**Core Java Interview Question**

**1. What is Abstraction?**

Hiding internal implementation and sharing set of services is called as abstarction.

1. We can achieve abstraction by using “Interface” and “Abstract Class”.

2. We can achieve 100% abstraction using “Interface” and partial by using “Abstract Class”.

3. We can achieve security using abstraction.

4.E.g. ATM machine, Car, Mobile, etc.

**2.What is Encapsulation?**

1. Wrapping of data member and methods called as Encapsulation.

2.We can achieve it by making data members “private”.

3.POJO class is good example of encapsulation.

4. In a class if it has every data member as a “private” then such class is called as tightly encapsulated.

5. E.g. Engine, Gear box within Car, etc.

**3. Difference between Abstraction and Encapsulation?**

**Abstraction Encapsulation**

1.Hiding internal implementation and sharing set. Wrapping of data member and methods.

2.We can achieve abstraction by using “Interface” We can achieve it by making data members and “Abstract Class”. “private”.

3.It increases code. It decreases code.

4. It solves problem at Design level. It solves problem at implementation level.

**4.What is Inheritance?**

1.Acquiring properties of Parent class is called inheritance.

2. It is also called as IS-A relationship.

3. It can be achieved by using “extends” keyword, by making Parent-Child relationship.

4. It helps in reusability of code.

5. E.g. New Car version inherits properties from old versions

**5. What is Polymorphism?**

1. It means one name many forms.

2.It makes reusability simple and also makes code understanding easy.

3.There are two types of Polymorphism : o Run-time Polymorphism (Dynamic Binding, Overriding)

Decision making at Runtime by using runtime object.

Used for adding additional functionality into existing one.

Useful only in Parent-Child Relationship. o Compile-time Polymorphism (Static Binding, Overloading)

4. Best example of polymorphism is “println” method of “printstream” class.

**6. What is Static?**

1.Static is keyword.

2.We can apply static keyword with variables, methods, blocks and classes.

3. The static variable gets memory only once in the class area at the time of class loading. It can be used to refer to the common properties of all objects, for example, the company name of employees, college name of students, etc.

4.A static method belongs to the class rather than the object of a class. It can be invoked without the need for creating an instance of a class. It can access static data member and can change the value of it. It can be used to setup database connection.

5.Static block is used to initialize the static data member. It is executed before the main method at the time of class loading.

**7.What is non-static block?**

1. It is used for non-initializing content
2. Before calling constructor non-static block is executed

**8.What are methods in Exception class?**

1.public String getMessage() Returns a detailed message about the exception that has occurred.

2. public Throwable getCause() Returns the cause of the exception.

3.public String toString() Returns the name of the class concatenated with the result of getMessage().

4. public void printStackTrace() Prints the result of toString() along with the stack trace, the error output stream.

5.public StackTraceElement [] getStackTrace() Returns an array containing each element on the stack trace.

6. public Throwable fillInStackTrace() Fills the stack trace of this Throwable object with the current stack trace, adding to any previous information in the stack trace

**9. What is final, finally, finalized?**

1.The final keyword in java is used to restrict the user. The java final keyword can be used in many context. Final can be used for variables, methods, class. o Final variable once assigned can’t be changed after. o Final method can’t be rewritten, can’t be inherited. o Final class can’t be accessed by creating child of it.

2. The finally is a block that always be executed either there is exception occur inside “try” block or there is no exception occur inside “try” block. In both situation, “finally” block code will be executed.

3. The finalize() is called by the garbage collector on an object when garbage collection determines that there are no more references to the object, it is used to perform cleanup activity.

**10.What is throws keyword?**

1. By using “throws” keyword we can give a chance to caller method to handle the exception.

2. The “throws” keyword is used for propagating the exception.

3. Whenever unchecked exception will occur, it will automatically propagated.

4.Whenever checked exception will occur, we need to write “throws” keyword for propagating the exception.

5.e.g. public void m1( ) throws IO Exception, SQL Exception { }

**11. Different Scenarios in Exception Handling?**

1.Once problem will occur inside “try” block, then remaining line of code of “try” block will not be executed. ¬ If there is no problem inside “try” block, then “catch” block will be skipped. In “catch” block we can write same class of exception and its parent classes only. ¬ We can write two or more “catch” for one “try” block, but we can write child classes of exception in first catch blocks and then parent class exception, we cannot write first parent exception before child exception.

2. If “try” with “finally” block and “try” have return statement, then before control return back to caller first of all “finally” block code will be executed. ¬ If “try” with “finally” block, “try” have return statement and if “finally” block change return statement value, then it will be changed only for “finally” block, it will not be changed for caller. ¬ If “try” and “finally” both have return statement, then “finally” block statement will be executed.

3. “throws” with method calling, in case of unchecked exception classes : ¬ For unchecked exceptions, it is not needed to write explicitly throws exception, the exception propagation is done implicitly to the caller. ¬ In case of checked exception classes, needed to write explicitly throws exception for propagating exception to the caller.

4. “throws” keyword with method Overriding: ¬ (In case of unchecked exceptions) If Parent class method throws unchecked exception or doesn’t throws any exception then at the time of overriding no need to write throws keyword, if we want we can write ay of unchecked exception class but not checked exception class. ¬ (In case of checked exceptions) If Parent class method throws checked exception then at the time of overriding no need to write throws keyword, if we want then we can write same exception class or their child class but not parent class.

5. “throws” keyword with Super Class constructor and Sub Class constructor: ¬ (For unchecked exception) If Parent class constructor doesn’t throws any exception or “throws” unchecked exception then child class constructor no need to write “throws keyword, if we want then any of exception class they can write. ¬ (For checked exception) If parent class constructor throws exception then at a time of inheritance child class constructor must write “throws” keyword with same exception class or their parent class

**12.How do you handled Exception in your project?**

1. We created separate package for Exception classes needed in project where handling code was written, then at a time of exception handling scenario in project used throws keyword

**13.What is a class?**

Class is the collection of object. Class is not a real world entity. It is just a template or blueprint or prototype .Class doesn’t occupy memory.

Classes are fundamental or basic unit in Object Oriented Programming.

A class is kind of blueprint or template for objects.

Class defines variables, methods.

A class tells what type of objects we are creating.

For example. Take department class tells us we can create department type objects. We can create any number of department objects.

**14. What is an object ?**

Object is an instance of class.

Object is real world entity.

Object occupies memory.

Every object contains state and behavior.

State is determined by value of attributes and behavior is called method.

Objects are also called as an instance.

To instantiate the class we declare with the class type.

**15. What is method in java ?**

It contains the executable body that can be applied to the specific object of the class. Method includes method name, parameters or arguments and return type and a body of executable code.

Syntax : type methodName(Argument List){ }

eg : public float add(int a, int b, int c) methods can have multiple arguments. Separate with commas when we have multiple arguments.

**16. what are static blocks and static initalizers in Java ?**

Static blocks or static initializers are used to initalize static fields in java. we declare static blocks when we want to intialize static fields in our class. Static blocks gets executed exactly once when the class is loaded . Static blocks are executed even before the constructors are executed.

**17. How to call one constructor from the other constructor ?**

With in the same class if we want to call one constructor from other we use this() method.

Based on the number of parameters we pass appropriate this() method is called. Restrictions for using this method :

1) this must be the first statement in the constructor .

2)we cannot use two this() methods in the constructor.

**18. What is super keyword in java ?**

Variables and methods of super class can be overridden in subclass . In case of overriding , a subclass object call its own variables and methods. Subclass cannot access the variables and methods of superclass because the overridden variables or methods hides the methods and variables of super class. But still java provides a way to access super class members even if its members are overridden. Super is used to access superclass variables, methods, constructors.

Super can be used in two forms :

1) First form is for calling super class constructor.

2) Second one is to call super class variables, methods.

Super if present must be the first statement.

**19. Difference between this() and super() in java ?**

this() is used to access one constructor from another with in the same class while super() is used to access superclass constructor.

Either this() or super() exists it must be the first statement in the constructor.

**20. What is constructor in java ?**

A constructor is a special method used to initialize objects in java. we use constructors to initialize all variables in the class when an object is created. As and when an object is created it is initialized automatically with the help of constructor in java. We have two types of constructors Default Constructor Parameterized Constructor

Signature : public classname() { }

Signature : public classname(parameters list) {}

**21. What is ‘IS-A ‘ relationship in java?**

‘is a’ relationship is also known as inheritance. We can implement ‘is a’ relationship or inheritance in java using extends keyword. The advantage or inheritance or is a relationship is reusability of code instead of duplicating the code. Ex : Motor cycle is a vehicle Car is a vehicle Both car and motorcycle extends vehicle

**22. What is ‘HAS A’’ relationship in java?**

‘Has a ‘ relationship is also known as “composition or Aggregation”. As in inheritance we have ‘extends’ keyword we don’t have any keyword to implement ‘Has a’ relationship in java. The main advantage of ‘Has-A‘ relationship in java code reusability

**23. Explain about abstract classes in java?**

Sometimes we may come across a situation where we cannot provide implementation to all the methods in a class. We want to leave the implementation to a class that extends it. In such case we declare a class as abstract. To make a class abstract we use key word abstract. Any class that contains one or more abstract methods is declared as abstract. If we don’t declare class as abstract which contains abstract methods we get compile time error. We get the following error. “The type must be an abstract class to define abstract methods.” Signature ; abstract class { } For example if we take a vehicle class we cannot provide implementation to it because there may be two wheelers , four wheelers etc. At that moment we make vehicle class abstract. All the common features of vehicles are declared as abstract methods in vehicle class. Any class which extends vehicle will provide its method implementation. It’s the responsibility of subclass to provide implementation. The important features of abstract classes are : 1) Abstract classes cannot be instantiated. 2) An abstract classes contains abstract methods, concrete methods or both. 3) Any class which extends abstract class must override all methods of abstract class. 4) An abstract class can contain either 0 or more abstract methods. 18 Though we cannot instantiate abstract classes we can create object references . Through superclass references we can point to subclass

**24. What is Exception handling in java?**

Exception handling is a mechanism what to do when some abnormal situation arises in program. When an exception is raised in program it leads to termination of program when it is not handled properly. The significance of exception handling comes here in order not to terminate a program abruptly and to continue with the rest of program normally. This can be done with help of Exception handling

**25. What is an error in Java?**

Error is the subclass of Throwable class in java. When errors are caused by our program we call that as Exception, but some times exceptions are caused due to some environment issues such as running out of memory. In such cases we can’t handle the exceptions. Exceptions which cannot be recovered are called as errors in java. Ex : Out of memory issues

**26. In how many ways we can do exception handling in java?**

We can handle exceptions in either of the two ways :

1) By specifying try catch block where we can catch the exception.

2) Declaring a method with throws clause .

List five keywords related to Exception handling

1) Try

2) Catch

3) throw

4) throws

5) final

**27. Explain importance of finally block in java?**

Finally block is used for cleaning up of resources such as closing connections, sockets etc. if try block executes with no exceptions then finally is called after try block without executing catch block. If there is exception thrown in try block finally block executes immediately after catch block. If an exception is thrown, finally block will be executed even if the no catch block handles the exception.

**28. What are checked Exceptions?**

All the subclasses of Throwable class except error, Runtime Exception and its subclasses are checked exceptions

. 2) Checked exception should be thrown with keyword throws or should be provided try catch block, else the program would not compile. We do get compilation error.

Examples : 1) IOException,

2) SQlException,

3) FileNotFoundException,

4) InvocationTargetException,

5) CloneNotSupportedException

6) ClassNotFoundException

7) InstantiationException

**29. What are unchecked exceptions in java?**

All subclasses of RuntimeException are called unchecked exceptions. These are unchecked exceptions because compiler does not check if a method handles or throws exceptions.

Program compiles even if we do not catch the exception or throws the exception.

If an exception occurs in the program, program terminates.

It is difficult to handle these exceptions because there may be many places causing exceptions. Example: 1) Arithmetic Exception

3) ArrayIndexOutOfBoundsException

4) ClassCastException

5) IndexOutOfBoundException

6) NullPointerException

7) NumberFormatException

8) StringIndexOutOfBounds

9) UnsupportedOperationException

**30. What is default Exception handling in java?**

When JVM detects exception causing code, it constructs a new exception handling object by including the following information.

1) Name of Exception

2) Description about the Exception

3) Location of Exception.

After creation of object by JVM it checks whether there is exception handling code or not. If there is exception handling code then exception handles and continues the program. If there is no exception handling code JVM give the responsibility of exception handling to default handler and terminates abruptly. Default Exception handler displays description of exception, prints the stack trace and location of exception and terminates the program.

Note : The main disadvantage of this default exception handling is program terminates abruptly.

**Collection**

**1.What is Collection?**

1. The Collection is a framework that provides an architecture to store and manipulate the group of objects.

2. It provides operations that you perform on a data such as searching, sorting, insertion and deletion on the group of objects.

3. Collection represents a single unit of objects as a group.

**2. Why do we use Collection?**

(Explain Difference between Array & Collection)

Array Collection

1. It is Fixed in Size. It is Growable in nature.

2. It can hold only Homogeneous Data Elements. It can hold both Homogeneous and Heterogeneous

With Respect to Memory Arrays are With Respect to Memory Collections are Not Recommended to Use Recommended to Use.

**3. What is Difference between Collection & Collections?**

Collection Collections

It is an interface. It is class.

It can be used to Represent a Group of It is used to sort and synchronize the

Individual Objects as a Single Entity.. collection elements

It provides the methods that can be used It provides the methods which can be used for various

for data structure operations on a collection

**4.How Array list works?**

1. When we create object of Arraylist, it create Arraylist instance with default capacity 10.

2.Arraylist capacity increases with formula – New Capacity = ((3/2) x Old Capacity)+1

3. When Arraylist increments with new capacity then data from old Arraylist is copied into new instance and old instance is destroyed.

4.When we add or delete data into the Arraylist then multiple data shift operations are performed.  Arraylist follows Indexing.

**5.Explain all Constructors of Arraylist?**

1.ArrayList l = new ArrayList(); It creates an Empty ArrayList Object with Default Initial Capacity 10.

2. ArrayList l = new ArrayList(intinitialCapacity); It creates an Empty ArrayList Object with specified Initial Capacity.

3. ArrayList l = new ArrayList(Collection c); It creates an equivalent ArrayList Object for the given Collection Object

**6.How Linkedlist works? (Why insertion & deletion is fast in Linkedlist?)**

1.When we create an object of Linkedlist and add an element to it.

2.It stores element as a node in which previous & next node address is also stored.

3. Node format = ||prev. node addr.| (value) |next node addr.||

4. Due to previous & next node address is stored, hence while updation or insertion & deletion operation data shift operation need not to perform and it makes Linkedlist fast.

**7.Define Linkedlist?**

1. It is one of implemented class of List interface in collection framework.

2.It allows duplicate values, Insertion order is preserved & indexing is maintained.

3. It implements Cloneable, Serializable interfaces.

4. It follows doubly linked list structure.

5.It is mostly preferable for insertion & deletion operation.

**8.What is Map?**

1. Map is used for store different object in the pair of “key” and “value”.

2.In map, “key” should be unique.  Insertion order will not be maintained in Map.

**9. Difference between Hashmap & Hashtable?**

HashMap Hashtable

1. It is not Synchronised. It is Synchronised.

2.It allows multiple threads at a time. It allows single thread at a time.

3.It is not thread safe. It is thread safe.

4. Null key (once) & Null value is allowed. Null key & Null value is not all

**10. How HashMap works?**

1. When we create HashMap object, HashMap instance as per default capacity 16 buckets is created.

2.When we perform add (put ( )) operation, it accepts data in key & value format.

3. Internally hashing technique is used, that generates hashcode for key and also calculate index to find bucket location for inserting data in HashMap instance.

4. It will store element at that location as a node format. ||previous node address| (Key) | (Value) |next node address||

5.Now when we perform retrieval (get ( )) operation, it asks for key.

6.Again hashing technique is used and bucket location is identified, then equals ( ) method is used to compare key content and if it returns true then value is retrieved.

**11.What is Contract between equals () & hashcode ()?**

1. If equals ( ) returns true, then objects must have same hashcodes.

2. If equals ( ) returns false, then objects may or may not have same hashcodes.

3.If hashcodes of objects are same, then we can’t conclude output of equals ( ), it may be true or may be false.

4.If hashcodes of objects are different, then output of equals ( ) must be false.

**12.Difference between Hashmap & Synchronised (or Concurrent) Hashmap?**

HashMap Synchronised or Concurrent HashMap

1.It is non-Synchronized in nature. It is Synchronized in nature.

2.It is not Thread-safe. It is thread-safe.

3.Performance is high. Performance is low.

4. It can throw ConcurrentModificationException. It doesn’t throw ConcurrentModificationException

**13.Difference between Comparable & Comparator?**

Comparable Comparator

1.This interface is from java.lang package. This interface is from java.util package.

2.It is used for Default sorting. It is used for Custom sorting.

3.It has only one method i.e. compareTo. It has two methods i.e. compare & equals

4. Programmer decides how sorting is to be done. User decides how sorting is to be done.

**14.What is Identity Hashmap?**

1. In IdentityHashMap JVM will Use == Operator to Identify Duplicate Keys, which is meant for Reference Comparison.

2. e.g. Integer i=new Integer (5);

Integer i1=new Integer (5);

Map m=new IdentityHashMap();

m.put(i, "java"); m.put(i1, "cjc");

System.out.println(m); => {5=java, 5=cjc}

**15. What is Hash Collision?**

In HashMap, if two keys have same hashcodes then such situation is called as hash collision.

1. In such case, while adding data, doubly Linkedlist is created to insert data. ||prev. node addr.| (Key1) | (Value1) |next node addr.|| ◊ || | (Key2) | (Value2) | ||

2.And retrieval operation is performed using equals ( ) method.

Spring

### 1) What is Spring?

It is a lightweight, loosely coupled and integrated framework for developing enterprise applications in java.

### 2) What are the advantages of spring framework?

1. Predefined Templates
2. Loose Coupling
3. Easy to test
4. Lightweight
5. Fast Development
6. Powerful Abstraction
7. Declarative support

### 3) What are the modules of spring framework?

1. Test
2. Spring Core Container
3. AOP, Aspects and Instrumentation
4. Data Access/Integration
5. Web

### 4) What is IOC and DI?

IOC (Inversion of Control) and DI (Dependency Injection) is a design pattern to provide loose coupling. It removes the dependency from the program.

### 5) What is the role of IOC container in spring?

IOC container is responsible to:

* create the instance
* configure the instance, and
* assemble the dependencies.

**6) What are the types of IOC container in spring?**

There are two types of IOC containers in spring framework.

1. Bean Factory
2. Application Context.

### 7) What is the difference between Bean Factory and Application Context?

|  |  |
| --- | --- |
| **BeanFactory** | **ApplicationContext** |
| It is an interface defined in org. springframework.beans.factory.**BeanFactory** | It is an interface defined in org. springframework.context.**ApplicationContext** |
| It uses Lazy initialization | It uses Eager/ Aggressive initialization |
| It explicitly provides a resource object using the syntax | It creates and manages resource objects on its own |
| It doesn’t supports internationalization | It supports internationalization |
| It doesn’t supports annotation based dependency | It supports annotation based dependency |

### 8) What is the difference between constructor injection and setter injection?

|  |  |  |
| --- | --- | --- |
| **No.** | **Constructor Injection** | **Setter Injection** |
| 1) | No Partial Injection | Partial Injection |
| 2) | Doesn't override the setter property | Overrides the constructor property if both are defined. |
| 3) | Creates new instance if any modification occurs | Doesn't create new instance if you change the property value |
| 4) | Better for too many properties | Better for few properties. |
|  |  |  |

### 9) What is autowiring in spring? What are the autowiring modes?

Autowiring enables the programmer to inject the bean automatically. We don't need to write explicit injection logic.

The autowiring modes are given below:

|  |  |  |
| --- | --- | --- |
| **No.** | **Mode** | **Description** |
| 1) | no | this is the default mode, it means autowiring is not enabled. |
| 2) | byName | injects the bean based on the property name. It uses setter method. |
| 3) | byType | injects the bean based on the property type. It uses setter method. |
| 4) | constructor | It injects the bean using constructor |

### 10) What are the different bean scopes in spring?

There are 5 bean scopes in spring framework.

|  |  |  |
| --- | --- | --- |
| **No.** | **Scope** | **Description** |
| 1) | singleton | The bean instance will be only once and same instance will be returned by the IOC container. It is the default scope. |
| 2) | prototype | The bean instance will be created each time when requested. |
| 3) | request | The bean instance will be created per HTTP request. |
| 4) | session | The bean instance will be created per HTTP session. |
| 5) | Global session | The bean instance will be created per HTTP global session. It can be used in portlet context only. |

### 11) In which scenario, you will use singleton and prototype scope?

Singleton scope should be used with EJB **stateless session bean** and prototype scope with EJB **stateful session bean**.

**12) What are the transaction management supports provided by spring?**

Spring framework provides two type of transaction management supports:

1. **Programmatic Transaction Management**: should be used for few transaction operations.
2. **Declarative Transaction Management**: should be used for many transaction operations.

**13) What are classes for spring JDBC API?**

1. JdbcTemplate
2. SimpleJdbcTemplate
3. NamedParameterJdbcTemplate
4. SimpleJdbcInsert
5. SimpleJdbcCall.

### 14) What are the different features of Spring Framework?

Following are some of the major features of Spring Framework :

* **Lightweight:** Spring is lightweight when it comes to size and transparency.
* **Inversion of control (IOC):** The objects give their dependencies instead of creating or looking for dependent objects. This is called Inversion Of Control.
* **Aspect oriented Programming (AOP):** Aspect oriented programming in Spring supports cohesive development by separating application business logic from system services.
* **Container:**Spring Framework creates and manages the life cycle and configuration of the application objects.
* **MVC Framework:** Spring Framework’s MVC web application framework is highly configurable. Other frameworks can also be used easily instead of Spring MVC Framework.
* **Transaction Management:** Generic abstraction layer for transaction management is provided by the Spring Framework. Spring’s transaction support can be also used in container less environments.
* **JDBC Exception Handling:** The JDBC abstraction layer of the Spring offers an exception hierarchy, which simplifies the error handling strategy.

**15. What are the various ways of using Spring Framework?**

Spring Framework can be used in various ways. They are listed as follows:

1. As a Full-fledged Spring web application.
2. As a third-party web framework, using Spring Frameworks middle-tier.
3. For remote usage.
4. As Enterprise Java Bean which can wrap existing POJOs (Plain Old Java Objects).

### ****16. What’s the difference between @Component, @Controller, @Repository & @Service annotations in Spring?****

**@Component:** This marks a java class as a bean. It is a generic stereotype for any Spring-managed component. The component-scanning mechanism of spring now can pick it up and pull it into the application context.

**@Controller:** This marks a class as a Spring Web MVC controller. Beans marked with it are automatically imported into the Dependency Injection container.

**@Service:** This annotation is a specialization of the component annotation. It doesn’t provide any additional behavior over the @Component annotation. You can use @Service over @Component in service-layer classes as it specifies intent in a better way.

**@Repository:** This annotation is a specialization of the @Component annotation with similar use and functionality. It provides additional benefits specifically for DAOs. It imports the DAOs into the DI container andmakes the unchecked exceptions eligible for translation into Spring DataAccessException.

### ****17. What do you understand by @Required annotation?****

@Required is applied to bean property setter methods. This annotation simply indicates that the affected bean property must be populated at the configuration time with the help of an explicit property value in a bean definition or with autowiring. If the affected bean property has not been populated, the container will throw BeanInitializationException.

### ****18. What do you understand by @Autowired annotation?****

The **@Autowired** annotation provides more accurate control over where and how autowiring should be done. This annotation is used to autowire bean on the setter methods, constructor, a property or methods with arbitrary names or multiple arguments. By default, it is a type driven injection.

### ****19. What do you understand by @Qualifier annotation?****

When you create more than one bean of the same type and want to wire only one of them with a property you can use the **@Qualifier** annotation along with **@Autowired** to remove the ambiguity by specifying which exact bean should be wired.

**20.  What do you understand by @RequestMapping annotation?**

@RequestMapping annotation is used for mapping a particular HTTP request method to a specific class/ method in controller that will be handling the respective request. This annotation can be applied at both levels:

* **Class level**: Maps the URL of the request
* **Method level**: Maps the URL as well as HTTP request method.

### 21.****What are the difference between Spring AOP and AspectJ AOP?****

|  |  |
| --- | --- |
| **Spring AOP** | **AspectJ AOP** |
| Runtime weaving through proxy is done | Compile time weaving through AspectJ Java tools is done |
| It supports only method level PointCut | It suports field level Pointcuts |
| It is DTD based | It is schema based and Annotation configuration |

### 22****. What do you mean by Proxy in Spring Framework?****

An object which is created after applying advice to a target object is known as a Proxy. In case of client objects the target object and the proxy object are the same.

### ****23. In Spring, what is Weaving?****

The process of linking an aspect with other application types or objects to create an advised object is called Weaving. In Spring AOP, weaving is performed at runtime.

**24. Name Some of the Design Patterns Used in the Spring Framework?**

* **Singleton Pattern** – singleton-scoped beans
* **Factory Pattern** – Bean Factory classes
* **Prototype Pattern** – prototype-scoped beans
* **Adapter Pattern** – Spring Web and Spring MVC
* **Proxy Pattern** – Spring Aspect-Oriented Programming support
* **Template Method Pattern** – *JdbcTemplate*, *HibernateTemplate*, etc.
* **Front Controller** – Spring MVC *DispatcherServlet*
* **Data Access Object** – Spring DAO support
* **Model View Controller**– Spring MVC

**25. What Is Spring JdbcTemplate Class and How to Use It?**

The Spring JDBC template is the primary API through which we can access database operations logic that we’re interested in:

* Creation and closing of connections
* Executing statements and stored procedure calls
* Iterating over the *ResultSet* and returning results

### ****26. What Is Spring WebFlux?****

[Spring WebFlux](https://docs.spring.io/spring/docs/current/spring-framework-reference/web-reactive.html#webflux) is Spring's reactive-stack web framework, and it's an alternative to Spring MVC. In order to achieve this reactive model and be highly scalable, the entire stack is non-blocking.

### ****27. What Is the Default Bean Scope in Spring Framework?****

By default, a Spring Bean is initialized as a singleton.

**28. Which Is the Best Way of Injecting Beans and Why?**

The recommended approach is to use constructor arguments for mandatory dependencies and setters for optional ones. This is because constructor injection allows injecting values to immutable fields and makes testing easier.

**29. What is Spring configuration file?**

Spring configuration file is an XML file. This file contains the classes information and describes how these classes are configured and introduced to each other

**30.What is Spring AOP?**

Aspect-oriented programming, or AOP, is a programming technique that allows programmers to modularize crosscutting concerns, or behavior that cuts across the typical divisions of responsibility, such as logging and transaction management. The core construct of AOP is the aspect, which encapsulates behaviors affecting multiple classes into reusable modules.

**Java8**

**1. List down the new features introduced in Java 8?**

Answer: New features that are introduced in Java 8 are enlisted below:

* Lambda Expressions
* Method References
* Optional Class
* Functional Interface
* Default methods
* Stream API

**2.What are Functional Interfaces?**

Answer: Functional Interface is an interface that has only one abstract method. The implementation of these interfaces is provided using a Lambda Expression which means that to use the Lambda Expression, you need to create a new functional interface or you can use the predefined [functional interface of Java 8](https://www.softwaretestinghelp.com/java-8-interface-changes/).

The annotation used for creating a new Functional Interface is “@FunctionalInterface”.

**3. What is an optional class?**

Answer: Optional class is a special wrapper class introduced in Java 8 which is used to avoid NullPointerExceptions. This final class is present under java.util package. NullPointerExceptions occurs when we fail to perform the Null checks.

**4. What are the default methods?**

Answer: Default methods are the methods of the Interface which has a body. These methods, as the name suggests, use the default keywords. The use of these default methods is “Backward Compatibility” which means if JDK modifies any Interface (without default method) then the classes which implement this Interface will break.

On the other hand, if you add the default method in an Interface then you will be able to provide the default implementation. This won’t affect the implementing classes.

Syntax:

|  |
| --- |
| public interface questions{            default void print() {    System.out.println("www.softwaretestinghelp.com");               }      } |

**5.What are the main characteristics of the Lambda Function?**

Answer: Main characteristics of the Lambda Function are as follows:

* A method that is defined as Lambda Expression can be passed as a parameter to another method.
* A method can exist standalone without belonging to a class.
* There is no need to declare the parameter type because the compiler can fetch the type from the parameter’s value.
* We can use parentheses when using multiple parameters but there is no need to have parenthesis when we use a single parameter.
* If the body of expression has a single statement then there is no need to include curly braces.

**6. How can you create a Functional Interface?**

Answer: Although Java can identify a Functional Interface, you can define one with the annotation

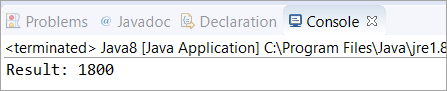
@FunctionalInterface

Once you have defined the functional interface, you can have only one abstract method. Since you have only one abstract method, you can write multiple static methods and default methods.

Below is the programming example of FunctionalInterface written for multiplication of two numbers.

|  |
| --- |
| @FunctionalInterface // annotation for functional interface  interface FuncInterface {        public int multiply(int a, int b);  }  public class Java8 {       public static void main(String args[]) {          FuncInterface Total = (a, b) -&gt; a \* b;          // simple operation of multiplication of 'a' and 'b'          System.out.println("Result: "+Total.multiply(30, 60));      }  } |

Output:

[](https://www.softwaretestinghelp.com/wp-content/qa/uploads/2020/04/8th-Question.png)

**7. What is Method Reference?**

Answer: In Java 8, a new feature was introduced known as Method Reference. This is used to refer to the method of functional interface. It can be used to replace Lambda Expression while referring to a method.

For Example: If the Lambda Expression looks like

num -> System.out.println(num)

Then the corresponding Method Reference would be,

System.out::println

where “::” is an operator that distinguishes class name from the method name.

**8. What is a Predicate? State the difference between a Predicate and a Function?**

Answer: Predicate is a pre-defined Functional Interface. It is under java.util.function.Predicate package. It accepts only a single argument which is in the form as shown below,

Predicate<T>

| Predicate | Function |
| --- | --- |
| It has the return type as Boolean. | It has the return type as Object. |
| It is written in the form of Predicate< T> which accepts a single argument. | It is written in the form of Function< T, R> which also accepts a single argument. |
| It is a Functional Interface which is used to evaluate Lambda Expressions. This can be used as a target for a Method Reference. | It is also a Functional Interface which is used to evaluate Lambda Expressions. In Function< T, R>, T is for input type and R is for the result type. This can also be used as a target for a Lambda Expression and Method Reference. |

**9. What is a Stream API? Why do we require the Stream API?**

Answer: Stream API is a new feature added in Java 8. It is a special class that is used for processing objects from a source such as Collection.

We require the Stream API because,

* It supports aggregate operations which makes the processing simple.
* It supports Functional-Style programming.
* It does faster processing. Hence, it is apt for better performance.
* It allows parallel operations.

**10.What are static methods in Interfaces?**

Static methods, which contains method implementation is owned by the interface and is invoked using the name of the interface, it is suitable for defining the utility methods and cannot be overridden