Software Testing

DATE: 3/2/21
PAGE: 1

☆ What is Software Engineering?

Software Engineering is the application of systematic and quantifiable approach to the design, development, and maintenance of Software.

☆ What are the Common SDLC Models?

I. Big Bang
II. Code and fix
III. Waterfall
IV. Iterative
V. V-model

1. Big-Bang Model :-

The Approach :- People + Money + Energy = Perfect Software or Nothing!

Advantage :-

Simple, needs less planning, less Formal development process.

Disadvantage :-

: No formal testing activities, hard to manage and monitor, risky.

DATE: __/__/____
PAGE: _____

2. Code and Fix Model:-

- The Approach

Informal Specification → loop of Code and Fix till enough - is - enough decision → Releso.

- Advantage

little planning and documentation work.

- Disadvantage

Hard to manage the testing and Control the quality.

3. WaterFall Model:

- The Approach

Discrete phase, review after each phase will decide to move next or stay at this phase till it's ready.

- Advantage.

Easy to test (Clear requirement), Easy to manage and Monitor.

- Disadvantage.

High bugs Fixing and changes Cost, the Customer has Minimal Involvement during the Project.

DATE: __/__/____
PAGE: _____

4. Iterative Model :-

- The Approach

Dividing the project into builds, each build can be executed by waterfall model.

- Advantage.

Easy to test. Customer is tightly Involved, Change cost is low.

- Disadvantage.

Time Consuming.

5. V- Model :-

- The Approach

For every single phase in the development cycle there is a directly associated testing phase. It is an extension of waterfall.

Advantage:

Testing is exist in every phase.
Defects are detected early.

Disadvantage.

Time Consuming.

Good Write

Introduction to Quality Control Engineering

DATE: __/__/____
PAGE: _____

★ What is Software Quality Control (QC)?

Software quality Control refers to Specified functional requirement as well as non-Functional requirement.

These see Specified procedures and outlined requirement lead to the idea of Verification and Validation and Software testing.

★ What is Software Testing?

A quality Control activity aimed at evaluating a Software item against the given system requirement.

This includes, but is not limited to, the process of executing a program or application with the intent of finding Software bugs.

Testing is QC activity.

★ Verification :- Confirming that something (Software) Meets its Specification.

Validation :- Confirming that it meets the user's requirement.

The two Major V and V (Verification and validation) activities are reviews, and testing.

Good Write

DATE: __/__/____
PAGE: _____

★ What is the Software 'Bug' ?

Things the Software does that while it is not supposed to do, or Something the Software doesn't while it is supposed to.

★ Why Does Software Have Bugs ?

- MisCommunication or no Communication.
- Programming Errors.
- Changing requirement.
- Time pressures.
- Ego
- Poorly documented Code
- Software development.

Error :- It is human action that produces the incorrect result that produces a fault.

Fault :- State of Software caused by an error.

Failure :- Deviation of the software from its expected result. It is an event.

Bug :- The presence of error at the time of execution of the Software.

Good Write

DATE: __/__/____
PAGE: _____

☆ Software Testing limitation:-

- you Can test a program Completely
- we Can only test against System requirements.
- Test results are used to make business decisions for released ~~dates~~ dates.
- Even if you do find the last bug. you'll never know it.
- you will run out of time before you run out of test cases.

☆ Important of Software Testing.

- Security
- Quality of the Product
- Satisfaction of the Customer
- Enhancing the development Process
- Determine the Performance of the Software.
- To avoid user detecting Problems.
- To find and Correct defects.

Good Write

DATE: __/__/____
PAGE: _____

☆ What Should the tester do?

i. Find the bugs.

ii. Find them early

iii. Make sure that they have been fixed.

☆ What Should not the tester do?

☆ Main Qualification for a good Software Tester.

- Be familier with the SDLC being used.
- Have excellent attention to details.
- Have Excellent Communication Skills (oral and written)
- Be able to run meeting and keep them focused.
- Patient, Mature and diplomatic.
- Able to prioritize things and take critical decisions.

Good Write

DATE: __/__/____
PAGE: _____

☆ Quality Control Life Cycle Through the SDLC:-

Testing Life cycle main activities.

1. Test Planning.
2. Test Analysis and Design.
3. Test Case Preparation.
4. Test Execution
5. Test Reporting and Monitoring.
6. Releasing Reporting.
7. User Acceptance Test Execution.

Good Write

DATE: ___/___/____
PAGE: _____

Software Bugs

Mistake. Metamorphism.

- Mistake
- Anomaly
- Fault
- Failure
- Error
- Exception.
- Crash
- Bug
- Defect
- Incident
- Side effect.

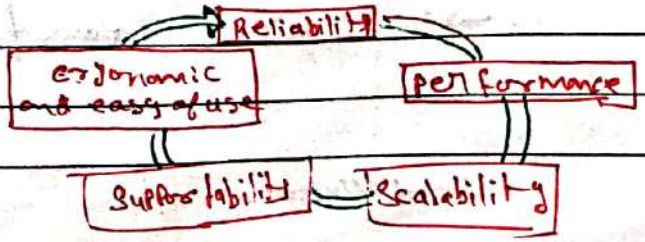
Reason why bugs get into Software.

- Last Minute Changes.
- Developed by humans
- Communication failure.
- Poor development techniques
- Lack of trained testing staff
- Buggy third-party instruments.
- Misapplication of technology.
- Unrealistic development time frame.

Software bug classification

- Priority / Urgency
- Severity / Impact
- Related module / Component.
- Probability / Visibility.
- Phase detected
- Phase injected

The 5 dimensions of quality.



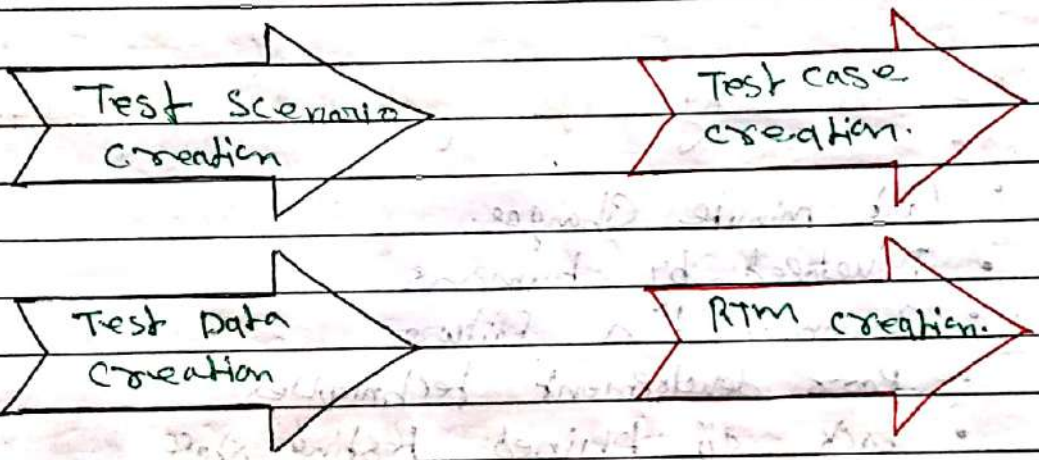
Good Write

DATE : __/__/____
PAGE : _____

Test Case development

Test Design

- Test Design aims to documenting "How" to do testing.
- Ensure that all requirements are met in the application.



Step for Test Design

- Test Design should start the moment the requirements have been approved and baselined.

- Test Design specification template.
  - → Features to be Tested
  - → Testing approach
  - → Test identification
  - → Test environment
  - → overall Pass / ~~Fail~~ fail criteria.

Good Write

DATE: __/__/____
PAGE: _____

★ Test Scenario :-

A Test Scenario is defined as any functionality that can be tested. It is also called Test Condition or Test Possibility. As a tester, you should put yourself in the end user's shoes and figure out real world scenarios and use cases of the Application under Test.

How do Identify Test Scenarios?

1. From use Case
2. Functionality breakdown.
3. Using State Transition technique.

Test Scenario Template :-

Test Scenario ID	Document index	module	Submodule	Scenario.
Ts-001	Section3, Page 4	Home Page	Login	User should be logged in only valid Password and user name.
Ts-002	Section3, Page 4	Home Page	Login	User with invalid username and Password. Should receive an error.
Ts-003	Section3, Page 4	Home Page	Login	User should have option of forgot Password
Good Write				

DATE: __/__/____
PAGE: _____

What is Tase Case.?

A test Case Psa document, which has a set of test data, Preconditions, expected results and Actual results, developed for a ~~Post~~ Particular test Scenario in order to verify any requirement.

Five required elements of a Test Caso.

- ID → Unique identifier of $ a test Case.
- Objective /step/Test Data → what you need to Do
- ~~Actual~~ ~~Expected~~ result → what output actually application Provides after testing
- Expected result → what you are supposed to get From application.
- Status → Pass/Fail

Type of Test Case :-

- Test Case categories :-
  1. Positive test case
  11. Negative test Case.

Good Write

DATE: __/__/____
PAGE: _____

- Test Case Types Comes under these Categories.

* Functionaly Test Case
* Performance Test Case
* Security Test case
* Integration Test Case.
* Data base Test Case.
* Usability Test Case.
* Acceptance Test Case.
* Happy Path Test Caso.

Positive Test Cases:

→ Performed on the System by providing the valid data as Input.

→ Checks whether and application behaves as expected with the positive Input.

Enter only Number
9999
Positive Testing.

Negative Test Cases:-

→ Performed on the System by providing invalid data as input.

→ Check whether an application does not work behaves as expected with negative input.

Enter only Number.
abcdef

Good Write

DATE: __/__/____
PAGE: _____

★ Test Case

* Input

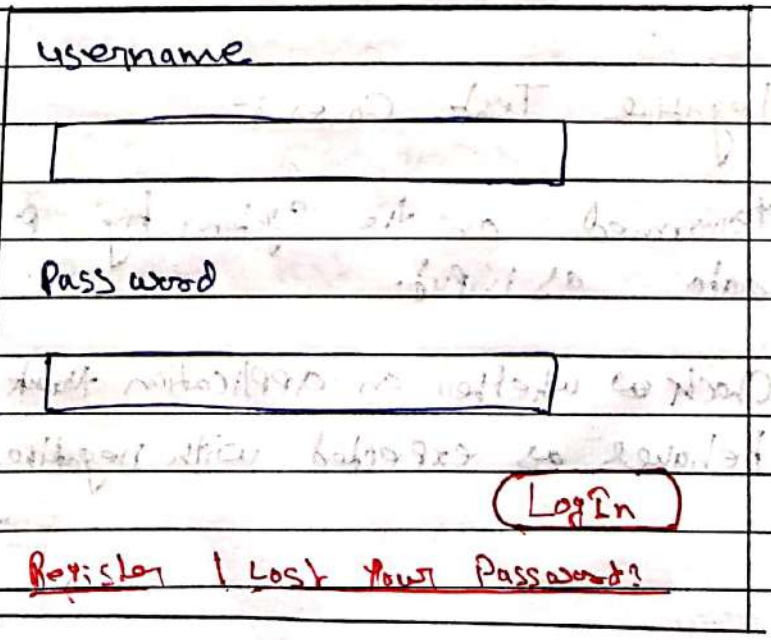
- Through the UI
- from interfacing Systems or device
- files
- Database
- State
- Environment

* Output

- To UI
- To interfacing Systems or device.
- files
- Database
- State
- Response Time.

★ Test Case of Login module.

- Check login box below.



Good Write

DATE: __/__/____
PAGE: _____

Testing metho

Introduction :-

- At first Stage :- Testing is the Process of executing a program with the intent of finding errors.
- At last Stage :- Testing is the process of demonstrating that errors are not Present. (after the Completion of all bugs).
- When you test a program, you want to add some value to it. Adding value through testing means raising the quality or reliability of the Program.

Misunderstandings about testing

- Testing is debugging.
- Testing is not the Job of a Progammer
- if programmers were more careful, testing would be unnecessary.
- Testing activity start only after the Coding is Complete.
- Testing never ends.
- Testing is not a creative task.

Good Write

DATE: __/__/____
PAGE: _____

Q1.

a.

Fault	Failure
i. Fault is error or mistake	i. Failure is the lack of success.
ii. Comparatively smaller	ii. Comparatively bigger
iii. Can be caused by ignorance, carelessness etc.	iii. Can be caused by fault.
iv. A Fault is a defect within the system.	iv. A failure occurs when the system fails to perform it required function.

b. • A bug report "should" explain how exactly the product is broken".

• A bug report should follow this simple formula:

"This is ~~not~~ what we have, this is what we should have instead, so fix it.

• Tester report bugs to programmers.

Good Write

DATE: __/__/____
PAGE: _____

- If the report is not clear and ~~understandable~~ understandable, the bug will not get fixed.

- Problem report forms are commonly used.

Q2 Verification and Validation

Advantage of Verification

- Verification help in lowering down the count of the defect in the laster stage of development

- Verifying the product of the starting phase of development will help in understanding the Product in a better way.

- It reduces the changes of failures in the software application or Product

- It help in building the Product as per the customer specifications and needs.

Good Write

DATE: __/__/____
PAGE: _____

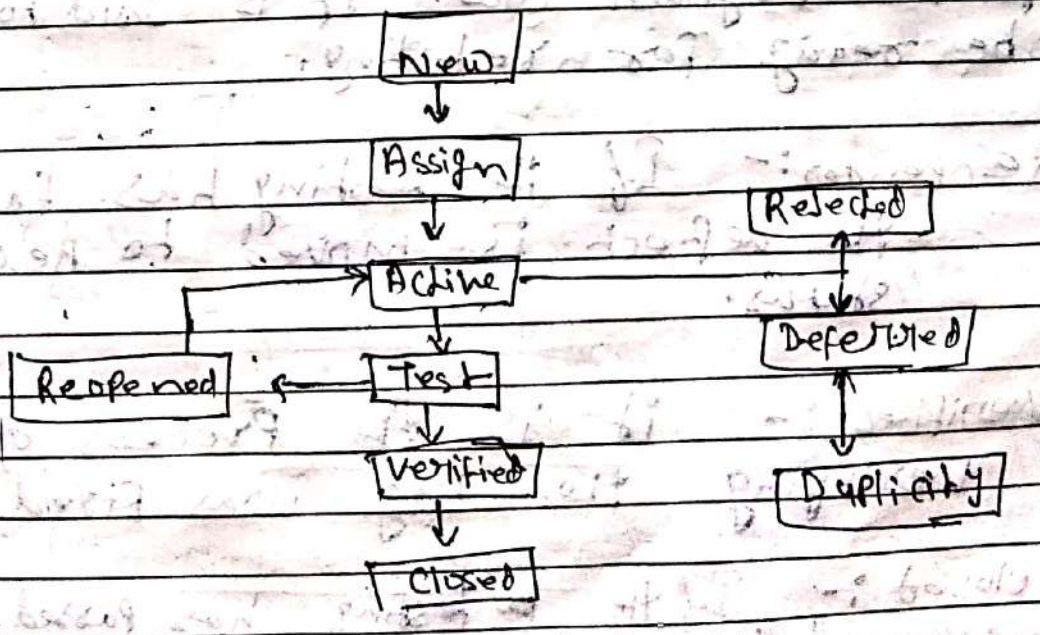
Advantage of validation:

- The defects missed during the validation process can be detected during the verification process.
- The validation is performed during the testing such as stress testing, Compatibility testing, system testing, etc.
- Validation assists is developing the right product according to the requirement of the customers.

Q4 ## Bug Life Cycle:-

A bug life cycle illustrates the journey of bug from the time it is created to the time it is fixed and closed.

A generic bug life :-



DATE: __/__/____
PAGE: _____

• New:- When defect is posted, the defaut Status is New.

• Assign:- Once the bug is posted by the tester, the lead of tester approves the bug and assign the bug to the developer team.

• Active:- This is the stage where the bug undergoes through analysis by the development team.

• Rejected:- Many a times, a flaw sent to the development team is so innocuous, that it is not even considered a bug and is rejected.

• Test:- After the development team has rectified the flaw with modification in the form of necessary changed to the Program code. It is said to be ready for retesting.

• Reopender:- If the testing has failed, the defect is moved to Reopen status.

• Verified:- it is the Process of verifying the bug was fixed or not.

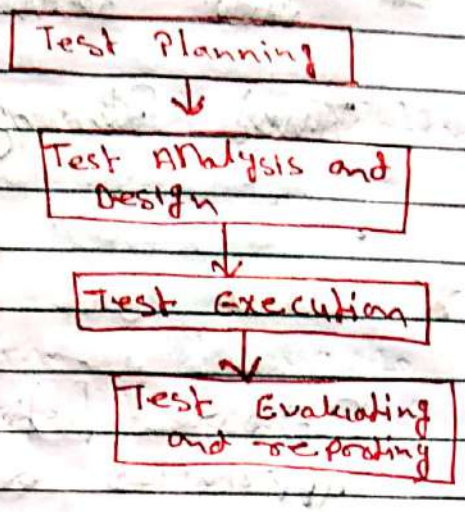
• Closed:- If the testing has passed, the defect is moved to 'closed' status

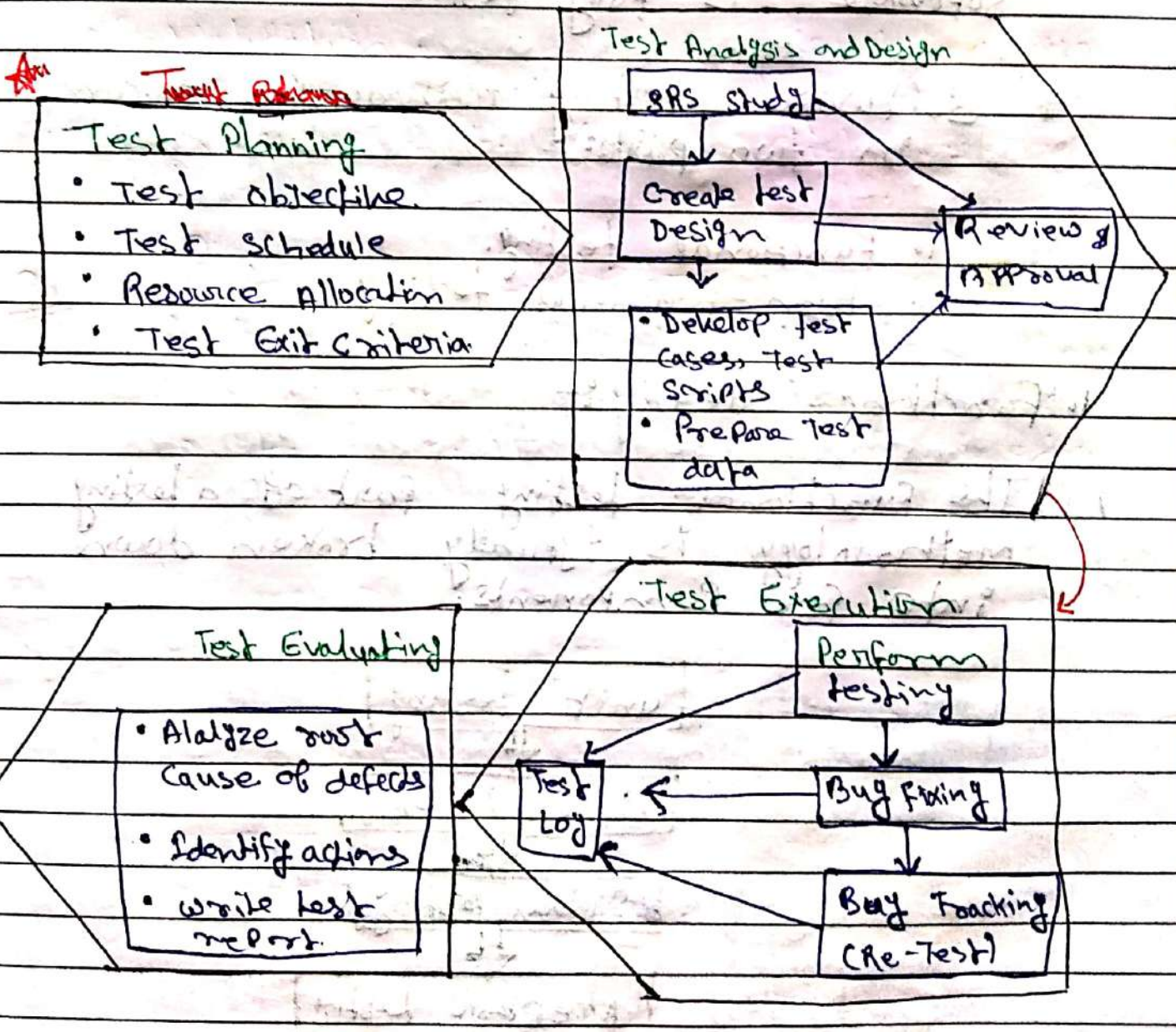
Good Write

DATE : __/__/____
PAGE : _____

☆ Testing Process :-

We can divide the activities within the test process into the following basic steps:-





Good Write

DATE: __/__/____
PAGE: _____

Test Planning:-

Test Planning has planning with.

What are Software Testing Methodologies.

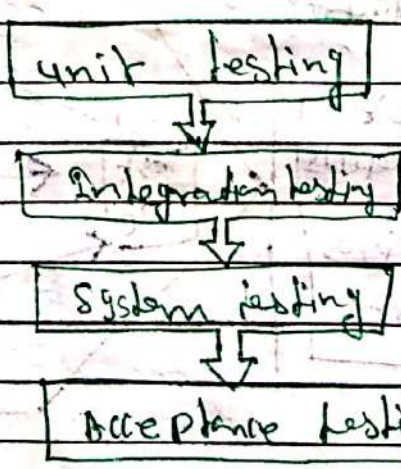
Software testing Methodologies are the different approaches and ways of ensuring that software is fully tested.

Software testing Methodologies encompass in two parts:

1. Functional testing.
2. Non-functional testing.

1. Functional Testing:-

The functional testing part of a testing Methodology is typically broken down into four components:



Good Write

DATE : __/__/____
PAGE : _____

- Unit testing :-

Purpose :- To verify that the Component / module functions work properly.

Check :-
- Internal data structure.
- Logic
- boundary condition for Input/output data.

Method :- White box testing.

Done by :- Developer

- Integration testing :-

Purpose :- To verify that Modules/ Components which have been successfully unit tested when integrated together to perform specific tasks and activities work properly.

This testing is usually done with a Combination of automated functional tests and manual testing.

Method :- Black box testing.

Done by :- Independent test team.

DATE: __/__/____
PAGE: _____

- System testing :-

Purpose :- Verifies that all system element work properly and that overall system function and performance has be achieved.

This test is carried out by interfacing the hardware and software components of the entire system.

Method :- Black box testing.

Done by :- Independent Test Team.

- Acceptance Testing :-

Purpose :- To ensure that the software that has been developed operates as expected at meets all user requirement.

There are two types of acceptance testing.

- Alpha Testing :- It is carried out by the members of the development team, knows as internal acceptance testing.
- Beta Testing :- it is carried out by the customer, known as external acceptance testing.

Good Write

Method :- Black box testing.

DATE: __/__/____
PAGE: _____

★ What is Regression Testing?

Regression Testing is type of testing carried out to ensure that changes made in the fixes or any enhancement changes are not impacting the previously working functionality.

Regression Testing is required when there is a

- Change in requirement and code is modified according to the new requirement.
- New Feature is added to the Software
- Defect fixing
- Performance issue fix.

Type of Regression Testing Techniques:-

1. Retest All :- It is of the Method of this Testing in which all tests in an ~~exsi~~ existing test bucket or suite re-executed. It is costly as it requires enormous time as well as resources.

Good Write

DATE: __/__/____
PAGE: _____

11. Regression Test Selection :—

Instead of re-executing an entire test suite, it is better to select a part of the test suite to run -

Test chosen Case categorized as

- Reusable Test Case.
- Obsolete Test Case.

Reusable test cases used in succeeding regression cycles. Obsolete Test cases not used in succeeding cycles.

Good Write

DATE: __/__/____
PAGE: _____

★ Black Box testing :-

This Technique Considers only the Functional requirement of the software. and also known as Functional testing.

These are of 4 type. :-

I. Boundary Value Analysis :-

II. Equivalance Class Partitioning :-

III. State table testing

IV. ~~Desi~~ Decision table testing.

This testing helps us to find the error in the following categories.

I. To Test the modules independently.

II. To Test the functional validity of software so that incorrect or missing functions can be recognise.

III. To look for interface errors.

IV. To ~~check~~ Test the System behaviour and Check its performance.

V. To Test the maximum load on the System.

Good Write

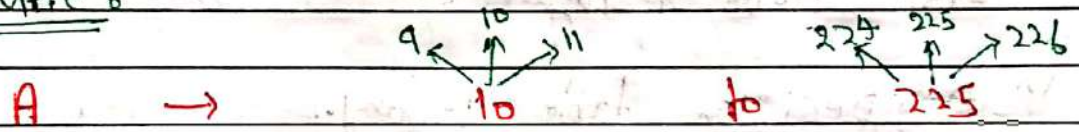
DATE: __/__/____
PAGE: _____

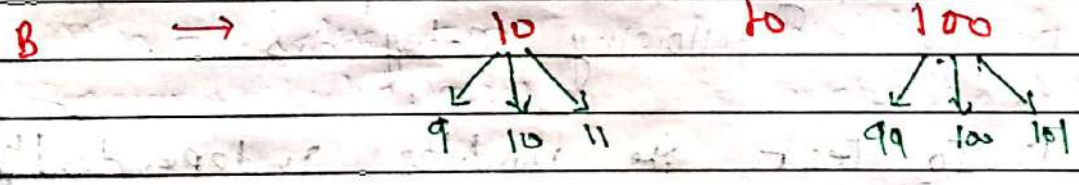
1. Boundary Value Analysis (BVA) :⇒ This is an effective Test Case design requires Test Cases to be design such that the maximize the probability of finding error.

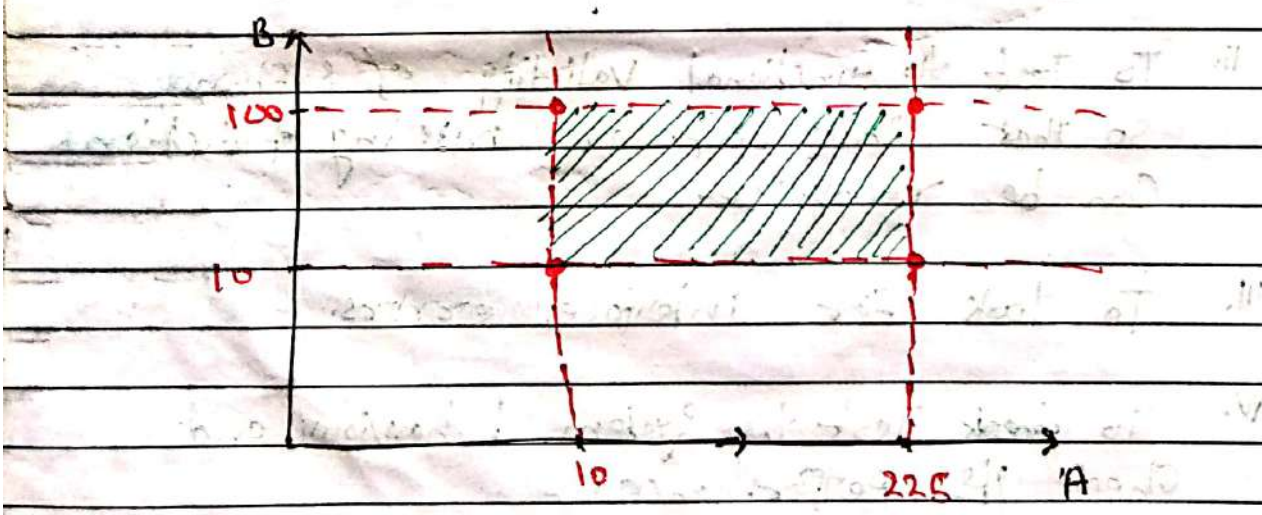
The technique to cover the bugs at the boundary of input value.

(boundary means Max or min value taken by input domen.)

Example :-







Good Write

DATE: __/__/____
PAGE: _____

☆1. Boundary Value Checking:-

In this method the test Case are designed by holding one variable and other variable at their normal value ~~value just above the minimum values~~ into the input domain.

a. Minimum Value (min) ~~into the input domain~~

b. Value just above the minimum value (min+)

c. maximum Value (MAX)

d. Value just below the maximum value (MAX-)

To check test Case. How many are possible.

$4n+1$

2. Robustness Testing method :- (6n+1)

e. A value just greater than max value (max+)

f. A value just less than min value (min-)

3. Worst Case Testing method. (5^n)

Good Write

DATE: __/__/____
PAGE: _____

Example

A → [1 to 100]

1. By BVC (4n+1) ⇒ 4×1 +1 = 5

Test Cases	Expected output
Min Value = 1	not a prime number
Min + Value = 2	prime number
Max Value = 100	not a prime number
Max - Value = 99	not a prime number
Nominal Value = 53	Prime number

II. Robustiness test Cases:- (6n+1)

Min - Value = 0	invaled input
Max + Value = 101	invalid input.

III. Wrost Case:- (5^n)

Same as BVC

Good Write

DATE: __/__/____
PAGE: _____

Ex 2 $a = [1 — 10]$

$b = [1 — 5]$

Taste Cases :-		A	B
	Min	1	1
	mint	2	2
	max	10	5
	max-	9	4
	nomin	5	3

1. By BVC :- $(4n+1) = 4 \times 2 + 1 = 9$

Taste Case	A	B	Expected Answer.
1	1	3	1
2	2	3	8
3	10	3	1000
4	9	3	729
5	5	1	5
6	5	2	25
7	5	5	3125
8	5	4	625
9	5	3	125

1. By Robostic :- $(6n+1)$ $6 \times 2 + 1 = 13$

Taste Cas	A	B
10	0	3
11	11	3
12	15	0
13	15	0

Good Write

DATE: __/__/____
PAGE: _____

III. Worst Case :- (5^n) ⇒

Test Case	a	b	expected output
1	1	1	1
2	1	2	1
3	1	5	5
4	1	4	4
5	1	3	3
6	2	1	2
7	2	2	4
8	3	5	32
9	2	4	16
10	2	3	8
11	10	1	10
12	10	2	100
13	10	5	100000
14	10	4	10000
15	10	3	1000
16	9	1	9
17	9	2	81
18	9	5	59049
19	9	4	6561
20	9	3	729
21	5	1	5
22	5	2	25
23	5	5	3125
24	5	4	625
Good Write!		3	125

DATE: ___/___/____
PAGE: _____

Q. A → [1-50], B → [1-50], C → [1-50]

	A	B	C
min	1	1	1
min +	2	2	2
max	50	50	50
max -	49	49	49
nom	25	26	27

Find greatest of the three num

Using ~~BVC~~ Using BVC :- (4n+1) 4×3+1 ⇒ 13

Test Case	A	B	C	EXPOUT
1	1	26	27	27
2	2	26	27	27
3	50	26	27	50
4	49	26	27	49
5	25	1	27	27
6	25	26	27	27
7	25	50	27	50
8	25	49	27	49
9	25	26	1	26
10	25	26	2	26
11	25	26	50	50
12	25	26	49	49
13	25	26	25	26

Good Write

DATE: __/__/____
PAGE: _____

2. Equivalence class partition

A, B, C → [1, 50]

Classes

I_1	A →	$1 \le A \le 50$	valid
I_2	B →	$1 \le B \le 50$	
I_3	C →	$1 \le C \le 50$	
I_4	A →	$A < 1$	invalid
I_5	A →	$A > 50$	
I_6	B →	$B < 1$	
I_7	B →	$B > 50$	
I_8	C →	$C < 1$	
I_9	C →	$C > 50$	

	A	B	C	Exp	Classes Covered
1	13	20	15	Valid	I_1, I_2, I_3
2	0	20	15	Invalid	I_4
3	51	20	15	Invalid	I_5
4	13	0	15	Invalid	I_6
5	13	51	15	Invalid	I_7
6	13	20	0	Invalid	I_8
7	13	20	0	Invalid	I_9

Good Write

DATE: ___/___/____
PAGE: _____

Q. Date - 1 to 31
Month - 1 - 12
Year - 1990 - 2025

I_1 Date → $1 \leq Date \leq 31$
I_2 month → $1 \leq month \leq 12$
I_3 year → $1990 \leq year \leq 2025$
} Valid

I_4 Date → Date < 1
I_5 Date → Date > 31
I_6 Month → Month < 1
I_7 month → Month > 12
I_8 year → Year < 1990
I_9 year → Year > 2025
} Invalid

Date	month	Year	Expansion
12	10	2010	13-10-2010
0	10	2010	Invalid
32	10	2010	Invalid
12	0	2010	Invalid
12	13	2010	
12	10	1989	Invalid
12	10	2026	Invalid.

Good Write

DATE: __/__/____
PAGE: _____

4. Error Guessing :- This method is use when all other method fails, Some times it is use to test some special cases. According to this method error or bugs can be gaised which donot fit feet in NDA of the earlier defined. It is a very practical case in which tester uses his intuition and makes a guess about where the bug can be.

5. State table based testing :-

State table based testing means

Tables is useful for representing and Document manage relating to test cases

State transition digram and state table are use in this type of testing.

1. New State :-

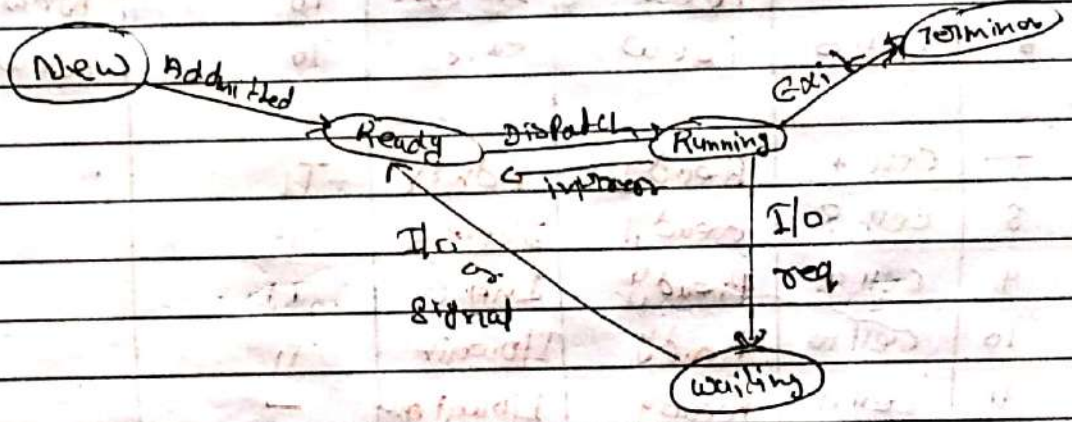
2. Ready :- When the task is waiting in the Ready queue for its term.

DATE: __/__/____
PAGE: _____

3. Running :- When instruction of the task is are being executed by CPU.

4. Waiting :- When the task is waiting for an input, output, reason or a signal

5. Terminated :- The Task has finish execution



State Table :- Each row of the table corresspond to the state each column corresspond to in Input condition.

	Admit	dispatch	Interrupt	I/O wait		
New	Ready/I	New	New	New	New	New
Ready	Ready/I	Ready T2	Ready T1	Ready T1	[illegible]	[illegible]
Running	Ready/T2	" T2	" T3	Waiting	"	
Waiting						

Good Write

DATE: __/__/____
PAGE: _____

White Box Testing Technique.

White Box testing is also known as glass box testing as every thing that is required to implement this s/w is visuable it is also known as structual and develement testing the intension of the testing is to test the entire logic of the Code.

It is of 3 type

- Logic Coverage criteria.
- Bassis path Testing.
- graph metrices.

Needs of white box testing.

- This testing is use for testing the module at very inisitial stage. after then Black box testing is required.

This is not an alltermative but an essentiond stage.

Good Write

DATE: __/__/____
PAGE: _____

- Their are Many types of Bugs. which Can be covered by white box testing only.
- Errors which have come from the design Phase will also be reflected in the Code So, White box testing is Must.

1. Logic Coverage criteria:
   - Statement Coverage
   - Decision coverage
   - Condition Coverage

ex:-

```
Scanf ("%d", &x);
Sconf ("%d", &y);
while (x != y)
{
  if (x > y)
    x = x - y;
  else
    y = y - x;
}
Printf ("x=", x);
Printf ("y=", y);
```

Test Case (1)
x = y = x

Test Case (2)
x > y
x < y } Boundary

Test Case (3)
x > y

Test Case (4)
x < y

DATE: __/__/____
PAGE: _____

1. Statement Coverage:

Statement Coverage:-

It is assumed that all the Module are executed once every bugs will be notify.

if we want to cover every statement in the given code. then the following test cases must be design.

Test case (1)

$x = y = n$:. when n is numbers

Test case (2)

$x = n$

$y = n'$

n and n' are different number.

Test case (3)

$x > y$

Test Case (4)

$x < y$

We Can See that test case 3 and 4 are sufficient to execute all the statement in the code but we execute only test case only 3 and 4 then condition and path in test case 1 will never be tested and error will go undetected. So Statement Coverage is nessocery but not sufficient.

Good Write

DATE : ___/___/____
PAGE : _____

II. Decision Coverage :-

Test Case 1 → $x = y$
2 → $x \neq y$
3 → $x > y$
4 → $x < y$

III. Condition Coverage :-

while ((I ≤ 5) && (J < count))

This states that each condition in a decision takes on all possible outcomes atlist once.

Test case 1 → I ≤ 5, J < count (True)

Test case 2 → I < 5, J > count (false)

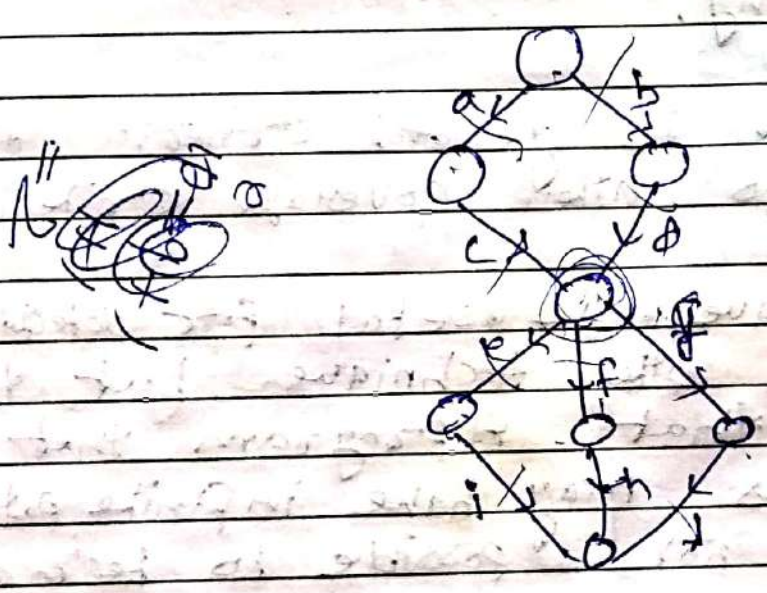
Good Write

DATE: __/__/____
PAGE: _____

3 Basis Path Testing:-

Cyclomatic Complexity:-

it is used in graph Problem which scan all the node present in the graph.
Root node to leaf node.



acei	bdei
acfh	bdfh
acgj	bdgj

edge

$V(G) = e - n + 1$ → node.

$10 - 8 + 1$

$\Rightarrow 2 + 1$

$= 3$

Good Write

DATE: __/__/____
PAGE: _____

This Technique is based on the control structure of the Program.

On the bases of Control structure a flow graph is prepaired and all the possible path can be covered and executed during testing.

Path Coverage is general criteria as Compaired to other coverage criteria.

This Technique is useful for detecting more error other techniaue but the problem is that A program that Contain loops may have infinite path so it is not possible to test all path.

Control flow Graph :-

i. NODE :- Procedural Statement →

ii. Edges / Links :- flow of the graph. →

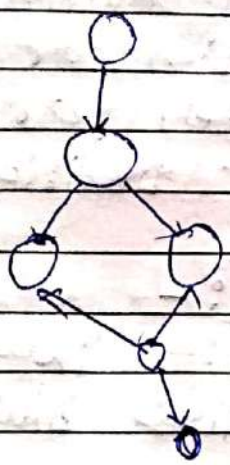
iii. Decision node :- node with more than 1 arrow leaving it is called Desision.

iv. Junction node :- node with more than 1 arrow entering it called Junction node.

Good Write

DATE: ___/___/____
PAGE: _____

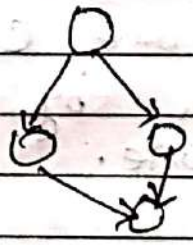
v. Regions :- Area ~~Surround~~ bounded by edges and node are called regions.



Flow graph notation for -

1. Sequence

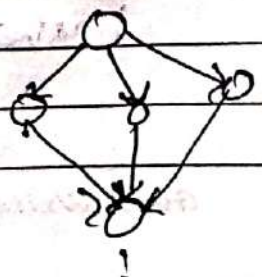
2. If - then - else



3. Do - while.

4. while - DO

5. Switch case



Good Write

DATE: __/__/____
PAGE: _____

Some important Terms:-

(i) Path:- A Path in a program is a sequence of instruction that starts at an entry, Decision, or Junction and ends at another or possibly the same Junction or Decision node or exit.

(ii) ~~Statement~~:-

(iii) Segment:- This a simple process that lies blw two Nodes

(iv) Path segment:- A Path segment is the Consiqutive link that belong to some Path.

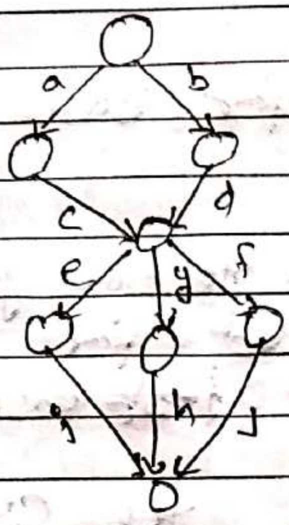
(v) Length of a path:- the length of a path is measured by the number of link.
Or we can measure the link of the path by counting the no. of node traverse.

vi. Independent Path:- is ~~a~~ Path of any path through the graph that introduces atlist one new set of processing statement or new Conditions.

An independent path must move along at list one edge that had not been traverse before the Path is defined.

Good Write

DATE: __/__/__
PAGE: ____



no of independ path =

no of Regions =

↳ cyclomatic complexity ज्ञात करने के लिए .

Good Write

DATE: __/__/____
PAGE: _____

☆ Guidelines for Basis Path Testing:-

Step 1:- Draw the flow graph using the code provided for which we have to write test cases.

Stp 2:- Determine the Cyclomatic Complexity of the flow graph.

Stp 3:- Cyclomatic Complexity provide the number of independent path.

Stp 4:- The basis set is in fact the base designing the test cases based on every independent path choose the data such that this path is executed.

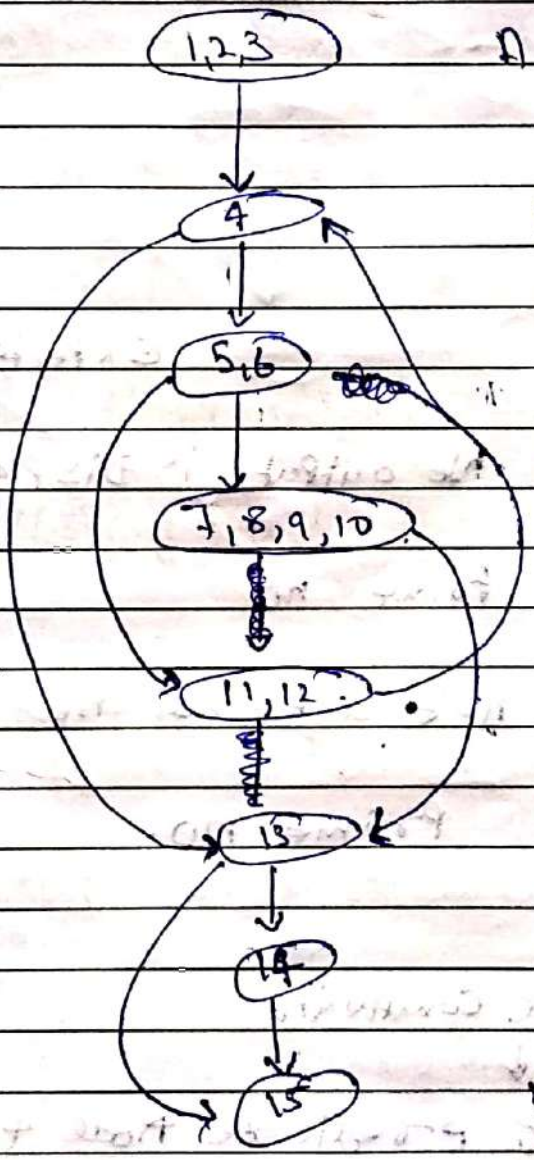
Example

```
main()
{
int number, index;
printf("Enter a number");
scanf("%d", &number);
index = 2;
while (index <= number - 1)
{
if (number % index == 0)
{
printf("not a prime no");
break;
}
```

DATE: ___/___/____
PAGE: _____

```
11  index ++;
12  {
13      if (index == number)
14          printf("Prime no");
15  }
```

Step 1



Step 2

Cyclomatic complexity

Total 4 Regions.

Good Write

DATE: __/__/____
PAGE: _____

Step 3

Independent Path:-

i. A-B-F-H
ii. A-B-F-G-H
iii. A-B-C-E-B-F-G-H
iv. A-B-C-D-F-H

Step 4

Test Case

Input	Expected Results.
1	No output is Display.
2	Prime no
3	not a prime no
4	Prime no.

* Cyclomatic Complexity
↓
no of predicate node + 1

Predicate :- जहाँ पर Point से देखने से ज्यादा Node जाते है, use Predicate node कहते है।

B, C, F

= 3 + 1 = 4

Good Write

DATE: __/__/____
PAGE: _____

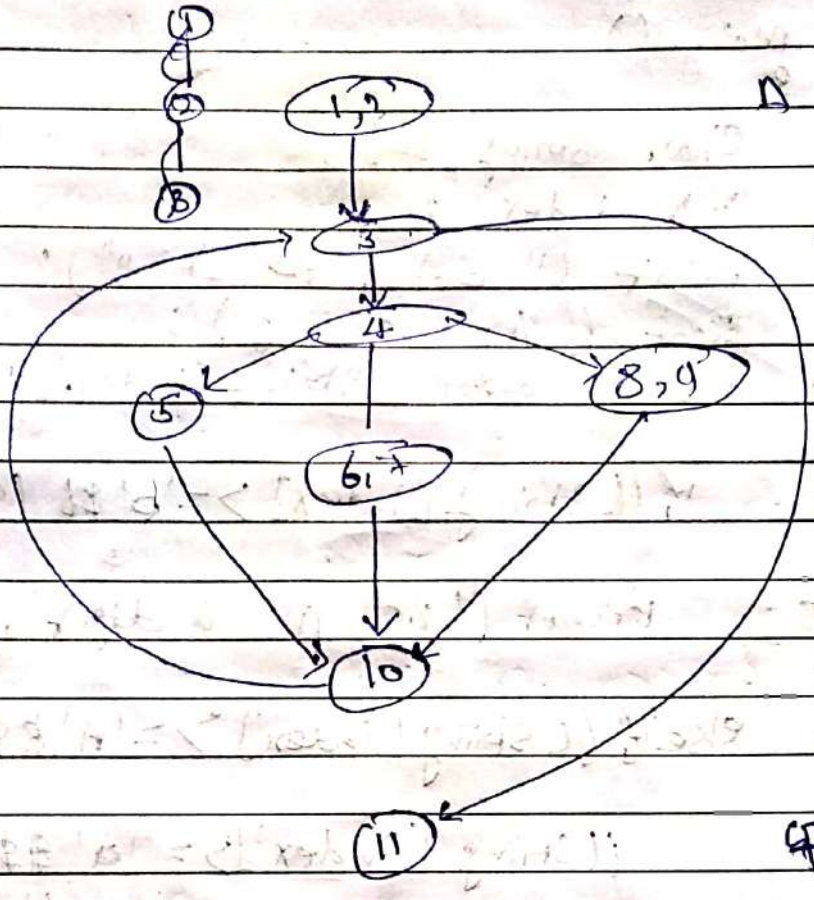
Q.

```
Main ()
{
  Char String[80];
  int index;
1 Print ("Enter the String");
2 scanf ("%s", string);
3 for (index=0, String[index] != '\0', ++index){
4   if ((String[index] >='0' && (String[index]<='9'
5     PrintF ("%c is a digit", String[index]);
6   elseif ((String[index] >='A' && String[index]
                                  <'Z')) ||
      ((String[index] >= 'a' && (String[index]
                                  <'z')))
7.    Printf("%c is an alphabat", String[index]);
8   else
9     Printf ("%c is a special Character", String[index])
10    }
11  }
```

Good Write

DATE: ___/___/____
PAGE: _____

Step 1



Step 2 — no. of Predicate node = no. of links outgoing of main node - 1

2 [illegible]

Cyclomatic complexity :- 8 - 1 + 2 + 1 = 4

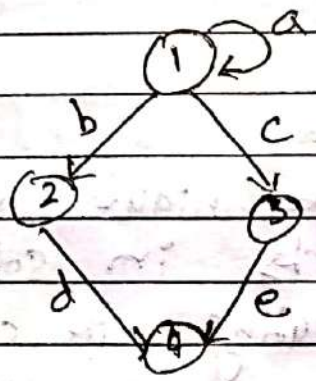
Cyclom Comp - 4

Step 3 — Independent Paths

Good Write

DATE: ___/___/____
PAGE: _____

Graph Matrix :-



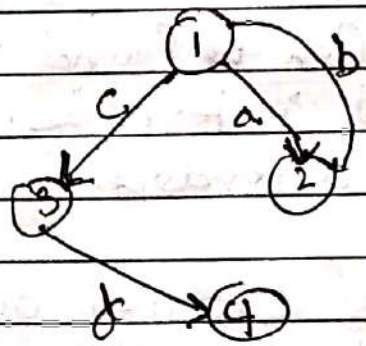
	1	2	3	4
1	a	b	c	
2				d
3				e
4				

Connection matrix

	1	2	3	4	
1	1	1	1		3-1=2
2			1		1-1=0
3			1		1-1=0
4					

2+0+0+1 ⇒ 3 cyclomatic complex

Q.



	1	2	3	4
1		ab	c	
2				
3				d
4				

	1	2	3	4	
1					
2		1-1=0	1		2-1=1
3				1	1-1=0
4					

1+1=2

Good Write

DATE: __/__/____
PAGE: _____

☆ Static Testing :-

i. Software Inspection.
ii. walkthroughs.
iii. Technical Review.

It is a s/w testing technique which is used to check defects in software application without executing the code. Static testing is done to avoid errors at an early stage of development as it is easier to identify the errors and solve the errors.

i. Software Inspection :- This process doesnot require Executable code. on test cases.
with Inspection Bugs Can be found out which are not included in test cases. and it is machine independent.

Inspection process is carried out by a group of Members which inspects the product at personal level after this they discuss the defect of the product observe in a formal. Meaning..

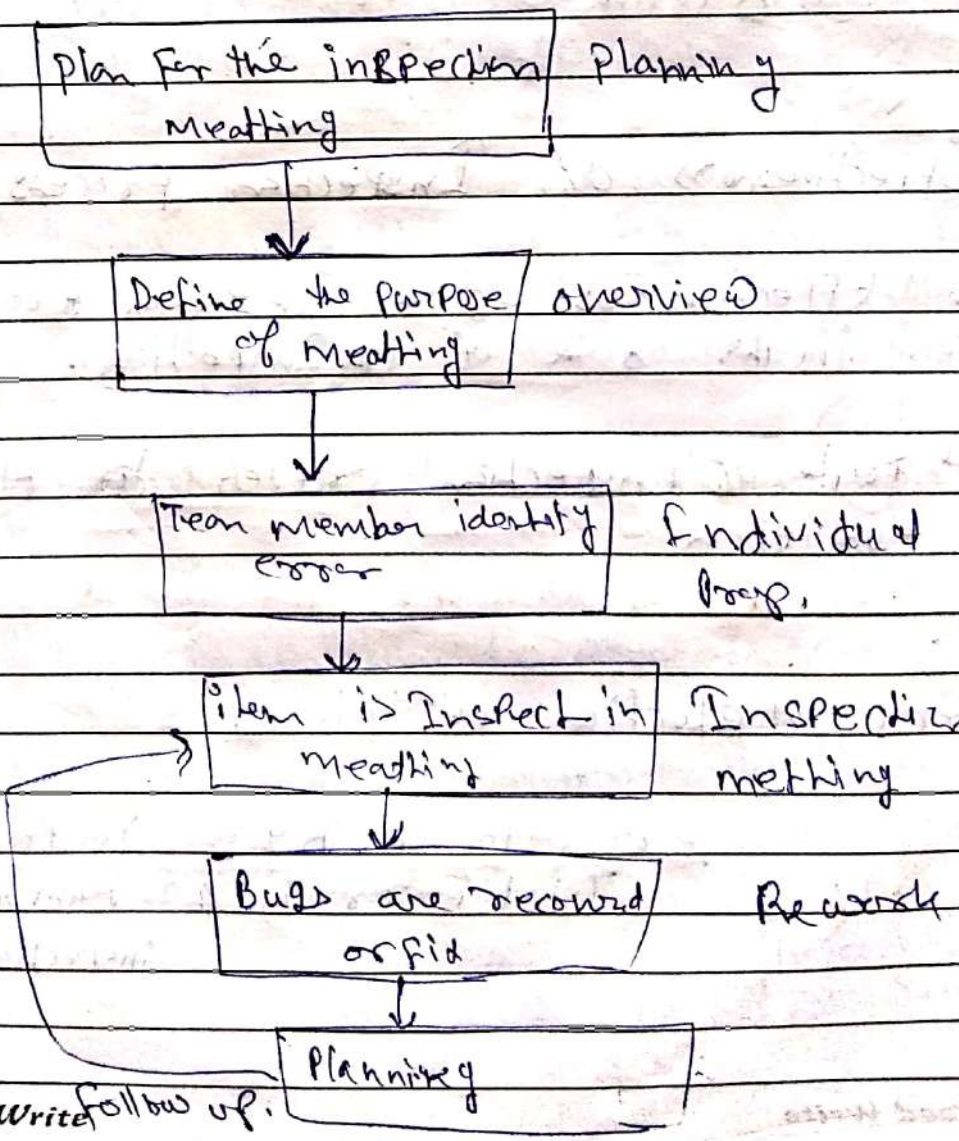
Good Write

DATE: __/__/____
PAGE: _____

Inspection Team :-

1. Author / Owner / Producer.
2. Inspect
3. Moderator
4. Recorder.

Inspection Process :-



Good Write

DATE: __/__/____
PAGE: _____

Benefits/Advantage of Inspection Process.

1. Bugs reduction:-
2. Bug prevention.
3. Increase Productivity
4. Quality improvement.
5. Process improvement.

Effectiveness of Inspection process.

The Effectiveness of the Inspection process lies in its rate of Inspection.

The rate of Inspection refer to How much evaluation

Error detection efficiency =

$$\frac{\text{Error found by an inspection}}{\text{Total error in the team before inspection}} \times 100$$

DATE: __/__/____
PAGE: _____

Structured walkthrough → less formal and less rigorous as compared to Inspection.

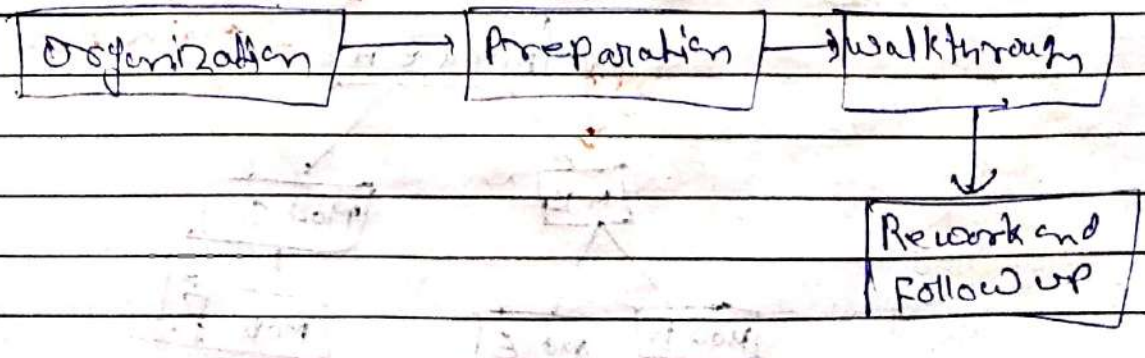
Walkthrough Teams :-

I. Coordinator.
II. Presenter / Developer.
III. Recorder.
IV. Tester
V. Maintenance, or etc. (feature maintain)
VI. Standard Bearer. (Laws maintain)
VII. User Representive.

Walkthrough is Different from inspection as inspection is a 6 Step regress and formal Process.

A walkthrough is less formal has few test

Steps to walkthrough process :-



Good Write

DATE : __/__/____
PAGE : _____

III. ★ Technical Review :-

↓

Management Team, Particular also

A review is similer to inspection or walkthrough accept that the review team also include Management.

It Consian that higher level technique.

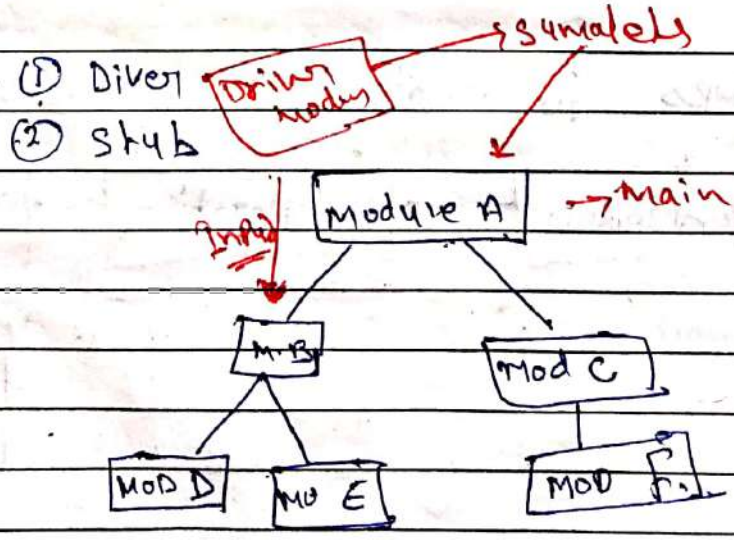
This technique focus on oversite more as compared to technical issue.

Output Document :-

- Defances
- other cone
- Error
- need

★ UNIT Testing

(1) Diver
(2) Stub



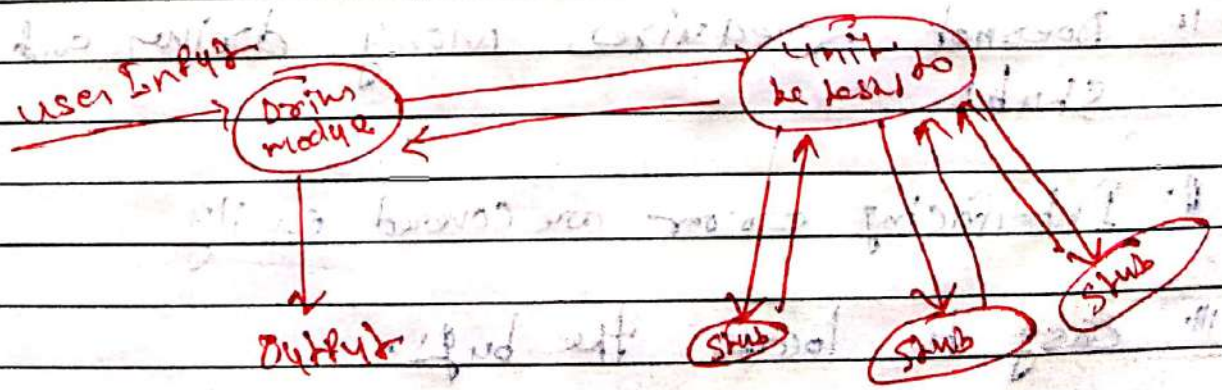
Good Write

DATE: __/__/____
PAGE: _____

Stubs: The module under testing may also called some other module which is not ready at the time of testing. Therefore these module lay to be ~~Sg~~ millard testing. In most cases ~~Dummy~~ DAMMY module instead of actual module are not ready ~~are Prepare~~.

Method of Creating Stubs and Driver.

- Stubs allow the programmer to call a method in the code being developed.
- By using stubs and Driver effectively we can cut our debugging and testing time.
- Stubs and Driver can also be an effective tool for demostrating progress in a business environment.



Good Write

DATE: __/__/____
PAGE: _____

Integration Testing :-

1. Decomposition based Integration
   - i. Incremental → Top Down / Bottom up
   - ii. non-Incremental (Big Bang Testing)

Why we not use non-Incremental :-

i. Big Bang require more works as more driver and stubs are required

ii. It is defecult to locate the error since the exect location of bugs can not be found easily.

iii. Actual modules are not interfaced directly.

Advantages of Incremental :-

i. Doesnot requires Many driver and Stubs

ii. Interfacing error are covered easily

iii. Easy to locate the bugs.

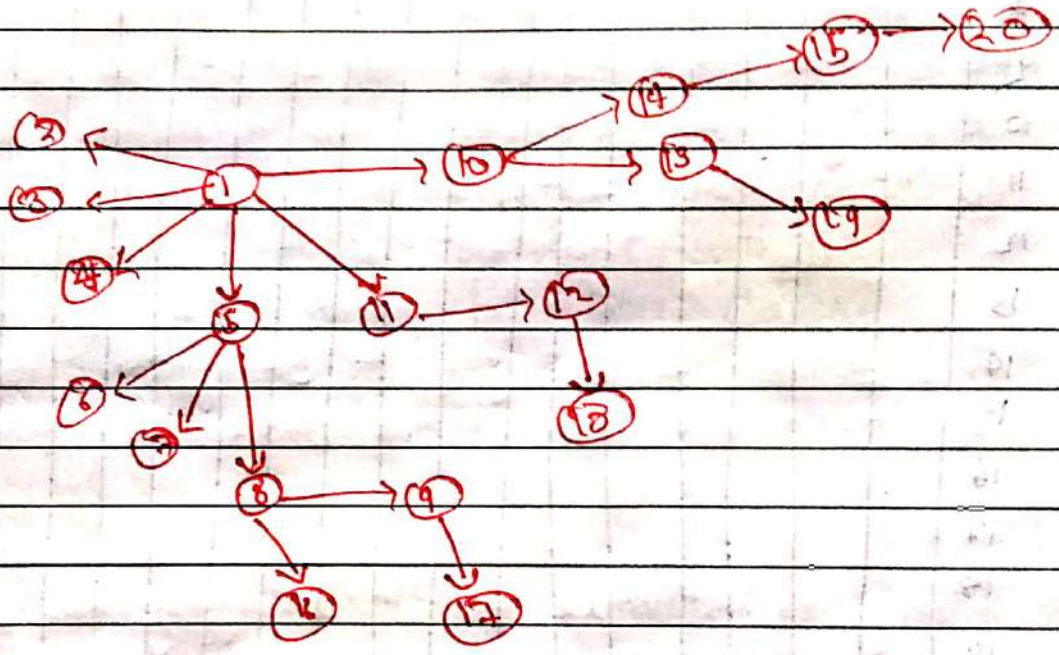
iv. Incremental testing is the More ~~thought~~ thorough process.

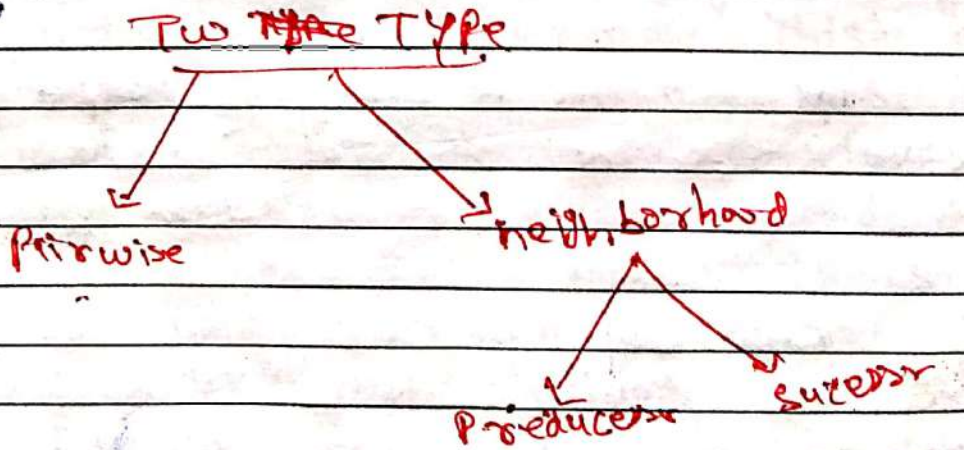
Good Write

DATE: __/__/____
PAGE: _____

Call graph Based Integration testing:

A Call graph is Directed graph in which nodes are either module or unit and directed edge from one node to another means one module is ~~called~~ call another module.





Test Case = Total node − Sink node

= 20 − 10

= 10

Good Write

SPP DATE: __/__/____
PAGE: _____

☆ Test Automation And bugging :-

Testing :-

Testing the Process of verifying and validating that a software or application is bug free, meets the technical requirement as guided by its design And development and meet the user requirement effectively and efficiently with handling all the exceptional and boundary cases.

Debugging :-

Debugging is the Process of fixing bug in the software. It can be defined as identifying, analyzing and removing errors. This activity begins after the software fails to execute properly and concludes by solving the problem and successfully testing the software. It is Considered to be an extremely Complex and tedious task because errors need to be resolved at all stage of debugging.

Good Write

DATE: __/__/____
PAGE: _____

Difference between testing and Debugging.

Testing	Debugging
I. Testing is the process to find bugs and errors	I. Debugging is the process to correct the bugs found during testing.
II. It is the process to identify the failure of implemented code.	II. It is the process to give the absolution to code failure.
III. Testing is the display of errors.	III. Debugging is a deductive process
IV. Testing is done by the tester.	IV. Debugging is done by either programmer or developer.
V. There is no need of design knowledge in the testing process.	V. Debugging can't be done without proper design knowledge.
VI. Testing can be manual or automated.	VI. Debugging is always manual. It can't be automated.

Good Write

DATE: __/__/____
PAGE: _____

Software test Automation :- It make use of specialized tools to control the execution of test and compares the actual result against the expected result. Usually, regression test, which are repetitive action, are automated.

Testing tool not only help us to perform regression test but also helps us to automate, data set up generation, product installation, GUI interaction, defect logging etc. Automation tools are used for both Functional and non-functional testing.

S/w Measurement

A measurement is a manifestation of the size, quantity, amount or dimension of a particular attributes of a product or process. Software measurement is a tangible impyle of a characteristic of a software product or the software process.

Good Write

DATE: __/__/____
PAGE: _____

Need of Software Measurement:-

Software is measured to:-

I. Create the quality of the current product or process.

II. Anticipate future qualities of the product or process.

III. Enhance the quality of a product or process.

IV. Regulate the state of the project in relation to ~~hg~~ budget and schedule.

Classification of Software Measurement:-

There are 2 types of SW measurement.

1. Direct Measurement:- In direct measurement the product, process or thing is measured directly using standard scale.

DATE: __/__/____
PAGE: _____

2. Indirect Measurement : In indirect measurement the quantity or quality to be measured is measured using related parameter ie. by use of reference.

Metrics :-

A Metrics is a measurement of the level that any impute belongs to a system product or process.

There Are 4 functions related to Software Metrics :-

i. Planning
ii. Organizing
iii. Controlling
iv. Improving.

DATE: __/__/____
PAGE: _____

Testing Metrics :-

Software Testing Metric is defined as a quantitative quantitative measure the helps to estimate the progress, quality, And health of a software testing effort.

Type of test Metrics :-

i. Process Metrics :- It Can be used to improve the process efficiency of the SDLC (Software Development life cycle)

ii. Product Metrics :- It deals with the quality of the Software product.

iii. Project Metrics :- It Can be use to measure the efficiency of a project team or any testing tools being used by the team member.

DATE: __/__/____
PAGE: _____

Type of Testing Tools:-

Tool Type	Used for	Used by
I. Test management tool	Test managing scheduling, defect logging, tracking and analysis.	Tester.
II. Configuration management tool	For implementation, execution, tracking change.	All teams member.
III. Static Analys Tools	Static testing	Developer.
IV. Test Execution Tools	Implementation, execution.	Tester.
V. Test Comparators.	Comparing expected and actual result	All Teams Member
VI. Coverage measurement tools.	Provides structual coverage	Developer.
VII. Incident management tools	For managing the test	Tester.

Good Write

DATE: __/__/____
PAGE: _____

Object Oriented Testing :-

⇒

Testing is a continuous activity during software development. In object oriented system testing encompasses three levels, namely, unit testing, subsystem testing, and system testing.

1. Use Case based testing :-

- They often use the language and terms of the business rather than technical terms, especially when the actor is a business user.
- They serve as the foundation for developing test cases mostly at the system and acceptance testing levels.
- Each use case must specify post condition that are observable results and a description of the final state of the system.

DATE: __/__/____
PAGE: _____

Characteristics of use Case Testing.

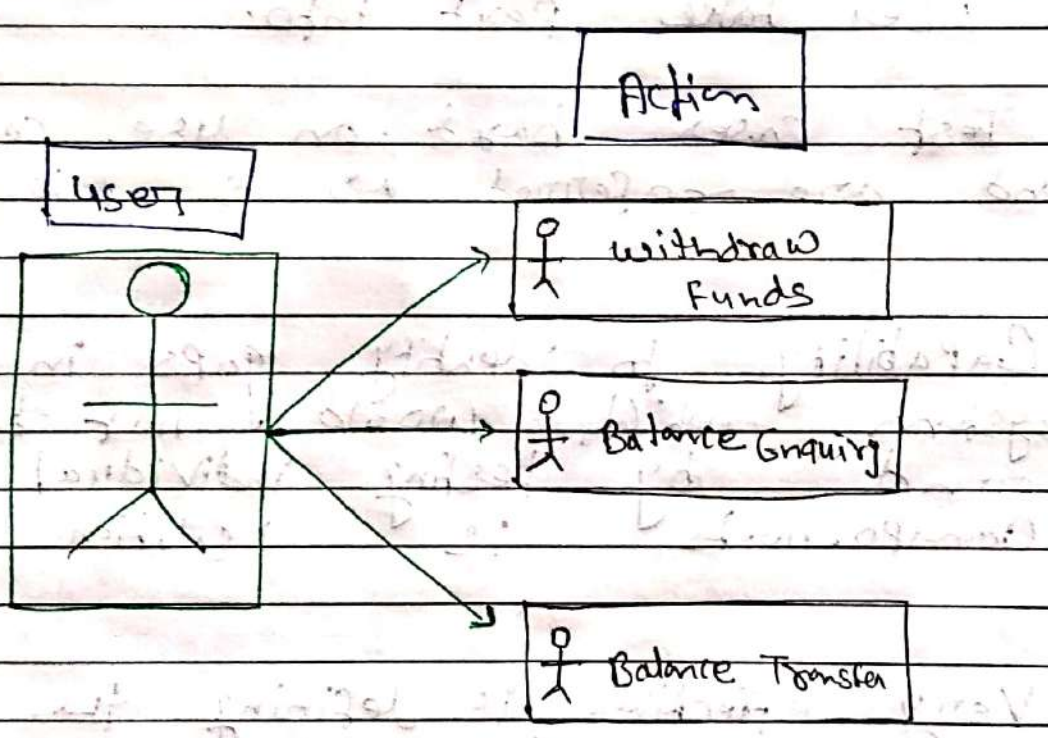
- Use Case capture the interactions between 'actors' and the system.
- 'Actors' represents user and their interactions that each user take part into.
- Test cases based on use cases and are referred as scenarios.
- Capability to identify gaps in the system which would not be found by testing individual components is isolation.
- Very effective is defining the scope of acceptance tests.

Good Write

DATE: __/__/____
PAGE: _____

Example

Below example clearly shows the interaction b/w user and possible Action.



Good Write