar question is, how do you determine this risk?

To answer this let’s do an exercise

In your opinion, Which operation is most likely to cause your Operating system to fail?

I am sure most of you would have guessed, Opening 10 different application all at the same time.

So if you were testing this Operating system, you would realize that defects are likely to be found in multi-tasking activity and need to be tested thoroughly which brings us to our next principle[Defect](https://www.guru99.com/defect-management-process.html)Clustering

## 2) Defect Clustering

Defect Clustering which states that a small number of modules contain most of the defects detected. This is the application of the Pareto Principle to software testing: approximately 80% of the problems are found in 20% of the modules.

By experience, you can identify such risky modules. But this approach has its own problems

If the same tests are repeated over and over again, eventually the same test cases will no longer find new bugs.

## 3) Pesticide Paradox

Repetitive use of the same pesticide mix to eradicate insects during farming will over time lead to the insects developing resistance to the pesticide Thereby ineffective of pesticides on insects. The same applies to software testing. If the same set of repetitive tests are conducted, the method will be useless for discovering new defects.

To overcome this, the test cases need to be regularly reviewed & revised, adding new & different test cases to help find more defects.

Testers cannot simply depend on existing test techniques. He must look out continually to improve the existing methods to make testing more effective. But even after all this sweat & hard work in testing, you can never claim your product is bug-free. To drive home this point, let’s see this video of the public launch of Windows 98

You think a company like MICROSOFT would not have tested their OS thoroughly & would risk their reputation just to see their OS crashing during its public launch!

## 4) Testing shows a presence of defects

Hence, testing principle states that – Testing talks about the presence of defects and don’t talk about the absence of defects. i.e. Software Testing reduces the probability of undiscovered defects remaining in the software but even if no defects are found, it is not a proof of correctness.

But what if, you work extra hard, taking all precautions & make your software product 99% bug-free. And the software does not meet the needs & requirements of the clients.

This leads us to our next principle, which states that- Absence of Error

## 5) Absence of Error – fallacy

It is possible that software which is 99% bug-free is still unusable. This can be the case if the system is tested thoroughly for the wrong requirement. Software testing is not mere finding defects, but also to check that software addresses the business needs. The absence of Error is a Fallacy i.e. Finding and fixing defects does not help if the system build is unusable and does not fulfill the user’s needs & requirements.

To solve this problem, the next principle of testing states that Early Testing

## 6) Early Testing

Early Testing – Testing should start as early as possible in the Software Development Life Cycle. So that any defects in the requirements or design phase are captured in early stages. It is much cheaper to fix a Defect in the early stages of testing. But how early one should start testing? It is recommended that you start finding the bug the moment the requirements are defined. More on this principle in a later training tutorial.

## 7) Testing is context dependent

Testing is context dependent which basically means that the way you test an e-commerce site will be different from the way you test a commercial off the shelf application. All the developed software’s are not identical. You might use a different approach, methodologies, techniques, and types of testing depending upon the application type. For instance testing, any POS system at a retail store will be different than testing an ATM machine.

1. Difference between QA v/s QC v/s Tester

To summarize, Quality assurance is a set of processes that help “avoid” defects and assure quality. While Quality Control is a set of activities that help detect defects and quality issues before the products reach the hands of end customers. Testing is one of the ways of detecting those defects.

23. difference between smoke and sanity testing

Smoke Testing is performed to ascertain that the critical functionalities of the program are working fine. Sanity testing is done at random to verify that each functionality is working as expected.

1. Difference between verification and Validation

Verification is a process of determining if the software is designed and developed as per the specified requirements. Validation is the process of checking if the software (end product) has met the client's true needs and expectations.

1. Explain types of Performance testing

* Load Testing. Load testing is sort of the simplest form of performance testing. ...
* Stress Testing. ...
* Scalability Testing. ...
* Spike Testing. ...
* Soak Testing. ...
* Configuration Testing. ...
* Availability & Resilience Testing.

1. What is Error, Defect, Bug and failure

We can say that a mistake made by a programmer during coding is called an error, an error found during the unit testing in the development phase is called a defect, an error found during the testing phase is called a bug and when an error is found at an end user's end is called as the failure.

1. Difference between Priority and Severity

**Priority is the order in which the developer should resolve a defect whereas Severity is the degree of impact that a defect has on the operation of the product**. Priority is categorized into three types: low, medium, and high whereas Severity is categorized into five types: critical.

**27** .What is Bug Life Cycle

Defect life cycle, also known as Bug Life cycle is **the journey of a defect cycle, which a defect goes through during its lifetime**. It varies from organization to organization and also from project to project as it is governed by the software testing process and also depends upon the tools used.

28.Explain the difference between Functional testing and NonFunctional testing

Functional testing is a type of software testing in which the system is tested against the functional requirements and specifications. Functional testing ensures that the requirements or specifications are properly satisfied by the application. This type of testing is particularly concerned with the result of processing. It focuses on simulation of actual system usage but does not develop any system structure assumptions.

Non-functional testing is a type of software testing that is performed to verify the non-functional requirements of the application. It verifies whether the behavior of the system is as per the requirement or not. It tests all the aspects which are not tested in functional testing.