1) What is Spring?
   Spring is a framework for building java applications. It provides comprehensive infrastructure support for developing applications that help developer create high performance, easily testable, and reusable code. With spring, developers can build applications from Plain Old Java Objects (POJO) and then apply enterprise services to those POJOs.

2) What is Spring Boot?
   Spring Boot is a layer built on top of Spring that auto-configures your application. Spring required a lot of configurations (XML documentations), application server (like Tomcat) had to be manually set up. Hence, a module that added auto configurations, embedded server and built in actuators was built on top of Spring known as Spring Boot.

3) What is the relation between Spring platform and Spring Boot?
   Spring Boot is a part of the larger Spring Platform. The Spring Platform refers to the entire ecosystem of Spring projects such as Spring Core, Spring MVC, Spring Data, Spring Security, and more. It provides all the essential building blocks for creating enterprise applications using POJOs but requires manual setup and configuration.

   Spring Boot, built on top of the Spring Platform, simplifies this process by combining many Spring modules, adding auto-configuration, and enabling rapid application development with minimal boilerplate.

4) What is the relation between Spring platform and Spring framework?
   Spring framework is the foundation or the core module of the spring ecosystem. Spring framework provides core features like dependency injection, inversion of control, spring MVC, spring context, aspect-oriented programming etc.

   Spring platform on the other hand is an umbrella term for the entire spring ecosystem which includes spring framework, spring boot, spring data, spring cloud and many more.

5) What is Dependency Injection and how is it done in the Spring platform/framework?
   Dependency Injection is a design pattern in which an object receives its dependencies (like other objects) from an external source or container rather than creating them itself. This helps objects to achieve loose coupling between them.

Spring manages objects (called Beans) and injects dependencies automatically through 3 methods:

a) Constructor Injection – dependencies are provided through class constructors.
b) Setter Injection – dependencies are provided through setter method and are useful in case of optional dependencies.
c) Field Injection - dependencies are injected directly into fields. This is however avoided since it makes testing harder and hides dependencies.
d)

6) What is Inversion of Control (IoC) and how is it related to Spring?
Inversion of Control (IoC) is a software design principle where the control flow of a program is inverted—instead of the application code managing object creation and dependencies, an external framework (like Spring) takes control.

Spring is an IoC container as it manages your objects (called beans). IoC is a concept, and spring is an implementation of it. Spring implements IoC by scanning components, creating and managing them, injecting their dependencies and destroying them when no longer required when the application starts. A specific way to achieve IoC is by injecting dependencies (DI).