1. **STATIC TESTING:**

Static Testing is defined as a software testing technique by which we can check the defects in software without executing it. Its counterpart is Dynamic Testing which checks an application when the code is run.

Static testing is done to avoid errors at an early stage of development as it is easier to find sources of failures then failures themselves. Static testing helps to find errors that may not be found by Dynamic Testing.

The two main types of static testing techniques are

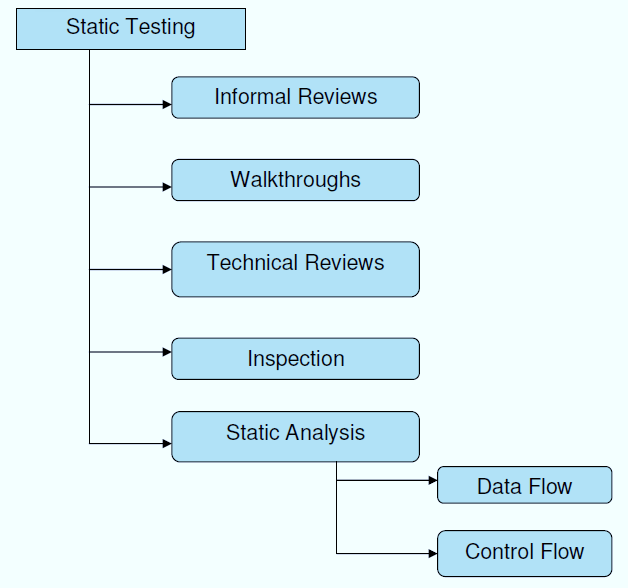
* **Manual examinations**: Manual examinations include analysis of code done manually, also known as **REVIEWS.**
* **Automated analysis using tools:** Automated analysis are basically static analysis which is done using tools.

What is Testing Review?

A review in a Static Testing is a process or meeting conducted to find the potential defects in the design of any program. Another significance of review is that all the team members get to know about the progress of the project and sometimes the diversity of thoughts may result in excellent suggestions. Documents are directly examined by people and discrepancies are sorted out.

Reviews can further be classified into four parts:

* Informal reviews
* Walkthroughs
* Technical review
* Inspections



Types of defects which can be easier to find during static testing are:

* Deviations from standards
* Non-maintainable code
* Design defects
* Missing requirements
* Inconsistent interface specifications

Why Static Testing?

Static testing is performed due to the following reasons

* Early defect detection and correction
* Reduced development timescales
* Reduced testing cost and time
* For improvement of development productivity
* To get fewer defect at a later stage of testing

What is Tested in Static Testing?

In Static Testing, following things are tested

* Unit Test Cases
* Business Requirements Document (BRD)
* Use Cases
* System/Functional Requirements
* Prototype
* Prototype Specification Document
* DB Fields Dictionary Spreadsheet
* Test Data
* Traceability Matrix Document
* User Manual/Training Guides/Documentation
* Test Plan Strategy Document/Test Cases
* Automation/Performance Test Scripts

1. **DYNAMIC TESTING:**

Dynamic Testing is defined as a software testing type, which checks the dynamic behaviour of the code is analysed.

We all know that Testing is verification and validation, and it takes 2 Vs to make testing complete. Out of the 2 Vs, Verification is called a Static testing and the other "V", Validation is known as Dynamic testing.

Let's understand How to do Dynamic Testing with an example:

Suppose we are testing a Login Page where we have two fields say "Username" and "Password" and the Username is restricted to Alphanumeric.

When the user enters Username as "Abcd99", the system accepts the same. Whereas when the user enters as Abcd99@123 then the application throws an error message. This result shows that the code is acting dynamically based on the user input.

Dynamic testing is when you are working with the actual system by providing an input and comparing the actual behaviour of the application to the expected behaviour. In other words, working with the system with the intent of finding errors.

So, based on the above statements we can say or conclude that dynamic testing is a process of validating software applications as an end user under different environments to build the right software.

What does dynamic testing do?

The main aim of the Dynamic tests is to ensure that software works properly during and after the installation of the software ensuring a stable application without any major flaws( this statement is made because no software is error free, testing only can show presence of defects and not absence)

The main purpose of the dynamic test is to ensure consistency to the software; lets discuss this with an example.

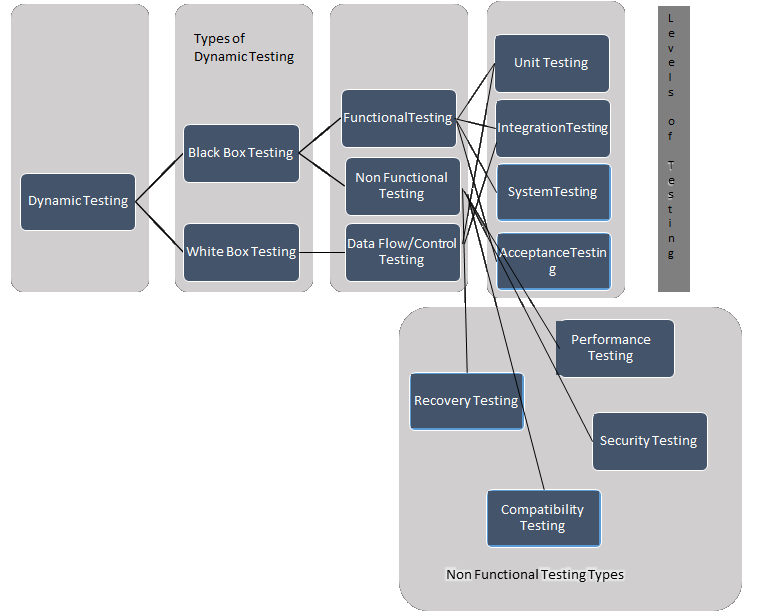
In a Banking Application, we find different screens like My Accounts Section, Funds Transfer, Bill Pay, etc. All these screens contain amount field which accepts some characters.

Let's say My Accounts field displays amount as 25,000 and Funds Transfer as $25,000 and Bill pay screen as $25000 though the amount is the same, the way amount is displayed is not the same hence making the software no consistent.

Consistency is not only limited to the functionality it also refers to different standards like performance, usability, compactivity etc, hence it becomes very important to perform Dynamic Testing.

Types of Dynamic Testing

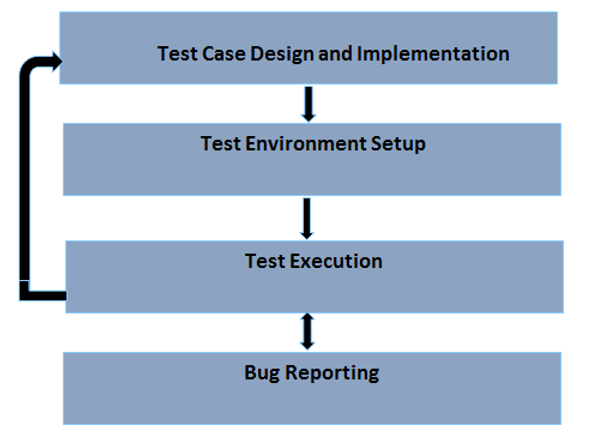
* 1. White Box Testing
  2. Black Box Testing



Dynamic Testing Techniques

We all know that[STLC](https://www.guru99.com/software-testing-life-cycle.html)is a process which consists of different tasks like Requirements Analysis, Test Planning, Test Case Design, Environment setup, Test Execution, and Test Closure.

In STLC we can say that Dynamic Testing Process starts from Test Case Design, let's discuss each activity in detail.



Before getting into the process lets discuss the strategy that needs to be followed for Dynamic Testing.

Test Strategy should mainly focus on the resources available and the timeframe. Based on these factors, the objective of the testing, the scope of testing, phases or cycles of testing, type of environment, assumptions or challenges that might be faced, risks, etc. must be documented.

Once the strategy is defined and is accepted by the management then the actual process test case design starts

What is Test design and Implementation

In this phase we identify the,

* Features to be tested
* Derive the Test Conditions
* Derive the coverage Items
* Derive the Test Cases

Test Environment Setup

We must ensure that Testing Environment should always be similar to the Production environment, in this phase we have to install the build and manage the test machines.

Test Execution

During this phase, test cases are executed.

Bug report captured

Based on the Execution if the Expected and Actual Results are not same then the Test case has to be marked as Fail and a Bug should be logged.

Advantages of Dynamic Testing

Dynamic Testing can reveal the uncovered defects that are too difficult or complicated and which cannot be covered through static Analysis

In Dynamic Testing, we execute the software, end to end, ensuring error free software which in turn increases the quality of a product and project.

Dynamic Testing becomes an essential Tool for detecting any security Threats

Disadvantages of Dynamic Testing

Dynamic Testing is Time Consuming because it executes the application/software or code which requires huge amount of Resources

Dynamic Testing increases the cost of project/product because it does not start early in the software lifecycle and hence any issues fixed in later stages can result in an increase of cost.

**3.WHITE BOX TESTING:**

White Box Testing is defined as the testing of a software solution's internal structure, design, and coding. In this type of testing, the code is visible to the tester. It focuses primarily on verifying the flow of inputs and outputs through the application, improving design and usability, strengthening security. White box testing is also known as Clear Box testing, Open Box testing, Structural testing, Transparent Box testing, Code-Based testing, and Glass Box testing. It is usually performed by developers.

It is one of two parts of the **"Box Testing" approach** to software testing. Its counterpart, **Blackbox testing**, involves testing from an external or end-user type perspective. On the other hand, Whitebox testing is based on the inner workings of an application and revolves around internal testing.

The term "Whitebox" was used because of the see-through box concept. The clear box or Whitebox name symbolizes the ability to see through the software's outer shell (or "box") into its inner workings. Likewise, the "black box" in "[Black Box Testing](https://www.guru99.com/black-box-testing.html)" symbolizes not being able to see the inner workings of the software so that only the end-user experience can be tested.

What do you verify in White Box Testing?

White box testing involves the testing of the software code for the following:

* Internal security holes
* Broken or poorly structured paths in the coding processes
* The flow of specific inputs through the code
* Expected output
* The functionality of conditional loops
* Testing of each statement, object, and function on an individual basis

The testing can be done at system, integration and unit levels of software development. One of the basic goals of Whitebox testing is to verify a working flow for an application. It involves testing a series of predefined inputs against expected or desired outputs so that when a specific input does not result in the expected output, you have encountered a bug.

How do you perform White Box Testing?

To give you a simplified explanation of white box testing, we have divided it into **two basic steps**. This is what testers do when test an application using the white box testing technique:

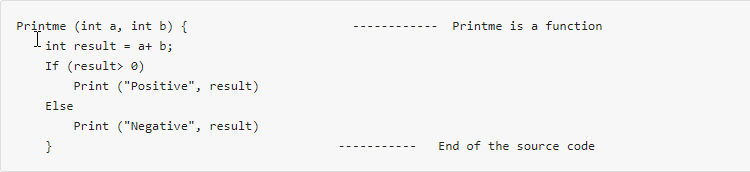
**STEP 1) UNDERSTAND THE SOURCE CODE**

The first thing a tester will often do is learn and understand the source code of the application. Since white box testing involves the testing of the inner workings of an application, the tester must be very knowledgeable in the programming languages used in the applications they are testing. Also, the testing person must be highly aware of secure coding practices. Security is often one of the primary objectives of testing software. The tester should be able to find security issues and prevent attacks from hackers and naive users who might inject malicious code into the application either knowingly or unknowingly.

**Step 2) CREATE TEST CASES AND EXECUTE**

The second basic step to white box testing involves testing the application's source code for proper flow and structure. One way is by writing more code to test the application's source code. The tester will develop little tests for each process or series of processes in the application. This method requires that the tester must have intimate knowledge of the code and is often done by the developer. Other methods include [Manual Testing](https://www.guru99.com/manual-testing.html), trial, and error testing and the use of testing tools as we will explain further on in this article.

WhiteBox Testing Example



The goal of WhiteBox testing is to verify all the decision branches, loops, statements in the code.

To exercise the statements in the above code, WhiteBox test cases would be

A = 1, B = 1

A = -1, B = -3

White Box Testing Techniques

A major White box testing technique is Code Coverage analysis. Code Coverage analysis eliminates gaps in a[Test Case](https://www.guru99.com/test-case.html)suite. It identifies areas of a program that are not exercised by a set of test cases. Once gaps are identified, you create test cases to verify untested parts of the code, thereby increasing the quality of the software product

There are automated tools available to perform Code coverage analysis. Below are a few coverage analysis techniques

**Statement Coverage**: - This technique requires every possible statement in the code to be tested at least once during the testing process of software engineering.

**Branch Coverage -**This technique checks every possible path (if-else and other conditional loops) of a software application.

Apart from above, there are numerous coverage types such as Condition Coverage, Multiple Condition Coverage, Path Coverage, Function Coverage etc. Each technique has its own merits and attempts to test (cover) all parts of software code. Using Statement and Branch coverage you generally attain 80-90% code coverage which is sufficient.

Types of White Box Testing

*White box testing*encompasses several testing types used to evaluate the usability of an application, block of code or specific software package. There are listed below --

**Unit Testing:**It is often the first type of testing done on an application.[Unit Testing](https://www.guru99.com/unit-testing-guide.html) is performed on each unit or block of code as it is developed. Unit Testing is essentially done by the programmer. As a software developer, you develop a few lines of code, a single function or an object and test it to make sure it works before continuing Unit Testing helps identify most bugs, early in the software development lifecycle. Bugs identified in this stage are cheaper and easy to fix.

**Testing for Memory Leaks**: Memory leaks are leading causes of slower running applications. A QA specialist who is experienced at detecting memory leaks is essential in cases where you have a slow running software application.

Apart from above, a few testing types are part of both black box and white box testing. They are listed as below

**White Box**[**Penetration Testing**](https://www.guru99.com/learn-penetration-testing.html)**:** In this testing, the tester/developer has full information of the application's source code, detailed network information, IP addresses involved and all server information the application runs on.  The aim is to attack the code from several angles to expose security threats

**White Box Mutation Testing**: Mutation testing is often used to discover the best coding techniques to use for expanding a software solution.

Advantages of White Box Testing

* Code optimization by finding hidden errors.
* White box tests cases can be easily automated.
* Testing is more thorough as all code paths are usually covered.
* Testing can start early in SDLC even if GUI is not available.

Disadvantages of Whitebox Testing

* White box testing can be quite complex and expensive.
* Developers who usually execute white box test cases detest it. The white box testing by developers is not detailed can lead to production errors.
* White box testing requires professional resources, with a detailed understanding of programming and implementation.
* White-box testing is time-consuming, bigger programming applications take the time to test fully.

Ending Notes:

White box testing can be quite complex. The complexity involved has a lot to do with the application being tested. A small application that performs a single simple operation could be white box tested in few minutes, while larger programming applications take days, weeks and even longer to fully test.

White box testing should be done on a software application as it is being developed after it is written and again after each modification

**4.BLACK BOX TESTING:**

Black box testing is defined as a testing technique in which functionality of the Application Under Test (AUT) is tested without looking at the internal code structure, implementation details and knowledge of internal paths of the software. This type of testing is based entirely on software requirements and specifications.

In BlackBox Testing we just focus on inputs and output of the software system without bothering about internal knowledge of the software program.

The above Black-Box can be any software system you want to test. For Example, an operating system like Windows, a website like Google, a database like Oracle or even your own custom application. Under Black Box Testing, you can test these applications by just focusing on the inputs and outputs without knowing their internal code implementation.

How to do BlackBox Testing

Here are the generic steps followed to carry out any type of Black Box Testing.

* Initially, the requirements and specifications of the system are examined.
* Tester chooses valid inputs (positive test scenario) to check whether SUT processes them correctly. Also, some invalid inputs (negative test scenario) are chosen to verify that the SUT can detect them.
* Tester determines expected outputs for all those inputs.
* Software tester constructs test cases with the selected inputs.
* The test cases are executed.
* Software tester compares the actual outputs with the expected outputs.
* Defects if any are fixed and re-tested.

Types of Black Box Testing:

There are many types of Black Box Testing but the following are the prominent ones -

**Functional testing** - This black box testing type is related to the functional requirements of a system; it is done by software testers.

**Non-functional testing**- This type of black box testing is not related to testing of specific functionality, but non-functional requirements such as performance, scalability, usability.

**Regression testing**- [Regression Testing](https://www.guru99.com/regression-testing.html) is done after code fixes, upgrades or any other system maintenance to check the new code has not affected the existing code.

Black Box Testing Techniques:

Following are the prominent[Test Strategy](https://www.guru99.com/how-to-create-test-strategy-document.html)amongst the many used in Black box Testing

**Equivalence Class Testing:** It is used to minimize the number of possible test cases to an optimum level while maintains reasonable test coverage.

**Boundary Value Testing:** Boundary value testing is focused on the values at boundaries. This technique determines whether a certain range of values are acceptable by the system or not. It is very useful in reducing the number of test cases. It is most suitable for the systems where an input is within certain ranges.

**Decision Table Testing**: A decision table puts causes and their effects in a matrix. There is a unique combination in each column.

## 5. DIFFERENCE BETWEEN STATIC AND DYNAMIC TESTING:

## 

## 6. DIFFERENCE BETWEEN BLACK BOX AND WHITE BOX TESTING:

