**Calculating Height of Binary Trees:**

class Node {

  constructor(val) {

    this.val = val;

    this.left = null;

    this.right = null;

  }

}

const a = new Node("a");

const b = new Node("b");

const c = new Node("c");

const d = new Node("d");

const e = new Node("e");

const f = new Node("f");

a.left = b;

a.right = c;

b.left = d;

b.right = e;

c.right = f;

const one = new Node(1);

const two = new Node(2);

const three = new Node(3);

const four = new Node(4);

const five = new Node(5);

const six = new Node(6);

const seven = new Node(7);

const eight = new Node(8);

one.left = two;

one.right = three;

two.right = five;

two.left = four;

five.right = seven;

five.left = six;

seven.left = eight;

const maxHeight = (root) => {

  if (!root) return 0;

  return Math.max(maxHeight(root.left), maxHeight(root.right)) + 1;

};

console.log(maxHeight(a));

console.log(maxHeight(one));

**Finding Middle of Linked Lists:**

class Node {

  constructor(data) {

    this.data = data;

    this.next = null;

  }

}

class LinkedList {

  constructor(data) {

    this.head = null;

  }

  addFirst(data) {

    const newNode = new Node(data);

    newNode.next = this.head;

    this.head = newNode;

  }

  print() {

    let current = this.head;

    let output = "";

    while (current) {

      output += current.data + " "; // Append each data element and a space to the output string

      current = current.next;

    }

    console.log(output.trim()); // Log the output string and trim any trailing space

  }

  size() {

    let current = this.head;

    let count = 0;

    while (current) {

      current = current.next;

      count++;

    }

    return count;

  }

  middleChild() {

    let current = this.head;

    let forward = this.head.next;

    while (forward && forward.next) {

      current = current.next;

      forward = forward.next.next;

    }

    if (this.size() % 2 === 0) {

      return current.next.data;

    } else {

      return current.data;

    }

  }

}

const linkedlist = new LinkedList();

linkedlist.addFirst(3);

linkedlist.addFirst(13);

linkedlist.addFirst(8);

linkedlist.addFirst(5);

linkedlist.addFirst(15);

linkedlist.addFirst(25);

linkedlist.print();

console.log(linkedlist.middleChild());