Data Structures Assignment #1

Name: Muhammad Ahmed
Reg No: SP24-BSE-066
Section: B

Question 1:

"Detecting and Removing Cycles in Linked List"

Task Breakdown

1) Cycle detection
2) Find Starting Node
3) Cycle Removal
4) Edge Case handling
5) Display Corrected Linked List

Code + Algorithm:

1) Node creation

```
public class LinkedListCycle{
        static class Node{
            int data;
            Node Next;
        Node(int d){
            this.data = d
            this.next = null
        }
    }
    //Node Created
Node head;
```

Step 1) Cycle detection:

```
private boolean detectCycle(){
    Node slow = head;
    Node fast = head;
    while(fast != null && fast.next != null){
        slow = slow.next
        fast = fast.next.next;
        if (slow == fast){
            return true;
        }
    }
    return false;
}
```

// This method check if cycle exist in linked list
slow move one step and fast moves 2 steps.
If there is cycle, slow and fast will meet, method will return true otherwise false.

Step 2) Finding starting Node:

```
private Node findStartNode(){
    Node slow = head;
    Node fast = head;
    boolean cycleExists = false;
    while (fast != null && fast.next != null){
        slow = slow.next;
        fast = fast.next;
```

```
        if (slow == fast) {
            cycleExists = true;
            break
        }
    }
    if (!cycleExists) return null;

    slow = head
    while (slow != fast) {
        slow = slow.next;
        fast = fast.next;
    }
    return slow;
}
```

// first check if cycle exist then, move slow and fast one step until they meet. Meeting point is start node of cycle.

Step 3) Remove cycle

```
if (startNode == null) return;
while (temp.next != startnode) {
    temp = temp.next;
} temp.next = null
```

// first find starting node, then traverse cycle until it reaches just before start node. then set next of last node null.

Step 4) Display Linked List:

```
if(detect(cycle1)){
        return;
    }
Node temp = head
while(temp != null){
    sout(temp.data);
    temp = temp.next;
}
Sout("null");
}
// If cycle detect, do not display list
Otherwise, traverse list and print
each node's value.
```