COMP 203 Lab 9

Linked List Implementation of Priority Queue

Implement Priority Queue abstract data structure using Node data structure. You can construct Priority Queue data structure similar to Singly Linked List but the rules of Priority Queue is different as you know. You will use the sorted list implementation. It means that you will add the nodes in PQ according to their priority.

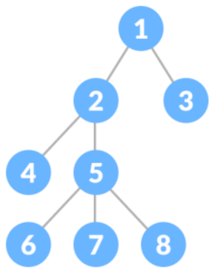
Example: your pq will be as follows: head-> (A,1)->(C,2) -> (F,3) -> (D,4) ->null. Here the head node has (A,1) and has the highest priority.

Implement the following classes and methods in java: **(100pt)**

1. Implement Node<char, int> class for Priority Queue with its constructor with the parameter char data and int priority. **(5pt)**
   Hint: Each node has a char data and int priority and Node next variables
2. Implement PQ class with the class variables and a constructor with char data variable. You should have *int size* variable to keep track of size of the priority queue. **(10pt)**
3. Insert(char addedData, int priority) that inserts the node with the given value *addedData and* given *priority* into the priority queue. **(20pt)**
4. RemoveMin() that removes the element in the priority queue with the highest priority and the returns the removed node's data. **(20pt)**
5. Min() returns the value of the node having the highest priority. **(10pt)**
6. Size() that returns the number of nodes in the tree. **(5pt)**
7. PrintQueue that prints the data in the priority queue. **(10pt)**
8. Test your methods in the main by creating the following priority queue. (Creating the given priority queue:**10 pt**, testing 5 methods: **5x2=10 pt**)

Head-> (A,1)-> (G,2)->(U,3)->(C,4)->(E,5)->NULL

Include comments of your code for each method and class.
Submit PQ.java to Canvas as a single java file. No other file types will be accepted.


1
2
3
4
5
6
7
8