CUI Abbottabad

Department of Computer Science

SOFTWARE TESTING

Lecture 1 & 2

Introduction to Software Testing:
Basic Concept and Preliminaries

CONTENT

- Failure, Error, Fault and Defect
- Role of Testing
- Activities for software quality assessment
- The objectives of Testing
- What is a Test Case?
- Sources of information for test case selection
- Testing activities
- Test levels
- Verification vs. Validation (V Model)
- White-box and Black-box Testing

FAILURE, ERROR, FAULT AND DEFECT

- **Failure**

  A *failure* is said to occur whenever the external behavior of a system does not conform to that prescribed in the system specification.

- **Error**

  An *error* is a state of the system.

  An *error* state could lead to a *failure* in the absence of any corrective action by the system.

  Software errors are sections of the code that are partially or totally incorrect as a result of a grammatical, logical or other mistake made by a system analyst, a programmer, or another member of the software development team.

FAILURE, ERROR, FAULT AND DEFECT

- **Fault**

  A *fault* is the adjudged cause of an *error.*

  A fault, also called bug or defect, is a design or coding mistake that may cause abnormal component behavior.

- **Defect**

  It is synonymous of *fault.*

  It is a.k.a. *bug.*

The process of failure manifestation can therefore be briefly represented as a behavior chain as follows:

fault→ error→ failure

ROLE OF TESTING

- **Testing** plays an important role in achieving and assessing the quality of a software product .
- We can improve the quality of the products as we repeat a test - find defects – fix cycle during development.
- The goal of testing is to maximize the number of discovered faults, which then allows developers to correct them and increase the reliability of the system.

ACTIVITIES FOR SOFTWARE QUALITY ASSESSMENT

Friedman and voas says,

"Software testing is a verification process for software quality assessment and improvement."

- Software quality assessment is divided into two categories,
  - Static Analysis
  - Dynamic Analysis

ACTIVITIES FOR SOFTWARE QUALITY ASSESSMENT

STATIC ANALYSIS

- It is based on the examination of a number of documents,
  - requirements documents,
  - software models,
  - design documents
  - source code.
- It also includes code review, inspection, walk-through, algorithm analysis, and proof of correctness.
- It does not involve actual execution of the code under development.
- Instead, it examines code and reasons over all possible behaviours that might arise during run time.

ACTIVITIES FOR SOFTWARE QUALITY ASSESSMENT

DYNAMIC ANALYSIS

- It involves actual program execution with input values in order to expose possible program failures.
- During execution, it observes the behavioural and performance properties of the program.
- For practical considerations, a finite subset of the input set can be selected.
- Therefore, in testing, we observe program behaviours and reach a conclusion about the quality of the system.
- Careful selection of a finite test set is crucial to reaching a reliable conclusion.

ACTIVITIES FOR SOFTWARE QUALITY ASSESSMENT

- Both analysis techniques are complementary in nature, and for better effectiveness, both must be performed repeatedly and alternated.

  Practitioners and researchers need to remove the boundaries between static and dynamic analysis and create a hybrid analysis that combines the strengths of both approaches.

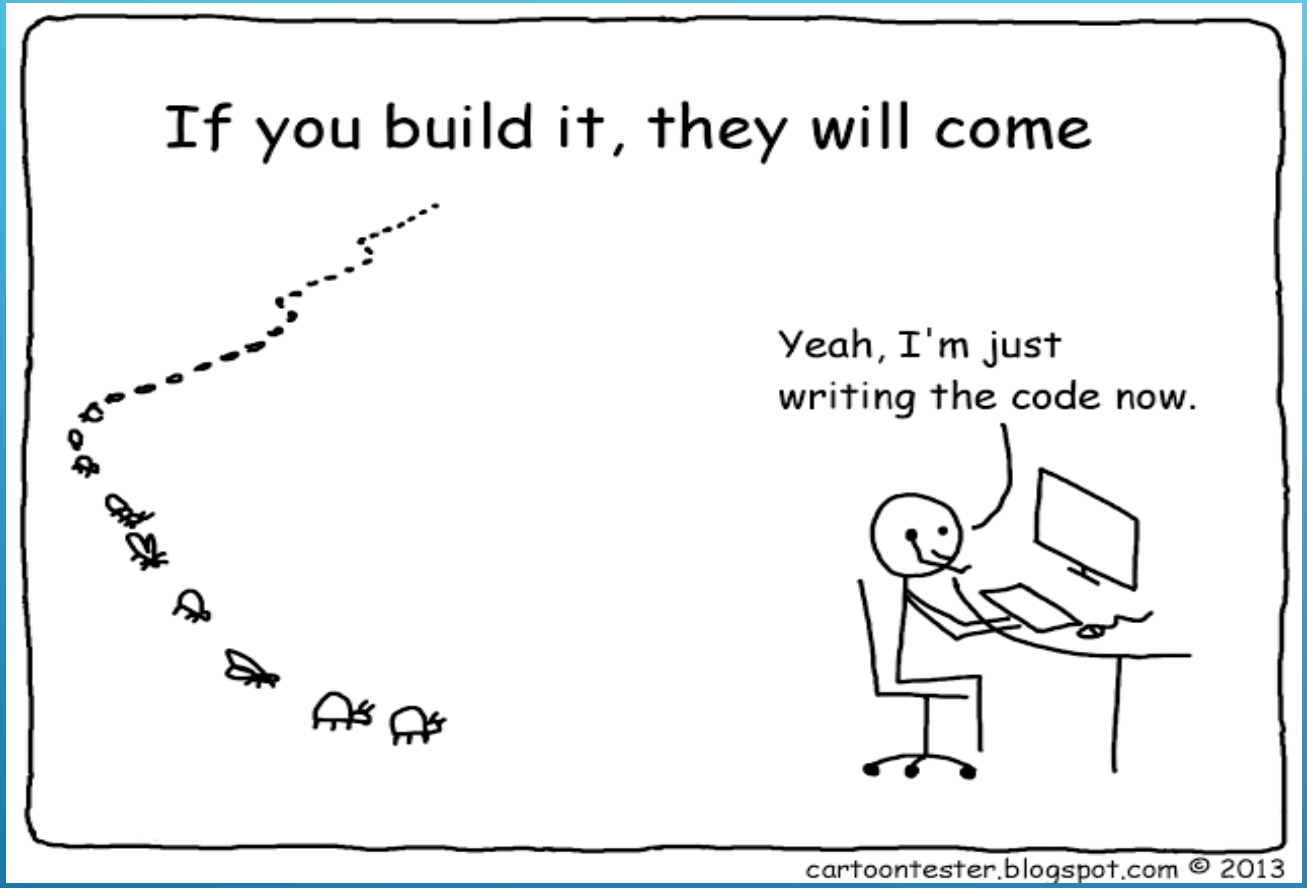
THE OBJECTIVES OF TESTING

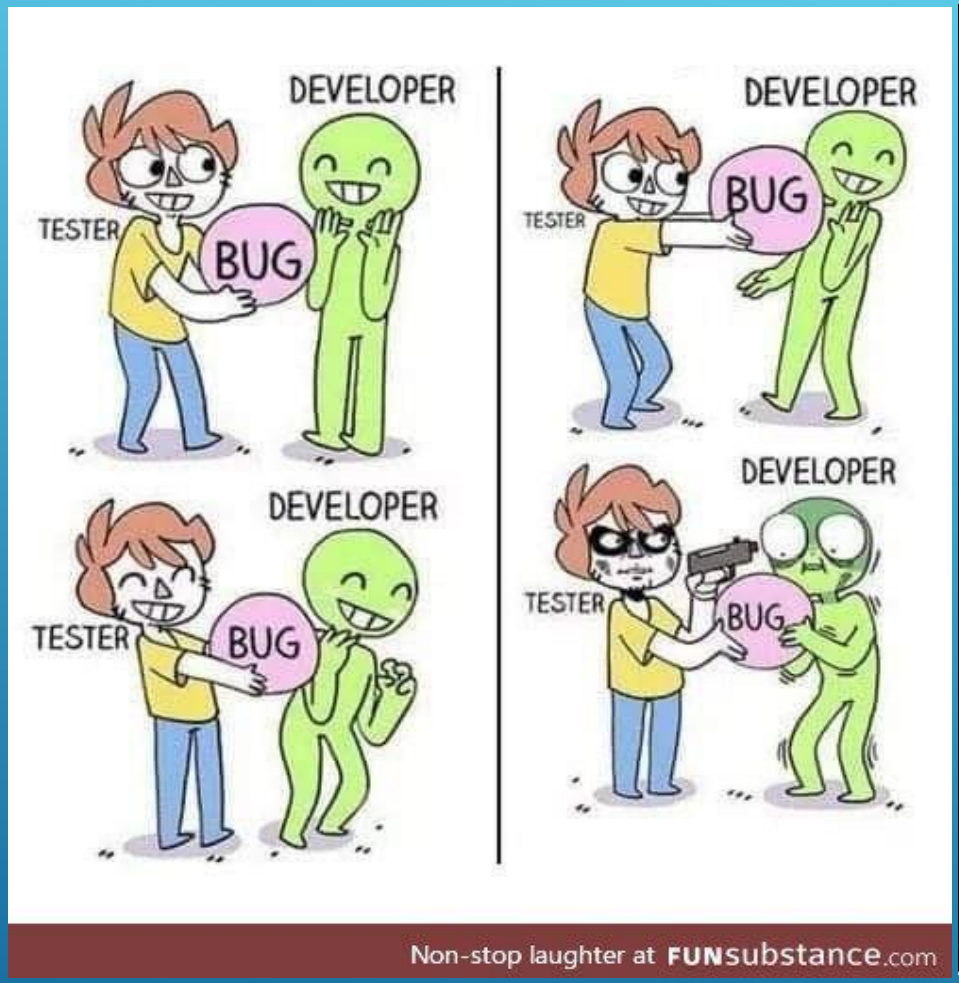
- Testing is define as the systematic attempt to find faults in a planned way in the implemented software.
- The stakeholders in a test process are the programmers, the test engineers, the project managers, and the customers.
- A stakeholder is a person or an organization who influences a system's behaviors or who is impacted by that system.
- Different stakeholders view a test process from different perspectives, as follows:

THE OBJECTIVES OF TESTING

- **It does work:**
  - Objective is to test that unit of code or system works.
- **It does not work:**
  - Once working, next objective is to find faults in unit or system to make system or unit fail.
- **Reduce the risk of failures**
  - Objective → To bring down the risk of failing to an acceptable level by iterative testing and removing faults.
- **Reduce the cost of testing**
  - Number of test cases are directly proportional to cost.
  - Objective: To produce low-risk software with fewer number of test cases.

SOME REAL POINTS THOUGH!!!!!




DEVELOPER
TESTER
BUG
DEVELOPER
TESTER
BUG
DEVELOPER
TESTER
BUG
DEVELOPER
TESTER
BUG
Non-stop laughter at FUNsubstance.com


It doesn't work..... why?
VIA 9GAG.COM
It works..... why?

WRITING FORMAL TEST CASES AND TEST PLANS

- A test case is an explicit set of instructions designed to detect a particular class of defects in a software system.

- Note: the test cases are DESIGNED to detect errors!!!!
  - There may be a number of individual tests in a test case, such as invalid inputs, invalid formats, missing attributes / fields, and more. A single test case may test for a number of these.

WHAT IS A TEST CASE?

- Test Case is a simple pair of *<input, expected outcome>.*
- A test case may consist of a sequence of < input, expected outcome > pairs.
- The test cases will have a generic format as below:
  - Test Case ID
  - Test Case Description
  - Test Prerequisite
  - Test Inputs
  - Test Steps
  - Expected Results
  - Actual Results
  - Verdict

EXAMPLE

TB_1: < 0, 0 >,
TB_2: < 25, 5 >,
TB_3: < 40, 6.3245553 >,
TB_4: < 100.5, 10.024968 >.

Figure 1.3 Examples of basic test cases.

TS_1: < check balance, $500.00 >, < withdraw, "amount?" >,
< $200.00, "$200.00" >, < check balance, $300.00 > .

Figure 1.4 Example of a test case with a sequence of < input, expected outcome >.

A	B	C	D	E	F	G	H	I	J
Test ID	**Date**	**Module**	**Test Cases**	**Test Data**	**Scenarios**	**Expected Result**	**Actual Result**	**Pass / Fail**	**Comments**
1	dd-mm-yy	Login page	Enter the valid email and password	test@test.com /Password	Use the url www.testingfreak.com and go to the login page.	Should allow to login	Not allowing to login	Fail	Invalid Password
2	dd-mm-yy	Login page	Try another valid Eamil and password	test@gmail.com /Password	Use the url www.testingfreak.com and go to the login page.	Should allow to login	Login Successful	Pass	NA

ANOTHER EXAMPLE…

Step	Step Description	Pre-Conditions	Steps	Expected Result	Actual Result	Status	Observations/ Comments	Bug ID
1	Gmail - Compose Mail – Able to Edit the email	User should be Logged in Gmail	1. Click on compose 2. Type some text in the Message box	User should be able to type the text in the Message box	User is able to type in the message box	PASS	NA	NA
2	Gmail – Send Email	Mail is composed and recipient name is added	1 Click on Send Button	Message should be sent to the recipient mentioned in the email	Mail Not received	FAIL	JS error occur when we click on Send button BUG ID # 451178	451178

Example: *Windows Calculator*

R-001:
The users should be able to add two numbers and view their result on the display.

Use Case: *UC01*	**Add Two Numbers**
Actors:	**User**
Purpose:	**Add two numbers and view their result**
Overview:	**The user inputs two numbers and (then adds them and) checks the result, displayed on the screen.**
Type:	**Primary, Real**
Cross References:	**R-001**

Typical Course of Events

Actor Action	System Response
1. The actor opens the calculator. The keypad and display screen appears.	
2. The actor input the first number by clicking on the keypad or using keyboard.	3. The digit is displayed on the screen.
4. The actor clicks or presses the "+" key.	
5.The actor then adds the second number as (2).	6. The pressed digit is displayed on the screen.
7. The actor clicks the "=" key.	8. The sum of the two digits is displayed on the screen.

Test Case ID: T-101 **Test Item:** Add Numbers

Wrote By: (tester name) **Documented Date:** 26th April 2005

Test Type: Manual **Test Suite#:** NA

Product Name: Windows Calculator **Release and Version No.:** V 1.0

Test case description:

Add any two Numbers

Operation procedure:

Open Calculator
Press "1"
Press "+"
Press "2"
Press "="

Pre-conditions:	Post-conditions:
Calculator Opened	Result Displayed
Inputs data and/or events:	**Expected output data and/or events:**
1 + 2 =	3

Required test scripts (for auto): NA

SOURCES OF INFORMATION FOR TEST CASE SELECTION

- In order to generate effective tests at a lower cost, test designers analyze the following sources of information:
  - Requirements and functional specifications
  - Source code
  - Input and output domains
  - Operational profile
  - Fault model

TESTING ACTIVITIES

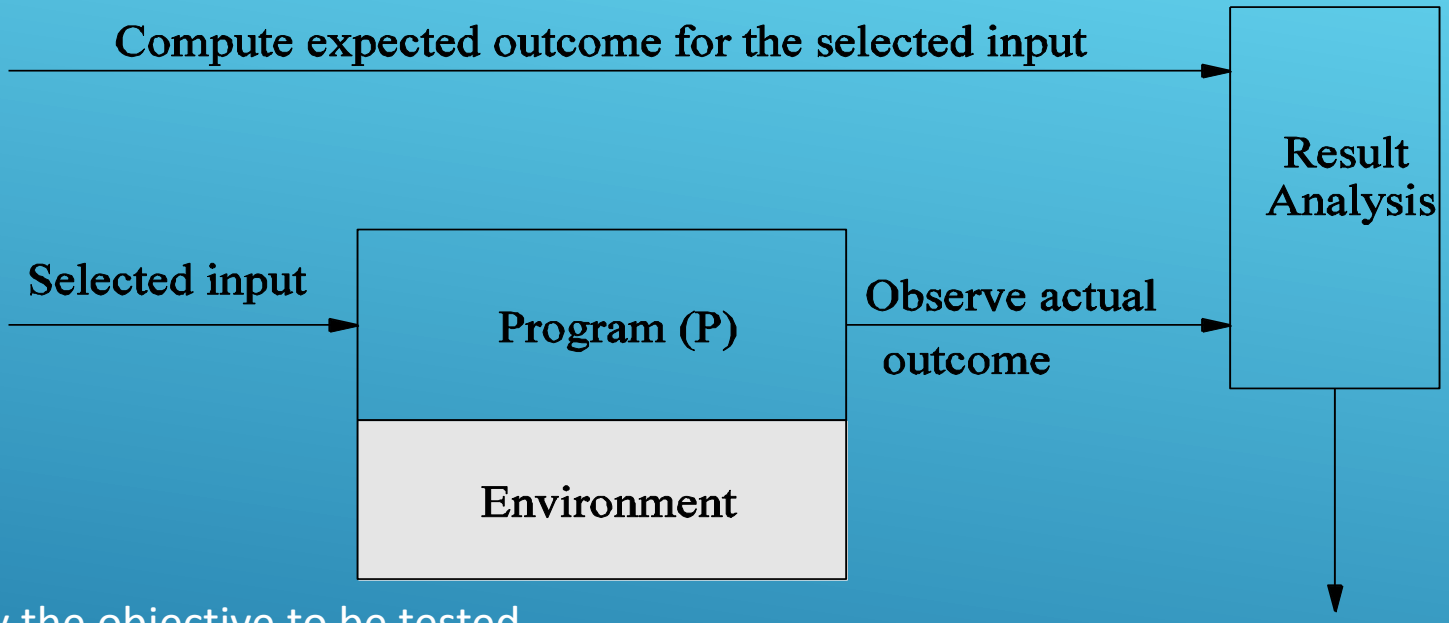


Figure: Different activities in process testing

- Identify the objective to be tested
- Select inputs
- Compute the expected outcome
- Set up the execution environment of the program
- Execute the program
- Analyze the test results

TEST LEVELS

- Performed at different levels involving complete system.
- Software goes through four stages:
  - Unit, Integration, System and Acceptance level testing.
- First three inside developing organization
- Last one is outside developing organization
- These four stages of testing is known as V.

TEST LEVELS

- **Unit testing**
  - Test Individual program units, such as procedure, methods, class, function in isolation
- **Integration testing**
  - Modules are assembled to construct larger subsystem and tested
- **System testing**
  - Includes wide spectrum of testing such as functionality and load
- **Acceptance testing**
  - Customer's expectations from the system
  - Various types of acceptance testing
  - User Acceptance Testing (UAT): Verifying if the user's specific requirements have been met.
  - Business Acceptance Testing (BAT): Assess whether the product meets the business goals set out in the design.
  - Alpha testing
  - Beta testing

V MODEL

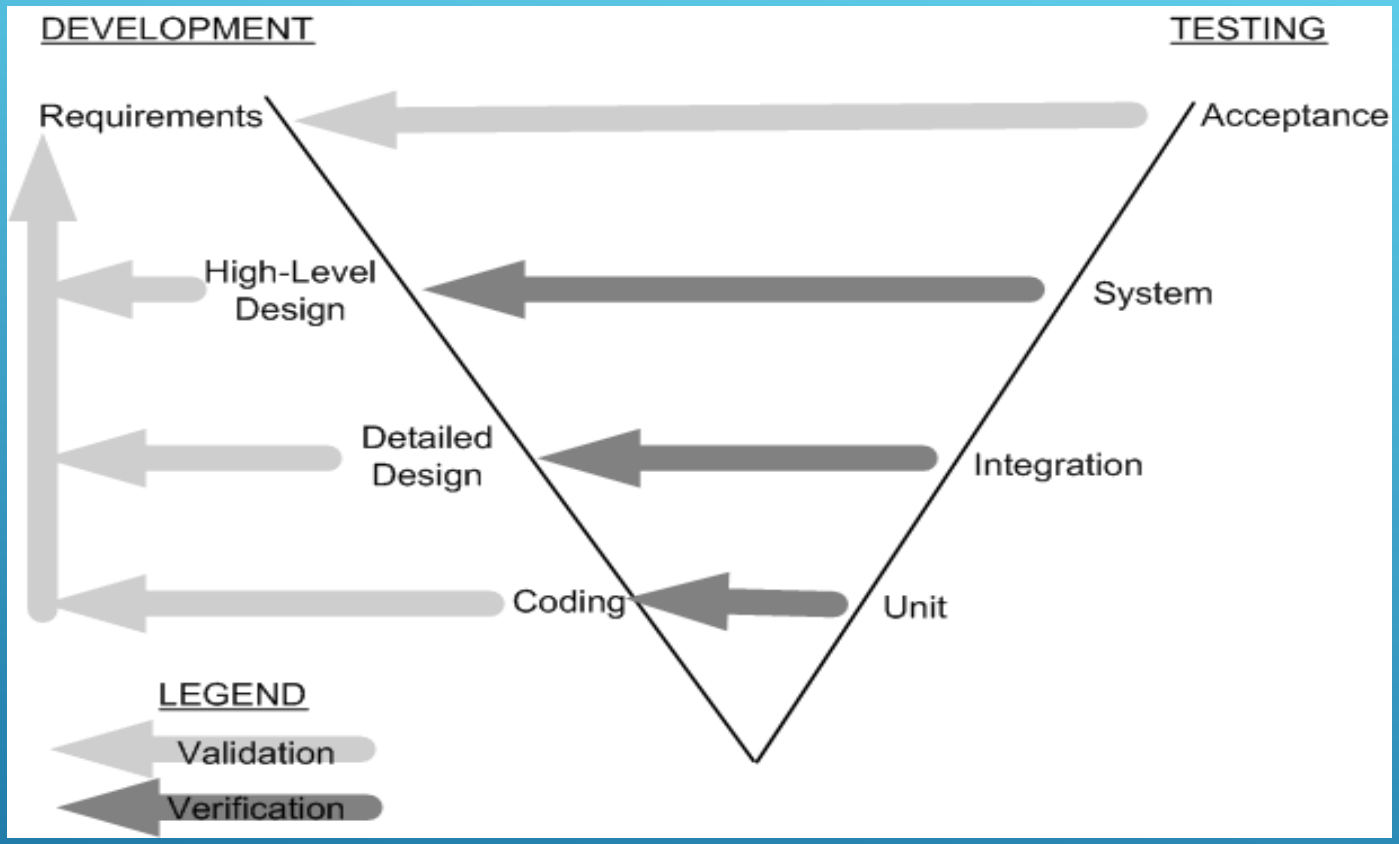


Figure : Development and testing phases in the V model

WHO PERFORMS THE TESTS?

- **Unit Testing** – done by the programmer and/or development team.
- **Integration Testing** – can be the development team or a testing unit.
- **System Testing** – usually done by an independent testing team (internal or external (consultants) team.
- For small companies, another testing team from another development team can be used and swapped.

VERIFICATION AND VALIDATION

VERIFICATION

- IEEE Standard defines Software Verification as the "process of evaluating a system or component to determine whether the products of a given development phase satisfy conditions imposed at the start of the phase".
- Verification is the process of examining a **work product** to discover its defects.
- Verify documents to determine if we are "building the product correctly".
- Also known as Static Testing.

VERIFICATION AND VALIDATION

VALIDATION

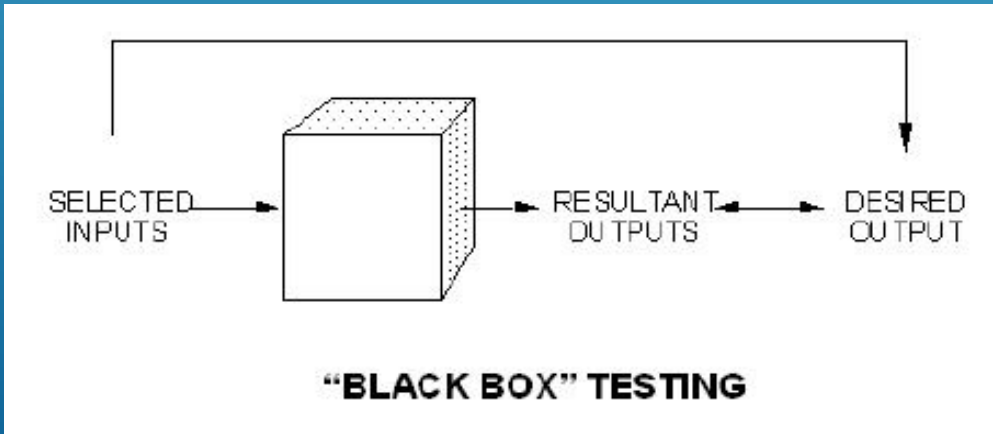
- Evaluation of software system that help in determining whether the product meets its intended use.
- Validation is the process of executing a ***product*** to expose its defects.
- Validate programs to determine if we have "Building the correct product".
- Also known as **Dynamic Testing.**

VERIFICATION VS. VALIDATION

VERIFICATION	VALIDATION
Confirming that Building product correctly	Confirming that building correct product
Review interim works like specification, design specification, code, user manual.	Is performed at the end of development phase of each part of system.
Verification considers quality attributes like consistency, correctness, completeness.	Validation considers only correctness and satisfaction.
Can be applied through Statistic Analysis Techniques like inspection, walkthrough, reviews, checklists, etc…	Can be applied by running system in its actual environment using verity of Tests.

BLACK BOX TESTING

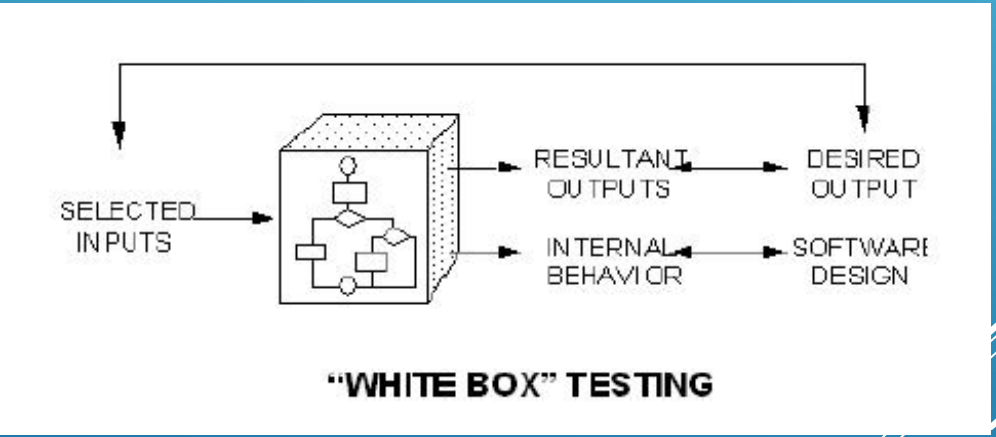
- **Black Box -** tests the functionality of the system by observing its external behavior and evaluating the outputs with requirements without knowledge of internal working of the system.
- Also called 'Functional/ Behavioral Testing' as it concentrates on testing of the functionality rather than the internal details of code.
- Test cases are designed based on the task descriptions of the system component.
- There are different techniques involved in Black Box testing,
  - like Equivalence Class, Boundary Value Analysis, Domain Tests, Orthogonal Arrays, Decision Tables, State Models, etc.



"BLACK BOX" TESTING

WHITE BOX TESTING

- **White Box -** tests the code structure of the system using criteria based on control flow, data flow and expected flows by having explicit knowledge of the internal workings of item (unit, component).
- Also called 'Structural Testing / Glass Box Testing' is used for testing the code keeping the system specifications in mind.
- Inner working is considered and thus Developers Test.
- White Box Testing Techniques
  - Statement Coverage
  - Branch Coverage
  - Path Coverage



WHITE-BOX VS BLACK-BOX TESTING

- **Black-box testing a.k.a. functional testing**
- Examines the program that is accessible from outside
- Applies the input to a program and observe the externally visible outcome
- It is applied to both an entire program as well as to individual program units
- It is performed at the external interface level of a system
- It is conducted by a separate software quality assurance group

- **White-box testing a.k.a. structural testing**
- Examines source code with focus on: Control and Data flow
- Control flow refers to flow of control from one instruction to another
- Data flow refers to propagation of values from one variable or constant to another variable
- It is applied to individual units of a program
- Software developers perform structural testing on the individual program units they write

REFERENCES

Book:

Software testing and quality assurance, Theory and Practice by *Kashira Sagar Naik and Priyadarshi Tripathy*

Chapter 1