

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

① List A head $\rightarrow 1 \rightarrow 4 \rightarrow 7 \rightarrow \text{tail}$
List B head $\rightarrow 2 \rightarrow 3 \rightarrow 6 \rightarrow \text{tail}$

Scenario 1

Scenario 2 - List A.size \neq List B.size

Scenario 3 - List A.size = null

Scenario 4 - Lists A & B empty

② head $\rightarrow 1 \rightarrow 2 \rightarrow 4 \rightarrow 3 \rightarrow 7 \rightarrow 6 \rightarrow \text{tail}$

Scenario 1
A node $\phi \rightarrow$ B node $\phi \rightarrow$ A node $1 \rightarrow$
B node

Scenario 2

① A $1 \rightarrow 7$
B $2 \rightarrow 4 \rightarrow 3$
② C $1 \rightarrow 2 \rightarrow 7 \rightarrow 4 \rightarrow 3$

Algo rathms

- 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 accordingly

- if both list sizes ==
instantiate new array list
loop A to ϕ & even
nodes in new
LL
Loop B to odd nodes
in new LL

• finally return new LL.toString
 include "head @ start"
 "→" in between each
 "X" at end
 loop through LL ~~to~~
 while (LL size \geq 0)
 string response
 = "[" + this.data
 + "→"
 return "head →" +
 response +
 "X"

Big O(N) time - several loops
Big O(1) space - one new ^{but no} linked list that is ^{very} but that's it

Code

```
public class mergeLists (LinkedList  
listA, LinkedList listB) {  
    int aSize = (int)listA.size();  
    int bSize = (int)listB.size();  
    if (aSize == 0 || bSize == 0) {  
        return "Lists are  
        empty"  
    }  
}
```

int
String →
response = " " ;
}

```

if (a.size == 0 || b.size == 0) {
    while (a.size > 0 || b.size > 0) {
        if (a.size > 0) {
            string currentA =
                a[" "] + all.data + " ] => "
            a.size++;
        }
        return currentA.root;
    }
}

```

```

if (b.size > 0) {
    string currentB =
        " [" + b[" "] + b.size +
        " ] => "
    b.size++;
    return currentB.root;
}

```

3 3

```

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

```

        index++;
    
```

```

    } // out
    return "head ->"
        + mergedString +
        " -> tail";
    
```

```

    // test to see if merge
    // String.length =
    // aSize + bSize
    return mergedString;
    }
    
```

```

    if (aSize > bSize) {
        //
    }
    }
    
```

```

int count = 0
while (a.size == b.size) {
    mergeString =
        all[count] +
        bLL[count];
}

```

```

}
return "head -> "
+ mergeString +

```

```

all[all.size - count]
+ "]" -> X";

```

```

}
if (b.size > a.size) {
    int node = 0
}

```

```

while (b.size == a.size) {

```

```

    mergeString = all[
        node] + bLL[node];

```

```

return "head" + merge
String + bLL[b.size - node]

```


$+^u \rightarrow x^u$; rechnung
ungetriggert
li. ist
root

3

3

(und class)