CSE 222 Programming Assignment 3 – Winter 2019

Assignment 3 deals with stacks and queues. You must implement these structures using linked lists, as described in class. Be sure to adhere to the concepts of abstract data types as much as possible.

Program Summary

You're to create a program that implements both a stack and a queue, and processes commands allowing the user to manipulate these structures.

Since your program will have *two distinct structures*, it will operate in two possible *modes*: stack mode; and queue mode. In stack mode you're working with the stack, and in queue mode you're working with the queue. The program should start in stack mode.

Legal commands are as follows:

- s – print the current contents of the stack on one line, separated by spaces, with the **top of the stack** at the left (don't show the sentinel); and then set the current mode to stack mode (so if you're already in stack mode, "s" just prints the stack).
- q – print the current contents of the queue on one line, separated by spaces, with the **head of the queue** on the left (don't show the sentinel); and then set the current mode to queue mode (so if you're already in queue mode, "q" just prints the queue).
- Any integer - push onto the stack (if you're currently in stack mode); or insert at the tail of the queue (if you're currently in queue mode)
- p - either pop the top of the stack (stack mode) or remove the item at the head of the queue (queue mode) and print the item popped or removed (not the entire stack or queue).
- Q - exit the program (**be sure to free all memory!)**

anything else should give a help message.

Details

Implement the stack using a linked list; implement the queue using a second linked list (so you'll have two sentinel nodes, each pointing to a different linked list). You should use these lists as described in class (summarized below):

For the stack, push new elements by adding a node immediately after the sentinel, and pop from the same location. For the queue, the head is the node immediately after the sentinel, and the tail is at the far end of the list.

Notes

You shouldn't run out of memory in the tests I'll conduct (assuming your code is well-written). If the user tries to remove an item from an empty structure (stack underflow, or removing from an empty queue), **give an appropriate error message** but continue accepting commands, and **do not corrupt your structures!** Moving between stack mode and queue mode **should not** affect the contents of these data structures.

Test your code thoroughly, include good comments, include a makefile to create your executable program (which must be named "pa3") and submit a .tar file via Canvas by the deadline.