UNIT- 4

SOFTWARE ANALYSIS AND TESTING

① There are two types of approaches for identifying defects in the software are -

Static analysis - Code is not executed but is evaluated through some process or some tools for locating defects.

Dynamic analysis - Code is executed, and the execution is used for determining defects.

② Code inspection -

It is often applied at the unit level. It can be viewed as "Static testing" in which defects are detected in the code not by executing the code but through a manual process. Three phases are

Planning → Self Review → Group review meeting.

③ Software Testing fundamentals -

Error → difference between actual output and the correct output.

Fault → Condition that causes a system to fail

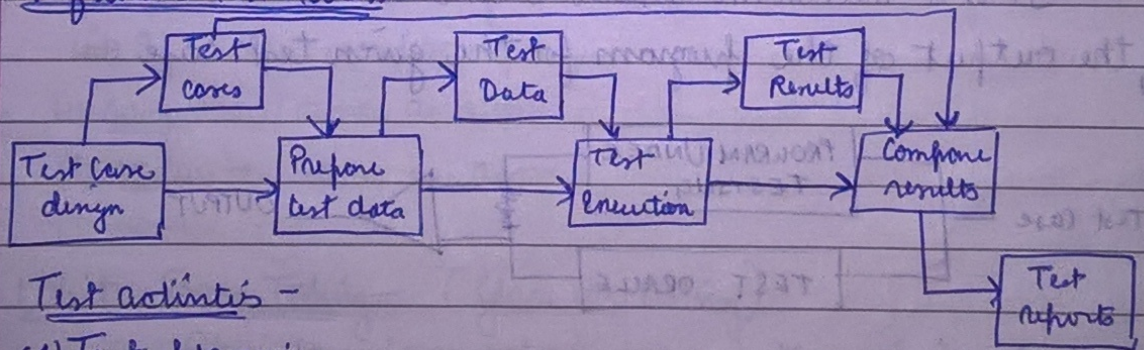
Failure → Inability of a system to perform required function.

Test case → set of test inputs and execution conditions

Test suite → Related test cases are executed to test some specific behaviour

Test Harness → testing framework

④ Software Test Process -



Test activities -

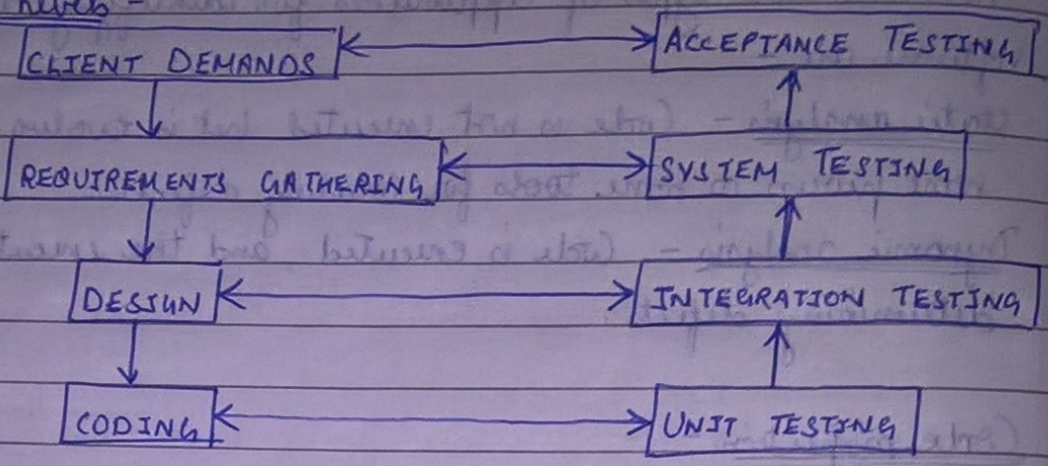
(1) Test planning

(2) Test case dingn

(3) Test execution

(4) Data collection (5) Effective evaluation

⑤ Testing levels -



⑥ Test Criteria - Reliability and Validity

A test criterion is reliable if all the test cases detect same x of errors.

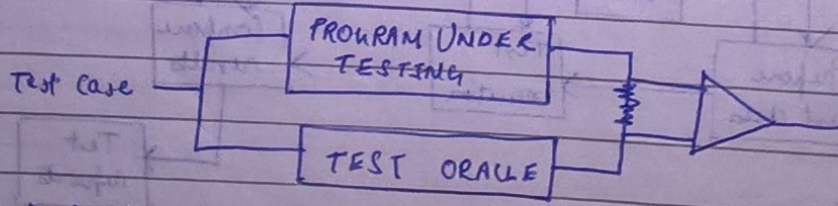
A test criterion is valid if for any error in the program the is some xt which causes error in the program.

⑦ Test Case Design -

Create a set of tests that are effective in testing. Two g
(1) Maximize the number of errors detected
(2) Minimize the number of test cases.

⑧ Test Oracles -

It is a mechanism which is used to check the correctness of the output of the program for the given test case.



⑨ Test techniques -

Two types -

(1) Black box testing - demonstrate software functions are operational
(2) White box testing - Procedural details are closely examined.

(10) Black-Box Testing - (Behavioural Testing)

focus on functional requirements of the software.

→ Uncovers following types of errors -

(1) Incorrect or missing functions

(2) Interface errors

(3) Errors in data structures

(4) Performance errors

5) Initialization of termination errors

→ Black-box techniques are -

(1) Equivalence Partitioning -

divides the input domain into classes of data and from this data test cases can be drived

If input condition specifies - (states of input condition)

(i) range → one valid and two invalid equivalence classes are defined

(ii) Specific value → one valid & two invalid equivalence classes are defined

(iii) set → one valid and one invalid equivalence classes are defined

(iv) Boolean → one valid and one invalid equivalence classes are defined

(2) Boundary Value Analysis (BVA) -

Done by checking boundary conditions.

If input condition specifies -

(i) range (x-y) → test cases for value x and y and value above & below x & y.

(ii) values → test cases of min. and max. values & also just above & below min & max

(iii) If output condition specifies -

(i) range → same as above

(ii) values → same as above

(1) White Box Testing - (Glass box testing)

It is based on close examination of procedural details.

Test cases are derived for -

Examining all indefendent paths, Exercising all logical paths, executing all loops, exercising internal data structures.

→ White-Box techniques are -

(1) Structural Testing -

Test cases is according to program structure that is executing all program statements.

(2) Conditional Testing -

To test the logical conditions in the program module.

(3) Loop Testing -

To test the loop constructs.

(i) Simple loops - n = m means m pass through the loop (n > m)

(ii) Nested loops - Start with innermost loops.

(iii) Concatenated loops - Same as simple loop test.

(iv) Unstructured loops - Cannot done effectively, needs to redesign.

(4) Path Testing -

It is a structural testing strategy. Steps are -

STEP-1 - Design the flow graphs for the program or a component

STEP 2 - Calculate the cyclomatic complexity

STEP 3 - Select a basis set of path.

STEP 4 - Generate test cases for these paths.

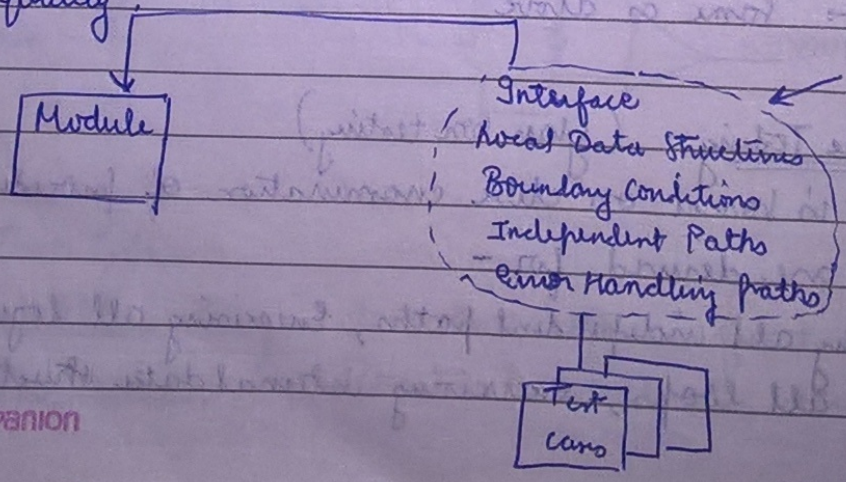
Cyclomatic complexity = Total no. of regions in the flow graph

= E - N + 2 (Edges → E, Node → N)

= P + 1 Predicate (decision making)

(12) Unit Testing -

Individual components are tested independently to ensure their quality.



(13) Unit Testing Frameworks -

Third-party products that are not distributed as part of the compiler suit. Eg:- TustMock, Isolator.NET, Isolator++ etc.

(14) Integration Testing -

Group of dependent components are tested together Uncover errors are

(1) Design and construction of software architecture
(2) Integrated functions or operations at subsystem level.
(3) Interfaces and interaction between them
(4) Resource integration and/or environmental integration.

Two approaches are -

(1) Non-incremental integration → Bigg Bang (approach is simple)
(2) Incremental integration -
   (1) Top down integration Testing -
   (2) Bottom up integration Testing
   (3) Regression Testing - Used to check ~~for~~ for defects propagated to other modules by changes made to existing program
   (4) Smoke Testing - Used for time critical projects wherein the project needs to accessed on frequent basis.

(15) Validation Testing - Testing is based on requirements Uncover errors are -

(1) System input/output
(2) System functions and information data
(3) System interfaces with external parts
(4) User interfaces
(5) System behaviour on Performance.

→ Acceptance Testing - To test ~~what~~ the software works correctly in the user work environment. Types are -

Alpha test - developer or customer under supervision of developer.
Beta test - Customer without the developer being present

(16) System Testing -

Main focus are system functions, performance, reliability, recoverability, installation, behaviour, user operations etc.

Various types are -

(1) Recovery Testing (2) Security Testing.
(3) Stress Testing (4) Performance testing.

(17) Test Plan - It gets generated during the development. Struc of test plan -

(1) Testing Process
(2) Requirements Traceability
(3) Tested Items
(4) Testing Schedule
(5) Test Recording Procedure
(6) Hardware and Software Requirements
(7) Constraints.

(18) Testing Tools - Used to automate the testing activity. Two type

(1) Static Testing tool -

Static analyzers → analyze the code systematically
Code inspectors → Coding Standards can be incorporated in progra

(2) Dynamic Testing tool -

Activities are input validation, stud processing, displaying resul and Test planning.

Various tools are output comparator, coverage analyzer an test data generator.

(19) Testing Metrics - Cyclomatic complexity and Halstead's metrics.

Halstead's metrics - Estimating the testing efforts (e)

$e = V/PL$ V → program volume, PL → program level.

$PL = \frac{1}{[(n_1/2) \times (N_2/n_2)]}$

n_1 = Total distinct operators N_2 → all operand
n_2 = Total distinct operands in program

Percentage of overall testing effort = $\frac{\text{Testing effort of specific module}}{\text{Testing effort of all the modules}}$