AZHAR ALI
CMS : 023-23-0314
Sec A

ASSIGMENT 1

DSA

ASSIGNMENT 1

1 . Write a function to get the nth node from the end of the linked list.

Function name: int nthFromLast(int n);

Case-1: (List Empty) Head=Null then return LIST_EMPTY

Case-2: (List Non-Empty) Head != Null then return nth element from the end of list

Example

Input: 10 -> 20 -> 30 -> 40 -> 50, n = 2

Output: 40 (From the last, second node conatins the data 40)

```
int nthFromLast(int n){
    if(head==null){
        System.out.println("List is Empty!");
        return 0;
    }
    else{
        Node temp = head;
        for(int i = size ; i!=n ; i--){
            temp=temp.next;
        }
        return temp.data;
}
}
```

```
10-->20-->30-->40-->50-->
Return 2 Element : 40
```

ASSIGNMENT 1

Write a function to sort the given single linked list. (Don't swap the data present in the nodes, swap the nodes itself.)

Function name: void sort();
Case-1: (List Empty) Head=Null then return LIST_EMPTY
Case-2: (List Non-Empty) Head != Null then swap the nodes to sort them

```java
public void sort() {
    if (size > 1) {
        for (int i = 0; i < size - 1; i++) {
            Node currentNode = head;
            Node previous = null;

            while (currentNode != null && currentNode.next != null) {
                Node nextNode = currentNode.next;

                if (currentNode.data > nextNode.data) {

                    currentNode.next = nextNode.next;
                    nextNode.next = currentNode;

                    if (previous != null) {
                        previous.next = nextNode;
                    } else {
                        head = nextNode;
                    }

                    previous = nextNode;
                } else {
                    previous = currentNode;
                    currentNode = currentNode.next;
                }
            }
        }
    }
}
```

Example

Input: 50 -> 40 -> 30 -> 20 -> 10

Output: 10 -> 20 -> 30 -> 40 -> 50

```
0-->10-->Null
 From last : 20

st
0-->50-->Null
```

ASSIGNMENT 1

1. Write a function to reverse the single linked list.

   Function name: void reverse();

   Case-1: (List Empty) Head=Null then return LIST_EMPTY

   Case-2: (List Non-Empty) Head != Null then reverse the list

   Example

   Input: 50 -> 40 -> 30 -> 20 -> 10

   Output: 10 -> 20 -> 30 -> 40 -> 50

```
Linked_List1 reverse = new Linked_List1();
if(head==null){
    System.out.println("List is Empty!");
}
else {
    Node temp = head;
    while(temp!=null){
        reverse.addF(temp.data);
        temp=temp.next;
    }

}
return reverse;
```

```
Sorted Linked List
10-->20-->30-->40-->50-->Null
After reversed Method.
50-->40-->30-->20-->10-->Null
```

ASSIGNMENT 1

2. Write a function to remove the duplicates data present in the single linked list.
   Function name: void removeDuplicates();
   Case-1: (List Empty) Head=Null then return LIST_EMPTY
   Case-2: (List Non-Empty) Head != Null then remove duplicate elements
   Example
   Input: 5 -> 3 -> 4 -> 5 -> 2 -> 1 -> 4 -> 5 -> 3
   Output: 5 -> 3 -> 4 -> 2 -> 1

```
void CheckDup(){
    if(head==null){
        System.out.println("List is Empty!");
        return;
    }
    else{
    Node temp = head;
    Node current = head;

    while (current != null) {
        Node runner = current;


        while (runner.next != null) {
            if (runner.next.data == current.data) {

                runner.next = runner.next.next;
            } else {
                runner = runner.next;
            }
        }

        current = current.next;
    }
}
}
```

```
50-->40-->30-->20-->10-->10-->Null
List After Check Duplicate method
50-->40-->30-->20-->10-->Null
PS C:\Users\HIJAZ TRADERS>
```