Remove Elements from a Linked List by Index

Before we move on to another data structure, let's get a couple of last bits of practice with linked lists.

Let's write a `removeAt` method that removes the `element` at a given `index`. The method should be called `removeAt(index)`. To remove an `element` at a certain `index`, we'll need to keep a running count of each node as we move along the linked list.

A common technique used to iterate through the elements of a linked list involves a *'runner'*, or sentinel, that 'points' at the nodes that your code is comparing. In our case, starting at the `head` of our list, we start with a `currentIndex` variable that starts at `0`. The `currentIndex` should increment by one for each node we pass.

Just like our `remove(element)` method, which we covered in a previous lesson, we need to be careful not to orphan the rest of our list when we remove the node in our `removeAt(index)` method. We keep our nodes contiguous by making sure that the node that has reference to the removed node has a reference to the next node.

Write a `removeAt(index)` method that removes and returns a node at a given `index`. The method should return `null` if the given `index` is either negative, or greater than or equal to the `length` of the linked list.

Note: Remember to keep count of the `currentIndex`.

	Your `LinkedList` class should have a `removeAt` method.
	Your `removeAt` method should reduce the `length` of the linked list by one.
	Your `removeAt` method should remove the element at the specified index from the linked list.

	When only one element is present in the linked list, your `removeAt` method should remove and return the element at specified index, and reduce the length of the linked list.
	Your `removeAt` method should return the element of the removed node.
	Your `removeAt` method should return `null` and the linked list should not change if the given index is less than `0`.
	Your `removeAt` method should return `null` and the linked list should not change if the given index is greater than or equal to the `length` of the list.