**Software Testing Assignment**

Module – 2 (Manual Testing)

Q-1. What is Software testing?

Ans. Software testing is executing a system in order to identify any gaps, errors or missing requirements in contrary to the actual desire or expectation.

Q-2. What is traceability matrix?

Ans. Traceability matrix is a table which is used to trace the requirements during the software development life cycle. It can be used for forward tracing or backward. There are many user defined templates for RTM.

Q-3. What is Equivalence partitioning testing?

Ans.

Q-4. 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.

Q-5 What is determines the level of risk?

Ans.

Q-6. What is alpha testing?

Ans. It is always performed by the developers at the software development site.It is always performed in Virtual Environment.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.

Q-7. What is Beta testing?

Ans. It is always performed by the customers at their own site.It is performed in Real Time Environment.Beta Testing is always performed at the time when software product and project are marketed.Beta testing can be considered “pre-release” testing.

Q-8. What is component testing?

Ans. 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 is a requirement that specifies a function that a system or system component must perform.

Q-10. Mention what big bang 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.

• Here all components are integrated together at once, and then tested.

Q-11. What is the purpose of exit criteria?

Ans. • Purpose of exit criteria is to define when we STOP testing either at the:

• End of all testing – i.e. product Go Live

• End of phase of testing (e.g. hand over from System Test to UAT)

Q-12. What is 7 key principles? Explain in detail?

Ans. 7 key principles

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
8. **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.

•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.

•That is we must Prioritise our testing effort using a Risk Based Approach.

1. **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.

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 a system

• They are ‘clustered’

• In other words, most defects found during testing 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.

1. **Testing is Context Dependent:**

• Testing is basically context dependent.

•Testing is done differently in different contexts

• Different kinds of sites are tested differently.

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

Q-13. Difference between QA v/s QC v/s Tester.

|  |  |  |  |
| --- | --- | --- | --- |
| No. | QA | QC | TESTER |
| 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 requirements. | Activities which ensure the identification of bugs/error/defects in the Software. |
| 2 | 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. |
| 3 | Process oriented activities. | Product oriented activities. | Product oriented activities. |
| 4 | Preventive activities. | It is a corrective process | It is preventive process |
| 5 | 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-14. Difference between verification and validation?

Ans.

|  |  |  |
| --- | --- | --- |
| **criteria** | **Verification** | **Validation** |
| Definition | The process of evaluating work-products 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. |
| Questions | Are we building the product right? | Are we building the right product? |
| Evaluation items | Plans  Requirements specs  Design specs  Code  Test cases | The actual product/software |
| Activities | Reviews  Walkthroughs  inspections | testing |

Q-15. 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.

Error: A discrepancy between a computed, observed, or measured value or condition and the true, specified, or theoretically correct value or condition.

Defect: Commonly refers to several troubles with the software products, with its external behaviour or with its internal features.

Bug: A fault in a program which causes the program to perform in an unintended or unanticipated manner. See: anomaly, defect, error, exception, and fault. Bug is terminology of Tester.

Failure: The inability of a system or component to perform its required functions within specified performance requirements. See: bug, crash, exception, and fault.

Q-16. Explain the difference between Functional testing and Non functional testing?

Ans.

|  |  |  |
| --- | --- | --- |
| No. | Functional testing | Non functional testing |
| 1 | Functional testing is performed using the functional specification provided by the client and verifies the system against the functional requirements. | Non-Functional testing checks the Performance, reliability, scalability and other non-functional aspects of the software system. |
| 2 | Manual testing or automation tools can be used for functional testing | Using tools will be effective for this testing |
| 3 | Functional testing is executed first | Non functional testing should be performed after functional testing |
| 4 | Functional testing describes what the product does | Nonfunctional testing describes how good the product works |
| 5 | Easy to do manual testing | Tough to do manual testing |

**Q-17.** What is Adhoc testing?

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

\* 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-18. What is Boundary value testing?

Ans. This is one of the software testing technique in which the test cases are designed to include

values at the boundary.

\* If the input data is used within the boundary value limits, then it is said to be Positive

Testing.

\* If the input data is picked outside the boundary value limits, then it is said to be Negative

Testing.

**Q-19.** What is Equivalence partitioning testing?

**Ans**. This is a software testing technique which divides the input date into many partitions .

\* Values from each partition must be tested at least once. Partitions with valid values are used

for Positive Testing.

\* While, partitions with invalid values are used for negative testing.

Q-20. What is non-functional testing?

Ans**.** Testing of those requirements that do not relate to functionality

\* non-functional requirements:

\* Performance

\* Load

\* Data volumes

\* Storage

\* Recovery

\*Usability

\* Stress

\* Security

Q-21. 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 Techniques.**

- Branch Condition testing

- Branch Condition Combination testing

- Modified Condition Decision testing

-Dataflow testing

Q-22. what is black box testing? What are the different black box testing technique?

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

* **Techniques of Black Box Testing**

- Equivalence partitioning

- Boundary value analysis

- Decision tables

- State transition testing

- Use-case Testing

- Other Black Box Testing

- Syntax or Pattern Testing

**Q-23.** When should “regression testing” be perfomed?

Ans. when the system is stable and the system or the environment changes

- when testing bug-fix releases as part of the maintenance phase

- It should be applied at all Test Levels

- It should be considered complete when agreed completion criteria for regression testing have been met

- Regression test suites evolve over time and given that they are run frequently are ideal candidates for automation

Q-24. Difference between smoke and sanity .

|  |  |
| --- | --- |
| **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 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 usually documented  or scripted | Sanity testing is usually not documented and  is unscripted |