Regression Testing

Dr. Durga Prasad Mohapatra
Professor
CSE Department
NIT Rourkela

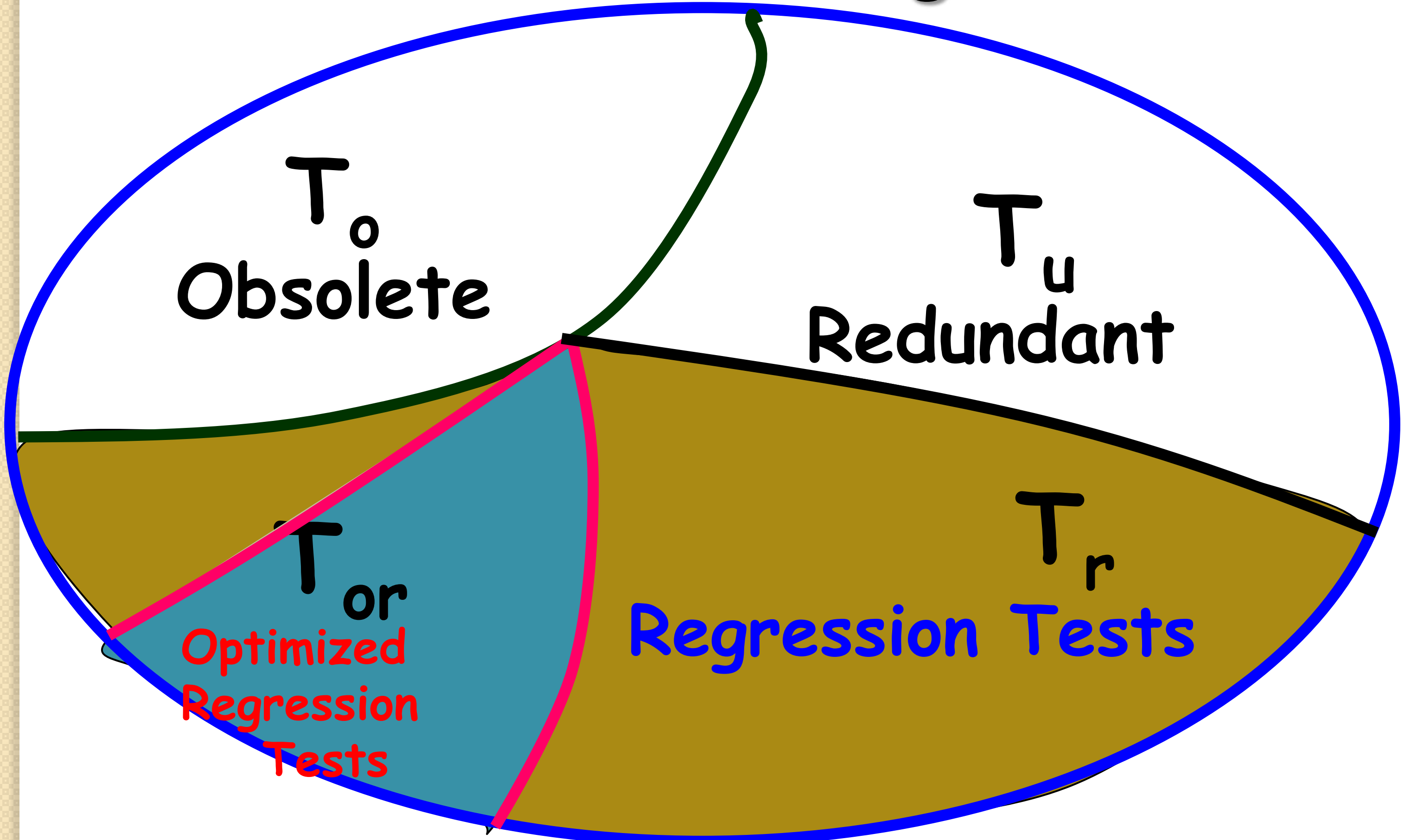
What is regression testing?

Regression testing is testing done to check that a system update does not cause new errors or re-introduce errors that have been corrected earlier.

Need for Regression Testing

- Any system during use undergoes frequent code changes.
  - Corrective, Adaptive, and Perfective changes.
- Regression testing needed after every change:
  - Ensures unchanged features continue to work fine.

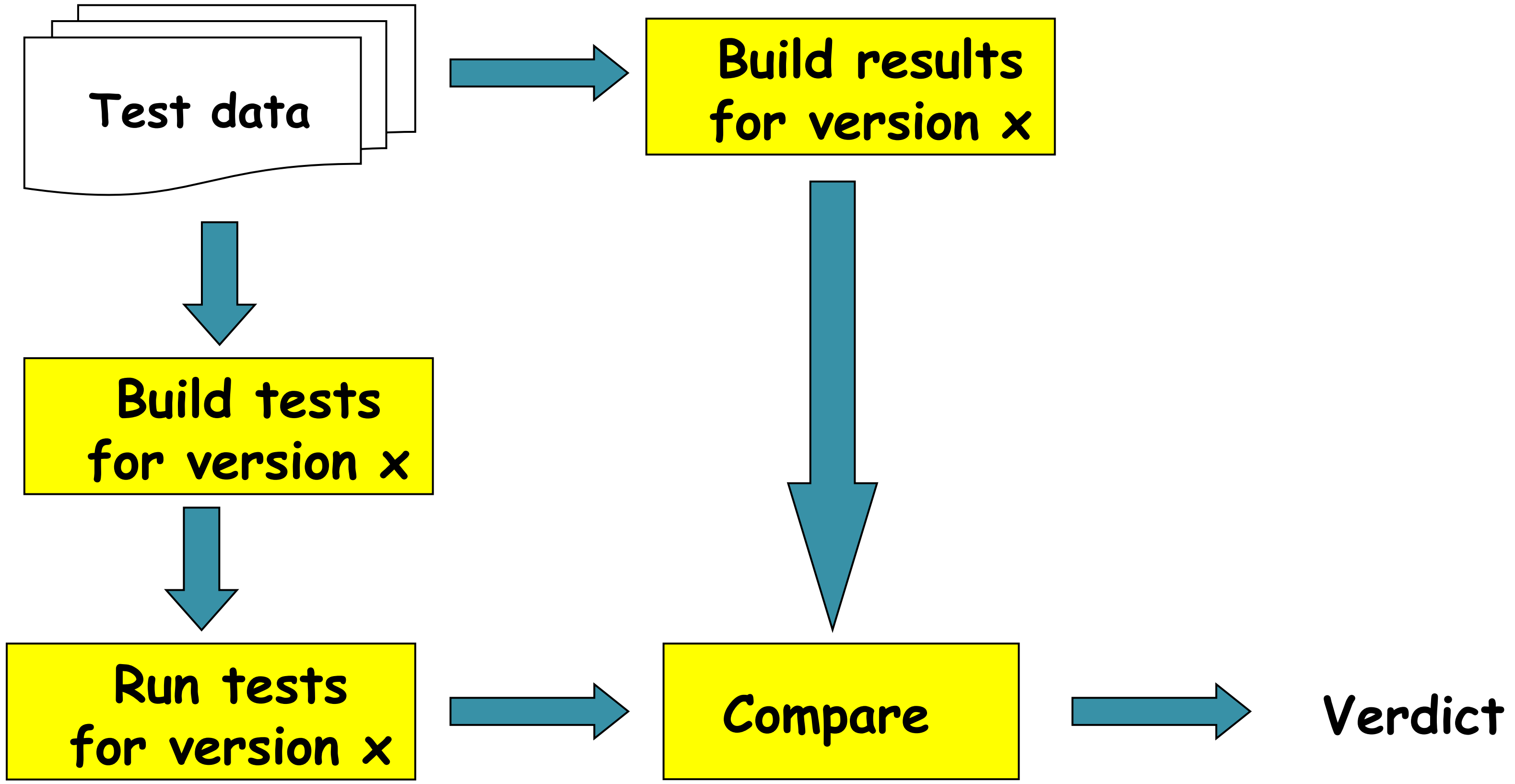
Partitions of an Existing Test Suite



Major Regression Testing Tasks

- Test revalidation (RTV):
  - Check which tests remain valid
- Test selection (RTS):
  - Identify tests that execute modified portions.
- Test minimization (RTM):
  - Remove redundant tests.
- Test prioritization (RTP):
  - Prioritize tests based on certain criteria.

Automating regression testing



Regression Testing

- Does not belong to either unit test, integration test, or system test.
  - In stead, it is a separate dimension to these three forms of testing.
- Regression testing (RT)is the running of test suite:
  - after each change to the system or after each bug fix
  - ensures that no new bug has been introduced due to the change or the bug fix.

- Regression tests assure:
  - the new system's performance is at least as good as the old system
  - always used during phased system development.

Regression Testing

- Regression testing is any type of software testing that seeks to uncover new errors, or regressions, in the existing functionality after changes have been made to the software, such as functional enhancements, patches or configuration changes.
- It can also be defined as the software maintenance task performed to instill confidence that changes are correct and have not adversely affected the unchanged portions of the program.

Progressive vs. Regressive Testing

- All different types of testings like Unit Testing, Integration Testing are Progressive testing or Developmental testing.
- From verification to validation, the testing process progresses towards release of the product.
- But, the intent of Regressive testing is to assure that a change, such as a bug fix, did not introduce new bugs. One of the main reasons for regression testing is that it's often extremely difficult for a programmer to figure out how a change in one part of the software will echo in other parts of the software.

Regression testing cont…

- A system under test (SUT) is said to regress if,
  - a modified component fails, or
  - a new component when used with unchanged component, causes failures in the unchanged component by generating side effects.
- Therefore the following versions will be there in the system:
  - Baseline Version: The version of a component that has passed a test suite.
  - Delta Version: a changed version that has not passed a regression test.
  - Delta Build: an executable configuration of the SUT that contains all the Delta & Baseline Components.

- So, we can say that, most test cases begin as progressive test cases and eventually become regression test cases.
- RT is not another testing activity; it is the re-execution of some or all of the already developed test cases.

Definition

- IEEE Definition:
  - *It is the selective retesting of a system or component to verify that the modifications have not caused unintended effects and that the system or component still complies with its specified requirements.*
- It can also be defined as
  - *The s/w maintenance task performed on a modified program to instill confidence that changes are correct and have not adversely affected the unchanged portions of the program.*

Importance of Regression Testing

- Regression testing is performed in case of bug fixing or whenever there is a need to incorporate a new requirement.
- Importance of Regression Testing is due to following Reasons:

➢ It validates the parts of software where changes occur.

➢ It validates the parts of the software which may be affected by the changes.

➢ It ensures proper functioning of software before changes occur.

➢ It enhances quality of software, as it reduces the risk & high risk bugs.

Regression Testability

- Regression Testability refers to the property of a program, modification, or test suite that lets it be effectively and efficiently regression-tested.
- It is a function of both the Design of the program and the Test Suite.
- Regression Number: Avg no. of affected test cases in the test suite that are affected by modification to a single instruction.
- If Regression Testability is done at early stages it reduces cost of development and maintenance of the Software.

Objectives of Regression Testing

- It tests to check that the bug has been addressed:
- It finds other related bugs
- It tests to check on the other parts in the program

When Regression Testing is Done?

- Corrective Maintenance: Changes made to the system after failure.
- Adaptive Maintenance: Changes made to achieve continuing compatibility with the target environment or the other system.
- Perfective Maintenance: Changes designed to improve/add capabilities.
- Rapid iterative Development: The extreme programming approach.
- First Step of Integration: Re-running accumulated test suite, as new components added.
- Compatibility Assessment & Benchmarking: Test suites for wide range of platforms and application.

Regression Testing Types

- Bug-Fix Regression: This testing is performed after bug has been reported & fixed. The goal is to repeat the test cases that expose the problem in the first place.
- Side-Effect regression/Stability Regression: It involves retesting substantial part of product. The goal is to prove that changes have no detrimental effect on earlier program. It tests the overall integrity of the program, not the success of software fixes.

Defining Regression Testing Problem

Let,

P :denotes a program or procedure.

P' :denotes modified version of P.

S :denotes the specification for program P.

S' : denotes the specification for program P'.

P(i):refer to output of P on input i,

P'(i):refer to output of P' on input i.

T={t1,t2…...tn}:denotes a set of test suite

Regression Testing: Problem Definition

- Given a program P, its modified version P', and a test set T that was used earlier to test P, find a way to utilize T to gain sufficient confidence in the correctness of P'.

Is Regression Testing a Problem?

The following difficulties occur in Regression Testing:

- Large system can take a long time to retest.
- It can be difficult & time consuming to create the tests.
- It can be difficult & time consuming to evaluate the tests.
- Cost of testing can reduce the resources available for maintenance.

Regression Testing Techniques

- **Regression Test Selection** : Reduce the time required to retest by selecting smaller Subset of test suite.
- **Test case Prioritization Technique**: Prioritize the test cases then execute according to priority. There are two types of Prioritization:
  - General Test case Prioritization – no knowledge of the modification.
  - Version-Specific Test case Prioritization - knowledge of the modification is known.
- **Test suite Reduction Technique** : reduce testing costs by eliminating redundant test cases.

Selective Retest Technique

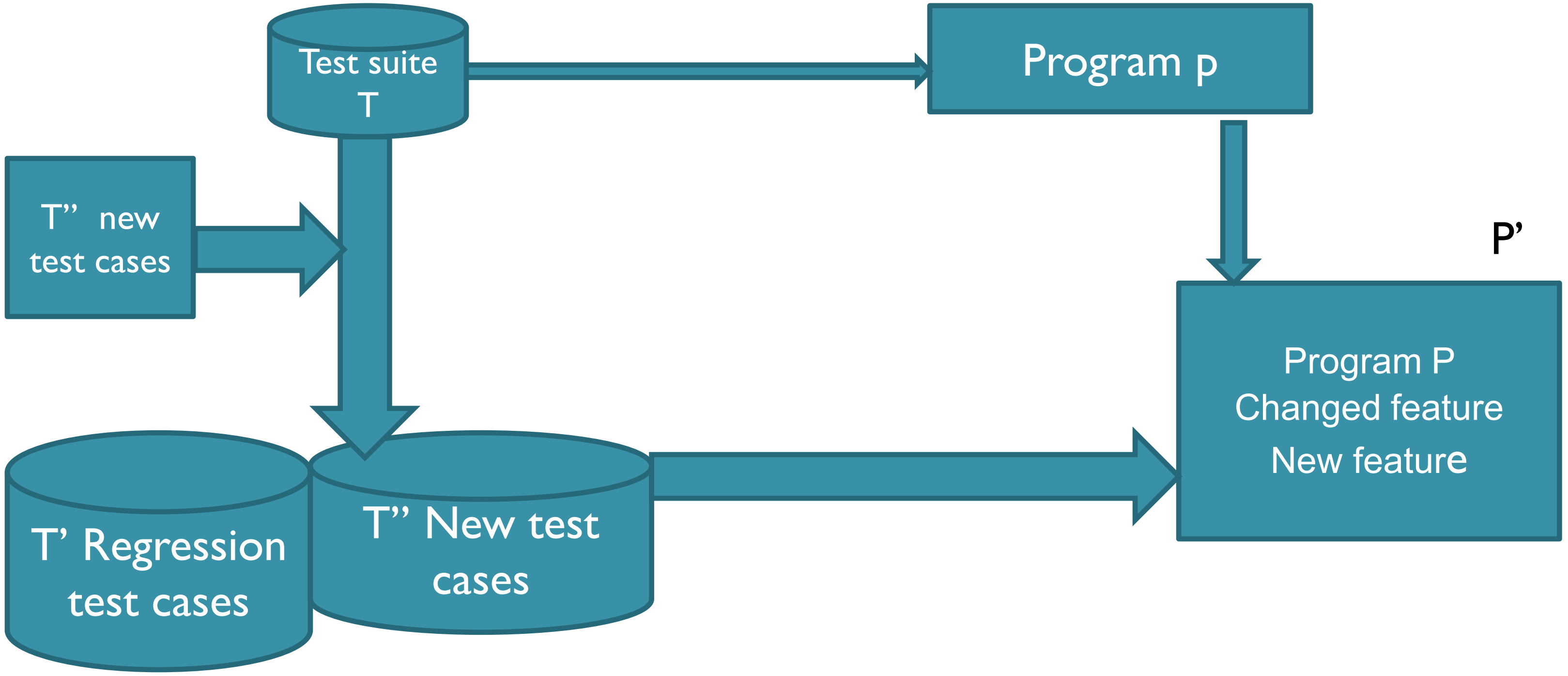
- Software maintenance includes more than 60% of development costs.
- In that case, testing costs dominate because many systems require labour-intensive manual testing. Selective retest techniques attempt to reduce the cost of testing by identifying the portions of *P’* that must be exercised by the regression test suite.
- Selective retesting is distinctly different from a retest-all approach that always executes every test in an existing *regression test suite* (RTS).
- The objective of Selective Retest Technique is *cost reduction*. It is the process of selecting a subset of the regression test suite that tests the changes.

Selective Retest Technique cont…

It has following characteristic features:

- It minimizes the resources required to regression test a new version.
- It minimizes no. of test cases.
- It is needed to remove obsolete, uncontrollable & redundant test cases.
- It analyses the relationship between test cases & software elements they cover.
- It uses the information about changes to select the test cases.

Selective Retest Technique cont…



Selective Retest Technique cont…

1. Select T' subset of T, a set of test cases to execute on P'.
2. Test P' with T', establishing correctness of P' with respect to T'.
3. If necessary create T'', a set of new functional or structural test cases for p'.
4. Test P' with T'', establishing correctness of P' with respect to T''.
5. Create T''', a new test suite and test execution profile for P', from T, T' and T''.

Each of these steps involves the following important problems:

Selective Retest Technique cont…

- **Regressive test selection problem**: The problem is to select a subset of T' of T with which P' will be tested.
- **Coverage test selection problem**: Specification that requires additional testing.
- **Test suite execution problem**: Execute test case efficiently and checking test results for correctness.
- **Test maintenance problem**: Update & store test information.

Strategies for Test case Selection

A procedure for selecting the test case is provided by a test criterion . The criteria are as follows:

Selection criteria based on code

- Motivation: Potential failures can only be detected if the parts of the code that can cause faults are executed.
- All the code based regression test selection techniques attempt to select a subset T' of T that will be helpful in establishing confidence that P's functionality has been preserved where required.
- All code-based test selection techniques are concerned with locating tests in T that expose faults in P'.
- The following tests are based on these criteria:

Selection criteria based on code cont…

- **Fault-revealing test cases**: A test case t detects a fault in P' if it causes P' to fail. Hence t is faulting revealing for P'.
  - There is no effective procedure to find the tests in T that are fault revealing for P'.
  - Under certain conditions, however, a regression test selection technique can select a superset of the tests in T that are fault-revealing for P'.
  - Under these conditions, such a technique omits no tests in T that can reveal faults in P'.

Selection criteria based on code cont…

- **Modification-revealing test cases**:
  - A test case t is modification revealing for P and P’ if and only if it causes the outputs of P and P’ to differ.
- **Modification-traversing test cases**:
  - A test case t is modification traversing if and only if it executes new or modified code in p’.
  - These tests are useful because a non-obsolete test t in T can only be modification-revealing for P and P’ if it is modification traversing for P and P’.

Regression Test Selection Techniques

- Minimization Techniques:
  - attempt to select minimal sets of test cases from T that yield coverage/affected portion of P.
- Dataflow Techniques:
  - It selects test cases that exercise data interaction that have been affected by modification.
  - Every definition-use pair that is deleted from P, new in P’, or modified for P’, should be tested.
  - The technique selects every test case in T that, when executed on P, exercised, deleted, or modified definition-use pairs, or executed a statement containing a modified predicate.

Continued…

- Safe Techniques:
  - An explicit set of safety condition needs to be satisfied here.
  - When an explicit set of safety conditions can be satisfied, safe regression test selection techniques guarantee that the selected subset T’ contains all the test cases in the original test suite T that can reveal faults in P’.

Continued…

- Ad doc/Random Techniques:
  - When time constraints prohibit the use of all test cases, select test cases based on ‘Intuitions’/Randomly select predetermined no. of test cases.
  - Another simple approach is to randomly select a predetermined number of test cases from T.
- Retest-all Technique:
  - The retest-all technique simply reuses all the existing test cases.
  - To test P’, the technique effectively selects all test cases in T.

Evaluating Regression Test Selection Techniques

- Rothermel et al. have recognized the following categories for evaluating Regression test selection techniques:
- Inclusiveness:
  - It measures the extent to which M chooses modification-revealing test from T for inclusion in T'. [M: Regression Test selection Technique]
  - Suppose T contains n tests that are modification revealing for P and P', suppose M selects m of these tests. The inclusiveness of M relative to P and P' and T is:
- 1.INCL(M) = (100 * (m / n) %, if n != 0
- 2. INCL(M) = 100%, if n = 0
- (here T contains n modification-revealing tests & M contains m modification-revealing tests .)

Example

- For example, if *T* contains 100 tests of which 20 are modification-revealing for *P* and *P'*, and *M* selects 4 of these 20 tests, then *M* is 20% inclusive relative to *P*, *P'*, and *T*.
- If *T* contains no modification-revealing tests, then every test selection technique is 100% inclusive relative to *P*, *P'*, and *T*.
- If for all *P*, *P'*, and *T*, *M* is 100% inclusive relative to *P*, *P'*, and *T*, then *M* is safe.

Evaluating Regression Test Selection Techniques cont…

- **Precision:** It measures the extent to which M omits tests that are non-modification-revealing.
  - Suppose T contains n tests that are non-modification revealing for P and P', and suppose M omits m of these tests.
  - The precision of M relative to P, P', and T is given by

    1. Precision = 100 * (m / n)%, if n != 0
    2. Precision = 100%, if n = 0

Example

- For example, if *T* contains 50 tests of which 22 are non-modification-revealing for *P* and *P*', and *M* omits 11 of these 22 tests, then *M* is 50% precise relative to *P*, *P*', and *T*.
- If *T* contains no non-modification-revealing tests, then every test selection technique is 100% precise relative to *P*, *P*', and *T*.

Evaluating Regression Test Selection Techniques cont …

- Efficiency: Measured in terms of their Space & Time Requirements.
  - Where time is concerned, test selection technique is more economical than the reset all technique, if the cost of selecting T’ is less than the cost of running the test in T- T’.
  - Space efficiency depends on the test history and program analysis information a technique must store.

Steps for Regression Test Case Prioritization

1. Select T' subset of T, a set of test cases to execute on P'.
2. Produce T'_p , a permutation of T', such that T'_p will have a better rate of fault detection than T'.
3. Test P' with T'_p, establishing correctness of P' with respect to T'.
4. If necessary create T'', a set of new functional or structural test cases for p'.
5. Test P' with T'', establishing correctness of P' with respect to T''.
6. Create T''', a new test suite and test execution profile for P', from T, T'_p and T'',

   (T'_p: Contains Execution ordering of regression tests)

Steps for Regression Test Case Prioritization

- Step 1 allows the tester to select T' such that it is proper subset of T or that it actually contains every test that T contains.
- Step 2 could produce T'_p so that it contains an execution ordering for the regression test that is the same or different than the order provided by T'.
- The intension behind prioritizing the execution of a regression test suite is to ensure that the defects normally introduced by the competent programmers are discovered earlier in the testing process.

Summary

- Regression testing is the selective retesting of a System or Component to verify that Modifications have not caused unintended effects and that the System or Component still complies with its Specified Requirements.
- It has positive Influence on software quality.
- Regression number is the average number of affected test cases in the test suite that are affected by any modification to a single instruction.
- It is of two types: bug-fix regression and side-effect regression.

References

1. Rajib Mall, Fundamentals of Software Engineering, (Chapter – 10), Fifth Edition, PHI Learning Pvt. Ltd., 2018.
2. Naresh Chauhan, Software Testing: Principles and Practices, (Chapter – 8), Second Edition, Oxford University Press, 2016.

Thank You