Assignment-1

1) What is the main difference between Programming and Software engineering?

Aspect	Programming	Software engineering
Definition	The act of writing code to make a computer perform specific tasks.	A systematic approach to designing, developing, testing, deploying and maintaining software systems.
Focus	Writing functional code.	Building reliable, scalable and maintainable software.
Scope	Narrow - mainly deals with coding and debugging.	Broad - includes planning, design, testing, project management, and lifecycle maintenance.
Goal	Make a program work.	Make a complete, high-quality software product.
Team involvement	Often individual work.	Collaborative work involving multiple roles.

2) What does the phrase "Hyrum's Law" state in software engineering?

→ Hyrum's Law in software engineering states that;-

"With a sufficient number of users of an API, it does not matter what you promise in the contract - all observable behaviours of your system will be depended on by somebody."

Hyrum's law warns developers that any observable behaviour can become a dependency, so changing software safely requires extreme care and good versioning practices.

3) Mention two reasons why long term projects need continuous upgrades?

1) Technology evolution:-

New technologies, frameworks, and tools emerge over time. Upgrading helps the project stay compatible, efficient and secure with modern systems.

iv) Changing Requirements and User needs:-
As business goals, user expectations or market conditions change, the software must be updated to meet new requirements and remain useful and competetive.

4) What is meant by the phrase "shifting left" in Software development?
→ In Software development "shifting left" means moving tasks - especially testing, quality checks, and security reviews - earlier (to the left) in the development ~~of~~ lifecycle.
In short it means :- "shifting left" means testing early and often instead of waiting until the end of development.

5) What is trade-offs in engineering decisions?
→ In engineering, trade-offs refer to the balancing of competing factors when making a decision - improving one aspect of a system often means compromising another.
A trade-off happens when you must sacrifice certain qualities or benefits to gain others that are more important for the situation.

6) Mention two challenges faced by software engineers when working in teams.
i) Communication and Coordination issues:
Misunderstandings or lack of clear communication can lead to inconsistent code, duplicated work or delays in project progress.
ii) Integrations and Code Conflicts:
When multiple engineers work on the same codebase, merging changes or integrating modules can cause version conflicts and compatibility problems.

7) How does knowledge sharing benefit teams?

i) Improves Collaboration and problem Solving:
When team members share ideas and experience, it becomes easier to find solutions and make better decisions together.

ii) Reduces dependency on individuals:
Sharing knowledge ensures that critical information is not limited to one person, preventing delays if someone is unavailable.

iii) Enhances learning and skill growth:
Team members learn from each other's expertise, leading to overall skill improvement and innovation.

8) Define the role of a team lead.
The role of a team lead is to guide, coordinate and support a team to ensure that projects are completed efficiently and successfully.
key responsibilities:-

i) Leadership and guidance:
provide direction, set goals, and motivate team members to achieve project objectives.

ii) Task management:
Assign work, monitor progress and ensure deadlines and quality standards are met.

iii) Communication Bridge:
Act as a link between the team and higher management, ensuring clear and effective communication.

iv) Problem solving and support:
Help resolve technical or interpersonal issues within the team.

9) Why should team members admit mistakes openly?
Team members should admit mistakes openly because it builds trust, learning and improvement within the team.

key reasons:-
i) Encourages learning and growth.
ii) Builds trust and transparency
iii) Prevents bigger problems.

10) How does knowledge sharing benefit teams?
i) Enhances collaboration:
When team members share information, ideas and experiences, it becomes easier to solve problems collectively.
ii) Reduces dependency on individuals:
Critical knowledge is distributed across the team, so project don't stall if someone is unavailable.
iii) Promotes learning and skill development:
Team members learn from each other, which strengthens the team's overall expertise and fosters innovation.

11) Explain with example why a low Bus factor is dangerous for software projects.
→ A low Bus factor is dangerous because it means that very few team members hold critical knowledge about a project. If one of them becomes unavailable, the project can face serious delays or even fail.
explanation:
→ Bus factor measures the number of people who need to be unavailable before a project is at risk.
→ A low Bus factor means most of the project knowledge is concentrated in just one or two people.

example:
Imagine a software project where only the lead developer knows how the main system works.

→ If the lead developer suddenly leaves, falls sick, or is unavailable, the rest of the team cannot continue development effectively.
* This can lead to project delays, loss of critical functionality or even failure.

12) Why is feedback culture important in software teams?
→ A feedback culture is important in software teams because it promotes continuous improvement, collaboration and high quality outcomes.
i) Improves performance and quality.
ii) Encourages learning and growth.
iii) Strengthens Team communication and trust.