1.What is the purpose of the core module in AEM?

Ans: It contains the backend java logic, services, business logic etc. It is responsible for data processing.

2.What kind of files and code can be found in the core folder?

Ans: Java Classes, servlets and sling modules.

3.Explain the role of ui.apps in AEM projects.

Ans: it is responsible for defining structure and it contains frontend resources like components, templates, dialogs and libraries.

4.How are components structured in the ui.apps folder?

Ans: Components are structured under /apps//components/. Each component have htl, js and .xml for editing properties and a .content.xml file

5.Hello World Component:

- Where is the Hello World component located in both core and ui.apps?

  Ans: In the core module:
  /core/src/main/java/com/example/models/HelloWorldModel.java
  In the ui.apps module: /apps/example/components/helloworld/

- Explain the Java class (in core) for the Hello World component.

  Ans: The java class retrieve and process the data for the component. It uses annotations to map AEM properties to java fields.

- How does the HTL script work in ui.apps for Hello World?

  Ans: HTL (HTML template language) retrieve and display values from the java model.

- How are properties and dialogs defined for this component?

  Ans: Dialogs are defined in dialog.xml file using Granite UI components and properties are stored in JCR nodes and accessed by Sling models.

6.What are the different types of AEM modules (core, ui.apps, ui.content, etc.)?

Ans:

1. all - aggregates all modules into a deployable package.
2. core - contains Java code and business logic.
3. ui.apps - holds components, templates, and client libraries.
4. ui.content - stores content structures like pages and configurations.
5. it.tests - integration tests.

7.How does Maven build these modules?

Ans: It uses POM hierarchy to compile java code, package components and generate aem packages for development purpose.

8.Explain the build lifecycle of Maven in the context of AEM.

Ans: Life cycle of maven is validate, compile, test, install and deploy. For aem it compiles java code, assembles the package content and deploys them.

9.How are dependencies managed in pom.xml?

Ans: the <dependencies> tag is used for dependencies, also by using groupId, artifactId and version tags.

10.Why is Maven used instead of other build tools?

Ans: It automates dependency management, enforces a standardized project structure, and integrates well with aem's package management system.

11.What advantages does Maven offer for AEM development?

Ans: supports multi-module projects, integration with aem, content packaging and automated building and deployment.

12.How does Maven help in managing dependencies and plugins in AEM projects?

Ans: It fetches dependencies from repositories and configures aem specific plugins for building and deploying.

13.What does mvn clean install do in an AEM project?

Ans: It clears log and cleans the project, compile the code, run tests and install the package in the local maven repository.

14.How to deploy packages directly to AEM using Maven commands?

Ans: by using mvn clean install -PautoInstallPackage to deploy the package to aem.

15.Explain the purpose of different Maven profiles in AEM (autoInstallPackage, autoInstallBundle).

Ans: autoInstallPackage: deploys the full content package to aem.

autoInstallBundle: deploys only the OSGi bundle (Java code) without affecting content

16.What is the purpose of dumplibs in AEM?

Ans: used for debugging the code and analyzing the client libraries. Helps in identifying dependencies, paths of client libraries.

17.How can you view client libraries using dumplibs?

Ans: by using http://localhost:4502/libs/granite/ui/content/dumplibs.html to see the structure and dependencies of client libraries.

18.Explain how client libraries are structured in AEM.

Ans: client libraries are stored under /apps/project/clientlibs/. They are categorized using categories, dependencies, and embed properties to manage CSS, JS, and assets efficiently.