public class Staff {
    protected String name;
    protected int id;

    public Staff(String n, int i) {
        name = n;
        id = i;
    }
}
public class Employee extends Staff {
    private int workHours;

    public Employee(String n, int i, int w) {
        super(n, i);
        workHours = w;
    }
}
public class Manager extends Staff {
    private boolean hasPrivileges;

    public Manager(String n, int i, boolean h) {
        super(n, i);
        hasPrivileges = h;
    }
}
public class Node<T> {
    private T data;
    private Node<T> next;

    public Node(T d) {
        data = d;
        next = null;
    }

    public void setData(T d) {
        data = d;
    }

    public T getData() {
        return data;
    }

    public void setNext(Node<T> n) {
        next = n;
    }

    public Node<T> getNext() {
        return next;
    }
}
public class LinkedList<T> {
    private Node<T> first;

    public LinkedList() {
        first = null;
    }

    public void insertFront(T d) {
        Node<T> newNode = new Node<T>(d);
        newNode.setNext(first);
        first = newNode;
    }

    public T removeFront() {
        T data = first.getData();
        first = first.getNext();
        return data;
    }

    public boolean search(T d) {
        Node<T> current = first;

        while (current != null) {
            if (current.getData() == d) {
                return true;
            }
            current = current.getNext();
        }
        return false;
    }

    public int length() {
        Node<T> current = first;
        int count = 0;

        while (current != null) {
            count++;
            current = current.getNext();
        }
        return count;
    }
}
public class Company
{
    private String name;
    private LinkedList<Staff> staff;

    public Company(String n) {
        name = n;
        staff = new LinkedList<Staff>();
    }

    public void add(Staff s) {
        staff.insertFront(s);
    }

    public Staff remove() {
        return staff.removeFront();
    }

    public int getNumOfStaff() {
        return staff.length();
    }
}
Bonus Question:
Is the way that LinkedList<T> class works is similar to a widely known data structure? If yes, What is the name of this data structure? and Why?
If no, just state that.

Answer:
Yes. Stack.
Because it inserts to the front and removes from the front - Last In First Out (LIFO).