HeapSort Implementation

1. Write a class called MaxHeap that implements a max-heap by wrapping an ArrayList. We will be storing the heap as an array instead of using linked nodes. Use proper generic notation so that your heap can store any Comparable elements.

2. Using the algorithms shown in class, implement the following instance methods for the MaxHeap class:
    - **isEmpty** – returns true if the heap is empty, false otherwise
    - **add** – adds an element to the heap (Remember: add the element to the end and swap with its parent until it bubbles up to the right place.)
    - **removeLargest** – removes the largest element from the heap (Remember: move the LAST element to the top and then swap with smallest child until it trickles down to the right place.)

3. Write JUnit tests to ensure that your heap works correctly.

4. Implement HeapSort by writing a static method with the following prototype:

    public static <M **extends** Comparable<M>> **void** printSortedList(Collection<M> x)

    The idea is to print out the elements of the collection, x, in sorted order by first inserting them all into a max-heap and then removing them one-by-one and printing.

    Note that this is a use of generics that was not covered in class. The scope of the parameterized type, M, is just for this one method. Be sure that the variable used here (M) is distinct from the variable used when declaring the MaxHeap class.

5. Write a JUnit test to see if your printSortedlist method is working.