Java Interview Questions

Explain JVM, JRE and JDK?

JVM (Java Virtual Machine): It is an abstract machine. It is a specification that provides run-time environment in which java bytecode can be executed. It follows three notations:

Specification: It is a document that describes the implementation of the Java virtual machine. It is provided by Sun and other companies.
Implementation: It is a program that meets the requirements of JVM specification.
Runtime Instance: An instance of JVM is created whenever you write a java command on the command prompt and run the class.

JRE (Java Runtime Environment) : JRE refers to a runtime environment in which java bytecode can be executed. It implements the JVM (Java Virtual Machine) and provides all the class libraries and other support files that JVM uses at runtime. So JRE is a software package that contains what is required to run a Java program. Basically, it's an implementation of the JVM which physically exists.

includes an interpreter/loader, a compiler (javac), an archiver (jar), a documentation generator (javadoc) and other tools needed in Java development. In short, it contains JRE + development tools.

Explain public static void main(String args[]).

public : Public is an access modifier, which is used to specify who can access this method. Public means that this Method will be accessible by any Class.
static : It is a keyword in java which identifies it is class based i.e it can be accessed without creating the instance of a Class.
void : It is the return type of the method. Void defines the method which will not return any value.
main: It is the name of the method which is searched by JVM as a starting point for an application with a particular signature only. It is the method where the main execution occurs.
String args[] : It is the parameter passed to the main method.

Why Java is platform independent?

Why java is not 100% Object-oriented?

Java is not 100% Object-oriented because it makes use of eight primitive datatypes such as boolean, byte, char, int, float, double, long, short which are not objects.

What are wrapper classes?

Wrapper classes converts the java primitives into the reference types (objects). Every primitive data type has a class dedicated to it. These are known as wrapper classes because they “wrap” the primitive data type into an object of that class. Refer to the below image which displays different primitive type, wrapper class and constructor argument.

What are constructors in Java?

Can be declared with & without parameters. It is called constructor because it constructs the values at the time of object creation.

– Types:

– – : Default – Does not take as input any parameters

– – : Parameterized – Takes as input some parameters

What is singleton class and how can we make a class singleton?

Singleton class is a class whose only one instance can be created at any given time, in one JVM. A class can be made singleton by making its constructor private.

Why should Main method be static?

Main Method should be static so that it can be called without creating any object of that class.

What is a class?

– A class is an entity that determines how an object will behave and what the object will contain.

What is Encapsulation?

Encapsulation in Java is a mechanism of wrapping the data/variables and code acting on the methods together as a single unit. In encapsulation, the variables of a class will be hidden from other classes, and can be accessed only through the methods of their current class. Therefore, it is also known as data hiding.
Encapsulation is the packing of data and functions operating on that data into a single component and restricting the access to some of the object's components. Encapsulation means that the internal representation of an object is generally
hidden from view outside of the object's definition.

What is Abstraction?

Abstraction is a mechanism which represent the essential features without including implementation details. It is the quality of dealing with ideas rather than events.

– Encapsulation means-hiding data like using getter and setter etc.

– Abstraction means- hiding implementation using abstract class and interfaces etc.

What is the difference between equals() and == ?

equal() | equalsIgnoreCase : checks for the contents/data of the objects

== : checks for the references made to the objects

String vs StringBuffer vs StringBuilder

Objects of String are immutable whereas StringBuffer and StringBuider are mutable classes.

StringBuffer is thread safe and synchronized whereas StringBuilder is not, thats why StringBuilder is more faster than StringBuffer.

String concat + operator internally uses StringBuffer or StringBuilder class.

What is polymorphism?

Polymorphism – is the ability of an object to take on many forms.
The most common use of polymorphism in OOP occurs when a parent class reference is used to refer to a child class object.
Poly – means many and morph means form. Allows to perform various operations by using same method to perform different functions Polymorphism can be static or dynamic.
Static | Early Binding is performed during compilation time
Dynamic | Late Binding occurs during runtime, depending on the type of object.

Static Binding is achieved by overloading methods.
Dynamic Binding is achieved by overiding methods.

What is Method Overloading?

Overloading occurs when two or more methods in one class have the same method name but different parameters.
Overloaded methods must change parameters.
Overloaded methods can Change: return types & access modifier

Method overloading is used to increase the readability of the program. Method overloading is performed within class.

– In case of method overloading, parameter must be different.

– Method overloading is the example of compile time polymorphism.

What is Method Overiding?

Method Overriding is a technique of reimplementing or rewriting a method of a superclass in its subclass.

In Method Overriding, the java compiler does not decide which method is called by the user, since it has to wait till the object of the subclass is created.

After creating the object, JVM has to bind the method call to an app method.

But the methods in super and sub classes have the same name and same method signatures.

** JVM calls the method depending on the reference type of the object which is used to call the method.

– Method signature must be same

– The return type must be same

– We can change access modifier in a way: (Default -> protected -> public) – Cannot override private methdos

– cannot override final methods

Java does not allow to extend multiple classes. The problem with multiple inheritance is that if multiple parent classes have a same method name, then at runtime it becomes difficult for the compiler to decide which method to execute from the child class. Therefore, Java doesn't support multiple inheritance. The problem is commonly referred as Diamond Problem.

What is association?

Association is a relationship where all object have their own lifecycle and there is no owner. Let's take an example of Teacher and Student. Multiple students can associate with a single teacher and a single student can associate with multiple teachers but there is no ownership between the objects and both have their own lifecycle. These relationship can be one to one, One to many, many to one and many to many.

What do you mean by aggregation?

Aggregation is a specialized form of Association where all object have their own lifecycle but there is ownership and child object can not belongs to another parent object. Let's take

Why Java does not support pointers?

Pointer is a variable that refers to the memory address. They are not used in java because they are unsafe(unsecured) and complex to understand.
All objects in Java are references and you can use them like pointers.

Constructors of Super and subclass

– the superclass contructor is invoked first and then the subclass constructor
– basically every constructor invokes the constructor of its superclass with an implicity call to the super() method before invoking subclass constructor.
1] every constructor invokes the default constructor of the super class
2] if the subclass constructor wants to call the superclass constructor, then it should use:
– super() in case of DC
– super(parameter) in case of PC (parameterized constructor) – the super statements should be the first statements
3] if the subclass method wants to call the superclass method, then it should use: – super.methodname() (no parameter)
– super.methodname(parameters) (no of parameters)

It can be used to invoke or initiate current class constructor It can be passed as an argument in the method call

It can be passed as argument in the constructor call

It can be used to return the current class instance

What are abstract classes in Java?

Java Abstract classes are used to declare common characteristics of subclasses.

An abstract class cannot be instantiated.

It can only be used as a superclass for other classes that extend the abstract class.

Abstract classes are declared with abstract keyword.

They are used to provide a template or design for concrete subclasses down the inhertance tree.

– Rules

1] Illegal to delare an abstract method in a class which is not declared abstract

2] An abstract class can be declared without abstract method

3] A method can be declared as both abstract and final

4] A method cannot be declared as both abstract and private

5] If the subclass is abstract, then it is not mandatory to implement all abstract methods of a superclass

6] A non-abstract class must implement all the abstract methods of a superclass

An abstract class is a class which contains some abstract methods as well as concrete method.
An interface is a class that contains all methods which are abstract.
An interface is a reference type in Java and is similar to class.
It is a collection of abstract methods.
A class “implements” an interface, thereby inheriting the abstract methods of the interface.
Along with abstract methods, an interface may also contain constants, default methods, static methods, and nested types.
Method bodies exist only for default methods and static methods.
Writing an interface is similar to writing a class. But a class describes the attributes and behaviors of an object.
And an interface contains behaviors that a class implements.
Interfaces help to implement multiple inheritance
– Rules
1] All methods in interface are implicitly public and abstract.
They cannot be private, protected, static or final
2] Inferfaces cannot have body
3] An interface declares. only constant and not instance variables. All variables declared in an interface are public, final and static
4] Interface method should have access specifier public

What are Packages in Java?

A package in java is an encapsulation mechanism that can be used to group related classes, interfaces, enums and subpackages.

Class cannot have private or protected modifer.

default class cannot be accessed from outside the packages.

Difference Between throw & throws?

1] throws clause is used to declare an exception and throw keyword is used to throw an exception explicitly.

2] If we see syntax wise then throw is followed by an instance variable and throws is followed by exception class names.

3] The keyword throw is used inside method body to invoke an exception and throws clause is used in method declaration (signature).

Eg:

– throw

```
throw new Exception("You have some exception")
throw new IOException("Connection failed!!")
```

– throws

```
public int myMethod() throws IOException, ArithmeticException, NullPointerException {}
```

You cannot declare multiple exceptions with throw. You can declare multiple exception e.g. public void method()throws IOException,SQLException.

Explain Exception:

Exception is an abnormal condition that arises in a code sequence at run time i.e exception is a run time error

Types of errors:

1] Run time

2] Compile time

3] Logical errors

EH is a process of handling the exception object by using try catch block to prevent the program from abnormal termination and continue executing.

Object Class

– Throwable class

— Exception class — Error class

Exception class

– Run Time exception

– IOException

– Instantiation Exception (Abstract class or interfce is instantiated)

– Interrupted Exception (a thread is sleeping or waiting)

– IllegalAccessException (no permissions)

– No statement is allowed between try and catch

– Catch block argument is always of type throwable

– superclass exception should apprear after all of its subclasses – cannot have multiple

Autoboxing & Unboxing

Conversion of primitives to wrapper objects is called autoboxing. Integer n = 25;
Conversion of wrapper objects to primitives is called unboxing. int y = n;

Widening: byte->short->int->long->float->double
Autoboxing:
byte->Byte->Number->Object
short->Short->Number->Object
int->Integer->Number->Object

Arrays Vs Collections?

Arrays size are fixed where as collections are growable
Arrays can hold primitive and objects and collection can hold only objects
No readymade method support for arrays while Readymade method support is available (In terms of memory)
Arrays are not recommended to use In terms of performance while collections are recommended.

LIST INTERFACE:

————————-Arraylist————————-

Growable array

Limitations:

– Can contain any object, even though we want to store strings.

– we can add integer and boolean which is a problem.

– we use generis to specify the data type of the elements in the ArrayList. ArrayList <String> collectionName = New ArrayList<String>(); Collections.sort(collectionName)

Advantages:

– ArrayList can only contain String objects, and the compiler enforces this rule.

– the add method of ArrayList only accepts String references.

– you dont need to cast the data when accessing elements in the collection which improves the readibility and reliability of code.

Methods:

add(Object o)

add(int index, Object o)

remove(Object o)

remove(int index)

contains(Object element) //Returns true

get(int index) // Returns the element at pos

size()

the beginning or end, which makes it an easy choice for implementing a stack/queue.

addLast(Object o)
addFirst(Object o)
Collections.sort(i)
Collections.binarySearch(i,k)
Collections.reverse(l)
iterator

————————-Vector————————-

Vectors are same as ArrayList, but Vector methods are synchronized for thread safety. Its better to ArrayList instead of Vector because the synchronized methods adds a performance hit we might not need. Vector can be used in single threaded and multi threaded programs.

Set Interface:

– A set cares about uniqueness – it doesnt allow duplicates.
– The equals() method determines whether two objects are identical.

————————-Hash Set————————-

– Hashset is an unsorted & unordered set
– It uses hashcode of the object being inserted, so the more efficient the hashcode – implementation, the better access performance we get.
– We should use this when we want no duplicates and dont care about the order we iterate.
– We cannot determine o/p of the hashset program bcoz it is unordered and unsorted – we

————————-LinkedHashSet————————-

An ordered version of HashSet that maintains a doubly-linked List across all elements –
Use it when we care about the iteration of the order
** When we iterate through a HashSet the order is unpredictable while a LinkedHashSet lets you iterate through the elements in the order in which they were inserted
When using HashSet OR LinkedHashSet, the objects you add must override hashCode. Or else they will allow duplicates.

————————-TreeSet————————-

The TreeSet guarantees that the elements will be in ascending order/natural order. (Automatically Sorted)
We can also construct a TreeSet with a constructor that lets us give the collection our own rules for what the order should be using a comparable/comparator operator.

————————-Navigating TreeSets————————-

Another set of collections which can be searched and sorted they are called TreeSets / TreeMaps

Map Interface:

A map care about unique identifiers.
We can map a unique key (ID) to a specific value where both key,value are objects – We can search for a value based on the key,
– ask for a collection of just the values

.

HashMap is an unsorted, unordered Map.
The keys land in the Map is based on the Key's hashCode() implementation, the better we access performance
HashMap allows one null key and multiple null values in a collection.

Methods
– put(k Key,v Value)
– containsKey(Object key)
– containsValue(Object value) – isEmpty()
– size()

————————-HashTable————————-

Hashtable is the synchronized counterpart to HashMap
Can be ised on single threaded and multi threaded program – use concurrent hashMap for better performance

————————-LinkedHashMap————————-

The LinkedHashMap collection maintains insertion order (access order)
Slower than HashMap for adding and removing elements, we can accept faster iteration with a linkedHashMap

————————-TreeMap————————-

TreeMap is a sorted Map, this means "sorted by the natural order of the elements".
Just like TreeSet, TreeMap lets you define a custom sort order (Comparable/Conparator)

– Support all of the standard collection methods and also add methods for add and subtract

————————-Priority Queue————————-

– PriorityQueue is to create a “Priority-in, Priority out” queue as opposed to typical FIFO
– elements are ordered either by natural ordering (in which case the elements sorted first will be accessed first).
– Elements ordering represents their relative property.

Methods
add() // Inserts the specified element in the queue
offer() // Inserts the specified element in the queue
clear() //Remove all elements
poll() // Retrives and removes the head of this queue
peek() // retrives and displays head of the queue
size()

– We can determine the OP
– We can add same value more than once

————————-Dequeue Interface————————-

A linear collection that supports element insertion and removal at both ends
Double Ended Queue
This interface defines methods to access the elements at both ends of the deque – Insert, Remove, examine

Not thread safe, in the absense of external synchronization , they do not support concurrent access by multiple threads

— addFirst() offerFirst() addLast() offerLast() getFirst() peekFirst() getLast() peekLast() removeFirst() poolFirst removeLast() poolLast()

LIFO

– Dequeues can also be used as LIFO (Last In First Out) Stack

– This Interface should be used in preference to the legacy stack class

– when a dequeue is used as stack, elements are pushed and popped from the beginning

— push() addFirst() pop() removeFirst() peek() peekFirst()

FIFO

– Can be used as a queue, FIFO behavior.

– Elements are added at the end of the deque and removed from the beginning.

— add() addLast() remove() removeFirst() peek() peekFirst()

What is Multithreading?

Java is a multi-threaded programming language which means we can develop multi-threaded program using Java.

A multi-threaded program contains two or more parts that can run concurrently and each part can handle a different task at the same time making optimal use of the available resources specially when your computer has multiple CPUs.

the threads can run in parallel.

What is a HashMap?

HashMap is a Map based collection class that is used for storing Key & value pairs, it is denoted as HashMap<Key, Value> or HashMap<K, V>.
This class makes no guarantees as to the order of the map. It is similar to the Hashtable class except that it is unsynchronized and permits nulls(null values and null key).
It is not an ordered collection which means it does not return the keys and values in the same order in which they have been inserted into the HashMap. It does not sort the stored keys and Values.
You must need to import java.util.HashMap or its super class in order to use the HashMap class and methods.

What is HashTable?

To successfully store and retrieve objects from a hashtable, the objects used as keys must implement the hashCode method and the equals method.
It is similar to HashMap, but is synchronised.

The key is then hashed, and the resulting hash code is used as the index at which the value is stored within the table.

What is Hashset?

HashSet is an implementation of Set. A Set is designed to match the mathematical model of a set. A HashSet does use a HashMap to back its implementation, as you noted. However, it implements an entirely different interface. HashSet class implements the Set interface

In HashSet, we store objects(elements or values) e.g. If we have a HashSet of string elements then it could depict a set of HashSet elements: {“Hello”, “Hi”, “Bye”, “Run”}

HashSet does not allow duplicate elements that mean you can not store duplicate values in HashSet. HashSet permits to have a single null value.

HashSet is not synchronized which means they are not suitable for thread-safe operations until unless synchronized explicitly. [similarity]