Lab: Priority Queues
CSC 207, “Algorithms and Object-Oriented Design”
Department of Computer Science
Grinnell College
October 19, 2018

Warmup Exercises

1. If one inserts the integers from 99 down to and including 0 into a priority queue that uses the “natural order” of the integers as its criterion for prioritization, but extracts an element after every other insertion, what integers will be extracted, and in what order?

2. At the end of the preceding exercise, there will still be fifty integers left in the priority queue. If one extracts them, one after another, what integers will appear, and in what order?

Sorting with a Priority Queue

The presence of a generic PriorityQueue class in the java.util package suggests a different sorting method that looks very easy to write: Traverse the array to be sorted and add each of its elements to a priority queue. Then traverse the array a second time, from left to right, filling each position with an element extracted from the priority queue. The elements will emerge in sorted order because the deleteMin operation always returns the smallest element that is still in the priority queue.

3. Write and test a PriorityQueueSorter class that implements this sorting algorithm as a public, static method.

4. Use the PriorityQueueSorter to sort the words and punctuation marks from Eight Cousins, and confirm that the method yielded the correct result.

5. For an array of N elements, this algorithm always calls the method that adds an element to a priority queue exactly N times, and calls the method that extracts an element from a priority queue exactly N times. Yet it turns out that the running-time function of this sort is not \( O(N) \), but \( O(N \log N) \). Explain this seeming contradiction.

Priority Queues and Comparators

6. Define a class called CountyComparator that implements the Comparator interface, providing it with a compare method that takes two County objects (from the lab on population densities of Iowa counties) and returns a negative integer if the population density of the first County is greater than the population density of the second County, a positive integer if the population density of the first County is less than the population density of the second County, and 0 if the populating densities are equal.

7. Suggest and implement a way, using a priority queue, to find the ten Iowa counties with the highest population densities without completely sorting the data. (Hint: One of the constructors for the PriorityQueue class takes as an argument an instance of any class that implements the Comparator interface, which the priority queue will then use as its criterion for prioritizing elements.)