White board 8

Merging linked lists

White board 8

Problem Domain

- Take 2 linked lists as parameters into a function to zip the lists together & return a reference to the single list head.
- Work to keep O(1) space
- all previously written methods available

Visual

I ListA head → 1 → 4 → 7 → tail
ListB head → 2 → 3 → 6 → tail

Scenario 1
Scenario 2 – listA.size ≠ listB.size
Scenario 3 – ListA.size = null
Scenario 4 – Lists A & B empty

O head 1 → 2 → 4 → 3 → 7 → 6 → tail

Scenario 1
A node∅ → B node∅ → A node1 → B node1 . . . . . .

Scenario 2
A 1 → 7
I B ~~2~~ → 4 → 3
O C 1 → 2 → 7 → 4 → 3

Algorithms

- check if both lists are empty
  return "empty lists"
- check if either list is empty
  - return non empty list unchanged
- check if list A.size != list.B size
  - modify main method acordingly
- if both list sizes ==
  instatiate new array list
  loop A to ∅ & even nodes in new LL
  Loop B to odd nodes in new LL
  —

· finally return newLL.toString

include "head @ start

"->" in between each

& "x" at end

loopthrough LL

```
while (LL size ≥ 0)
  string response
    = "[" + this.data
      + "]->"

return "head=>" +
    response +
    "x"
```

Big O(N) time - several loops

Big O(1) space - one new linkedlist that is big but no nest but that's it

Code

```
public class mergelists (linkedlist
 listA, Linkedlist listB) {
     int aSize = (int) listA.size();
     int bSize = (int) listB.size();
     if (a size = 0 && bSize = 0) {
        return "lists are
                  empty"
     }
```

int

String → int

response = " ";
)

```
if (asize==0 || bSize==0){
    while(asize>0 || bsize>0){
if
(asize>0) { String currentA=
    "[" + all.data+"]->"
    asize++; }
    return currentA.root;

if
(bsize>0) { String crrntB=
    "[" + bll.size +
    "]->"
    bSize++;
    return currntB.root
    }
}
```

```
if (aSize==bSize) {
int index    → String mergedString = new String();
=0             while(aSize>0) {
                  mergString = aString(index)
                    + bString(index);

                  index++;
               } ~~sout~~ ~~return~~ "head =>"
                  + mergeString +
                  "=>tail";
               // test to see if merge
               //   String.length =
               //     aSize + bSize
                  return mergeString
            }
if (aSize>bSize) {
```

```
int count=0
while(a size==bSize) {
  merge string =
     aLL[count] +
     bLL [count];

  } Sysout
  ~~return~~ "head->"
  + mergeString +
   all [all.size-count]
}  +"]->x"; return
                merge
                String;
                root
if (bSize > aSize) {
   int noder=0
   while(bsize==aSize {
   mergeString = all[
Sout noder] + bLL[noder];
   ~~return~~ "[head" + merge
       String + bll[bll.size-noder]
+ "->x"; return string [illegible]
            }
        }
    }
//end class
```