**Lab4 - PriorityQueue**

Create a ADT to read the Speech.txt file in this week's document folder.  Store each character in the Doubly Linked PriorityQueue and increment it's frequency every time it's encountered in the file.  See the UpdateFrequency Algorithm listed below for guidelines on writing your algorithm.  When finished processing the file, print the PriorityQueue and you should see the frequency in order from highest to lowest.

Algorithm Greater<T>(first,second)

return first > second

class node<T>

T data;

node<T> next

node<T> prev

node(item)

data = item

// Modify a singly linked list PriorityQueue into

// a doubly linked list priority queue

class PriorityQueue<T>

node<T> front

node<T> rear // add this

PriorityQueue()

front = NULL

bool isempty()

return front == NULL

void enqueue(item)

node<T> tmp = new node<T>(item)

if front == NULL OR greater(item,front)

tmp.next = front

front = tmp

else

node<T> rover = rear

while rover.next != NULL AND greater(rover, item)

rover = rover.next

tmp.next = rover.next

rover.next = tmp

void dequeue()

front = front.next

void print()

node<T> rover = front

while rover != rear

output(rover->data)

rover = rover.next

// inserts a node into a PriorityQueue

// based on frequency of a character

Algorithm UpdateFrequency<T>(character)

node<T> nuChar = node<T>(character)

Search for character in PriorityQueue

If character found

increment frequency counter

Else

set frequency counter to

IF greater(nuChar,front)

InsertFront(nuChar)

Else

Node<T> rover = rear

While Greater(rover,nuChar) AND rover.prev != front

rover = rover.prev

InsertNode(nuChar,rover)