**Java Questions.**

**J.Sairam Sampath**

**CAPG48LSRB313**

1. **What is difference in Overloading and Over riding ?**

**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.

**2)If both the functions are statis can we over ride?**

**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.

**3)Difference between Final , Finally and Finalize ?**

**Ans:**

**Final :** is a keyword and it can be used to mark a variable "unchangeable" . Actually, it is used to apply restrictions on class, method and variable. Final class can't be inherited, final method can't be overridden and final variable value can't be changed **.**

**Finally :**is a code block. It is used with try-catch block for handling exception. finally code block will be executed whether exception is handled or not  **. Finalize :**is a method of Object class. It is invoked before an object is discarded by the garbage collector , allowing it to clean up its state.

**4)What are maker Interfaces ?**

**Ans:** A marker interface is an interface that has no methods or constants inside it. It provides run-time type information about objects, so the compiler and JVM have additional information about the object .A marker interface is also called a tagging interface.

Though marker interfaces are still in use, they very likely point to a code smell and should be used carefully. The main reason for this is that they blur the lines about what an interface represents since markers don't define any behavior. Newer development favors annotations to solve some of the same problems.

**5)What is Singleton class ?**

**Ans:** In object-oriented programming, a singleton class is a class that can have only one object 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 Array List and Linked list?**

**Ans:** Both Array List and LinkedList are two different implementations of the List interface. Array List is a resizable-array implementation, whereas LinkedList is a Doubly-linked list implementation of the List interface.

Array List is a resizable array implementation in java. The backing data structure of Array List is an array of Object class. Like arrays, Linked List is a linear data structure.

**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. The simplest lambda expression contains a single parameter and an

expression: parameter -> expression

To use more than one parameter, wrap them in parentheses:

(parameter1, parameter2) -> expression

**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. From Java 8 onwards, lambda expressions can be used to represent the instance of a functional interface. A functional interface can have any number of default methods**.**

**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. This class has various utility methods to facilitate code to handle values as ‘available’ or ‘not available’ instead of checking null values.

Class Declaration

public final class Optional<T> extends Object

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

**Ans:** A Factory Pattern or Factory Method Pattern says that just define an interface or abstract class for creating an object but let the subclasses decide which class to instantiate. In other words, subclasses are responsible to create the instance of the class. An Adapter pattern acts as a connector between two incompatible interfaces that otherwise cannot be connected directly. An Adapter wraps an existing class with a new interface so that it becomes compatible with the client's interface

**12)What are Solid principles ?**

**Ans:** SOLID principles are object-oriented design concepts relevant to software development. SOLID is an acronym for five other class-design principles: Single Responsibility Principle, Open-Closed Principle, Liskov Substitution Principle, Interface Segregation Principle, and Dependency Inversion Principle.

**13) What does @component and @scan do in Sprinboot annaotation ?**

**Ans: @Component:** is an annotation that allows Spring to automatically detect our custom beans. In other words, without having to write any explicit code, Spring will: Scan our application for classes annotated with @Component. Instantiate them and inject any specified dependencies into them.Spring provides a mechanism to identify Spring bean candidates explicitly through the **@ComponentScan** :annotation. If the application is a Spring Boot application, then all the packages under the package containing the Spring Boot application class will be covered by an implicit component scan.

**14)What is the Difference between JPA , Hibernate and ORM?**

**Ans:**

**ORM:** Object Relational Mapping is concept/process of converting the data from Object oriented language to relational DB and vice versa. For example in java its done with the help of reflection and jdbc.

**Hibernate:** Its the implementation of above concept.

**JPA:** Its the one step above ORM. Its 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 (for example if application uses the JPA api and implementaion is hibernate. In future it can switch to IBatis if required. But on the other if application directly lock the implementation with Hibernate without JPA platform, switiching is going to be herculean task)

**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 ?**

**Ans:** 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

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:@**Bean

@Bean works in conjunction with a configuration class (with @Configuration) and thus in the annotation based configuration. It also is used on the methods inside of such a configuration class. Telling Spring to add whatever the method returns to the Spring Context. It’s done explicitly .By default, it will use the name of the method as the bean id/name. An alternative, you can specify it in the @Bean annotation.

@Component

@Component is used on our classes, so Spring knows that it should add it. However, it only works, if we enabled a component scan for our application and our class is included. With a component scan, Spring will scan the entire class path and will add all @Component annotated classes to the Spring Context (with adjustable Filtering).

@Service

We mark beans with @Service to indicate that they're holding the business logic. Besides being used in the service layer, there isn't any other special use for this annotation.

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

**Ans:** REST and SOAP are 2 different approaches to online data transmission. Specifically, both define how to build application programming interfaces (APIs), which allow data to be communicated between web applications. Representational state transfer (REST) is a set of architectural principles. Simple object access protocol (SOAP) is an official protocol maintained by the World Wide Web Consortium (W3C). The main difference is that SOAP is a protocol while REST is not. Typically, an API will adhere to either REST or SOAP, depending on the use case and preferences of the developer.

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

**Ans :**PUT method is called when you have to modify a single resource while POST method is called when you have to add a child resource.PUT method response can be cached but you cannot cache POST method responses.You can use UPDATE query in PUT whereas you can use create query in POST.In PUT method, the client decides which URI resource should have, and in POST method, the server decides which URI resource should have.PUT works as specific while POST work as abstract.If you send the same PUT request multiple times, the result will remain the same but if you send the same POST request multiple times, you will receive different results.PUT method is idempotent whereas POST method is not idempotent.

**20)IS react Uni directional or Bi directional ?**

**Ans :**Unidirectional data flow describes a one-way data flow where the data can move in only one pathway when being transferred between different parts of the program. React, a Javascript library, uses unidirectional data flow. The data from the parent is known as props. You can only transfer data from parent to child and not vice versa. This means that the child components cannot update or modify the data on their own, making sure that a clean data flow architecture is followed. This also means that you can control the data flow better.