

# Software Testing

## Chapter 2: Types of software testing

### White Box Testing:

White box testing is a testing technique, that examines the program structure and derives test data from the program logic/code

- White box testing is a strategy in which testing is based on:
  - the internal paths
  - structure
  - implementation of the software under test (SUT).
- Also known as Structural testing, Clear box Testing, and Glass box Testing
- Requires detailed programming skills

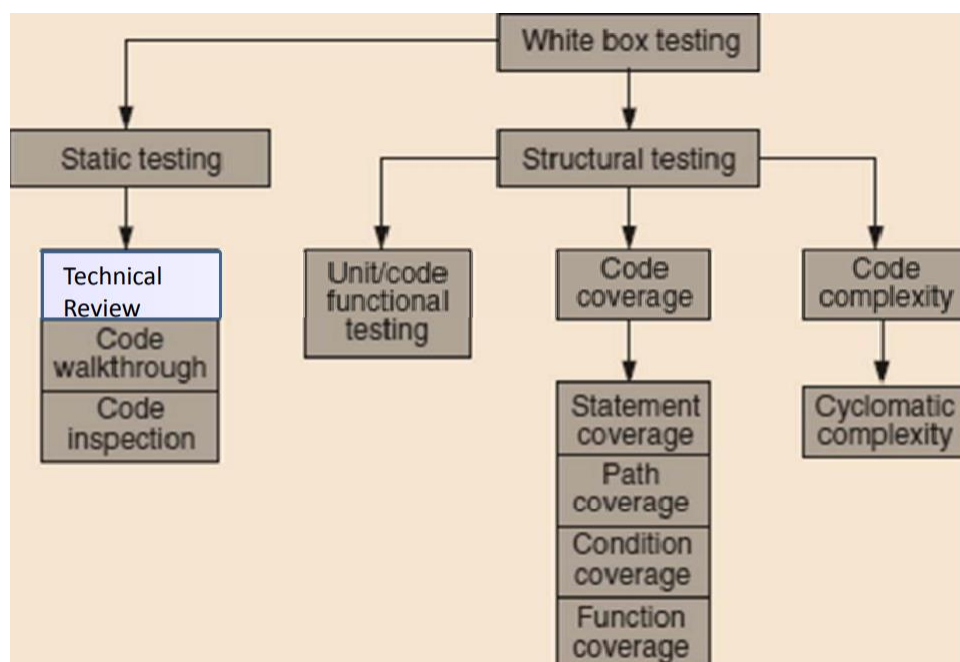
The goal of White box testing is that the test case ensure that all internal structures of the program are developed according to design specifications.

Test Cases are designed to:

- (1) exercise Independent paths within a module or unit;
- (2) exercise logical decisions on both their true and false side;
- (3) execute loops at their boundaries and within their operational bounds
- (4) exercise internal data structures to ensure their validity

Why white box techniques?

- Find bugs early
- Find bugs that are hard to find with BB techniques
- Provides good ideas to BB testers



### Static testing:

- Static testing is a software testing technique in which the software is tested without executing the code/components/system that is to be tested.
- High quality documents lead to high quality product
- Static white box testing is the process of carefully and methodically reviewing the software design, architecture or code for bugs without executing it .
- To avoid the errors, we will execute Static testing in the initial stage of development because it is easier to identify the sources of errors, and it can fix easily.
- The usage of static testing will decrease the testing cost, development timescales, and time.
- Static testing include: • reviews • static analysis

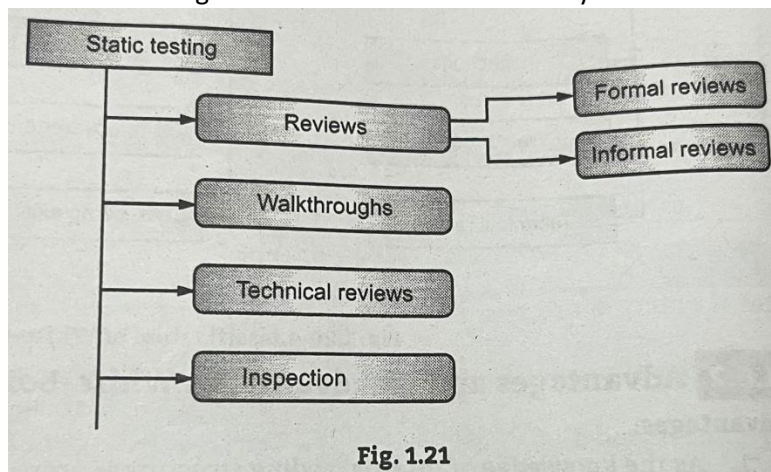


Fig. 1.21

### Walkthrough:

- Walkthrough is the next step up in formality from peer reviews.
- In a walkthrough the programmer who wrote the code, formally presents it to a small group of five or so other programmers and testers.
- The team members ask questions and spot possible errors against development standards and other issues.
- The meeting is usually led by the author of the document under review and attended by other members of the team.
- Review sessions may be formal or informal.
- The reviewers should receive copies of the software in advance of the review so they can examine it and write comments and questions that they want to ask at the review.
- The scribe, who is not the author, marks the minutes of meeting and note down all the defects/issues for tracking to closure.
- The main purpose of walkthrough is to enable learning about the content of the document under review to help team members gain an understanding of the content of the document and also to find defects.

### Example for using walkthroughs

- Walkthrough of documents
- Walkthrough of drafts for user interfaces
- Walkthrough of business process modelling data

**Walkthrough: advantage**

- Little effort in preparing a review session, but it is an open- end session
- Session can be initiated on short term notice
- To detect defects in developed software products.

**Walkthrough: drawbacks**

- Author has a great influence on the outcome: since she/he moderates the review, there is a danger of domination by the author
- Little control possible, since author is also in charge of any follow-up activities

**Reviews:**

- Reviews are done in order to improve product quality
- Reviews are used to verify the correct transition from one phase to another phase
- Detecting defects early saves costs
- During reviews, the following

defects might be detected:

- defects in the specifications
- defects in the design and architecture
- defects in the interface specifications
- insufficient maintainability
- deviations from agreed standards

**Technical reviews:**

- A Technical review is a static white-box testing technique which is conducted to spot the defects early in the life cycle that cannot be detected by black box testing techniques.
- Technical Reviews are documented and uses a defect detection process that has peers and technical specialist as part of the review process.
- The Review process doesn't involve management participation.
- It is usually led by trained moderator who is NOT the author.
- The report is prepared with the list of issues that needs to be addressed.

**Activities of a Formal Review**

- Planning
- Defining the entry and exit criteria for formal review
- Checking entry criteria
- Noting potential defects, questions and comments
- Review meeting
- Examining/Evaluation/Recording
- Rework
- Follow up
- Checking on exit criteria

**1. Planning**

- Defining the review criteria

- Selecting the personnel
- Allocating roles
- Defining the entry and exit criteria for more formal review types (e.g., inspections)
- Selecting which parts of documents to review
- Checking entry criteria (for more formal review types)

## 2. Kick-off

- Distributing documents
- Explaining the objectives, process and documents to the participants

## 3. Individual preparation

- Preparing for the review meeting by reviewing the document(s)
- Noting potential defects, questions and comments

## 4. Examination/evaluation/recording of results (review meeting)

- Discussing or logging, with documented results or minutes (for more formal review types)
- Noting defects, making recommendations regarding handling the defects, making decisions about the defects
- Examining/evaluating and recording issues during any physical meetings.

## 5. Rework

- Fixing defects found (typically done by the author)
- Recording updated status of defects (in formal reviews)

## 6. Follow-up

- Checking that defects have been addressed
- Gathering metrics
- Checking on exit criteria (for more formal review types)

## ❖ Roles and Responsibilities

- (Project) Manager : Managers are basically included and involved in the reviews as he/she simply decides on execution of reviews, allocates time in project schedules, and generally identifies whether or not objectives of review project have been met.
- Moderator : Moderator, also known as review leader, generally leads review process. It simply co-ordinates with author and checks entry criteria for review. During review, they also lead discussion.
- Author : As the writer of 'document under review', author's main goal must be to learn and know as much as possible with regard to improving and increasing quality of document. He/She is writer of documents under review and aims to gain maximum from review. They also help in improving document quality.
- Reviewers (inspectors or checkers) : Reviewers only have to check all of the defects and errors and then further improvements also in accordance to business specifications, standards, and domain knowledge. Reviewers must be technical.

- Scribe (recorder) : As the name suggests, scribe or recorder simply has to record each and every defect and error that is found and any suggestions or feedback given in the meeting for improvement of process. Basically, author plays role of scribe, but it's very good and advantageous to have another person to scribe so that author can only concentrate on review.

### **Inspection:**

Inspection is the most formal form of reviews, a strategy adopted during static testing phase. Characteristics of Inspection:

- Inspection is usually led by a trained moderator, who is not the author. Moderator's role is to do a peer examination of a document and to assure that the rules are followed.
- Inspection is most formal and driven by checklists and rules.
- This review process makes use of entry and exit criteria.
- It is essential to have a pre-meeting preparation.
- Inspection report is prepared and shared with the author for appropriate actions.
- Post Inspection, a formal follow-up process is used to ensure timely corrective action.
- Aim of Inspection is NOT only to identify defects but also to bring in for process improvement.

The participants are called inspectors. Inspectors review the code from different perspective, such as a user, tester, or a product support person. Inspectors are also assign to review the code backwards.

inspection: advantages

- Well organized, formal session with clear roles
- Needs intensive preparation and follow up activities

inspection: drawbacks

- Moderator and scribe are necessary

### **Structural testing/ Dynamic White Box Testing:**

- Structural testing is also known as dynamic white box testing.
- The dynamic testing is working with the software by giving input values and verifying if the output is expected by implementing a specific test case.
- This can be done manually or with an automation process.
- The dynamic testing can be done when the code is executed in the run time environment.

### Code functional testing:

#### Control flow testing:

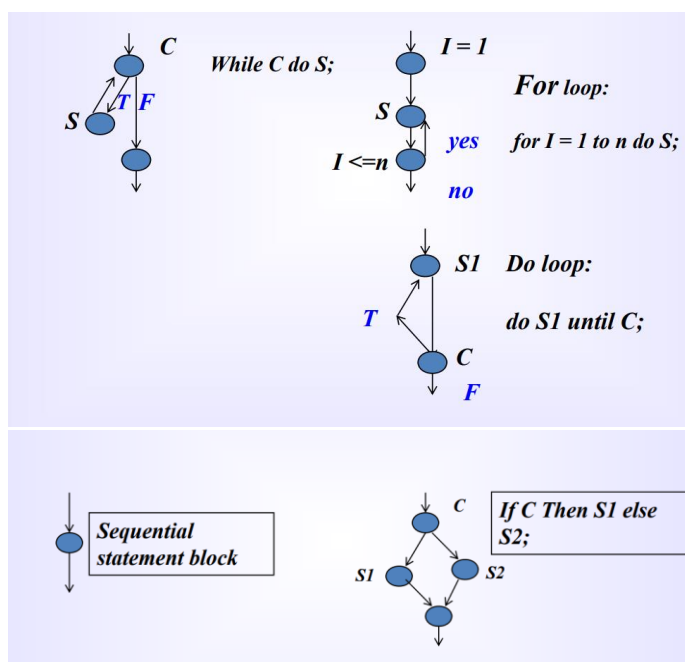
- The SUT's(system under tests) implementation is analysed.
- Paths through the SUT are identified.
- Inputs are chosen to cause the SUT to execute selected paths - path sensitization
- Expected results for those inputs are determined.
- Actual outputs are compared with the expected outputs.
- Can be applied at all levels of system development
- It is path testing.
- Identifies the execution paths through a module of program code
- Creates and executes test cases to cover those paths.

#### Advantages of Control flow testing:

- It detects almost half of the defects that are determined during the unit testing.
- It can be performed manually or automated as the control flow graph that is used can be made by hand or by using software also.
- every path within the software gets identified

#### Disadvantages of Control flow testing:

- Unlikely to find spurious features.
- The number of execution paths may be so large
- The tester must have the programming skills
- The test cases chosen may not detect data sensitivity errors.



### Data flow testing:

- 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.
  - life cycle of variable Created -> used -> killed/ Destroyed
  - Variables can be used
    - in computation
    - in conditionals
  - Possibilities for the first occurrence of a variable through a program path
    - ~d it is defined (d)
    - ~u it is used (u)
    - ~k it is killed or destroyed (k)
  - Data Flow Testing is used to find the following issues-
    - 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
  - Static data flow testing: The declaration, usage, and deletion of the variables are examined without executing the code. A control flow graph is helpful in this.
  - Dynamic data flow testing: The variables and data flow are examined with the execution of the code.
  - The data flow testing process is to choose enough test cases so that:
    - Every "define" is traced to each of its "uses"
    - Every "use" is traced from its corresponding "define"
- ```
{ // begin outer block
int x; ...; //define x
{ // begin inner block
print(x); ...; //use x
}
}
} //kill x after program executes
```

### Code Coverage testing:

- Coverage is a measure of the completeness of the set of test cases.
- You must attempt to enter and exit every module, execute every line of code, and follow every logic and decision path through the software.
- Examining the software at this level of detail is called code-coverage analysis.
- Code-coverage analysis is a dynamic white-box testing technique because it requires you to have full access to the code to view what parts of the software you pass through when you run your test cases.
- The simplest form of code-coverage analysis is using your compiler's debugger to view the lines of code you visit as you single-step through the program.

### Advantages of Using Code Coverage :

- It helps in determining the performance and quality aspects of any software.
- It helps in easy maintenance of code base.

- It helps in exposure of bad, dead, and unused code.
- It helps in finding new test cases which are uncovered.

Disadvantages of Using Code Coverage :

- Some times it fails to cover code completely and correctly.
- It can not guarantee that all possible values of a feature is tested with the help of code coverage.
- It fails in ensuring how perfectly the code has been covered.

### **Program statement coverage:**

*Statement Coverage = (Number of statements executed)/(Total Number of statements)\*100.*

- Statement coverage is a measure of the percentage of program statements that are run when your tests are executed.
- The most straightforward form of code coverage is called statement coverage or line coverage.
- If you're monitoring statement coverage while you test your software, your goal is to make sure that you execute every statement in the program at least once.
- Statement coverage covers only the true conditions.
- You could run your tests and add test cases until every statement in the program is touched.

Advantages:

- It measures the quality of the code.
- Checks the flow of different paths in the program.

Drawbacks:

- Cannot check the false condition.
- Different input values to check all the conditions.

### **Branch coverage testing:**

*Decision Coverage = (Number of decision/branch outcomes exercised)/(Total number of decision outcomes in the source code)\*100.*

- Attempting to cover all the paths in the software is called path testing. The simplest form of path testing is called branch coverage testing.
- Branch coverage is also called as decision coverage.
- Decision or branch coverage is a measure of how many of the Boolean expressions of the program have been evaluated as both true and false in the testing.
- It aims to ensure that each one of the possible branch from each decision is executed atleast once.



### **Condition coverage testing:**

*Condition Coverage = (Number of executed operands)/(Total Number of Operands)\*100.*

- Condition coverage is also known as predicate coverage.
- The number of Boolean condition/expression statements executed in the conditional statement is called condition coverage testing.
- As with branch coverage, conditional code coverage analyzers can be configured to consider conditions when reporting their results.
- If you test for all the possible conditions, you will achieve branch coverage and therefore achieve statement coverage.
- Each of the Boolean expression have been evaluated to both TRUE and FALSE.
- There are no industry standard objectives for condition coverage, but we suggest that you keep condition coverage in mind as you develop your test cases.
- Our condition coverage revealed that some additional test cases were needed.

### **Method/Function Coverage testing:**

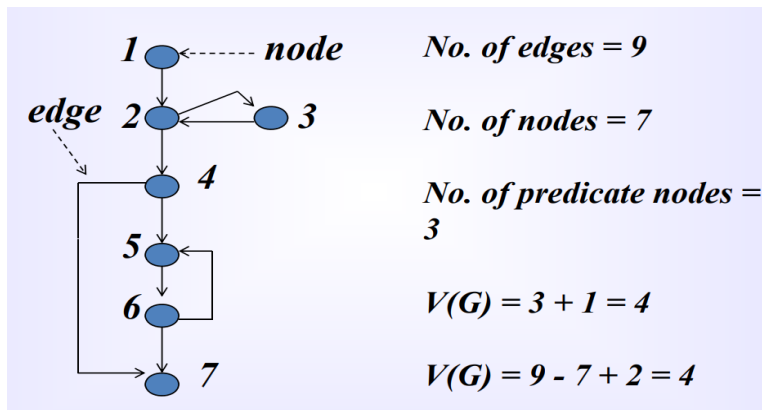
*Function Coverage = (Number of functions called)/(Total number of function)\*100.*

- This is to identify how many program functions are covered by the test cases.
- The requirements of a product are mapped into functions during the design phase and each of the functions forms a logical unit.
- While providing the function coverage, test cases can be written so as to exercise each of the different functions in the code.
- We can test how many times a given function is called.
- Functions are easier to identify in a program.
- It is easier to achieve 100% function coverage.
- Functions have a more logical mapping to requirements and hence can provide more direct correlation to test coverage of the product.
- Function coverage can help in improving the performance and quality of the product.

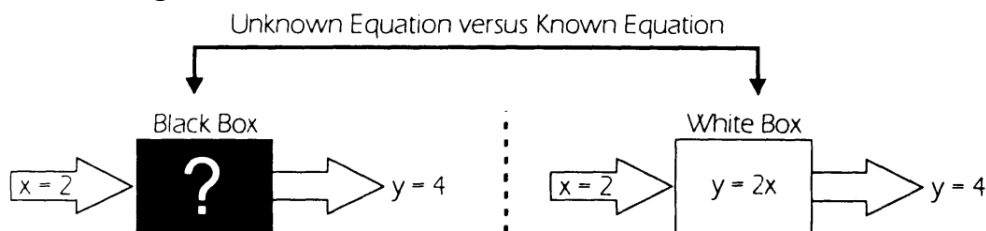
### **Cyclomatic complexity:**

- Cyclomatic complexity is a software metrics
- Defines the number of independent paths in a program.
- Cyclomatic complexity measures the number of decision logic through a program module.
- Cyclomatic complexity  $V(G) = E - N + 2$  where E= number of graph edges, N= number of graph nodes.
- Cyclomatic complexity,  $V(G) = P + 1$  where P = number of predicate nodes in the flow graph.
- Each node that contains a condition is called a predicate node and is characterized by two or more edges originating from it

Note: Independent Paths : An independent path in the control flow graph is the one which introduces at least one new edge that has not been traversed before the path is defined



### Black Box testing:



- Black box testing is a technique of software testing which examines the functionality of software without peering into its internal structure or coding.
- The primary source of black box testing is a specification of requirements that is stated by the customer.
- In this method, tester selects a function and gives input value to examine its functionality, and checks whether the function is giving expected output or not.
- Black Box testing is also known as closed box testing/Behavioural testing.
- Black Box testing helps to find:
  - Incorrect or missing functions
  - Interface errors
  - Errors in data structures or external data base access
  - Behaviour or performance errors
  - Initialization and termination errors
- Types of Black box testing
  - Requirement Based testing
  - Positive testing
  - Negative testing
  - Equivalence class partitioning
  - Boundary value analysis
  - Decision Table based testing

### Advantages of Black Box Testing

- more effective on larger units of code than glass box testing
- tester needs no knowledge of implementation, including specific programming languages

- tester and programmer are independent of each other
- tests are done from a user's point of view
- bb will help to expose any ambiguities or inconsistencies in the specifications
- test cases can be designed as soon as the specifications are complete

#### Disadvantages of Black Box Testing

- only a small number of possible inputs can actually be tested, to test every possible input stream would take nearly forever
- without clear and concise specifications, test cases are hard to design
- there may be unnecessary repetition of test inputs if the tester is not informed of test cases the programmer has already tried
- may leave many program paths untested
- cannot be directed toward specific segments of code which may be very complex (and therefore more error prone)

#### **Requirement Based testing:**

Requirements-based testing is a testing approach in which test cases, conditions and data are derived from requirements. It includes functional tests and also non-functional attributes such as performance, reliability or usability.

Stages in Requirements based Testing:

- Defining Test Completion Criteria - Testing is completed only when all the functional and non-functional testing is complete.
- Design Test Cases - A Test case has five parameters namely the initial state or precondition, data setup, the inputs, expected outcomes and actual outcomes.
- Execute Tests - Execute the test cases against the system under test and document the results.
- Verify Test Results - Verify if the expected and actual results match each other.
- Verify Test Coverage - Verify if the tests cover both functional and non-functional aspects of the requirement.
- Track and Manage Defects - Any defects detected during the testing process goes through the defect life cycle and are tracked to resolution. Defect Statistics are maintained which will give us the overall status of the project.

#### **Positive Testing:**

- Positive testing is a testing technique to show that a product or application under test does what it is supposed to do.
- Positive testing verifies how the application behaves for the positive set of data.
- Positive Testing verifies if the application is not showing error when it is not supposed to and showing error when it is supposed to.
- Positive testing is a sort of testing that is conducted on a software program using legitimate data sets as input.
- It determines whether or whether the software program acts as predicted when given favourable input.
- Positive testing is done to ensure that the software program accomplishes precisely what it is supposed to accomplish.

### Negative Testing:

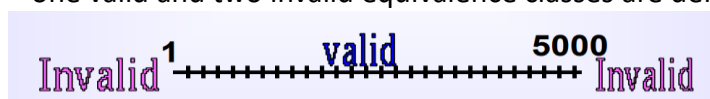
- Negative testing is a way of evaluating a software program that uses faulty or incorrect sets of data as input.
- It determines if the software program responds as predicted to negative or inappropriate user input.
- Negative testing ensures that the software program does not collapse and stays stable in the presence of incorrect data inputs.
- Negative Testing is performed to expose the software weakness and potential for exploitation.
- It is carried out to show data corruption or security breaches.
- Negative Testing is carried out to spot the faults that can result in significant failures.

The following factors must be considered in both tests –

- Input data
- An activity that must be carried out
- Output Result

### Equivalence Partitioning Method:

- Equivalence Partitioning Method is also known as Equivalence class partitioning (ECP)
- Equivalence partitioning is a software testing technique that divides the input and/or output data of a software unit into partitions of data from which test cases can be derived.
- It is a software testing technique of black-box testing
- The equivalence partitions are usually derived from the requirements specification for input
- Test cases are designed to cover each partition at least once.
- If member of set is given as an input condition, then one valid and one invalid equivalence class is defined.
- If Boolean no. is given as an input condition, then one valid and one invalid equivalence class is defined.
- If the range condition/specific value is given as an input, then one valid and two invalid equivalence classes are defined.
- Example:  
the input data to the program is specified by a range of values:
  - e.g. numbers between 1 to 5000.
  - one valid and two invalid equivalence classes are defined.



There are three equivalence classes:

- the set of negative integers
- set of integers in the range of 1 and 5000
- integers larger than 5000.

**Example-1:**

Let us consider an example of any college admission process. There is a college that gives admissions to students based upon their percentage.

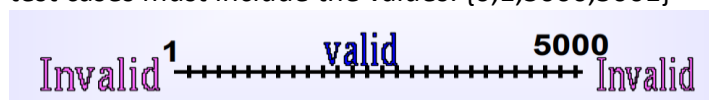
Consider percentage field that will accept percentage only between 50 to 90 %, more and even less than not be accepted, and application will redirect user to an error page. If percentage entered by user is less than 50 % or more than 90 %, that equivalence partitioning method will show an invalid percentage. If percentage entered is between 50 to 90 %, then equivalence partitioning method will show valid percentage.

Percentage  \* Accepts Percentage value between 50 to 90

| Equivalence Partitioning |       |           |
|--------------------------|-------|-----------|
| Invalid                  | Valid | Invalid   |
| $\leq 50$                | 50-90 | $\geq 90$ |

**Boundary Value Analysis:**

- Boundary Value Analysis is based on testing the boundary values of valid and invalid partitions.
- Boundary value is an input or output value on the border of an equivalence partition, includes minimum and maximum values at inside and outside the boundaries.
- The behaviour at the edge of the equivalence partition is more likely to be incorrect than the behaviour within the partition.
- Boundaries are an area where testing is likely to yield defects.
- BVA is a test case design to test boundary value between partitions (both valid partition and invalid partition).
- A boundary value for a valid partition is a valid boundary value.
- A boundary value for an invalid partition is an invalid boundary value.
- For each variable we check-
  - Minimum value.
  - Just above the minimum.
  - Nominal Value.
  - Just below Max value.
  - Max value.
- For a function that computes the square root of an integer in the range of 1 and 5000: – test cases must include the values: {0,1,5000,5001}

**Decision table:**

Decision table technique is one of the widely used case design techniques for black box testing.

This is a systematic approach where various input combinations and their respective system behaviour are captured in a tabular form.

It is also known as a cause-effect table.

Now, let's see how a decision table is created for the login function in which we can log in by using email and password. Both the email and the password are the conditions, and the expected result is action.

|                                 |              |                    |                 |                 |
|---------------------------------|--------------|--------------------|-----------------|-----------------|
| <b>Email (condition1)</b>       | T            | T                  | F               | F               |
| <b>Password (condition2)</b>    | T            | F                  | T               | F               |
| <b>Expected Result (Action)</b> | Account Page | Incorrect password | Incorrect email | Incorrect email |

#### Advantages

- When the system behaviour is different for different input and not same for a range of inputs both equivalence partitioning and boundary value analysis won't help, but decision table can be used.
- The representation is simple so that it can be easily interpreted and is used for development and business as well.
- This table will help to make effective combinations and can ensure a better coverage for testing
- Any complex business conditions can be easily turned into decision tables
- In a case we are going for 100% coverage typically when the input combinations are low, this technique can ensure the coverage.

#### Disadvantage:

- When the number of inputs will increase the table will become more complex

|                  |                      |                |
|------------------|----------------------|----------------|
|                  | <b>Stubs</b>         | <b>Entries</b> |
| <b>Condition</b> | c1<br>c2<br>c3       |                |
| <b>Action</b>    | a1<br>a2<br>a3<br>a4 |                |

- Condition Stubs : The conditions are listed in this first upper left part of the decision table that is used to determine a particular action or set of actions.
- Action Stubs : All the possible actions are given in the first lower left portion (i.e., below condition stub) of the decision table.

- Condition Entries : In the condition entry, the values are inputted in the upper right portion of the decision table. In the condition entries part of the table, there are multiple rows and columns which are known as Rule.
- Action Entries : In the action entry, every entry has some associated action or set of actions in the lower right portion of the decision table and these values are called outputs.

### **Grey box Testing:**

- Gray Box Testing is a software testing technique which is a combination of Black Box Testing technique and White Box Testing technique.
- In Black Box Testing technique, tester is unknown to the internal structure of the item being tested and in White Box Testing the internal structure is known to tester.
- The internal structure is partially known in Gray Box Testing.
- It involves access to internal coding(internal data structures and algorithm) to design test cases as white box testing and testing practices are done at functionality level as black box testing.
- Gray Box Testing is named so because the software program is like a semi-transparent or grey box inside which tester can partially see.
- It gives the ability to test both presentation layer as well as internal coding structure.
- It commonly focuses on context-specific errors related to web systems.

### **Why to use?**

- It provides combined benefits of both black box testing and white box testing both
- It combines the input of developers as well as testers and improves overall product quality
- It reduces the overhead of long process of testing functional and non-functional types
- It gives enough free time for a developer to fix defects
- Testing is done from the user point of view rather than a designer point of view

### **Advantages of Grey Box Testing:**

- Users and developers have clear goals while doing testing.
- Gray box testing is mostly done by the user perspective.
- Testers are not required to have high programming skills for this testing.
- Gray box testing is non-intrusive.
- Overall quality of the product is improved.
- In grey box testing, developers have more time for defect fixing.
- By doing grey box testing, benefits of both black box and white box testing is obtained.
- Gray box testing is unbiased. It avoids conflicts between a tester and a developer.
- Gray box testing is much more effective in integration testing.

### **Disadvantages of grey box testing:**

- Defect association is difficult when grey testing is performed for distributed systems.
- Limited access to internal structure leads to limited access for code path traversal.
- Because source code cannot be accessed, doing complete white box testing is not possible.
- Gray box testing is not suitable for algorithm testing.
- Most of the test cases are difficult to design



### Techniques of grey box testing:

#### 1. Matrix technique:

- This testing technique comes under Grey Box testing.
- Matrix technique is a method to remove unused and uninitialized variables by identifying used variables from the program.
- This testing technique involves defining all the variables that exist in their programs.
- It defines all the used variables of a particular program.
- In any program, variable are the elements through which values can travel inside the program.
- It should be as per requirement otherwise, it will reduce the readability of the program and speed of the software.

#### 2. Regression Testing:

- Regression testing is used to verify that modification in any part of software has not caused any adverse or unintended side effect in any other part of the software.
- During confirmation testing, any defect got fixed, and that part of software started working as intended, but there might be a possibility that fixed defect may have introduced a different defect somewhere else in the software.
- So, regression testing takes care of these type of defects by testing strategies like retest risky use cases, retest within a firewall, retest all, etc.

#### 3. Orthogonal Array Testing or OAT:

- The purpose of this testing is to cover maximum code with minimum test cases.
- Test cases are designed in a way that can cover maximum code as well as GUI functions with a smaller number of test cases.

#### 4. Pattern testing:

- This testing is performed on the historical data of the previous system defects.
- Unlike black box testing, grey box testing digs within the code and determines why the failure happened
- To perform the testing, previous defects are analyzed.
- It determines the cause of the failure by looking into the code.



| Black Box Testing                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         | Gray Box Testing                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| It is a software testing technique in which the tester doesn't know the internal structure of the application being tested.                                                                                                                                                                                                                                                                                                                                                                                                               | It is a software testing technique in which the tester partially know the internal structure of the application being tested.                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <p>It is known as closed box testing.</p> <p>No knowledge of implementation is required.</p> <p>It is based on external expectations and outer behaviour of the software.</p> <p>It is less time consuming.</p> <p>It is done by trial and error method.</p> <p>It improves some of the qualities of the software.</p> <p>The problems or suggestions gathered during Black Box testing will be incorporated in the future iterations of the product.</p> <p>Black box testing provides endurance and security against viral attacks.</p> | <p>It is known as translucent testing.</p> <p>Knowledge of implementation is required but need not to be expert.</p> <p>It is based on database diagrams and data flow diagrams.</p> <p>It is time consuming but not too much.</p> <p>It is done on the basis of data domains.</p> <p>It improves the overall quality of the software.</p> <p>The addressing of the problems found or bugs during the gray box testing process is carried out in the same release.</p> <p>White box testing does not provide endurance and security against viral attacks.</p> |
| It's not as exhaustive as Gray box testing.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               | It is partially exhaustive depending on the type of test cases- coding based or GUI based.                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <p><b>Black box test design techniques-</b></p> <ul style="list-style-type: none"> <li>• Decision table testing</li> <li>• All-pairs testing</li> <li>• Equivalence partitioning</li> </ul>                                                                                                                                                                                                                                                                                                                                               | <p><b>Gray box test design techniques-</b></p> <ul style="list-style-type: none"> <li>• Matrix testing</li> <li>• Regression testing</li> <li>• Pattern testing</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                     |

| S. No. | White Box Testing                                                                                                                                                                                               | Gray Box Testing                                                                                                                                                   |
|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1.     | It is a type of software testing in which the internal structure and design of the software application is fully known to the tester.                                                                           | It is a type of software testing in which the internal structure and design of the software application is partially known to the tester.                          |
| 2.     | It is also known as clear box testing or transparent testing.                                                                                                                                                   | It is known as translucent testing.                                                                                                                                |
| 3.     | It is performed by testers and developers.                                                                                                                                                                      | It is performed by end users, testers and developers.                                                                                                              |
| 4.     | Full knowledge of the implementation is required.                                                                                                                                                               | Small knowledge of the implementation is enough.                                                                                                                   |
| 5.     | High programming skills are required to perform white box testing.                                                                                                                                              | basic programming skills are enough to perform this testing.                                                                                                       |
| 6.     | It is a time consuming testing.                                                                                                                                                                                 | It is a less time consuming testing.                                                                                                                               |
| 7.     | It is used for algorithm testing.                                                                                                                                                                               | It is not used in algorithm testing.                                                                                                                               |
| 8.     | As application's internal working is known to the testers so they prepare test data accordingly and attempt to test nearly every code scenario using these test cases.                                          | High-level data flow diagrams and database flow diagrams serve as the foundation for this testing because they provide the basic required functionality knowledge. |
| 9.     | As major part of this type of testing depends on the internal implemented code so both possibilities exist here- one is automation of this testing and other one is development of algorithms for this testing. | There is less possibility of automation in gray box testing as it does not have access to the internally implemented code.                                         |
| 10.    | It takes more time to design test cases due to lengthy code.                                                                                                                                                    | It takes less time in designing test cases.                                                                                                                        |
| 11.    | The coding form the basis for internal working in this testing.                                                                                                                                                 | High-level database diagrams and dataflow diagrams form the basis for testing.                                                                                     |

| Black Box Testing                                                                                                                   | White Box Testing                                                                                                                              |
|-------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------|
| It is a way of software testing in which the internal structure or the program or the code is hidden and nothing is known about it. | It is a way of testing the software in which the tester has knowledge about the internal structure or the code or the program of the software. |
| Implementation of code is not needed for black box testing.                                                                         | Code implementation is necessary for white box testing.                                                                                        |
| It is mostly done by software testers.                                                                                              | It is mostly done by software developers.                                                                                                      |
| No knowledge of implementation is needed.                                                                                           | Knowledge of implementation is required.                                                                                                       |
| It can be referred to as outer or external software testing.                                                                        | It is the inner or the internal software testing.                                                                                              |
| It is a functional test of the software.                                                                                            | It is a structural test of the software.                                                                                                       |
| This testing can be initiated based on the requirement specifications document.                                                     | This type of testing of software is started after a detail design document.                                                                    |
| No knowledge of programming is required.                                                                                            | It is mandatory to have knowledge of programming.                                                                                              |
| It is the behavior testing of the software.                                                                                         | It is the logic testing of the software.                                                                                                       |
| It is applicable to the higher levels of testing of software.                                                                       | It is generally applicable to the lower levels of software testing.                                                                            |
| It is also called closed testing.                                                                                                   | It is also called as clear box testing.                                                                                                        |
| It is least time consuming.                                                                                                         | It is most time consuming.                                                                                                                     |
| It is not suitable or preferred for algorithm testing.                                                                              | It is suitable for algorithm testing.                                                                                                          |
| Can be done by trial and error ways and methods.                                                                                    | Data domains along with inner or internal boundaries can be better tested.                                                                     |

| Black Box Testing                                                                                                                                                                                                                                                                         | White Box Testing                                                                                                                                                                                                                         |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p><b>Example:</b> Search something on google by using keywords</p> <p><b>Black-box test design techniques-</b></p> <ul style="list-style-type: none"> <li>• Decision table testing</li> <li>• All-pairs testing</li> <li>• Equivalence partitioning</li> <li>• Error guessing</li> </ul> | <p><b>Example:</b> By input to check and verify loops</p> <p><b>White-box test design techniques-</b></p> <ul style="list-style-type: none"> <li>• Control flow testing</li> <li>• Data flow testing</li> <li>• Branch testing</li> </ul> |
| <p><b>Types of Black Box Testing:</b></p> <ul style="list-style-type: none"> <li>• Functional Testing</li> <li>• Non-functional testing</li> <li>• Regression Testing</li> </ul>                                                                                                          | <p><b>Types of White Box Testing:</b></p> <ul style="list-style-type: none"> <li>• Path Testing</li> <li>• Loop Testing</li> <li>• Condition testing</li> </ul>                                                                           |
| It is less exhaustive as compared to white box testing.                                                                                                                                                                                                                                   | It is comparatively more exhaustive than black box testing.                                                                                                                                                                               |