What is IOC (inversion of control) Container ?  
Ioc is a principle or paradigm. we have some set of rules or guidelines to develop a application in a decouple manner.  
Ioc is a collabrating the object and managing the lifecycle of those objects is called ioc container.  
Ioc container says that you don't bother about object creation or you don't create your objects only describe how they should be created i will manage it.  
The basic concept of the Dependency Injection or Inversion of Control is that, programmer do not need to create the objects, instead just describe how it should be created.  
Benefits of Ioc:  
--------------------  
Minimize the code in our application.  
It provides loose coupling between components in our application.  
If any modification have to do then it doesnt effect other components.  
It supports eager instantiation and lazy instantiation of services.  
Enhancement will be easy.  
2. What is dependency injection? Types of dependency injection?  
Dependency injection is a software design pattern that deals with how components are organizing their dependencies.  
It is the process of injecting the dependencies in dependent class automatically. We no need to create obj and no need to map with obj.  
Dependency Injection means injecting the dependency between two object as per as our requirement in our application, this help to reducing the dependency to each other.  
Dependency Injection (DI) is a design pattern that removes the dependency from the programming code so that it can be easy to manage and test the application.  
Dependency Injection makes our programming code loosely coupled  
It is internally follow strategy design pattern means favour composition over inheritance.  
Always design to interface never code to implementation.  
3. Difference between setter injection and constructor injection?  
Setter Injection:  
To perform setter injection we will use <property> tag.  
Partial dependencies is possible. means if we have 3 dependencies like int, long, String it is not necessary to inject all values.  
If we have more dependencies eg 15 to 20 are there in our bean class then in this case setter injection is not recomonded to use as we need to write almost 20 setters right bean lenght will be increased.  
Setter injection makes bean class object as mutable( we can change).  
Setter injection support cyclic dependencies.  
Constructor injection:  
To perform constructor injection <constructor-arg> tag is required.  
Partial injection of dependencies cannot be possible because for calling a constructor we must pass all the arguments.  
if we have more dependencies in this case constructor injection is highly recomonded to use because we can inject all the dependencies with in the 3 to 4 lines (by calling one constructor).   
Constructor injection makes bean class obj is immutable( we cannot change).  
Constructor injection doesn't support cyclic dependencies.  
4. Difference between BeanFactory and ApplicationContext?  
BeanFactory:

BeanFactory is a basic container. it can only manage a bean life cycle. but it can not provide service like transaction, security etc.  
If we developing small scale application like mobile application embeded system then we use beanfactory.  
Beanfactory is lazy initializer. beanfactory container will not create a bean obj upto the request time.  
Beanfactory container supports only two scope(singeltone & prototype).  
Beanfactory doesnt support internationalization, event handling, event processing.  
Application Context:

ApplicationContext is a advanced container it manage bean life cycle and also provide transaction security etc.  
If we are developing enterprise application like(web application, distributed application) then ApplicationContext is recomended to use.  
ApplicationContext container creates bean object of singelton bean at the time of loading only.it is eager initialzer.  
ApplicationContext container support all the bean scope (singletone,prototype, session,request).  
It supports internationalization, event handling, event processing also.  
5. What is bean autowiring and types of autowire modes?  
Injecting the dependencies between the obj is called wiring.  
Instead of telling the spring to manage the dependency by writting <property> or <constructor> tag in spring bean configuration file.  
If we instruct the spring to automatically detect the dependencies and perform the injection between them it is called bean autowiring.  
It is used only when rapid application development is required.  
In Spring framework, you can wire beans automatically with auto-wiring feature. To enable it, just define the “autowire” attribute in.  
The Spring container can autowire relationships between collaborating beans without using and elements which helps cut down on the amount of XML configuration.  
<bean id="countryBean" class="org.arpit.java2blog.Country" autowire="byName">

Modes of Autowire:  
1. Autowire= "byname"

If u enable autowiring byname, spring will inject the bean based on property name. it uses setter method.  
Autowiring by property name. Spring container looks at the properties of the beans on which autowire attribute is set to byName in the XML configuration file and it tries to match it with name of bean in xml configuration file.

2. Autowire= "bytype"  
If u enable autowire bytype, spring will inject the beans based on the property type. it uses setter method.  
Autowiring by property datatype. Spring container looks at the properties of the beans on which autowire attribute is set to byType in the XML configuration file.  
It then tries to match and wire a property if its type matches with exactly one of the beans name in configuration file

3. Autowire= "byconstructor"  
If u enable autowire byconstructor, spring will injects the beans uses constructor.  
byType mode in constructor argument.

4. Autowire="byautodetect"

Spring first tries to wire using autowire by constructor, if it does not work, Spring tries to autowire by byType.  
6. What is bean scope? and types of bean scope. Difference between singleton and prototype bean scope?  
Beanscope is a concept which is provided by spring people. in spring when u declare a class as a bean by default the bean will be created under the singleton scope.

Types of beanscope:  
1. Singleton

bydefault every bean declared in the configuration file is singleton.  
Scopes a single bean definition to a single object instance per Spring IOC container.  
Singleton is default scope of a bean in Spring. You have to explicitly change scope of a bean if you want different scope.  
If bean scope is singleton then Ioc container creates the bean class obj and keeps in the HashMap element as value by having bean 'id' as key and uses that obj across the multiple "factory.getBean() method.

2. Prototype

Prototype – Return a new bean instance each time when requested.  
When we declare a bean scope as a prototype then Ioc container doesn't keep the created bean class object in HashMap so it returns new obj for every factory.getBean().

3. Request

Request – Return a single bean instance per HTTP request.  
When we declare a bean scope as request ,for every Http request new bean instance will be injected.

4. Session

Session – Return a single bean instance per HTTP session.  
For every new Http session ,new bean instance will be injected.

5. Global session  
The global session scope is deprecated in the market from spring 3.0.   
Return a single bean instance per global HTTP session.

7. what is bean life cycle?  
-------------------------------------------  
-> every obj in this world have life cycle. whatever the obj is performing after the birth and before the death is known as life cycle of the obj.  
-> the spring container find the beans defination from the xml file and instantiate the bean .  
-> using dependency injection spring populates all of the properties as specified in the bean defination.  
-> in servlet life cycle we are used following life cycle methods . 1> init() 2> service() 3> destroy().  
-> spring bean allows two life cycle methods. 1> init() 2> destroy().

Spring AOP:  
1. What is Aop ? What is the principles of Aop? and where we apply Aop in projects?

Aop is not a programming language. it is a methodlogy or principles like oops. we have some set of rules or guidelines to make our application in decouple manner.  
Aop is the process of separating the primary logic from the secondry logic (crosscutting logic).  
in every application there will be two types of logic, one is called primary buisness logic and other one is helper logic which makes your primary business logic work better.  
In the enterprise level application we used to add diff crosscutting functionlaties (means adding diff types of services to the application at runtime).