Middle of a linked list

BRUTE FORCE

Count number of node by traversing the whole linked list. Divide by 2 to find the middle node then traverse again as soon as pointer points to cnt/2 return the node.

```
Node *findMiddle(Node *head) {
    // Write your code here
    Node* temp=head;
    int cnt=0;
    while(temp!=NULL)
    {
        cnt++;
        temp=temp->next;
    }
    temp=head;
    cnt=cnt/2;
    while(cnt--)
    {
        temp=temp->next;
    }
    return temp;
}
```

- Time Complexity : O(N)+O(N/2)
- Space Complexity : O(1)

Optimal Approach : using tortoise method

using fast and slow pointers. when either fast pointers reaches the last node or points to end of linked list then slow pointer will point to the middle element. We move slow by 1 and fast by 2 which means fast will reach end by N/2 time as it moves two steps at a time while sow moves only 1 step which means when fast reaches end slow will be in the middle.

```
Node *findMiddle(Node *head) {
    // Write your code here
    Node* slow=head;
    Node* fast=head;
    while(fast!=NULL && fast->next!=NULL)
    {
        slow=slow->next;
        fast=fast->next->next;
    }
    return slow;

}
```

- Time Complexity : O(N/2) as when fast reaches end the loop ends.
- Space Complexity : O(1)