**Using Assignment Operator to create copy of reference variable:**

In Java, there is no operator to create copy of an object. Unlike C++, in Java, if we use assignment operator then it will create a copy of reference variable and not the object.

Test ob1 = new Test();

// Creating a new reference variable ob2

         // pointing to same address as ob1

         Test ob2 = ob1;

         // Any change made in ob2 will be reflected

         // in ob1

         ob2.x = 100;

**Deep Copy vs Shallow Copy**

**Shallow copy:**any changes made in referenced objects in object X or Y will be reflected in other object. Object.clone() supports only **shallow copying** but we will **need to override** it if we need **deep cloning.**

**Deep Copy:**

 So any changes made in ‘c’ object fields by t3 ,will not be reflected in t1.

The **object cloning** is a way to create exact copy of 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**.

The **clone() method** is defined in the Object class.

* Object.clone() supports only **shallow copying** but we will **need to override** it if we need **deep cloning.**

In java, if a class needs to support cloning it has to do following things:

1. You must implement Cloneable interface.
2. You must override clone() method from Object class.

See this example for better understanding: <https://www.geeksforgeeks.org/clone-method-in-java-2/>