Hugh A. Miles II

09/21/16

COP3530

Circular Linked List

- Summary
    - My Circular Linked List a basic Linked List Class, but instead of the last Node (tail) pointing to nothing its points to the head. This is a templated so the Nodes can hold any type based upon the type templated at its instantiation. Each node has its own index, a bool value indicating whether it's the head or not, and the address of the next Node in the list.
- Structure
    - Node<type>
        - Constructor (type initValue, int strIndex)
            - In the constructor you need to indicate the value to be set, and where the Node is being placed with the strIndex.
        - Int index
            - Location in the list
        - Type value
            - The data value stored in side the node.
        - Node<type>* nextNode
            - Address for the next Node in the list
        - Bool isHead

                - Value that indicated whether you are at the front of the list.
        - LinkedList
            - Head<type>
                - Pointer to start of the list (index = 0)
            - Int curLength
                - Keeps track of the number of Nodes in the current list.
- Methods
    - Insert (type value, int index)
        - Allow users to insert element at a certain spot in the current list that is less than the this.length + 1.
    - DeleteNode(int index)
        - Allows user to delete Node at any point in the current list that is less than this.length + 1
    - Print
        - Prints entire list sequentially.
    - PrintAtIndex(int index)
        - Prints element at index given in the parameter
    - Helper Methods
        - Append(type value)
        - ResetIndexs
            - Since each node contains its own index. This method resets the node.index for each node in the list after a delete or insert.

- Test Cases

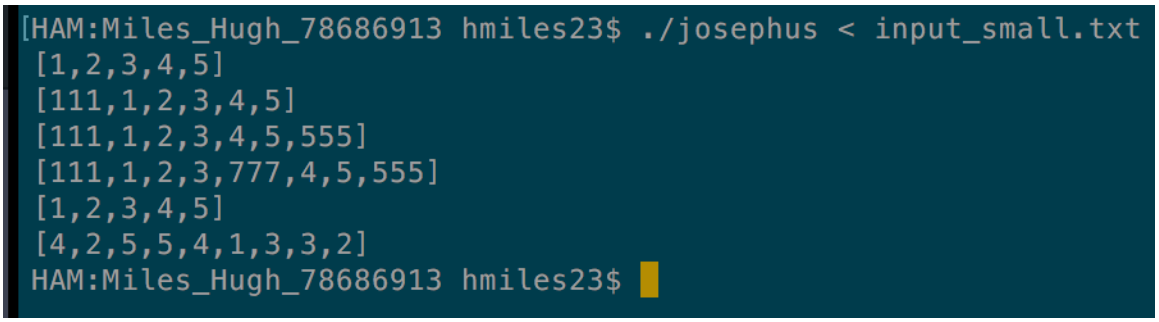
    1. Given Set >> Inputs

```
19
I 1 0
I 2 1
I 3 2
I 4 3
I 5 4
I 6 5
I 7 6
I 8 7
I 9 8
I 10 9
S
I 0 2
S
P 5
D 5
P 5
S
P 165
J 3 2 34 23 22
```

```
HAM:Miles_Hugh_78686913 hmiles23$ make
g++ josephus.o -o josephus
HAM:Miles_Hugh_78686913 hmiles23$ ./josephus < input_given.txt
[1,2,3,4,5,6,7,8,9,10]
[1,2,0,3,4,5,6,7,8,9,10]
5
6
[1,2,0,3,4,6,7,8,9,10]
indexToPrint(165) is out of range
[23,1,0,4,7,9,22,2,6,10,3,34,8]
HAM:Miles_Hugh_78686913 hmiles23$
```

2. Small Set

```
17
I 1 0
I 2 1
I 3 2
I 4 3
I 5 4
S
I 111 0
S
I 555 7
S
I 777 4
S
D 0
D 6
D 3
S
J 4 3 5 4 3 2
```



3. Medium Set

```
34
S
I 1 0
I 2 1
I 3 2
I 4 3
I 5 4
I 6 5
I 7 6
I 8 7
I 9 8
I 10 9
I 11 10
I 12 11
S
D 0
S
D 11
D 5
S
P 0
P 1
P 2
P 3
P 4
P 5
P 6
P 7
P 8
P 9
P 10
P 11
P 19
S
J 10 5 100 201 342 54 66 234 1312 313 808 2131
```

```
[HAM:Miles_Hugh_78686913 hmiles23$ ./josephus < input_medium.txt
List is Empty
[1,2,3,4,5,6,7,8,9,10,11,12]
[2,3,4,5,6,7,8,9,10,11,12]
deleteNode.indexToDelete is out of range
[2,3,4,5,6,8,9,10,11,12]
2
3
4
5
6
8
9
10
11
12
indexToPrint(10) is out of range
indexToPrint(11) is out of range
indexToPrint(19) is out of range
[2,3,4,5,6,8,9,10,11,12]
[234,100,6,12,66,3,10,1312,4,2131,201,11,2,313,9,8,808,342,5,54]
```