# TYPES OF SOFTWARE TESTING

2.1 Software Unit>>>>

1) What is white box testing?

- White box testing is a testing technique, that examines the program structure and derives test data from the program logic / code.
- White box testing is a strategy in which testing is based on : the internals paths, structure and implementation of the software under test.
- It is also known as structural testing, clearbox testing, and glass box testing.
- It requires detailed programming skills.
- White box testing has a risk. You might unconsciously test the software based on how the code is written, which can lead to not finding all the problems. This is because you're too focused on the way the code works.

Test cases are designed for white box testing :-

- Exercise independent paths within a module or unit.
- Exercise logical decisions on both their true and false side.
- Execute loops at their boundaries and within their operational bounds.
- Exercise internal data structure to ensure their validity.

Why white box testing :-

- Find bugs early.
- Finds bugs that are hard to find with black box testing.
- Provides good ideas to black box testing.

Classification of White box testing is as follows :-

1. Static Testing
    - Reviewing
    - Walk through

- Inspection

2. Structural Testing
    - Code Functionality Testing
    - Code Coverage Testing
    - Code Complexity Testing

2) What is Static Testing ?

Static testing is a type of software testing that does not involve the execution of the software or its components. Instead, it focuses on reviewing and analyzing software artifacts, such as source code, requirements, design documents, and test plans, to identify defects, inconsistencies, or areas that need improvement. Static testing methods include techniques like code reviews, inspections, walkthroughs, and static analysis tools. These methods aim to improve software quality by finding issues early in the development process before code is executed or integrated into the system.

3) What is Inspection ?

- Inspection is the most-formal type of review a strategy adopted during the static testing phase.
- It is the most formal review type. It is led by the trained moderators. During inspection the documents are prepared and checked thoroughly by the reviewers before the meeting. It involves peers to examine the product. A separate preparation is carried out during which the product is examined and the defects are found. The defects found are documented in a logging list or issue log. A formal follow-up is carried out by the moderator applying exit criteria.

Characteristics of Inspection :-

1.Trained Moderator: An inspection is typically led by a trained moderator who is not the author of the document being inspected. The moderator's role is to facilitate the inspection process and ensure that it is conducted effectively and impartially

2. Formal Process : Inspections are formal review processes that follow specific procedures, guidelines, and checklists. This formal approach helps ensure consistency and thoroughness in evaluating the document.

3 Entry and Exit Criteria : Entry criteria define the conditions that must be met for a document to enter the inspection process. Exit criteria define the conditions under which the inspection process can be concluded. These criteria help maintain the quality and readiness of the document for inspection.

4. Pre-Meeting Preparation : Before the inspection meeting, participants, including the author and reviewers, should prepare by reviewing the document, noting potential issues, and familiarizing themselves with the inspection guidelines.

5 Inspection Report : Following the inspection, an inspection report is generated. This report includes the findings of the inspection, identified defects, areas for improvement, and recommendations for corrective actions.

6 Follow-Up Process : After the inspection, a formal follow-up process is carried out to ensure that identified defects are addressed in a timely manner. This might involve re-inspecting the document after corrections have been made to verify that the issues have been resolved.

4) What is Structured Walkthroughs ?

- A structured walkthrough, a static testing technique performed in an organized manner between a group of peers to review and discuss the technical aspects of software development process. The main objective in a structured walkthrough is to find defects in order to improve the quality of the product.
- Structured walkthroughs are usually NOT used for technical discussions or to discuss the solutions for the issues found. As explained, the aim is to detect error and not to correct errors. When the walkthrough is finished, the author of the output is responsible for fixing the issues.

Benefits:

- Saves time and money as defects are found and rectified very early in the lifecycle.
- This provides value-added comments from reviewers with different technical backgrounds and experience.

Structured walkthrough participants :-

- Author :- author of the document under review .
- Presenter :-develops the agenda for walkthrough and it presents the output .
- Moderator:- it facilitates the walkthrough session.
- Reviewers:- they evaluate the document under test to determine if it is technically accurate .
- Scribe:-it is the recorder who records the issues identified.

5) What is Technical Review ?

- A technical review is a static white box testing technique which is conducted to spot the defects early in the lifecycle that cannot be detected by black box testing.
- Technical review are documented and uses a defect detection process that has peers and technical specialist as part of the review process.
- The review process doesn't involve management participition.
- It is usually led by trained moderator who is not the author.
- The report is prepared with the list of issues that needs to be addressed.
- Detects defects early saves cost.
- Review are done in order to improve product quality.

Activity of Technical Review :-

- Planning
- Defining the entry and exit criteria for formal review.
- Checking entry criteria.
- Nothing potential defects, questions and comments.
- Review meeting.

- Examing / Evalution / Recording
- Rework
- Follow up
- Checking and exit criteria.

Roles and Responsibility :-

- Manager
- Moderator
- Author
- Reviewer
- Scribe

6) What is Structural Testing ?

- Structural testing is a type of software testing which uses the internal design of the software for testing or in other words the software testing which is performed by the team which knows the development phase of the software, is known as structural testing.
- Structural design is basically related to the internal design and implementation of the software.
- It involves the development team members in the testing team.
- It is basically test different aspects of the software according to its types.

7) Types of Structural Testing ?

Control Flow Testing :-

- Control flow testing ensures that control structures like loops and conditional statements are exercised throughly, including all possible execution paths.
- The SUT's (Software Under Tests) implementation is analysed.
- Paths through the SUT are identified.
- Inputs are choosen to cause the SUT to execute selected paths.
- Expected results for those inputs are determined.
- Actual outputs are compared with the expected outputs.
- It is path testing.

Data Flow Testing :-

- Data flow testing examines the flow of data through the code, ensuring that variables are used and manipulated correctly at various points in the program.
- It is also classifies into 2 types static and dynamic data flow testing.

8) What is Code Functional Testing ?

- The type of testing is the early phase of testing the code. This is done before submitting the code to more extensive phases like coverage testing and complexity testing.
- Code functional testing involves debugging sort of activities must require the code knowledge like loop interations, condition, inputs and corresponding output, etc.

Some methods of code functional testing :-

1. Sometimes being developer we know obvious errors for some inputs.so at first perform test by passing some input and check it's output.we can repeat this to move our confidence to next level.

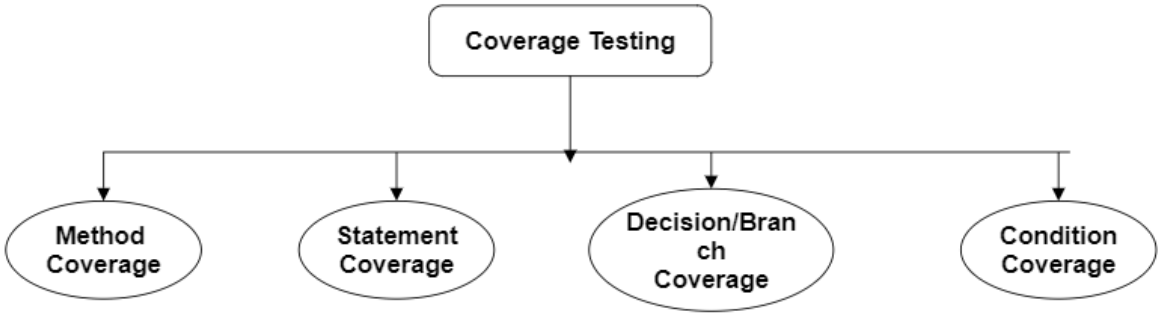
2. Consider module with more complex logic and conditions. Check the loops and iterations by putting print statement at intermediate points one they are executed with correct output then remove the added print statements.

3. We can use different IDE or debugging tools to resolve early defects and these tools provides function which allows us to run module statement by statement ,it also provides facility of adding breakpoints.

9) What is Code Coverage Testing ?

- Coverage is a measure of the completeness of the set of test cases.
- You must attempt to enter and exit every modul, execute every line of code, and follow every logic and decision path through the software.

- Examining the software at this level of details is called code coverage analysis.
- Code coverage analysis is a dynamic white box testing technique because it requires you to have full access to the code to view what parts of the software you pass through when you run your test cases.




Method Coverage :- Method Coverage testing ensures that all methods or subroutines in a software program are executed at least once during testing, validating the invocation and behavior of each individual function.

Statement Coverage :- Testing This technique aims to ensure that every individual statement in the code is executed at least once during testing It helps identify dead code (code that is never executed) and ensures that all statements are syntactically correct.

Branch Coverage :- Testing Also known as decision coverage, this technique focuses on testing all possible branches or decisions within the code. It ensures that each decision point (such as if statements) is tested both when the condition is true and when it's false.

Condition Coverage Testing :- This technique focuses on testing all possible conditions within a decision point (such as individual Boolean conditions). It ensures that each condition is tested both when true and when false.

10) What is Code Complexity Testing ?

- Cyclomatic complexity is a software metric.
- It defines the number of independent paths in a program.
- It is quantitive measure of the number of linearly independent paths in it.

- It is used to describe the complexity of a program.
- It is computed using control flow graph of the program.

Cyclomatic Complexity V(G) :-

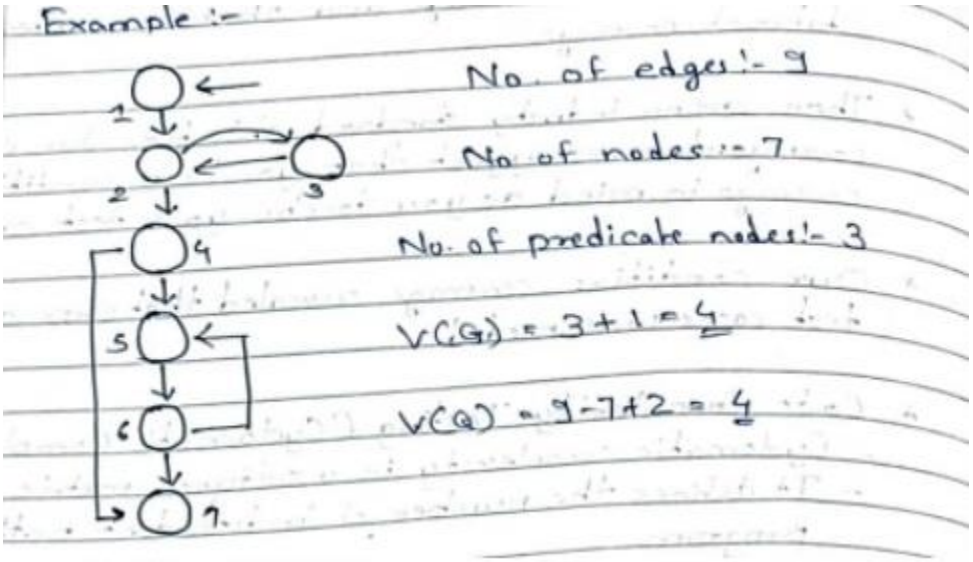
V(G) = E - N + 2

Where E = Number of graph Edges.

N = Number of graph Nodes.

V(G) = P + 1

Where P = No of predicate nodes in the flow graph.

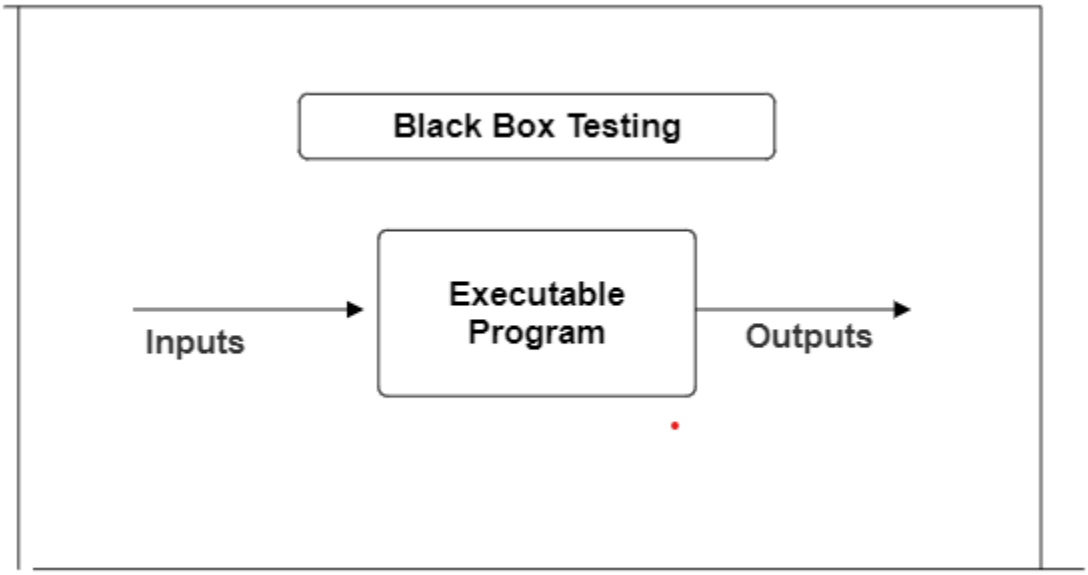



2.2 Software Unit>>>>

11 ) What is Black box testing ?

- Black Box Testing, also known as Behavioural Testing, is a software testing method in which the internal structure/design/implementation of the item being tested is not known to the tester. These tests can be functional or non-functional, though usually functional.

- This method is named so because the software program, in the eyes of the tester, is like a black box; inside which one cannot see.




Advantages:-

- . Independent of the code: Testers do not need to have access to the source code.
- . Focuses on functionality: Tests the software from the user's perspective.
- . Easy to automate: Can be automated to save time and resources.
- . Can be used in conjunction with other testing techniques: Can be used with other testing techniques to ensure thorough testing.

Disadvantages :-

- . May not be as thorough as white box testing
- . May be more time-consuming
- . May not be suitable for all types of software

Examples :-

A tester, without knowledge of the internal structures of a website, tests the web pages by using a browser, providing inputs (clicks, keystrokes) and verifying the outputs against the expected outcome.

12) What is Requirement Based Testing ?

Requirements-based testing is a testing approach in which test cases, conditions and data are derived from requirements. It includes functional tests and also non-functional attributes such as performance, reliability or usability.

Stages in Requirements based Testing :-

- Defining Test Completion Criteria :- Testing is completed only when all the functional and non- functional testing is complete.
- Design Test Cases :- A Test case has five parameters namely the initial state or precondition, data setup, the inputs, expected outcomes and actual outcomes.
- Execute Tests :- Execute the test cases against the ystem under test and document the results. Verify Test Results: Verify if the expected and actual results match each other. Verify Test Coverage: Verify if the tests cover both functional and non-functional requirement. aspects of the
- Track and Manage Defects :- Any defects detected during the testing process goes through the defect life cycle and are tracked to resolution. Defect Statistics are maintained which will give us the overall status of the project.

13) What is Positive and Negative testing ?

Positive Testing :-

- Positive Testing is testing process where the system validated against the valid input data. In this testing tester always check for only valid set of values and check if a application behaves as expected with its expected inputs.
- The main intention of this testing is to check whether software application not showing error when not supposed to and showing error when supposed to. Such testing is to be carried out keeping positive point of view and only execute the positive scenario. Positive Testing always tries to prove that a given product and project always meets the requirements and specifications.
- Under Positive testing is test the normal day to day life scenarios and checks the expected behaviour of application.

Example of Positive testing:

Consider scenario where you want to test an application which contains a simple textbox to enter nge and requirements say that is should take only integers values. So here provide only positive integer values to check whether it is working as expected or not the Positive Testing. Most of the applications developers are implement Positive scenario where testers get loss defects count around positive testing.

Negative Testing :-

- Negative Testing in testing process where the system validated against the invalid input data. A negative test checks if a application behaves as expected with its negative inputs.
- The main intention of this testing is to check whether software application not showing error when supposed to and showing error when not supposed to. Such testing is to be carried out keeping negative point of view and only execute the test cases for only invalid set of input data
- Negative testing is a testing process to identify the inputs where system is not designed or un- handled inputs by providing different invalid.
- The main reason behind Negative testing is to check the stability of the software application against the influences of different variety of incorrect validation data set.

Example of Negative Testing :-

Consider a same above age textbox example which should accpet only integers values. So here provide the characters like “abcd’ in the age textbox & check the behaviour of application, either it should show a validation error message for all invalid inputs (for all other than integer values) or systems should not allow to enter a non integer values.

14) Difference between Positive and negative Testing ?

| Positive Testing | Negative Testing |
|---|---|
| It is performed only for expected conditions. | It is performed for unexpected conditions. |
| It doesn't cover all possible cases. | It covers all possible cases. |
| It doesn't ensure a good quality product. | It ensures a good quality product. |
| It is less important as compared to Negative testing. | It is more important as compared to Positive testing. |
| It can be performed by people having less knowledge. | It can only be performed by professionals. |
| It takes less time. | It takes more time. |
| It is performed on every application. | It is performed where are chances of unexpected conditions. |
| It ensures software is normal. | It ensures 100% defect free software. |

15 )What is Decision Table ?

- Software testing technique used to test system behaviour for different input combinations. This is a systematic appraoch where the different inputs combination and their corresponding system behaviour(output) are captured in a tabular format.
- It is also called as a Cause-Effect table where cause and effects are captured for better test coverage.
- Two dimensional mapping of a condition against actions to be performed

1) Conditions evaluate to Boolean.
2) Action corresponding to expected activity.

- They can be derived from cause effect graph.

1) Map cause as condition.
2) Map effect as action.

16) What is EQA?

- Equivalence Partitioning also called as equivalence class partitioning. It is abbreviated as ECP.
- ECP is technique of software testing in which input data is divided into partitions of valid and invalid values, and it is mandatory that all partitions must exhibit the same behavior.
- The principal of equivalence partitioning is, test cases should designed to cover each partition at least once.
- The equivalence partitioning are derived from requirements and specifications of the software.
- The advantage of this approach is, it helps to reduce the time of testing due to a smaller number of test cases from infinite to finite.
- It is applicable at all levels of the testing process.

Example :-

- The Below example best describes the equivalence Partitioning:
- Assume that the application accepts an integer in the range 100 to 999.
- Valid Equivalence Class partition: 100 to 999 inclusive.
- Non-valid Equivalence Class partitions: less than 100, more than 999, decimal numbers and alphabets/non-numeric characters.

17) What is BVA?

- 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.

- A boundary value for a valid partition is a valid boundary value.
- A boundary value for an invalid partition is an invalid boundary value.

Example :-

**Example:** Consider a system that accepts ages from 18 to 56.

| Boundary Value Analysis(Age accepts 18 to 56) | | |
|---|---|---|
| Invalid (min-1) | Valid (min, min + 1, nominal, max – 1, max) | Invalid (max + 1) |
| 17 | 18, 19, 37, 55, 56 | 57 |

2.3 Software Unit>>>>

18 ) What is Gray Box testing ?

Gray box testing is a software testing technique which is combination of black box testing technique and white box testing technique.

Strategy :-

Grey box testing does not make necessary that the tester must design test cases from source code. To perform this testing test cases can be designed on the base of, knowledge of architectures, algorithm, internal states or other high-level descriptions of the program behavior. It uses all the straightforward techniques of black box testing for function testing. The test case generation is based on

requirements and preset all the conditions before testing the program by assertion method.

Why to use Gray box testing ?

- It provides combined benefits of both black box testing and white box testing.
- It combines input of developers as well as testers and improves overall product quality.
- It reduces the overhead of long process of testing functional and non-functional types.
- It gives enough free times for a developers to fix defects.
- Testing is done from the user-point-of-view rather than a designer-point-of-view.

Gray Box Testing Techniques :-

Matrix Testing :- In matrix testing technique, business and technical risks which are defined by the developers in software programs are examined. Developers define all the variables that exist in the program. Each of the variables has an inherent technical and business risk and can be used with varied frequencies during its life cycle.

Regression Testing :- Regression testing is testing the software after every change in the software to make sure that the changes or the new functionalities are not affecting the existing functioning of the system. Regression testing is also carried out to ensure that fixing any defect has not affected other functionality of the software.

Orthogonal Array Testing :- It provides maximum code coverage with minimum test cases.

Pattern Testing :- This testing is performed on the historical data of the previous system defects. Unlike black box testing, gray box testing digs within the code and determine why the failure happened.

Advantages:

. Real-world Insight: Blends real scenarios with some code knowledge.

. Better Coverage: Improved testing of critical code areas.

. Efficient Bugs: Early detection of issues and vulnerabilities.

. Optimized Efforts: Testing efforts align with goals and expectations.

. Enhanced Communication: Clearer bug reporting for quicker fixes.

Disadvantages:

. Limited Coverage: May miss some corner cases.

. Dependency Risk: Effectiveness hinges on tester's code understanding.

. and Skill: Requires technical expertise and balancing priorities.

. Bias Potential: Testers might make assumptions based on code knowledge.

. Security Concerns: Access to code could pose security and confidentiality risks.

2.4 Software Unit>>>>

19) Difference between Black box | Gray Box | White Box Testing ?

| Black Box Testing | Gray Box Testing | White Box Testing |
|---|---|---|
| | | |
| The testing has low granularity. | This testing has a medium level of granularity. | This testing has high-Level granularity. |
| It is done by end-users and also done by testers, developers. | It is done by end-users(called user acceptance testing ), also done by testers and developers. | It is generally done by testers and developers. |
| Here internal are not required to be known. | Here, internal relevant to the testing is known. | Here, the internal code of the application and database is known. |
| Less exhaustive than the other two. | It is kind of in-between. | More exhaustive among all three. |
| Also called functional testing, closed box testing and data-driven testing. | Also called translucent testing. | Also called glass box testing, code-based testing, clear-box testing and structural testing. |
| It is not considered for algorithm testing. | It is not considered for algorithm testing. | It is well suitable and recommended for algorithm testing. |
| It is very difficult to discover hidden errors of software because internal working is unknown in this testing. | Difficult to discover the hidden errors. Might be found in user level testing. | It is simple to discover hidden errors because all internal working is known in this testing. |
| The testing space of tables for inputs (input to be used to create test cases) is huge and largest among all testing spaces | The testing space for inputs is small as compared to black box and white box testing. | The testing space for inputs is less as compared to black box testing. |
| Time consumption in black box testing depends upon the availabiltiy of functional specifications. | Test cases designing can be done in a short time period. | Testing takes a long time to design test cases due to lengthy code. |
| It is suited for functional or business testing. | It is suited for functional or business domain testing deeply. | It is used for all. |