What is structured based Technique in software Testing ?

Structure-based techniques serve two purposes: **test** coverage measurement and **structural test** case design. They are often used first to assess the amount of **testing** performed by **tests** derived from specification-**based techniques**, i.e. to assess coverage.

In structure based testing the knowledge of the code and internal architecture of the system is required to carry out the testing.

This technique is a good way of generating accessory test cases which varies from other subsisting tests cases. They can help safeguard and verify more scope of testing techniques. And therefore secure the application functionality. Hence, in the sense that test cases that achieve 100% coverage of testing in any measure will be exercising all the parts.

Structure Based Testing Techniques

- Statement testing – Statement testing is a white box testing approach in which test scripts are designed to execute code statements.
- Decision testing/branch testing – Decision testing or branch testing is a white box testing approach.
- Condition testing – Testing the condition outcomes (TRUE or FALSE). So, getting 100% condition coverage required exercising each condition for both TRUE and FALSE results using test scripts.
- Multiple condition testing – Testing the different combinations of condition outcomes. Hence for 100% coverage we will have 2^n test scripts. This is very exhaustive and very difficult to achieve 100% coverage.
- Path testing – Testing the independent paths in the system(paths are executable statements from entry to exit points).

Purpose of Structure Based Testing

Structure-based techniques server two purposes: Test coverage measurement and structural test case design.

- They are often used first to assess the amount of testing. Thus performed by tests derived from specification-based techniques.
- They are then used to design additional tests with the aim of increasing the test coverage.
- What Is Structure Based Technique
- Structure-based test design techniques are a good way of generating additional test cases that are different from existing tests.
- They can help ensure more breath of testing.

Steps in Structure Based Testing

- Verifying security loops in the code
- Verifying the broken paths in the code
- The verification of the specified flow structure
- Verifying the desired output
- Verification of the condition loop to check the functionality
- Verifying each line and section

Advantages of Structure Based Testing

- Provides a more thorough testing of the software.
- Helps finding out defects at an early stage.
- Helps in eliminating dead code.
-

Disadvantages of Structure Based Testing

- Requires knowledge of the code.
- Requires training in the tool used for testing
- It is expensive.

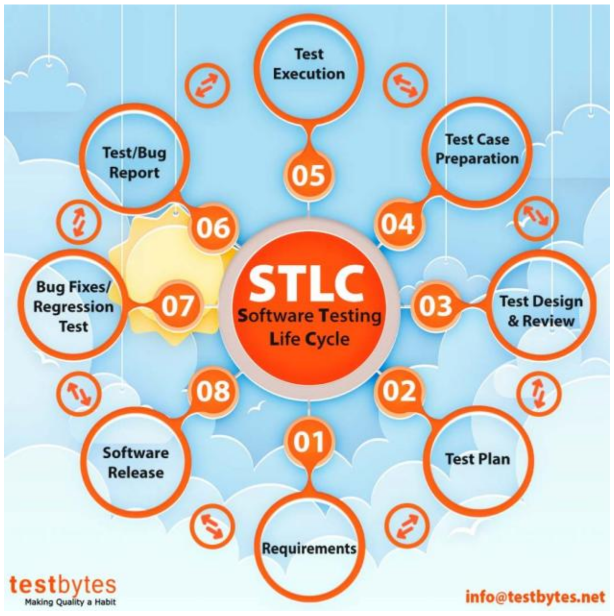
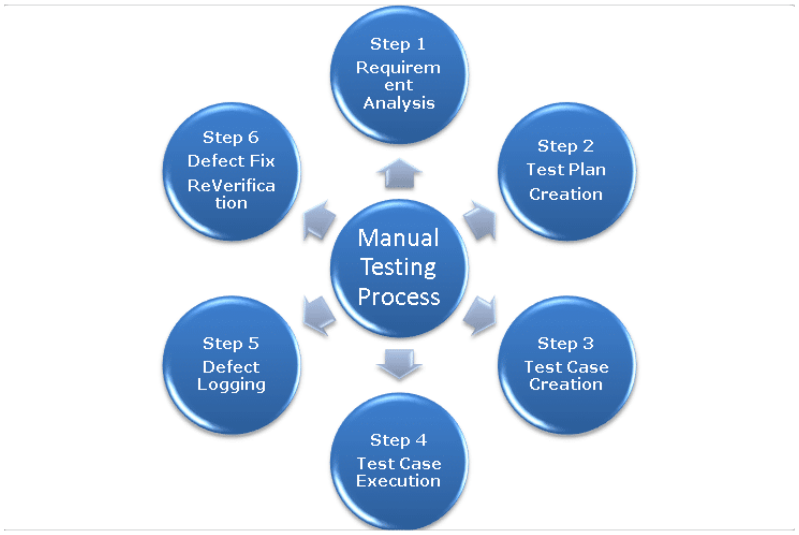
- This white box testing technique involves 100% condition coverage of the code. Here, the each condition of the code is executed at least once.
- **Decision/Condition Coverage**
- This is a hybrid or mixed model that involves entire decision/condition coverage of the code. Here, the each condition/decision in the code is executed at least once.
- **Techniques in Structure Based Testing**
- Statement Coverage
- It aims to test all the statements present in the program. Adequacy Criterion should be equal to 1 to ensure 100% coverage. It is a good measure of testing each part in terms of statements but it is not a good technique for testing the control flow.
- **Branch Coverage**
- It aims to test all the branches or edges at least once in the test suite or to test each branch from a decision point at least once. It provides solution for the problem faced in Statement coverage.
- Branch Testing provides a better coverage than Statement testing but it too has its shortcomings. It does not provide a good coverage from different conditions that lead from node 1 to node 2 as it covers that branch only once.
- **Condition Coverage**
- It aims to test individual conditions with possible different combination of Boolean input for the expression.
- It is modification of Decision coverage but it provides better coverage and the problem discussed under Branch coverage can be resolved here.

- **Path Coverage**
- **cyclomatic complexity** helps aiming to test all linearly independent paths in a program at least once. These were some of the test coverage under this Testing.

- What is dynamic analysis in software testing ?
- **Dynamic analysis** is the **testing** and evaluation of a program by executing data in real-time. The objective is to find errors in a program while it is running, rather than by repeatedly examining the code offline. ... A daily build and smoke **test** (also known as smoke **testing**) is one type of **dynamic analysis**.

- Dynamic analysis is the testing and evaluation of a program by executing data in real-time. The objective is to find errors in a program while it is running, rather than by repeatedly examining the code offline.
- By debugging a program in all the scenarios for which it is designed, dynamic analysis eliminates the need to artificially create situations likely to produce errors. Other advantages include reducing the cost of testing and maintenance, identifying and eliminating unnecessary program components, and ensuring that the program being tested is compatible with other programs.
- A daily build and smoke test (also known as smoke testing) is one type of dynamic analysis.

- What is testing process in software testing ?
- **Testing** is the **process** of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. In simple words, **testing** is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements.

- 
STLC
Software Testing Life Cycle
01 Requirements
02 Test Plan
03 Test Design & Review
04 Test Case Preparation
05 Test Execution
06 Test/Bug Report
07 Bug Fixes/ Regression Test
08 Software Release
testbytes
Making Quality a Habit
info@testbytes.net
- 
Manual Testing Process
Step 1 Requirement Analysis
Step 2 Test Plan Creation
Step 3 Test Case Creation
Step 4 Test Case Execution
Step 5 Defect Logging
Step 6 Defect Fix ReVerification