**What does Comparer<T> do?**

Comparer<T> allows you to create custom comparison logic for objects of type T. This is useful when you want to sort objects in a particular order based on certain criteria

**How does it work?**

Let's say you have a class Box with properties like Height, Length, and Width. You want to sort a list of Box objects based on their dimensions. You can create a custom comparer by inheriting from Comparer<Box> and overriding the Compare method. In this method, you define your comparison logic, such as comparing boxes by length first, then height, then width.

public class BoxLengthFirst : Comparer<Box>

{

// Compares by Length, Height, and Width.

public override int Compare(Box x, Box y)

{

if (x.Length.CompareTo(y.Length) != 0)

{

return x.Length.CompareTo(y.Length);

}

else if (x.Height.CompareTo(y.Height) != 0)

{

return x.Height.CompareTo(y.Height);

}

else if (x.Width.CompareTo(y.Width) != 0)

{

return x.Width.CompareTo(y.Width);

}

else

{

return 0;

}

}

}

***Why use it?***

Using Comparer<T> gives you flexibility in sorting objects. You can define custom sorting orders based on your specific requirements. For example, you might want to sort a list of products by price, or a list of students by their grades.

Comparer<T> Class

Purpose: Allows custom comparison of objects of type T.

Namespace: System.Collections.Generic

Assembly: System.Collections.dll

Compare(T x, T y) Method

Signature: public abstract int Compare(T? x, T? y);

Parameters:

x: The first object to compare.

y: The second object to compare.

Returns:

An integer indicating the relative values of x and y:

Less than zero: x is less than y.

Zero: x equals y.

Greater than zero: x is greater than y.

Exceptions:

ArgumentException if type T does not implement IComparable<T> or IComparable.

Implementation:

Override this method in a derived class to provide custom comparison logic.

Allows customized sort order comparison for type T.

Notes:

Null can be compared with any reference type.

Null reference is considered less than any non-null reference.