1. Maven Life Cycle (5 Marks)

Maven has three main build life cycles:

- Clean: Cleans the target directory (compiled files).
- Default (Build): Handles project deployment (compilation, testing, packaging, etc.).
- Site: Creates project documentation.

Each life cycle contains phases (e.g., validate, compile, test, package, install, deploy), executed in order.

2. What is pom.xml and Why We Use It? (5 Marks)

- pom.xml stands for Project Object Model.
- It's the core configuration file for a Maven project.
- Defines project info (name, version), dependencies, plugins, build settings.
- Ensures consistent builds across environments.

3. How Dependencies Work? (5 Marks)

- Dependencies are external libraries/tools needed for your project.
- Declared in pom.xml with groupId, artifactId, and version.
- Maven downloads them from central or remote repositories and stores them in local .m2 repository.
- Reduces manual effort in library management.

4. Check the Maven Repository (5 Marks)

- Central Repository: https://mvnrepository.com
- You can search for artifacts by name, groupId, etc.
- Add the dependency snippet directly to your pom.xml.
- Maven pulls required JARs automatically.

5. How All Modules Build Using Maven? (5 Marks)

- Multi-module projects are defined in a parent pom.xml.

- Submodules listed under <modules> tag.

- Building the parent with mvn install builds all submodules in defined order.

- Ensures cohesion across all parts of the application.

6. Can We Build Specific Module? (5 Marks)

- Yes, using the command:

```
  mvn install -pl <module-name> -am
```

  - -pl: Specifies project to build.

  - -am: Builds dependencies required by the specified module.

7. Role of ui.apps, ui.content, and ui.frontend Folder? (5 Marks)

- ui.apps: Contains code deployed under /apps, like components, templates, and configurations.

- ui.content: Contains code deployed under /content, like content structure, policies, and pages.

- ui.frontend: Contains front-end resources (JavaScript, CSS, etc.), often built using Node.js/Webpack.

8. Why We Are Using Run Modes? (5 Marks)

- Run modes are used to differentiate environments (e.g., dev, stage, prod).

- Allows deploying different configurations or content per environment.

- Example: config.dev, config.prod under /apps/project/config.

9. What is Publish Environment? (5 Marks)

- AEM environment that serves content to the end-users.

- No authoring capabilities.

- Replicated content from the Author environment gets published here.

- Optimized for speed and scalability.

10. Why We Are Using Dispatcher? (5 Marks)

- A caching and load balancing tool for AEM.

- Increases performance by caching static content.

- Protects AEM publish instance from unnecessary load.

- Filters and serves requests efficiently.

11. From Where Can We Access crx/de? (5 Marks)

- CRX/DE Lite is accessed via browser:

  http://localhost:4502/crx/de

- It is a web interface to the JCR repository.

- Used to view/edit nodes, properties, and content structure.

- Only accessible in Author environment (default port 4502).