Algorithms: Assignment 1

Maria Molloy

September 17, 2020

1. **Stack**

   The stack code is all in the class Stack.java. There is a stack constructor on line 8 that creates a top of the stack. Stack has three methods:

   - isEmpty(): isEmpty() is found on line 12. It checks if the stack is empty by seeing if the top is null or not.
   - push(String element): push is found on line 23. It takes one parameter, a string of the data you want to push onto the stack. It pushes a node by setting the new node's pointer to the current head of the list. Then, the head is set to the new node.
   - pop(): pop() is found on line 37. Pop removes the most recently added item by making sure the stack is not empty and then saving the current top node as a new node, setting the new node's pointer as the new top, and returning the new node.

2. **Queue**

   The queue code is all found in the class Queue.java. On line 10, there is a queue constructor that makes a head and a tail for each queue. Queue has three methods:

   - isEmpty(): isEmpty() is found on line 15 and checks if the queue is empty by seeing if the head is null or not.
   - enqueue(String l): enqueue is found on line 25. It takes one parameter, a string of data for the node you want to add to the queue. It adds a node to the queue by setting the old tail of the queue to point to the new node. Then it sets the tail to the new node, unless the list is empty, in which case it sets the new node as the head and the tail.
   - dequeue(): Dequeue is found on line 42. It returns the node we are removing from the queue. It uses the same logic as the Stack method pop(). Basically, it makes sure the queue is not empty and then creates a new node that is equal to the queue's head. Then it sets the head of the list to the previous head's pointer.

3. **Main**

   The main class is (obviously) Main.java. First, the magicitems.txt file is scanned into a linked list line by line (where each line is a node) using the built in Java scanner. On line 26, each line is added to a linked list of magic items. Then, on line 37, the list is traversed. The data of each node checked to be a palindrome by using the isPalindrome() Node method, found in Node.java on line 38. The isPalindrome() method works by added the data of each node letter by letter to both a stack and a list (lines 66-67). Then, on line 74, the stack is popped and the queue is dequeued letter by letter. For each letter, it checks if they are the same and if they are, increments a palindrome counter (line 75). If the palindrome counter is equal to the length of the word, it is a palindrome and returns true. Otherwise it returns false. Back in Main.java, once we know a node is a palindrome, the data from that node is added to a new linked list of just palindromes (line 39). Then, on line 47, the list of palindromes is printed.