The goal of this assignment is to have fun with maps and queues! The user will input text and your program will output the number of occurrences of each character in descending order. This will be done with you writing very little code due to your (my) appropriate choices of data structures!

1. [20] Create a new Eclipse project named Assignment6. Get input from the user using a method of your choice (i.e., command line or GUI).

2. [20] Use a java.util.HashMap to track the number of occurrences of each character (i.e., including punctuation, whitespace, etc characters). In class, we’ll discuss the benefits of using a map for this operation. The map key should be the distinct character and the map value should be the number of occurrences.

3. [20] Create a class which implements java.lang.Comparable to store the number of occurrences associated with a specific character (i.e., your class should maintain a private char and a private int instance variable. Implementing Comparable will allow you to use these instances for sorting purposes, and you’ll want to sort based on the number of character occurrences, so set up the compareTo method such that comparisons will be based on the number of character occurrences. Add a toString method to your class for pretty printing.

4. [20] Create a java.util.PriorityQueue and add an instance of your above Comparable class (from part 3) to the queue for each entry in the above map (from part 2). The PriorityQueue will call compareTo() internally such that calling the poll() method will always return the highest-priority item.

5. [20] Loop through and print each entry in the queue. Note that I used the PriorityQueue poll method because the iterator did not seem to preserve the priority of the queue (which is strange, but noted in the documentation).

6. [+20] (Extra credit) java.util.PriorityQueue is backed by an array (i.e., data is internally stored in an array), so implement your own priority queue class which also implements java.util.Queue<E> but is backed by your own sequential-access list implementation (you may extend java.util.LinkedList, if you’d like). You may implement this class however you’d like, provided that your poll() method returns the highest-priority item. Use this implementation in parts 4 and 5 rather than java.util.PriorityQueue!

Zip the Assignment6 folder in your Eclipse workspace directory and upload the .zip file to Blackboard (see Assignment 6 assignment in the Course Documents area).