Challenge 7: Find Middle Node of Linked List

Here's another interesting coding challenge to test your knowledge on linked lists.

We'll cover the following

Problem Statement#

You have to implement the `find_mid()` function which will take a linked list as an input and return the value of the **middle node**. If the length of the list is even, the middle value will occur at $\frac{length}{2}$. For a list of odd length, the middle value will be $\frac{length}{2} + 1$.

Input#

A singly linked list.

Output#

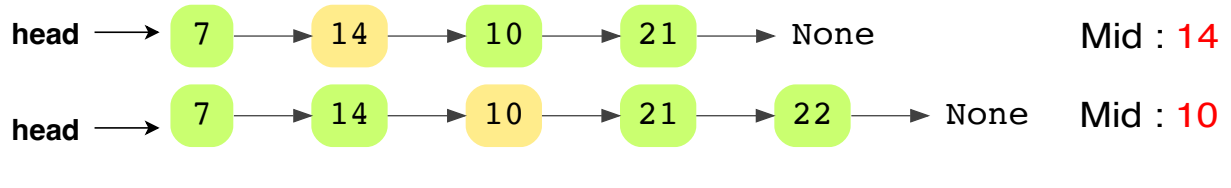
The integer value of the middle node.

Sample Input#

```
LinkedList = 7->14->10->21
```

Sample Output#

```
14
```



Finding Middle Value

Coding Exercise

This one shouldn't be too tricky. It can be done in more than one way. Take some time to solidify your logic and then try it out below.

If you get stuck, the hints and the solution are always available for assistance.

Good luck!

main.py

LinkedList.py

Node.py

```
from LinkedList import LinkedList
# Access head_node => list.get_head()
# Check if list is empty => list.is_empty()
# Length of the list => list.length()
# Node class  { int data ; Node next_element;}


def find_mid(lst):
    # Write your code here
    slow = fast = lst.get_head()
    while slow and fast and fast.next_element:
        fast = fast.next_element.next_element
        if not fast:
            break
        slow = slow.next_element

    return slow.data
```


Back

Solution Review: Detect Loop in a Link...

Next

Solution Review: Find Middle Node of ...

Completed

Report an Issue