1. What is IOC?

Ans : IOC is to Invert the control flow of a program. It is an Idea to Decouple or remove dependencies of the object and provide control to another layer(XML, Java Class, Properties,…) to instantiate required Objects. To achieve IOC we can use many proven design principles like Service Locator and Dependency Injection.

1. What is dependency injection?

Ans : It is a Design pattern with the idea to separate object creation logic to a specific layer and inject it to the required class.

Dependency Injection pattern will change binding on Objects at Compile time to Run time. Hence we can reusable the class more than before. Unit testing also easy to perform.

There are three form of Dependency Injection :

1.Constructor Injection

2.Setter Injection

3.Interface Injection

1. What is Autowiring?

Ans : Autowiring feature of spring framework enables you to inject the object dependency implicitly. It internally uses setter or constructor injection. Autowiring can't be used to inject primitive and string values. It works with reference only.

1. Difference between pojo and bean.

Ans : POJO refers to Plain old Java object

The only difference between both the classes is Java make java beans objects serialized so that the state of a bean class could be preserved in case required. So due to this a Java Bean class must either implements Serializable or Externalizable interface.

Thus it is stated that all JavaBeans are POJOs but not all POJOs are JavaBeans.

1. Difference between jar and war.

Ans : JAR or Java Archive is a package file format. JAR files have the .jar extension and may contain libraries, resources, and metadata files.

Essentially, it's a zipped file containing the compressed versions of .class files and resources of compiled Java libraries and applications. We can [create a JAR](https://www.baeldung.com/java-create-jar) file using the jar command or with tools like [Maven](https://www.baeldung.com/executable-jar-with-maven).

WAR stands for Web Application Archive or Web Application Resource. These archive files have the .war extension and are used to package web applications that we can deploy on any Servlet or JSP container.

We can use the same tools and commands that we used to build a JAR to build a .war archive.

The second main difference is their purpose and the way they function. JAR files allow us to package multiple files in order to use it as a library, plugin, or any kind of application. On the other hand, WAR files are used only for web applications.

The structure of the archives is also different. We can create a JAR with any desired structure. In contrast, WAR has a predefined structure with WEB-INF and META-INF directories.

Finally, we can run a JAR from the command line if we build it as an [executable JAR](https://www.baeldung.com/executable-jar-with-maven) without using additional software. Or, we can use it as a library. In contrast, we need a server to execute a WAR.

1. What is Maven and Gradle?

Ans : Maven, or [Apache Maven](https://www.educative.io/blog/apache-maven-tutorial), is an XML-based build tool and project manager. Maven is an Apache open-source project. Its default repository is the Maven Central Repository. Maven projects are primarily defined by [Project Object Model (POM)](https://www.educative.io/blog/apache-maven-tutorial#pom) files written in XML. These POM.xml files contain the project’s dependencies, plugins, properties, and configuration data. Maven uses a declarative approach and has a predefined life cycle.

Gradle was build on Maven’s concepts, introduced as Maven’s successor. It was designed with multi-project builds in mind. Gradle’s build script is inherently more versatile and powerful than Maven’s. This is because Gradle is based on a programming language (Groovy), while Maven’s is based on a markup language (XML).