* 1. What is Agile methodology ? Explain how xp extreme programming is used in agile methodology ?

gile methodology is a **software development approach** that emphasizes **iterative development**, **collaboration**, **continuous feedback**, and **flexibility** to changing requirements. It is based on the principles outlined in the **Agile Manifesto** (2001), which values:

* **Individuals and interactions** over processes and tools
* **Working software** over comprehensive documentation
* **Customer collaboration** over contract negotiation
* **Responding to change** over following a plan

Agile breaks the development process into small, manageable units called **iterations** or **sprints** (typically 1–4 weeks), where a functional piece of software is developed, tested, and reviewed.

#### Key Features of Agile:

* Continuous customer involvement
* Frequent delivery of working software
* Adaptive planning
* Emphasis on team communication and feedback
* Face-to-face communication and collaboration
* Regular retrospectives for continuous improvement

**Extreme Programming (XP)** is one of the most well-known **Agile frameworks**. It focuses on **technical excellence** and is designed to improve software quality and responsiveness to changing customer requirements.

#### Key Practices of XP in Agile:

1. **Short Development Cycles (Iterations):**
   * XP uses 1-2 week iterations, where new features are designed, coded, tested, and delivered.
   * Encourages early and frequent releases.
2. **Test-Driven Development (TDD):**
   * Writing automated tests **before** the code.
   * Ensures code correctness and reduces bugs.
3. **Pair Programming:**
   * Two developers work together at one workstation: one writes code, the other reviews it in real time.
   * Enhances code quality and team collaboration.
4. **Continuous Integration:**
   * Code is integrated and tested frequently (multiple times a day).
   * Detects issues early and ensures a stable product.
5. **Refactoring:**
   * Continuous improvement of code structure without changing its behavior.
   * Makes the codebase clean and maintainable.
6. **Simple Design:**
   * Design only what is necessary for the current iteration.
   * Avoids over-engineering and complexity.
7. **Collective Code Ownership:**
   * Anyone can improve any part of the code.
   * Encourages team responsibility and faster progress.
8. **Sustainable Pace:**
   * Avoids overwork and burnout.
   * Teams maintain a consistent, productive work rhythm.
9. **On-site Customer:**
   * A real customer or stakeholder is available to answer questions and provide feedback.
   * Ensures the product meets user needs.
10. **Coding Standards:**
    * Team follows consistent coding conventions.