**Software testing assignment**

**Module – 2 (Manual Testing)**

**Q-1 : What is Exploratory Testing?**

**Ans :** Exploratory testing is a concurrent process where,

* Test design, execution and logging happen simultaneously
* Testing is often not recorded
* Makes use of experience, heuristics and test patterns

Though the current trend in testing is to push for automation, exploratory testing is a new way of thinking

**Q-2 : What is traceability matrix?**

**Ans :** 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.

**Q-3 : What is Boundary value testing?**

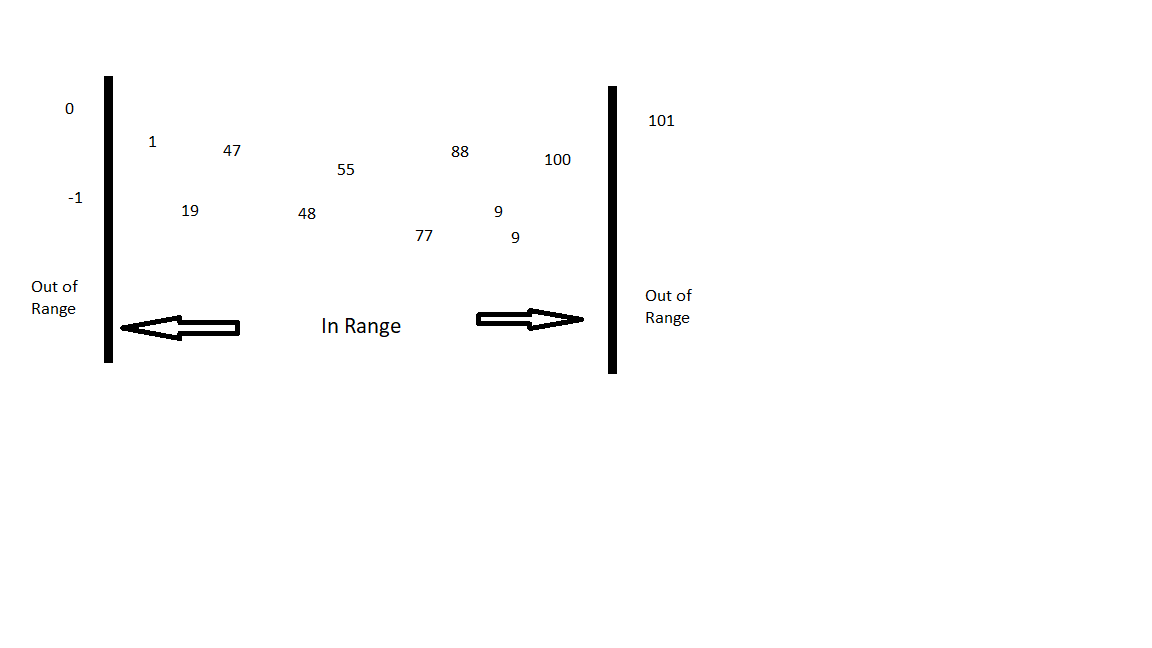
**Ans :**

* Boundary value analysis is a methodology for designing test cases that concentrates software testing effort on cases near the limits of valid ranges
* Boundary value analysis is a method which refines equivalence partitioning.
* Boundary value analysis generates test cases that highlight errors better than equivalence partitioning.

**Q-4 : What is Equivalence partitioning testing?**

**Ans :** 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

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...”

We could put a range of numbers as shown in the below figure.

* EP says that by testing just one value we have tested the partition (typically a mid-point value is used). It assumes that:
  + If one value finds a bug, the others probably will too
  + If one doesn't find a bug, the others probably won't either

**Q-5 : What is Integration testing?**

**Ans :** 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.

**Q-6 : What is Alpha testing?**

**Ans :**

* 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.

**Q-7 : What is beta testing?**

**Ans :**

* 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.
* 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.

**Q-8 : What is component testing?**

**Ans :** 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.

**Q-9 : What is functional system testing?**

**Ans :** 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 Test Approach
  + Requirement Based Functional Testing
  + 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 |

**Q-10 : What is Non-Functional Testing?**

**Ans:** Non-Functional Testing: Testing the attributes of a component or system that do not relate to functionality, e.g. reliability, efficiency, usability, interoperability, maintainability and portability

* It is the testing of “how” the system works. Non-functional testing may be performed at all test levels.
* The term non-functional testing describes the tests required to measure characteristics of systems and software that can be quantified on a varying scale, such as response times for performance testing.

**Q-11 : What is GUI Testing?**

**Ans :** 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.

**Q-12 : What is Adhoc testing?**

**Ans :** 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.
* Main aim of this testing is to find defects by random checking. Adhoc testing can be achieved with the testing technique called Error Guessing.

**Q-13 : What is load testing?**

**Ans :**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.

**Q-14 : What is stress Testing?**

**Ans :** 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.

**Q-15 : What is white box testing and list the types of white box testing?**

**Ans :** White Box Testing: Testing based on an analysis of the internal structure of the component or system.

* Structure-based testing technique is also known as ‘white-box’ or ‘glass-box’ testing technique because here the testers require knowledge of how the software is implemented, how it works.

White box testing can take several forms:

* Unit testing — tests written as part of the application code, which test that each component is working as expected.
* Mutation testing — a type of unit testing that checks the robustness and consistency of the code by defining tests, making small, random changes to the code and seeing if the tests still pass.
* Integration testing — tests specifically designed to check integration points between internal components in a software system, or integrations with external systems.
* White box penetration testing — an ethical hacker acts as a knowledgeable insider, attempting to attack an application based on intimate knowledge of its code and environment.
* Static code analysis — automatically identifying vulnerabilities or coding errors in static code, using predefined patterns or machine learning analysis.

**Q-16 : What is black box testing? What are the different black box testing techniques?**

**Ans :** Black-box testing: Testing, either functional or non-functional, without reference to the internal structure of the component or system.

* The testers have no knowledge of how the system or component is structured inside the box. In black-box testing the tester is concentrating on what the software does, not how it does it.
* There are four specification-based or black-box technique:
  + Equivalence partitioning
  + Boundary value analysis
  + Decision tables
  + State transition testing
  + Use-case Testing

**Q-17 : Mention what are the categories of defects?**

**Ans :1)** Security Defects: Application security defects generally involve improper handling of data sent from the user to the application. These defects are the most severe and given highest priority for a fix.

* Examples: Authentication: Accepting an invalid username/password
  + Authorization: Accessibility to pages though permission not given

2)User Interface Defects: As the name suggests, the bugs deal with problems related to UI are usually considered less severe.

* Examples: Improper error/warning/UI messages
  + Spelling mistakes
  + Alignment problems

3)Data Quality/Database Defects: Deals with improper handling of data in the database.

* Examples: Values not deleted/inserted into the database properly
  + Improper/wrong/null values inserted in place of the actual values

4)Critical Functionality Defects: The occurrence of these bugs hampers the crucial functionality of the application. Examples: - Exceptions

5)Functionality Defects: These defects affect the functionality of the application.

* Examples: All JavaScript errors
  + Buttons like Save, Delete, Cancel not performing their intended functions
  + A missing functionality (or) a feature not functioning the way it is intended to
  + Continuous execution of loops

**Q-18 : Mention what bigbang testing is?**

**Ans :** 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.

**Q-19 : What is the purpose of exit criteria?**

**Ans :**  Exit criteria ensure that the software being tested meets the quality standards set by the project stakeholders. This includes requirements such as functionality, performance, reliability, usability, security, and other quality attributes.

* Exit criteria provide a basis for decision-making regarding whether to proceed to the next phase of testing, release the software, or continue testing. Meeting the exit criteria indicates that the software has undergone sufficient testing and is ready for the next steps in the development lifecycle.
* Exit criteria help optimize resource allocation by defining when testing activities can be concluded. This ensures that testing efforts are focused on areas of the software that require additional attention and that resources are not wasted on unnecessary testing..

**Q-20 : When should "Regression Testing" be performed?**

**Ans :**

* Change in requirements and code is modified according to the requirement
* New feature is added to the software
* Defect fixing
* Performance issue fix

**Q-21 : What is 7 key principles? Explain in detail?**

**Ans :**

1. Testing shows presence of 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.

* As we find more defects, the probability of undiscovered defects remaining in a system reduces.

1. 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.

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.

4. 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 a system They are ‘clustered’

5. 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.

6. 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.
  + Whilst, Testing can be 50% of development costs, in NASA's Apollo program it was 80% testing
  + 3 to 10 failures per thousand lines of code (KLOC) typical for commercial software
  + 1 to 3 failures per KLOC typical for industrial software
  + 0.01 failures per KLOC for NASA Shuttle code!
  + Also different industries impose different testing standards

7. 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.

* Even after defects have been resolved it may still be unusable and/or does not fulfil the users’ needs and expectations

**Q-22 : Difference between QA v/s QC v/s Tester**

**Ans :**

|  |  |  |
| --- | --- | --- |
| **QA** | **QC** | **Tester** |
| 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. | Activities which ensure the identification of bugs/error/defects in the Software. |
| 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. | Focuses on actual testing. |
| Process oriented activities. | Product oriented activities | Product oriented activities. |
| Preventive activities. | It is a corrective process | It is a preventive process. |
| It is a subset of Software Test Life Cycle (STLC). | QC can be considered as the subset of Quality Assurance. | Testing is the subset of Quality Control. |

**Q-23 : Difference between Smoke and Sanity?**

**Ans :**

|  |  |
| --- | --- |
| **Smoke testing** | **Sanity testing** |
| Smoke Testing is performed to ascertain the that the critical functionalities of the program is working fine | Sanity Testing is done to check new functionality / bugs have been fixed |
| Smoke testing is done to assure that the acute functionalities of program is working fine. | Sanity testing is done to check the bugs have been fixed after the build. |
| Smoke testing is also called subset of acceptance testing. | Smoke testing is also called subset of acceptance testing. |
| Smoke testing is documented. | Sanity testing isn’t documented. |
| Smoke testing is performed by either developers or testers. | Sanity testing is normally performed by testers. |
| Smoke testing is done to measure the stability of the system/product by performing testing. | Sanity testing is done to measure the rationality of the system/product by performing testing. |

**Q-24 : Difference between verification and Validation**

**Ans :**

|  |  |  |
| --- | --- | --- |
| 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 |

**Q-25 : Explain types of Performance testing.**

**Ans :** Here the types of performance testing :

1. 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.
2. 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.
3. Endurance testing : Stress testing is also known as endurance testing.
4. Spike testing : a type of performance testing in which an application receives a sudden and extreme increase or decrease in load.
5. Volume testing : a type of load testing that measures the application's ability to scale up or down as a reaction to an increase in the number of users.
6. Scalability testing : a type of load testing that measures the application's ability to scale up or down as a reaction to an increase in the number of users.

**Q-26 : What is Error, Defect, Bug and failure?**

**Ans :** A mistake in coding is called error, error found by tester is called defect, defect accepted by development team then it is called bug, build does not meet the requirements then it is failure.

**Q-27 : Explain the difference between Functional testing and NonFunctional testing**

**Ans :**

|  |  |
| --- | --- |
| **Functional testing** | **Non-Functional testing** |
| Functional testing is performed using the functional specification provided by the client and verifies the system against the functional requirements | Non-Functional testingchecksthe Performance, reliability, scalability and other non-functional aspects |
| Functional testing is executed first | Non functional testing should be performed after functional testing |
| Manual testing or automation tools can be used for functional | testing Using tools will be effective for this testing |
| Business requirements are the inputs to functional testing | Performance parameters like speed , scalability are inputs to non-functional testing. |
| Functional testing describes what the product does | Nonfunctional testing describes how good the product works |
| Easy to do manual testing | Tough to do manual testing |
| Types of Functional testing are ∙  Unit Testing  ∙ Smoke Testing  ∙ Sanity Testing  ∙ Integration Testing  ∙ White box testing  ∙ Black Box testing  ∙ User Acceptance testing  ∙ Regression Testing | Types of Nonfunctional testing are  ∙ Performance Testing  ∙ Load Testing  ∙ Volume Testing  ∙ Stress Testing  ∙ Security Testing  ∙ Installation Testing  ∙ Penetration Testing  ∙ Compatibility Testing  ∙ Migration Testing |

**Q-28 : What is the difference between the STLC (Software Testing Life Cycle) and SDLC (Software Development Life Cycle)?**

**Ans :**

|  |  |
| --- | --- |
| 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. |

**Q-29 : What is the difference between test scenarios, test cases, and test script?**

**Ans :** Test Scenarios: A Test Scenario is any functionality that can be tested. It is also called Test Condition or Test Possibility.

Test Cases: It is a document that contains the steps that has to be executed, it has been planned earlier.

Test Script: It is written in a programming language and it's a short program used to test part of functionality of the software system. In other words a written set of steps that should be performed manually.

**Q-30 : Explain what Test Plan is? What is the information that should be covered.**

**Ans :** A document describing the scope, approach, resources and schedule of intended test activities

A test plan is a detailed document which describes software testing areas and activities. It outlines the test strategy, objectives, test schedule, required resources (human resources, software, and hardware), test estimation and test deliverables. The test plan is a base of every software's testing.

**Q-31 : What are the different Methodologies in Agile Development Model?**

**Ans :**

* Customer satisfaction through early and continuous software delivery – Customers are happier when they receive working software at regular intervals, rather than waiting extended periods of time between releases.
* Accommodate changing requirements throughout the development process – The ability to avoid delays when a requirement or feature request changes.
* Frequent delivery of working software – Scrum accommodates this principle since the team operates in software sprints or iterations that ensure regular delivery of working software.
* Collaboration between the business stakeholders and developers throughout the project – Better decisions are made when the business and technical team are aligned.
* Support, trust, and motivate the people involved – Motivated teams are more likely to deliver their best work than unhappy teams.
* Enable face-to-face interactions – Communication is more successful when development teams are co-located
* Working software is the primary measure of progress – Delivering functional software to the customer is the ultimate factor that measures progress.
* Agile processes to support a consistent development pace – Teams establish a repeatable and maintainable speed at which they can deliver working software, and they repeat it with each release.
* Attention to technical detail and design enhances agility – The right skills and good design ensures the team can maintain the pace, constantly improve the product, and sustain change
* Simplicity – Develop just enough to get the job done for right now.
* Self-organizing teams encourage great architectures, requirements, and designs – Skilled and motivated team members who have decision-making power, take ownership, communicate regularly with other team members, and share ideas that deliver quality products
* Regular reflections on how to become more effective – Self-improvement, process improvement, advancing skills, and techniques help team members work more efficiently

**Q-32 : Explain the difference between Authorization and Authentication in Web testing.What are the common problems faced in Web testing?**

**Ans :**

|  |  |
| --- | --- |
| **Authentication** | **Authorization** |
| Authentication verifies who the user is. | Authorization determines what resources a user can access. |
| Authentication works through password, one-time pins, biometric information, and other information provided or entered by the user. | Authorization works through settings that are implemented and maintained by the organization. |
| Authentication is the first step of a good identity and access management process. | Authorization always takes place after authentication. |
| Authentication is visible to and partially changeable by the user. | Authorization isn’t visible to or changeable by the user. |
| Example: By verifying their identity, employees can gain access to a human resources (HR) application that includes their personal pay information, vacation time, and 401K data. | Example: Once their level of access is authorized, employees and HR managers can access different levels of data based on the permissions set by the organization. |

**Q-33 : When to used Usablity Testing?**

**Ans :** Usability testing is typically used during the design and development phases of a product or service, but it can also be valuable during post-launch phases to gather feedback for improvements.

**Q-34 : What is the procedure for GUI Testing?**

**Ans :**

* 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.