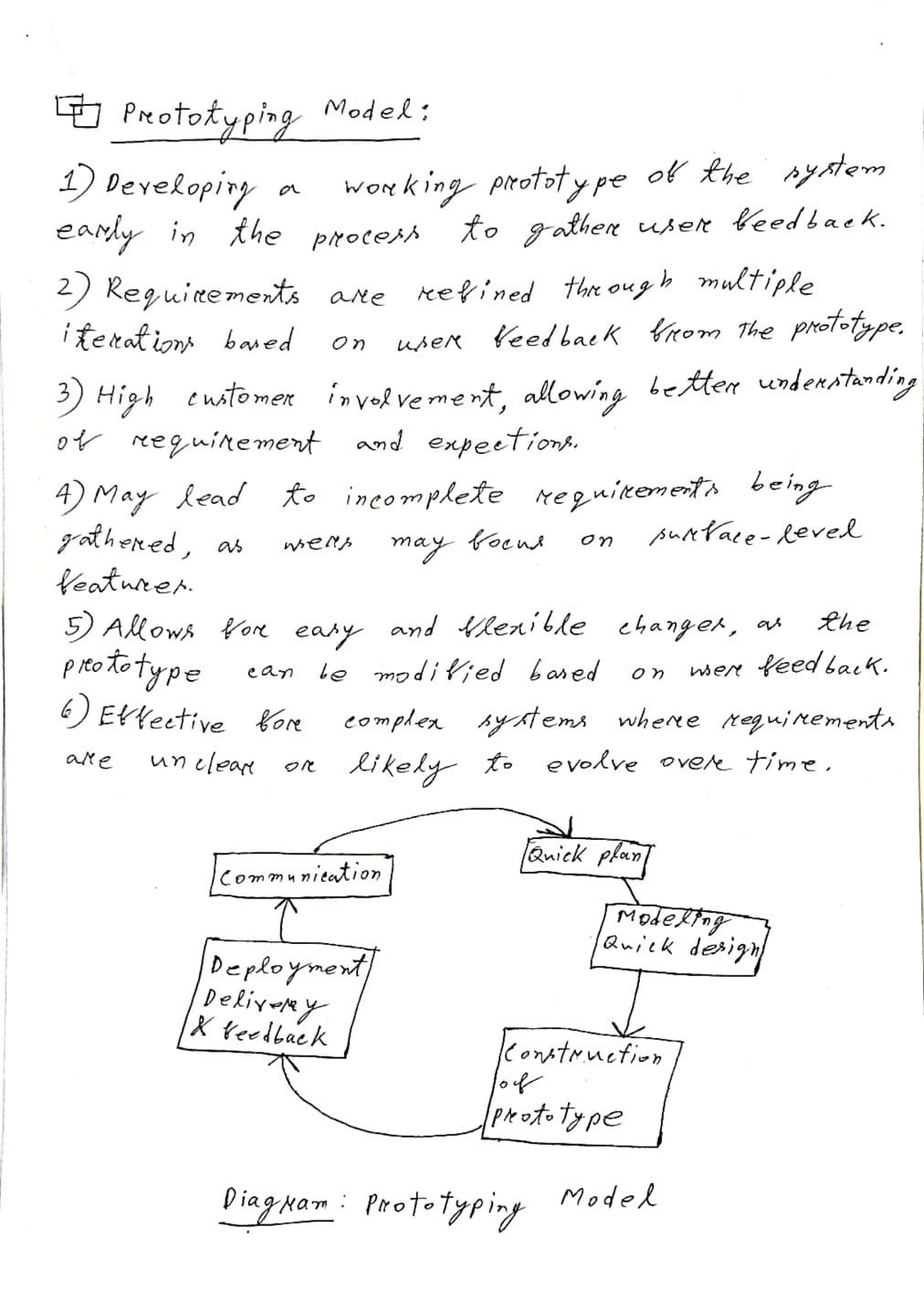
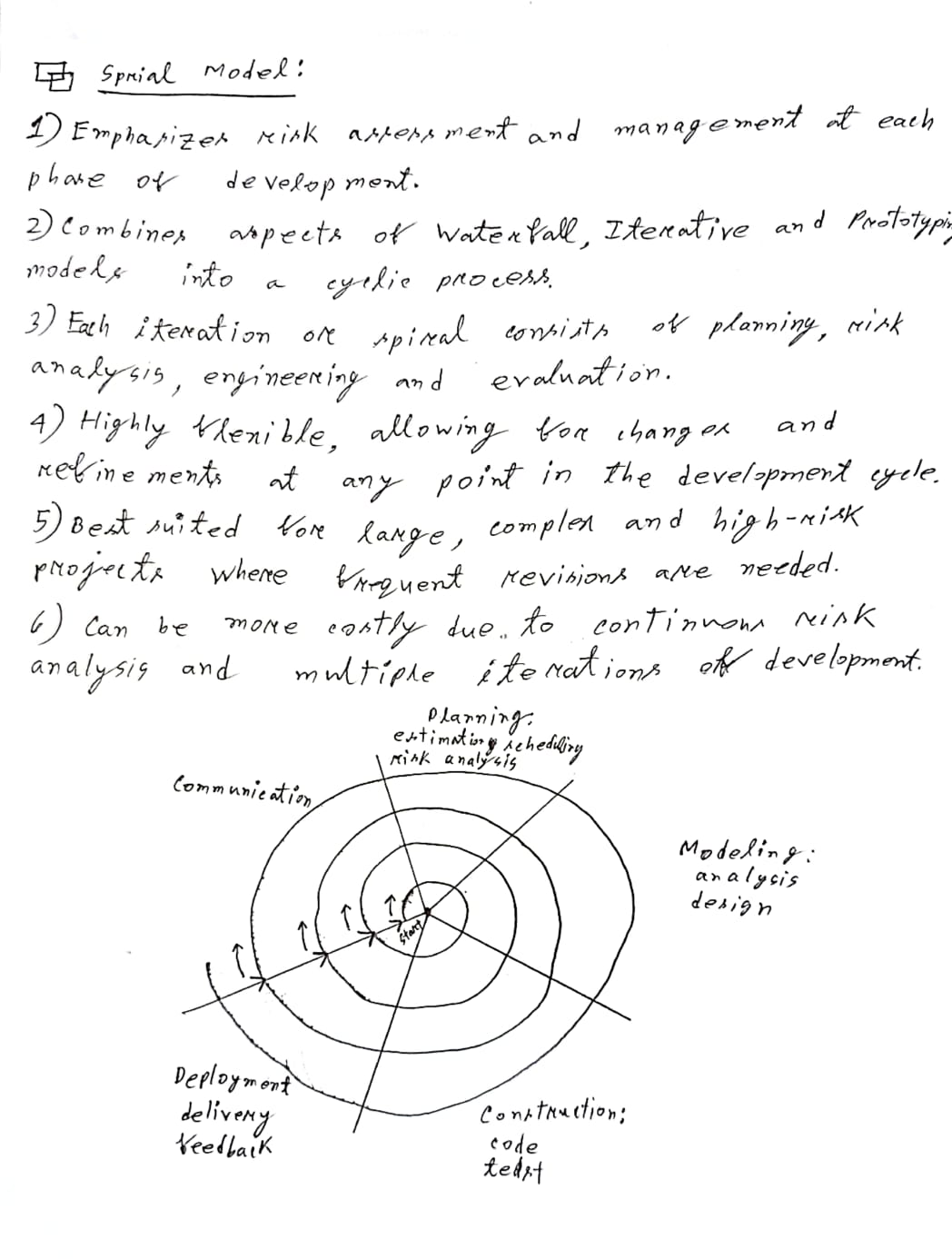
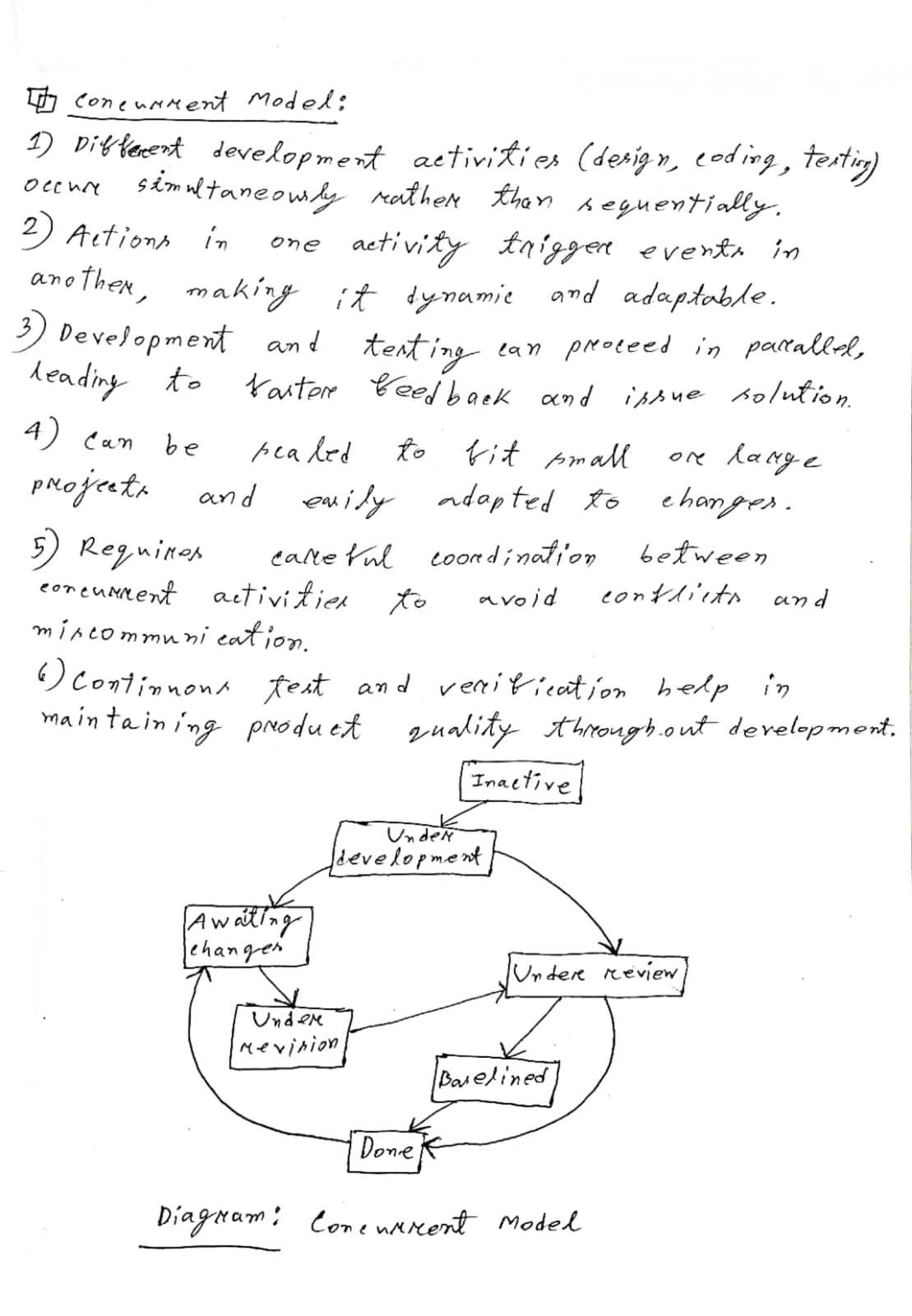
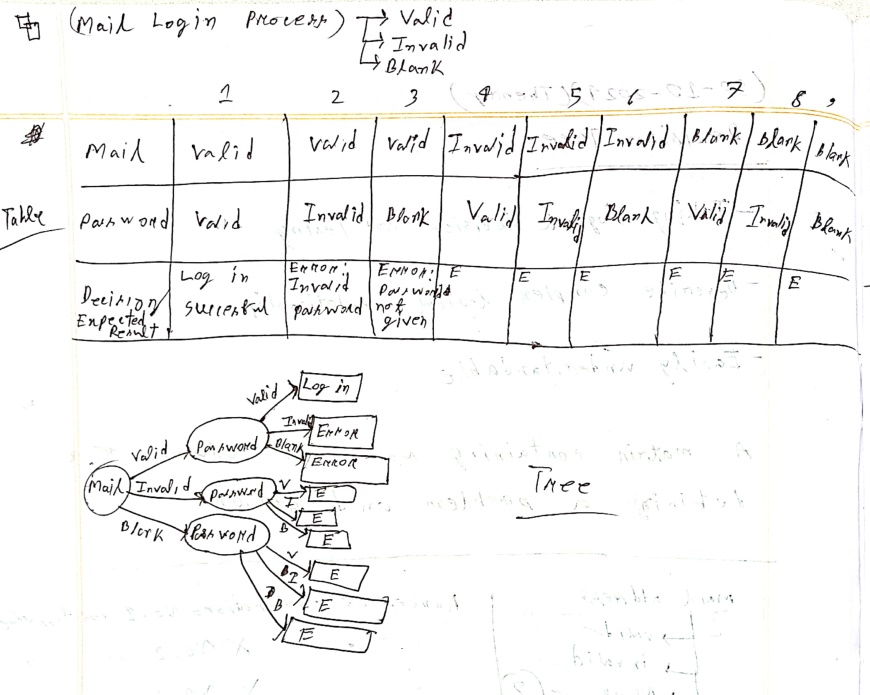
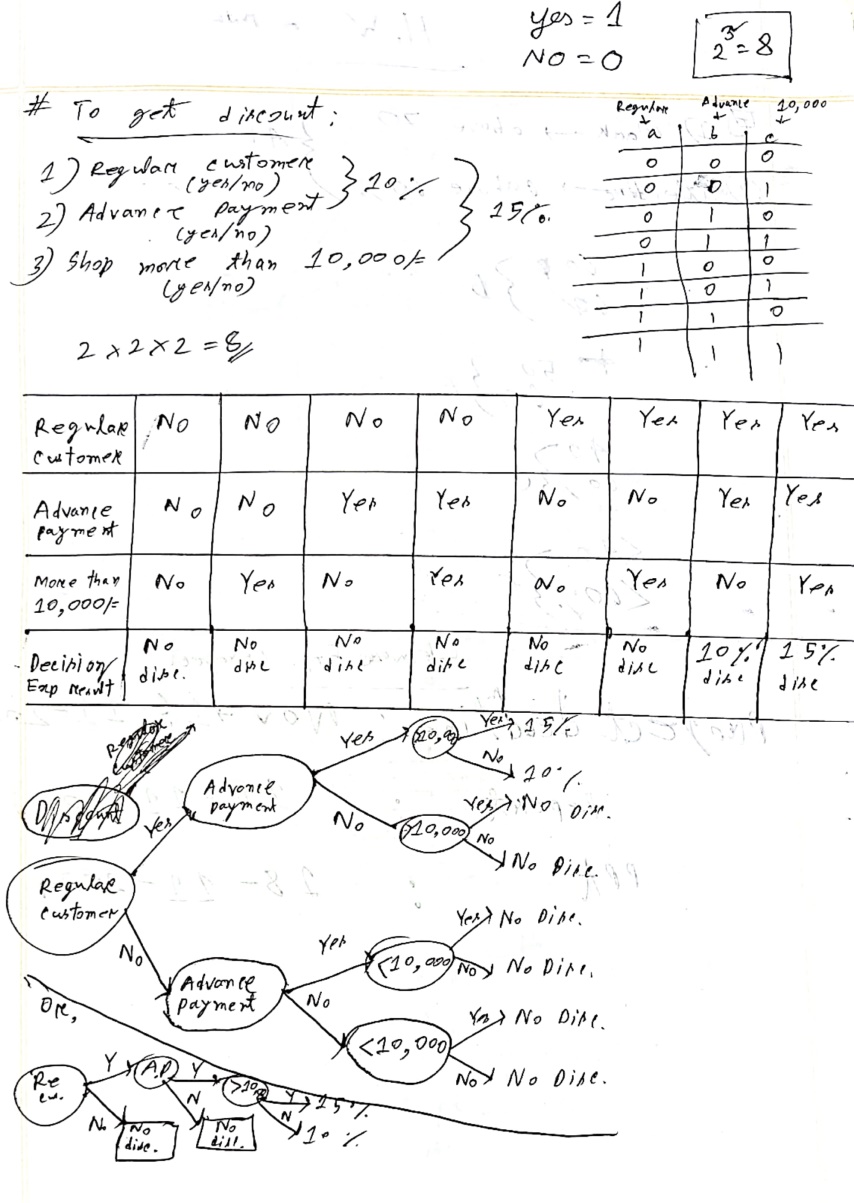
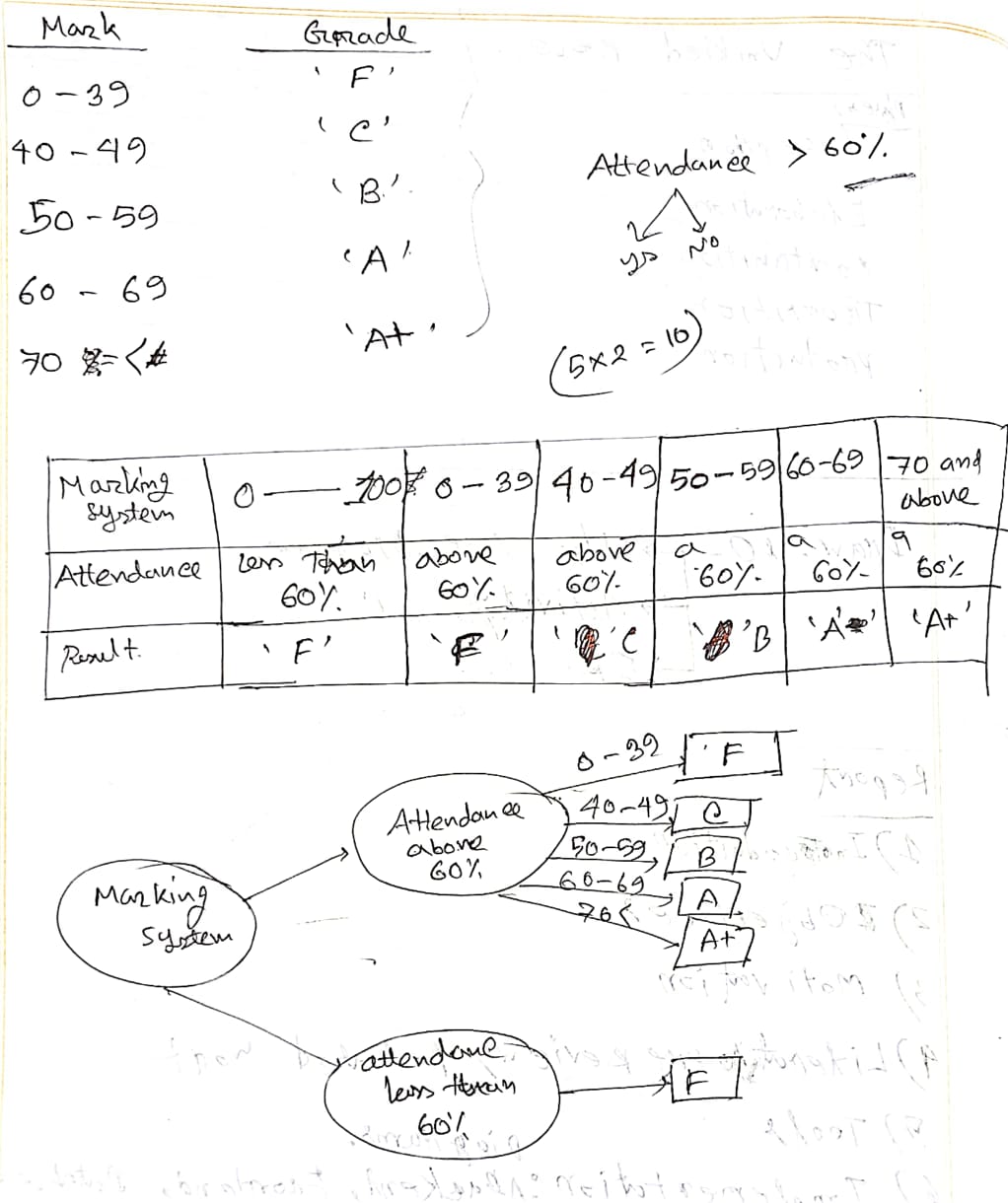
**Topic-1 (Evolutionary Process Model)**

**Evolutionary Process Model**: The Evolutionary Process Model is a software development approach where the system is developed incrementally, allowing for gradual improvement and refinement. It focuses on building the system through multiple iterations or versions, with each version incorporating feedback from previous phases. This model accommodates evolving requirements, as new features can be added and modified over time based on user feedback and emerging needs. The process is adaptive, ensuring that the system progressively evolves toward a more complete and robust solution.

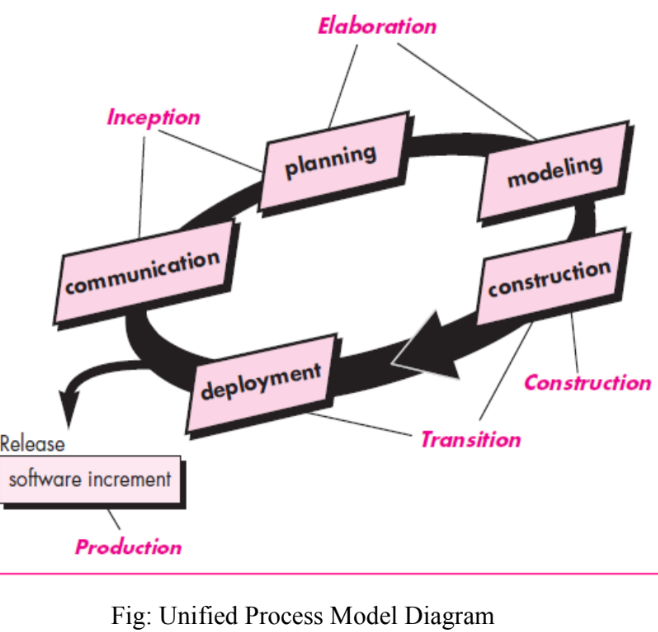


**Topic-2 (Decision Tree)**

**** ****



**Topic-3 (THE UNIFIED PROCESS MODEL)**



**1) Unified Process Model**: The Unified Process is a use case-driven, architecture-centric, iterative, and incremental software development process that aligns closely with UML. It progresses through 5 phases—Inception, Elaboration, Construction, Transition, and Production.

**a) Inception Phase**: Involves customer communication and planning, where business requirements are gathered, a preliminary system architecture is proposed, and an initial project plan is developed.

**b) Elaboration phase**: Focuses on refining requirements and expanding the architectural model. It includes creating five key views: the use-case model, requirements model, design model, implementation model, and deployment model. Key goals are addressing risks and validating the system architecture.

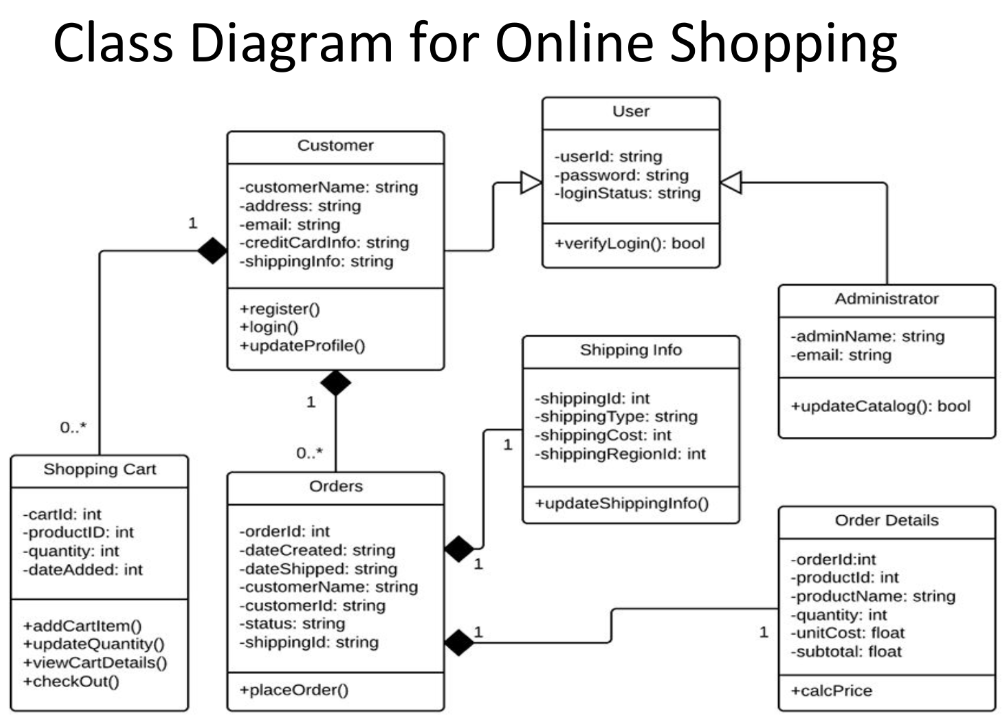
**c) Construction phase**: Involves building the software components based on the architectural model, finalizing design and requirements, implementing features in source code, and conducting unit and integration testing.

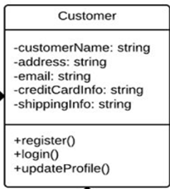
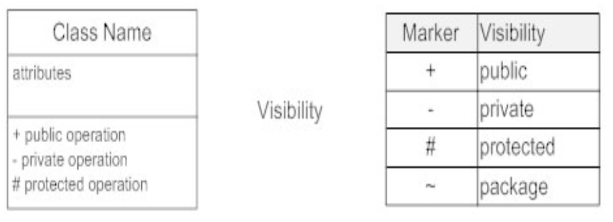
**d) Transition Phase:** Delivers the software to end users for beta testing, gathers feedback, and addresses defects or necessary changes.

**e) Production phase**: Focuses on monitoring the software's ongoing use, supporting the operating environment, and handling defect reports and change requests, ensuring the software remains stable.

**Topic- 4 (Class Diagram)**

1) Explain the relationship between the Customer, Shopping Cart, and Orders classes in the online shopping class diagram. Additionally, describe the purpose of the Shipping Info and Order Details classes and how they interact with Orders.



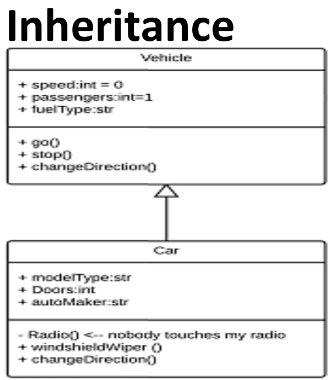
 

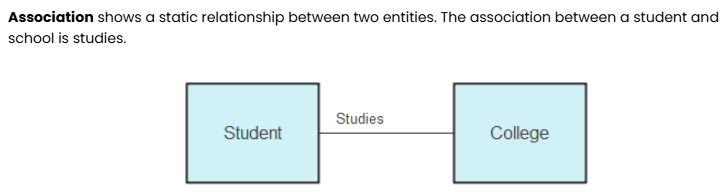
Customer- **Class**

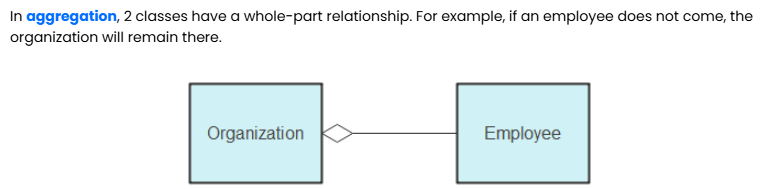
customerName: **Attributes**

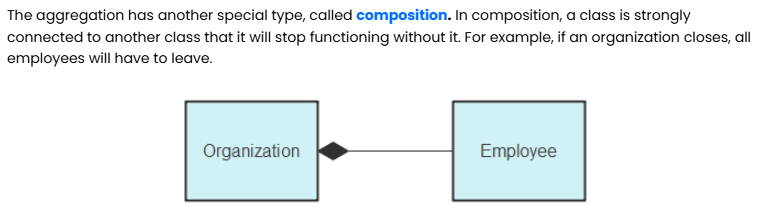
......

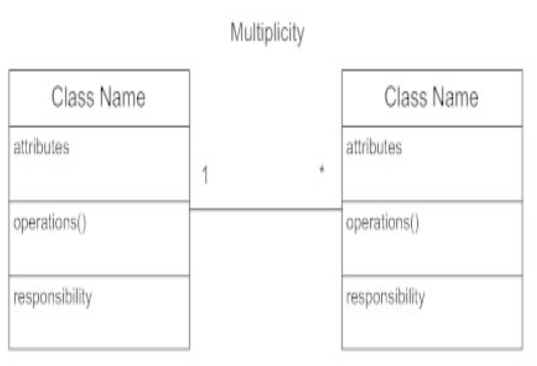
.......



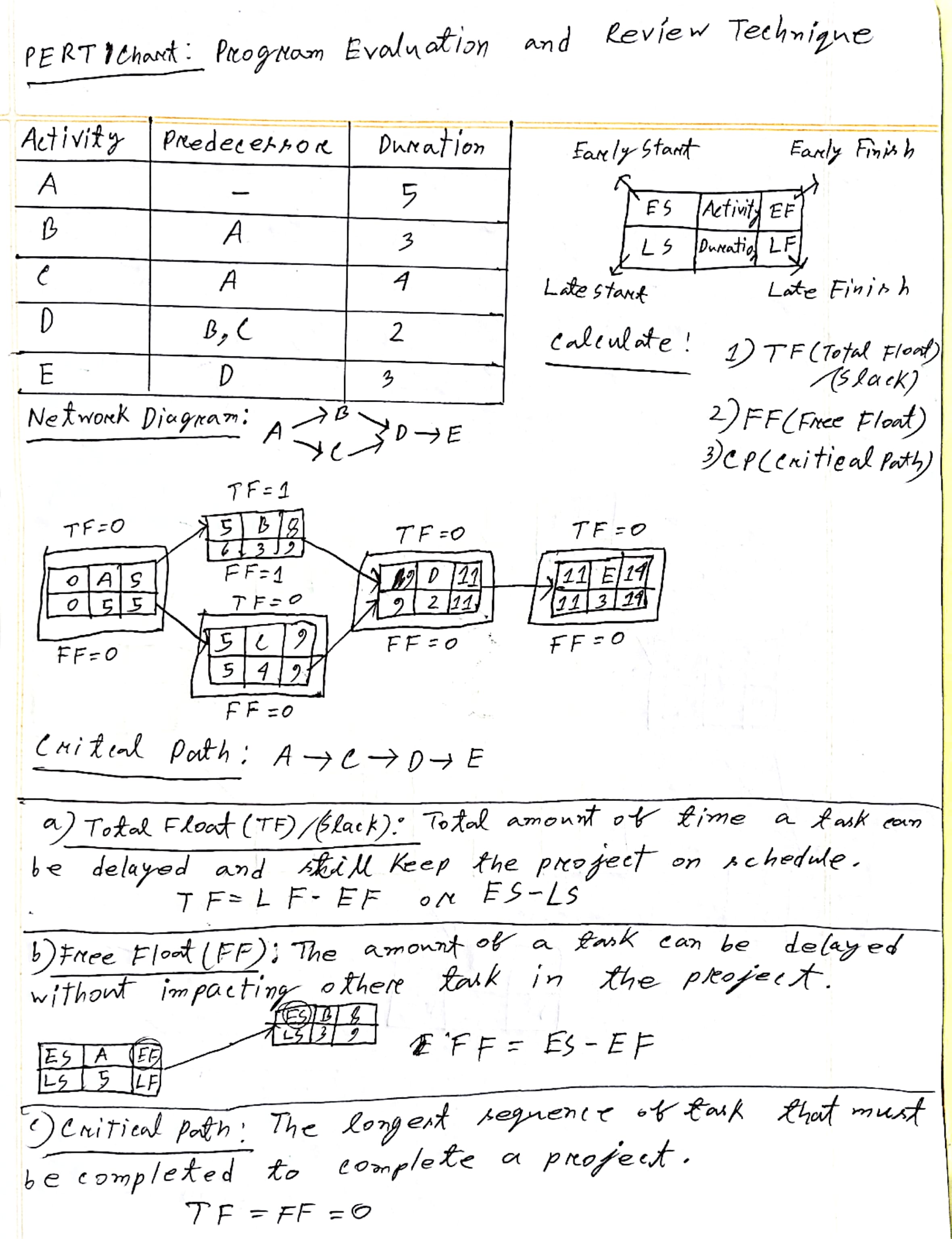








**Topic-5 (Pert Chart)**



**Topic-6 (Agile Model)**

**1) Agile Model**: Agile development is a phrase used in software development to describe methodologies for incremental software development

**2) Agile Manifesto**: **a)** Individuals and interactions over processes and tools.

**b)** Working software over comprehensive documentation.

**c)** Customer collaboration over contract negotiation. **d)** Responding to change over following a plan.

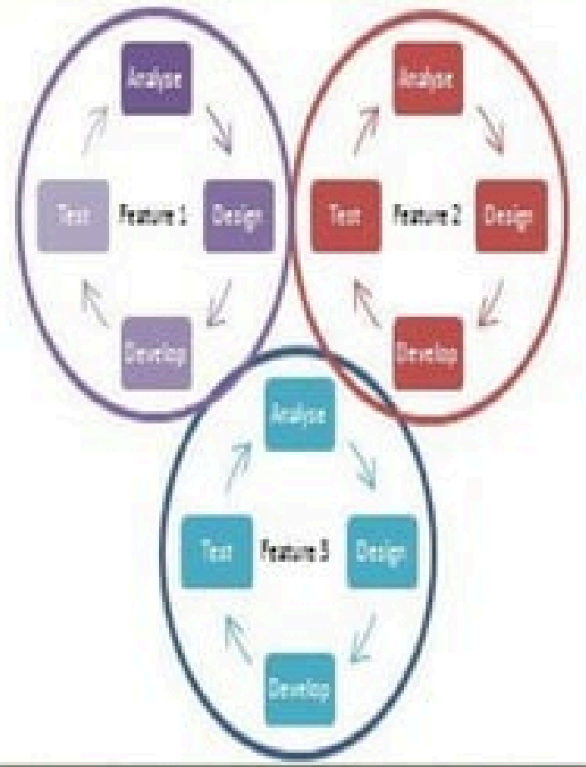
**3) Principles Of Agile**: a) Customer Satisfaction b) Working Software c) Measure Of Progress

d) Late Changes Are Welcome e) Face\_To\_Face Communication f) Motivated Individuals

g) Technical Excellence h) Simplicity i) Self\_organizing j) Regular Adoption

**4) Characteristics of Agile**: a) Modularity b) Iterative c) Time-bound d) Incremental e) People oriented

f) Less defect g) Collaborative h) Motivating the team

**5)** 

**6) Existing Agile Methods**

• Extreme Programming ("XP")

• Agile Unified Process

• Scrum

**7)** **Agile Unified Process**: Phases of AUP- a) Inception b) Elaboration c) Construction d) Transition

**8) Disciplines of AUP**: **a)**Model **b)**Implementation **c)**Test **d)**Deployment **e)**Project Management

**f)** Environment

**9) Extreme Programming: a)** Most prominent Agile Software development method.

**b)** Prescribes a set of daily stakeholder practices

**c)** "Extreme" levels of practicing leads to more responsive software.

**d)** Changes are more realistic, natural, inescapable.

**10) Scrum**: Scrum is an Agile framework for completing complex projects. Scrum originally was formalized for software development projects, but it works well for any complex, innovative scope of work. The possibilities are endless. The Scrum framework is deceptively simple.

**11) Advantages Of Agile Model**: **a)**Customer Satisfaction **b)**People and interactions **c)**Customers, developers and testers constantly interact with each other **d)**Working software is delivered frequently

**e)**Face-to-face conversation **f)**Close, daily cooperation between business people and developers

**g)**Continuous attention to technical good design **h)**Regular adaptation to changing circumstances.

**i)**Even late changes in requirements are welcomed

**12) Advantages Of Agile Model**: **a)** Effort estimation for large projects is difficult in the Agile model.

**b)** Necessary design and documentation are often neglected.

**c)** Projects can derail if customer expectations are unclear.

**d)** Only experienced programmers can make critical decisions, limiting opportunities for newbies.

**Topic-7 (Graphical User Interface)**

**1) Interface Design Advantages**: **a)** Easy to learn **b)** Easy to use **c)** Easy to understand

**2) Typical Design Errors**: **a)** lack of consistency **b)** too much memorization **c)** no guidance / help

**d)** no context sensitivity **e)** poor response **f)** Arcane/unfriendly

**3) Golden Rules of GUI**: **a)** Place the user in control **b)** Reduce the user’s memory load

**c)** Make the interface consistent

**4) Place the User in Control**: Define interaction modes in a way that does not force a user into unnecessary or undesired actions.

**5) Reduce the User’s Memory Load**: **a)** Reduce demand on short-term memory.

**b)** Establish meaningful defaults. **c)** Define shortcuts that are intuitive.

**6) Make the Interface Consistent: a)** Allow the user to put the current task into a meaningful context.

**b)** Maintain consistency across a family of applications.

**c)** If past interactive models have created user expectations, do not make changes unless there is a

compelling reason to do so.

**7) User Interface Design Models**: **a) User model** — a profile of all end users of the system

**b) Design model** — a design realization of the user model

**c) Mental model (system perception)** — the user’s mental image of what the interface is

**d) Implementation model** — the interface “look and feel” coupled with supporting information that

describe interface syntax and semantics

**8) Interface Design Steps**: **a)**  **Define Objects and Actions**: Identify interface elements and their operations based on interface analysis.

**b) Model Events**: Specify user actions that trigger interface state changes and model this behavior..

**c) Visualize States**: Show how each interface state appears to the user.

**d) Interpret System State**: Ensure users can understand the system's state through the provided interface information.

**9) Interface Design Principles**: **a) Anticipation**: Design should predict the user's next action.

**b) Communication**: Inform users about the status of their actions.

**c) Consistency**: Use uniform navigation, menus, icons, and layouts.

**d) Controlled Autonomy**: Guide users while enforcing navigation rules.

**e) Efficiency**: Optimize user productivity, not just system performance.

**Topic-8 (Software Testing and Quality Assurance)**

**1) What is Software Testing?**

**Software Testing** is the process of evaluating software quality by applying test data, comparing actual results with expected results, and ensuring it meets client requirements. It is a structured approach involving standards, methods, strategies, and tools to identify defects and validate functionality as the last quality checkpoint before release.

**2) Why Testing is Important**?

**a)** Identifies defects and errors made during development.

**b)** Ensures customer satisfaction and reliability in the application.

**c)** Delivers a quality product, building customer confidence.

**d)** Provides high-quality, low-maintenance software with accurate and reliable results.

**e)** Ensures effective performance of the software application or product.

**f)** Prevents expensive failures in later stages of development.

**g)** Helps businesses remain competitive.

**3) Seven Principle of Software Testing**?

**a) Testing shows presence of defects**: Testing reveals defects but cannot confirm the software is 100% defect-free.

**b) Exhaustive testing is impossible**: Complete testing of all inputs and conditions is infeasible; focus on priorities and risks.

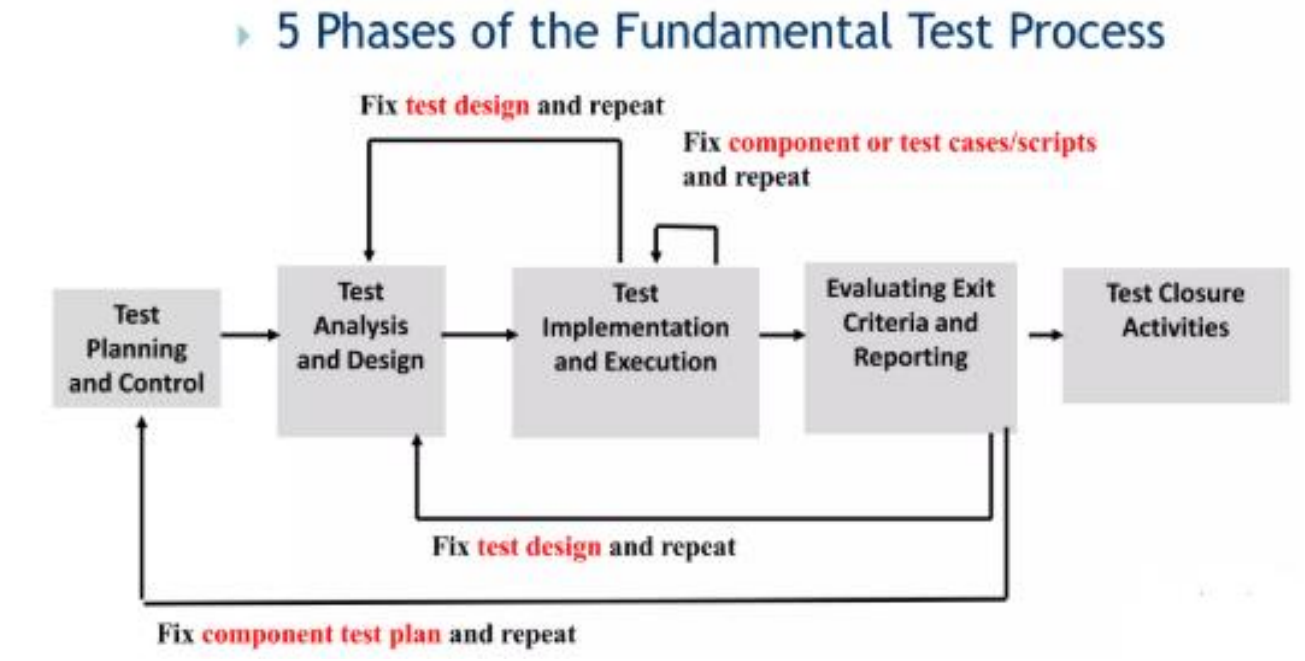
**c) Early testing**: Begin testing early in the development lifecycle to align with objectives.

**d) Defect clustering**: Most defects are found in a small number of modules.

**e) Pesticide paradox**: Repeating the same tests stops finding new defects; update tests regularly.

**f) Testing is context-dependent:** Testing methods vary based on the type of software (e.g., critical systems vs. e-commerce).

**g) Absence of errors fallacy:** A defect-free system is useless if it does not meet user needs.



|  |  |  |  |
| --- | --- | --- | --- |
| **Term** | **Focus** | **Activities** | **Type of Activity** |
| **Quality Assurance (QA)** | Processes and Procedures | Ensures processes and standards are followed. | Process-oriented, preventive |
| **Quality Control (QC)** | Product Verification | Checks if the product meets requirements. | Product-oriented, corrective |
| **Testing** | Bug/Defect Identification | Finds and reports bugs in the software. | Product-oriented, preventive |

**4) Types of Testing?**

**a) Manual Testing:** Testers test the software manually, without using any automated tools.

They act like real users to find bugs and unexpected behavior.

They use test plans, test cases, and exploratory testing to ensure everything works as it should.

**b) Automation Testing**: Testers write scripts and use other software to automate the testing process.

This helps them re-run tests quickly and repeatedly.

