**What is Inheritance?**

Inheritance is a concept where one class shares the structure and behaviour defined in another class. If inheritance applied on one class is called Single Inheritance, and if it depends on multiple classes, then it is called multiple Inheritance.

**Explain importance of inheritance in java?**

Reusability: The major advantage of inheritance is code reuse. We can avoid duplicating code by using inheritance. We can place all common state and behaviour in that class, by extending that class we can

Extendability : We can add new functionality to our application without touching the existing code. For example if we take Ms word we came across number of versions of msword such as word 2003,2007. Everytime they won’t write new code they reuse the existing code and some more features.

**Does Java support multiple inheritance?**

Java doesn't support multiple inheritance.

**Does a class inherit the constructors of its superclass?**

A class does not inherit constructors from any of its super classes.

**Explain about field hiding in java?**

If superclass and subclass have same fields subclass cannot override superclass fields. In this case subclass fields hides the super class fields. If we want to use super class variables in subclass we use super keyword to access super class variables.

**What are the different types of inheritances supported by Java?**

Java supports:

Single Inheritance

Multilevel Inheritance

Hierarchical Inheritance

Java doesn’t support Multiple Inheritance and Hybrid Inheritance.

**How to use Inheritance in Java?**

You can use Inheritance in Java by extending classes and implementing interfaces. Java provides two keywords extends and implements to achieve inheritance.  A class which is derived from another class is known as a subclass and an interface which is derived from another interface is called subinterface. A class which implements an interface is known as implementation.

**What is the difference between Inheritance and Encapsulation?**

Inheritance is an object oriented concept which creates a parent-child relationship. It is one of the ways to reuse the code written for parent class but it also forms the basis of Polymorphism. On the other hand, Encapsulation is an object oriented concept which is used to hide the internal details of a class e.g. HashMap encapsulate how to store elements and how to calculate hash values.

**What is the difference between Inheritance and Abstraction?**

Abstraction is an object oriented concept which is used to simply hide things by abstracting details. It helps in the designing system. On the other hand, Inheritance allows code reuse. You can reuse the functionality you have already coded by using Inheritance.

**Can we override static method in Java?**

No, you cannot override a static method in Java because it's resolved at compile time. In order for overriding to work, a method should be virtual and resolved at runtime because objects are only available at runtime.

**What is method hiding in Java?**

Since the static method cannot be overridden in Java, but if you declare the same static method in subclass then that would hide the method from the superclass. It means, if you call that method from subclass then the one in the subclass will be invoked but if you call the same method from superclass then the one in superclass will be invoked. This is known as method hiding in Java.

**Can we overload a static method in Java?**

Yes, you can overload a static method in Java. Overloading has nothing to do with runtime but the signature of each method must be different. In Java, to change the method signature, you must change either number of arguments, type of arguments or order of arguments.

**Can we override a private method in Java?**

No, you cannot override a private method in Java because the private method is not inherited by the subclass in Java, which is essential for overriding. In fact, a private method is not visible to anyone outside the class and, more importantly, a call to the private method is resolved at compile time by using Type information as opposed to runtime by using the actual object.

**Can a class implement more than one interface in Java?**

Yes, A class can implement more than one interface in Java e.g. A class can be both Comparable and Serializable at the same time. This is why the interface should be the best use for defining Type as described in Effective Java. This feature allows one class to play a polymorphic role in the program.

**Can a class extends more than one class in Java?**

No, a class can only extend just one more class in Java.  Though Every class also, by default extend the java.lang.Object class in Java.

**Can an interface extend more than one interface in Java?**

Yes, unlike classes, an interface can extend more than one interface in Java.

**How to call a method of a subclass, if you are holding an object of the subclass in a reference variable of type superclass?**

You can call a method of the subclass by first casting the object hold by reference variable of superclass into the subclass. Once you hold the object in subclass reference type, you can call methods from the subclass. See how type casting works in Java for more details.