**1. What is difference in Overloading and Overriding:**

Method Overloading is a Compile time polymorphism: More than one method shares the same method name with a different signature in the class.The return type may or may not be the same, but we have to change the parameter because, in java, we can not achieve the method overloading by changing only the return type of the method.  
Method Overriding is a Run time polymorphism: The derived class provides the specific implementation of the method that is already provided by the base class or parent class-They have the same method name and signatures. In method overriding, the return type must be the same or co-variant.

**2. If both the functions are static can we over ride**

It is not possible to override static methods because there is nothing to override, as they would be two different methods. Overriding is based on dynamic binding at runtime and the static methods are bonded using static binding at compile time. This means static methods are resolved before objects are even created, Hence it's not possible to override static methods in Java. However you can declare a method with the same name and method signature in the subclass which does look like you can override static methods (Thi s is known as method hiding).

**3. Difference between Final, Finally and Finalize**

final is the keyword and access modifier which is used to apply restrictions on a class, method or variable.  
finally is the block in Java Exception Handling to execute the important code whether the exception occurs or not.  
finalize is the method in Java which is used to perform clean up processing just before object is garbage collected.

**4. What are maker Interfaces?**

An interface that does not contain methods, fields, or any constants is known as marker interface -An empty interface is known as marker interface or tag interface. It delivers the run-time type information about an object. It indicates a signal or command to the JVM. It is the reason that the JVM and compiler have additional information about an object. It indicates a signal or command to the JVM.   
eg: Serializable, Cloneable interfaces, Remote interface.

**5. What is Singleton class?**

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.

**6. What is the underlying Data structure for Arraylist and Linked list?**

Both ArrayList and LinkedList are two different implementations of the List interface. ArrayList is a resizable-array implementation of an Array, whereas LinkedList is a Doubly-linked list implementation of the List interface. This will lead to further differences in performance.

**7. What is hash coalition?**

A collision/ 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 location/index.The hash value in this case is derived from a hash function which takes a data input and returns a fixed length of bits. A mapping from a large value space to a much smaller value space will most probably lead to collisions.

**8. What are lambda Expressions?**

A lambda expression were added in JAVA 8. It 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. Hence we don't need to define the method again for providing the implementation. Here, we just write the implementation code. It provides a clear and concise way to represent one method interface using an expression. It is very useful in collection library. It helps to iterate, filter and extract data from collection. The Lambda expression is used to provide the implementation of an interface which has functional interface. It saves a lot of code. Java lambda expression is treated as a function, so compiler does not create .class file.

**9.Whats is functional interface**  
An Interface that contains exactly one abstract method is known as functional interface. It can have any number of default, static methods but can contain only one abstract method. It can also declare methods of object class.  
Functional Interface is also known as Single Abstract Method Interfaces or SAM Interfaces. It is a new feature in Java, which helps to achieve functional programming approach.  
  
**10.What are optional classes in Java 8**  
Optional is a container object used to contain not-null objects. Optional object is used to represent null with absent value. It is a public final class and has various utility methods to facilitate code to handle values as ‘available’ or ‘not available’ instead of checking null values. It is used to deal with NullPointerException in Java application. You must import java.util package to use this class.   
Declaration of Optional class: public final class Optional<T> extends Object

**11. What is factory design pattern and Adapter and difference between them?**

Adapter pattern Convert the interface of a class into another interface clients expect. Adapter lets classes work together that couldn't otherwise because of incompatible interfaces.

**12. What are Solid principles?**

SOLID stands for:

S **- Single-responsiblity Principle**

A class should have one and only one reason to change, meaning that a class should have only one job.

O **- Open-closed Principle**

Objects or entities should be open for extension but closed for modification.

L **- Liskov Substitution Principle**

Let q(x) be a property provable about objects of x of type T. Then q(y) should be provable for objects y of type S where S is a subtype of T.

I **- Interface Segregation Principle**

A client should never be forced to implement an interface that it doesn’t use, or clients shouldn’t be forced to depend on methods they do not use.

D **- Dependency Inversion Principle**

Entities must depend on abstractions, not on concretions. It states that the high-level module must not depend on the low-level module, but they should depend on abstractions

**13. What does @component and @scan do in Spring boot annotation?**

@Component is an annotation that allows Spring to automatically detect our custom beans.

@ComponentScan without arguments tells Spring to scan the current package and all of its sub-packages. With the specified package @ComponentScan will scan the particular package for beans.

**14. What's is the Difference between JPA , Hibernate and ORM?**

|  |  |  |
| --- | --- | --- |
| **JPA** | **Hibernate** | **ORM** |
| Spring Boot JPA is a Java specification for managing relational data in Java applications. It allows us to access and persist data between Java object/ class and relational database.  JPA is high level API and specification so that different ORM tools can implement so that it provides the flexibility to developer to change the implementation from one ORM to another | Hibernate understands the mappings that we add between objects and tables. It ensures that data is stored/retrieved from the database based on the mappings.  Hibernate also provides additional features on top of JPA. But depending on them would mean a lock in to Hibernate. | Object Relational Mapping (ORM) is concept/process of converting the data from Object oriented language to relational DB and vice versa.  In java its done with the help of reflection and jdbc. |

**15. What is indexing in Database?**

Indexing is a way to optimize the performance of a database by minimizing 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. Indexes are created using a few database columns.

**16. What are 4 types of Drivers in JDBC?**

JDBC Driver is a software component that enables Java application to interact with the database. There are 4 types of JDBC drivers:

1. **JDBC-ODBC bridge driver:**

The JDBC-ODBC bridge driver uses ODBC driver to connect to the database.

1. **Native-API driver (partially Java driver)**

The Native API driver uses the client-side libraries of the database

1. **Network Protocol driver (fully Java driver)**

The Network Protocol driver uses middleware (application server) that converts JDBC calls directly or indirectly into the vendor-specific database protocol. It is fully written in java.

1. **Thin driver (fully Java driver)**

The thin driver converts JDBC calls directly into the vendor-specific database protocol. That is why it is known as thin driver.

**17. What is difrence bew @component and @bean and @service?**

|  |  |  |
| --- | --- | --- |
| **@component** | **@bean** | **@service** |
| 1.Is an annotation that allows Spring to automatically detect our custom beans.  2.It is class level annotation | 1.Is applied on a method to specify that it returns a bean to be managed by Spring context. This annotation is part of the spring core framework.  2.It is method level annotations | 1.It is used to mark the class as a service provider.  2.@Service annotation is used with classes that provide some business functionalities. |

**18. What is difference between SOAP and restful services technically?**

SOAP (Simple Object Access Protocol) : It is a standardized protocol for certain rules to follow.  
By default SOAP is stateless but can be made stateful  
It is functional-driven, means that data here is available as services, eg, getUser  
SOAP API calls cannot be cached.   
It requires more bandwidth and computing power  
It only supports XML format.  
It works on HTTP, SMP, UDP, etc. So it is recommended  
SOAP cannot use REST because SOAP is a protocol and REST has an architectural style.  
  
RESTful (REpresentational State Transfer) services: It has an architectural style with less strict guidelines.   
REST is stateful.  
It is data-driven, meaning that data is available as resources.  
REST calls can be cached  
It requires less resources, and this makes it more powerful.  
It supports various formats like HTML, XML and JSON.  
It only works on HTTPS.  
REST can use SOAP as a protocol for web services.

**19. Difference in POST and PUT?**

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

**20. IS react uni directional or Bi directional?**

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. Thus react is unidirectional. This means: state is passed to the view and to child components, actions are triggered by the view, actions can update the state, the state change is passed to the view and to child components