(1) Define Term: Testing, Failure, Fault, Error, Test case, Testware, Incident, Test Oracle

→ Testing: Testing is the Process of executing a Program with the intent of finding error

→ Failure: Failure means the inability of a system or Component to Perform a required function according to its Specification

→ Fault: Fault is a Condition that in actual Causes a System to Produce failure

→ Error: It's quality Problem in Software. Generally identify before Software deliver to Client

→ Test Case: A test case is a define format for Software testing required to check if a Particular Software is working or not

→ Testware: The document Created during testing activity are known as testware

→ Incident: When failure Occurs it may or may not visible to user

→ Test Oracle: An Oracle is the means to Judge the Success or failure of a test

(2) What is Bug? Explain life cycle of bug in details

-> Bug is a Condition that in actual Cause a System Produce failure

-> In the testing Phase we analyze the incidents when the failure occurs

-> The whole life cycle of bug can be Classified into two Phases
   (i) bug in Phase
   (ii) bug out Phase

(i) bug in Phase
-> In this Phase where the errors and bug introduced in the Software
-> whenever we Commit a mistake it Creates Errors on a Specific location of the Software when this Error goes Unnoticed it Causes Conditions to fail leading to a bug in the Software
-> if an error has been Produced in the Requirement Specification Phase and not detected in the Same Phase then it results in a bug in the next Phase.
-> In design Phase a bug has come from the Previous Stage but an error can also be Produce in this Stage. if error not detect in this Phase ~~and~~ it Passes on next Stage
-> then it becomes a bug In this way errors and bugs appear and travel throught Various Stage of SDLC

VISION DATE : PAGE NO.

Bug out Phase

-> if failure occur while testing a Software Product we come to the Conclusion that it is affected by bugs

-> However there are situations when bugs are Present even though we don't observe any failure

-> In this Phase when we observe failure the following activity are Performed to get rid of the bugs

* Bug Classification

In this Part we observe the failure and classify the bugs according to its nature

A bug can be Critical or Catastrophic in nature or it may have no adverse effect on the output behaviour of the Software.

* Bug Isolation

We observe the Symtoms and back-trace the design of the Software and reach the module and the Condition inside it which has caused the bug This is known as bug isolation

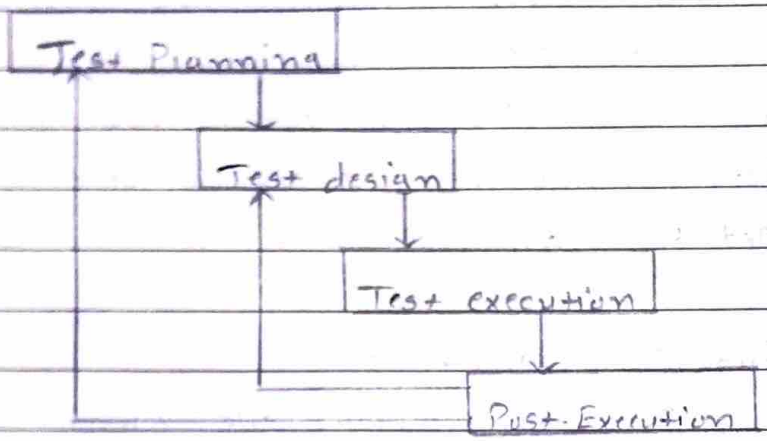
* Bug Resolution

Once we have isolated the Bug we back trace the design to PinPoint the location of the error.

In this way a bug is resolved when we have found the exact location of its occurrence

(a) Explain STLC in detail with diagram

-> The testing Process divided into well-defined sequence of steps is termed as Software testing life cycle



-> Test Planning

- The goal of test Planning is to take into account the important issues of testing Strategy resource, Schedules, responsibilities, Risks and Priorities as a roadmap
- The following are the activity during test Planning
  - * Defining Strategy
  - * Estimate number of testcase
  - * Plan the resource
  - * Identifying areas of Risks
  - * Defining test Completion Criteria

=> Test Design

- One of the major activities in testing is the design of test cases.
- Test Design is an important Phase after test Planning it includes the following Critical activities
  * Determining the test objectives and their Prioritization
  * Preparing list of item to be tested
  * Mapping item to test cases
  * Selection of test case design technique
  * Creating test case & test duty
  * Setting up test environment and Supporting tools
  * Creating test Procedure Specification

=> Test Execution

- In this Phase all test case are executed including Verification and Validation
- Verification test case are started at the end of each Phase of SDLC
- Validation test case are started after Completion of Module

=> Post-Execution

- After Succesful test execution bugs will be Reported to the Concerned developers
- This is Primary goal of all test activity done earlier

(a) Differentiate between Black Box Testing and White box testing

Black Box	White Box
- It is way of Software testing in which the internal structure is hidden and nothing is known about it	- It is a way of testing the Software in which the tester has knowledge about the internal structure
- Implementation of code is not needed	- Code implementation is necessary
-> It is mostly done by Software tester.	- It is mostly done by Software developers
- It can be referred to as outer or external Software testing	- It is the inner or the internal Software testing
- It is a functional test of Software	- It is a structural test of the Software.
- It is the behaviour testing of the Software	- It is logic testing of the Software.
- It is applicable to the higher levels of testing of Software	- It is applicable to the lower levels of Software testing
- It is called Closed testing	- It is also called as Clear box testing
- It is less time Consuming	- It is most time Consuming
- It is not Suitable for Algorithm testing	- It is Suitable for algorithm testing

(5) Defining term : Stub & Driver, Unit Test, Integration Testing

Stub : In most cases dummy modules instead of actual modules are Prepared for these Secondary modules. These dummy modules are called Stubs.

Driver : The module where required inputs for the module under test are Simulated for the Purpose of module or unit testing is known as a driver module

Unit Test : Unit test ensure that the Software meet at least a baseline level of functionality Prior to integration and System testing

Integration Testing : In the modular design of a Software System where the System is Composed of different modules integration is the activity of combining the modules together when all the modules have been Prepared

(6) Write Comparison between top down and buttom up integration testing

⇒ Top down integration testing.

The Strategy in top down integration is to look at the design ~~hire~~ hierarchy from top to buttom

Start with the high level modules and move downward through the design hierarchy

(1) Start with the top or initial module in the Software Substitute the stubs for all the Subordinate modules of top module Test the top module.

(2) After testing top module Stub are Replaced one at a time with the actual modules for integration

(3) Perform testing on this Recent integrated environment

(4) Regression testing may be Conducted to ensure that new errors have not appeared

(5) Repeat Step 2-4 for the whole design hierarchy

-> Bottom-up integration testing

The bottom up strategy begins with the terminal or module at the lowest level in the software structure.

After testing these modules they are integrated and tested moving from bottom to top level

Like top-down strategy this strategy does not required the architectural design of the system to be complete.

Thus, bottom-up integration can be performed at an early stage in the development process

(1) Start with lowest module in the design hierarchy. These are the modules from which no other module is being called

(2) Look for the super oriented module which calls the module selected in step 1.

(3) Test module selected in step 1 with the driver designed in step 2

(4) The next module to be tested is any module whose subordinate module have all been tested

(5) Repeat step 2 to 5 and move up in the design hierarchy

(6) Whenever, the actual module are available. Replace stubs and driver with the actual one and test again

(3). Differentiate Alpha testing and Beta testing

Alpha testing	Beta testing
• Alpha testing involves both the white box and black box testing	- Beta testing commonly use black box testing
• Alpha testing is performed by testers who are usually internal employees of the organization	- Beta testing is performed by clients who are not part of the organization
• Alpha testing performed at the developer's site	- Beta testing performed at the end user of the product
• Reliability and security testings are not checked in alpha testing	- Reliability and security are checked in Beta testing
- Alpha testing requires a testing environment or lab	- Beta testing doesn't require environment or lab
- Alpha testing requires a long execution cycle	- Beta testing requires only a few weeks of execution
- Multiple test cycle are in Alpha	- only one or two cycle in Beta

(8) Explain Master test Plan in detail

-> The Master test Plan Provides the highest level description of Verification and Validation efforts and drives the testing at Various level

-> General Project information is Used to develop the master test Plan

-> The following topic must be addressed before Planning

- Project identification
- Plan Goals
- Summary of Verification and Validation
- Responsibilities Conveyed with the Plan
- Software to be Verified and Validated.
- Identification of Changes to Organization Standards

-> Master Schedule

The master Schedule Summarizes Various V&V tasks and their Relationship to the overall Project

Describe the Project life cycle and Project milestones including Completion dates

Summarizes the Schedule of V&V tasks and how Verification and Validation Results Provide feedback to the development Process to Support overall Project Management function

-> Resource Summary

This activity Summarizes the resources needed to perform V&V tasks including staffing, facilities, tools, finances and special Procedural requirements such as Security access rights and documentation Control In this activity

Use graph and table to Present resource utilization

Summarize the Purpose and cost of hardware and Software tools to be employed

-> Responsibilities

Identify the organization responsible for Performing V&V tasks

There are two level of responsibilities. General responsibilities assigned to different organization and specification responsibilities for the V&V tasks to be performed assigned to individuals

-> Tools, Techniques, and Methodology

Identify the special Software tools, techniques and methodologies to be employed by the V&V team

The Purpose of each should be defined and Plans for the acquisition, training support and qualification of each should be described

VISION DATE : PAGE NO.

(9) Explain CMM Model in details

-> Capability Maturity Model is a framework meant for Software development Processes

-> The structure of CMM Consists of five maturity level These levels Consist of key Process areas which are organized by Common feature

-> The maturity levels indicate the Process Capability Key Practices describe the activities to be done.

-> Common feature address implementations In this way by working on the activities describe in Key Practices Under Some Common feature we achive a goal Under a KPA

-> Each KPA identifies a cluster of related activities that when performed Collectively achieve a set of goals Considered important for Enhancing the Process Capability

-> Maturity Levels

* Initial : At this level the Process is Chaotic or ad-hoc. where there is no Control on development Progress

VISION DATE: PAGE NO.

* Repeatable Earlier Project Successes are used here and the lessons learned from Past Projects are incorporated in the current Project

* Defined The Management Scope is widened to the organization here

* Managed A quantitative Understanding of the Processes is established in order to monitor and Control them

* Optimizing The Process is Continuously Improved through incremental and innovative technological changes or improvements

VISION DATE : PAGE NO.

(10) Explain Software quality matrix in detail

-> Software quality metrics are a subset of Software metrics that focus on the quality aspects of the Product Process and Project

-> Software quality metrics can be grouped in to the following three categories in accordance with the Software lifecycle

1. Product Quality Metrics

-> Mean-time to failure (MTTF) Metric is an estimate of the average or mean time until a Product's first failure occurs

-> Defect density metrics It measure the defect relative to the Software size.

-> Customer Problem metrics This metric measures the Problems which Customers face while using the Product

-> Customer Satisfaction metrics Customer Satisfaction is usually measured through various method of Customer Surveys via the five-Point Scale

2. In-Process Quality Metrics

-> Defect-density during testing Higher defect rates found during testing is an indicator that the Software has experienced higher error injection during its development Process

-> Defect arrival Pattern during testing The Pattern of defect arrivals or the time between Consecutive failures gives more information

3. Matrics for Software Maintenance

-> Fix backlog and backlog management index
Fix backlog metrics is the count of reported problems that remain open at the end of a month or a week

-> Fix response time and fix responsiveness:
The goal is to get high fix responsiveness to meet Customer expectations and have highly Satisfied Customers

-> Fix quality It is the metric to measure the number of fixes that turn out to be defective