**Module–2(Manual Testing)**

**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.

* Exploratory Testing is widely used in Agile models and is all about

discovery, investigation, and learning. It emphasizes personal

freedom and responsibility of the individual tester.

**2] What is traceability matrix?**

* To protect against changes you should be able to trace back from every system component to the original requirement that caused its presence.
* A software process should help you keeping the virtual table up-to-date.

• Simple technique may be quite valuable (naming convention)

**3] What is Boundary value testing?**

* Boundary Value Analysis 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

**4] What is Equivalence partitioning testing?**

* Aim is to treat groups of inputs as equivalent and to select one representative input to test them all EP can be used for all Levels of Testing

• Equivalence partitioning is the process of defining the optimum number of tests by:

• Reviewing documents such as the Functional Design Specification and Detailed Design Specification, and identifying each input condition within a function, Selecting input data that is representative of all other data that would likely invoke the same process for that particular condition. If we want to test the following IF statement: “If value is between 1 and 100 (inclusive) (e.g value >=1 and value <=100) Then...”

45 23 12

75 56

-1 99 7 39

0 11 78 69 101

**OUT OF RANG IN RANG OUT OF RANG**

**5] What is Integration testing?**

* Integration Testing - Testing performed to expose defects in the interfaces and in the interactions between integrated components or systems
* Integration Testing is a level of the software testing process where individual units are combined and tested as a group.
* The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing.
* Integration testing tests integration or interfaces between components, interactions to different parts of the system such as an operating system, file system and hardware or interfaces between systems.
* Integration testing is done by a specific integration tester or test team.

• Components may be code modules, operating systems, hardware and even complete systems

• There are 2 levels of Integration Testing

1. Component Integration Testing

2. System Integration Testing

**6] What determines the level of risk?**

Determining the level of risk usually involves trying to assess not only the likelihood of an identified risk from actually occurring, but also the potential magnitude the consequences this risk could have on an organisation and its stakeholder, should it occur.

**7] What is Alpha testing?**

* It is always performed by the developers at the software development site.

• Sometimes it is also performed by Independent Testing Team.

• Alpha Testing is not open to the market and public

• It is conducted for the software application and project.

• It is always performed in **Virtual Environment.**

• It is always performed within the organization

.• It is the form of Acceptance Testing

.• Alpha Testing is definitely performed and carried out at the developing organizations location with the involvement of developers.

* It comes under the category of both White Box Testing and Black Box Testing.

• During this phase, the following will be tested in the application:

1. Spelling Mistakes

2. Broken Links

3. Cloudy Directions

• Alpha Testing is always performed at the time of Acceptance Testing when developers test the product and project to check whether it meets the user requirements or not.

* It is always performed at the developer’s premises in the absence of the users.

• It is considered as the User Acceptance Testing (UAT) which is done at developer’s area.

• Unit testing, integration testing and system testing when combined are known as alpha testing.

**8] What is beta testing?**

* It is always performed by the customers at their own site.
* It is not performed by Independent Testing Team.
* Beta Testing is always open to the market and public.

• It is usually conducted for software product.

• It is performed in Real Time Environment.

• It is always performed outside the organization.

• It is also the form of Acceptance Testing.

• Beta Testing (field testing) is performed and carried out by users or you can say people at their own locations and site using customer data. It is only a kind of Black Box Testing

.• Beta Testing is always performed at the time when software product and project are marketed.

* It is always performed at the user’s premises in the absence of the development team.

• It is also considered as the User Acceptance Testing (UAT) which is done at customers or users area.

* Beta testing can be considered **“pre-release”** testing

.• Pilot Testing is testing to product on real world as well as collect data on the use of product in the classroom.

**9] What is component testing?**

* **Component ( Unit – A minimal software item that can be tested in isolation. It means “A unit is the smallest testable part of software.”**
* **Component Testing – The testing of individual software components.**

• Unit Testing is a level of the software testing process where individual units/components of a software/system are tested. The purpose is to validate that each unit of the software performs as designed.

* Unit testing is the first level of testing and is performed prior to Integration Testing.
* Sometimes known as Unit Testing, Module Testing or Program Testing Component can be tested in isolation – stubs/drivers may be employed Unit testing frameworks, drivers, stubs and mock or fake objects are used to assist in unit testing.
* Test cases derived from component specification (module/program spec)
* Functional and Non-Functional testing

• Unit tests are typically written and run by software developers to ensure that code meet its design and behaves as intended with debugging tool.

**10] What is functional system testing?**

Functional System Testing Functional System Testing : A requirement that specifies a function that a system or system component must perform

* A Requirement may exist as a text document and/or a model

• There is two types of techniques

1. Requirement Based Functional Testing

2. Process Based Testing

• Functional System Testing Functionality As below:

|  |  |
| --- | --- |
| Accuracy | Provision of right or agreed results or effects |
| Interoperability | Ability to interact with specified systems |
| Compliance | Adhere to applicable standards, conventions, regulations or laws |
| Auditability | Ability to provide adequate and accurate audit data |
| Suitability | Presence and appropriateness of functions for specified tasks |

**11] What is Non-Functional Testing?**

Testing of those requirements that do not relate to functionality

**• Emphasis on non-functional requirements:**

• Performance

• Load

• Data volumes

• Storage

• Recovery

• Usability

• Stress

• Security\*

• \* Note that ISTQB treats this as a Functional test. From the syllabus:

**• ‘Security Testing A type of functional testing, security testing, investigates the functions (e.g. a l) relating to detection of threats, such as viruses, from malicious outsiders.’**

The non-functional aspects of a system are all the attributes other than business functionality, and are as important as the functional aspects.

These include:

* the look and feel and ease of use of the system

• how quickly the system performs

• how much the system can do for the user

• It is also about:

• how easy and quick the system is to install

• how robust it is

• how quickly the system can recover from a crash

**12] What is GUI Testing?**

**Graphical User Interface (GUI) testing is the process of testing the system’s GUI of the System under Test. GUI testing involves checking the screens with the controls like menus, buttons, icons, and all types of bars – tool bar, menu bar, dialog boxes and windows etc.**

WHAT DO YOU CHECK IN GUI TESTING?

* Check all the GUI elements for size, position, width, length and acceptance of characters or numbers. For instance, you must be able to provide inputs to the input fields.
* Check you can execute the intended functionality of the application using the GUI Check Error Messages are displayed correctly
* Check for Clear demarcation of different sections on screen

• Check Font used in application is readable

• Check the alignment of the text is proper

• Check the Color of the font and warning messages is aesthetically pleasing

• Check that the images have good clarity

• Check that the images are properly aligned

• Check the positioning of GUI elements for different screen resolution

**13] What is Adhoc testing?**

**Adhoc testing is an informal testing type with an aim to break the system**.

• It does not follow any test design techniques to create test cases.

• In fact is does not create test cases altogether!

• This testing is primarily performed if the knowledge of testers in the system under test is very high.

* Testers randomly test the application without any test cases or any business requirement document.
* Adhoc Testing does not follow any structured way of testing and it is randomly done on any part of application. Main aim of this testing is to find defects by random checking.

• Adhoc testing can be achieved with the testing technique called Error Guessing.

• Error guessing can be done by the people having enough experience on the system to “guess” the most likely source of errors.

* The Error guessing is a technique where the experienced and good testers are encouraged to think of situations in which the software may not be able to cope.
* Some people seem to be naturally good at testing and others are good testers because they have a lot of experience either as a tester or working with a particular system and so are able to find out its weaknesses.
* This is why an error guessing approach, used after more formal techniques have been applied to some extent, can be very effective.
* It also saves a lot of time because of the assumptions and guessing made by the experienced testers to find out the defects which otherwise won’t be able to find.
* Using experience to postulate errors.
* Use Error Guessing to Complement Test Design Techniques.

**14] What is load testing?**

* Introduction Load testing - Its a performance testing to check system behavior under load.
* Testing an application under heavy loads, such as testing of a web site under a range of loads to determine at what point the system’s response time degrades or fails. Load testing is a kind of performance testing which determines a system’s performance

• under real-life load conditions. This testing helps determine how the application behaves when multiple users access it simultaneously.

**This testing usually identifies** –

* The maximum operating capacity of an application
* Determine whether current infrastructure is sufficient to run the application

• Sustainability of application with respect to peak user load

* Number of concurrent users that an application can support, and scalability to allow more users to access it.
* It is a type of non-functional testing.
* Load testing is commonly used for the Client/Server, Web based applications – both Intranet and Internet.

**15] What is stress Testing?**

* **Stress testing - System is stressed beyond its specifications to check how and when it fails.**
* Performed under heavy load like putting large number beyond storage capacity, complex database queries, continuous input to system or database load.
* Stress testing is used to test the stability & reliability of the system. This test mainly determines the system on its robustness and error handling under extremely heavy load conditions.
* It even tests beyond the normal operating point and evaluates how the system workS under those extreme conditions.
* Stress Testing is done to make sure that the system would not crash under crunch situations.
* Stress testing is also known as endurance testing.
* Under Stress Testing, AUT is be stressed for a short period of time to know its withstanding capacity.
* Most prominent use of stress testing is to determine the limit, at which the system or software or hardware breaks.

**16]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

**17] What is black box testing?**

Black Box Testing is a software testing method in which the functionalities of software applications are tested without having knowledge of internal code structure, implementation details and internal paths.

Black Box Testing mainly focuses on input and output of software applications and it is entirely based on software requirements and specifications.

It is also known as Behavioral Testing.

**18] What are the different black box testing techniques?**

**Equivalence Class Testing**: It is used to minimize the number of possible test cases to an optimum level while maintains reasonable test coverage.

**Boundary Value Testing**: Boundary value testing is focused on the values at boundaries. This technique determines whether a certain range of values are acceptable by the system or not. It is very useful in reducing the number of test

cases. It is most suitable for the systems where an input is within certain ranges.

**Decision Table Testing**: A decision table puts causes and their effects in a matrix. There is a unique combination in each column.

**19] Mention what are the categories of defects?**

CATEGORIES OF DEFECTS

1 2 3 4

1. Error of commission
2. Error of omission
3. Error of clarity
4. Error of speed of capacity

**20] Mention what bigbang testing is?**

* In Big Bang integration testing all components or modules is integrated simultaneously, after which everything is tested as a whole.
* Big Bang testing has the advantage that everything is finished before integration testing starts.
* The major disadvantage is that in general it is time consuming and difficult to trace the cause of failures because of this late integration.
* Here all component are integrated together at once, and then tested.

**21] 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. Exit criterion should be part of test plan and decided in the planning stage.

**22] When should "Regression Testing" be performed?**

**Regression Testing: Testing of a previously tested program following modification to ensure that defects have not been introduced or uncovered in unchanged areas of the software, as a result of the changes made. It is performed when the software or its environment is changed.**

**Retest All**

• This is one of the methods for regression testing in which all the tests in the existing test bucket or suite should be re-executed. This is very expensive as it requires huge time and resources.

**Regression Test Selection**

• Instead of re-executing the entire test suite, it is better to select part of test suite to be run Test cases selected can be categorized as

1) Reusable Test Cases

2) Obsolete Test Cases.

* Re-usable Test cases can be used in succeeding regression cycles

.• Obsolete Test Cases can’t be used in succeeding cycles.

**• Prioritization Of Test Cases**

• Prioritize the test cases depending on business impact, critical & frequently used functionalities. Selection of test cases based on priority will greatly reduce the regression test suite.

**23] What is 7 key principles? Explain in detail?**

**7 Key Principles of Testing**

1. Testing shows presence of Defects
2. Exhaustive Testing is Impossible!
3. Early Testing
4. Defect Clustering
5. The Pesticide Paradox
6. Testing is Context Dependent
7. Absence of Errors Fallacy

**DETAILS**

1. Testing shows presence of Defects

* Testing can show that defects are present, but cannot prove that there are no defects.
* 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.
* We test to find Faults As we find more defects, the probability of undiscovered defects remaining in a system reduces.

2) Exhaustive Testing is Impossible!

* Testing everything including all combinations of inputs and preconditions is not possible.
* So, instead of doing the exhaustive testing we can use risks and priorities to focus testing efforts.
* **For example**: In an application in one screen there are 15 input fields, each having 5

possible values, then to test all the valid combinations you would need 30 517 578 125 (515) tests.

* This is very unlikely that the project timescales would allow for this number of tests.
* So, accessing and managing risk is one of the most important activities and reason for testing in any project.
* We have learned that we cannot test everything (i.e. all combinations of inputs and preconditions).
* That is we must Prioritise our testing effort using a Risk Based Approach.

3) Early Testing

* Testing activities should start as early as possible in the software or system development life cycle, and should be focused on defined objectives.
* Testing activities should start as early as possible in the development life cycle

• These activities should be focused on defined objectives – outlined in the Test Strategy Remember from our Definition of Testing, that Testing doesn’t start once the code has been written!

1. Defect Clustering

* A small number of modules contain most of the defects discovered during pre-release testing, or are responsible for the most operational failures.
* Defects are not evenly spread in as ystem They are ‘clustered’ In other words, most defects found during testing are usually confined to a small number of modules Similarly, most operational failures of a system are usually confined to a small number of modules

1. The Pesticide Paradox

* If the same tests are repeated over and over again, eventually the same set of test cases will no longer find any new defects.
* To overcome this “pesticide paradox”, the test cases need to be regularly reviewed and revised, and new and different tests need to be written to exercise different parts of the software or system to potentially find more defects.
* Testing identifies bugs, and programmers respond to fix them As bugs are eliminated by the programmers, the software improves As software improves the effectiveness of previous tests erodes Therefore we must learn, create and use new tests based on new techniques to catch new bugs
* Testing is done differently in different contexts
* Different kinds of sites are tested differently.

1. Testing is Context Dependent

* Testing is basically context dependent.
* Testing is done differently in different contexts
* Different kinds of sites are tested differently.
* For example Safety – critical software is tested differently from an e-commerce site.

1. Absence of Errors Fallacy

* If the system built is unusable and does not fulfill the user’s needs and expectations then finding and fixing defects does not help.
* If we build a system and, in doing so, find and fix defects....
* It doesn’t make it a good system
* Even after defects have been resolved it may still be unusable and/or does not fulfil the users’

**24] Difference between QA v/s QC v/s Tester**

|  |  |  |  |
| --- | --- | --- | --- |
| S.N. | **Quality Assurance** | | **Quality Control** |
| **1** | Activities which ensure the implementation of processes, procedures and standards in context to verification of developed software and intended requirements. | | Activities which ensure the verification of developed software with respect to documented (or not in some cases) requirements. |
| Focuses on processes and procedures rather than conducting actual testing on the system. | | Focuses on actual testing by executing Software with intend to identify bug/defect through implementation of procedures and process. | | |
| **3** | Process oriented activities. | | Product oriented activities. |
| **4** | Preventive activities. | | It is a corrective process. |
| **5** | It is a subset of Software Test Life Cycle (STLC). | | QC can be considered as the subset of Quality Assurance. |

**25] Difference between Smoke and Sanity?**

Smoke Testing vs Sanity Testing

|  |  |
| --- | --- |
| SMOKE TESTING | SANITY TESTING |
| Smoke Testing is performed to ascertain that the critical functionalities of the program is working fine | Sanity Testing is done to check the new functionality / bugs have been fixed |
| The objective of this testing is to verify the "stability" of the system in order to proceed with more rigorous testing | The objective of the testing is to verify the "rationality" of the system in order to proceed with more rigorous testing |
| This testing is performed by the developers or testers | Sanity testing is usually performed by testers |
| Smoke testing is usually documented or scripted | Sanity testing is usually not documented and is unscripted |
| Smoke testing is a subset of Regression testing | Sanity testing is a subset of Acceptance testing |
| Smoke testing exercises the entire system from end to end | Sanity testing exercises only the particular component of the entire system |
| Smoke testing is like General Health Check Up | Sanity Testing is like specialized health check up |

**26]Difference between verification and Validation Explain types of Performance testing?**

|  |  |  |
| --- | --- | --- |
| Criteria | Verification | Validation |
| Definition | The process of evaluating work-products (not the actual final product) of a development phase to determine whether they meet the specified requirements for that phase. | The process of evaluating software during or at the end of the development process to determine whether it satisfies specified business requirements. |
| Objective | To ensure that the product is being built according to the requirements and design specifications. In other words, to ensure that work products meet their specified requirements. | To ensure that the product actually meets the user’s needs, and that the specifications were correct in the first place. In other words, to demonstrate that the product fulfills its intended use when placed in its intended environment. |
| Question | Are we building the product right? | Are we building the right product? |
| Evaluation Items | Plans, Requirement Specs, Design Specs, Code, Test Cases | The actual product/software. |
| Activities | * Reviews   • Walkthroughs  • Inspections | * Testing |

**27] What is Error, Defect, Bug and failure?**

**Error**

An error is a mistake made by a human that leads to a discrepancy between the actual and the expected result.

**Defect**

A defect is a problem in the functioning of a software system during testing. ISTQB defines a defect as “A flaw in a component or system that can cause the component or system to fail to perform its required function, e.g., an incorrect statement or data definition.”

**Bug**

A bug is a flaw in a software system that causes the system to behave in an

unintended manner.

**Failure**

A failure is the inability of a software system to perform its operations within the specified performance benchmark. As per ISTQB, “a defect, if encountered during execution, may cause a failure of the component or system”.

**28] 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, major, moderate, minor and cosmetic.

Priority is associated with scheduling while Severity isassociated with functionality or standards.

**29] What is Bug Life Cycle?**

**program that prevents it from working correctly or produces an incorrect result. Bugs arise from mistakes and errors, made by people, in either a program’s source code or its design.” “A computer bug is an error, flaw, mistake, failure, or fault in a computer**

* The duration or time span between the first time defects is found and the time that it is closed successfully, rejected, postponed or deferred is called as ‘Defect Life Cycle’.
* When a bug is discovered, it goes through several states and eventually reaches one of the terminal states, where it becomes inactive and closed.
* The process by which the defect moves through the life cycle is depicted next slide.

**Bug (Defect) Life Cycle**

NEW

REJECTED

OPEN

ASSIGN

DEFERRED

REOPENED

TEST

VERIFIED

CLOSED

* As you can see from above diagram, a defect‘s state can be divided into Open or Closed.

• When a bug reaches one of the Closed or Terminal states, its lifecycle ends. Each state hasone or more valid states to move to.

* This is to ensure that all necessary steps are taken to resolve or investigate that defect. For example, a bug should not move from Submitted state to resolved state without having it open.
* In a typical scenario, as soon as a bug is identified, it is logged into the bug tracking system with status as Submitted. After ascertaining the validity of the defect, it is given the “Open” Status.

**30] Explain the difference between Functional testing and NonFunctional testing?**

|  |  |
| --- | --- |
| **Functional Testing** | **Non-functional Testing** |
| It verifies the operations and actions  of an application. | It verifies the behavior of an application. |
| It is based on requirements of  customer. | It is based on expectations of customer. |
| It helps to enhance the behavior of  the application. | It helps to improve the performance of the  application. |
| Functional testing is easy to execute  manually. | It is hard to execute non-functional testing manually. |
| It tests what the product does. | It tests what the product does. |
| It tests what the product does. | It tests what the product does. |
| Examples:   1. Unit testing 2. Smoke testing 3. Integretion testing 4. Regression testing | Examples:   1. Performance testing 2. Load testing 3. Stress testing 4. Scalability testing |

**32) What is the difference between the STLC (Software Testing Life Cycle) and SDLC (Software Development Life Cycle)?**

|  |  |
| --- | --- |
| SDLC | STLC |
| SDLC is mainly related to software  development. | STLC is mainly related to software testing. |
| Besides development other phases  like testing is also included. | It focuses only on testing the software. |
| SDLC involves total six phases or  steps. | STLC involves only five phases or steps. |
| In SDLC, more number of members  (developers) are required for the  whole process. | In STLC, less number of members (testers)  are needed. |
| In SDLC, development team makes  the plans and designs based on the  requirements. | In STLC, testing team(Test Lead or Test  Architect) makes the plans and designs. |
| Goal of SDLC is to complete  successful development of software. | Goal of STLC is to complete successful  testing of software. |
| It helps in developing good quality  software. | It helps in making the software defects free. |
| SDLC phases are completed before  the STLC phases. | STLC phases are performed after SDLC  phases. |
| Post deployment support ,  enhancement , and update are to be  included if necessary. | Regression tests are run by QA team to check deployed maintenance code and maintains test cases and automated scripts. |
| Creation of reusable software  systems is the end result of SDLC. | A tested software system is the end result of  STLC. |