Middle of linked list

Given a singly linked list, find the middle of the linked list. For example, if the given linked list is 1->2->3->4->5 then the output should be 3.
If there are even nodes, then there would be two middle nodes, we need to print the second middle element. For example, if the given linked list is 1->2->3->4->5->6 then the output should be 4.

Method 1: Traverse the whole linked list and count the no. of nodes. Now traverse the list again till count/2 and return the node at count/2.
Below is the implementation of the above approach:

C++ Java

```
 // Java Program for the above approach
class GFG {

    Node head;

    /*Creating a new Node*/
    class Node {
        int data;
        Node next;
        public Node(int data)
        {
            this.data = data;
            this.next = null;
        }
    }

    /*Function to add a new Node*/
    public void pushNode(int data)
    {
        Node new_node = new Node(data);
        new_node.next = head;
        head = new_node;
```

Dash

All

Articles

Videos

Problems

Quiz

Contest

```
    public void printNode()

            System.out.print(temp.data + "->");
            temp = temp.next;
        }
        System.out.print("Null"+"\n");
    }

    /*Finding the length of the list.*/
    public int getLen()
    {
        int length = 0;
        Node temp = head;
        while (temp != null) {
            length++;
            temp = temp.next;
        }
        return length;
    }

    /*Printing the middle element of the list.*/
    public void printMiddle()
    {
        if (head != null) {
            int length = getLen();
            Node temp = head;
            int middleLength = length / 2;
            while (middleLength != 0) {
                temp = temp.next;
                middleLength--;
            }
            System.out.print("The middle element is ["
                                + temp.data + "]");
            System.out.println();
        }
    }

    public static void main(String[] args)
    {
```

Menu

Track Progress

64 of **132** Complete. (49%)

```
            list.pushNode(i);
        }
    }
```

Dash
All
Articles
Videos
Problems
Quiz
Contest

Output

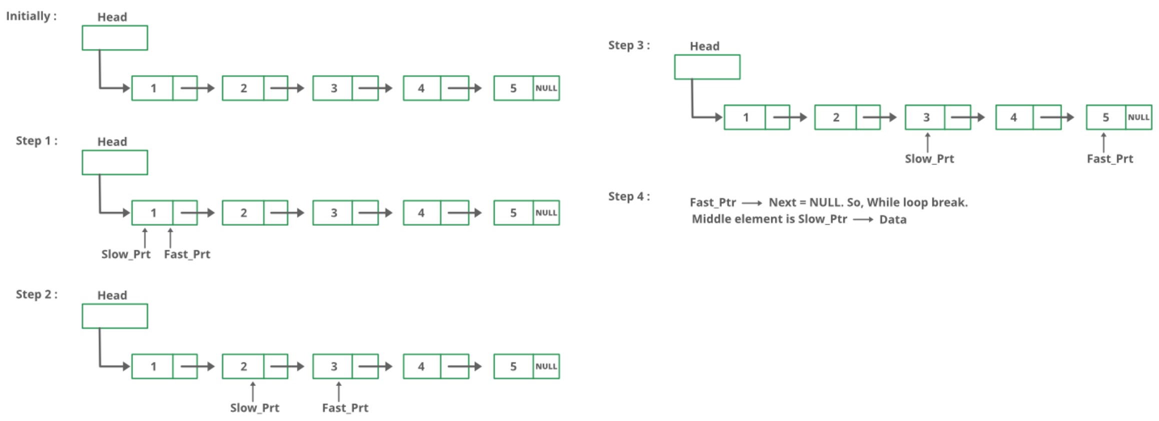
```
5->NULL
The middle element is [5]
4->5->NULL
The middle element is [5]
3->4->5->NULL
The middle element is [4]
2->3->4->5->NULL
The middle element is [4]
1->2->3->4->5->NULL
The middle element is [3]
```

Time Complexity: O(n) where n is no of nodes in linked list

Auxiliary Space: O(1)

Method 2: Traverse linked list using two-pointers. Move one pointer by one and the other pointers by two. When the fast pointer reaches the end, the slow pointer will reach the middle of the linked list.

Below image shows how printMiddle function works in the code :



Menu

Track Progress
64 of **132** Complete. (49%)

Dash
All
Articles
Videos
Problems
Quiz
Contest

```
Node head; // head of linked list

/* Linked list node */
class Node
{
    int data;
    Node next;
    Node(int d)
    {
        data = d;
        next = null;
    }
}

/* Function to print middle of linked list */
void printMiddle()
{
    Node slow_ptr = head;
    Node fast_ptr = head;

        while (fast_ptr != null && fast_ptr.next != null)
        {
            fast_ptr = fast_ptr.next.next;
            slow_ptr = slow_ptr.next;
        }
        System.out.println("The middle element is [" +
                            slow_ptr.data + "] \n");

}

/* Inserts a new Node at front of the list. */
```

Courses
Tutorials
Jobs
Practice
Menu
Contests
P

```
    /* 1 & 2: Allocate the Node &
            Put in the data*/
    Node new_node = new Node(new_data);
```

Track Progress
64 of 132 Complete. (49%)

Dash
All
Articles
Videos
Problems
Quiz
Contest

```
    /* This function prints contents of linked list
    starting from the given node */
    public void printList()
    {
        Node tnode = head;
        while (tnode != null)
        {
            System.out.print(tnode.data+"->");
            tnode = tnode.next;
        }
        System.out.println("NULL");
    }

    public static void main(String [] args)
    {
        LinkedList llist = new LinkedList();
        for (int i=5; i>0; --i)
        {
            llist.push(i);
            llist.printList();
            llist.printMiddle();
        }
    }
}
```

Output

```
5->NULL
The middle element is [5]
4->5->NULL
The middle element is [5]
3->4->5->NULL
The middle element is [4]
```

Menu

Track Progress
64 of **132** Complete. (49%)

```
1->2->3->4->5->NULL
```

Dash

All

Auxiliary Space: O(1), As constant extra space is used.

Articles

Mark as Read

Videos

Report An Issue

If you are facing any issue on this page. Please let us know.

Problems

Quiz

Contest

Menu

Track Progress

64 of **132** Complete. (49%)