**Comparable<T>**

public int compareTo(T o);

In order to implement the Comparable<T> interface a class must have a compareTo method.

compareTo returns 0 when the invoking object equals the parameter object with respect to the natural ordering of the class.

compareTo returns a negative int when the invoking object is before the parameter object with respect to the natural ordering of the class.

compreTo returns a positive int when the invoking object is after the parameter object with respect to the natural ordering of the class.

invokingObject.compareTo(parameterObject);

public int compareTo(T o);

**Comparator<T>**

Int compare(T o1, T o2);

In order to implement the Comparator<T> interface a class must have a compare method.

compare returns 0 when the first parameter is the same as the second parameter with respect to the ordering imposed by the comparator.

compare returns a negative int when the first parameter is before the second parameter with respect to the ordering imposed by the comparator.

compare returns a positive int when the first parameters is after the second parameter with respect to the ordering imposed by the comparator.

comparatorObject.compare(firstParameter, secondParameter);

**int** compareFamily = n2.myFamily.compareTo(n1.myFamily);

**if**(compareFamily != 0) {

**return** compareFamily;

}//family names were different

**int** compareFirst = n2.myFirst.compareTo(n1.myFirst);

**if**(compareFirst != 0) {

**return** compareFirst;

}//family names were different

**return** n2.myMiddle.compareTo(n1.myMiddle);