CORE JAVA

Q.What is Java?

Java is a **programming language** and a **platform**. Java is a high level, robust, object-oriented and secure programming language.

Java was developed by *Sun Microsystems* (which is now the subsidiary of Oracle) in the year 1995. ***James Gosling*** is known as the father of Java. Before Java, its name was *Oak*. Since Oak was already a registered company, so James Gosling and his team changed the name from Oak to Java?

Platform: Any hardware or software environment in which a program runs, is known as a platform. Since Java has a runtime environment (JRE) and API, it is called a platform.

Q.What is features of Java?

1) **Simple**

   Java is very easy to learn, and its syntax is simple, clean and easy to understand. According to Sun Microsystems, Java language is a simple programming language.

2) **Object-oriented**

   Java is an object-oriented programming language. Everything in Java is an object. Object-oriented means we organize our software as a combination of different types of objects that incorporate both data and behavior.

3) **Platform Independent**

   Java is platform independent because it is different from other languages like C, C++, etc. which are compiled into platform specific machines while Java is a (WORA) write once, run anywhere language. A platform is the hardware or software environment in which a program. There are two types of platforms software-based and hardware-based. Java provides a software-based platform.

4) **Secured**

   Java is secured because it doesn't use explicit pointers. Java provides more security than other language.

5) **Multi-threaded**
Simultaneously works on multiple executions.

6) **Robust**

Java is robust because: having robusting nature, it uses strong memory management.

7) **Architecture-neutral**

Java is architecture neutral because there are no implementation dependent features, for example, the size of primitive types is fixed.

8) **Portable**
We can transfer .class file from one platform (os) to another platform (os) it means java is portable. Java supports (ROWA) read-once-write-anywhere approach.

9) **High-performance**

Java is faster than other traditional interpreted programming languages because Java byte code is "close" to native code.

10) **Dynamic**

Java is a dynamic language. It supports the dynamic loading of classes. It means classes are loaded on demand.

Parameters used in First Java Program

```
class Simple{
public static void main(String args[]){
System.out.println ("Hello Java");
}
}
```

- **Class** keyword is used to declare a class in Java.
- **Public** keyword is an access modifier that represents visibility. It means it is visible to all.
- **Static** is a keyword. Once we write static method then we can overload by using static method, we can't override.
- **Void** is the return type of the method. It means it doesn't return any value.

- **Main** represents the starting point of the program.
- **String** is class and this class is available in .lang package.
- **args** it as argument .we can pass anything here.
- **[]** is used for store results.
- **System.out.println ()** is used to print statement. Here, System is a class; this class is available in .lang package, .lang package by default available in java.
- out is an object of the PrintStream class,
- println() is a method of the PrintStream class

Q.What is JDK?

The Java Development Kit (JDK) is a software development environment which is used to develop Java applications and applets. It physically exists. It contains JRE + development tools.

The JDK contains a private Java Virtual Machine (JVM) and a few other resources such as an interpreter/loader (java), a compiler (javac), an archiver (jar), a documentation generator (Javadoc), etc. to complete the development of a Java Application.

- Standard Edition Java Platform
- Enterprise Edition Java Platform
- Micro Edition Java Platform

Q.What is JRE?

The Java Runtime Environment is a set of software tools which are used for developing Java applications. It is used to provide the runtime environment. It is the implementation of JVM. It physically exists. It contains a set of libraries + other files that JVM uses at runtime.

Q.What is JVM (Java Virtual Machine) Architecture

JVM (Java Virtual Machine) is an abstract machine. It is a specification that provides runtime environment in which java byte code can be executed.

JVMs are available for many hardware and software platforms (i.e. JVM is platform dependent).

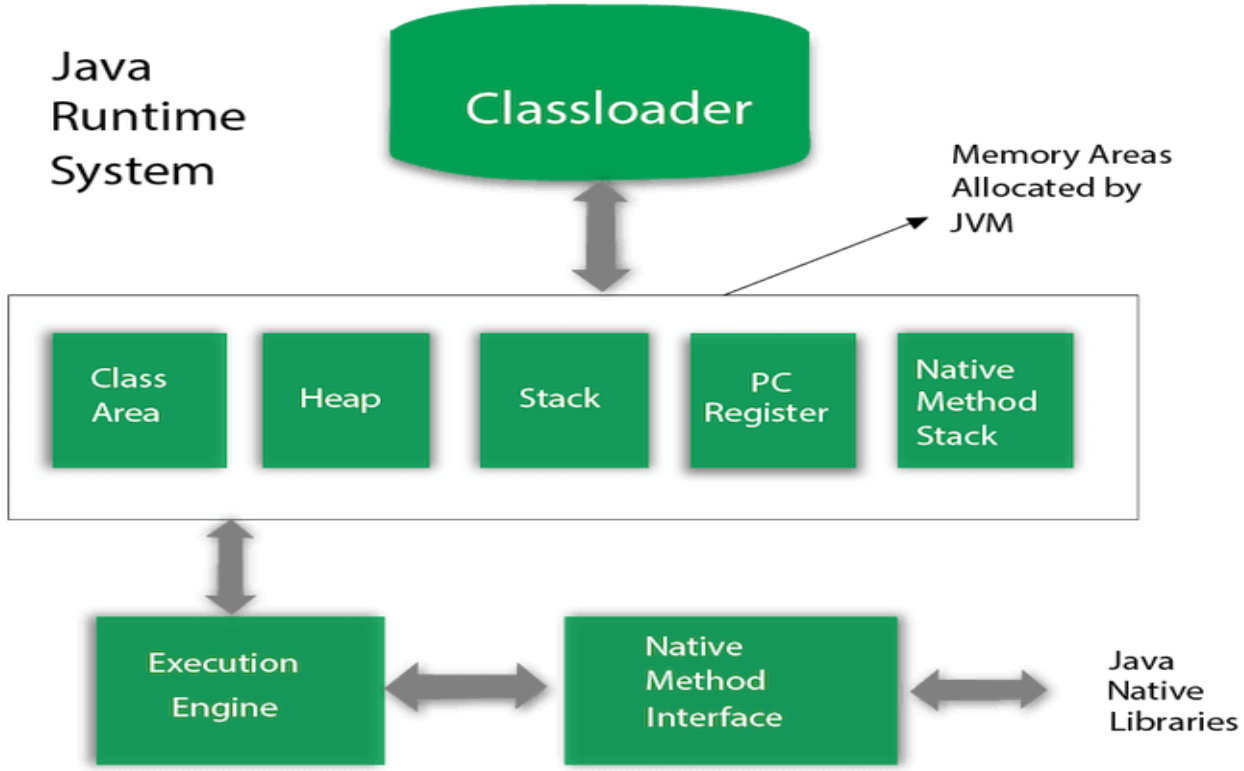
What is JVM

It is:

A specification- where working of Java Virtual Machine is specified. But implementation provider is independent to choose the algorithm. Its implementation has been provided by Oracle and other companies.

An implementation -Its implementation is known as JRE (Java Runtime Environment).

Runtime Instance- Whenever you write java command on the command prompt to run the java class, an instance of JVM is created.



Q.What is class loader?

Classloader is a subsystem of JVM which is used to load class files. Whenever we run the java program, it is loaded first by the classloader. There are three built-in classloaders in Java.

1) **Bootstrap ClassLoader**:
   This is the classloader which is the super class of Extension classloader. It loads the rt.jar file.

2) **Extension ClassLoader**:
   It is the ClassLoader which loads the jar files present in the directory. This is the child Bootstrap classLoader and parent of System classloader.
3) **System/Application ClassLoader**:
   It is the classLoader which loads the class files from the classpath. This is the child of the Extension classloader.

Q.How many types of memory areas are allocated by JVM?

1) Class (Method) Area

Class (Method) Area stores per-class structures such as the runtime constant pool, field and method data, the code for methods.

2) Heap

It is the runtime data area in which objects are allocated.

3) Stack

Java Stack stores frames. It holds local variables and partial results, and plays a part in method invocation and return.

4) Program Counter Register

PC (program counter) register contains the address of the Java virtual machine instruction currently being executed.

5) Native Method Stack

It contains all the native methods used in the application.

Execution Engine

A virtual processor
Interpreter: Read bytecode stream then execute the instructions.
Just-In-Time (JIT) compiler: It is used to improve the performance.

Q.What is JIT compiler?

Just-In-Time (JIT) compiler: It is used to improve the performance. JIT compiles parts of the bytecode that have similar functionality at the same time, and hence reduces the amount of time needed for compilation. Here the term "compiler" refers to a translator from the instruction set of a Java virtual machine (JVM) to the instruction set of a specific CPU.

Q. What is Class?

It is a predefined template .it is collection of objects, variable and methods. **Or**

A class is a group of objects which have common properties.

Class is not real word entity. It is a just template or blueprint or prototype.

Class does not memory occupy.

Q. What is Objects?

The object is *an instance of a class*.
An object is *a real-world entity*.
An object is *a runtime entity*.
The object is *an entity which has state and behavior*.

For example: Pen is an object. Its name and; color, known as its state. So writing is its behavior.

Q.What are the different ways to create objects in java?

1) By new keyword
2) By newInstance () method
3) By clone () method
4) By deserialization
5) By factory method

Q. Why to Create Object?

Objects are required in oops,because they are create to call the non static method, which is not represent inside the main method but it is represent inside the class.

Q.What is a method in Java?

A **method** is a block of code or collection of statements or a set of code grouped together to perform operation. It is used to achieve the **reusability** of code. once we write a method and use it many times. The most important method in Java is the **main()** method.

Method Declaration

The method declaration provides information about method attributes, such as visibility, return-type, name, and arguments.

Types of Method

There are two types of methods in Java:

1) Predefined Method

In Java, predefined methods are the method that is already defined in the Java class libraries is known as predefined methods. It is also known as the **standard library method** or **built-in method**.

2) User-defined Method

The method written by the user or programmer is known as **a user defined** method

Q.What is Data Types in Java

1) byte data type-

The byte data type is used to save memory in large arrays. Byte size is the 1 byte.

2) short data type-

The short data type can also be used to save memory just like byte data type. Short size is the 2 byte.

3) int data type-

The int data type is generally used as a default data type for integral values. Int size is the 4 byte.

4) long data type-
The long data type is used when you need a range of values more than those provided by int. Long size is the 8 byte.

5) float data type-
Its value range is unlimited. Float size is the 4 byte.

6) double data type-
Its value range is unlimited. The double data type is generally used for decimal values just like float. Double size is the 8 byte.

7) char data type-
The char data type is used to store characters. Char size is the 2 byte.

8) Boolean data type-
The Boolean data type is used to store only two possible values: true and false. Boolean size is the 1 bit.

Q.Tell me all default value of data type?

Byte=0, Short=0, Int=0, Long=0L, Float=0.0f, Double=0.0, Char='/u0000', Boolean=false.

Q.What are the operators in Java

Operators are used to perform operations on variables and values.

For example: **(+, -, *, /)** etc.

1) Java Unary Operator (++ and --)

Unary operators are used to perform various operations i.e.:

- incrementing/decrementing a value by one
- negating an expression
- inverting the value of a Boolean.

2) Java Arithmetic Operators

Java arithmetic operators are used to perform addition, subtraction, multiplication, and division **(+, -, *, /)**

3) Java Relational Operators

Comparison/relational operators are used to compare two values (==, !=, <, > ,>=, <=)

4) Java Assignment Operator

Assignment operators are used to assign values to variables.(= , += , -= , *= ,/= ,%= , &=, ^= , |= , <<= , >>= , >>>=)

5) Java logical Operator

Logical operators are used to determine the logic between variables or values:

logical AND	&&
logical OR	||

6) Java Bitwise Operator

Bitwise operator works on bits and performs the bit-by-bit operation. There are six types of the bitwise operator in Java:

- Bitwise AND (&)
- Bitwise exclusive OR (^)
- Bitwise inclusive OR (|)
- Bitwise Compliment (~)====`
- Bit Shift Operators (<< , >>)

7) Java Ternary Operator

It is the only conditional operator **(?, :)**

STATIC CONCEPTS

Q.What is Static keyword

The **static keyword** in Java is used for memory management mainly.

The static can be:

1. Variable (also known as a class variable)
2. Method (also known as a class method)
3. Block
4. Nested class

Q.What is the static variable?

Variable is a name of memory location. Static variable belongs to the class rather than the object.

There are three types of variables in java:

a) Local Variable

A variable declared inside the body of the method is called local variable. A local variable cannot be defined with "static" keyword.

b) Instance Variable

A variable declared inside the class but outside the body of the method, is called an instance variable. It is not declared as static.

c) Static variable

A variable that is declared as static is called a static variable. It cannot be local. You can create a single copy of the static variable and share it among all the instances of the class. Memory allocation for static variables happens only once when the class is loaded in the memory.

Q.Difference between static and non static methods in java?

Instance variable	Static variable
1) Instance variable are class level.	Static variable are class level.
2) Instance variable can access with object reference. And non-static variable.	Static variable can access with class reference. And static & non-static variable.

Q.What is the static method?

Static method allows to overload, but it can't allows override.
Static method in Java is a method which belongs to the class and not to the object.
A static method can access only static data. It cannot access non-static data (instance variables).

- A static method can call only other static methods and cannot call a non-static method from it.
- A static method can be accessed directly by the class name and doesn't need any object
- A static method cannot refer to "this" or "super" keywords in anyway.

Syntax:

<class-name>.<method-name>

Q.Difference between static and non static methods in java

Static method	Non-static method
1) A static method belongs to the class	Non-static method belongs to an object of a class.
2) A static method can access only static members.	Non-static method can access both static and non-static members.
2) Static method is having only one instance where you're going to use the method, and you don't need multiple copies (objects).	Non-static method is used if you're going to use your method to create multiple copies.

Q. What is the static block?

The static block is a block of statement .it is executed before the main method at time of class loading.

Q. What is the Nested Class

Nested static class doesn't need a reference of outer class. Outer class never static, it allows to use inner class as a static.

Q. Why main method is static?

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.

Q.Can we override the static method?

No, you can't override the static method because they are the part of the class, not the object.

Q. Why can we not override static method?

It is because the static method is the part of the class, and it is bound with class whereas instance method is bound with the object, and static gets memory in class area, and instance gets memory in a heap.

Q.What is Final keyword in java?

Java final keyword is a non-access specifier that is used to restrict :

1. Variable
2. Method
3. Class.

Q.What is the final variable?

If we initialize the final variable, we can't change its value.

Q.What is the final method?

If we change any method to a final method, we can't override it.

Q.What is the final class?

If you make any class as final, you cannot extend it.

Q.What is this keyword in java?

The this keyword is a reference variable that refers to the current object. There are the various uses of this keyword in Java. It can be used to refer to current class properties such as instance methods, variable, constructors, etc

Q.What are the main uses of this keyword?

There are the following uses of **this** keyword.

- **this** can be used to refer to the current class instance variable.
- **this** can be used to invoke current class method (implicitly)
- **this()** can be used to invoke the current class constructor.
- **this** can be passed as an argument in the method call.
- **this** can be passed as an argument in the constructor call.
- **this** can be used to return the current class instance from the method.

Q.What is super keyword in java?

The **super** keyword in Java is a reference variable that is used to refer to the immediate parent class object. Whenever you create the instance of the subclass, an instance of the parent class is created implicitly which is referred by super reference variable. The super () is called in the class constructor implicitly by the compiler if there is no super or this.

Q.What are the main uses of the super keyword?

There are the following uses of super keyword.

- super can be used to refer to the immediate parent class instance variable.
- super can be used to invoke the immediate parent class method.
- Super () can be used to invoke immediate parent class constructor.

Q. Difference between this and super keyword?

this keyword	super keyword
1) "**this"** keyword is a reference variable that refers to the current object.	"Super" is a reference variable that refers to immediate parent class objects.
2) **this** can be used to refer to the current class instance variable.	super can be used to refer to the immediate parent class instance variable.
3) **this** can be used to invoke current class method (implicitly) & constructor.	super can be used to invoke the immediate parent class method & constructor.

Q. Use of Volatile keyword?

- Modify the value of variable by different threads.
- It is used to make class thread safe.
- It is used with primitive data type or object.
- Volatile keyword doesn't cached the value of variable and always read variable from main memory.

Q. Difference between Volatile and Synchronized keyword?

Volatile Keyword	Synchronized keyword
1) Volatile keyword is a field modifier.	Synchronized keyword modifies code block and methods.
2) Improve thread performance.	It degrades the thread performance.
3) The thread cannot be blocked for waiting in case of volatile.	Threads can be blocked for waiting in case of synchronized.

Q.Can you use this() and super() both in a constructor?

No, because this () and super () must be the first statement in the class constructor.

Q.What is object cloning?

The object cloning is used to create the exact copy of an object. The clone () method of the Object class is used to clone an object.

The **java.lang.Cloneable** interface must be implemented by the class whose object clone we want to create. If we don't implement Cloneable interface, clone () method generates CloneNotSupportedException.

Q.What is aggregation?

Aggregation can be defined as the relationship between two classes where the aggregate class contains a reference to the class it owns.

Q.What is composition?

Holding the reference of a class within some other class is known as composition.

Q. Difference Aggregation and Composition?

The relation between parent and child class,
Aggregation implies a relationship child can exist independently of the parent.

e.g.: Bank & Employee, delete the bank and employees still exist.

Whereas Composition implies a relationship child cannot exist independently of the parent.

Java Control Statements | Control Flow in Java

1) **if statement**

if statement is used to test the condition. it checks Boolean condition: true or false. It executes the if block if condition is true.

Syntax:

```
if (condition) {
//code to be executed
}
```

2) if – else statement

If-else statement also tests the condition. It execute the if block if condition is true otherwise else block is executed.

Syntax:

```
if(condition){
//code if condition is true
}else{
//code if condition is false
}
```

3) Nested-if statement

The nested if statement represents the *if block within another if block*. Here, the inner if block condition executes only when outer if block condition is true.

Syntax:

```
if(condition){

//code to be executed
if(condition){
//code to be executed
}}
```

1) Switch statement

The Java switch statement executes one statement from multiple conditions.

Syntax:

```
switch (expression) {
case value:
// Statements
break;

case value :
// Statements
```

```
break;

// you can have any number of case statements.
default :
// Statements
}
```

Java Break Statement

The break statement can also be used to jump out of a **loop**.

Syntax:

```
jump-statement;
break;
```

Java Continue Statement

The continue statement is used to be jump to the next iteration of the loop immediately. It can be used with for loop or while loop.

Syntax:

```
jump-statement;
continue;
```

Looping statement in java

1) For loop statement

This statement which allows code to be repeatedly executed. For loop three part Initialization, Condition and Increment/Decrement.

Syntax:

```
for(initialization ; condition ; increment/decrement){
//statement or code to be executed
}
```

2) While loop statement

While loop in java first check the condition if condition is true then control goes inside the loop body otherwise goes outside of the body.

Syntax:

```
while (condition){
//code to be executed
Increment / decrement statement
}
```

3) Do-while statement

A do-while check for the condition after executing the statements or the loop body.

Syntax:

```
do {
//code to be executed / loop body
//update statement
} while (condition);
```

Q.What are the Access Modifiers in Java

There are four types of Java access modifiers:

1. **Private**: The private access modifier is only within the class. It cannot be accessed from outside the class.
2. **Default**: The default access modifier is only within the package. It cannot be accessed from outside the package. If you do not specify any access level, it will be the default.
3. **Protected**: The protected access modifier is within the package and outside the package through child class. If you do not make the child class, it cannot be accessed from outside the package.
4. **Public**: The public access modifier is everywhere. It can be accessed from within the class, outside the class, within the package and outside the package.

Q.What is Type Casting in Java

Convert a value from one data type to another data type is known as type casting.

There are two types of type casting:

1) **Up Casting(automatically / Wedding)**
   Converting a lower type to a higher type. (Int to double).

2) **Down Casting(manually / Narrowing)**
   Converting a higher type to lower type. (Double to int).

Q. What is Wrapper Class (auto boxing unboxing) use of Wrapper Class?

A Wrapper class provides the mechanism *to* convert primitive into object and object into primitive.

Two uses of Wrapper Class:

- To convert simple data types into objects, that is, to give object form to a data type; here constructors are used.
- To convert strings into data types (known as parsing operations).

Q.What is autoboxing and unboxing? When does it occur?

The autoboxing is converting primitive data type to wrapper class object.

Example: int to Integer.

The unboxing is converting wrapper class object to primitive data type.

Example: Integer to int.

Unboxing and autoboxing occur automatically in Java. However, we can externally convert one into another by using the methods like valueOf () or xyzValue ().

OOPS CONCEPTS

Q.Explian OOPS Concepts (4 pillars of oops)

1) Abstraction

Abstraction is a process of hiding the implementation details and showing only functionality to the user. For example, sending SMS where you type the text and send the message. You don't know the internal processing about the message delivery. In Java, there are two ways to achieve the abstraction.

Abstract class	Interface
1) Abstract class can have abstract and non-abstract methods.	Interface can have only abstract methods. Till JDK 1.7 from JDK 1.8 it can have default and static methods also.
2) Abstract class doesn't support multiple inheritances.	Interface supports multiple inheritances.
3) Abstract class can have final, non-final, static and non-static variables.	Interface has only static and final variables.
4) Abstract class can provide the implementation of interface.	Interface can't provide the implementation of abstract class.
5) The abstract keyword is used to declare abstract class.	The interface keyword is used to declare interface.
6) An abstract class can extend another Java class and implement multiple Java interfaces.	An interface can extend another Java interface only.
7) An abstract class can be extended using keyword "extends".	An interface can be implemented using keyword "implements".
8) A Java abstract class can have class members like private, protected, etc.	Members of a Java interface are public by default.
9) In class will extends only one class at a time.	In class will implements n number of interface by using **“,”** separated.
10) The abstract class has allows constructor.	The interface has no constructor.
11)**Example:** public abstract class Shape {public abstract void draw(); }	**Example**: public interface Drawable {void draw();}

Q.Can there be an abstract method without an abstract class?

No, if there is an abstract method in a class, that class must be abstract.

Q.Can you declare an interface method static?

No, because methods of an interface are abstract by default, and we cannot use static and abstract together.

Q.Can the Interface be final?

No, because an interface needs to be implemented by the other class and if it is final, it can't be implemented by any class.

Q.Why An Interface Cannot Have Constructor In Java?

Interface cannot have constructor in java, because interfaces not have any instance member so nothing to construct.

2) Encapsulation in Java

Encapsulation in Java is a process of wrapping code and data together into a single unit, for example, a capsule which is mixed of several medicines.

We can create a fully encapsulated class in Java by making all the data members of the class private. Now we can use setter and getter methods to set and get the data in it.

3) Polymorphism in Java

The ability to take more than one from is called polymorphism.

There are two types of polymorphism in Java:

1) Compile-time polymorphism [method overloading].
2) Runtime polymorphism [method overriding.].

No.	Method Overloading	Method Overriding
1)	Same method name but different parameters, it is known as **Method Overloading**.	Same method name and same parameters, it is known as **method overriding.**
2)	Method overload will achieve in single class.	A single class we are not able to achieve method override.
3)	Static allows method overload.	Static not allows to method override.
4)	Final allows method overload.	Final not allows to method override.
5)	Method overloading is the example of compile time polymorphism.	Method overriding is the example of run time polymorphism.
6)	It allows any return type.	Return type must be same.

Q. Which is better to hide data Encapsulation or Abstraction? If no why?

Encapsulation is better for hiding of data. Because it's provide security to data making variables as private.

Q.Can we overload the main () method?

Yes, we can have any number of main methods in a Java program by using method overloading.

4) Inheritance in Java

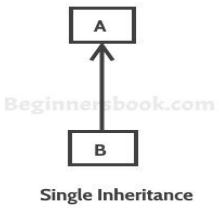
Inheritance in Java is a one Class can acquire the properties of another Class (e.g.: Parent Child). **Or**

Child (sub class or derive) class acquires all the properties of parent (super class or base) class.

Types of inheritance in java

A) Single Inheritance

When a class extends another one class it is known as a single inheritance. The below flow diagram shows that class B extends only one class which is A.



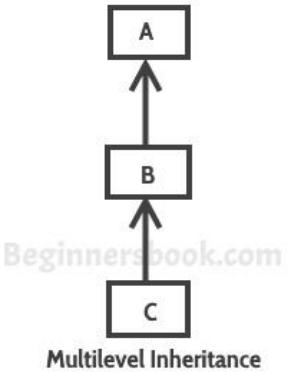
Single Inheritance

Example:

```
class A{
void eat() { System.out.println("eating..."); }
}
class B extends A{
void bark(){System.out.println("barking...");}
}
class TestInheritance{
public static void main(String args[]){
B d=new B();
d.bark();
d.eat();
}}
```

B) Multilevel Inheritance

When a class extends a class, which extends anther class then this is called multilevel inheritance**.** For example class C extends class B and class B extends class A then this type of inheritance is known as multilevel inheritance.

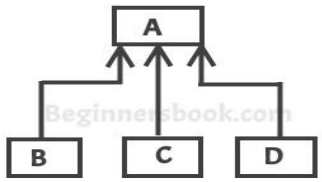


Example:

```
class A{
void eat(){System.out.println("eating...");}
}
class B extends A{
void bark() { System.out.println("barking..."); }
}
class C extends B {
void weep() { System.out.println("weeping..."); }
}
class TestInheritance2{
public static void main(String args[]){
C d=new C ();
d.weep();
d.bark();
d.eat();
}}
```

C) Hierarchical Inheritance

When more than one classes extends a same class then this is called hierarchical inheritance. For example class B, C and D extends a same class A.



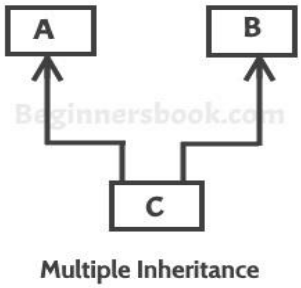
Hierarchical Inheritance

Example:

```
class A{
void eat() { System.out.println("eating..."); }
}
class B extends A{
void bark() { System.out.println("barking..."); }
}
class C extends A{
void meow() { System.out.println("meowing..."); }
}
class TestInheritance3{
public static void main(String args[]){
C c=new C();
c.meow();
c.eat();
//c.bark();//C.T.Error
}}
```

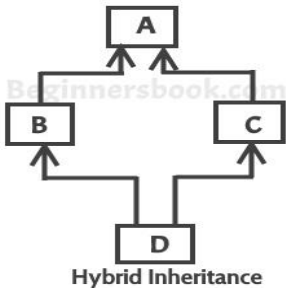
4) Multiple Inheritance

When one class extends more than one classes then this is called **multiple inheritance**. For example: Class C extends class A and B then this type of inheritance is known as multiple inheritances.



5) Hybrid Inheritance

A hybrid inheritance is a combination of more than one types of inheritance. For example when class A and B extends class C & another class D extends class A then this is a hybrid inheritance, because it is a combination of single and hierarchical inheritance.



Q.What is the package?

A package is a group of similar type of classes, interfaces, and sub-packages. It provides access protection and removes naming collision. The packages in Java can be categorized into two forms, inbuilt package, and user-defined package.

Q.What is the interface?

The interface is a blueprint for a class that has static constants and abstract methods. It can be used to achieve full abstraction and multiple inheritances.

Q.Why is inheritance used in java?

Inheritance provided code Re-usability.

Runtime polymorphism cannot be achieved without inheritance.

Inheritance provided data hiding.

Q.Why is multiple inheritances not supported in java?

To reduce the complexity and simplify the language, multiple inheritance is not supported in java. Consider a scenario where A, B, and C are three classes. The C class inherits A and B classes. If A and B classes have the same method and you call it from child class object, there will be **ambiguity** to call the method of A or B class.

Example:

```
class A{
void msg(){System.out.println("Hello");}
}
class B{
void msg(){System.out.println("Welcome");}
}
class C extends A,B{//suppose if it were

Public Static void main(String args[]){
C obj=new C();
obj.msg();//Now which msg() method would be invoked?
}
}
```

Q.Which class is the superclass for all the classes?

The object class is the superclass of all other classes in Java.

CONSTRUCTOR

Q.What is the constructor?

1. Constructor is used to initialize the state of an object.

2. A constructor in Java is a block of code similar to a method.

3. Constructor name must be the same as its class name.

4. Constructor does not have any return type.

5. Constructor allows overload.

6. Constructor does not allow override.

Q.How many types of constructors are used in Java?

a) Default Constructor: default constructor is the one which does not accept any value. The default constructor is mainly used to initialize the instance variable with the default values. A default constructor is invoked implicitly by the compiler if there is no constructor defined in the class.

b) Parameterized Constructor: The parameterized constructor is the one which can initialize the instance variables with the given values.

Q. What is the purpose of default constructor?

The purpose of data –is assign the default value to the objects.

The java compiler creates a default constructor implicitly if there is no constructor is the class.

Q.Does constructor return any value?

Yes: The constructors implicitly return the current instance of the class (you cannot use an explicit return type with the constructor).

Q. Is constructor inherited?

No: the constructor not inherited, because of subclass name is different.

Q. Which is the constructor by Default available in java?

Default constructor.

Q.Can we overload the constructors?

Yes, the constructors can be overloaded by changing the number of arguments accepted by the constructor or by changing the data type of the parameters.

Q.Can you make a constructor final?

No: The constructor cannot be final, because it inherently it cannot be modified.

Q. Difference between constructor and methods?

Constructor	Method
1) Constructor is used to initialize the state of an object	A method is used to expose the behavior of an object.
2) A constructor must not have a return type.	Method must have return type.
3) The constructor is invoked implicitly.	The method is invoked explicitly.
4) The constructor name must be same as the class name.	The method name may or may not be same as class name.

Q.What is constructor chaining in Java?

Constructor chaining is the process of calling one constructor from another constructor with respect to current object.

Constructor chaining can be done in two ways:

- **Within same class**: It can be done using **this ()** keyword for constructors in same class.
- **From base class:** by using **super ()** keyword to call constructor from the base class.

EXCEPTION HANDLING IN JAVA

The **Exception Handling in Java** is one of the powerful mechanisms to handle the runtime errors so that the normal flow of the application can be maintained.

Q. What is Exception in Java?

It is a abnormal condition which is interrupt normal flow of program.

Types of Java Exceptions:

1. Checked Exception
2. Unchecked Exception

Q. Difference between Checked and Unchecked Exceptions

1) Checked Exception

The classes that directly inherit the Throwable class except RuntimeException and Error are known as checked exceptions.

For example: IOException, SQLException, etc. Checked exceptions are checked at compile-time.

2) Unchecked Exception

The classes that inherit the Runtime Exception are known as unchecked exceptions.

For example: ArithmeticException, NullPointerException, ArrayIndexOutOfBoundsException, etc.

Unchecked exceptions are not checked at compile-time, but they are checked at runtime.

3) Error

Error is irrecoverable. Some example of errors is OutOfMemoryError, VirtualMachineError, AssertionError etc.

Few blocks of exception.

1) **Try block-** it will include exceptional code in try block.
   A try block must be followed by catch blocks or finally block or both.

2) **Catch block-** it is used to catch the exception.
   Java *catch* block is used to handle the Exception. It must be used after the *try* block only. You can use multiple *catch* block with a single try.

3) **Throw** = it is a user defined exception.
   The "throw" keyword is used to throw an exception.

4) **Throws** = it a system generated exception.
   The "throws" keyword is used to declare exceptions.

5) **Finally** = Java *finally* block is always executed whether an exception is handled or not.
   Java *finally* block follows try or catch block.

Q.Differentiate throw and throws.

Throw	Throws
1) It is a user defined exception.	It a system generated exception.
2) Throw will declare inside method.	Throws will declare at the time method declaration.
3) In throw will handle only one exception at a time.	In throws will handle multiple exception at a time.

Q.What Happens When an Exception Is Thrown by the Main Method?

When an exception is thrown by main () method, Java Runtime terminates the program and prints the exception message and the stack trace in-system console.

Q.Differentiate final, finally, finalize?

Final	Finally	Finalize
1) Final is used to apply restriction 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 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 collated.
2) Final is a keyword.	Finally is a block.	Finalize is a method.

Q. Difference between Exception & Error?

Exception	Error
1) Exceptions include both checked as well as unchecked type.	All errors in java are unchecked type.
2) They are defined in java.lang.Exception package	They are defined in java.lang.Error package.
3) It can occur at run time ,compile time Both.	It can't be occur at compile time.
4) It is recoverable.	It is irrecoverable.
4) **Example:** NullPointerException, SqlException.	**Example:** OutOfMemoryError, IOError.

Q.What is the base class for Error and Exception?

The Throwable class is the base class for Error and Exception.

Q. How to create Custom Exception?

By using throw we will create custom exception. We have to create one class of exception and extend it to exception and use super (message).

Step to create exception:

1) Create class extends by exception class.

2) Create local variable message to store exception message locally in class object.

3) We passing a string argument to constructor of custom exception object.

4) toString () method to print custom exception message.

Q. What Is the OutOfMemoryError in Java?

The OutOfMemoryError in Java is a subclass of the java.lang.VirtualMachineError and it's thrown by the JVM when it runs out of heap memory.

Q.What Is a Chained Exception in Java?

Chained Exceptions allows relating one exception with another exception, i.e. one exception describes cause of another exception.

Q.How we can handle Exceptions?

The block of code in which an exception may occur is enclosed in a try block. This block is also called "protected" or "guarded" code. If an exception occurs, the catch block that matches the exception being thrown is executed. If not, all catch blocks are ignored. The finally block is always executed after the try block exits, whether an exception was thrown inside it or not.

Example:

```
try {
// ...
} catch (ExceptionType1 ex) {
// ...
} catch (ExceptionType2 ex) {
// ...
} finally {
// ...
}
```

Q.How to achieve garbage collection in java?

System.gc ().

Q.What is gc ()?

The gc () method is used to invoke the garbage collector for cleanup processing. This method is found in System and Runtime classes.

Q.What is Garbage Collector?

Garbage Collection is process by which the programs perform memory management automatically. In other words, it is a way to destroy the unused objects.

JAVA STRING

It is a collection of character. In Java, string is basically an object that represents sequence of char values. An array of characters works same as Java string.

For example:char[] ch={'j','a','v','a','t','p','o','i','n','t'};

String s=new String (ch);
Is same as: String s="javatpoint";

Java String class provides a lot of methods to perform operations on strings such as compare(), concat(), equals(), split(), length(), replace(), compareTo(), intern(), substring() etc.

The java.lang. String class
implements Serializable, Comparable *and* CharSequence interfaces.

Q. How many ways can we create the string object?

There are two ways to create String object:

1. By string literal
2. By new keyword

1) String Literal

Java String literal is created by using double quotes.

For Example: String s="welcome";

2) by new keyword
Java string is created by using a keyword “new”.

String s=**new** String ("Welcome"); //creates two objects and one reference variable.

Q.How many objects will be created in the following code?
String s = **new** String ("Welcome");

Two objects, one in string constant pool and other in non-pool (heap memory).

Q.What is String Pool?

String pool is the space reserved in the heap memory that can be used to store the strings.

Q.What is String Constant Pool?

It is a part of heap memory used to store literals (String Constant Pool).
In JVM There are Two Memory: Heap Memory and SCP

Whenever we are going to Create Object by using new keyword it will go to Heap Memory. Whenever we are going to define literal, Variables it will go to SCP.

Q.Difference between String and String Buffers, String Builder?

String	String Buffer	String Builder
1) String is immutable.	String Buffer is mutable.	String Builder is mutable.
2) String is thread safe.	String Buffer is thread safe i.e. it means two threads of string buffer simultaneously	String Builder is not thread safe i.e. it means two threads can call the methods of string builder simultaneously.
3)String performance is slow.	String Buffer performance is fast as compared to string.	String Builder performance is fast as compared to string buffer.
4) String is synchronized.	String Buffer is synchronized.	String Builder is not synchronized.
5) Introduce in jdk 1.2	Introduce in jdk 1.2	Introduce in jdk 1.5

Q. How we can make a class Immutable?

Declare the class as final so it can't be extended.
Make all fields private so that direct access is not allowed.

Q. Why String class is immutable?

String is immutable for Security, Synchronization, and concurrency Purpose.
String is immutable means it is constant and cannot be changed once it has been created.

Q.How to make string Mutable?

The mutable String in java can be created using String Buffer and String Builder classes. Difference between two is that String Buffer is Thread Safe & String Builder is not.

Q.How to Create Custom Immutable class?

We can declare class is “final” & whatever variable getter and setter all those things is make as “private” within the class scope only.

Q.What is Marker Interface? And how many Marker interface are there?

Marker Interface is a Empty Interface, means there is no method.
There is Three Marker Interface available in java:

- **Cloneable:** Cloneable marker interface is an indicator to the JVM that we can call the Object. clone () method. Java.lang package. Generates copy of object with different name.
- **Serializable:** Serializable is a marker interface means that it contains no methods. Java.io package.
- **Remote:** Remote interface is a marker interface that belongs to java.rmi package. It marks an object as remote that can be accessed from another machine (host).

Q.What is serialization?

Serialization in Java is a mechanism of writing the state of an object into a byte stream.

Serialization can achieve with the ObjectOutputStream.

It is used primarily in Hibernate, RMI, JPA, EJB and JMS technologies. It is mainly used to travel object's state on the network (which is known as marshaling). Serializable interface is used to perform serialization.

Q.What is Deserialization?

Deserialization in Java is a mechanism of writing the state of a byte stream into object.

Deserialization can achieve with the ObjectInputStream.

Q.How can you make a class serializable in Java?

A class can become serializable by implementing the Serializable interface.

Q.What is Filter Streams?

Filter Stream classes are used to add additional functionalities to the other stream classes.

Q.What is Enum in Java?

An Enum is a special “class” that represents a group of constants, it is used to declare fixed constant. All letters must be capital, Enum means constant.
For example: 12 months, 7 days etc.
It is available since JDK 1.5.
Enum improves type safety.
Enum can be easily used in switch.

COLLECTION IN JAVA

Q.What is Collection in Java

A Collection it is a group of object, which is used to store and manipulate the data in the form of objects.

It provides various classes such as ArrayList, Vector, Stack, and HashSet, etc. and interfaces such as List, Queue, Set, etc. for this purpose.

Q.What is the difference between Collection and Collections?

Collection	Collections
1) The Collection is an interface.	Collections are a class.
2) It doesn't has all static methods.	It has all static method.
3) The Collection interface provides the methods that can be used for data structure	Collections class provides the static methods which can be used for various operations on a collection.

Q.What is Array?

It is a similar data type of element.

Example: int arr [] = {7, 8, 5, 4}.

Q.What are the main differences between array and collection?

Array	Collection
1) It is a similar data type of element.	A Collection it is a group of object.
2) Arrays are always of fixed size.	Collection size can be changed dynamically as per need.
3) Arrays can only store homogeneous or similar type objects.	Collection can store heterogeneous objects.

Q. Explain various interfaces used in Collection?

1) List

2) Set

3) Map

1) List

It is an ordered collection of objects. It contains duplicate elements. It also allows random access of elements.

Q.What is the difference between Array List and Vector?

ArrayList	Vector
1) ArrayList is not synchronized.	Vector is synchronized.
2) ArrayList is not a legacy class.	Vector is a legacy class.
3) ArrayList is not? Thread-safe? As it is not synchronized.	Vector list is? Thread-safe? as its every method is synchronized

Q.Difference between Array & Array list?

Array	ArrayList
1) An array is a dynamically-created object.	The ArrayList is a class of Java Collections framework.
2) Array is static in size.	ArrayList is dynamic in size
3) An array is a fixed-length data structure.	ArrayList is a variable-length data structure.
4) An array can store both objects and primitives type	We cannot store primitive type in ArrayList. It automatically converts primitive type to object.

Q.Difference between Array list & Linked List?

Array list	Linked List
1) Array List is Data Definition Languages (DDL). We want to use Select/ Read / Retrieve / Fetch.	Linked List is Data Manipulation Language (DML). Manipulation means Insert / Update / Delete.
2) Array List internally uses a dynamic array to store the element.	Linked List internally uses a doubly linked list to store the elements.
3) Manipulation with Array List is slow, Because it internally uses an array.	Manipulation with Linked List is faster than Array List because it uses doubly linked list.
4) It is better for sharing and accessing data.	It is better for manipulation data.

2) Set

It is uniqueness. It cannot contain duplicate elements.

a) Hash Set – hash set is doesn't maintain the order (order is alpha numeric).

b) Tree Set- tree set is follows the order. Java tree set class maintain ascending order.

Q.What is the difference between List and Set?

List	Set
1) The List can contain duplicate elements.	Set includes unique items.
2) The List is an ordered collection which maintains the insertion order.	Set is an unordered collection which does not preserve the insertion order.
3) The List interface can allow n number of null values.	Set interface only allows a single null value.

Q.What is Java Iterator?

It is the interface which is used for iterate the collation of java objects components one by one, Also known as universal cursor. Used to traverse collection object.

3) Map

Key and value pair / no duplicate key, but allows multiple duplicate values.

a) Hash Map – Hash map is doesn't order. It allows only one null key and multiple null values.

K – It is the type of keys maintained by this map.

V – it is the type of mapped values.

B) Tree Map – Tree Map is follows order. Java Tree Map cannot have a null key but can have multiple null values. Java tree map class maintains ascending order.

Q.What is the difference between Set and Map?

Set	Map
1) Set contains values only.	Map contains key and values both.
2) Set contains unique values.	Map can contain unique Keys with duplicate values.
3) Set holds a single number of null values.	Map can include a single null key with n number of null values.

Q.What is the difference between HashSet and TreeSet?

HashSet	TreeSet
1) HashSet maintains no order.	TreeSet maintains ascending order.
2) HashSet implemented by hash table	TreeSet implemented by a Tree structure.
3) HashSet is backed by HashMap	TreeSet is backed by TreeMap.

Q.Difference between comparable and comparator

comparable	Comparator
1) Comparable is a present in java.lang. Package.	Comparator is a present in java.util. Package.
2) It is used to single sorting sequence.	It is used to multiple sorting sequences.
3) Comparable provides compareTo () method to sort elements.	Comparator provides compare() And equal () method to sort elements.
4) 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.

Q.What is the difference between HashMap and TreeMap?

HashMap	TreeMap
1) HashMap maintains no order.	TreeMap maintains ascending order.
2) HashMap is implemented by hash table	TreeMap is implemented by a Tree structure.
3) HashMap can be sorted by Key or value	TreeMap can be sorted by Key.
4) HashMap may contain a null key with multiple null values	TreeMap cannot hold a null key but can have multiple null values.

Q.What is the difference between HashMap and Hashtable?

HashMap	Hash table
1) HashMap is not synchronized.	Hash table is synchronized.
2) HashMap can contain one null key and multiple null values.	Hashtable cannot contain any null key or null values.
3) HashMap is not? Thread-safe,? so it is useful for non-threaded applications	Hashtable is thread-safe, and it can be shared between various threads.
4) HashMap inherits the AbstractMap class	Hashtable inherits the Dictionary class.

Q.Differences between HashMap and ConcurrentHashMap.

Hash Map	Concurrent Hash Map
1) HashMap is not synchronized.	ConcurrentHashMap is synchronized.
2) HashMap is not thread safe.	ConcurrentHashMap is thread safe.
3) HashMap allows key and value to be null.	ConcurrentHashMap does not allow null key/value. It will throw NullPointerException.
4) HashMap is faster.	ConcurrentHashMap is slower than HashMap

Q.What is the difference between HashSet and HashMap?

HashSet	HashMap
1) HashSet is doesn't maintain the order (order is alpha numeric).	HashMap maintains no order.
2) HashSet implements Set interface	HashMap implements the Map interface.
3) HashSet cannot have any duplicate value	HashMap can contain duplicate values with unique keys.
4) HashSet contains the only single number of null value.	HashMap can hold a single null key with n number of null values.

Q.Differences between Map and Flat Map.

Map	Flatmap
1) Map it is processes stream of values.	Flatmap it is processes of stream of stream of values.
2) Map it does only mapping.	Flatmap it performs mapping as well as flatting.
3) Map it is one to one mapping.	Flatmap it is one too many mapping.
4) Its mapper function produces single value for each input value.	Its mapper function produces multiple values for each input value.

Q.How Hash Map internally works?

Map :< K, V> pair / no duplicate key multiple null values. Classes- Hash Map and Tree Map.

There is one bucket of size 16 blocks in int index [0, 1, 2, 3……15]. Using hashing principle and internally its uses the linked list structure then there are available three methods.

1) Equals ()

2) Hash code ()

3) Collision

Hash code is integer value [0, 1, 2, 3…].

If two hash map object are equals then it must our hash map is same.

It two hash map is same collision method enter in program.

Once collision arrived then it internally override same block itself.

JAVA FILES

File handling is an important part of any application.

Java has several methods for creating, reading, updating, and deleting files.

Java File Handling

The File class from the java.io package, allows us to work with files.To use the File class, create an object of the class, and specify the filename or directory name:

1) Create a File

To create a file in Java, you can use the createNewFile () method. This method returns a Boolean value: true if the file was successfully created and false if the file already exists.

2) Write a File

In the following example, we use the File Writer class together with its write () method to write some text to the file we created in the example above. Note that

3) Read a File

We use the Scanner class to read the contents of the text file we created in the previous chapter:

4) Delete a File

To delete a file in Java, use the delete () method:

THREAD CONCEPT IN JAVA

Q.What is thread?

A **Thread** is a very light-weight part of process.

Q.How to create thread

There are two ways to create a thread:

1. By extending Thread class
2. By implementing Runnable interface.

1) By extending Thread class

If the class extends the Thread class, the thread can be run by creating an instance of the class and call its start () method:

Example:

```
public class Main extends Thread {

public static void main(String[] args) {

Main thread = new Main();

thread.start();

System.out.println("This code is outside of the thread");

}

public void run() {

System.out.println("This code is running in a thread");

}
```

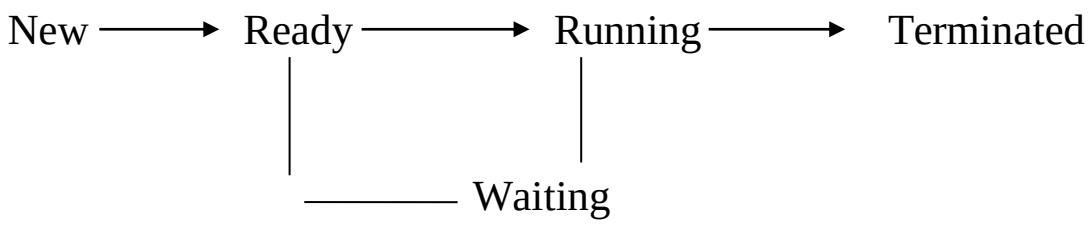
2) By implementing Runnable interface.

If the class implements the Runnable interface, the thread can be run by passing an instance of the class to a Thread object's constructor and then calling the thread's start() method:

Implement Example:

```
public class Main implements Runnable {

public static void main(String[] args) {

Main obj = new Main();

Thread thread = new Thread(obj);

thread.start();

System.out.println("This code is outside of the thread");

}

public void run() {

System.out.println("This code is running in a thread"); }}
```

Q.What is the states in the lifecycle of a Thread?



1. **New:** Thread class object is created using a new operator by using start () method.
2. **Ready:** This thread is ready to run after calling the start () method.
3. **Running:** This thread from the ready state and the thread are running.
4. **Waiting/Blocked:** This thread is waiting state and the thread waits for another perform task.
5. **Dead/Terminated:** A thread is in terminated or dead state when the run () method exits.

Q.Difference between Process and Thread

Process	Thread
1) A process is an instance of a program that is being executed or processed.	A **Thread** is a very light-weighted process.
2) Process does not require synchronization.	Thread may required synchronization.
3) Resource consumption is more in process.	Resource consumption Less in thread.
4) Process requires more time for creation.	Thread requires less time for creation.
5) The operating system takes more time to terminate a process.	Threads can be terminated in very little time.

Q.Can We Start a Thread Twice?

No. After starting a thread, it can never be started again. ... In such case, thread will run once but for second time, it will throw exception. IllegalThreadStateException.

Q.Can we Call run () method instead of start () method?

No, you can not directly call run method to start a thread. ... If you call run method directly, it won't create a new thread and it will be in same stack as main.

Q.What is Daemon Thread?

Daemon thread is a low priority thread that runs in background to perform tasks such as garbage collection.

Q.What is Object lock/Deadlock? How to avoid Deadlock?

Deadlock describes a situation where two or more threads are blocked forever, waiting for each other.

Avoid Nested Locks: A deadlock mainly happens when we give locks to multiple threads. Avoid giving a lock to multiple threads if we already have given to one.

.

MULTITHREADING IN JAVA

Q.What is multithreading?

Multithreading in Java is a more than in simultaneously Comes into execution state. **Or** Multithreading is a process of executing multiple threads simultaneously. Multithreading is used to obtain the multitasking. Its main advantages are:

- Threads share the same address space.
- The thread is lightweight.
- The cost of communication between the processes is low.

Q.What is Multitasking?

Multitasking is a process of executing multiple tasks simultaneously. We use multitasking to utilize the CPU. Multitasking can be achieved in two ways:

- Process-based Multitasking (Multiprocessing)
- Thread-based Multitasking (Multithreading)

1) Process-based Multitasking (Multiprocessing)

- Each process has an address in memory. In other words, each process allocates a separate memory area.
- A process is heavyweight.
- Cost of communication between the process is high.
- Switching from one process to another requires some time for saving and loading registers, memory maps, updating lists, etc.

2) Thread-based Multitasking (Multithreading)

- Threads share the same address space.
- A thread is lightweight.
- Cost of communication between the thread is low.

Q.What is thread synchronization?

More than one simultaneously comes into execution state, but only one thread execute at a time.

Q.What is inter-thread communication in java?

Already one thread is in Execution, then we are going to call another thread is called Inter-thread Communication.

Example: One Thread is executed and another thread we are going to execute.

It is implemented by following methods of **Object class**:

- wait()
- notify()
- notifyAll()

1) Wait () method
The wait () method is provided by the Object class in Java. This method is used for inter-thread communication in Java. The **wait ()** method causes the current thread to wait until another thread invokes the **notify ()** or **notify All ()** methods for that object.

2) notify () method
It wakes up one single thread that called wait () on the same object. It should be noted that calling notify () does not actually give up a lock on a resource.

Syntax: Public final void notify ()

3) Notify All () method

Wakes up all threads that are waiting on this object's monitor.

Syntax: Public final void notifyAll ()

Q.What is the difference between notify() and notifyAll()?

The notify () is used to unblock one waiting thread whereas notify All () method is used to unblock all the threads in waiting state.

Q.Difference between Wait () and Sleep () in Java

Wait()	Sleep()
The Wait () method is related to the Object class.	The Sleep () method is related to the Thread class.
It is not a static method.	It is a static method.
The method wait () is defined in the object class.	The method sleep () is defined in the thread class.
At the time of the Synchronization, the Wait() method releases obj.	At the time of the Synchronization, the Sleep () method doesn't release the obj, i.e., lock.
We can call the Wait () method only from the Synchronized context.	We can call the Sleep () method from outside the Synchronized context.
The Sleep() method has two overloaded methods, which are as follows: o sleep(long milliseconds, int nanoseconds) o sleep(long milliseconds)	The Sleep() method has three overloaded methods, which are as follows: o Wait() o wait(long timeout, int nanoseconds) o wait(long timeout)
The constructor of the Wait() method is defined in the following way: public final void Wait(long timeout)	The constructor of the Sleep () method in the following way: public static void Sleep (long millis) throws Interrupted_Execption

Q.How we can make Synchronized?

Synchronization is built around an internal entity known as the lock or monitor. Every object has an lock associated with it.

Q.What does join () method?

A join () is a final method of Thread class. Java.lang package Thread class provides the join () method which allows one thread to wait until another thread completes its execution.

Q.Difference between Sleep () & join() & yield() in Java

Yield()	Join()	Sleep()
1) This method allows the current thread to release its lock the object to other thread with same priority.	join () method which allows one thread to wait until another thread completes its execution	Sleep () method is used to sleep current thread to suspend its execution for a specified period and it does not release the lock of the object.
2) Yield () is overloaded.	Join () is overloaded.	Sleep () is overloaded.
3) Yield () is not final method.	Join () is final method.	Sleep () is not final method.
4) Yield () is static.	Join () is not static.	Sleep () is static.
5) Yield () doesn't it throw IntertuptedException.	Join () does it throw IntertuptedException.	Sleep () does it throw IntertuptedException.

Q.What is Thread Scheduler in java?

In Java, when we create the threads, they are supervised with the help of a Thread Scheduler, which is the part of JVM. Thread scheduler is only responsible for deciding which thread should be executed. Thread scheduler uses two mechanisms for scheduling the threads: Preemptive and Time Slicing.

Java thread scheduler also works for deciding the following for a thread:

- It selects the priority of the thread.
- It determines the waiting time for a thread
- It checks the Nature of thread

JAVA JDK 1.8 FEATURES

1) Lambda expressions

Lambda expression is used for to enable function programming.

No need to creating object.

If we are using multiple abstract methods then we used lambda expression.

Lambda expression is used to provide implementation of interface which has functional interface. So we lot of code.

Lambda expressions can be passed as a parameter to another method.

Lambda expressions can be standalone without belonging to any class.

2) Functional Interface

An Interface that contains only one abstract method is known as functional interface.

Functional interfaces are also known as Single Abstract Method Interfaces (SAM Interfaces).

Functional interface is used to enables functional programming.

It can have any number of default and static methods.

It can also declare methods of object class.

3) Static Method

A static method can access only static data member and can change the value of it.

A static method doesn't need to create instance of class.

If you apply static keyword with any method, it is known as static method.

4) Default Methods

Java provides a facility to create default methods inside the interface.

Methods which are defined inside the interface and tagged with default are known as default methods.

These methods are non-abstract methods.

`5) for each loop

Java provides a new method for each () to iterate the elements.

It is defined in Iterable and Stream interface.

It is a default method defined in the Iterable interface.

Collection classes which extends Iterable interface can use for each loop to iterate elements.

This method takes a single parameter which is a functional interface. So, you can pass lambda expression as an argument.

6) Date/Time API

There are two classis in:

a) Local Date: is an immutable class.

b) Local Date Time: is an immutable object.

7) Stream API

Stream API is used to process the collection of objects. Stream available in java.util package. Stream API is used in collection to filter the record and sorting the record.

Create class and declare default variable then generate (constructor/ string / getter setter).You can used stream API to filter, collect, print & convert from one data structure to other.

Q.What is the difference between Streams and Collections in Java 1.8?

Stream	Collections
1) Stream is used to complex data processing operations. **Example:** Filter, Map & Matching.	Collections are group of objects. **Example:** List, Set & Map.
2) We cannot add & remove element from stream.	We can add & remove element from collections.
3) Stream is lazy constructor.	Collections are eagerly constructor.
4) Stream are traversable only one.	Collections can be traversed multiple times.

8) Collectors

A collector is used for grouping & counting elements.

Collectors counting () method is used to count the number of element passed in stream as the parameter.

9) String Joiner

Java added a new final class String Joiner in java.util package.

It is used to construct a sequence of characters separated by a delimiter.

Now, you can create string by passing delimiters like comma (**,**), hyphen (-) etc.

You can also pass prefix and suffix to the char sequence.

10) Method reference

Method reference is used to refer method of functional interface.

It is compact and easy form of lambda expression.

There are following types of method references in java

1. Reference to a static method.
2. Reference to an instance method.
3. Reference to a constructor.

Each time when you are using lambda expression to just referring a method, you can replace your lambda expression with method reference.

11) Parallel Sorting

It is used to sorting the elements.

Array class which is used to sort array elements parallel.

New methods have added to java.util. Arrays package.

The methods are called parallel Sort () and are overloaded for all the primitive data types and Comparable objects.

12) Optional class

It is a public final class and used to deal with NullPointerException in Java application.

You must import java.util package to use this class.

It provides methods which are used to check the presence of value for particular variable.