***OOPS CONCEPT***

**INHERITANCE:**

**Q. What is inheritance?**

* Inheritance in Java is a mechanism in which one object acquires all the properties and behaviors of a parent object.
* When you inherit from an existing class, you can reuse methods and fields of the parent class. Moreover, you can add new methods and fields in your current class also.
* Inheritance represents the IS-A relationship which is also known as a parent-child relationship.
* The extends keyword indicates that you are making a new class that derives from an existing class. The meaning of "extends" is to increase the functionality.

**Q. Why do we use inheritance?**

* Code reusability.
* **Inheritance provides data hiding.** The base class can hide some data from the derived class by making it private.
* Method overriding cannot be achieved without inheritance. By method overriding, we can give a specific implementation of some basic method contained by the base class.
* To implement parent-child relationship.

**Q. Types of inheritance?**

**1. Single:** When a class inherits another class, it is known as a single inheritance.

**2. Multilevel:** A class is inherited by a class which again is inherited by another class, thereby forming a multilevel

chaining mechanism of inheritances.

**3. Hierarchical:** When two or more classes inherits a single class, it is known as hierarchical inheritance.

**4. Multiple Inheritance (Not supported for classes in Java):** A single class extends multiple classes, and therefore it

can be said to have the above inheritance’s combination. This can be achieved by making use of interfaces.

**5. Hybrid Inheritance: (Not supported for classes in Java):** It is essentially a combination of multilevel inheritance

and multiple inheritances where a single class is derived from more than one class, and the parent class is supposedly a

derived class and not a base class.

go to the page: https://www.javatpoint.com/inheritance-in-java

**Q. Why 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. Since compile-time errors are better than runtime errors, Java renders compile-time error if you inherit 2 classes. So, whether you have same method or different, there will be compile time error.

**Q. Can the subclass inherit static members?**

Yes, Static members are also inherited to sub classes in java.

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

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

**Q. How Inheritance can be implemented in java?**

Inheritance can be implemented in Java using two keywords, they are. I) extends 2) Implements

extends is used for developing inheritance between two classes and two interfaces.

Implements keyword is used to developed inheritance between interface and class.

**Q. Which of these keywords is used to refer to member of base class from a sub class?**

a) upper

b) super

c) this

d) None of these

**Answer: b**

**Q. Which inheritance are not supported bb Java?**

Multiple Inheritance is not supported by java.

**Q. How do you implement multiple inheritance in java?**

Using interfaces java can support multiple inheritance concept in java. In java cannot extend more than one classes, but a class can implement more than one interfaces.

**Q. Why we need to use Inheritance?**

It is used for code re-usability and for Method Overriding.

**Q. Can a class extend itself?** No, A class can't extend itself.

**Q. What happens if super class and sub class having same field name?**

Super class field will be hidden in the sub class. You can access hidden super class field in sub class using super keyword.

**Q. Which of the following is tightly bound? Inheritance or Composition?** Inheritance.

**Q. Does a class inherit the constructor of its super class?** Ans. No

**\*\*\*ABSTRACTION\*\*\***

**Q. What is abstraction?**

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

**Q. What are the ways to achieve Abstraction?** There are two ways to achieve abstraction in java:

i) Abstract class (0 to 100%)

ii) Interface (100%)

**\*\*\*Points to Remember:**

* An abstract class must be declared with an abstract keyword.
* It can have abstract and non-abstract methods.
* We can’t create object of Abstract Class, but we can create reference variable of Abstract Class. Because these classes are incomplete, they have abstract methods that have no body.
* It can have constructors and static methods also.
* It can have final methods which will force the subclass not to change the body of the method.
* An abstract class has no use until unless it is extended by some other class.

**Q. What is Abstract Method in Java?**

A method which is declared as abstract and does not have implementation/body is known as an abstract method.

**Q. Why we need an abstract class OR understanding the real scenario of Abstract class?**

Let’s say we have a class Animal that has a method sound () and the subclasses(see inheritance) of it like Dog, Lion, Horse, Cat etc. Since the animal sound differs from one animal to another, there is no point to implement this method in parent class. This is because every child class must override this method to give its own implementation details, like Lion class will say “Roar” in this method and Dog class will say “Woof”.

**Q. Can we make the abstract methods static in Java?**

In Java, if we make the abstract methods static, it will become the part of the class, and we can directly call it which is unnecessary. Calling an undefined method is completely useless therefore it is not allowed.

**Q. Can we declare the static variables and methods in an abstract class?**

Yes, we can declare static variables and methods in an abstract method. As we know that there is no requirement to make the object to access the static context, therefore, we can access the static context declared inside the abstract class by using the name of the abstract class. Consider the following example.

Rule: If there is an abstract method in a class, that class must be abstract.

Rule: If you are extending an abstract class that has an abstract method,

you must either provide the implementation of the method or make this class abstract.

**Q. Real life example of Abstraction?**

Mobile phone is one of example of abstraction. We only know about how to call and use mobile but we can't know about internal functionally of mobile phone.

<https://www.javatpoint.com/abstract-class-in-java>

**Q. Why can’t static methods be abstract in java?**

public abstract static void func();

* **Scenario 1:** When a method is described as abstract by using the abstract type modifier, it becomes responsibility of the subclass to implement it because they have no specified implementation in the super-class. Thus, a subclass must override them to provide method definition.

* **Scenario 2:** Now when a method is described as static, it makes it clear that this static method cannot be overridden by any subclass (It makes the static method hidden) as static members are compile-time elements and overriding them will make it runtime elements (Runtime Polymorphism).

**\*\*\*POLYMORPHISM\*\*\***

**Q. What is polymorphism?**

* The process of representing one form in multiple forms is known as Polymorphism.
* Many forms (one to many method---method overriding)
* Polymorphism means "many forms", and it occurs when we have many classes that are related to each other by inheritance.

**Q. Types of polymorphism:**

1) Compile Time/Static or Early Binding

2) Run Time/Dynamic/Late Binding

**Q. How to achieve polymorphism in Java?**

1. Static polymorphism in Java is achieved by method overloading and

2. Dynamic polymorphism in Java is achieved by method overriding

**Q. What is static polymorphism and dynamic polymorphism?**

1. Static polymorphism: Static polymorphism is polymorphism that occurs at compile time

2. Dynamic polymorphism: When child class object refers to a parent class reference variable is called dynamic

polymorphism. Car c = new BMW ();

**\*\*\*ENCAPSULATION\*\*\***

**Q. What is encapsulation?**

Encapsulation is a process of wrapping of data and methods in a single unit and protecting data by declaring them as a private. Private data will be hidden from other classes, and they can only be accessed through the method of their current classes, this is called data hiding.

**Q. How to do Encapsulation?**

1) Declare the variable as private---so that these variables cannot be accessed directly from outside of the class

2) Provide public getter and setter method to SET AND GET THE VALUES OF VARIABLES/FIELDS.

The get method returns the variable value, and the set method sets the value.

**Q. Benefits of Encapsulation:**

1) Provides data hiding. 2)Re-usability 3) Code can be modified without breaking the code.

**Q. What are the features of encapsulation?**

Combine the data of our application and its manipulation at one place.

**Q. What is difference between Encapsulation and Abstraction?**

Abstraction solves the problem at design level while encapsulation solves the problem at implementation level.

**\*\*\*Interface\*\*\***

**Q. What is interface?**

An interface is similar to class which is collection of public static final variables (constants) and abstract methods.

A class implements an interface, thereby inheriting the abstract methods of the interface.

**Q. Main features of Interface?**

=>Interface cannot be instantiated

=>An interface does not contain any constructors

=>All of the methods in an interface are abstract

=>All the data members of interface are implicitly public static final.

**Q. Can an Interface be final?**

No, because its implementation is provided by another class.

**Q. What is marker interface?**

An interface that have no data member and method is known as a marker interface. For example, Serializable, Cloneable etc.

**Q. What is difference between abstract class and interface?**

An abstract class can have method body (non-abstract methods) but Interface have only abstract methods.

**Q. Why we use interface?**

1) It is used to achieve 100% abstraction.

2) For achieve multiple inheritance in Java.

**Q. How interface is different form class?**

1. You cannot instantiate an interface.

2. It does not contain any constructors.

3. All methods in an interface are abstract.

4. Interface cannot contain instance fields. Interface only contains public static final variables.

5. Interface cannot extend by a class; it is implemented by a class.

**Q. When we use abstract and when Interface?**

* If we do not know about any things about implementation just, we have requirement specification then we should be go for Interface.
* If we are talking about implementation but not completely (partially implemented) then we should be go for abstract.

**Q. Why interface have no constructor?**

* Because constructor is used for eliminating the default values by user defined values, but in case of interface all the data members are public static final that means all are constant so no need to eliminate these values.
* Other reason because constructor is like a method, and it is concrete method and interface does not have concrete method it have only abstract methods that's why interface have no constructor.

**Q.** [**Are interfaces also inherited from Object class?**](https://www.w3schools.blog/are-interfaces-also-inherited-from-object-class)

**NO.**