Software Engineering

1. What is the Personal Software Process (PSP)?

- PSP is a structured process for individual software engineers to improve productivity and quality.
- It includes planning, development, and postmortem phases to track defects and time spent.
- Uses statistical methods for performance measurement and continuous improvement.
- Example: A developer records time spent on coding and defects found to optimize future projects.

---

2. What is the Software Development Life Cycle (SDLC)?

- SDLC is a systematic approach to developing software through defined phases.
- Phases include planning, analysis, design, implementation, testing, and maintenance.
- Helps in improving software quality and project management.
- Example: In SDLC, a banking app undergoes requirement gathering before coding starts.

---

3. What are umbrella activities in software engineering?

- Umbrella activities support core SDLC phases to ensure software quality.
- Includes risk management, configuration management, quality assurance, and project tracking.
- These activities run parallel to development activities for better process control.
- Example: Formal technical reviews are conducted to identify defects early.

---

4. What are process patterns in software engineering?

- Process patterns provide reusable solutions for common software development problems.
- They help in improving efficiency, reducing errors, and maintaining consistency.
- Classified into task patterns, stage patterns, and phase patterns.
- Example: A team follows the "Code Review" pattern to ensure bug-free code before deployment.

---

1. What is the Agile Manifesto?

- The Agile Manifesto emphasizes flexibility, collaboration, and customer satisfaction.
- It values individuals & interactions, working software, customer collaboration, and responding to change.
- Focuses on iterative development with continuous feedback.
- Example: Agile teams adapt requirements frequently based on stakeholder feedback.

---

2. What are Prescriptive & Specialized Process Models?

- Prescriptive models define structured phases for software development (e.g., Waterfall, Incremental).
- Specialized models address specific needs, like Component-Based Development and Formal Methods.
- Prescriptive models provide discipline, while specialized models offer flexibility for unique cases.
- Example: The Spiral model is a prescriptive model, while Agile is a specialized model.

---

3. What are the phases of the Waterfall model?

- **Requirement Gathering:** Understanding project needs and constraints.

- **Design:** Creating system architecture and detailed specifications.
- **Implementation & Testing:** Writing code and validating it against requirements.
- **Deployment & Maintenance:** Delivering the software and handling updates.
- Example: A payroll system follows Waterfall, where requirements are fixed before development.

---

4. What is concurrent development in software engineering?

- Concurrent development allows multiple software processes to execute simultaneously.
- Activities like design, coding, and testing happen in parallel to speed up delivery.
- Helps in handling dependencies and reducing bottlenecks in large projects.
- Example: A team develops the UI and backend of an e-commerce website concurrently.

---

5. What is iterative development?

- Iterative development builds software in repeated cycles, refining features progressively.
- Each iteration results in a functional version with incremental improvements.
- Helps in adapting to changing requirements without reworking the entire system.
- Example: A mobile app releases a basic version first, then adds features like payments in later iterations.

---

6. What are the roles in a Scrum team?

- **Product Owner:** Defines the product vision and prioritizes the backlog.
- **Scrum Master:** Facilitates the process, removes obstacles, and ensures Agile principles.
- **Development Team:** Cross-functional members who design, code, and test the product.
- **Stakeholders:** Provide feedback and influence product decisions.

- Example: In a startup, the CEO may act as the Product Owner while developers form the team.

---

1. What are requirements in software engineering?

- Requirements define what a system should do and how it should perform.
- They are categorized as functional (what it does) and non-functional (how it performs).
- Serve as a foundation for design, development, and testing.
- Example: A banking app requires login authentication (functional) and must respond within 2 seconds (non-functional).

---

2. What is feasibility study implementation?

- Evaluates if a project is viable before development begins.
- Includes technical, economic, legal, operational, and schedule feasibility.
- Helps in decision-making and risk assessment.
- Example: A company checks if it has enough resources to build an AI chatbot before starting development.

---

3. What are the different origins of changes in software?

- **Business Changes:** Market demands or company strategies evolve.
- **Technology Changes:** New tools or platforms replace old ones.
- **Customer Feedback:** Users request modifications or improvements.
- **Regulatory Changes:** Laws and standards require compliance updates.
- Example: A payment app updates its security features due to new banking regulations.

---

4. What are the information domain characteristics of function points?

- **External Inputs (EI):** User-provided inputs processed by the system.
- **External Outputs (EO):** Processed data delivered to the user.
- **Internal Logical Files (ILF):** Data stored within the system.
- **External Interface Files (EIF):** Data exchanged with other systems.
- Example: An inventory system has ILFs for stock data and EO for generating reports.

---

5. What are the activities for Requirement Elicitation and Analysis?

- **Stakeholder Interviews:** Gathering insights from users and clients.
- **Surveys & Questionnaires:** Collecting structured feedback.
- **Prototyping:** Developing mock-ups for better understanding.
- **Document Analysis:** Reviewing existing system records.
- Example: A software team uses prototyping to refine a hospital management system's interface.

---

6. What is requirement validation in requirements engineering activity?

- Ensures that gathered requirements are correct, complete, and feasible.
- Involves checks for consistency, correctness, and testability.
- Uses techniques like reviews, prototyping, and test case generation.
- Example: A banking system requirement is validated by simulating a money transfer process.
