Write short notes on following

**SCRUM**

Scrum works well for long-term, complex projects that require stakeholder feedback, which may greatly affect project requirements. So, when the exact amount of work can’t be estimated, and the release date is not fixed, Scrum may be the best choice.

Scrum team: 5 to 9 people

This team does not include any of the traditional software engineering roles such as programmer, designer, tester or architect. Everyone on the project works together to complete the set of work they have collectively committed to complete within a sprint.

Product owner: The product owner is often someone from product management or marketing, a key stakeholder or a key user.

Scrum master: The Scrum Master is responsible for making sure the team is as productive as possible. The Scrum Master does this by helping the team use the Scrum process, by removing impediments to progress, by protecting the team from outside, and so on.

Product backlog: The product backlog is a prioritized features list containing every desired feature or change to the product.

Sprint planning meeting: At the start of each sprint, a sprint planning meeting is held, during which the product owner presents the top items on the product backlog to the team. The Scrum team selects the work they can complete during the coming sprint. That work is then moved from the product backlog to a sprint backlog,

Daily Scrum: Each day during the sprint, a brief meeting called the daily scrum is conducted. This meeting helps set the context for each day’s work and helps the team stay on track.

Sprint review meeting: At the end of each sprint, the team demonstrates the completed functionality at a sprint review meeting, during which, the team shows what they accomplished during the sprint.

Sprint retrospective: Also at the end of each sprint, the team conducts a sprint retrospective, which is a meeting during which the team (including its ScrumMaster and product owner) reflect on how well Scrum is working for them and what changes they may wish to make for it to work even better.

Sprint: a Sprint, a time-box of one month or less during which a “Done”, useable, and potentially releasable product increment is created. Usually, Sprints are limited to one calendar month.

**LEAN DEVELOPMENT**

Lean Software Development (LSD) is an [agile framework](https://www.productplan.com/glossary/agile-framework/) based on optimizing development time and resources, eliminating waste, and ultimately delivering only what the product needs. The Lean approach is also often referred to as the [Minimum Viable Product (MVP)](https://www.productplan.com/resources-pp/mvp-truly-compelling/) strategy, in which a team releases a bare-minimum version of its product to the market, learns from users what they like, don’t like and want to be added, and then iterates based on this feedback.

LSD actually borrows its philosophy from the manufacturing industry, which originated the lean development process as a way to optimize production and assembly lines to minimize waste and maximize customer value. In fact, it was originally called the Toyota Production System, because automaker Toyota invented this approach in the middle of the twentieth century as a way to streamline its production of cars and eliminate wasted time and resources. (Any action that did not impact the functionality of the car being built and delivered was considered a waste under this system, and therefore removed from the process.)

Eventually, other manufacturing organizations across many industries began using this system, and the name later changed to Lean. The methodology was first applied to the creation of software in 2003 with the publication of the now-famous book Lean Software Development.

## What are the Strengths and Weakness of Lean Software Development?

### LSD’s strengths include:

* Streamlined approach allows more functionality to be delivered in less time
* Eliminates unnecessary activity, and as a result can reduce costs
* Empowers the development team to make decisions, which can also boost morale

### LSD’s weaknesses include:

* Heavily depends on the team involved, making it not as scalable as other frameworks
* Depends on strong documentation, and failure to do so can result in development mistakes

**EXTREME PROGRAMMING**

**VALUES OF XP**

The five values of XP are communication, simplicity, feedback, courage, and respect and are described in more detail below.

Communication

Software development is inherently a team sport that relies on communication to transfer knowledge from one team member to everyone else on the team. XP stresses the importance of the appropriate kind of communication – face to face discussion with the aid of a white board or other drawing mechanism.

Simplicity

Simplicity means “what is the simplest thing that will work?” The purpose of this is to avoid waste and do only absolutely necessary things such as keep the design of the system as simple as possible so that it is easier to maintain, support, and revise. Simplicity also means address only the requirements that you know about; don’t try to predict the future.

Feedback

Through constant feedback about their previous efforts, teams can identify areas for improvement and revise their practices. Feedback also supports simple design. Your team builds something, gathers feedback on your design and implementation, and then adjust your product going forward.

Courage

Kent Beck defined courage as “effective action in the face of fear” (Extreme Programming Explained P. 20). This definition shows a preference for action based on other principles so that the results aren’t harmful to the team. You need courage to raise organizational issues that reduce your team’s effectiveness. You need courage to stop doing something that doesn’t work and try something else. You need courage to accept and act on feedback, even when it’s difficult to accept.

Respect

The members of your team need to respect each other in order to communicate with each other, provide and accept feedback that honors your relationship, and to work together to identify simple designs and solutions.

**Adaptive Software Development (ASD)**

Adaptive Software Development (ASD) is a direct outgrowth of an earlier [agile framework](https://www.productplan.com/glossary/agile-framework/), Rapid Application Development (RAD). It aims to enable teams to quickly and effectively adapt to changing requirements or market needs by evolving their products with lightweight planning and continuous learning. The ASD approach encourages teams to develop according to a three-phase process: speculate, collaborate, learn.

## Strengths and Weakness of ASD

### ASD’s strengths include:

* Focused on the end users, which can lead to better and more intuitive products
* Allows for on-time and even early delivery
* Encourages more transparency between developers and clients

### ASD’s weaknesses include:

* Demands extensive user involvement, which can be difficult to facilitate
* Integrates testing into every stage, which can add to a project’s costs
* Emphasis on rapid iterating and continuous feedback can lead to scope creep

· **Feature Driven Development**

Feature Driven Development (FDD) is an [agile framework](https://www.productplan.com/glossary/agile-framework/) that, as its name suggests, organizes software development around making progress on features. Features in the FDD context, though, are not necessarily product features in the commonly understood sense. They are, rather, more akin to [user stories](https://www.productplan.com/break-product-features-into-user-stories/) in [Scrum](https://www.productplan.com/scrum-agile-framework/). In other words, “complete the login process” might be considered a feature in the Feature Driven Development (FDD) methodology.

FDD was designed to follow a five-step development process, built largely around discrete “feature” projects. That project lifecycle looks like this:

1. Develop an overall model
2. Build a features list
3. Plan by feature
4. Design by feature
5. Build by feature

The framework has since gained widespread use particularly in larger organizations, and today there is a [thriving Feature Driven Development community](http://www.featuredrivendevelopment.com/) with its own website.

## Strengths and Weakness of Feature Driven Development

### FDD’s strengths include:

* Simple five-step process allows for more rapid development
* Allows larger teams to move products forward with continuous success
* Leverages pre-defined development standards, so teams are able to move quickly

### FDD’s weaknesses include:

* Does not work efficiently for smaller projects
* Less written documentation, which can lead to confusion
* Highly dependent on lead developers or programmers