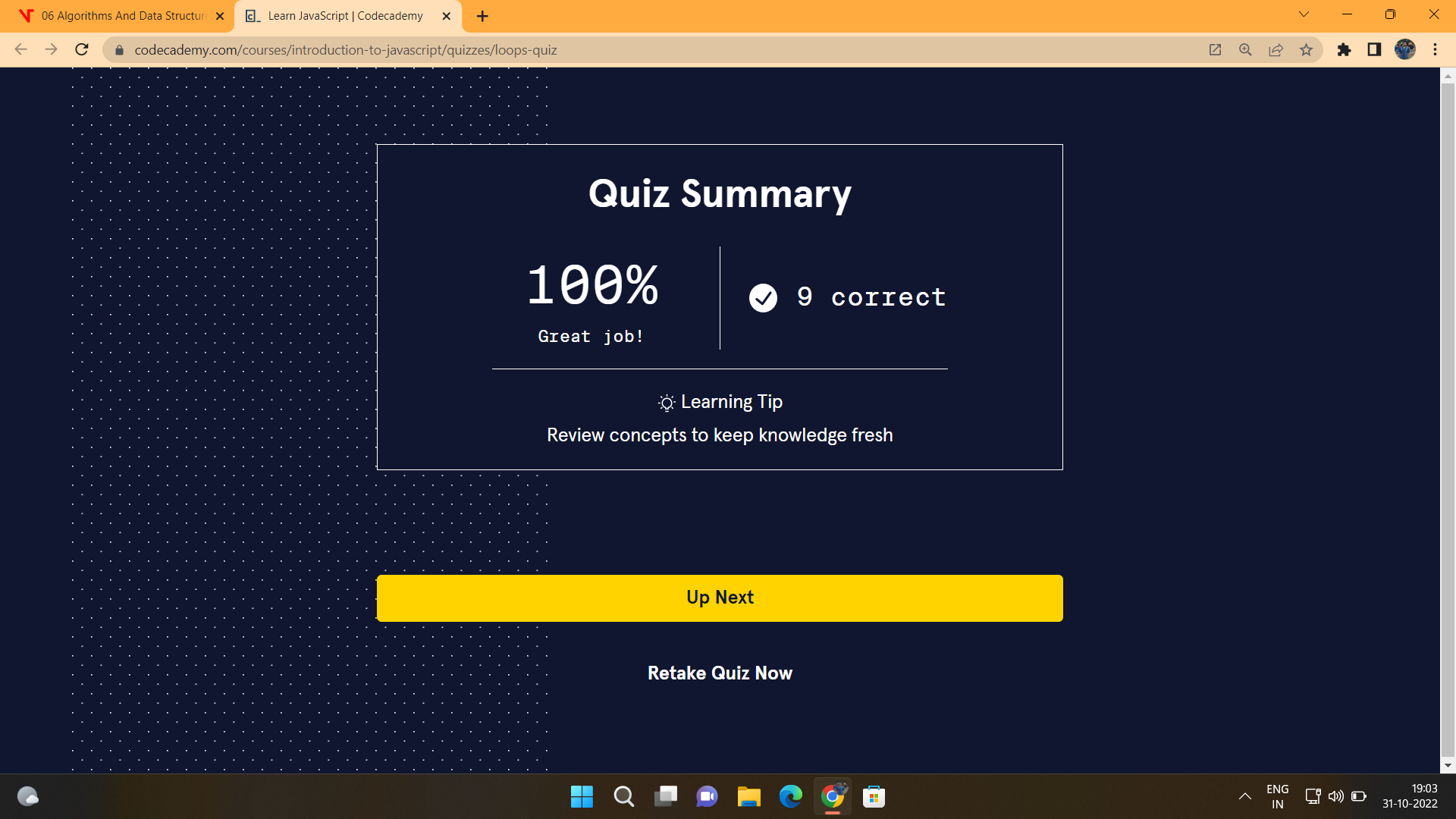
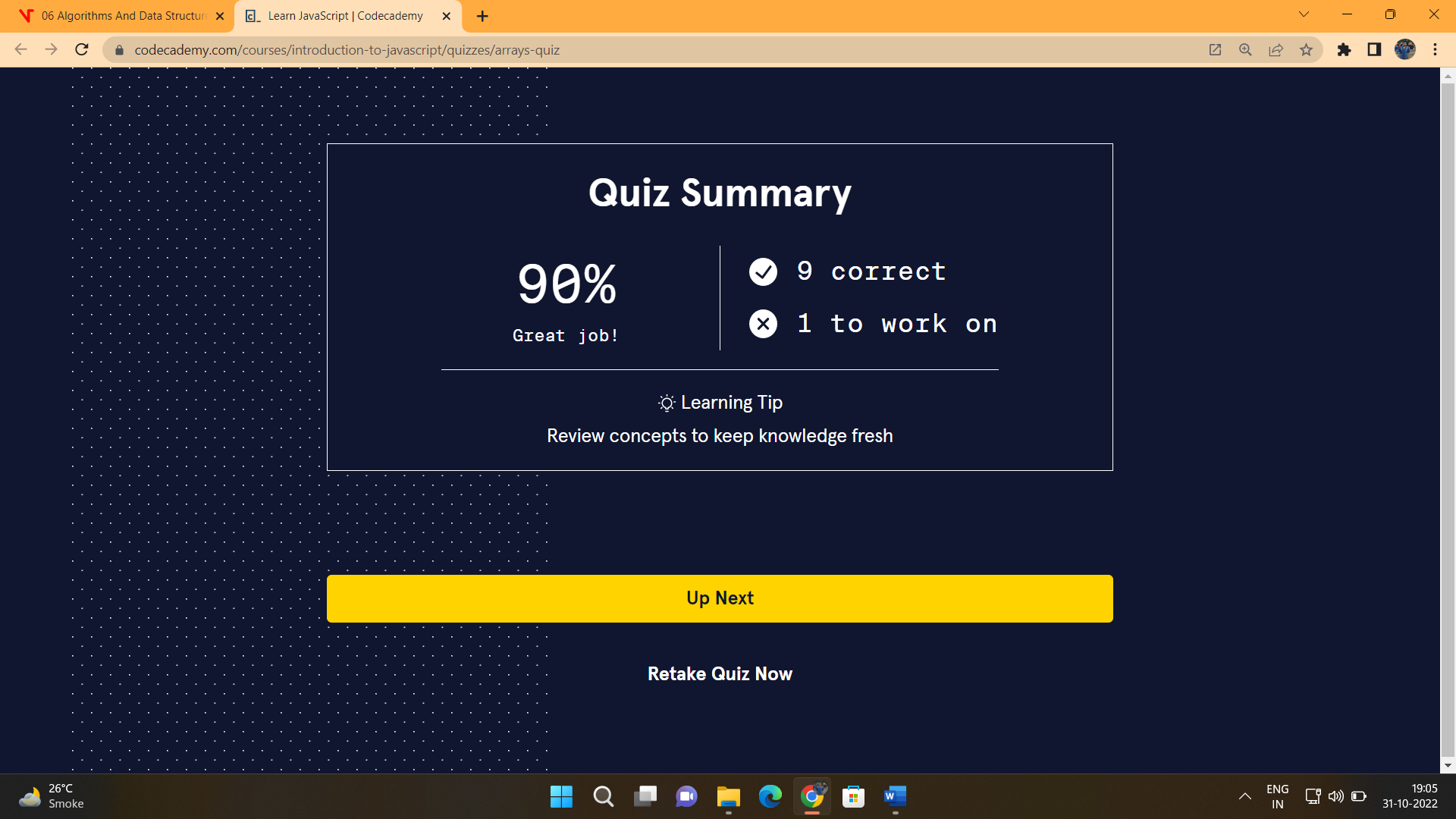
GITHUB LINK:-

Quiz Loops



Quiz Arrays



Question 1 : Write a program to reverse a linked list with a pointer given to head node as given below :

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

<body>

    <script>

    var head;

    class Node {

        constructor(ele) {

            this.data = ele;

            this.next = null;

        }

    }

    function reverse(node) {

    var prev = null;

    var curr = node;

    var next = null;

        while (curr != null) {

            next = curr.next;

            curr.next = prev;

            prev = curr;

            curr = next;

        }

        node = prev;

        return node;

    }

    function printList(node) {

        while (node != null) {

            document.write(node.data + " ");

            node = node.next;

        }

    }

        head = new Node(2);

        head.next = new Node(3);

        head.next.next = new Node(9);

        head.next.next.next = new Node(23);

        document.write("Given linked list<br/>");

        printList(head);

        head = reverse(head);

        document.write("<br/>");

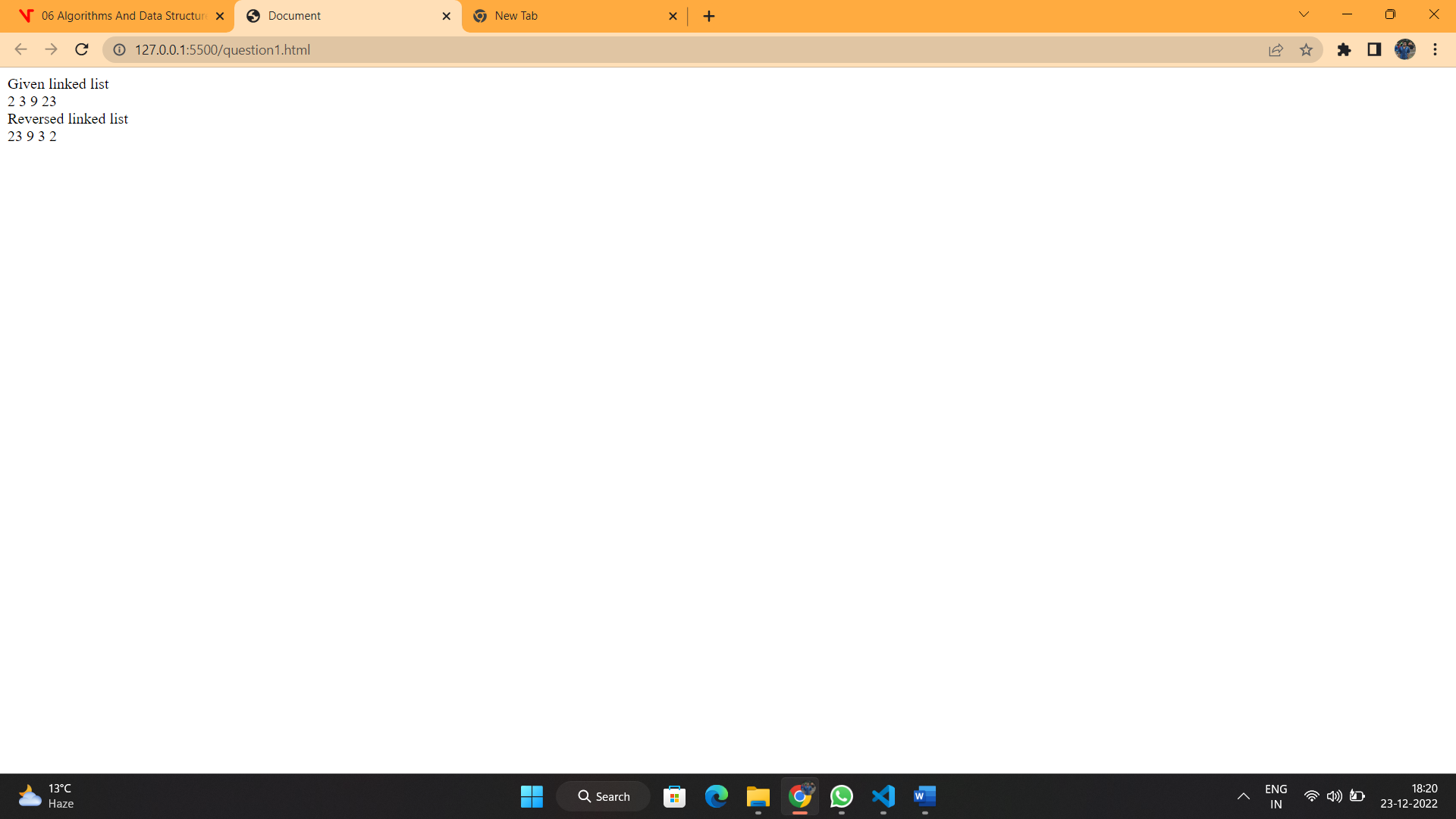
        document.write("Reversed linked list<br/> ");

        printList(head);

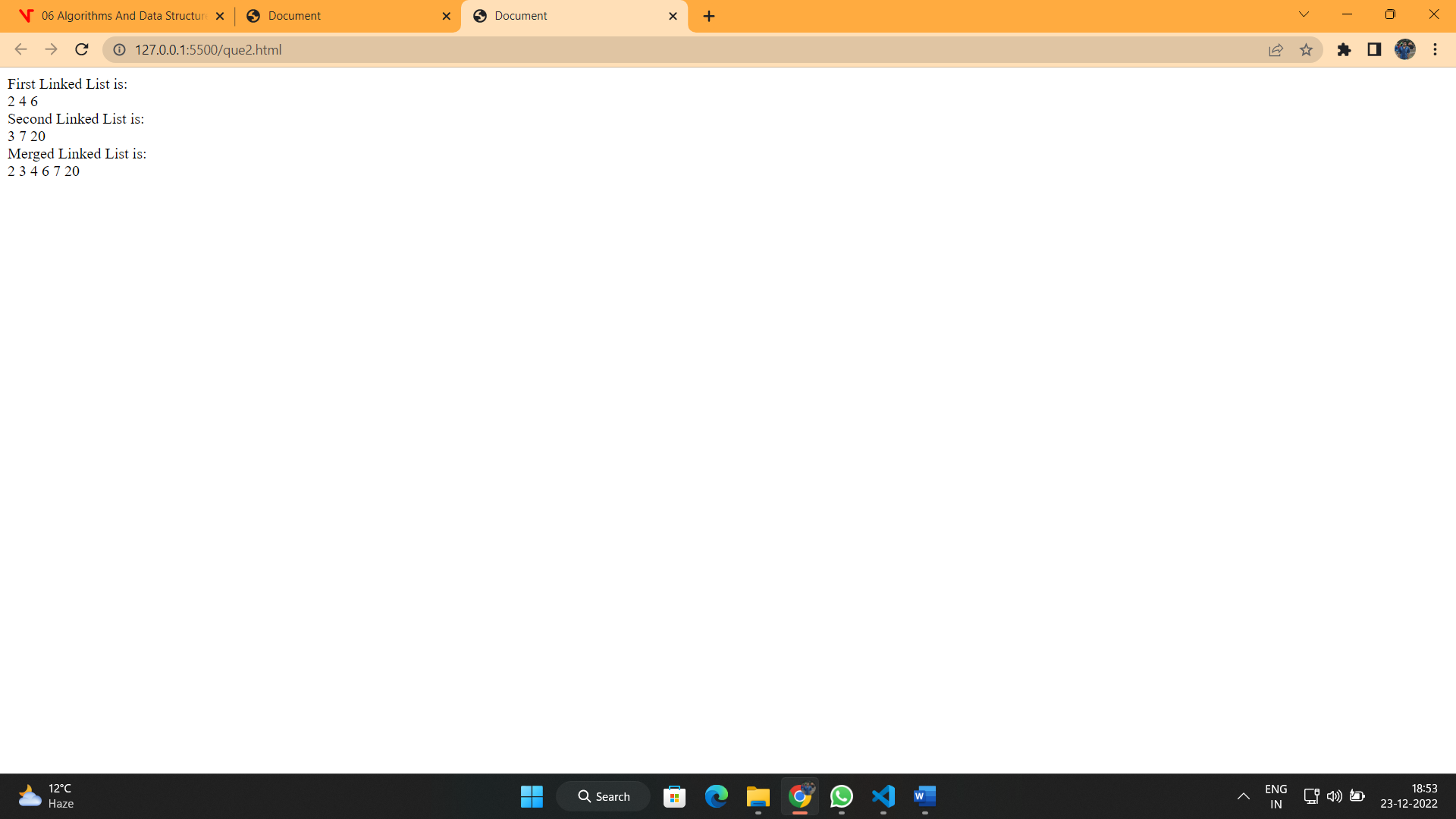
    </script>

</body>

</html>



Question 2: Write a program to create a function which takes two sorted linked lists in ascending order as an input and returns a sorted linked list in ascending order.



<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

<body>

    <script>

        class Node{

            constructor(ele){

                this.data=ele;

                this.next = null;

            }

        }

        class LinkedList{

        constructor(){

            this.head=null;

        }

        addofnode(node){

            if (this.head == null){

                this.head = node;

            }

            else{

                let temp = this.head;

                while (temp.next != null)

                    temp = temp.next;

                temp.next = node;

            }

        }

        printList(){

            let temp = this.head;

            while (temp != null){

                document.write(temp.data + " ");

                temp = temp.next;

            }

            document.write("<br>");

        }

        }

        function sortedMerge(headA,headB){

            let dNode = new Node(0);

            let tail = dNode;

            while(true){

                if(headA == null){

                    tail.next = headB;

                    break;

                }

                if(headB == null){

                    tail.next = headA;

                    break;

                }

                if(headA.data <= headB.data){

                    tail.next = headA;

                    headA = headA.next;

                }

                else{

                    tail.next = headB;

                    headB = headB.next;

                }

                tail = tail.next;

            }

            return dNode.next;

        }

        let l1 = new LinkedList();

        let l2 = new LinkedList();

        l1.addofnode(new Node(2));

        l1.addofnode(new Node(4));

        l1.addofnode(new Node(6));

        l2.addofnode(new Node(3));

        l2.addofnode(new Node(7));

        l2.addofnode(new Node(20));

        document.write("First Linked List is:<br>")

        l1.printList();

        document.write("Second Linked List is:<br>")

        l2.printList();

        l1.head = sortedMerge(l1.head,l2.head);

        document.write("Merged Linked List is:<br>")

        l1.printList();

        </script>

</body>

</html>