## Why We use Comparator or Comparable--->

MORE details are in collection docx. also

First, we need to understand why comparable & comparator comes into the picture. They give you the answer of comparison. Like, which one is greater which one is smaller. And the main reason for doing that is to Sort the collection of objects. We can easily compare the Integer or String naturally, but the complex or regular POJO object has multiple fields. In that case, we need to specify how we are determining which objects should come first or which one is greater.

In java String and Wrapper classes had implemented the Comparable interface. So if you store string or wrapper class objects then it will be used Comparable & get Sorted.

In custom class if dont use comparable or comparator then **classCastException** occurred.

Because when JVM tried to sort custom class, it starts the search the logic for comparing this custom class. Since you didn’t define any, it throws the Exception.

## Comparable

A Comparable interface in java.lang package is used to order the objects of the user-defined class. It contains only one method named compareTo(Object object)

A Comparable object is capable of comparing itself with another object.

It provides a single comparison only, meaning all comparisons are tied to one specific logic only. You can’t scale with multiple sorting methods depending on the requirement.

### When to Use Comparable :->

for natural sorting order.

### When to Use Comparator :->

The Comparator provides multiple sorting sequences. In other words, we can sort the collection based on multiple elements such as rollno, name.

The best-case scenario for Comparator is when we have different comparison logic for various fields and want to sort something based on requirements.

Comparator should be used as a utility to sort objects which Comparable should be provided by default.