Loop Detection and Deletion in Linked List

```
class Node {
  constructor(data) {
    this.data = data;
    this.next = null;
  }
}

class Solution {
  deleteLoop(head) {
    let tortoise = head;
    let hare = head;
    let loopExists = false;

    while (hare !== null && hare.next !== null) {
      tortoise = tortoise.next;
      hare = hare.next.next;
      if (tortoise === hare) {
        loopExists = true;
        break;
      }
    }
    if (!loopExists) return "No Loop Found";
    tortoise = head;
    let prevHare = null;
    while (tortoise !== hare) {
      prevHare = hare;
      tortoise = tortoise.next;
      hare = hare.next;
    }
    if (hare === head) {
      while (hare.next !== head) {
        hare = hare.next;
      }
    } else {
      prevHare = hare;
      while (prevHare.next !== tortoise) {
        prevHare = prevHare.next;
      }
    }
    prevHare.next = null;
  }
  detectLoop(head) {
    let tortoise = head;
    let hare = head;
    while (hare !== null && tortoise.next !== null) {
      tortoise = tortoise.next;
      hare = hare.next.next;
      if (tortoise === hare) {
        return "Loop Found";
      }
    }
    if (hare === null) return "No Loop Found";
  }
}

let head = new Node(1);
let second = new Node(2);
let third = new Node(3);
let fourth = new Node(4);

head.next = second;
second.next = third;
third.next = fourth;
fourth.next = second;

let solution = new Solution();
console.log(solution.detectLoop(head));
solution.deleteLoop(head);
console.log(solution.detectLoop(head));
```