**Write short notes on following**

1. Scrum

Scrum software development methodology is an agile framework. It focuses on developing and maintaining complex products. It can be applied to other fields such as research. It is best suited for small teams of 5-10 members. The whole work is divided into iterations which are called sprints. These sprints are usually small periods of time, usually 2 weeks and no more than 1 month. During these sprints some predetermined task is to be completed and no extension of this period is allowed.

Daily meetings known as scrum meetings are held, which are 15 min timed meetings to demonstrate and track progress. At the end of a sprint, a sprint review takes place to review and improve work.

There are 3 roles in a Scrum framework to make it work ideally:

1. Product Owner - the stakeholder of the product and who represents the voice of the customer. A scrum team should have only one product owner. The product owner takes care of the business aspect of the product and is responsible for the success of the product. They should have a deep understanding of both the customer side and the developer side. They are the key communicator for the whole product.
2. Development Team - This consists of a team of 3-9 members who carry out all the tasks which are needed to build the product in increments. They give valuable output every sprint. The team should be self-organizing in nature and they are given all the required information only through Scrum Master and Product Owner.
3. Scrum Master - Scrum master is a person who is responsible for the implementation of scrum framework for development. They oversee the whole process of scrum and lead the daily scrum meetings. They are not the typical managers or leaders, their job is to keep the development team from getting distracted and help them to team in various things like determining the definition of finished product, coaching the team for delivering high quality product, facilitating team events to make progress faster.
4. Lean Development

Lean development is a software development methodology under Agile development, it derives its principles from lean manufacturing and translates them into software development domain. Its main purpose is to eliminate wasteful steps and do only what a customer would pay to be done for their product, all other steps are unnecessary and hence to be avoided. Lean is based on a solid conceptual framework and values which are gained from experience.

The main 7 elements of of Lean Software Development are:

1. Eliminate Waste - This refers to the activities which are not adding value to the final product that the customer needs. These activities should be eliminated. Some examples are - partially finished product, tasks switching, defects, extra features, relearning, waiting.
2. Amplify Learning - The software development process is sped up by using shorter iteration cycles. This allows for faster code refactoring and unit integration testing.
3. Decide as late as possible - Software development includes certain decisions which are to be made sometime during the whole process and once they are made, the further progression will be based on those decisions, to ensure that the crucial decisions are not uncertain, they should be delayed as long as possible to avoid any unnecessary work.
4. Deliver as soon as possible - Customers value fast delivery of quality products, the aim should be to deliver a functioning product without any defects as fast as possible to get feedback from customers and start working on the next iteration.
5. Empower the team - The managers should respect and trust the developers on making the right decisions and not micro manage them. The developers shouldn’t be seen just as a resource, but as an individual who needs motivation and a higher sense of purpose to continue working at an excellent level.
6. Build Integrity in - Customers need to be informed and should be up with the progress and each step of development, they should be given information in a regular manner and in an easy to understand, yet detailed manner. Large documentation should be avoided for this purpose.
7. Optimize the whole - The final product is a system which contains many small units and subunits. During the final testing, the product should be optimized as a whole rather than working individually on the units.
8. Extreme programming (XP)

Extreme Programming is a software development methodology which comes under Agile Development. It aims to provide a high quality product and cater to changing requirements of the customer and also aims to improve the quality of life of the developers. Of all the agile development frameworks, extreme programming is the most specific one regarding appropriate programming practices. It works on interactions of releases of software to incorporate new requirements with each checkpoint. Extreme programming takes the beneficial practices of software development to the extreme.

There are 4 main activities in Extreme Programming:

1. Coding - The actual activity of a software development process.
2. Testing - It is the central focus of extreme programming. It eliminates the flaws of the product, a lot of testing means a lot of flaws are eliminated. Unit integration testing takes place frequently which tests a feature. Acceptance tests are used to verify fulfillment of customer’s requirements. System wide integration testing is also performed regularly.
3. Listening - This refers to the requirement elicitation process which programmers must go through with their customers to deliver them the best possible product for their needs.
4. Designing - For the long term functioning and maintenance of a product, good design is essential. There shouldn’t be complex dependencies between in the product, which might make future revisions or feature additions too expensive and time consuming.

There are 5 main values of Extreme Programming:

1. Communication - Software development is a process which is carried out by a group of people, hence, good and effective communication is essential. Communication involving explaining mechanisms through usage of white boards or presentations are an effective way to communicate one team member’s knowledge to the whole team.
2. Simplicity - The purpose is to avoid wasting time and resources on unnecessary things. The simplest solution of a problem is usually the best solution, as this makes current development easier as well as makes future development and maintenance easier.
3. Feedback - Constant feedback about ongoing and past development is essential to improve the quality of product. It also helps to improve the design of the whole implementation.
4. Courage - This refers to the ability of a team member to face the tough decisions and make them with the product and development team’s best interest in mind. It also means to have the courage to stop doing something which isnt going to be effective and move on to a better option.
5. Respect - The members of the team need to respect each other to communicate effectively and trust them for working together.
6. Adaptive Software Development (ASD)

Adaptive Software Development is a software development methodology evolved from Rapid Application Development method. ASD replaces the traditional waterfall cycle method with a series of Speculate, Collaborate and Learn. This provides for continuous improvement and learning. The ASD life cycle is feature driven, mission focused, iterative and time limited. It is dynamic to adapt to changes in requirements. It focuses on results, not tasks.

The ASD life cycle is divided into 3 phases:

1. Speculate - When the plan is determined and is certain of some specific outcomes and strategies, it is hard to innovate. Therefore, the term plan is replaced by speculate. This allows the team to view the uncertainty of complex problems and helps to come up with new and innovative solutions.
2. Collaborate - This basically refers to the communication which is essential for the development of a complex product. The amount of information which flows around the development team is large and for proper evolution of complex products, collaboration is needed between team members.
3. Learn - the learn part refers to enhancement of the knowledge of team members using practices like Technical Reviews, Project Retrospectives and Customer Focus Groups.
4. Feature Driven Development

Feature Driven Development is an agile software development methodology. It is an incremental and iterative development process. The aim is to create a tangible and functional software in a timely manner. Feature Driven Development is an architecture-centric short iteration process which consists of 5 basic activities. There are milestones set for each feature to track the progress of the whole project.

The five basic activities are:

1. Develop an Overall Model
2. Build a Feature List
3. Plan by feature
4. Design by Feature
5. Build by Feature

The first 2 activities give a high level understanding of the project and the overall shape of the model is specified. The last 3 activities are iterated over for each feature.

A collection of supporting roles are defined in FDD:

* Domain Manager
* Release Manager
* Build Engineer
* Toolsmith
* System Administrator
* Tester
* Deployer
* Technical Writer