Base Case

5 min

Before we consider the base and recursive cases, let's think about the two parameters required to traverse a linked list recursively:

- data – the first

Preview: Docs Loading link description

parameter

. This is the value of the Node that is being searched for in the linked list.

- currentNode – the second parameter. This is the current node in the linked list. During each recursive call, the function will pass the next node as this

Preview: Docs Loading link description

argument

.

```
class LinkedList {

  findNodeRecursively(data, currentNode = this.head) {
   // Some code
  }
}
```

Notice, we added this.head as the default argument for currentNode. This is useful because, if you call findNodeRecursively() with only a data argument, the

Preview: Docs Loading link description

method

will traverse the entire linked list beginning from its head.

Now let's consider the base case for our linked list. We should return a value under the following two cases:

- If the method finds a node with the matching value, it should return the node.
- If the method reaches the end of the list, it should return null.

Instructions

1. Checkpoint 1 Passed

1.

Return null when .findNodeRecursively() reaches the end of the linked list.

Hint

When .findNodeRecursively() is called by the last node, the data argument will be null.

Use the if statement below to check if the currentNode is null:

```
...
} if (currentNode === null) {
  //Do something
}
```

Within the if block, return null.

2. Checkpoint 2 Passed

2.

In .findNodeRecursively(), add an else if statement that returns the currentNode if the node's data attribute is equal to the input data argument.

Hint

Use the else if statement below to check for a match with the current node:

```
else if (currentNode.data  === data) {
  // Do something
}
```

Within the if block, return the currentNode.

3. Checkpoint 3 Passed

3.

Add the following code to **index.js**:

```
const myNodeRecursive = myList.findNodeRecursively('Node 4');
console.log(myNodeRecursive);
```

This code should find the Node with a data argument equal to 'Node 4' and log it to the console.

4. Checkpoint 4 Passed

4.

Change the call to .findNodeRecursively() so it searches for 'Node 3'. Run the code.

This will print undefined, because we have not set our recursive case.