TESTING TECHNIQUES

Assistant Professor
Kusum Lata Dhiman
Computer Science & Engineering

Software Testing Fundamentals

- The program is executed with desired input(s) and the output(s) is/are observed accordingly.

- The observed output(s) is/are compared with expected output(s). If both are same, then the program is said to be correct as per specifications
- "Testing is the process of executing a program with the intent of finding faults"
- Who should test the software?
- Role of developer, tester, customer etc in testing process
- Verification and Validation
- Test, Test Case and Test Suite
- Acceptance Testing: customer is involved during acceptance testing
- Alpha Testing: conducted at the developer's site by the customer.
- Beta Testing: conducted by potential customers at their sites

Image source : Google

TESTING TECHNIQUES

Assistant Professor
Kusum Lata Dhiman
Computer Science & Engineering

Software Testing Fundamentals

- Analyzing the requirement from the client
- Participating in preparing the test plan
- Preparing test scenario, test cases.
- Defect tracking
- Preparing suggestions to improve the quality
- Communication with test lead/Test manager
- Conducting review meeting with the team

Image source : Google

Software Tester Roles

- A Software tester (software test engineer) should be capable of designing test suites and should have the ability to understand usability issues.
- Such a tester is expected to have sound knowledge of software test design and test execution methodologies.
- It is very important for a software tester to have great communication skills so that he can interact with the development team efficiently.
- A Software Tester is responsible for designing testing scenarios for usability testing.
- He is responsible for conducting the testing, thereafter analyze the results and then submit his observations to the development team.

Image source : Google

Software Tester Roles

- He may have to interact with the clients to better understand the product requirements or in case the design requires any kind of modifications.
- Software Testers are often responsible for creating test-product documentation and also has to participate in testing related walk through.
- Creation of test designs, test processes, test cases and test data.
- Carry out testing as per the defined procedures.
- Participate in walkthroughs of testing procedures.
- Prepare all reports related to software testing carried out.
- Ensure that all tested related work is carried out as per the defined standards and procedures

Image source : Google

White Box Testing and Black Box

- **White-box testing**, sometimes called glass-box testing, is a test case design method that uses the control structure of the procedural design to derive test cases
- Guarantee that all independent paths within a module have been exercised at least once
- exercise all logical decisions on their true and false sides
- execute all loops at their boundaries and within their operational bounds
- exercise internal data structures to ensure their validity.
- Basis path testing is a white-box testing technique , in which Test cases derived to exercise the basis set are guaranteed to execute every statement in the program at least one time during testing.

Image source : Google

White Box Testing - BASIS PATH TESTING

- The **basis path method** enables the test case designer to derive a logical complexity measure of a procedural design and use this measure as a guide for defining a basis set of execution paths.
- This method is used to select values for test cases
- Whatever methods & techniques we are going to see will help us to select data/test cases for testing software
- The quality of test cases selected helps testers to ensure that the software is working according to specifications given

Image source : Google

White Box Testing - flow graph notation

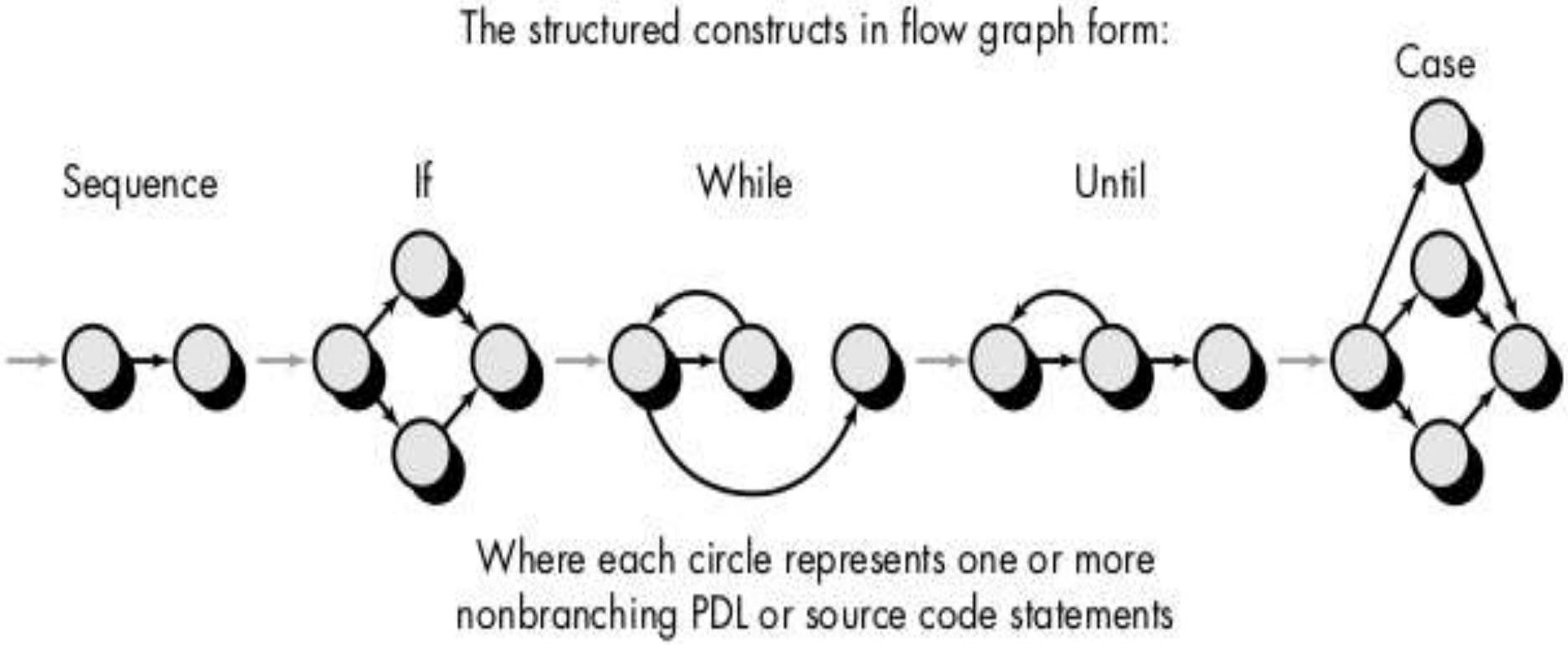


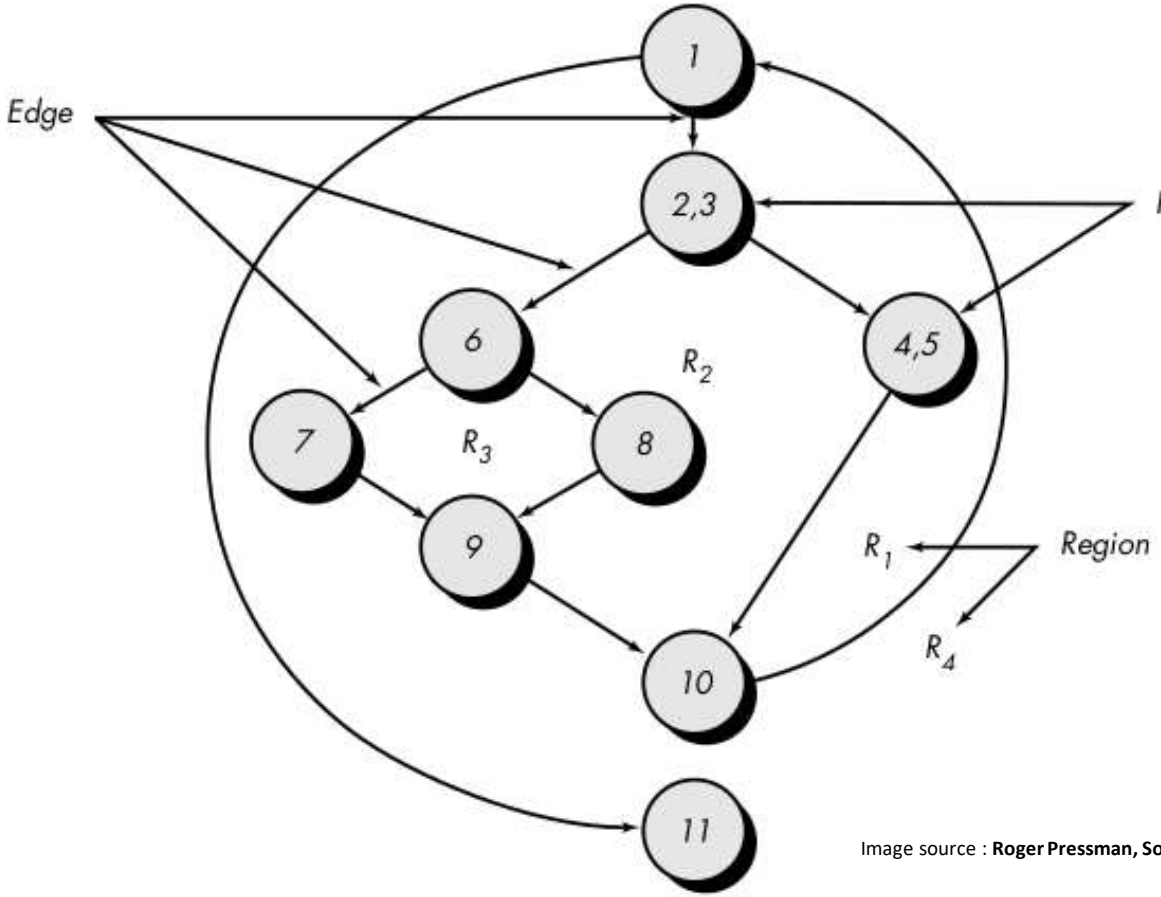
Image source : **Roger Pressman, Software engineering-A practitioner's Approach, McGraw-Hill International Editions**

White Box Testing - Cyclomatic Complexity

- Cyclomatic complexity is a software metric that provides a quantitative measure of the logical complexity of a program.
- When used in the basis path testing method, the value computed for cyclomatic complexity defines the number of independent paths in the basis set of a program
- Cyclomatic complexity provides us with an upper bound for the number of tests that must be conducted to ensure that all statements have been executed at least once.
- An **independent path** is any path through the program that introduces at least one new set of processing statements or a new condition.

Image source : Google

White Box Testing - Cyclomatic Complexity



path 1: 1-11
path 2: 1-2-3-4-5-10-1-11
path 3: 1-2-3-6-8-9-10-1-11
path 4: 1-2-3-6-7-9-10-1-11

So on…….

Image source : **Roger Pressman, Software engineering-A practitioner's Approach, McGraw-Hill International Editions**

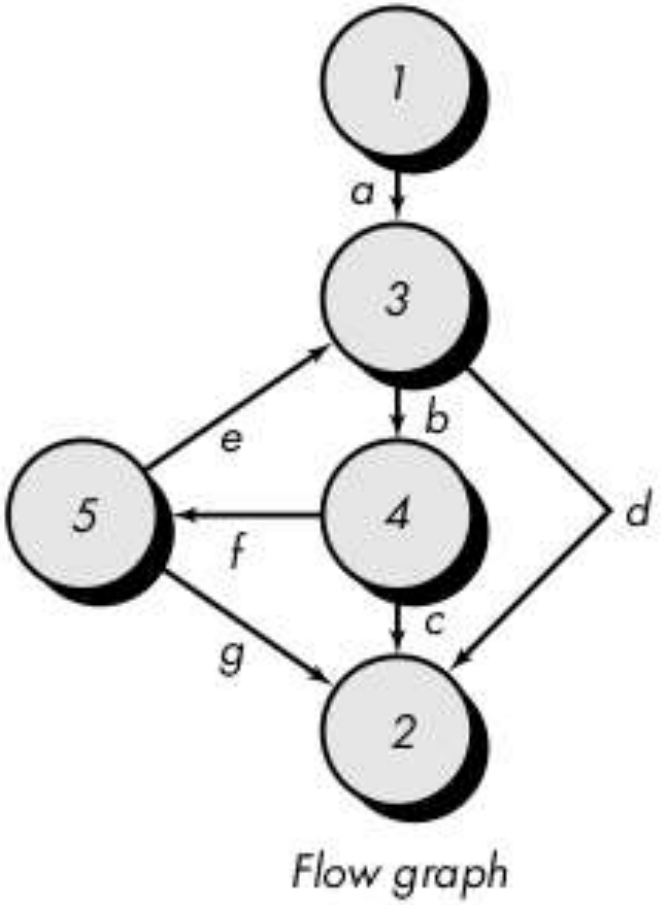
White Box Testing

- Cyclomatic complexity, V(G), for a flow graph, G, is defined as **V(G) = E - N + 2** , where E is the number of flow graph edges, N is the number of flow graph nodes.

- Cyclomatic complexity, V(G), for a flow graph, G, is also defined as
- V(G) = P + 1 , where P is the number of predicate nodes contained in the flow graph G
- The flow graph has four regions.
- V(G) = 11 edges - 9 nodes + 2 = 4.
- V(G) = 3 predicate nodes + 1 = 4.

Image source : Google

White Box Testing - Cyclomatic Complexity



Flow graph

Node	Connected to node 1	2	3	4	5	Connections
1			1			1 – 1 = 0
2						
3		1		1		2 – 1 = 1
4		1			1	2 – 1 = 1
5		1	1			2 – 1 = 1

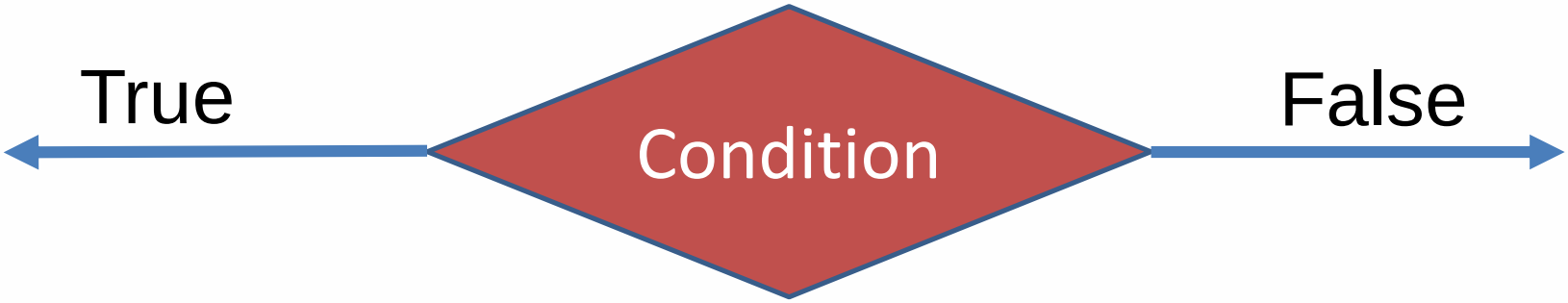
Graph matrix

3 + 1 = 4 ← Cyclomatic complexity

Image source : **Roger Pressman, Software engineering- A practitioner's Approach, McGraw-Hill International Editions**

White Box Testing - CONTROL STRUCTURE TESTING

- **Condition Testing**: Condition testing is a test case design method that exercises the logical conditions contained in a program module.
- Branch testing: compound condition C, the true and false branches of C and every simple condition in C need to be executed at least once



- Boolean expression with n variables, all of 2^n possible tests are required (where n>0)

Image source : Google

White Box Testing - CONTROL STRUCTURE TESTING

- **Data Flow Testing**: The data flow testing method selects test paths of a program according to the locations of definitions and uses of variables in the program.
- Data Flow Testing is a type of structural testing.
- It has nothing to do with data flow diagrams.
- It is concerned with:

Statements where variables receive values

Statements where these values are used or referenced

It can be to find out :

- A variable is defined but not used or referenced
- A variable is used but never defined
- A variable is defined twice before it is used

Image source : Google

White Box Testing - CONTROL STRUCTURE TESTING

- **Loop Testing:** Loop testing is a white-box testing technique that focuses exclusively on the validity of loop constructs.
- **Simple loops:**
- 1. Skip the loop entirely.
- 2. Only one pass through the loop.
- 3. Two passes through the loop.
- 4. m passes through the loop where m < n.
- 5. n - 1, n, n + 1 passes through the loop
- **Nested loops**
- **Concatenated loops**
- **Unstructured loops**

Image source : Google

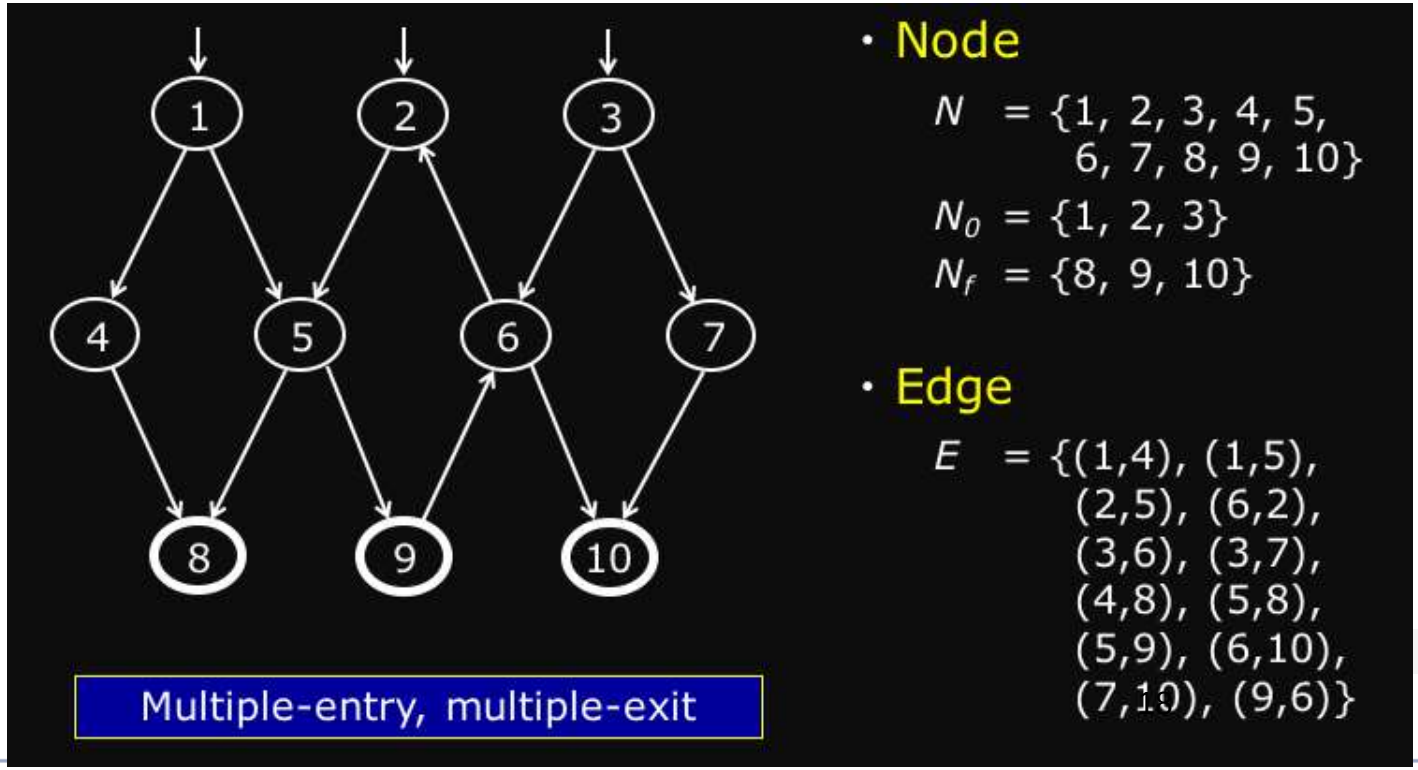
Black Box

- Black-box testing/behavioral testing/Functional testing, focuses on the functional requirements of the software.
- Black-box testing is not an alternative to white-box techniques. Rather, it is a complementary approach that is likely to uncover a different class of errors than white-box methods.
- Black-box testing attempts to find errors in the following categories:
- Incorrect or missing functions
- interface errors
- errors in data structures or external database access
- behavior or performance errors
- initialization and termination errors.

Image source : Google

Graph: Nodes and Edges

- Node represents : statement, State, Method,, Basic block
- Edge represents: Branch, Transition, Method call



Graph: Nodes and Edges

- A test path represents the execution test cases:
- Some test paths can be executed by many test cases
- Some test paths cannot be executed by any test cases
- Some test paths cannot be executed because they are infeasible

- Minimal set of test paths = the fewest test paths that will satisfy test requirements

Image source : Google

Equivalence Partitioning

- Equivalence partitioning is a black-box testing method that divides the input domain of a program into classes of data from which test cases can be derived.
- An equivalence class represents a set of valid or invalid states for input conditions.
- Typically, an input condition is either a specific numeric value, a range of values, a set of related values, or a Boolean condition.
- For example , if a system accepts names with character ength between 6 to 12 characters then input data will be divided into 3 classes
- 1. strings having length less than 6 i.e. invalid range
- 2. strings having length between 6 to 12 i.e. valid range
- 3. string having length greater than 12 i.e. invalid range

Image source : Google

Boundary Value Analysis

- Boundary value analysis leads to a selection of test cases that exercise bounding values.
- Boundary value analysis is a test case design technique that complements equivalence partitioning.
- Rather than selecting any element of an equivalence class, BVA leads to the selection of test cases at the "edges" of the class.
- If an input condition specifies a range bounded by values a and b, test cases should be designed with values a and b and just above and just below a and b.
- For example, if a system expected to works fine for 5000 users then it will be tested for 4999 users and performance will be checked.

Image source : Google

Parul® University

NAAC GRADE A++

https://paruluniversity.ac.in/