CS 261
DATA STRUCTURES

Oregon State University

CS261 : Programming Assignment 3

This assignment is comprised of 3 parts:

Part 1: Linked List Deque and Bag implementation

First, complete the linked list implementation of the deque (as in Worksheet 19) and bag ADTs (Worksheet 22 due in week 4). To do this, implement all functions with the `// FIXME...` comments in `linkedList.c` .

Grading (30 pts):

- `init` -- 2 pts
- `addLinkBefore` -- 3 pts
- `removeLink` -- 3 pts
- `linkedListAddFront` -- 2 pts
- `linkedListAddBack` -- 2 pts
- `linkedListFront` -- 2 pts
- `linkedListBack` -- 2 pts
- `linkedListRemoveFront` -- 2 pts
- `linkedListRemoveBack` -- 2 pts
- `linkedListIsEmpty` -- 2 pts
- `linkedListPrint` -- 2 pts
- `linkedListContains` -- 3 pts
- `linkedListRemove` -- 3 pts

Part 2: Circularly Linked List Deque implementation

In this part, you will implement the Deque ADT with a Circularly-Doubly-Linked List with a Sentinel. As you know, the sentinel is a special link, does not contain a meaningful value, and should not be removed. Using a sentinel makes some linked list operations easier and cleaner in implementation. This list is circular, meaning the end points back to the beginning, thus one sentinel suffices. Implement all functions with the `// FIXME...` comments in `circularList.c`.

Grading (30 pts):

- `init` -- 1 pts
- `createLink` -- 1 pts
- `addLinkAfter` -- 2 pts
- `removeLink` -- 2 pts
- `circularListAddBack` -- 2 pts
- `circularListAddFront` -- 2 pts
- `circularListFront` -- 2 pts
- `circularListBack` -- 2 pts
- `circularListRemoveFront` -- 2 pts
- `circularListRemoveBack` -- 2 pts
- `circularListDestroy` -- 2 pts
- `circularListIsEmpty` -- 2 pts
- `circularListPrint` -- 2 pts
- `circularListReverse` -- 6 pts (Importantly, you must perform the list reversal in place and may not allocate any new memory in this function)

Part 3: Linked List Stack implementation using two Queues

In this part, you will use two instances of Queue ADT to implement a Stack ADT. You will use only one C file (stack_from_queue.c) containing all the functions to design the entire interface. Also, make sure your stack-from-queues implementation does not have any memory leaks!

Grading (40 pts) :

- `listQueueInit` -- 2 pts
- `listQueueCreate` -- 2 pts
- `listQueueIsEmpty` -- 3 pts
- `listQueueAddBack` -- 5 pts
- `listQueueRemoveFront` -- 5 pts
- `listQueueFront` -- 3 pts
- `linkedListStackInit` -- 2 pts
- `linkedListStackCreate` -- 2 pts
- `linkedListStackIsEmpty` -- 3 pts
- `linkedListStackPush` -- 5 pts
- `linkedListStackPop` -- 5 pts
- `linkedListStackTop` -- 3 pts

What to turn in:

You will turn in the following three(3) files to both TEACH and Canvas -

- `linkedList.c` -- your linked list deque and bag implementation.
- `circularList.c` -- your circularly linked list deque implementation.
- `stack_from_queue.c` -- your linked list stack implementation.

Use the provided makefiles/header files to compile your code on flip. Also, you must test your compiling on flip.engr.oregonstate.edu. Zero credit if we cannot compile your programs. Finally, please don't submit in zipped format to TEACH.