1.What is difference in Overloading and Overriding

Ans -

* When two or more methods in the same class have the same name but different parameters, it’s called Overloading.
* When the method signature (name and parameters) are the same in the superclass and the child class, it’s called Overriding.
* Overriding implements Runtime Polymorphism whereas Overloading implements Compile time polymorphism.
* The method Overriding occurs between superclass and subclass. Overloading occurs between the methods in the same class.
* Overriding methods have the same signature i.e. same name and method arguments. Overloaded method names are the same but the parameters are different.
* With Overloading, the method to call is determined at the compile-time. With overriding, the method call is determined at the runtime based on the object type.
* If overriding breaks, it can cause serious issues in our program because the effect will be visible at runtime. Whereas if overloading breaks, the compile-time error will come and it’s easy to fix.

2.If both the functions are statis can we override

Ans -

* No, we cannot override static methods because method overriding is based on dynamic binding at runtime and the static methods are bonded using static binding at compile time.
* So, we cannot override static methods.The calling of method depends upon the type of object that calls the static method. It means:
* If we call a static method by using the parent class object, the original static method will be called from the parent class.
* If we call a static method by using the child class object, the static method of the child class will be called.
* In the following example, the ParentClass has a static method named display() and the ChildClass also has the same method signature. The method in the derived class (ChildClass) hides the method in the base class.

3.Difference between Final , Finally and Finalize

Ans- Final:

* final is the keyword and access modifier which is used to apply restrictions on a class, method or variable.
* final is the keyword and access modifier which is used to apply restrictions on a class, method or variable.
* Once declared, final variable becomes constant and cannot be modified.
* the final method cannot be overridden by subclass.
* final class cannot be inherited.
* Final method is executed only when we call it.

Finally:

* finally is the block in Java Exception Handling to execute the important code whether the exception occurs or not.
* Finally, block is always related to the try and catch block in exception handling.
* Finally block runs the important code even if an exception occurs or not.
* finally block cleans up all the resources used in try block
* Finally block is executed as soon as the try-catch block is executed.
* It's execution is not dependent on the exception.

Finalize:

* Finalize is the method in Java which is used to perform clean up processing just before an object is garbage collected.
* Finalize() method is used with the objects.
* Finalize method performs the cleaning activities with respect to the object before its destruction.
* Finalize method is executed just before the object is destroyed.

4.What are marker Interfaces

Ans-

* An interface that does not contain methods, fields, and constants is known as marker interface.
* In other words, an empty interface is known as marker interface or tag interface. It delivers the run-time type information about an object.
* It is the reason that the JVM and compiler have additional information about an object. The Serializable and Cloneable interfaces are examples of marker interfaces.
* In short, it indicates a signal or command to the JVM.

Uses of Marker Interface:

* Marker interface is used as a tag that inform the Java compiler by a message so that it can add some special behavior to the class implementing it.
* Java marker interface are useful if we have information about the class and that information never changes, in such cases, we use marker interface to represent the same.
* Implementing an empty interface tells the compiler to do some operations.
* It is used to logically divide the code and a good way to categorize code. It is more useful for developing API and in frameworks like Spring.

Built-in Marker Interface:

In Java, built-in marker interfaces are the interfaces that are already present in the JDK and ready to use. There are many built-in marker interfaces some of them are:

* Cloneable Interface
* Serializable Interface
* Remote Interface

5.What is Singleton class

Ans-

* In object-oriented programming, a singleton class is a class that can have only one object (an instance of the class) at a time.
* After the first time, if we try to instantiate the Singleton class, the new variable also points to the first instance created.
* So whatever modifications we do to any variable inside the class through any instance, affects the variable of the single instance created and is visible if we access that variable through any variable of that class type defined.

Purpose of Singleton Class:

* The primary purpose of a Single class is to restrict the limit of the number of object creation to only one.
* This often ensures that there is access control to resources, for example, socket or database connection.
* The memory space wastage does not occur with the use of the singleton class because it restricts the instance creation.
* As the object creation will take place only once instead of creating it each time a new request is made.
* We can use this single object repeatedly as per the requirements.
* This is the reason why the multi-threaded and database applications mostly make use of the Singleton pattern in Java for caching, logging, thread pooling, configuration settings, and much more.

6.What is the underlying Data structure for Arraylist and Linkedlist

Ans- Data structure of ArrayList is an array of Object class and Linkedlist is Doubly linkedlist

7.What is hash collision

Ans- A collision, or more specifically, a hash code collision in a HashMap, is a situation where two or more key objects produce the same final hash value and hence point to the same bucket location or array index.

8.What are lambda Expressions

Ans- A lambda expression is a short block of code which takes in parameters and returns a value. Lambda expressions are similar to methods, but they do not need a name and they can be implemented right in the body of a method.

9.Whats is functional interface

Ans- A functional interface is an interface that contains only one abstract method. They can have only one functionality to exhibit.

10.What are optional classes in Java 8

Ans- Optional is a container object used to contain not-null objects. Optional object is used to represent null with absent value.

11.Whats is factory design pattern and Adapter and difference between them

Ans -

* Adapter pattern works as a bridge between two incompatible interfaces. This type of design pattern comes under structural pattern as this pattern combines the capability of two independent interfaces.
* This pattern involves a single class which is responsible to join functionalities of independent or incompatible interfaces

12.What are Solid principles

Ans -

1. Single Responsibility Principle - Each class should be responsible for a single part or functionality of the system.
2. Open-Closed Principle - Software components should be open for extension, but not for modification.
3. Liskov Substitution Principle - Objects of a superclass should be replaceable with objects of its subclasses without breaking the system.
4. Interface Segregation Principle - No client should be forced to depend on methods that it does not use.
5. Dependency Inversion Principle - High-level modules should not depend on low-level modules, both should depend on abstractions.

13. What does @component and @scan do in Sprinboot annotation

Ans-

* @Component is an annotation that allows Spring to automatically detect our custom beans.
* Spring provides a mechanism to identify Spring bean candidates explicitly through the @ComponentScan annotation.

14.Whats is the Diference bew JPA , Hibernate and ORM

Ans -

* Java Persistence API (JPA) -
  + Java Persistence API (JPA) defines the management of relational data in the Java applications.It is just a specification. Various ORM tools implement it for data persistence.
* Hibernate -
  + Hibernate is an Object-Relational Mapping (ORM) tool which is used to save the state of Java object into the database.
  + It is one of the most frequently used JPA implementation.
  + It is just a specification. Various ORM tools implement it for data persistence.

15.What is indexing in Database

Ans- Indexing is a way to optimise the performance of a database by minimising the number of disk accesses required when a query is processed.

It is a data structure technique which is used to quickly locate and access the data in a database.

16.What are 4 types of Drivers in Jdbc

Ans- 1.JDBC-ODBC bridge driver

2.Native-API driver (partially java driver)

3.Network Protocol driver (fully java driver)

4.Thin driver (fully java driver)

17.What is difference between @component and @bean and @service

Ans-

* @Component is an annotation that allows Spring to automatically detect our custom beans.
* @Bean annotation which is applied on a method to specify that it returns a bean to be managed by Spring context. Spring Bean annotation is usually declared in Configuration classes methods.
* @Service annotation is used with classes that provide some business functionalities. Spring context will autodetect these classes when annotation-based configuration and classpath scanning is used.

18.What is difference between SOAP and restful services technically

Ans-

* SOAP is a protocol whereas REST is an architectural pattern.
* SOAP uses service interfaces to expose its functionality to client applications while REST uses Uniform Service locators to access the components on the hardware device.
* SOAP needs more bandwidth for its usage whereas REST doesn’t need much bandwidth.
* Comparing SOAP vs REST API, SOAP only works with XML formats whereas REST works with plain text, XML, HTML and JSON.
* SOAP cannot make use of REST whereas REST can make use of SOAP.

19.Difference in POST and PUT

Ans-

* PUT method is called when you have to modify a single resource, which is already a part of resource collection. We use UPDATE query in PUT.
* POST method is called when you have to add a child resource under resources collection. We use create query in POST.

20.Is react unidirectional or Bidirectional

Ans-

* React doesn't support bi-directional binding to make sure you are following a clean data flow architecture.
* The major benefit of this approach is that data flows throughout your app in a single direction, giving you better control over it.