**Using Java Sorting Methods**

The Java API provides a class Arrays with several overloaded sort methods for different array types

The Collections class provides similar sorting methods for Lists

Sorting methods for arrays of primitive types are based on the quicksort algorithm (quadratic for worst case)

Sorting methods for arrays of objects and Lists are based on the merge sort algorithm (nlogn for the worst case)



Text

Description automatically generated with medium confidence



Declaring a Generic Method

FORM:

methodModifiers <genericParameters> returnType methodName(methodParameters)

EXAMPLE:

public static <T extends Comparable<T>> int binarySearch(T[] items,   
 T target)

MEANING:

To declare a generic method, list the genericParameters inside the symbol pair < > and between the   
 methodModifiers (e.g., public static) and the return type. The genericParameters can then be used in   
 the specification of the methodParameters

Sample declarations:

--> public static <T> void sort(T[] items, Comparator<? super T> comp)

T represents the generic parameter for the sort method. T should also appear in the method parameter list.

The second method parameter means that comp must be an object that implements the Comparator interface for type T or for a superclass of type T.

For example, you can define a class that implements Comparator<Number> and use it to sort an array of Integer objects or an array of Double objects since Number is superclass of Integer or Double



--> public static <T extends Comparable<T>> void sort(List<T> list)

Generic parameter T must implement the interface Comparable<T>

Text, letter

Description automatically generated

Text, letter

Description automatically generated