David Fiel

Professor Dobor

CIS 2168

16 October 2020

Lab 5 Code Explanation

Question One

The only data structure that my solution used was a Queue. I enqueued every integer from 1 to the given input n, and then started a loop that runs until there is one object left in the queue. This loop re-enqueues the first m-1 objects in the queue, and then prints the value of the dequeued m object. This loop will continue until there is only one object remaining, which is the leader. This question could also be done with a looping singly-linked list, where the next pointer of the last object is to the first object.

Question Two

My solution to the bracket matching problem included multiple data types. I created a helper class Bracket to hold the position and character of each opening bracket, two lists to hold the opening and closing brackets, and a stack to hold all the Bracket objects. The Bracket class was simple, with only two fields bracket and position, holding a character and integer respectively. I used two lists to hold the valid opening and closing brackets because of the contains() method, it made checking if the current character is an open or closing bracket very simple. This could have easily been replaced with simple logic statements. The stack held all the opening bracket objects,

which were pushed as we traversed the list. When a closing bracket was encountered, a bracket object was popped from the stack and checked to see if the bracket matched.