Q #1) what is a Framework? Explain the Spring Framework.

Answer: Framework is already built software that helps the developers to add more functionality to their applications by using already built classes and libraries.
Spring Framework is a dependency injection framework, which helps the developers to develop applications based on Java. Spring is an open source framework for Java Enterprise.

Q #2) why is Spring preferred over the other Frameworks?

Answer: Spring is preferred over the other frameworks due to the below features.

- Very small size and lightweight
- Helps to achieve loosely coupled application by Inversion of Control.
- AOP support
- MVC framework
- Good Transaction Management feature
- Spring container
- Exception handling

Q #3) Categorize the different types of modules in Spring Framework.

Answer: The five main modules in Spring Framework are mentioned below:

1. Spring Core Container which includes core, beans, context, and expression language.
2. AOP and Aspects.
3. Data Integration Module which includes JDBC, OXM, ORM, transaction modules etc.
4. Web Module that includes web, servlet, struts, and portlet.
5. Test

Q #4) Explain the Spring Configuration File.

Answer: An XML file contains all the information about classes along with their configuration process and how these classes have interacted with the other classes.

Q #5) Explain the concept of Dependency Injection.

Answer: Firstly, Injection means transferring the dependency to any dependent object. Dependency injection is a kind of design pattern that helps to develop the loosely coupled application. It is an implementation of inversion of control that helps in solving dependencies in an application. It avoids self-object creation and rather guides how objects should be created through configurations and then the IOC decides which services need to be matched by which components. Dependency Injection is the core feature of Spring Framework.

Q #6) what are different types of Dependency Injection? Explain them.

Answer: There are two types of dependency injection. They are as follows:
(i) Construction Based Dependency Injection– It is achieved when the Spring container invokes a constructor with arguments and each having some dependency on the other class.
(ii) Setter-based Dependency Injection- It is achieved when the Spring container calls the setter method of beans after invoking constructor with no arguments to instantiate the bean.

Q #7) which is good to use – Constructor or Setter-based dependency?

Answer: Both types of dependency injection can be used accordingly based on the situations. It is a thumb rule, that for mandatory dependency, constructor based dependency injection is used while for optional dependency, setter-based dependency injection is used.

Q #8) what are the advantages of Inversion of Control?

Answer: There are several advantages of IOC, and few among them are mentioned below:

1. IOC is capable of reducing code complexity by reducing code in the application.
2. By using IOC in the application, testing becomes more simple and easy as no lookup and singletons are required.
3. Loose coupling is achieved by IOC and that, in turn, makes the code more maintainable.
4. IOC supports early instantiation and late loading of services.

Q #9) Explain the concept of AOP.

Answer: AOP stands for Aspect Oriented Programming. It is another approach of programming that helps the developers to restructure the behavior of responsibilities like Transaction Management and logging.
AOP is implemented for cross-cutting concerns, i.e. the definition is provided in one place and functionally it can be used in many places with the help of the script.

Q #10) Explain the use of Spring Container.

Answer: It is the core part and backbone of Spring framework. The Spring container helps to create objects, combine the objects together, manage their configurations and complete the life cycle of creation, implementation, and destruction.
The Spring container takes the bits of the help of Dependency Injection to manage the components which build up the application.

Q #11) what are the different types of IOC container?

Answer: IOC container is of two types as mentioned below:

1. **Bean Factory**– It is a simple container and provides support for dependency injection.

2. **Spring ApplicationContext**– It is an advanced container that adds on complex features like decode textual message from files and is capable of publishing events to the listeners.

Q #12) What is the implementation of the Bean Factory container?

Answer: XmlBeanFactory class is the most important implementation of the bean factory and is useful for reading data from the XML files.

Q #13) what is the implementation of ApplicationContext container?

Answer: Some of the most commonly used ApplicationContext containers include FileSystemXmlApplicationContext, ClassPathXmlApplicationContext, and WebXmlApplicationContext etc.

Q #14) what are beans in Spring?

Answer: Spring Beans are nothing but simple Java objects that are managed by the Spring container.

Example:

```
package com.javaworld;

public class Demo {

private String message;

public void setMessage(String message){

this.message = message;

}

public void getMessage(){

System.out.println("Display Information: " + message);

}

}
```

Q #15) what are the components in Bean Definition?

Answer: Basically, bean definition holds the configuration metadata that is used by the Spring container to know details like, bean creation process, the life cycle of bean and dependencies of a bean.

Q #16) what are the ways by which configuration metadata can be provided to the Spring container?

Answer: The configuration metadata can be provided to the Spring Container in three ways i.e. through XML based configuration file, annotation based configuration and Java-based configuration.

Q #17) what is the syntax to add a bean in the Spring application?

Answer:

```
<? xml version ="1.0" encoding ="UTF-8"?>

<beansxmlns="HTTP://www.Springframework.org/schema/beans"xmlns:xsi="HTTP://www.w3.org/2001/XMLSchema-
instance"xsi:schemaLocation="HTTP://www.Springframework.org/schema/beans

HTTP://www.Springframework.org/schema/beans/Spring-beans.xsd">

<beanid="helloWorld"class="com.demoprogram.HelloWorld">

<propertyname="info"value="Hello World!"/>

</bean>

</beans>
```

Q #18) what are the bean scope types?

Answer: Bean scope can be defined as singleton and prototype, request, session, global-session etc.
A prototype is declared when a new bean instance is required every time whereas singleton is declared when the same bean instance is used every time.

A request is used for HTTP request scope, the session is used for HTTP session scope, and global-HTTP session scopes the bean to the global HTTP session.

Syntax: <bean>

Q #19) which is the default scope of bean in Spring? In addition, are they synchronized?

Answer: The default scope of bean in Spring is Singleton. Moreover, they are not synchronized i.e. they do not thread safe.

Q #20) what is the Life Cycle of a Bean in Spring?

Answer: The life cycle of a Spring Bean follows certain steps, as mentioned below:

1. Instantiation – Spring container from the XML file finds the bean definition and then the bean is instantiated.
2. Populate properties – Spring container populates all the properties mentioned in the bean definition with the help of dependency injection.
3. Setting the name of Bean.
4. Setting bean factory.
5. Pre-initialization and initialization of bean.
6. Post initialization of bean.
7. Destroy the bean by calling destroy () method.

Q #21) what do you mean by inner beans?

Answer: The bean which is defined inside the property or constructor element is called an inner bean.
No specific ID or name is required for inner bean, in fact, the Spring container avoids those values along with the scope definition. Inner beans are said to be anonymous and their scope is always defined as prototypes.

Q #22) what are the ways to insert the collection concept in Spring?

Answer: Basically, there are four collection elements to insert in Spring.
They are:

1. **<set> element** – It wires the set of values by eliminating the duplicates from them.
2. **<list> element** – It is useful to insert or inject values and also allows duplicate values.
3. **<map> element** – It is used to insert a key or name-value pair which can be of any type.
4. **<props> element** – It is used to insert key or name-value pair, but the type should only be the string.

Q #23) explain the concept of Auto wiring.

Answer: Auto-wiring is an essential concept in Spring framework. It is used to implicitly inject object dependency by use of a setter or constructor based injection. Auto-wiring works with reference types only, so it is not useful for injecting values for primitive and string types.

The best advantage of auto wire is that the developers need to write less code as dependency injection is taken care by auto wire. Also, the programmer has not controlled over the process then.

Syntax: <bean id="customer" class="com.sthelp.same.Customer" autowire="byName" />

Q #24) explain the different modes of Auto wiring.

Answer: Auto wiring comes with five modes, which guide the Spring container for using Dependency Injection feature.
(i) No mode – It is called to be the default setting and implies auto wiring disabled and an explicit bean should be used for reference wiring.
(ii) byName – Auto wiring can be done by property name. Spring container searches the XML configuration file for the beans whose autowire attribute is set to byName.
(iii) byType – Auto wiring can be done by property type. Spring container searches the XML configuration file for the beans whose autowire attribute is set to byType.
(iv) Constructor – It is almost same to byType but the type is applicable for constructor arguments if no constructor argument type is found in the container then an error is thrown.
(v) Autodetect – It refers to the behavior of Spring showing its choosing priority. Firstly, Spring chooses auto wiring using the constructor, if it does not happen then it changes the priority to byType.

Q #25) is there any limitation of auto wiring? If yes, explain.

Answer: Yes, there are some limitations of Auto wiring which are as mentioned below.

1. There is always a possibility of overriding.
2. The developer will not be able to auto-wire primitive and Spring properties.
3. Auto-wiring becomes complex when used in big applications compared to explicit wiring.

Q #26)what is annotation wiring and how do we turn it on?

Answer: The alternative use of XML is annotations in which the developers move the entire configuration in one class with the help of annotations for a particular class or method.
It is turned on in the Spring Configuration file by declaring <context:annotation-config/>.

Q #27) what is the use of @Required annotation?

Answer: It is used to indicate that at configuration time, the bean property should be populated through auto wiring or explicit property value in the bean definition.
Example:

```
package com.softwaretestinghelp;

import org.Springframework.beans.factory.annotation.Required;
```

```
public class Employee {

private Integer age;

private String Lname;

@Required
public void setAge(Integer age) {
this.age = age;
}
public Integer getAge() {
return age;
}
@Required
public void setLName(String Lname) {
this.name = name;
}
public String getLName() {
return name;
}
}
```

Q #28) what is the use of @Autowired annotation?

Answer: It helps to get better control and understanding of how and where auto wiring should be achieved. It can be used to the autowire bean on any setter method, property or constructor.

Example:

```
package com.softwaretestinghelp;

import org.Springframework.beans.factory.annotation.Autowired;

public class code-editor {

private SpellCheck spellCheck;

 @Autowired
public void setSpellCheck( SpellCheck spellCheck ){
this.spellCheck = spellCheck;
}
public SpellCheck getSpellCheck( ) {
return spellCheck;
}
public void spellCheck() {
spellCheck.checkSpelling(); }}
```

Q #29) what is the use of @Qualifier annotation?

Answer: It is mainly used when the developer is bound to create many beans of the same type and want to wire only one of them with the property, in this scenario @Qualifier with @Autowired is used for removing confusion and specifying the exact bean to be wired.

Q #30) how is an event handled in Spring?

Answer: Event handling is achieved through ApplicationEvent class and ApplicationListner interface.
When the bean implements ApplicationListner then ApplicationEvent gets generated to ApplicationContext and notifies that the bean is generated.

Q #31) what are the examples of standard Spring Events?

Answer: There are many standard Spring events serving respective purpose and few among them are ContextStartedEvent, ContextRefreshedEvent, ContextStoppedEvent, ContextClosedEvent, RequestHandledEvent etc
.

Q #32) what does the Joint Point denote?

Answer: It denotes a specific point where the AOP aspect can be plugged in. It is the original location in the application where some action may be taken using the AOP framework.

Q #33) what is JDBC template and how to use it in Spring?

Answer: JDBC template is a template provided by Spring framework to use JDBC more efficiently.
JDBC template is generally used for conversion of database data into objects, execution of prepared and callable statements, and also supports in error handling for a database.

Q #34) what is Transaction Management in Spring? Explain the different types of Transaction Management.

Answer: Transaction is basically some operation performed on some data in the database. Transaction Management comes under Relational Database management system and is used to ensure data ethics and consistency.
The core advantage of Transaction Management is that it supports declarative and programmatic Transaction Management and API like Hibernate, JTA, and JDBC by correct integration.

There are two types of Transaction Management, which are mentioned below:

1. Programmatic Transaction Management is used to help the transaction in terms of coding or scripting.

2. Declarative Transaction Management is used to isolate business code and transactions.

Q #35) which is the most commonly used Transaction Management?

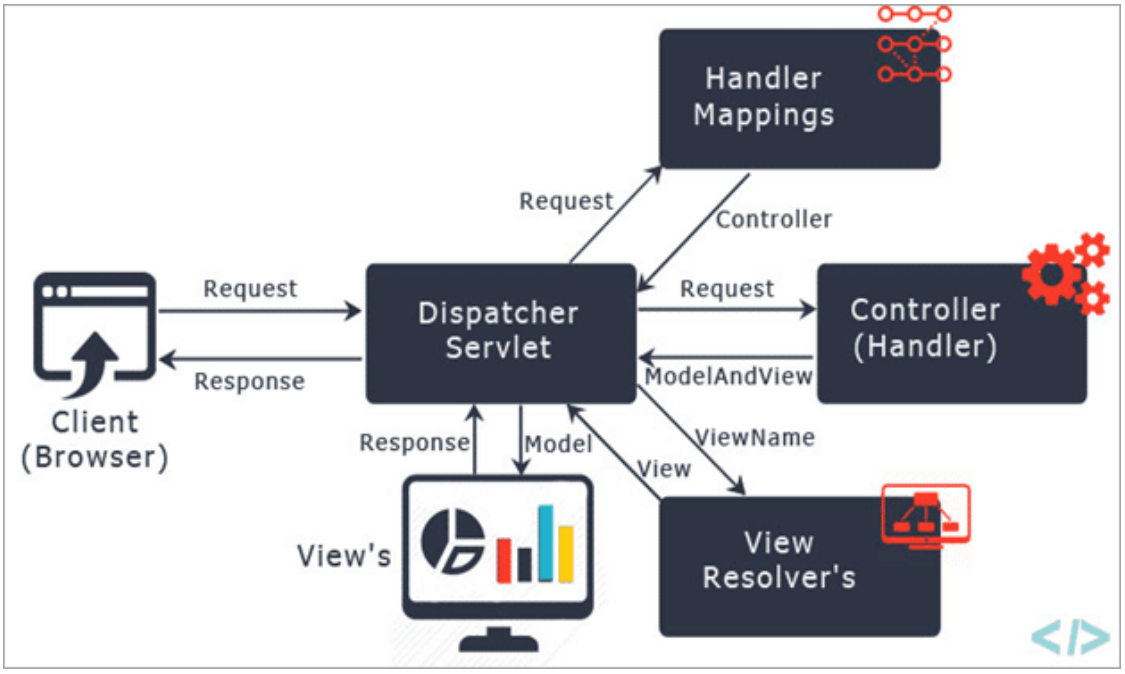
Answer: Declarative Transaction Management is widely used by developers.

Q #36) Explain the Spring MVC framework.

Answer: MVC stands for Model, View, and Controller. Spring MVC framework is used to develop web applications with good architecture flow and configurations. It is highly flexible in integration with the other frameworks.

Q #37) Explain Spring MVC Architecture.

Answer: Spring MVC architecture is based on Model, View, and Controller.



The flow of the Spring Architecture goes in the following way:

1. The request is received by the dispatcher servlet.
2. Dispatcher servlet sends the request to the handler mapping which provides the response in terms of controller class name.
3. Now the request is sent to the Controller from the dispatcher servlet, hence the controller processes the request and returns the model view object as a response to the dispatcher servlet.
4. Again, the dispatcher servlet sends the request to view resolver to get the correct view page.

5. Lastly, the dispatcher servlet sends the model object received to the browser page to display the result.

The @Controller and @RequestMapping are the two main annotations that are used in Spring MVC flow.

Q #38) what is the use of Dispatcher Servlet?

Answer: Dispatcher Servlet is used to handle all the incoming HTTP requests and responses from the client. Overall, it controls all the communications from the handler to the controller to view the resolver to the actual view page.

Q #39) explain the use of @Controller and @RequestMapping annotations in Spring MVC.

Answer:
@Controller – It denotes that particular class which behaves as a controller.
@RequestMapping – It is basically used for mapping an URL to the whole class or to any particular method.

Q #40) explain Inversion of Control and Dependency Injection through a simple example.

Answer: As we know, it is used to remove the dependency from an application.

Code without DI:

```
public class Student

{

    Address address;

   Employee()

            {

            address=new Address();// here we are creating instance
            }
}
```

Student and Address will be using the same instance, hence there will be a dependency created here.

Code with DI:

```
public class Student{
```

```
    Address address;

   Employee(Address address)

            {

            this.address=address;//not creating instance

            }

}
```

No instance is created here, so dependency is not created hence the code becomes more flexible and loosely coupled.

Q #41) explain Advice and its types.

Answer: Any action taken by AOP is called an Advice.
There are five types of Advice as mentioned below:

- Before advice.
- After advice.
- After returning advice.
- Around advice.
- Throws advice.

Q #42) what are the different types of Object Relational Mapping that Spring supports?

Answer: Spring supports ORM like Hibernate, IBatis, TopLink, Java Data Object, OJB, and JPA etc.

Q #43) why is Spring preferred over frameworks or core benefits of Spring Framework?

Answer: There are several reasons for Spring to be preferred and few of which are mentioned below:

1. Spring solves many complex problems of application development by its sub-modules like core, web, data access, test etc.
2. Spring provides POJO classes for development which in turn reduces stress.
3. Powerful integration with the other frameworks.
4. Good Application Testing.
5. Transaction Management feature and modularity.

Q #44) what do you mean by Spring Batch?

Answer: Spring batch is also a lightweight framework that is created to support the development of robust batch applications for the enterprise system.
Spring batch improves the productivity, approach for development, and makes certain features that the users have learned in Spring to be much easier.

Q #45) what is the main use of Spring batch framework?

Answer: Spring batch is mainly used to read & write files, and also in performing certain operations in the database like reading or writing, data transformation, report creation, also import and export data etc.

Q #46) explain Spring batch Architecture.

Answer: Spring batch architecture mainly comprises three component layers i.e. an application, core, and infrastructure.
An application consists of script and batch jobs created by the developers through the spring batch. The core contains all the important and necessary classes that are required for launching and controlling a batch job. Lastly, infrastructure handles both application and core including services, readers, writers etc.

Q #47) explain Tasklet in Spring Batch.

Answer: Tasklet is an interface, which is often called to perform one task only like clean up, or deletion or setting up of resources during execution time.

Q #48) explain the working principle of Spring Batch.

Answer: Spring batch mainly works on four steps as mentioned below:

1. Firstly, step-it guides the job to do its respective work.
2. It has an itemReader interface to provide the data.
3. It has the itemProcessor interface for a transformation of data.
4. Lastly, it has an itemStreamWriter interface to generate the desired result.