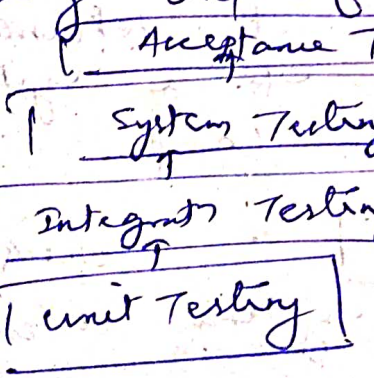
Short notes

a) System Testing :-

(i) It is a type of software testing that evaluates the overall functionality and performance of a complete and fully integrated software soln

(ii) This type of testing is performed after the integration testing and before the acceptance testing.



(iii) focuses on both functional and non-functional aspects.

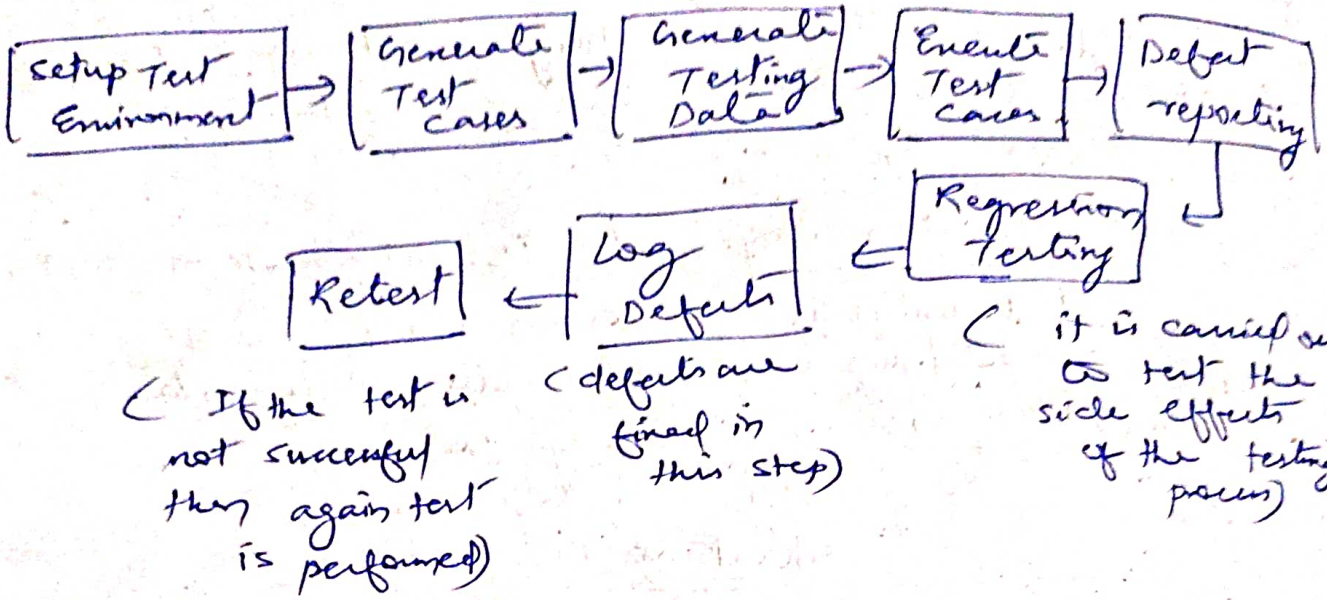
(iv) identifies defects related to integration & system components & interactions with external systems.

(v) cons :- time consuming.

→ difficult to isolate issues to specific component.

→ Finding and fixing defects at this stage can be costly compared to earlier testing phases.

System Testing process



Types of System Testing :-

1. **Performance Testing** :- This is carried out to test the speed, scalability, stability and reliability of the s/w product or applicat.
2. **Load Testing** :- it determines the behaviour of a system or s/w product under extreme load.
3. **Stress Testing** :- check the robustness of the system under the varying loads.
4. **Scalability Testing** :- used to check the performance of the system in terms of its capability to scale up or scale down the no of user request load.

Common Tools :-

JMeter, LoadRunner, Selenium, microsoft Test manager, Apache JSer

Advantages of System Testing :-

(i) Testers do not require more knowledge of programming to carry out this testing.

(ii) It will test the entire product or s/w so that we will easily detect the errors or defects which cannot be identified during the unit & integration testing.

(iii) The testing env is similar to real time product / Business env.

(iv) After this testing, the product will almost cover all the possible bugs or errors & hence the development team will confidently go ahead with acceptance testing.

(v) increases user confidence & reduce risk.

(vi) improves system reliability & quality.

Disadvantages :-

(1) Time consuming & Expensive.

(2) Requires adequate resources & infrastructure.

(3) complex & challenging for large & complex systems.

(4) requires proper planning, coordination & execution.

(5) requires expertise.

(6) may require multiple test cycles to achieve desired results.

b) Integration Testing :— it is the process of testing the interface btn 2 software units or modules.

(i) once all the modules have been unit-tested integration testing is performed.

(iii) it focuses on exchanging the data btn different components or modules.

(iv) it is performed after unit testing & before system testing.

⇒ 4 types -

(1) Big-Bang IT :— simplest integration testing approach.

→ modules of the system are simply put together and tested at once.

↓ only applicable to small systems.

↓ If an error is found during the IT, it is very difficult to localize the error.

advantages :—

① convenient for small systems.
② simple & straightforward approach.
③ can be completed quickly.
④ doesn't require a lot of planning or coordination.

Disadvantages :—

① not good for big projects.
② It can result in long and complex debugging and troubleshooting efforts.
③ This can lead to decreased efficiency & productivity.
④ it can lead to system failure and decreased user satisfaction.

(2) **Bottom-Up Integration Testing** :- Each module at lower levels are tested with higher modules until all modules are tested.

Advantages

(1) Best for applications that uses bottom up design approach.
(2) easy to observe the test results.
(3) easy to create test conditions.

Disadvantages

(1) There is no working model can be represented.
(2) complexity occurs when the system is made up of large no of small subsystems.
(3) Driver modules must be produced.

(3) **Top-Down IT** :- Testing takes place from top to bottom. First high level modules are tested & then low-level modules and finally integrating the low level modules to a high level to ensure the system is working as intended.

Advantages :-

- separately debugged module.
- Few or no drivers needed.
- more stable and accurate at the aggregate level.
- Easier isolation of interface error.
- design defect can be found in the early stage.

disadvantages :-

(1) Need many stubs.
(2) difficult to stub design.
(3) difficult to observe the test o/p.
(4) modules at lower level are tested inadequately.

mixed Integration Testing :-

also called (Sandwiched IT) [Top-down + Bottom-up]

→ In top down approach, testing can start only after the top-level module have been coded & unit tested.

→ In bottom-up approach, testing can start only after the bottom level modules are ready.

Advantages :-

① very useful for larger projects having several sub projects.

② This approached overcomes this shortcomings of the top-down and bottom-up approaches.

③ parallel tests can be performed in top & bottom layer tests

Disadvantages :-

① requires high cost.

② cannot be used for smaller systems.

Unit Testing :-

→ it is a type of s/w testing that focuses on individual units or components of a s/w system.

→ typically performed by developers and it is performed (early in the development process) before the code is integrated and tested.

→ unit tests are (automated) and are run each time the code is changed to ensure that new code doesn't break existing functionality

→ this allows developers to quickly identify & fix any issues early in the development process, improving the overall quality of the s/w

→ Objective of Unit Testing :-

- → to isolate a section of code.
- → to verify the correctness of the code.
- → to test every function and procedure.
- → to find bugs early in the development cycle and to save costs.
- → to help with code reuse.

→ Types of UT

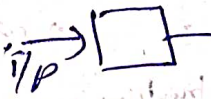
Manual | Automated

→ Workflow

Create Test cases → Review → Baseline → Execute Test Cases

→ Unit Testing Techniques :-

① **Black Box Testing** :- This testing technique is used in covering the unit tests for i/p, user-interface & o/p parts.



② **White Box Testing** :-

used in testing the functional behaviours of the system by giving the i/p and checking the functionality o/p including the internal design structure and code of the modules.

③ **Gray Box Testing** :- this is used in executing the relevant test cases, test methods and test functions & analyzing the code performance of the modules.

Testing Tools

① JTest ② Junit ③ nunit ④ Emma ⑤ PHP unit

Advantages of unit Testing

(i) It allows programmer to refine code and make sure the module works properly.
(ii) Early detection of Errors
(iii) Improved code quality
(iv) Increased confidence of the developers.
(v) Faster development
(vi) Better documentation
(vii) Reduces Time & cost

Disadvantages :-

(i) It will not cover all the errors in the module
(ii) not sufficient for checking the errors in UI.
(iii) cannot cover the non-functional testing parameters such as scalability / performance of the system.
(iv) Time & Effort.
(v) Difficulty in Testing complex units
(vi) maintenance overhead.

Unit Testing	Integration Testing
→ each module is separately tested.	→ all modules of the s/w are tested combined.
→ tester knows the internal structure of the s/w.	→ doesn't knows
→ 1st testing process	→ testing performed after unit testing.
→ also it is called white box testing.	→ also it is called black box testing.
→ performed by developers.	→ performed by testers.
→ detection of defect is easy.	→ difficult.
→ less costly	→ more costly.
→ module specification is done initially.	→ interface specification is done initially.
→ maintenance is cost effective.	→ maintenance is expensive
→ Fast execution	→ speed is slow.

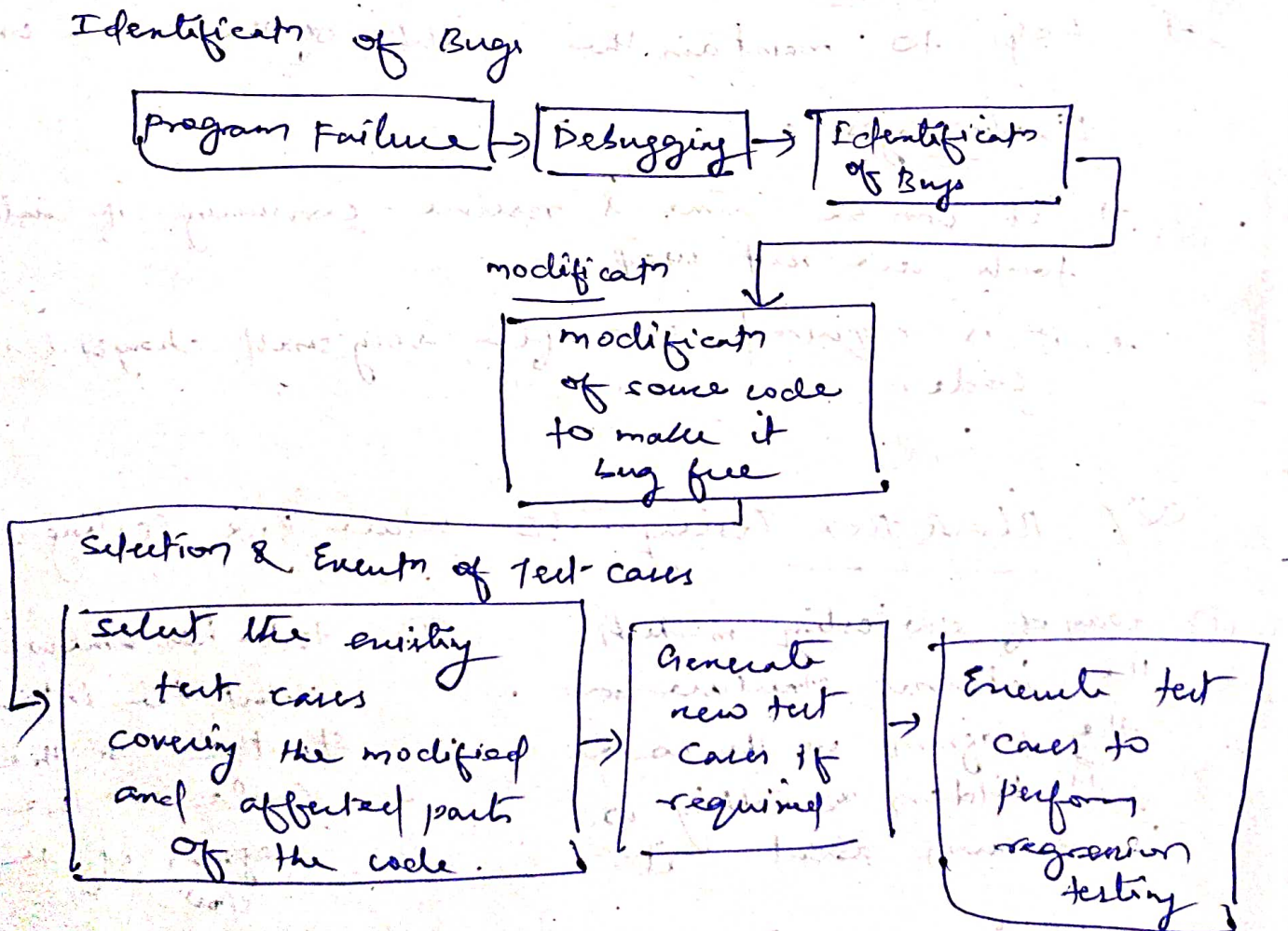
(d) Regression Testing :-

It is the process of testing the modified parts of the code and the parts that might get affected due to the modifications to ensure that no new errors have been introduced in the s/w after the modifications have been made.

When to do regression testing ?

1. when new functionality is added to the system
2. when some defect has been identified in the s/w and the code is debugged to fix it.
3. when the code is modified to optimize its working.

Process :-



Tools
→ selenium
→ QFT
→ win Runner
→ SilkTest

Advantages of Regression Testing :-

→ it ensures that no new bugs have been introduced after adding new functionalities to the system.

→ As most of the test cases used in Regression Testing are selected from the existing test suite, and we already know their expected o/p's. Hence, it can be easily automated by the automated tools.

→ helps to maintain the quality of the source code.

Disadvantages :-

→ it can be time & resource - consuming if automated tools are not used.

→ it is required even after very small changes in the code.

Q) Black Box Testing vs White Box Testing.

Black Box Testing	White Box Testing
① way of s/w testing in which the internal structure or the program or the code is hidden & nothing is known about it.	① tester has knowledge about the internal structure or the code or the program of the s/w.
② Integration Testing	② Unit Testing.
③ done by software testers.	③ done by software developers
④ no knowledge of implementation	④ knowledge of implementation

(5) implementation of code is not needed.	(5) code implementation is necessary.
(6) It is a functional test of the s/w.	(6) it is a structural test of the s/w.
(7) no knowledge of programming is required.	(7) mandatory to have knowledge of programming.
(8) it is the behaviour testing of the s/w.	(8) Logical testing
(9) also called closed testing.	(9) also called as clear box testing.
(10) least time consuming	(10) more time consuming
(11) not applicable for algo testing.	(11) ~
(12) less exhaustive	(12) more exhaustive
Eg- search something on Google by using keywords.	Eg- By i/p to check & verify loops.
Types → Functional Testing, non-" ", Regression Testing	Types → Path Testing, Loop Testing, Condition Testing

---

Q) Code coverage :-

→ Testing which helps in determining how much code of the source is tested which helps in assessing the quality of the test suite.

→ also considered as one of the forms of white box testing.

→ helps in determining the performance & quality aspects of any s/w.

Code coverage = $\frac{\text{no of lines of code executed}}{\text{no of lines of code in a system component}} \times 100$

Different Types of coverage are :-

① **Statement coverage** :- no of statements successfully executed in the program source code.

= $\frac{\text{no of statements executed}}{\text{Total no of statement}} \times 100$

⇒ basic form of coverage & ensures that every line of code has been tested.

* ② **Decision coverage / Branch coverage** :-

= $\frac{\text{no of decision executed}}{\text{Total no of decision outcomes in the source code}} \times 100$

⇒ it ensures that every possible branch of the code has been executed.

③ **Function coverage** :-

no of functions that are called & executed at least once in the source code.

= $\frac{\text{no of function called}}{\text{Total no of fun}} \times 100$

④ **Condition coverage / Expression coverage** :-

no of boolean cond^n / expression statements executed in the cond^n statement.

= $\frac{\text{no of executed program operands}}{\text{Total no of operands}} \times 100$