INTRODUCTION TO SOFTWARE TESTING'

1.1 Software Unit>>>>>

1) What is bug?

- A software bug is an error, flaw, or fault in the design, development or operation of computer software that causes it to produce an incorrect or unexpected result, or to behave in unintended ways.
- The process of finding and correcting bugs is termed 'debugging' and often uses formal techniques or tools to pingpoint bugs.
- Raised by – The test Engineers submit the bug.
- Types – Algorithm bugs || Resource bugs
- Reason – Missing Coding || Wrong Coding || Extra Coding

2) What is Defects ?

- The defect is the difference between the actual outcome and expected outcome.
- Raised by – The tester identify the defects and it was also solved by developer.
- Types – High level || Low level || Medium Level defects

- Reason – Giving incorrect and wrong inputs || An error in coding or logic affect the software.

3) What is error ?

- An error is a mistake made in the code, that's why we cannot execute or compile code.
- Raised by – The developers and automation test engineers.
- Types – Syntactic Error || Flow control Error || Testing Error || Hardware Error
- Reason – Confusions and Issues in programming || Inconsistency btwn actual and expected outcome || Blunders in design or requirement actions.

4) What is Fault ?

- The fault is a state that causes the software to fail to accomplish its essential function.
- Raised by – Human mistakes cause fault.
- Types – Business logic fault || Functional and logical fault || Security fault || Software/Hardware fault.
- Reason – Improper step in the initial stage, process, or data definition || Inconsistency or issuse in the program.

5) What is Failure ?

- If the software has lots of defects, it lead to failure or cause failure.
- Raised by – The manual test engineer through the development cycle.
- Types –

- Reason – Environmental condition || System usage users || Human error

6) What do bugs occur ?
   - There are several reasons but specification are the largest bugs creator.
   - In many instances specifications is not written sometimes the applications isn't detailed enough, it's constantly changing, or it's not communicated well to the entire development team. Planning software is absolutely important. If it's not done correctly, bugs will be created.
   - The next largest source of bugs is the design. Bugs occur here for the same reason they occur in the specifications. It's quick, changed, or not well communicated.
   - Coding error can be traced to the software pressure, or just plan dumb mistake.

7) Source of problems in software testing ?
   - Requirement Def – Erroneous, incomplete, inconsistent requirement.
   - Design – Fundamental design flaws in the software.
   - Implementation – Mistakes in chip fabrication, wiring, programming faults, malicious code.
   - Support Systems – Poor programming langauges, faulty compilers and debuggers, misleading development tools.
   - Inadequate testing of software – Incomplete testing, poor verification, mistakes in debugging.

8) What is software testing ?

Software testing is a process, to evaluate the functionality of a software application with an intent to find whether the developed software met the specified requirements or not and to identify the defects to ensure that the product is defect-free in order to produce a quality product.

9) What is Role of testing ?

These are roles, so it is quite possible that one person could fulfill many roles within the testing organization. Some testing roles are given below :

Test lead: A test lead is responsible for the overall testing process for a project. They work with the development team to define the test plan, execute the tests, and track the progress of the testing. They also communicate with stakeholders to ensure that their needs are met.

Test architect: A test architect is responsible for designing the test framework and test environment for a project. They work with the development team to understand the system architecture and design tests that will effectively test the system. They also work with the test lead to ensure that the test plan is feasible and that the tests are executed efficiently.

Test designer: A test designer is responsible for creating test cases for a project. They work with the test architect to understand the test framework and environment, and they work with the test lead to understand the project requirements. They then create test cases that will effectively test the system and meet the project requirements.

Test automation engineer: these engineers work with the test lead and test designers to create automated tests that can be executed quickly and repeatedly. They also work with the development team to integrate the automated tests into the development process.

Test methodologies Test methodologies are the processes and techniques used to test software. There are many different test methodologies, each with its own strengths and weaknesses. The best test methodology for a project will depend on the specific needs of the project.

10)What is Software testing myths ?

Myth 1: Testing is Too Expensive

Reality - There is a saying, pay less for testing during software development or pay more for maintenance or correction later. Early testing saves both time and cost in many aspects, however reducing the cost without testing may result in improper design of a software application rendering the product useless.

Myth 2: Testing is Time-Consuming

Reality - During the SDLC phases, testing is never a time-consuming process. However diagnosing and fixing the errors identified during proper testing is a time-consuming but productive activity.

Myth 3: Perfect Testing is possible.

Reality - Often a client thinks that if the software application is tested then it a perfect product but the fact is something else. There might exist some vulnerabilities that can't be executed during the testing process and it creates problems once the product is launched.

Myth 4: Only completely developed products are tested.

Reality - It is always a consideration that once the software product is fully developed only then it is tested. But the fact is that after each functionality addition or removal software product is tested. If testing is carried out only after full development it might create + chaos and may take a lot of effort to fix it.

Myth 5: Tested software defects less.

Reality - Software can't be 100% defect less even after thousands of testing processes. Often the clients and also developers think that if the software is tested in multiple levels then it has zero bugs but it is only a myth while the fact is a software can't be 100% defectless.

Myth 6: Software testing is a cakewalk.

Reality - Often people outside the testing team think that testing can be done by a person with less experience and less knowledge. The reality is that testing is done by testing experts that are experienced and knowledge full personnel in order to get productive software.

Myth 7: Testing is all about finding bugs.

Reality - It is true that finding bugs is one of the prominent objectives of the testing but it is not the only objective. Besides finding bugs there are several areas that are checked during the testing process. Software functional and non-functional requirements are checked during the testing process.

Myth 8: Test automation should be used wherever possible to reduce time.

Reality – Yes, is true that Test Automation reduces the testing time, but it is not possible to start test automation at any time during software development. Test automaton should be started wher the software has been manually tested and is stable to some extent. Moreover, test automation can never be used if requirements keep changing.

Myth 9: Testers are Responsible for Quality of Product.

Reality-It is a very common misinterpretation that only testers or the testing team should be responsible for product quality. Testers responsibilities include the identification of bugs to the stakeholders and then it is their decision whether they will fix the bug or release the software Releasing the software at the time puts more pressure on the testers, as they will be blamed for any error.

11) What are objective for testing ?

- To find out the defects.
- To prevent defects.
- To evaluate the software performance, usability and reliability.
- Verify and validate the user requirements.
- Ensure quality of products.

12) What is the STLC ?

STLC stands for Software Testing Life Cycle. It is a systematic approach to testing a software application to ensure that it meets the requirements and is free of defects. It is a process that follows a series of steps or phases, and each phase has specific objectives and deliverables.

1. Requirement Analysis: In this phase, the testers work with the stakeholders to understand the requirements of the software application. They identify the features that the software should have, as well as the non-functional requirements, such as performance, security, and usability.

2. Test Planning: In this phase, the testers develop a test plan that outlines the scope of testing, the test cases that will be developed, and the resources that will be needed.

3. Test Case Design: In this phase, the testers design test cases that will be used to verify that the software meets the requirements. The test cases should be comprehensive and should cover all of the possible scenarios that the software may encounter.

4. Test Environment Setup: In this phase, the testers set up the environment that will be used to execute the tests. This includes installing the software, configuring the test tools, and creating data sets.

5. Test Execution: In this phase, the testers execute the test cases and record the results. They also identify any defects that are found during the testing process.

6. Test Closure: In this phase, the testers review the test results and close out the test cases. They also report the results to the stakeholders and make recommendations for further testing.

13) what is V model for software testing ?

The V-model is a type of SDLC model where process executes in a sequential manner in V-shape. It is also known as Verification and Validation model. It is based on the association of a testing phase for each corresponding development stage. Development of each step directly associated with the testing phase. The next phase starts only after completion of the previous phase i.e. for each development activity, there is a testing activity corresponding to it.

The V-Model is a software development life cycle (SDLC) model that provides a systematic and visual representation of the software development process. It is based on the idea of a "V" shape, with the two legs of the "V" representing the progression of the software development process from requirements gathering and analysis to design, implementation, testing, and maintenance.

Phase of Vmodel :

Design phase/ Verification phase.

1) Requirement analysis: This phase contains detailed communication with the customer to understand their requirement and expectation. This stage is known as requirement gathering.
2) System Design: This phase contains the system design and the complete hardware and communication setup for developing product.
3) Architectural Design: System design is broken down future into modules taking up different functionalities. The data transfer and communication between the internal modules and with the outside world(other system) is clearly uderstood.
4) Module Design – In the phase the system broken down into small modules. The detailed design of modules is specified, also known as low-level-design(LLD).

Testing phase/Validation phase.

1) Unit testing: Unit testing are developed during module design phase. These unit test plan are executed to eliminate bugs at code or unit level.

2) Integration testing: After completion of unit testing integration testing is performed. In integration testing the modules are integrated and the system is tested. Integration testing is performed on the Architectur design phase. The test verifies the communication of modules among themselves.
3) System testing: System testing test the complete application with its functionality, inter-dependency and communication. It tests the functional and non-functional requirement of the developed applications
4) Acceptance testing: It is performed in user environment that resembles the production environment. It verifies that the delivered system meets user's requirement and system is ready for use in real world.

14) What makes a good software tester ?

- They are explorers
- They are troubleshooters.
- They are relentless.
- They are creative.
- They are (mellowed) perfectionists.
- They exercise good judgement.
- They are tactful and diplomatic.
- They are persuasive.
- Good communications.
- You can wear's end user shoes.

- Report negative thing in positive way.
- Constant learner

15) What software testing terms: Precision and Accuracy.

Features	Precision	Accuracy
Def	How close repeated measurements of the same quantity are to each other.	How close a measurement is to the true value.
Example	Throwing a dart and it lands in the same spot each time.	Throwing a dart and it lands in the bullseye.
Combinations	Both precision and accuracy, precision without accuracy, accuracy without precision.	Both precision and accuracy, precision without accuracy, accuracy without precision.
Importance	Desirable to have both in measurements, but may be more important to have one than the other depending on the context.	Important in medical settings, less important in quality control settings.
Formula	P:True positive/(True positive + False positive)	A: (True positive + True negative)/Total

16) What software testing terms: Verification and Validation.

Sr.No	Verification	Validation
1	Means: Are we building the product right?	Means: Are we building the right product?
2	It's a static process of verifying documents, design, code and program.	It's a dynamic mechnasim of validating and testing the actual product.
3	It does not involve executing the code.	It always involves executing the code.
4	It is human based checking of documents and files.	It is computer based execution of program.
5	Used methods like inspections, reviews, walkthroughs, and desk-checking.	Used method like black box(functional), white box(structural) , gray box testing.
6	Verification is done by QA.	Validation is done by testing team.
7	It generally comes first-done before validation.	It generally follows after verification.
8	It can catch error that validation cannot catch. It is low level exercise.	It can catch error that verification cannot catch. It is high level exercise.
9	To check whether the software confirms to specifications	To check whether software meet the customer expectations and requirements.

17) What is Quality assurance and Quality control.

Features	Quality assurance	Quality control

Purpose	Prevent defects from occuring.	Find and fix defects that have already occurred.
Focus	Proactive	Reactive
Activites	Planning, testing, documentation.	Inspections, testing, acceptance testing.
Timings	Throughout the development process.	Late in the development process.