Most Important Question for an interview:

Topic : Java

Q. What is the difference between method overloading and method overriding?

A. 1) Method overloading is used to increase the readability of the program. Method overriding is used to provide the specific implementation of the method that is already provided by its super class.

2) Method overloading is performed within class. Method overriding occurs in two classes that have IS-A (inheritance) relationship.

3) In case of method overloading, parameter must be different. In case of method overriding, parameter must be same.

4) Method overloading is the example of compile time polymorphism. Method overriding is the example of run time polymorphism.

5) In java, method overloading can't be performed by changing return type of the method only. Return type can be same or different in method overloading. But you must have to change the parameter. Return type must be same or covariant in method overriding.

Q. what are the advantage and disadvantage of overloading and overriding?

A. **Advantages of function overloading are as follows:**

• The main advantage of function overloading is that it improves code readability and allows code reusability.

• The use of function overloading is to save memory space, consistency, and readability.

• It speeds up the execution of the program

• Code maintenance also becomes easy.

• Function overloading brings flexibility to code

Disadvantage of function overloading are as follows:

• Function declarations that differ only in the return type cannot be overloaded

Q. Which is better approach overloading & overriding?

A. **Overloading** gives **better** performance compared to **overriding**. The reason is that the binding of **overridden** methods is being done at runtime. ... It means a class can have **more** than one private/final methods of same name but a child class cannot **override** the private/final methods of their base class.

Q what are the type of polymorphism?

A. the type of polymorphism are **run time** (overriding). **and compile time** (overloading) **polymorphism**

Q. Difference between run time and compile time polymorphism?

A. In **Compile time Polymorphism**, the call is resolved by the **compiler**. In **Run time Polymorphism**, the call is not resolved by the **compiler**. It is also known as **Static binding**, **Early binding** and overloading as well. It is also known as **Dynamic binding**, **Late binding** and overriding as well.

Q. difference between final,finally,finalize?

A. **Final** class can't be inherited, **final** method can't be overridden and **final** variable value can't be changed. **Finally** is used to place important code, it will be executed whether exception is handled or not. **Finalize** is used to perform clean up processing just before object is garbage collected.

Q. difference between abstract class and abstract method?

A. We use the **abstract** keyword to create **abstract classes** and **methods**. An **abstract method** doesn't have any implementation (**method** body). A **class** containing **abstract methods** should also be **abstract**. We cannot create objects of an **abstract class**

Q. What is the difference between functional interface and abstract class?

A. **Functional interface** are is used **in a** "safe" multiple inheritance. ... A **class** may extend multiple **functional** interfaces. **Functional** interfaces may have only a single **abstract** method. **Functional** interfaces may not have fields unlike C++ **abstract classes**.

Q. what is encapsulation and what are the different type of encapsulation?

A. **Encapsulation** is one of the fundamental concepts in object-oriented programming (OOP). It describes the idea of bundling data and methods that work on that data within one unit, e.g., a class in Java

Types of encapsulation

: • Member Variable Encapsulation: In Object Oriented Programming, all data members should be declared as Private members of the Class

• Function Encapsulation: Functions used only for internal implementation of your API must always be declared Private

. • Class Encapsulation: The same logic applies to classes used for internal implementations of your API. These classes should not be part of any public interface of an API. They should be hidden from your users and made Private

Q. what is abstraction and their types also?

A. Abstraction "hides" unnecessary information.

There are two types of abstraction.

- • Data **Abstraction**.
- • Process **Abstraction**.

Q. what is specifier in java ?

A. public, protected, private & default.

Q what is string and string builder?

A. Objects of **String** are immutable, and objects of StringBuffer and **StringBuilder** are mutable.

Q. **Difference between String Buffer and String Builder?**

A. String Buffer is synchronized i.e. thread safe. It means two threads can't call the methods of String Buffer simultaneously. String Builder is non-synchronized i.e. not thread safe. It means two threads can

call the methods of String Builder simultaneously.StringBuffer is less efficient than String Builder. String Builder is more efficient than String Buffer.

Q. what is the difference between collection and collections?

A. **Collection** is the interface where you group objects into a single unit. **Collections** is a utility class that has some set of operations you perform on **Collection.**

Q. what is the type of collection?

A. Collection types are the common variations of data collections, such as hash tables, **queues**, **stacks**, bags, dictionaries, and lists.

Q. difference between array list and linked list?

A. **ArrayList** internally uses a dynamic **array** to store its elements. **LinkedList** uses Doubly **Linked List** to store its elements. **ArrayList** is slow as **array** manipulation is slower. **LinkedList** is faster being node based as not much bit shifting required.

Q. difference between array and array list?

A. Array	ArrayList
Must include **System** namespace to use array.	Must include **System.Collections** namespace to use ArraList.
Array Declaration & Initialization: int[] arr = new int[5] int[] arr = new int[5]{1, 2, 3, 4, 5}; int[] arr = {1, 2, 3, 4, 5};	ArrayList Declaration & Initialization: ArrayList arList = new ArrayList(); arList.Add(1); arList.Add("Two"); arList.Add(false);
Array stores a fixed number of elements. The size of an Array must be specified at the time of initialization.	ArrayList grows automatically and you don't need to specify the size.
Array is strongly typed. This means that an array can store only specific type of items\elements.	ArrayList can store any type of items\elements.
No need to cast elements of an array while retrieving because it is strongly typed	The items of ArrayList need to be cast to an appropriate data type while retrieving. So, boxing and unboxing happens.

Q. what is thread and how it work?

A. Threads allows a program to operate more efficiently by doing multiple things at the same time.

Threads can be used to perform complicated tasks in the background without interrupting the main program.

There are two ways to create a thread.

- It can be created by extending the Thread class and overriding its run() method
- Another way to create a thread is to implement the Runnable interface.

Note:

Life cycle of thread include method are :

Start()-> to initiate

Run()->all the declaration

Stop ()-> stop the thread or dead state

wait() -> when thread execute another will run till it will in wait

Q. What is EJB and why it is used?

A. EJB is **a server-side software component that encapsulates business logic of an application**. An EJB web container provides a runtime environment for web related software components, including computer security, Java servlet lifecycle management, transaction processing, and other web services.

Q. What are the types of EJB?

A. There are three types of EJBs: Session Bean, Entity Bean, and Message-Driven Bean.

Q. What is a Comparator interface?

A comparator interface is **used to order the objects of user-defined classes**. A comparator object is capable of comparing two objects of two different classes

Q. Difference between Comparable and Comparator in Java

A. Comparator interface sort collection using two objects provided to it, whereas comparable interface compares" this" refers to the one objects provided to it.

Q. what is **Dependency Injection ?**

A. Dependency **Injection in Java** is a way to achieve **Inversion of control (IoC)** in our application by moving objects binding from compile time to runtime.

Q. What is inject annotation?

A. The @Inject annotation lets us define an injection point that is injected during bean instantiation.

Q. annotation in java?

A. **Annotations** are **used** to provide supplement information about a program. **Annotations** start with '@'. **Annotations** do not change action of a compiled program. **Annotations** help to associate metadata (information) to the program elements i.e. instance variables, constructors, methods, classes, etc.

Built-In Java Annotations used in Java code

- o @Override
- o @SuppressWarnings
- o @Deprecated

Built-In Java Annotations used in other annotations

- o @Target

- o @Retention
- o @Inherited
- o @Documented

Q. what are the scopes of java bean?

A. Scope & Description

1 Singleton: This scopes the bean definition to a single instance per Spring IoC container (default).

2 Prototype :This scopes a single bean definition to have any number of object instances.

3 Request : This scopes a bean definition to an HTTP request. Only valid in the context of a web-aware Spring ApplicationContext.

4 Session: This scopes a bean definition to an HTTP session. Only valid in the context of a web-aware Spring ApplicationContext.

5 global-session : This scopes a bean definition to a global HTTP session. Only valid in the context of a web-aware Spring ApplicationContext

Note: By default Java Bean use Singleton scope.

Q. what is **Inversion of control (IoC)?**

A. Inversion of Control (**IoC**) means to create instances of dependencies first and latter instance of a class (optionally injecting them through constructor), instead of creating an instance of the class first and then the class instance creating instances of dependencies.

Q. How does a hash map work?

A. A **HashMap** is a map used to store mappings of key-value pairs.

https://www.javatpoint.com/working-of-hashmap-in-java

Q. what are the annotation in spring?

A. the annotation in spring are:

1 @Required : The @Required annotation applies to bean property setter methods.

2 @Autowired: The @Autowired annotation can apply to bean property setter methods, non-setter methods, constructor and properties.

3 @Qualifier: The @Qualifier annotation along with @Autowired can be used to remove the confusion by specifiying which exact bean will be wired.

4 JSR-250 Annotations : Spring supports JSR-250 based annotations which include @Resource, @PostConstruct and @PreDestroy annotations.

Q. How does hibernate.cfg.xml works ?

A. In configuration class, we have the configure() method which loads the **hibernate**. **cfg**. **xml** file and returns SessionFactory reference, Which further used to get session reference. By default, **Hibernate** tries to find the file with the name **hibernate.**

Q. what are the hibernate properties?

A. Properties & Description

1 hibernate.dialect : This property makes Hibernate generate the appropriate SQL for the chosen database.

2 hibernate.connection.driver_class: The JDBC driver class.

3 hibernate.connection.url :The JDBC URL to the database instance.

4 hibernate.connection.username :The database username.

5 hibernate.connection.password: The database password.

6 hibernate.connection.pool_size: Limits the number of connections waiting in the Hibernate database connection pool.

7 hibernate.connection.autocommit: Allows autocommit mode to be used for the JDBC connection

Q. what is configure method in hibernate?

A. An instance of Configuration allows the application to specify properties and mapping documents to be used when creating a SessionFactory.

Q. what is ORM model?

A. Object–relational mapping (**ORM**, O/RM, and O/R mapping tool) in computer science is a programming technique for converting data between incompatible type systems using object-oriented programming languages.

Q Is ORM better than SQL?

A ORM and **SQL** are two tools available that web developers can use in database management. When comparing them, **SQL** has a **higher** hands-on management **than ORM**. Because **ORM** has a **higher** level of abstraction and more complexity **than SQL**, less hands-on management is required; this makes data management more efficient.

Q . **what are the method of object class in java?**

- **Object clone() : Creates and returns a copy of the existing class object.**
- **boolean equals(Object obj): Used to compare two objects of the same class. Returns true if both objects are equal or else false**
- **void finalize(): This method is called by the garbage collector on an object when garbage collection determines that there are no more references to the object exists.**
- **Class getClass():Returns a hash code value (numeric value) of the object of a class**

Q difference between abstract class and interface?

A **Abstract class** can inherit another **class** using extends keyword and implement an **interface**. **Interface** can inherit only an inteface. **Abstract class** can be inherited using extends keyword. **Interface** can only be implemented using implements keyword.

Q what is wrapper class?

A **Wrapper class** is a **class** which contains the primitive data types (int, char, short, byte, etc).

Q. What is json?

A. JSON stands for **J**ava**S**cript **O**bject **N**otation

JSON is a lightweight format for storing and transporting data

JSON is often used when data is sent from a server to a web page

JSON is "self-describing" and easy to understand

Q. what is synchronization?

A. Synchronization is the coordination of events to operate a system in unison. For example, the conductor of an orchestra keeps the orchestra synchronized or in time

Q. What is polymorphism?

A. Polymorphism in Java is a concept by which we can perform a single action in different ways. ... So polymorphism means many forms. There are two types of polymorphism in Java: compile-time polymorphism and runtime polymorphism. We can perform polymorphism in java by method overloading and method overriding.

Q.Why static keyword is used in main method in Java?

Java main() **method** is always **static**, so that compiler can call it without the creation of an object or before the creation of an object of the class. ... **Static method** of a class can be called by using the class name only without creating an object of a class

Q. difference between static and final?

The main **difference between** a **static and final** keyword is that **static** is keyword is used to define the class member that can be used independently of any object of that class. **Final** keyword is used to declare, a constant variable, a method which can not be overridden and a class that can not be inherited.

Q. What is the difference between error and exceptions?

A. **Errors**	**Exceptions**
Recovering from Error is not possible.	We can recover from exceptions by either using try-catch block or throwing exceptions back to caller.
All errors in java are unchecked type.	Exceptions include both checked as well as unchecked type.
Errors are mostly caused by the environment in which program is running.	Program itself is responsible for causing exceptions.
Errors occur at runtime and not known to the compiler.	All exceptions occurs at runtime but checked exceptions are known to compiler while unchecked are not.
They are defined in java.lang.Error package.	They are defined in java.lang.Exception package
Examples : java.lang.StackOverflowError, java.lang.OutOfMemoryError	Examples : Checked Exceptions : SQLException, IOException Unchecked Exceptions : NullPointerException, ArithmeticException.

Q difference between hashtable and hashmap?

A. HashMap	Hashtable
1) HashMap is non synchronized. It is not-thread safe and can't be shared between many threads without proper synchronization code.	Hashtable is synchronized. It is thread-safe and can be shared with many threads.
2) HashMap allows one null key and multiple null values.	Hashtable doesn't allow any null key or value.
3) HashMap is a new class introduced in JDK 1.2.	Hashtable is a legacy class.
4) HashMap is fast.	Hashtable is slow.
5) We can make the HashMap as synchronized by calling this code Map m = Collections.synchronizedMap(hashMap);	Hashtable is internally synchronized and can't be unsynchronized.
6) HashMap is traversed by Iterator.	Hashtable is traversed by Enumerator and Iterator.
7) Iterator in HashMap is fail-fast.	Enumerator in Hashtable is not fail-fast.

Q. What is difference between Stack and queue?

A. Stacks	Queues
Stacks are based on the LIFO principle, i.e., the element inserted at the last, is the first element to come out of the list.	Queues are based on the FIFO principle, i.e., the element inserted at the first, is the first element to come out of the list.
Insertion and deletion in stacks takes place only from one end of the list called the top.	Insertion and deletion in queues takes place from the opposite ends of the list. The insertion takes place at the rear of the list and the deletion takes place from the front of the list.
Insert operation is called push operation.	Insert operation is called enqueue operation.
Delete operation is called pop operation.	Delete operation is called dequeue operation.
In stacks we maintain only one pointer to access the list, called the top, which always points to the last element present in the list.	In queues we maintain two pointers to access the list. The front pointer always points to the first element inserted in the list and is still present, and the rear pointer always points to the last inserted element.
Stack is used in solving problems works on recursion.	Queue is used in solving problems having sequential processing.

Q REAL LIFE EXAMPLE OF STACK AND QUEQUE?

A. Queque:- Luggage checking machine: Luggage checking machine checks the luggage first that comes first. Therefore, it follows FIFO principle of queue.

• Stack :- To reverse a word. You push a given word to stack - letter by letter - and then pop letters from the stack.

Q. Difference between throw and throws?

A.Throw is a keyword which is used to throw an exception explicitly in the program inside a function or inside a block of code. Throws is a keyword used in the method signature used to declare an exception which might get thrown by the function while executing the code.

Q what is the difference between Comparator or comparable ?

A The main difference between Comparator and comparable:-

Comparable	Comparator
1) Comparable provides a **single sorting sequence**. In other words, we can sort the collection on the basis of a single element such as id, name, and price.	The Comparator provides **multiple sorting sequences**. In other words, we can sort the collection on the basis of multiple elements such as id, name, and price etc.
2) Comparable **affects the original class**, i.e., the actual class is modified.	Comparator **doesn't affect the original class**, i.e., the actual class is not modified.
3) Comparable provides **compareTo() method** to sort elements.	Comparator provides **compare() method** to sort elements.
4) Comparable is present in **java.lang** package.	A Comparator is present in the **java.util** package.
5) We can sort the list elements of Comparable type by **Collections.sort(List)** method.	We can sort the list elements of Comparator type by **Collections.sort(List, Comparator)** method

Q can we create object of abstract class?

A **We cannot create objects of an abstract class**. To implement features of an abstract class, we inherit subclasses from it and create objects of the subclass.

Q can we create constructor of abstract class?

A **We can declare a constructor with no arguments in an abstract class**. It will override the default constructor, and any subclass creation will call it first in the construction chain.

Q can we create constructor of interface? A **No, you cannot have a constructor within an interface in Java.**

Q What is Singleton pattern used for?

A In Java the Singleton pattern will ensure that there is only one instance of a class is created in the Java Virtual Machine. It is used **to provide global point of access to the object**. In terms of practical use Singleton patterns are used in logging, caches, thread pools, configuration settings, device driver objects.

Q what is session factory in hibernate?
A SessionFactory is **an Interface which is present in org. hibernate package and it is used to create Session Object**. It is immutable and thread-safe in nature. buildSessionFactory() method gathers the meta-data which is in the cfg Object.

Q what is factory design pattern?
A 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**.

Q What is ORM explain?

A Object–relational mapping (ORM, O/RM, and O/R mapping tool) in computer science is **a programming technique for converting data between incompatible type systems using object-oriented programming languages**. This creates, in effect, a "virtual object database" that can be used from within the programming language.

Q what is ean factory in spring?
A Beans are java objects that are configured at run-time by Spring IoC Container. **BeanFactory represents a basic IoC container which is a parent interface of ApplicationContext**.

Q what are the difference between == or equals?

A. Sr. No.	Key	==	equals() method
1	Type	== is an operator.	equals() is a method of Object class.
2	Comparisio n	== should be used during reference compariso n. == checks if both references points to same	equals() method should be used for content compariso n. equals() method evaluates the content to check

		location or not.	the equality.
2	Object	== operator can not be overriden.	equals() method if not present and Object.equals()