CS 302 – Assignment #4

Purpose: Learn to about stacks and queues using linked lists
Due: Tuesday (9/26)
Points: 125 Part A → 75 pts, Part B → 50 pts

Assignment

Part A:

A palindrome is a word, phrase, number, or other sequence of characters which reads the same backward as forward, such as madam or racecar. Sentence-length palindromes may be written when allowances are made for adjustments to capital letters, punctuation, and word dividers, such as "A man, a plan, a canal, Panama!", "Was it a car or a cat I saw?" or "No 'x' in Nixon", or “Aibolhphobia” (fear of palindromes).

Implement a series of C++ classes to determine of a string is palindromic using a stack and a queue as follows:

The classes are as follows:

- ***linkedStack*** to implement the stack using a linked list
- ***linkedQueue*** to implement the queue using a linked list

The final output will be the string and a “is a palindrome” or “is not a palindrome” message. A main will be provided that can be used to test the classes.

Part B:

When completed, create and submit a write-up (PDF format) not to exceed ~500 words including the following:

- Name, Assignment, Section
- Summary of the implemented linked stack and linked queue data structures.
- Big-O for each of the stack operations (push, pop).
- Big-O for each of the queue operations (enqueue, dequeue).
- Comparison of linked stack data structure to the linked queue data structure.

It should be noted that there are many implementation variations on these data structures and algorithms. These are the data structures and algorithms that must be implemented. Copying code from the net will result in a zero for the assignment and referral to the Office of Student Conduct.

Submission:

- Part A → Submit a compressed zip file of the program source files, header files, and makefile via the on-line submission by 23:50.
- Part B → A copy of the write-up. Must use PDF format.

All necessary files must be included in the ZIP file. The grader will download, uncompress, and type **make** (so you must have a valid, working *makefile*).

Basic Queue and Stack Palindrome Algorithm

The basic algorithm is to read a string, one character at a time and simultaneously place the character in a stack and a queue. When done, check the queue and stack, and if any mis-matches occur, it is not a palindrome. The algorithm should ignore all non-alpha and non-digit characters. As such, spaces and punctuation are not considered as part of the palindrome decision. The *isalpha()*, *isdigit()*, *tolower()*, and/or *toupper()* functions may be useful. If all characters are the same, it is a palindrome. While there are other ways to perform this function, you ***must*** use this approach.

Main

Write a main the will use the linked stack and linked queue objects to check if a string is a palindrome. The string should be read from the command line. If the string is not provided or too many arguments are provided, an appropriate error message should be displayed. Refer to the example executions for formating examples.

Class Descriptions

- Linked List Stack Class
  The linked stack class will implement a stack with a linked list including the specified functions. We will use the following node structure definition.

```
template <class myType>
struct nodeType {
      myType item;
      nodeType<myType> *link;
};
```

linkedStack<myType>
-nodeType<myType> *stackTop
-itemCount: int
+linkedStack()
+~linkedStack()
+isEmpty() const: bool
+push(const myType &newItem): void
+pop(): myType

Function Descriptions

- The *linkedStack()* constructor should initialize the stack to an empty state (*stackTop*=NULL, etc.). *Note*, to initialize a variable in a template, use {}.
- The *~linkedStack()* destructor should delete the stack (including releasing the allocated memory).
- The *isEmptyStack()* function should determine whether the stack is empty, returning *true* if the stack is empty and *false* if not.
- The *push()* function will add the passed item to the top of the stack.
- The *pop()* function will remove the top item from the stack and return return it. If the stack is empty, nothing should happen and it should return 0.

- Linked List Queue Class

  The linked queue class will implement a queue with a linked list including the specified functions. We will use the following node structure definition.

```
template <class myType>
struct queueNodeType
{
    myType info;
    queueNodeType<myType> *link;
};
```

linkedQueue<myType>
-queueNode<myType> *queueFront
-queueNode<myType> *queueRear
-count: int
+linkedQueue()
+~linkedQueue()
+isEmptyQueue() const: bool
+enqueue(const myType& newItem): void
+dequeue(): myType
+printQueue(): void

Function Descriptions - Queue

- The *linkedQueueType()* constructor should initialize the queue to an empty state (*queueFront* = NULL and *queueRear* = NULL). *Note*, to initialize a variable in a template, use {}.
- The *~linkedQueueType()* destructor should delete the queue (including releasing all the allocated memory).
- The *isEmptyQueue()* function should determine whether the queue is empty, returning *true* if the queue is empty and *false* if not.
- The *enqueue()* function will add the passed item to the back of the queue.
- The *dequeue()* function will remove and return the front item from the queue. If the queue is empty, nothing should happen.
- The *printQueue()* function should print the current elements of queue.

Refer to the example executions for output formatting. Make sure your program includes the appropriate documentation. See Program Evaluation Criteria for CS 302 for additional information. ***Note, points will be deducted for especially poor style or inefficient coding.***

Example Execution – Testing Program:

Below are example program executions.

```
ed-vm%
ed-vm% ./main "Eva, can I see bees in a cave?"
The string: 'Eva, can I see bees in a cave?' is a palindrome.
ed-vm%
ed-vm% ./main
Error, missing command line argument.
ed-vm%
ed-vm% ./main hello world
Error, too many command line arguments.
ed-vm%
ed-vm% ./main "123 321"
The string: '123 321' is a palindrome.
ed-vm%
ed-vm%
ed-vm% ./main "hello world"
The string: 'hello world' is not a palindrome.
ed-vm%
ed-vm%
```