**Problem Domain**:

- Write a method for the Linked List class which takes a number, **k**, as a parameter. Return the node's value that is **k** from the end of the linked list. You have access to the Node class and all the properties on the Linked List class as well as the methods created in previous challenges.

**Input**:

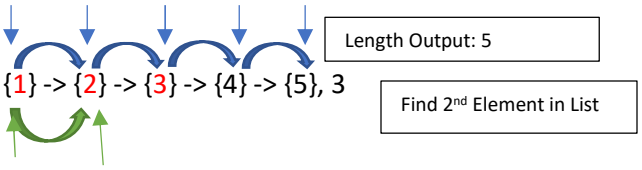
1. int

**Output**:

1. node value

**Visual**:

- Input {1} -> {2} -> {3} -> {4} -> {5}, 3




- Output 2

**Edge Case**:

1. K is greater than length - 1
2. K is Negative
3. Empty List

**Big O:**

1. Time: O(n)
2. Space: O(1)

**Algorithm**:

1. Create a method called KthFromEnd that takes in a parameter of type int
2. Declare a variable of type int for counting the nodes
3. Declare a variable of node for current pointer
4. Check to make sure k is greater than -1, and list is not empty
5. Iterate through to the end of the list
6. Increment counter variable for each node hit
7. Check if counter is greater than 0 and if counter is less than k
8. Reset current pointer to Head
9. Iterate through the list until we reach counter subtracting k times
10. Return node value

**Psuedo Code**:

Algorithm KthFromEnd(k)

- node current = head
- int counter = 0;
-
- If k > -1 && head != null
  - While current != null
    - counter++
    - current = current.next
- else
  - throw new ExceptionOutofRange
- current = head
- if Counter > k
  - for I to counter – k
    - current = current.next
- else
  - throw new ExceptionOutofRange
- return current.value

**Verification**:

- Input {1} -> {2} -> {3} -> {4} -> {5}, 3

| Current | Counter | Counter – K | I | I < counter - k |
|---|---|---|---|---|
| 1 | 1 | | | |
| 2 | 2 | | | |
| 3 | 3 | | | |
| 4 | 4 | | | |
| 5 | 5 | | | |
| 1 | 5 | 5 – 3 = 2 | 0 | 0 < 2 true |
| 2 | 5 | 5 – 3 = 2 | 1 | 1 < 2 true |
| 2 | | | 2 | 2 < 2 false |

- Output of 2