**INHERITANCE:**

* **Inheritance provides data hiding. The base class can hide some data from the derived class by making it private.**

**Q. Can the subclass inherit static members?** Yes, Static members are also inherited to sub classes in java.

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

**Q. Why we need to use Inheritance?** It is used for code re-usability and for Method Overriding.

**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? No because its not a member but can be invoked by sc.**

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

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

* **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. 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.

**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).

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

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

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

**\*\*\*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 or How interface is different from class?**

**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. 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 has 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.**