Assignment Questions 12

Question 1
Given a singly linked list, delete middle of the linked list. For example, if given linked list is 1->2->3->4->5 then linked list should be modified to 1->2->4->5.If there are even nodes, then there would be two middle nodes, we need to delete the second middle element. For example, if given linked list is 1->2->3->4->5->6 then it should be modified to 1->2->3->5->6.If the input linked list is NULL or has 1 node, then it should return NULL
Example 1:
Input:
LinkedList: 1->2->3->4->5
Output:1 2 4 5

Example 2:
Input:
LinkedList: 2->4->6->7->5->1
Output:2 4 6 5 1

code:-

```
class Solution {
    public ListNode deleteMiddle(ListNode head) {

        //when only 1 node are there
        if(head.next==null){
            return null;
        }

        //when only 2 node are there
        if(head.next.next==null){
            head.next = null;
            return head;
        }

        ListNode slow = head;
        ListNode fast = head;

        while(fast!=null && fast.next!=null){
            slow = slow.next;
            fast = fast.next.next;
        }

        slow.val = slow.next.val;
        slow.next = slow.next.next;

        return head;
    }
}
```

Question 2.
Given a linked list of N nodes. The task is to check if the linked list has a loop. Linked list can contain self loop.
Example 1:
Input:
N = 3
value[] = {1,3,4}

x(position at which tail is connected) = 2
Output:True
Explanation:In above test case N = 3.
The linked list with nodes N = 3 is
given. Then value of x=2 is given which
means last node is connected with xth
node of linked list. Therefore, there
exists a loop.
Example 2:
Input:
N = 4
value[] = {1,8,3,4}
x = 0
Output:False
Explanation:For N = 4 ,x = 0 means
then lastNode->next = NULL, then
the Linked list does not contains
any loop.

code:-

```
public class Solution {
    public boolean hasCycle(ListNode head) {
        if (head == null || head.next == null)
            return false;
        ListNode p1 = head;
        ListNode p2 = head.next;
        while (p1 != p2){
            if (p2 == null || p2.next == null)
                return false;
            p1 = p1.next;
            p2 = p2.next.next;
        }
        return true;
    }
}
```

Question 3.
Given a linked list consisting of L nodes and given a number N. The task is to find the Nth node from the e
nd of the linked list.
Example 1:
Input:
N = 2
LinkedList: 1->2->3->4->5->6->7->8->9
Output:8
Explanation:In the first example, there
are 9 nodes in linked list and we need
to find 2nd node from end. 2nd node
from end is 8.
Example 2:
Input:
N = 5
LinkedList: 10->5->100->5
Output:-1

Explanation:In the second example, there are 4 nodes in the linked list and we need to find 5th from the end. Since 'n' is more than the number of nodes in the linked list, the output is -1.

code:-

```
import java.io.*;
class LinkedList {
 Node head;
 class Node {
  int data;
  Node next;
  Node(int d)
  {
   data = d;
   next = null;
  }
 }
 void printNthFromLast(int N)
 {
  int len = 0;
  Node temp = head;
  while (temp != null) {
   temp = temp.next;
   len++;
  }
  if (len < N)
   return;
  temp = head;
  for (int i = 1; i < len - N + 1; i++)
   temp = temp.next;
  System.out.println(temp.data);
 }
 public void push(int new_data)
 {
  Node new_node = new Node(new_data);
  new_node.next = head;
  head = new_node;
 }
 public static void main(String[] args)
 {
  LinkedList llist = new LinkedList();
  llist.push(20);
  llist.push(4);
  llist.push(15);
  llist.push(35);
  llist.printNthFromLast(4);
 }
}
```

Question 4.
Given a singly linked list of characters, write a function that returns true if the given list is a palindrome, el se false.
Examples:

Input: R->A->D->A->R->NULL
Output: Yes
Input:C->O->D->E->NULL
Output: No

code:-

```
class linkedList {
 public static void main(String args[])
 {
  Node one = new Node(1);
  Node two = new Node(2);
  Node three = new Node(3);
  Node four = new Node(4);
  Node five = new Node(3);
  Node six = new Node(2);
  Node seven = new Node(1);
  one.ptr = two;
  two.ptr = three;
  three.ptr = four;
  four.ptr = five;
  five.ptr = six;
  six.ptr = seven;
  boolean condition = isPalindrome(one);
  System.out.println("isPalidrome :" + condition);
 }
 static boolean isPalindrome(Node head)
 {

  Node slow = head;
  boolean ispalin = true;
  Stack<Integer> stack = new Stack<Integer>();

  while (slow != null) {
   stack.push(slow.data);
   slow = slow.ptr;
  }

  while (head != null) {

   int i = stack.pop();
   if (head.data == i) {
    ispalin = true;
   }
   else {
    ispalin = false;
    break;
   }
   head = head.ptr;
  }
  return ispalin;
 }
}

class Node {
 int data;
```

```
 Node ptr;
 Node(int d)
 {
  ptr = null;
  data = d;
 }
}
```

Question 5.
Given a linked list of N nodes such that it may contain a loop.
A loop here means that the last node of the link list is connected to the node at position X(1-based index).
If the link list does not have any loop, X=0.
Remove the loop from the linked list, if it is present, i.e. unlink the last node which is forming the loop.
Example 1:
Input:
N = 3
value[] = {1,3,4}
X = 2
Output:1
Explanation:The link list looks like
1 -> 3 -> 4
     ^ |
     |____|
A loop is present. If you remove it
successfully, the answer will be 1.
Example 2:
Input:
N = 4
value[] = {1,8,3,4}
X = 0
Output:1
Explanation:The Linked list does not
contains any loop.
Example 3:
Input:
N = 4
value[] = {1,2,3,4}
X = 1
Output:1
Explanation:The link list looks like
1 -> 2 -> 3 -> 4
^ |
|_____________|
A loop is present.
If you remove it successfully,
the answer will be 1.

code:-

```
class Solution
{
    //Function to remove a loop in the linked list.
    public static void removeLoop(Node head){
        Node fast = head;
        Node slow = head;
```

```
        Node prev = slow;
        while(fast!=null && fast.next!=null){
            prev = slow;
            fast = fast.next.next;
            slow = slow.next;
            if(fast == slow){
                break;
            }
        }
        if(fast!=slow) return;
        fast = head;
        while(fast!=slow && slow!=head){
            prev = slow;
            fast = fast.next;
            slow = slow.next;
        }
        prev.next = null;
    }
}
```

Question 6
Given a linked list and two integers M and N. Traverse the linked list such that you retain M nodes then de lete next N nodes, continue the same till end of the linked list.

Difficulty Level: Rookie
Examples:
Input:
M = 2, N = 2
Linked List: 1->2->3->4->5->6->7->8
Output:
Linked List: 1->2->5->6

Input:
M = 3, N = 2
Linked List: 1->2->3->4->5->6->7->8->9->10
Output:
Linked List: 1->2->3->6->7->8

Input:
M = 1, N = 1
Linked List: 1->2->3->4->5->6->7->8->9->10
Output:
Linked List: 1->3->5->7->9

code:-

```
class GFG
{
static class Node
{
 int data;
 Node next;
};
static Node push( Node head_ref, int new_data)
{
```

```
 Node new_node = new Node();
 new_node.data = new_data;
 new_node.next = (head_ref);
 (head_ref) = new_node;
 return head_ref;
}
static void printList( Node head)
{
 Node temp = head;
 while (temp != null)
 {
  System.out.printf("%d ", temp.data);
  temp = temp.next;
 }
 System.out.printf("\n");
}
static void skipMdeleteN( Node head, int M, int N)
{
 Node curr = head, t;
 int count;
 while (curr!=null)
 {
  for (count = 1; count < M && curr != null; count++)
   curr = curr.next;
  if (curr == null)
   return;
  t = curr.next;
  for (count = 1; count <= N && t != null; count++)
  {
   Node temp = t;
   t = t.next;
  }
  curr.next = t;
  curr = t;
 }
}
public static void main(String args[])
{
 Node head = null;
 int M=2, N=3;
 head=push(head, 10);
 head=push(head, 9);
 head=push(head, 8);
 head=push(head, 7);
 head=push(head, 6);
 head=push(head, 5);
 head=push(head, 4);
 head=push(head, 3);
 head=push(head, 2);
 head=push(head, 1);
 System.out.printf("M = %d, N = %d \nGiven" +"Linked list is :\n", M, N);
 printList(head);
 skipMdeleteN(head, M, N);
 System.out.printf("\nLinked list after deletion is :\n");
 printList(head);
```

```
}
}
```

Question 7.
Given two linked lists, insert nodes of second list into first list at alternate positions of first list.
For example, if first list is 5->7->17->13->11 and second is 12->10->2->4->6, the first list should become 5->12->7->10->17->2->13->4->11->6 and second list should become empty. The nodes of second list sh ould only be inserted when there are positions available. For example, if the first list is 1->2->3 and secon d list is 4->5->6->7->8, then first list should become 1->4->2->5->3->6 and second list to 7->8.
Use of extra space is not allowed (Not allowed to create additional nodes), i.e., insertion must be done in-place. Expected time complexity is O(n) where n is number of nodes in first list.

code:-

```
class LinkedList
{
 Node head; // head of list
 class Node
 {
  int data;
  Node next;
  Node(int d) {data = d; next = null; }
 }
 void push(int new_data)
 {
  Node new_node = new Node(new_data);
  new_node.next = head;
  head = new_node;
 }
 void merge(LinkedList q)
 {
  Node p_curr = head, q_curr = q.head;
  Node p_next, q_next;
  while (p_curr != null && q_curr != null) {
   p_next = p_curr.next;
   q_next = q_curr.next;
   q_curr.next = p_next; // change next pointer of q_curr
   p_curr.next = q_curr; // change next pointer of p_curr
   p_curr = p_next;
   q_curr = q_next;
  }
  q.head = q_curr;
 }

 void printList()
 {
  Node temp = head;
  while (temp != null)
  {
  System.out.print(temp.data+" ");
  temp = temp.next;
  }
  System.out.println();
 }
 public static void main(String args[])
 {
```

```
  LinkedList llist1 = new LinkedList();
  LinkedList llist2 = new LinkedList();
  llist1.push(3);
  llist1.push(2);
  llist1.push(1);
  System.out.println("First Linked List:");
  llist1.printList();
  llist2.push(8);
  llist2.push(7);
  llist2.push(6);
  llist2.push(5);
  llist2.push(4);
  System.out.println("Second Linked List:");
  llist1.merge(llist2);
  System.out.println("Modified first linked list:");
  llist1.printList();
  System.out.println("Modified second linked list:");
  llist2.printList();
 }
}
```

Question 8.
Given a singly linked list, find if the linked list is [circular](https://www.geeksforgeeks.org/circular-linked-list/amp/) or not.
A linked list is called circular if it is not NULL-terminated and all nodes are connected in the form of a cycle. Below is an example of a circular linked list.

code:-

```
class GFG {
 static class Node {
  int data;
  Node next;
 }

 static boolean isCircular(Node head)
 {
  if (head == null)
   return true;
  Node node = head.next;
  while (node != null && node != head)
   node = node.next;
  return (node == head);
 }
 static Node newNode(int data)
 {
  Node temp = new Node();
  temp.data = data;
  temp.next = null;
  return temp;
 }
 public static void main(String args[])
 {
  Node head = newNode(1);
  head.next = newNode(2);
  head.next.next = newNode(3);
```

```
  head.next.next.next = newNode(4);
  System.out.print(isCircular(head) ? "Yes\n": "No\n");
  head.next.next.next.next = head;
  System.out.print(isCircular(head) ? "Yes\n": "No\n");
 }
}
```