OOPS: OOPs stands for Object-oriented programming. OOPs in Java organizes a program around the various objects and well-defined interfaces. The OOPs Concepts in Java are abstraction, encapsulation, inheritance, and polymorphism. These concepts aim to implement real-world entities in programs. Inheritance:  one object acquires all the properties and behaviours of a parent object. We can achieve Inheritance by using extends keyword. Single Level Inheritance: When a class inherits another class, it is known as a single inheritance. Multi-Level inheritance: Subclass inheriting the properties of superclass and that superclass inheriting the properties from another superclass is called as Multilevel inheritance. Hierarchical inheritance: When two or more classes inherits a single class, it is known as hierarchical inheritance. Polymorphism: we can perform a single action in different ways. Polymorphism is derived from 2 Greek words: poly and morphs. The word "poly" means many and "morphs" means forms. So, polymorphism means many forms.

1.Compile time Polymorphism: In java compile time polymorphism is achieved by using method overloading. compile time polymorphism is also called as Early Binding.

2.Runtime Polymorphism: Runtime polymorphism is achieved by using method overriding. Runtime Polymorphism is also known as Late Binding.

Method Overloading: If a class has multiple methods having same name but different in parameters, it is known as Method Overloading. (Compile time).

Method Overriding: If subclass (child class) has the same method as declared in the parent class, it is known as method overriding. (Run time).

ENCAPSULATION: Encapsulation refers to binding the data and the code that works on that together in a single unit. For example, a class. Encapsulation also allows data-hiding as the data specified in one class is hidden from other classes.

We can achieve encapsulation by

Declare the variables of a class as private. Provide public setter and getter methods.

Abstraction: Abstraction is a process of hiding the implementation details and showing only functionality to the user.

Abstract Method: It is an incomplete Method which has only method declaration and without method implementation.

Abstract method must be declared by using keyword ‘abstract’.

Abstract Class: It is an incomplete class which may have both abstract as we as concrete methods.

Interface: • It is an intermediate between the service and the consumer. • It is also called 100% abstract class.• Interface cannot have Constructors and instance variable. o It can be used to achieve loose coupling. o It is used to achieve abstraction. o A class can inherit an interface by using keyword implements There are 3 types of interface

1.Regular Interface: Regular Interface is an interface which contains more than one abstract method.

2.Marker Interface: Marker interface is empty interface which does not have any methods in it.

3.Functional Interface: it Is an interface which has only one abstract method in it.

Constructors: Constructors are special type of methods which have same name as the class name. Constructors cannot be declared as static or final.

Default Constructor: A constructor is called "Default Constructor" when it doesn't have any parameter.

parameterized constructor: A constructor which has a specific number of parameters is called a parameterized constructor.

Exception Handling: Exception Handling is a mechanism to handle runtime errors such as ClassNotFoundException, IOException, SQLException, RemoteException, etc.

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.

Unchecked Exception: The classes that inherit the RuntimeException 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 examples of errors are OutOfMemoryError, VirtualMachineError, AssertionError etc.

TRY: The "try" keyword is used to specify a block where we should place an exception code. It means we can't use try block alone. The try block must be followed by either catch or finally.

Catch: The "catch" block is used to handle the exception. It must be preceded by try block which means we can't use catch block alone. It can be followed by finally block later.

Finally: The "finally" block is used to execute the necessary code of the program. It is executed whether an exception is handled or not.

Throw: The "throw" keyword is used to throw an exception.

Throws: The "throws" keyword is used to declare exceptions. It specifies that there may occur an exception in the method. It doesn't throw an exception. It is always used with method signature.

Private: The access level of a private modifier is only within the class. It cannot be accessed from outside the class. Default: The access level of a default 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. Protected: The access level of a protected 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. Public: The access level of a public modifier is everywhere. It can be accessed from within the class, outside the class, within the package and outside the package

this: this keyword can be used to refer the current object in a method or constructor.

super: super keyword is a reference variable that is used to refer to parent class objects. It can be used to invoke the immediate parent class method.

Collection framework: The Collection framework represents a unified architecture for storing and manipulating a group of objects. It has: Interfaces and its classes and Algorithm.

Array List: Array List uses a dynamic array for storing the elements. It is like an array, but there is no size limit. We can add or remove elements anytime.

LinkedList: LinkedList uses a doubly linked list to store the elements.

doubly linked list, we can add or remove elements from both sides.

HashSet: HashSet class is used to create a collection that uses a hash table for storage. It inherits the AbstractSet class and implements Set interface. it stores the elements by using a mechanism called hashing. it contains unique elements only. it allows null value. it doesn't maintain the insertion order. Here, elements are inserted on the basis of their hashcode.

List: List provides the facility to maintain the ordered collection. It contains the index-based methods to insert, update, delete and search the elements. It can have the duplicate elements also. We can also store the null elements in the list.

A list can contain duplicate elements whereas Set contains unique elements only (Diff list & set).

Map: A map contains values on the basis of key, i.e. key and value pair. Each key and value pair is known as an entry. A Map contains unique keys. A Map is useful if you have to search, update or delete elements on the basis of a key. A Map doesn't allow duplicate keys, but you can have duplicate values. HashMap and LinkedHashMap allow null keys and values, but TreeMap doesn't allow any null key or value.

HashMap: HashMap class implements the Map interface which allows us to store key and value pair, where keys should be unique. If you try to insert the duplicate key, it will replace the element of the corresponding key. It is easy to perform operations using the key index like updation, deletion, etc.

Vector: Vector is like the dynamic array which can grow or shrink its size. Unlike array, we can store n-number of elements in it as there is no size limit. Array: array is an object which contains elements of a similar data type. the elements of an array are stored in a contiguous memory location. We can store only a fixed set of elements in a array. it is index-based, the first element of the array is stored at the 0th index, 2nd element is stored on 1st index and so on. String: string is an object that represents a sequence of characters. There are two ways to create String object: 1).By string literal(String s="welcome"; ) 2).By new keyword(String s=new String("Welcome");)). Java StringBuilder: Java StringBuilder class is used to create mutable (modifiable) String. Java StringBuffer: Java StringBuffer class is used to create mutable (modifiable) String objects. mutable String: A String that can be modified or changed is known as mutable String. StringBuffer and StringBuilder classes are used for creating mutable strings.

Immutable String: A String is an unavoidable type of variable while writing any application program. String references are used to store various attributes like username, password, etc. In Java, String objects are immutable. Immutable simply means unmodifiable or unchangeable. Once String object is created its data or state can't be changed but a new String object is created.

toString(): The toString() method returns the String representation of the object. By overriding the toString() method of the Object class, we can return values of the object.(Adv).

Static: static keyword in Java is used for memory management mainly. We can apply static keyword with variables, methods, blocks and nested classes.

static variable: If you declare any variable as static, it is known as a static variable. static variable can be used to refer to the common property of all objects (which is not unique for each object).

static Method: If you apply static keyword with any method, it is known as static method. A static method belongs to the class rather than the object of a class. it can be invoked without the need for creating an instance of a class. it can access static data member and can change the value of it.