Date.....

Software Testing (KCS076)

Unit-1 :- Review of Software Engineering : Overview of software Evolution, SDLC, Testing Process, Terminologies in Testing : Error, Fault, Failure, Verification, Validation, Difference Between Verification and Validation, Test Cases, Testing Suite, Test, Oracles, Impracticality of testing All Paths. Verification : Verification Methods, SRS Verification, Source code Reviews, User Documentation Verification, Software Project Audit, Tailoring Software Quality Assurance Program by Review, Walkthrough, Inspection and Configuration Audits.

Unit-2 :- Functional Testing : Boundary Value Analysis, Equivalence class Testing, Decision Table Based Testing, Cause Effect Graphing Technique. Structural Testing : Control Flow Testing, Path Testing, Independent Paths, Generation of Graph from Program, Identification of Independent Paths, Cyclomatic Complexity, Data Flow Testing, Mutation Testing.

Unit-3 :- Regression Testing : What is Regression Testing? Regression Test cases selection, Reducing the number of test cases, Code coverage prioritization technique, Reducing the number of test cases : Prioritization guidelines, Priority category, Scheme, Risk Analysis.

Spiral

Teacher's Sign.....

Date.....

Unit-4:- Software Testing Activities: Levels of Testing, Debugging, Testing techniques and their applicability, Exploratory Testing Automated Test data Generation: Test Data, Approaches to test data generation, test data generation using genetic algorithm, Test data Generation Tools, Software Testing Tools, and Software Test Plan.

Unit-5:-

Object Oriented Testing: Definition, Issues, Class Testing, Object Oriented Integration and System Testing. Testing Web Applications: Web Testing, User Interface Testing, Usability Testing, Security Testing, Database testing, Post Deployment Testing.

Date: 13/08/18 Day:- Monday

SOFTWARE TESTING & AUDIT

SOFTWARE EVOLUTION :-

It refers to the process of developing software initially then repeat-edly updating it for various reasons-

The aim of s/w evolution would be to implement & re-validate the possible major changes to the system without being able to predict how user requirements will evolve. The existing system is never complete & continue to evolve. The main objectives of S.E. are ensuring the reliability & flexibility of the system. It was recently found that a system should be evolved once every few months to ensure that it is adapted to the real world environment.

SOFTWARE DEVELOPMENT LIFE CYCLE (S.D.L.C)

S.D.L.C is used to

describe a process for planning, creating, testing & deploying an information system.

Phase-1 Planning & Requirement Analysis.

Each model starts with the analysis in which the stack holders of the process discuss the requirement of the final product. The goal is to obtain the detailed information of the proposed system.

Phase-2 Designing Project Architecture

All the technical questions that may appear on this stage are discussed by all the stack-holders including - customers, technology used, team-lead, time-frames, repitions, budget, appropriate decisions are made this stage.

Phase - 3. Development & Progr-amming

Programmer's start here with the source-code writing while keeping in mind previously defined requirement. The programming itself assumes 4 stages-

a) Algorithm development.
b) Source code writing.
c) Compilation.
d) Testing.
e) Debugging.

Phase-4. Testing

It includes the debugging process. All the code flaws missed during the development are detected there, documented & passed back to the developer to fix.

Phase-5 Deployment

When the program has finalized & has no critical issues, then its time to launch for end users.

collection of test-cases - Test-script
Date:- 14/08/18 selected test-case - Test suit Day:- Tuesday

Testing Process :-

Testing is a process rather than a single activity. The quality & effectiveness of s/w testing process depends on test-processes we use. It is the phase that follows coding & proceeds deployment.

The objective is to show in-correctness & testing is considered to succeed when an error is detected. Testing process consists of 5 steps

(i) Planning & Control.
(ii) Analysis & Design.
(iii) Implementation & Execution.
(iv) Evaluating exit criteria & Reporting.
(v) Test closure activity.

ERROR, fault, failure

Error :- A human-action that produces an in-correct result.

Fault :- A 'Flaw' in a component that can cause the component to fail to perform its require function.

Failure :- Deviation of a system from its expected delivery of servis or result.

Test-Cases- Set of steps & their expected result.

Date:- 16/08/18

Verification & Validation:-

Verification:-

It is a static practice of verify documentation, design, code & program.

It includes all the activities associated with producing high quality s/w-

(i) Inspection.
(ii) Walk through.
(iii) Reviews
(iv) Design Analysis & specific Analysis.

Validation:-

Dynamic mechanism of validating & testing the actual product.

Verification	Validation
1. Are we building the system right?	1. Are we building the right system?

2. Process of evaluating products of a development phase to find out whether they meet the specified requirements.	2. Process of evaluating s/w at the end of development phase to findout whether software meets the customer expectations & requirements.
3. Objective is to make sure that the product being developed is as per the requirement & design specification.	3. Objective is to make sure that the product actually meet up the user's requirement.
4. It is carried out by quality assurance team.	4. It is carried out by testing team & end users.
5. Execution of code doesn't comes.	5. Execution of code comes.
6. Explains whether output is according to input or not.	6. Explains whether s/w is accepted by the user or not.

Test-Case :-

1. Set of steps & expected results / conditions.
2. Inputs for writing test cases.
3. Test - case syntax -
   - Steps.
   - Expected Results.
   - Optional (test-case id, Test-data, Executed, Passed / Failed / Blocked, Notes).

Steps	Test-Case: Browse Product	Expected Result	status
1.	Login as a visitor	Login successful	P
2.	Select a Product Category	Show all the relevant products of product category	P
3.	Select Product	Show all the attributes of the product (Item code, Item price, Item name etc.)	P if tax is included, the status becomes Fails (F)

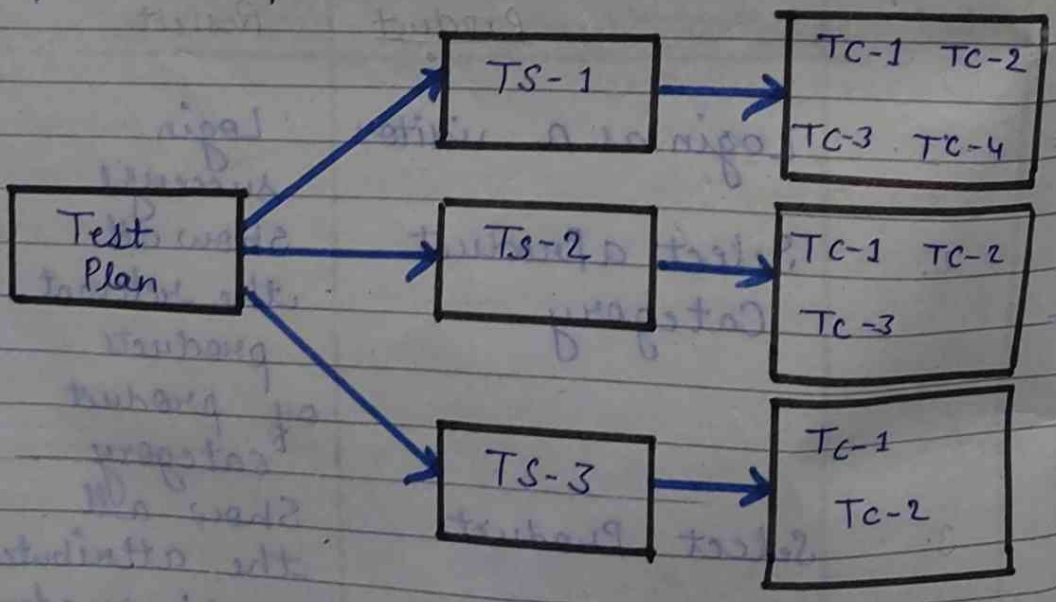
Date:- 18/08/18

Day : Saturday

Date :

Test- Suit :-

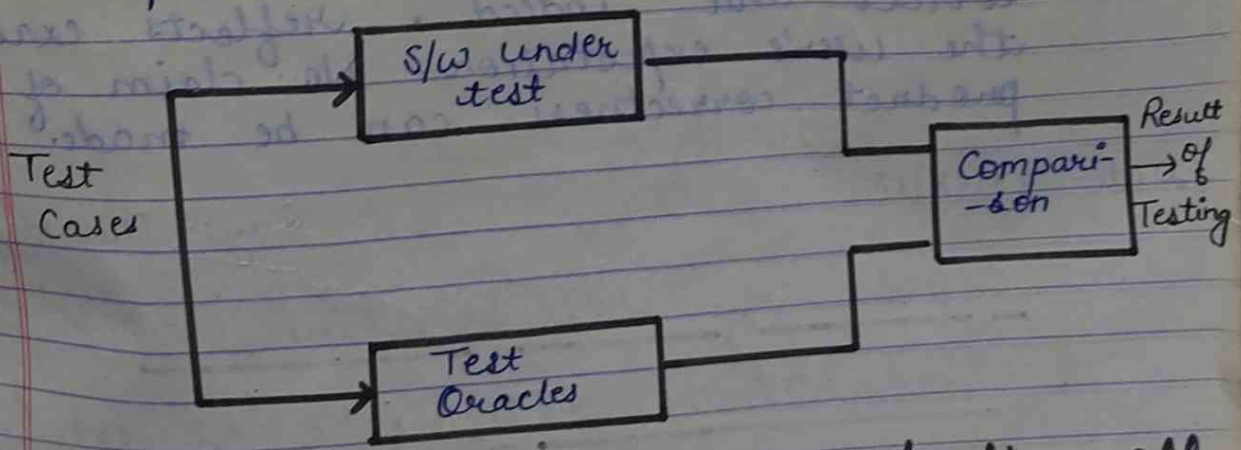
It is a collection of test-cases that are intended to the user. To test a software program to show that it have some specified set of behaviours. It is a container that has a set of test which help tester in executing & reporting the test-execution stages. A test-case can be adept to multiple test-suit & test-plan.



Date:- 20/08/18 Day:- Monday

→ Test-Oracles:-

To test any program we need to have a description of its expected behaviour & a method of determining whether the observed behaviour confirms to the expected behaviour. For this, we need test-oracle. It is a mechanism different from the program itself. It is a mechanism that determines whether s/w executed correctly for a test-case. We define a test-oracle to contain two essential parts that - Oracle Information that represent expected output and Oracle procedure that compares the Oracle information with the actual output.



Impracticality - (of testing all paths)

For most programs, it is impractical to attempt to test all execution path through the product due to the combinatorial explosion. It is also not possible to develop an algorithm for generating test data for paths in arbitrary product due to the inability to determine path feasibility.

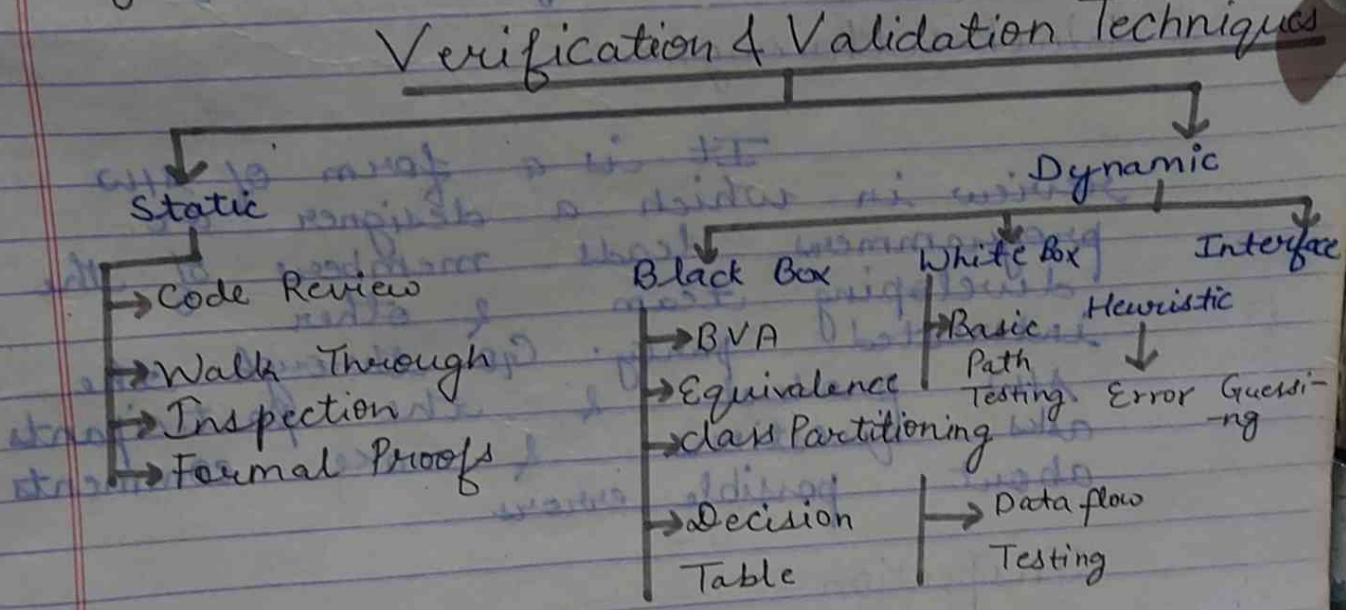
theoratical **Howden** Claims that there is no such thing as an absolute proof of correctness. Instead, he suggest that there are proves of equivalency that is proves that one description of a product is equivalent to another description. Hence, unless a formal specification can be shown to be correct and indeed, reflects exactly the user's expectation. No claim of product correctness can be made.

Date- 23/08/18 Day:- Thursday

Types of Verification & Validation Approaches with their objectives & limitation

Majority of s/w engineering practices attend to create & modify the s/w in a manner that measures the probability of satisfying its user expectations as a result several approaches for verification & validation evolve across the development cycle. Verification & Validation is divided into 2 parts -

1. Static methods - It involves the living process.
2. Dynamic methods :-



The overall objectives of s/w verification is to ensure that

the product is free from failure & meets the user expectation.

Management Review -

It is also k/a s/w quality assurance. It focuses on s/w process more rather than the s/w work product.

Quality Assurance is a set of activities assign to ensure that the project manager follows the standard process which is already pre-defined.

Walk-through -

It is a form of s/w review in which a designer or programmers leads members of the developing team & other interested party. Go through the s/w product & the participants asks questions & make comments about possible errors.

Inspection -

It is one of the most common sort of review

practices, found in S/W projects. The goal of inspection is to find defec-s commonly inspected work includ-s SRS & Test Plans.

Test cases in inspection process are -

1) Planning - The inspection is planned by the moderated.

2) Over view meeting - The moderator describe the background of the work produ-ction.

3) Preparation - Each Inspector examin the work-product to identify possible defects.

4) Inspection Meeting - During this meeting the reader reads throu-gh the work-product part-by-part & the inspector points out the defect for every part.

5) Rework - The author makes changes to the work product according to the action plan from the inspection meeting.

c) Follow up - The changes by the author are checked to make sure that everything is correct.

Configuration Audit :-

It is the formal examination of the as built configuration of a configuration item against its technical documentation to establish & verify the configuration items, products - base line. In configuration management a base line is an agreed description of a attributes of a product

→ Documentation Testing

It involves testing of the documented artifacts that are usually developed before or during the testing of s/w. Documentation for s/w testing helps in estimating the testing efforts required test-coverage, requirement-tracking etc. Some documents related to s/w development & testing are test-plan, test-cases, tracability method.

Software Project- Audit :-

Audit means an independent examination of a software product or processes to access compliance

with specifications standards contractual agreement or other criteria.

- A Physical Configuration Audit is a formal examination to verify the configuration item, product, baseli-ne.
- A s/w licensing audit where a user of a s/w is audited for license compliance.
- A s/w quality assurance where the s/w is audited for quality.

Date: 27/08/18 Day: Monday

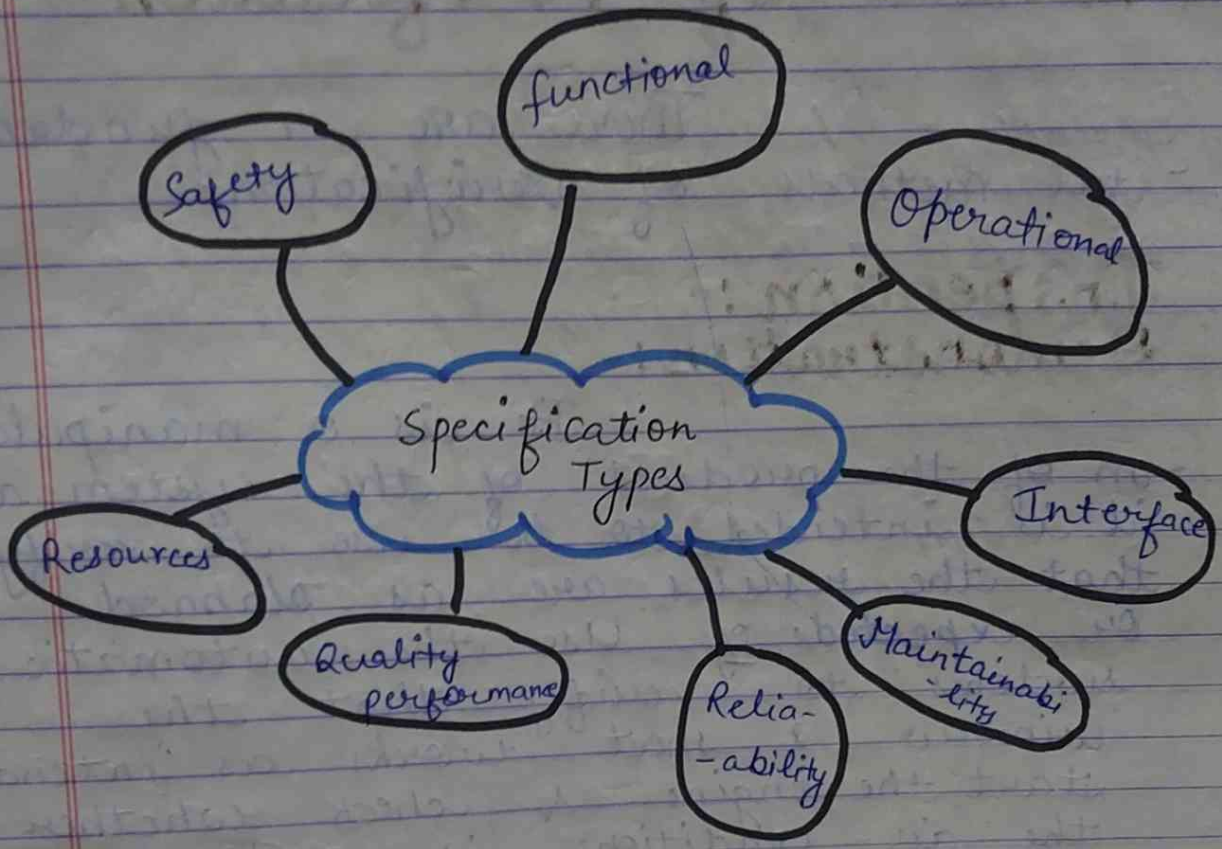
SRS :-

It is a document that captures the complete description about how a system is expected to perform. It is usually completed that the end of requirement engineering phase.

Quality of SRS -

Correct, Unambiguity, Easy to understand, complete, consistent, Verifiable, modifiable, tracable

Types of Requirements :-



Software Code- Review-

It is a systematic examination which can find & remove the draw-backs in the code such as memory leaks & buffer overflows. Technical Reviews are well documented & use a well defined defect detection process that includes peers & technical experts. It is ideally led by the trained moderator which is not the author.

Methods of Verification -

There are 4 fundamental methods of Verification:-

1. **Inspection :-**
2. **Demonstration :-**

It is a manipulation of the product of the system as it is intended to be use to verify that the results are as planned or expected. eg. Use the automatic switches to verify that the windows & sheet works as intended, start the engine & check whether the air-condition is working or not.

3. **Test :-**

It is the verification of the product or system using a control & pre-defined series of inputs, data to ensure that the product will produce a very specific & pre-defined output.
eg. Accelerate the car from a complete stop to 60 Km & verified at it can be done in 5.08 sec.

4. **Analysis :-**

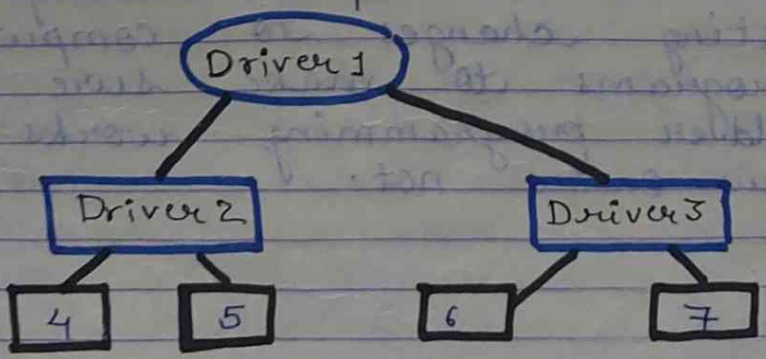
It is the verification of the product using models, calculation & testing equipment. It is often used

to predict the breaking point or failure of a system by using non-destructive test to extrapolate the failure-point. eg. Complete the series of test which reverse the engine while monitoring the vibrations & temperature to verify that the expected results of achieved

Date: 28/08/18 Day:- Tuesday

→ S.T.A. Driver :- (Test Driver)

Test Drivers are used ~~to~~ during bottom-up integration in order to simulate the behaviour of the upper level modules that are not yet integrated. These are the modules that act as a temporary replacement for a calling module & ~~its~~ gives the same output as the original



Order of Integration - 4,2
5,2
6,3
7,3
2,1
3,1

Project - Decision - Table based Testing

Test - Case Design Techniques -

1. Boundry - Value Analysis
2. Equivalence - Partitioning
3. Decision - Table Testing
4. Use - Case Testing
- Test - Coverage -

It is define as a technique which determines whether our test-cases are actually covering the application - code & how much code is exercised when we run those test-cases.

Regression Testing :-

It is the process of testing changes to computer programs to make sure that the older programming works with the new one or not.

Date:- 29/08/18

Day:- Wedn

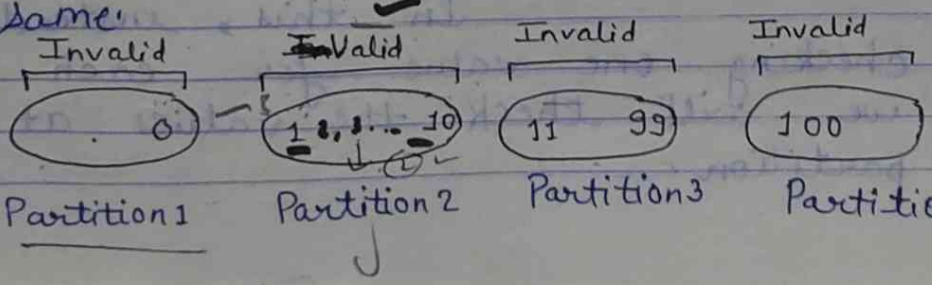
Unit-2

* Equivalence class Partition -ing-

It is a black-box testi
It divide the set of test conditi
-on into a partition that can be considered the same. It divides the input data of s/w into different equivallence data classes.

You can apply this technique where there is a range of input fields.

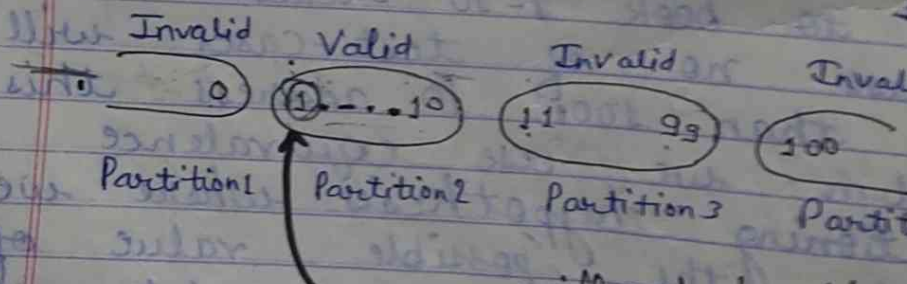
We can not test all the possible values in the given eg. of online ticket booking which allows only to book 1-10 tickets at a time because no. of test-cases will more than 100. To address this problem we use equivalence partitioning hypothesis where we divide the possible value of tickets into gaps or sets as shown below where the system behaviour can be consider as the same.



The divided sets are called Equivalence partitions then we pick 1 value from each partition for testing. Hypothisis behind this technique is that if one condition/ value passes or others will also pass.

* Boundry Value Analysis.

It is a process of testing b/w extreme ends or boundries b/w partitions of the input values. These extreme ends like start end, lower- upper, min max, just- inside, just- outside values are called Boundry values. In this we test boundries b/w equivalence partition.



you will check the boundry values like- 0, 1, 10, 11, 99, 100.

In this, instead of checking one value for each partition we will check the values at the partition.

It is also called 'Range Checkin
Both are used, & co-related, used
at all level of testing into a
manageable chunks. Appropriate for
calculation intensive application with
large no. of variable / inputs.

Decision Table Based Testi-ng:-

It is a testing technique used to test system behaviour for different input combination it is a systematic approach where the different input combination & their corresponding system behaviour are captured in a ~~tabular form~~ also k/a 'cost-effect table'.

Decision-table for login-screen-

Username : [ ]

Password : [ ]

[Login]

The condition is simple if the user provide correct user name & password then the user will be

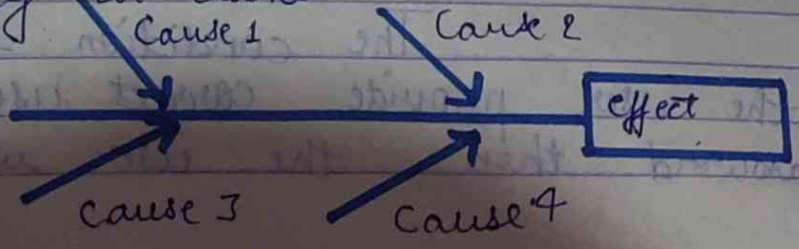
redirected to the home-page if any of the i/p is wrong then the error message will be displayed.

	Username	Pswd	O/P
Rule-1	T	T	H
Rule-2	T	F	E
Rule-3	F	T	E
Rule-4	F	F	E

⇒ T ⇒ Correct.
F ⇒ False
H ⇒ Home Screen.

* Cause - effect Graph :-

Cause-effect Graph is a black-box testing technique that graphically illustrates the relationship b/w a given o/p & all the factors that influences the o/p. It is also k/a fish-bone diagram because of the way it looks.



Steps for Drawing Cause Effect Gr
(C.E.G.)

(A) Identify & defined the eff
(D) Fill in the effect-box & draw the spine.
(E) Identify the main cause contributing the effect being studied.
(d) For each major branch identify other specific factors which may be causes of the effects.
(E) Categorize relative causes & provide detailed levels of causes.

Date:- 10/09/18 Day:- Monday

Structural Testing :-

→ Control Flow Testing :-

It is a structural testing strategy that uses the program's control flow as a model.

C.F.T. techniques are based on judiciously selecting a set of test path through the program. The set of paths choosen is used to achieve a certain measure of testing thoroughness.

→ Input to test-generation process.

- Source - code.
- Path selection Criteria, statement, branch.

→ Generation of C.F.G. - (Graphical representation of programs' control structure.)

- Compilers are modified to C.F.G.
-

→ Selection of paths -

- ~~Inner~~ Enough entries / ~~Enough~~ Exit path are selected to satisfy path - selection criteria.
- Generation of test - input data.

Two kinds of paths -

1. Executable Path -

There exist input so that the path is executed.

2. Infeasible Path -

No input to execute the path.

Solve the path condition to produce

test input for each path.

Flow graph consist of 3 primitives / notations

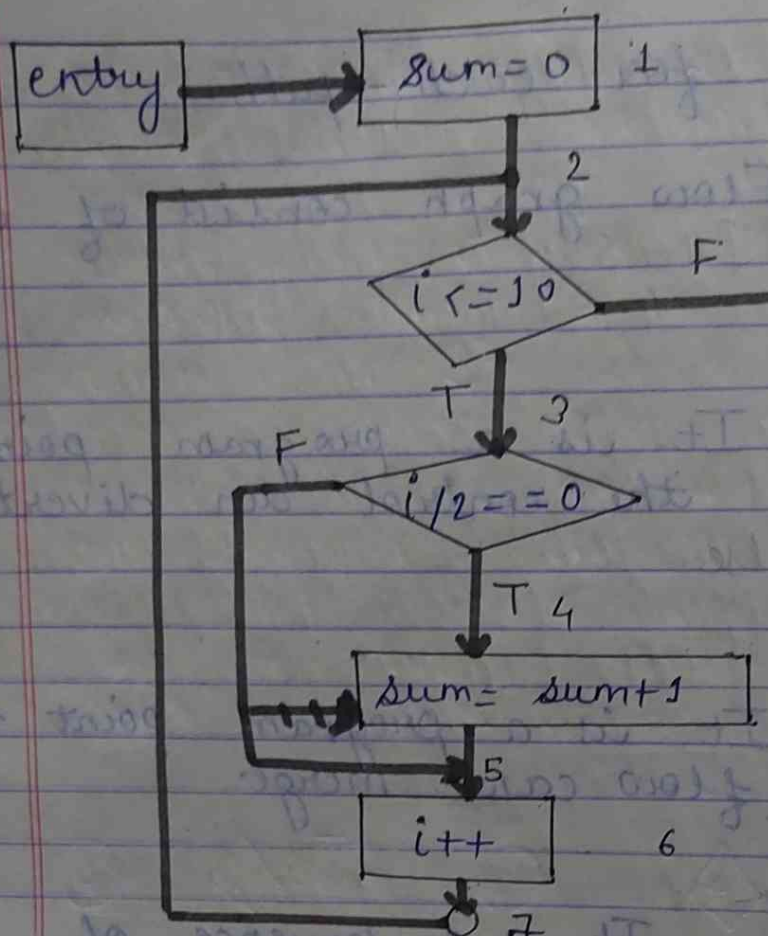
1. → Decision -
It is a program point at which the control can divert.
eg if - else

2. → Junction -
It is a program point where a control flow can merge.

3. → Process Block -
It is a sequence of program statements uninterrupted by either decision or function.

Note:- A program has one entry & one exit.
A program doesn't jump into or out of process.

Note
Edges or links
Decision node
Junction node
Regions



Possible Path

1, 2, 8

1, 2, 3, 4, 5, 6, 7, 2, 8

1, 2, 3, 5, 6, 7, 8

Test Cases- It is a complete path from the entry node to the exit node of a C.F.G.

imp Statement coverage- Every statement in the program has been executed at least once.

1, 2, 3, 4, 5, 6, 7, 2, 8,

Decision coverage-

Every decision in the

program has taken, all outcomes atleast once

1, 2, 3, 5, 6, 7, 2, 3, 4, 5, 6, 7, 2, 8

Path - coverage-

Every complete path in the program has been executed at least once. A loop usually has an infinite no. of completes path.

eg.

Path- Testing :- White-box

It is an approach to test where you enswrethat every path, through a program has been executed atleast once. It is a white-box method for designing test-cases. It analyzes the C.F.G. of a program to find a set of linearly independent paths of execution. It guarantees complete branch coverage. (All edges of C.F.G.) but achieves that without covering of all possible paths of C.F.G.

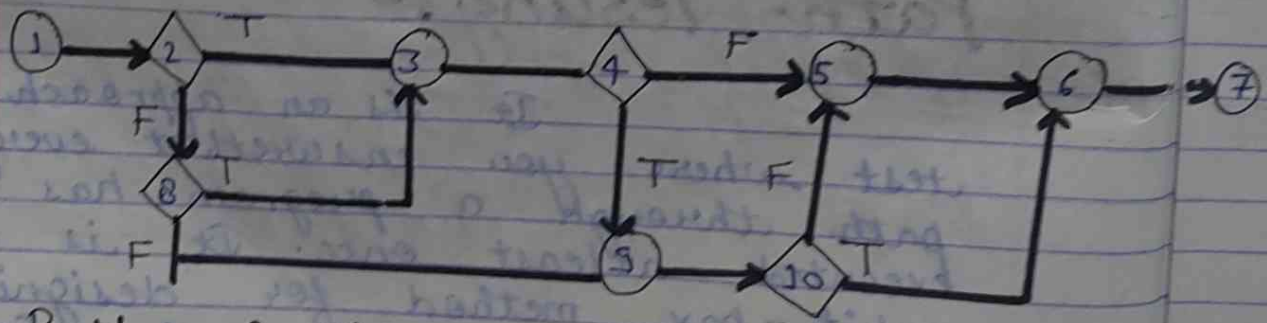
Path - Selection -

1-2-3-4-5
1-2-3-4
1-2-4

It is better to take many simple paths than a few complicated once. There is no harm in taking path that will exercise the same ~~part~~ code more than once.

Select paths as small variation of previous paths.

Try to change one thing in each path at a time.



Path - Sentization -

It is the act of finding a set of solution to a path predicate

If a path - predicate expression has a solution then the corresponding path is achievable otherwise it is unachievable.

P.T.O

Date:- 17/09/18 Day:- Monday

Path Predicate Expression-

1. A complete path may contain succession of decision.

2. An input vector is a type of values corresponds to the vector of input variable.

3. It is a boolean Expression that characterizes the set of input vectors that will cause a complete path to be traversed.

Process Independent Predicates-

A predicate whose truth value can not change as a result of the processing is said to "process independent predicate".

* Independent Paths-

An independent program path is one that traverses at new edge in the flow-graph. It is any path through the graph that introduces

at least one new set of processing statements or new condition.

Cyclometric Complexity-

It is a s/w metrics used to that indicate the complexity of a program it is a quantitative measure of linearly independent path to a program source code developed by "Thomas, J. McCabe in 1976

It is computed using the C.F.G. of the program. It may also be applied to individual function, modules, classes, methods within in our program.

1. It measures the no. of independent path through a program source code.
2. It is computed using C.F.G.
3. Formula used for calculating C.C. ⇒

$$M = E - N + 2P$$

$M = D + 1$

↳ Decision - mode

$N = R + 1$

↳ enclosed region

4. One testing — strategy called - basis path testing by MC-Cabe who first proposed it, is to test each linearly independent path through the program. In this case, the no. of test-cases will equal the C.C. of complexity.

Path Testing techniques

① CFG

② Decision - to - Decision Path (DD path)

③ Independent Path

Date:- 17/09/18 Day:- Monday

S.T.A. Assignment-1

Ques:1. Design Test-cases to select a Product from a given list.

Soln.

TEST-CASE

1. Set of steps & Expected results / condition.
2. Inputs for writing test-cases.
3. Test-case Syntax:
   - Steps.
   - Expected Results.
   - Optional (Test-case id, Test-data, Executed, Passed/Failed/Blocked; Notes)

Steps	Test-case: Browse Product	Expected Result	Status
1.	Login as a visitor	Login successful	P
2.	Select a Product category	Show all the relevant products.	P
3.	Select a Product	Show all the attributes of the Product	P

	(item-code, item-price, item-name)	if tax is includ-ded, the status becomes Fails(F)

Ques.2 Explain Equivalence class of Partition-ing with example.

Soln

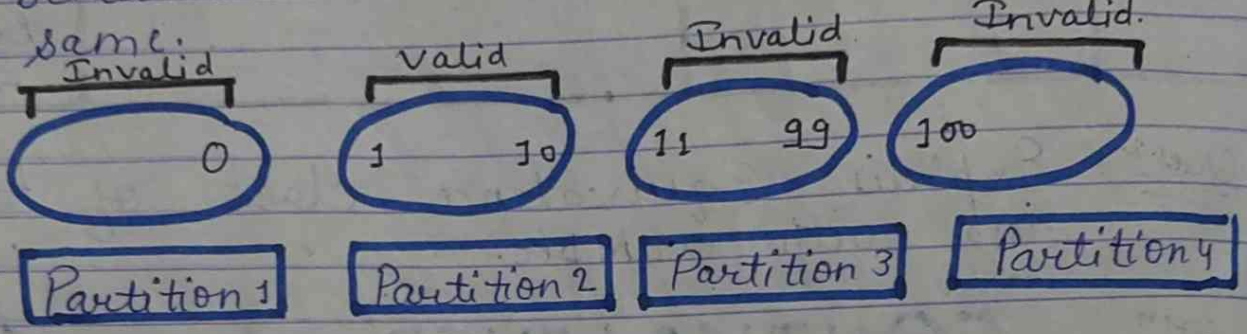
EQUIVALENCE class of PARTITIONING

It is a 'black-box testing'. It divide the set of test condition into a partition that can be considered the same. It divides the input data of s/w into Equivalence data classes.

You can apply this technique where there is a range of input fields.

We can not test all the possible values in the given eg. of online-ticket booking which allows only to book 1-10 tickets at a time because no. of test-cases will more than 100. To address this problem we use equivalence.

partitioning hypothesis where we divide the possible value of tickets into gaps or sets of as shown below where the system behaviour can be consider as the same.



The divided sets are called 'Equivalence partitions' then we pick one value from each partition for testing.

Hypothesis behind this technique is that if one condition /value passes or others will also pass.

10

AB
18/9

Date:- 18/09/18 Day:- Tuesday

Identification of Independent Path:-

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

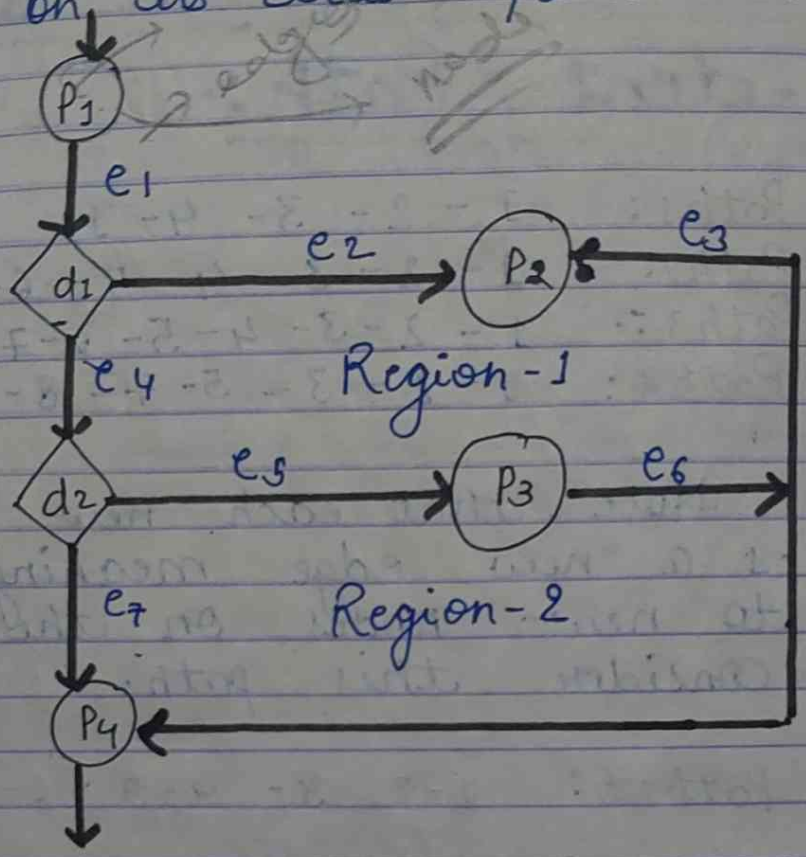
Note that each new path introduces a new edge meaning a new line to new nodes on the path. Now, consider this path.

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

Path-5 is not considered to be an independent path because its simply a combination of the already specified paths (2,3) & doesn't traverse any new edge.

Path 1, 2, 3 & 4 constitute a basic set for a possible C.F.G.i.e. if test can be design to force execution of those initial 4 paths which make up a basic set then every statement in the program will have been guaranteed to be executed at least one time & every condition will have been

executed on its true - false side.



E = N + 2P

I.P. = Edges − Nodes + 2P ⇒ no. of components
= 7 − 6 + 2 × 1 P = 1 (if not given)
= 3

I.P. = R + 1
= 2 + 1 = 3

I.P. = D + 1
= 2 + 1 = 3

Specifically - Verge
graph Theory defines V- cyclometric

no. [V(G)] of a strongly connected graph with N- nodes, E- edges & one connected component.

Data Flow Testing -

It is a family of test strategy based on selecting paths to the program's control flow based on selecting path through control flow in order to explore sequence of events related to the status of variables or data-objects. It focuses on the point at which variables receive values & the points at which these values are used.

It helps us to pin-point any of the following issues -

Advantages of data flow testing

(i) A variables i.e. declare but in never used within the program.

(ii) A variable i.e. used but never declare.

(iii) A variable i.e. define multiple times before it is used.

(iv) Deallocating a variable before it is used.

Generation of ... from Program.

17

Date: 20/09/18 AKTU NOTES HUB Day:- Thursday

White-box unit testing

Mutation Testing

It is a type of S/w Testing where we mutate (change) certain statements in the source-code & check if the test-cases are able to find the errors.

It is a type of white box testing mainly used for unit testing. The goal is to asses the quality of the test-cases which should be robust enough to fail muted-code. It is also called "Fault-based testing strategy" as it envolves creating fault in the program. A mutation is nothing but a single syntactic change i.e. make to the program statement at a time. Each muted program should differ from the original program by one mutation.

eg: Mutation Score - % of killed mutants with the total no of mutants

$$\text{Mutation Score} = \frac{(\text{killed mutants})}{(\text{Total no. of mutants})} \times 100$$

Types - Value Mutations
Decision "
Statement "

→ It is performed when some changes are made to an existing system.

Unit -3

Regression Testing Techniques

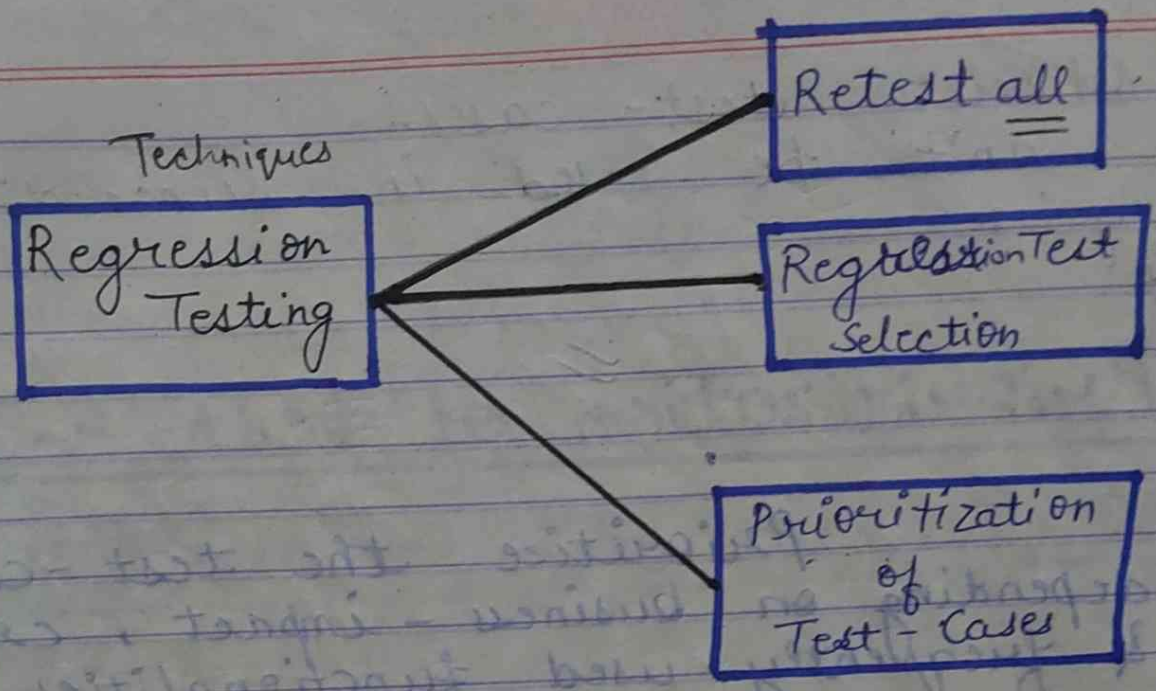
It is rerunning, functional & non-functional test to ensure that previously develop & tested s/w still performs after a change. If not that would be called a 'Regression'.

Changes that may require Regression Testing include bug-fixes, s/w enhancement, Configuration testing & substitution of electronic testing.

It is an integral part of the extreme programming development method. In this method, design documents are replaced by extensive, repeatable & automated testing of the entire s/w package throughout each state of the s/w development process. It is done after functional testing concluded. To verify that other functionalities are working. Traditionally, s/w quality assurance being performed by code testing specialist team or development team as completed work.

However, defects at this stage, most costly to fix. It is the process of testing changes to computer programs to make sure that the older programming still works with the new changes. Test department coder develops code test scenario & exercises that will test new units of code after they have been return these test-cases form the test bucket before a new version of S/W product is released. The old test-cases are run against the new version to make sure that all the old capabilities still working.

S/W maintenance is an activity which include enhancement, error-correction & optimization & deletion of existing features. These modifications may cause the system to work incorrectly. Therefore, Regression testing becomes necessary which is performed using following techniques -



Retest all -

It is one of the methods for Regression Testing in which all the test in the existing-test bucket or suite - should be reexecuted. It is very expensive as it requires huge time & resources.

Regression test selection -

Instead of reexecuting the entire test-suit. It is better to select part of test-suite to be run. Test-case selected can be categorized as-

- Reusable Test Cases
  ⇒ Used in succeeding regression cycle.

- Obsolete test-cases-

⇒ Can't be used in succeeding cycles.

- *

Prioritization of test-cases-

Prioritize the test-cases depending on business-impact, critical & frequently used functionalities. Selection of test-cases based on priority will greatly reduce the test-suite.

Date:- 25/09/18 Day:- Tuesday

Selection of Test-cases

Attractive regression test by selecting following test-cases.

(i) Test cases which have frequent defects. Test-cases having functionalities which are more visible to the users.

(ii) Test-cases which verify core features of the product.

(iii) Test cases of functionalities which undergoes more & decent changes

Reduction Schemes- Priority category scheme
Risk Analysis
Interviewing to identify problem area
Combination schemes

(iv) All integrated test-cases.
(v) All complex test-cases.
(vi) Boundry value test-cases.
(vii) Sample of successful test-cases.
(viii) Sample of failure test-cases.

→ Reducing the no. of test-cases-

Given, a test-suit 'T' represents a set of test-cases ~~{T1 & Te}~~ $T = \{t_1, t_2, \ldots, t_n\}$. A set of test-requirements, a set of test-requirements 'R' = $\{r_1, r_2 \ldots r_n\}$ to be covered and subsets of T,

$S = \{S_1, S_2 \ldots S_n\}$

where each test-suit is associated with Ri.

The objective is to find the representative sub-set

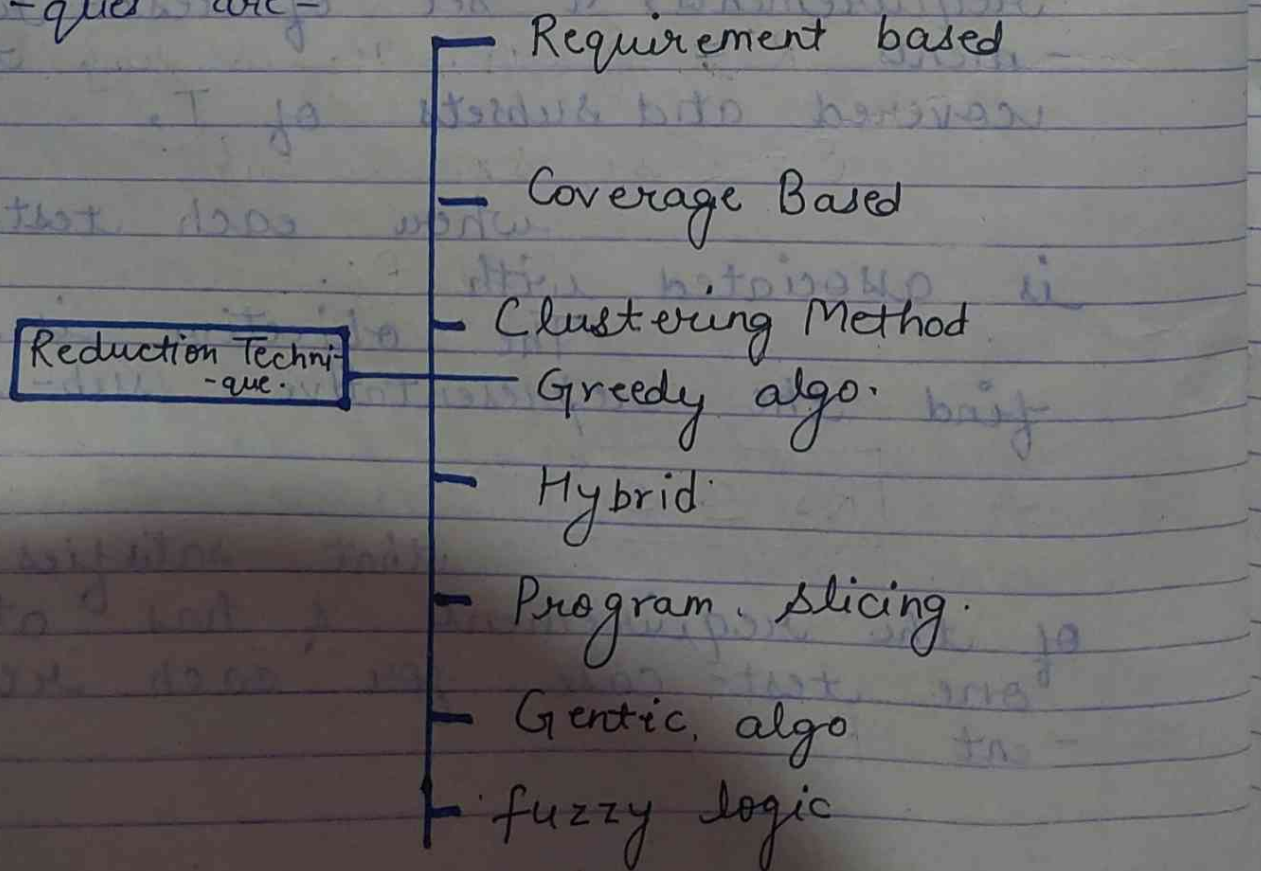
$$RS \subseteq S$$

that satisfies all of the requirements & has atleast one test-case for each requirement R.

Test-suit reduction techniques-

Regression-testing is defined as a s/w maintenance activity which is done to ensure the proper functionality of the s/w. Test-suits that are developed during the development phase have a large size due to time and cost.

Therefore, Regression-test reduction process is advisable in order to reduce the test-suite to minimal set of test-cases that will covers all the faults in minimum time. Test-case reduction techni-ques are-



Date:- 01/10/18 Day: Monday

Code Coverage Prioritazat-ion Technique:-

Test case prioritazation envolves scheduling test cases in an order that increases their effectivenes in meeting some performance goals. One of the common performance goal is to run those test-cases that achieve total code coverage at the earliest. Even with a reduced no. of executable test cases. It must be assure that as many as possible critical faults are found. This means that test-cases must be prioritized. It ensures that important test-cases are executed first so that important problems found early.

Prioritizing test-cases based on perceived risk & customer expresed needs can efficiently deduced the no. of test-cases necessary for comprchensive testing of a s/w application to meet s/w project milestone while ensuri-ng customer's requirements & expectati-ons have been met.

Test-cases prioritazation Techniques:-

Date:- 04/10/18 Day:- ~~Wednesday~~ Thursday.

1. Prioritazation for rate of fault-detection -

The goal is to improve our test-suite rate of fault-detection. & improved rate of fault detection during regression testing can lead s/w Engineers begin there debugging activities earliers.

2. Comporator Technique -

Random ordering of test-~~suite~~ Cases in the test-suite. Optimal ordering. Let us, determine the ordering of test--cases that maximizes the test-suites for fault detection.

3. Statement Level Technique -

Using program instrument-ation we can measure the coverage of statement in a program by its test--cases. We can then prioritize test-cases in terms of total no. of statements they cover by sorting them in order of coverage achieve.

4. Functional Level Technique -

Regression testing & Reference ... Risk ...

Operating at the level of function total function coverage prioritization prioritization test-cases by sorting them in order of the total no. of functions they execute.

Technique is developed in order to done of higher priority in order to to minimize, cost & effort during s/w testing phase.

Date:- 19/10/18 Day:- Wednesday

ASSIGNMENT - 2

RISK ANALYSIS :-

Risk is anything that threatens the successful achievements of a project's goal. The fundamental principle of risk-based testing is to do more through testing to those parts of the s/w system that present the highest risk.

The tester's job is to reveal high-risk problems in product. Using risk-metrices to quantitatively measure the quality of a test-suite seems more reasonable.

Test-Selection Strategy Based on Risk

Major goals of regression - testing are to assure system stability & reliability. Our test- method consists of two parts : test - case selection & end- to- end scenario selection.

A. A sample Risk-model-

1.) The probability of a fault being present.
2.) The cost of a fault in the corresponding function if it occurs during normal operation.

B. Risk- based Test Case Selection -

Step-1. Estimate the test- case for each test-case.
Step-2 Derive severity probability for each test-case.
Step-3. Calculate Risk - Exposure for each test-case

≡ Risk Exposure = Cost x severity probability

Step-4. Select the test -cases that have the highest-values of Risk- Exposure as Safety Tests.

C. Risk- based Test- Scenario Selection

≡ Since end- to-end scenarios many components of the system working together, they are highly effective at finding regression faults. It obeys two rules-

R1. Select Scenarios to cover the most critical

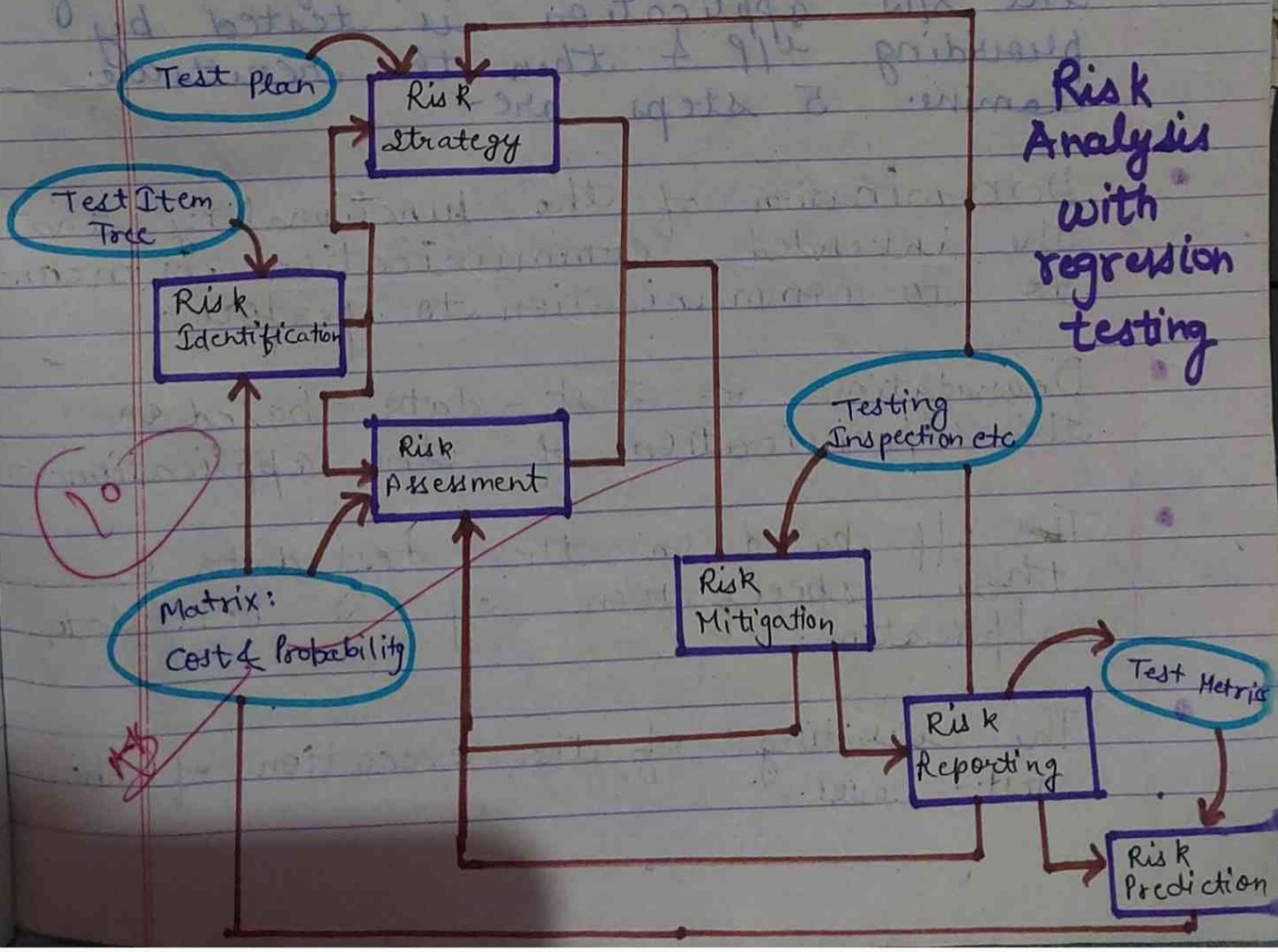
Reducing the no. of test-cases - Prioritization guidelines, Priority Category, scheme; Risk Analysis.

test-cases.

R2. Ensure scenarios cover as many test-cases as possible.

Risk analysis includes - Activities-

(i) Group similar risks
(ii) Determine risk drivers
(iii) Determine source of risks
(iv) Estimate risk exposure
(v) Evaluate against criteria



Date: 12/10/18 Day:- Friday

Unit-4
Software Testing Activities

* Levels of Testing:-

It include different methodology can be used while conducting s/w testing.
The main levels are-

① Functional testing - It is a type of "Black-Box testing" i.e. to based on the specification of the s/w application is tested by providing i/p & then the results are examine. 5 steps are-

- Determination of the functionality that the intended communication is meant to communication to perform.
- Degradation of test-data based on the specification of the application.
- The o/p based on the test-data & they specifation of the test-data application.
- The writing of the execution of the test-cases.

- The comparison of actual & expected result based on the executed test cases

Types -

(i) Unit testing -
(ii) Integration testing -
(iii) Regression testing -
(iv) Acceptance < α - testing
β - testing
(v) System testing -

(ii) **Non-Functional Testing**

It is based on testing an application from its non-functional attributes. It involves testing a s/w from the requirements such as performance, security & user-interface.

(a) **Performance testing -**

Used to identify any performance issues rather than finding bugs in a s/w. It is considered as one of the important & mandatory testing type in terms of following aspects that speed, capacity, stability & scalability

It can be divided into different sub-types - such as-

(i) Load- Testing - Process of testing the behaviour of s/w by applying maximum load in terms of s/w accessing & manipulating large i/p data. It is perform with the help of automated tool such as -
* Load - Runner & App Loader, IBM Rational performance - tester, Visual studio Load Test.

(ii) Stress Testing - Aim is to test the s/w by applying load to the system & taking over the resources used by the s/w to identify the breaking point, can be performed by testing different scenarios -
(a) Shut Down or restart of Network ports randomly.
(b) Turning the database on or off
(c) Running different processes that consumes resources such as- CPU, Memory, Server etc.

(b) Usability testing -

It is a Black-box testing used to identify any errors & improvements in the s/w by observing the users through their usage & operation.

(c) Security Testing -

It involves testing a s/w in order to identify any flaws & gaps from security & vulnerability (attack) point of view. It ensures confidentiality, integrity, availability, authentication, accountability, Non-repudiation, s/w data is secure.

(d) Portability Testing -

It includes testing a s/w with the aim to ensure its reusability & that it can be moved from another s/w as well. It is consider as one of the sub-parts of system-testing. following are the strategies that can be used for portability testing.

(i) Transferring & install s/w from one computer & other.
(ii) Building .exe files to run the s/w different platforms.

Date:- 19/9/18

Day:- Tuesday

Exploratory Testing -

It means testing of s/w without any specific plans & schedules This is formal testing process where we don't have any test-cases or test-planning document to test the application by exploring the application on & learning the application tester's design. The test-cases & simultaneously execute them as well. It is a testing approach that allows you to apply your ability & skill as a tester in a powerful way.

Advantages:-

1. It doesn't require preparation for testing as we don't have document for testing.

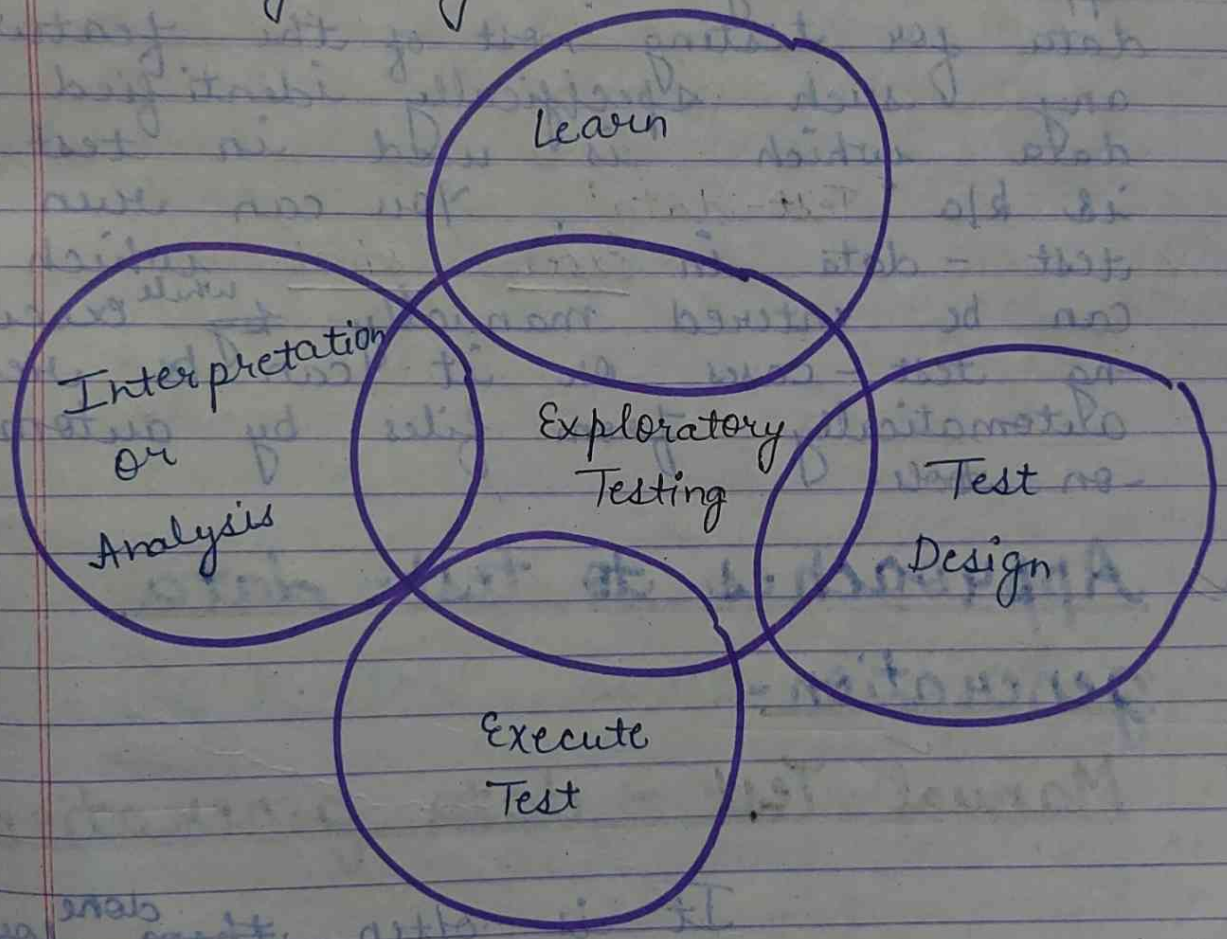
2. In this time saves due to all task are doing simultaneously like- testing, designing, test-scenarios & executing test-scenarios.

3. Tester can report many issues, due to incomplete, or missing requirement document.

Disadvantage

* Few issues can not be catched in this type of testing.

① There is review of test - planning & designing of test - cases / scenario while testing may cause issues.



Date: 31/10/18 Day:- Wednesday

Test - Data :-

It is data which has been specifically identified for use in test typically of a computer program.

Some data may be used in a confirmatory way typically to verify that a given set of input to a given function produces some expected result in order to test a S/W application you need to enter some data for testing most of the features any such specifically identified data which is used in test is k/a 'Test-data'. You can run test-data in Excel sheet which can be entered manually ~~by~~ while executing test-cases or it can be read automatically from files by automation tools.

Approaches to test-data generation-

1. Manual Test-data generation-

It is often done for carefully covering the essential test-cases. This is particularly straightforward way of creating test-data for various scenarios are tested with different types of test-data' such as-
(i) Null test data.
(ii) Invalid test data.

(iii) Valid Test-data
(iv) Data set- call performance.

2. Automated Test-data generation-

The chief-differentiating factor of automated testing over manual testing is the significant acceleration of speed.

Data generation tools speed up this process & help to reach higher volume levels of data. Tools such as-
- Selenium / Lean F.T. Pump data into the system considerably faster.
- Web-services API can also be used.

3. Back-end data-Injection:-

The back-end servers which comprise the database are required in this technique. Test-data are stored at the database can be used directly update the existing database. There by acquiring data voluminess data instantly through SQL queries.

4. Third-Party Tools:-

These are available in the market heard significantly with data - creation & injection. They understand data- residing in the back- end application & help pump- in data i.e. similar to a real - time scernario. It enables wide - test coverage.

Automated Test - data generat- -ion using a genetic algorithm :-

. The use of meta heuristic search technique for the automatic generation of test - data has been a great - interest for many researchers in recent years. Previous attempts to automate the test- gener- -ation process have been limited & constraint by the size & complexity of s/w & the basic fact is that test - data generation is an undec- -idable problem. Meta heuristic search techniques much often promise in regard to these problems. This technique have been high level frame work which utilize to seek solutions for combinatorial problem at a reasonable cost.

A new revolutionary approach for automated test-data generation for structural-testing is also introduced. It uses a newly defined program modelling allowing an easy program manipulation. Further more, we define a cross over operator allowing to effectively improving individuals. The need to automate s/w testing has provided ~~whi~~ rich set of challenging problems for the research community one approach to s/w test automation that has achieve a great deal of recent attention in search-based s/w testing.

SBST uses meta heuristic algorithm to automate the generation of test-inputs.

Automatic test data generation using a genetic algorithm would be computationally expensive. So, we do not perform a complete symbolic execution. Rather we compute smaller amounts of symbolic information.

This is possible by using weakest pre-condition.

- Code Coverage.
- Branch Coverage.

} Coverage - Criteria to automatically generate test-cases.

Backward Symbolic Execution by weakest pre-condition allows you to not execute entirely the program But just the desired parts.

Automated Test-tools :-

Telerik Test - Studio-

It is a comprehensive & one of the most intuitive automation testing tools available. It offers robust functional U.I. testing, Exploratory, load, performance, testing in visual studio & mobile testing apart from manual testing capabilities.

Selenium-

Automated S/W testing tools for testing web application it automates browser, enabling user to sail through various browser testing purpose.

Robotium-

Popular automated

testing framework for android. It supports native & hybrid authentication & makes writing automated black-box test-cases is easy.

Watir :-

Watir is an open source tool to automate web browser. Ruby enables to connection to database, read-files, export XML etc. & also structures your code as reusable libraries.

~~Test~~

Test-Drive-

It helps in rapid automation, it effectively test browsers & applications like- AJAX, Java & Silverlight. apart from G.U.I.

Object-Oriented

Day: Saturday

Date:- 03/11/18

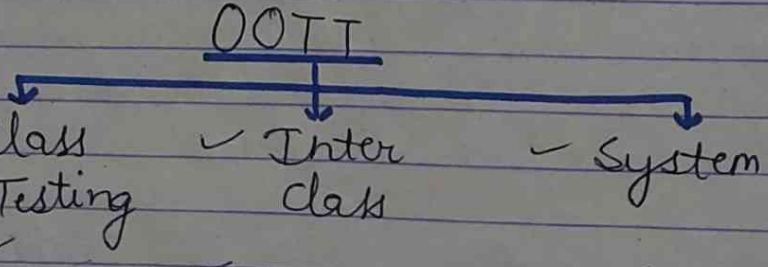
Unit - V

Object-Oriented Testing

Whenever large scale systems are designed, OO testing is done rather than the conventional testing. The OO testing revolves around the fundamental entity i/a class. With the help of class system larger system can be divided into small well-defined units which may then be implemented separately.

The OO testing can be classified as like conventional systems. These are called as levels of testing.

OO testing - levels / techniques -



Date:- 04/11/18

① Acceptance - criteria - product acceptance
① " " - procedure Day. - Sunday
⑧ " " - service - level agreement

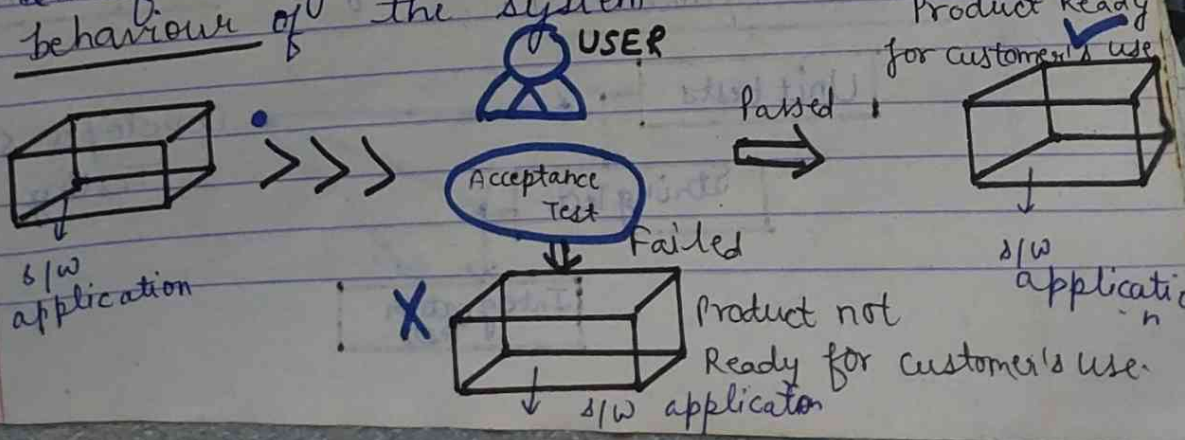
Assignment - 4 & 5

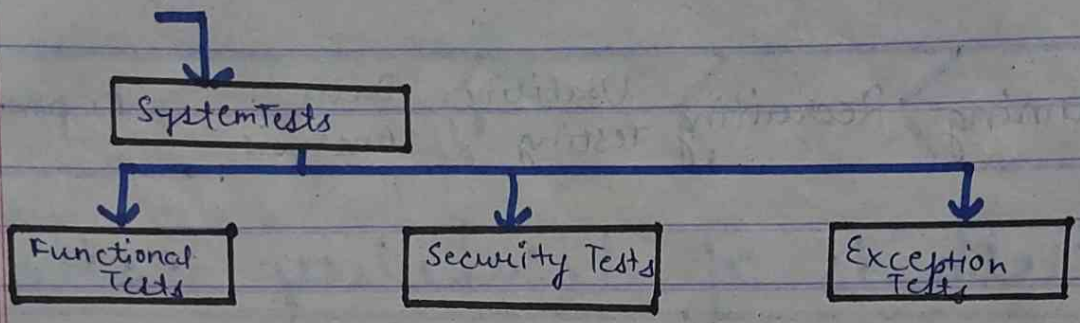
Define -

1. Acceptance Testing :-

Acceptance testing is formal testing conducted to determine whether or not a system satisfies its acceptance criteria & to enable the customer to determine whether or not to accept the system.

The purpose of this test is to evaluate the system's compliance with the business requirements & assess whether it is acceptable for delivery. These tests are often pre-specified by the customer & given to the test team to run before attempting to deliver the project / product. The customer reserves the right to refuse delivery of the s/w if the acceptance test-cases do not pass. This testing is performed with realistic data of the client to demonstrate that the s/w is working satisfactorily. It focuses on external behaviour of the system.





3. Usability-Testing -

Usability Testing is a type of s/w testing where, a small set of target end-users, of a s/w system, "use" it to expose usability defects. This testing is mainly focuses on the user's ease to use the application, flexibility in handling controls & ability of the system to meet its objectives. It is also called User Experience Testing.

This testing is recommended during the initial design phase of SDLC, which gives more visibility on expectations of the users.

Goals of Usability Testing -

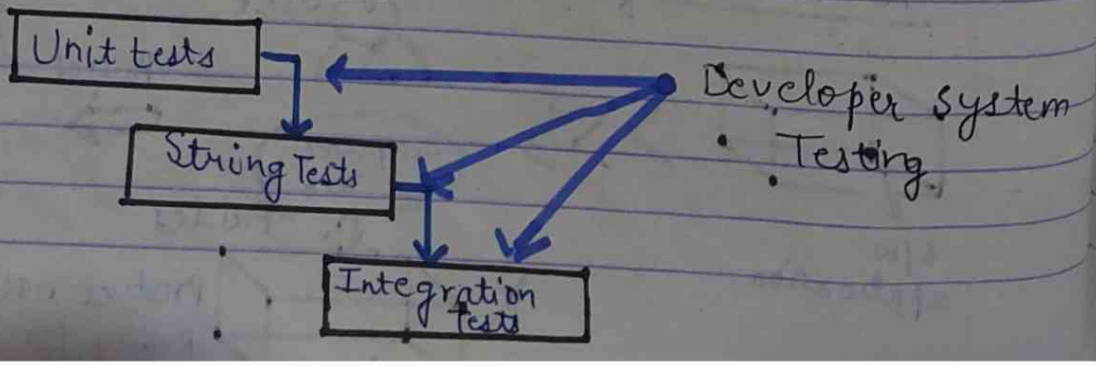
It determines whether an application is -

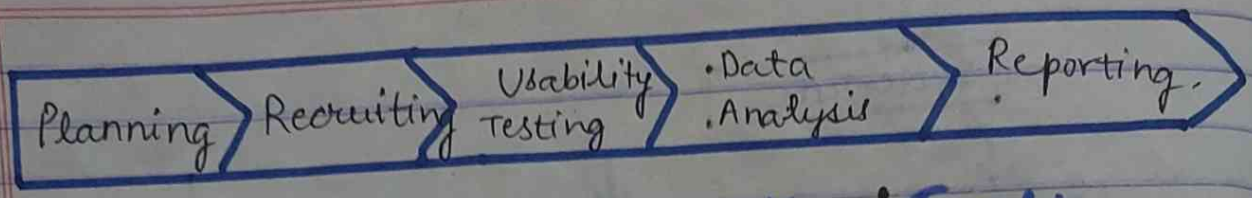
- Useful
- Findable
- Accessible
- Usable
- Desirable

Usability Testing Process -

2. System testing -

- As the name implies, all the components of the s/w are tested as a whole in order to ensure that the overall product meets the requirements specified.
- It is very important step as the s/w is almost ready to ship & it can be tested in an Environment which is every close to that which the user will experience once it is deployed.
- System testing enables testers to ensure that the product meets business requirements, as well as determine that it runs smoothly within the its operating Environment.
- This type of test is typically performed by a specialized testing team.
- The goal is to see if the s/w meets its requirements.





Methods of Usability Testing -

There are two methods available to do usability testing -

1. Laboratory Usability Testing -

This testing is conducted in a separate lab room in presence of the observers. The testers are assigned tasks to execute.

2. Remote Usability Testing -

Under this testing observers & testers are remotely located. Testers access the System Under Test, remotely & perform assigned tasks. Tester's voice, screen activity, testers facial expressions are recorded by an automated s/w.

Pros & Cons of Usability Testing -

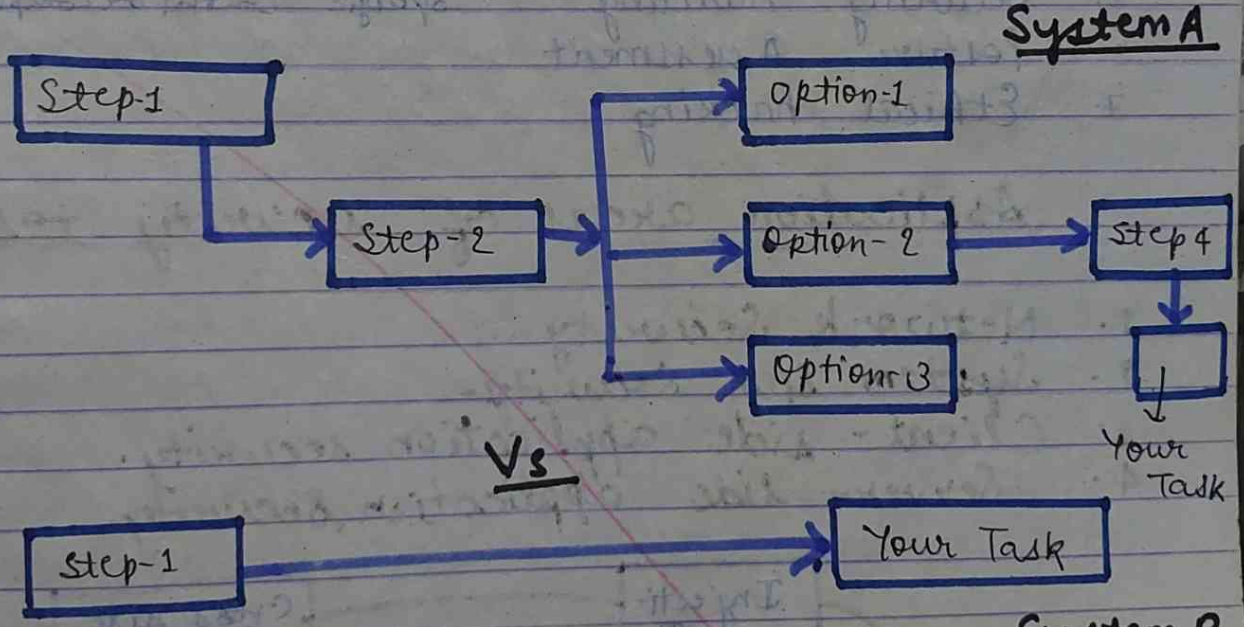
Pros -

1. It helps uncover usability issues before the product is marketed.

It helps improve end-user satisfaction.
It makes your system highly effective & efficient.

Cons-

Cost is a major consideration in usability testing. It takes lot of resources to set up a Usability Test Lab. Recruiting & mgmt. of usability testers can also be expensive.



4. Security Testing-

Security Testing is a variant of sw Testing which ensures, that system & applications in an organization, are free from any loopholes that may cause a big loss. Security testing of any system is about finding all possible loopholes & weaknesses of the

Concepts of security = Authentication, Authorization, Confidentiality, Integrity, Non-Repudiation, Availability (deny)

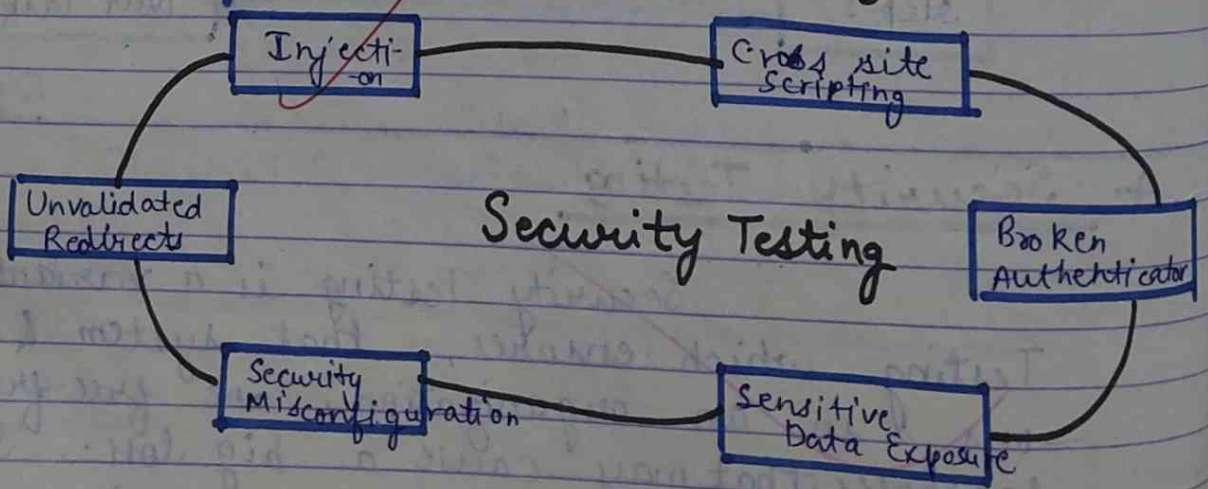
system which might result into a loss of information at the hands of the employees or outsiders of the Organization.

Types of Security Testing -

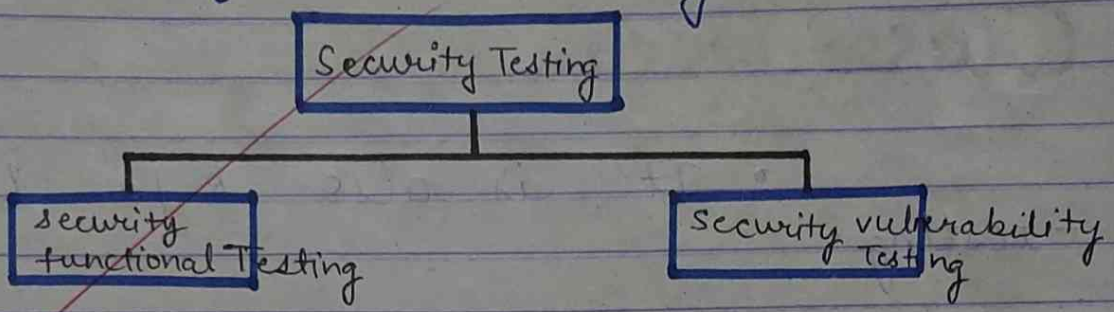
0. Discovery - Version detection, identify systems.
1. Vulnerability Scanning ⇒ known security issues with its vulnerability
2. Security Scanning verification in form of authorized access
3. Penetration testing simulates an attack by a malicious party depth of attack
4. Risk Assessment
5. Security Auditing Driven by an audit/risk function at specific control or compliance issue
6. Posture Assessment
7. Ethical hacking

Application areas of security testing -

1. Network Security -
2. System s/w Security -
3. Client-side application security.
4. Server-side application Security.



Security testing is the activity of assessing a system for the presence of security weakness, reveal flaws in the security mechanism.



[o P1o

AB 12/11/18

Category :-

(1) Test mgmt. Tool
- TET
- Test manager, QA complete, DevTest, Test log

(2) functional Testing tools
(1) Selenium
(2) Watir

(3) Load Testing tool -
- HP Load Runner
- Forecast
- Webserver Stress Tool

Date :- 12/Nov/18 Day - Monday

Class-testing-

- It is also k/a Unit-testing.
- In class-testing every individual classes are tested for errors or bugs.
- It ensures that the attributes of class are implemented as per the design & specification.

Inter-class Testing-

- It is also k/a Integration or sub-system testing.
- It involves the testing of modules & their co-ordination with other modules.
- The system is tested & primarily by functional testing techniques are used to test the system. Non-functional requirements are also tested.

Web-application testing-

Web testing is checking your web application for potential Bugs before Its may live or before code is move into the production - Environment.

During this stage issues such as- that of web- application security, the functioning of the site. Its access to handicapped as well as regular users & its ability to handle traffic is checked.

Some of the following testing types may be performed depending on your web - testing requirement.

Web - Application testing checklist-

1. Functional - Testing -

Test all links in your web- pages are working correctly & make sure there are no broken - links.
Links include- outgoing links, anchor -link, Mailto links.

Test- cookies are working as expected Cookie- testing will include- testing - -cookies are deligent either when cache is clear or when they reach their expiry & delete Cookies & testi that login credentials are asked

when you next visit the site.
Test HTML & CSS to ensure that search-engines can crawl your site easily.

Test - business workflow diagram -

2.→ Usability testing -

Test the site navigation.
Test the content.

3.→ Interface Testing -

Three areas to be tested there are -
application - Test Request are sent correctly to the database & output at the client-site is displayed correctly.
Errors if any must be caught by the application & must be only shown to the administrator not the end user.

Web - server -

Test web-server is handling all application request without any service denial.

Database - server -

Make sure that queries sent to the D.B. give extracted result.

4→ Database Testing –

Testing activities will include Test if any errors are shown while executing queries data integrity maintain while creating, updating by deleting data in database.

Test data- retrieve from your database in your web- application.

Check response- time of queries & find accurate

5→ Compatibility testing-

It ensures that your web- application displays correctly across different devices. This would include-

Browser compatibility test –

Same website in different browsers will display differently. You need to test if your web application is being displayed correctly across browsers, Javascript, AJAX & authentication is working fine. You may also check for

mobile bro
o.s. compatibility test

→

6. → Performance testing -

This will ensure your site works under all loads.

7. → Security testing -

It is vital for E-com. websites that stores sensitive customer information like- credit card.

8. → Cloud testing -

You will select a large no. of people to execute test which otherwise would have been executed via selected group of people of a company.

Date- 13/11/18 Day - Tuesday

* User- Interface Testing -

It is a testing

technique used to identify the presence of defects in a product / s/w under test by using GUI.

Characteristics -

① GUI is a hierarchical, graphical, front-end to the application, contains graphical objects with a set of properties.

② Able to provide inputs to the GUI objects.

③ It has capabilities to exercise GUI events like keypress, mouse-click.

Approaches -

Manual Based - Based on the domain & application knowledge of the tester.

Capture & Replay - Based on Capture & Replay of user actions.

Model Based Testing - Based on the execution of user-sessions. Based on GUI models. Various GUI models are ① Event Based

model-
Based on all events of the GUI need to be executed atleast once.

(ii) State Based Model-
All states of the GUI are to be exercised atleast once.

(iii) Domain model-
Based on the application domain & its functionalities.

Checklists -

(i) Check screen validation, verify all navigations.
(ii) Check usability testing.
(iii) Verify data integrity
(v) Verify the object states.
(iv) Verify the data - fields & numeric data field format.

(vi) Database Testing -

It is checking the schema, triggers of the database under test. It may involve creating complex queries to load / stress test the database &

check its responsiveness. It checks data integrity & consistency.

Types of Database Testing-

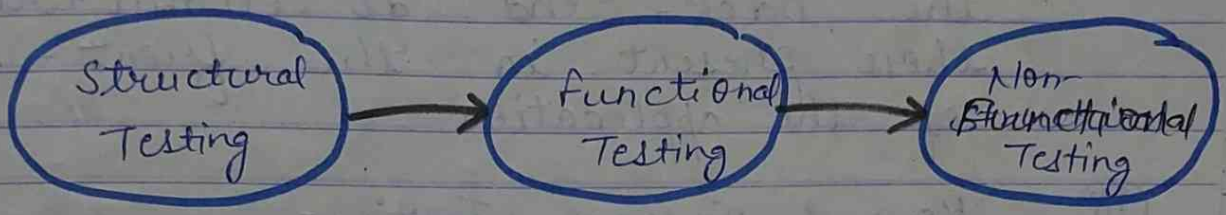


Fig Types of db- testing

Structural Database Testing-

Structural Data Testing involves the validation of all those elements inside the data- repository that are used primarily for storage of data & which are not allowed to be directly manipulated by the end- users.

(1) Schema- Testing - To ensure that the schema -mapping b/w the front-end & back- end are similar. Validation of the various schema formats associated with the databases need for verification in the case of unmapped tables & columns

(ii) Database - Table, columns Testing-

Various checks are the - Variation of the compatibility of the data-type, field-type of the back-end db column with those present in the front-end of the application.

(iii) Keys & Indexes Testing-

Check whether the primar-if & foreign key have been created for the required table. Check the size & length of the required field & Indexes. Check whether the refrences for foreign keys are valid.

(iv) Trigger Testing-

Validation of the requir-ed, update, insert, detete, trigger functionality in the realm of the application under test

(v) DB. Sewer Validation-

Check the db server configuration & as specified by

the business requirement.

Functional db testing -

It needs to ensure most of those transactions & operations as performed by the end-users are consistent with the requirement specification.

(i) Check integrity & consistency.
(ii) Login & User Security.

1. Black-box testing 2. White-Box testing

Non-Functional Testing - ① Load, ② Stress

Date 15/11/18 Day:- Thursday

Post-deployment testing / planning (in use)

It is a critical part of any s/w implementation. Items should be tested & documented as soon as possible. ~~for eg~~ - documents for-m specific deployment. See the deploym-ent project pages. Hiring / staffing to support this function. Everyone should understand who is responsible for each function in this area ensure that each area is covered by the existing staff or new start outsour-ced / hired.

Troubleshooting & End-user support -

Troubleshooting & End-user support consists of processes of end-user support & resolution. Hiring / staffing to support this function

Application administration -

Processes for performing application administration duties such as - making configuration changes to adding / changing ~~pro~~ offices, adding / changing ~~product~~ offices rows & cols & permissions.
Processes for performing user maintenance

Documentations.
Hiring / staffing support this function.

The objective of the deployment testing activity is to ensure that the ~~gaube~~ s/w -

(i) Can be automatically deploy ;
(ii) Once deployed, it starts and activated correctly.

Deployment testing activity forsees the involvement of the

following goo roles -

Tester - responsible for the execution of the deployment test-phase & to submit deployment test-bugs.

Developers - responsible for fixing deployment test bugs.

The following criteria can be used to decide which suggestions needs attenti-on.

1. Frequency of suggestion.
2. Source of feedback
3. Cost of implementing suggestion.
4. Impact of implementing the suggestion.

* Post deployment testing may reveal those problems which went undetected before deployment of the web-application.
* Despite all planning & testing carried out before deployment, obtaining user opinion is important for improvement of a website & it ensures that the website adapts to the need of the user.

Crypto Numericals

Qu① Find the result of - 7 mod 10

6 | 27 (3
-18
9

Soln Dividing - 7 by 10

Add -7 to modulus 10
Add -7 + 10 = 3
And So, -7 mod 10 = 3

② 27 mod 6
Dividing 27 by 6 ⇒ 3
Add 3 + 6 = 9 Ans

③ Find the value of $\phi(10)$

Soln By using Euler's ϕ function
10 can be factorized as 5 × 2
$\phi(10) = \phi(5) \times \phi(5)$
$\Rightarrow 1 \times 4 = 4$ Ans

④ Use RSA algo. on Plaintext alphabet G, the values e = 3, n = 15; find out what this plain-text alphabet encrypts to?

$d \cdot e \equiv 1 \bmod \phi(n)$
$d \cdot 3 \equiv 1 \bmod 8$

Soln Plain-text G = 7
Encryption = $m^e \bmod n$
$= 7^3 \bmod 15$
$C = 13$

$\phi(15)$ $\phi(5)$ $\phi(3)$ 4 2 1 1

⑤ Prove that modular arithmatic property given as -

$[(a \bmod n) - (b \bmod n)] \bmod n$
$= (a - b) \bmod n$

* No. of Secret key = $\frac{n(n-1)}{2}$

Sol^n Define $(a \bmod n) = r_a$
$(b \bmod n) = r_b$

$a = r_a + jn$ for some integer j
$b = r_b + kn$ for some integer k.

$$(a-b) \bmod n = (r_a + jn - r_b + kn) \bmod n$$
$$= r_a - r_b + (k+j)n \bmod n$$
$$= (r_a - r_b) \bmod n$$
$$\Rightarrow [(a \bmod n) - (b \bmod n)] \bmod n$$

LHS ≡ RHS

Qu-5 Operation of mod Operator.

Sol^n The mod operator creates a non-negative residue (r) which is denoted as-

$a \bmod n \equiv r$

$Z = \{\ldots, -2, -1, 0, 1, 2, 3 \ldots\}$

$a = q \times n + r$ Relation

n (positive) → q, r (non-ve)

Q-6. Encrypt the message "THIS IS AN EXERCISE" using Playfair Cipher with key = DOLLARS.

Sol^n

P.T.O

D	O	L	A	R
S	B	C	E	F
G	H	I/J	K	M
N	P	Q	T	U
V	W	X	Y	Z

जो letter पहले आयेगा उसके left में एक छोड़कर

THIS IS AN EXERCISE ← Message

TH – PK
IS – GC
AN – DT
Ex – CY
ER – FA
CI – IQ
SE – BF

Qu 7. Use CRT- $x = 2 \bmod 7$ — (i)
$x = 3 \bmod 9$ — (ii)

$$M = 7 \times 9 = 63$$

$$M_1 = \frac{M}{M_1} = \frac{63}{7} = 9$$

$$M_2 = \frac{M}{m_2} = \frac{63}{9} = 7$$

$M_1 y_1 \equiv 1 \bmod M$
$9 \cdot y_1 \equiv 1 \bmod 63$

$M_2 y_2 \equiv 1 \bmod M$
$7 \cdot y_2 \equiv 1 \bmod 63$

Q. Compute the value of $5^{17} \bmod 11$ & $11^{17} \bmod 5$

Soln $(5^2 \bmod 11)(5^{15} \bmod 11)$

$\Rightarrow a^y \bmod n$

$17 = 10001$

2	17	1
2	8	0
2	4	0
2	2	0
2	1	

* $5^{17} \bmod 11$

$\Rightarrow$

i	x_i	$y \leftarrow a \times y \bmod n$	$a \leftarrow a^2 \bmod n$
			$a \leftarrow (5)^2 \bmod 11 = 3$
0	1	$y \leftarrow 5 \times 1 \bmod 11 = 5$	$a \leftarrow 9 \bmod 11 = 9$
1	0	⟶	$a \leftarrow 81 \bmod 11 = 4$
2	0	⟶	$a \leftarrow 16 \bmod 11 = 5$
3	0	⟶	
4	1	$y \leftarrow 5 \times 5 \bmod 11 = 3$	

$\Rightarrow$ 3 Ans

* $11^{17} \bmod 5$

i	x_i	$y \leftarrow a \times y \bmod n$	$a \leftarrow a^2 \bmod n$
			$a \leftarrow (11)^2 \bmod 5 = 1$
0	1	$y \leftarrow 11 \times 1 \bmod 5 = 1$	$a \leftarrow 1 \bmod 5 = 1$
1	0	⟶	$a \leftarrow 1$
2	0	⟶	$a \leftarrow 1$
3	0	⟶	
4	1	$y \leftarrow 1 \times 1 \bmod 5 = 1$	

$\Rightarrow$ 1 Ans

2 | 44
2 | 22
11 | 11

Qus d Find the value of Euler's Totient no.

$\phi(88) = \phi(8) \cdot \phi(11) \Rightarrow \phi(4\times2)\cdot\phi(11)$

$= 3\times1 \times 10 = 30$

$\boxed{\phi(88) = 30}$

* Group $G = \{G, \cdot\}$ → binary operation
  1. Closure = $c = a \cdot b$
  2. Associativity $(a\cdot b)\cdot c = a\cdot(b\cdot c)$
  3. Identity $e\cdot a = a\cdot e = a$
  4. Inverse $a\cdot a' = a'\cdot a = e$

* Ring $R = \{R, +, \times\}$
  1. Abelian group +
  2. Closure under ×
  3. Associative ×
  4. Distributive law
     $a(b+c) = ab+ac$
  5. Commutative
  6. Multiplicative Identity
     $a\cdot1 = 1\cdot a$.
  7. No zero divisors $ab = 0$

* Field $F = \{F, +, \times\}$

1. F is an integral domain i.e. Satisfies A1 to A5 & M1 to M6.

M7 ⇒ Multiplicative inverse
⇒ except 0, $\exists\ a^{-1} \in F$
$a\cdot a^{-1} = (a^{-1})a = 1$

imp

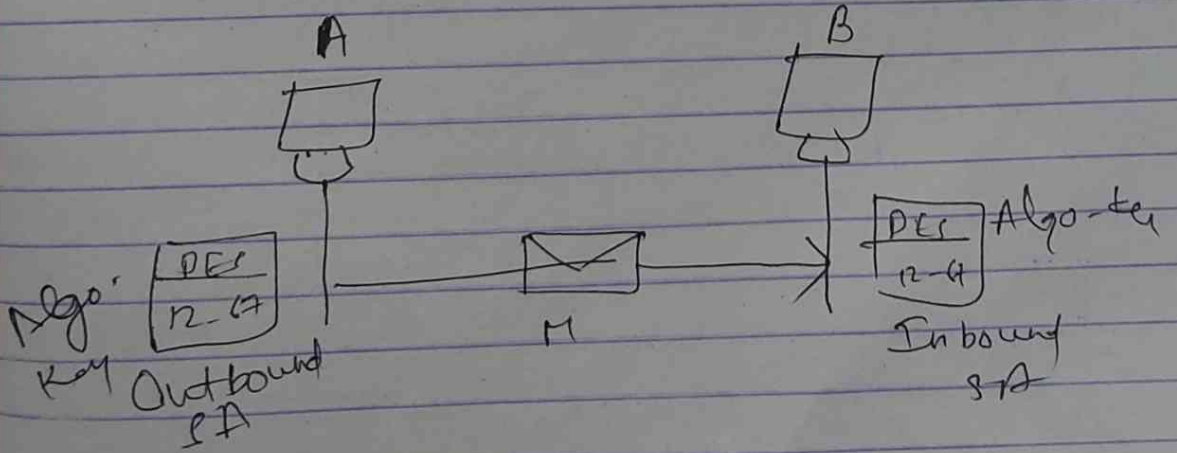
Que Concept of Security Association-

* A key concept that appears in both the authentication & confidentiality mechanisms for IP in SA.
* An association is a one-way relationship b/w a sender & a receiver that affords security services to the traffic carried on it. A/c for peer relationship = two S.A.s
* Security services are offered to an SA for the use of AH or ESP but not both.
* 3 parameters-

(i) SPI - A bit string is assigned to this SA & has local significance only. It is carried in A.H. & E.S.P headers to enable the receiving system to select the SA under which a received packet will be processed.

(ii) IP destination address - Currently, only unicast addresses are allowed

(iii) Security protocol Identifier - indicates whether the association is an AH or ESP S.A.



Firewall

Types of ~~Gateways~~:

1. Application-level
2. Circuit-level
3. Packet-filtering

1. Application level

- May not ~~perfect~~ protect fully against internal threats.
- Improperly secure wireless LAN
- Laptop, PDA, portable storage device infected outside then used inside. ×

Many organisations buy a expensive firewall but neglect o numerals ~~of~~ other back~~bones~~ door into the d networks.

Another thing, a firewall can't rarely protect you against this status inside our network.

Types of Firewall-

There are 3 basic types of 'Firewall'.

(i) Packet Filter
(ii) Application level gateway
(iii) Proxy Gateway or circuit-level gateway.

(i) Packet Filters:-

Packet filtering router applies to a set of rules to each incoming IP packet & then forward and discard it packet filter it physically set up as a list of set of goals based on matches of filter in the I.P.

(ii) Application-level Gateway:-

It is also called a 'Proxy server' act at a relay of application level packages. User contact gateway using an application & the request is successful after authentication. It is the service specific such as- FAP, Telnet, SMTP or UDP.

(iii) Circuit Gateway:-

Circuit level gateway can be a stand alone all a specialized system. It doesn't allow it to end to end TCP connection. The gateway sends a 2 TCP connection. Once the TCP connection are established, the gateways relays TCP segments from one connection to the another without examine the content.