6. Explain about XP rules and rules of planning game?

- The rules help everyone on the team stay on the same page.
- They remind the team what to do when things are going well.
- They also guide the team when things are not going well.

Rules of Planning Game

Planning Game Rules:

In Extreme Programming, the Planning Game starts with the first release planning meeting and continues until the final release. For example, in a project to build an e-commerce website, the team will first define the rules (like how features are prioritized and how updates are made) and follow them throughout the project. However, if the business needs change or new technology is discovered, the rules can be adjusted, as long as they still follow Extreme Programming practices.

Maximizing Value:

The goal is to deliver the most valuable features quickly while minimizing costs and risks. For example, in the e-commerce website project, the team might first release the shopping cart feature, as it's crucial for the site to function. This feature is developed and released as quickly as possible, and after each release, the team learns from user feedback to improve the website in the next iteration. The process continues with updates until the full website is completed.

The basic rules of planning game can be categorized into the following areas –

Planning
Managing
Designing
Coding
Testing

Planning

Planning is done during release planning and iteration planning. The rules are same for both.

Managing

1. The team is given a dedicated open workspace.
2. Each workstation is to be arranged such that two developers can sit side by side and easily slide the key
3. board and mouse.

Designing

1. The rule is “always use the simplest possible design that gets the job done”

Coding

Coding Standards are to be followed

Testing

All coders must pass unit test before it released

7. Brief explanation about Extreme programming artifacts?

User Story Cards:

- These are simple, short descriptions of features from the user's perspective.
- They help the team understand what the user wants from the software.
- Each card represents one small feature or function.

Acceptance Tests:

- These tests define whether a feature works as expected.
- They are created from the user's perspective.
- The feature must pass these tests to be considered complete.

Estimates:

- Estimates are predictions of how long tasks will take to complete.

- They help the team plan and allocate resources.
- Estimates are based on the complexity of the tasks.

Release Plan:

- This plan outlines when features will be released to the customer.
- It helps the team stay on track and meet deadlines.
- The plan is updated regularly based on progress and feedback.

Iteration Plan:

- This plan details the work to be done in a specific iteration or sprint.
- It focuses on achieving small, manageable goals in a short time.
- The iteration plan is flexible and can be adjusted as needed.

Task Cards:

- These are smaller, detailed tasks that break down user stories.
- They describe specific actions needed to complete a feature.
- Task cards help the team stay organized and focused on specific tasks.

Design:

- Design refers to how the software will be structured and built.
- It includes architecture, user interface, and overall system layout.
- The design is often flexible and evolves during development.

Unit Test Cases:

- These tests check individual pieces of code to ensure they work correctly.
- Unit tests are written before the code itself, as part of Test-Driven Development (TDD).
- Passing unit tests ensure that each component functions as intended.

Customer and Developer Communication Records:

- These records track discussions between customers and developers.
- They include feedback, questions, and decisions made during the project.
- Communication records help maintain clarity and alignment between the team and the customer.