Group 19:

|  |  |
| --- | --- |
| Ndungutse Tuyizere Eric | 220002591 |
| UWAYEZU Oscar | 221006589 |
| Ishimwe Pacifique | 221005961 |

**Question**

Delete N Nodes after M Nodes of a Linked List: Create Linked list of size 20, keep first m nodes and delete next n nodes. Repeat the process until you reach the end of linked list and return the head of modified list  
  
File 1: **Node.java**

package assignment;

**public** **class** Node {

**public** **int** data;

**public** Node next;

**public** Node(**int** data) {

        this.data = data;

        this.next = null;

    }

    @**Override**

**public** String toString() {

        return "Node [data=" + data + ", next=" + next + "]";

    }

}

File 2: **Linked List**

package assignment;

**public** **class** LinkedList {

    Node head = null;

    Node tail = null;

**int** size = 0;

**public** **void** insertNode(**int** data) {

        Node newNode = new Node(data);

        if (head == null) {

            head = newNode;

            tail = newNode;

        } else {

            tail.next = newNode;

            tail = newNode;

        }

        size += 1;

    }

**public** **void** print() {

        Node currNode = head;

        while (currNode.next != null) {

            System.out.println(currNode.data);

            currNode = currNode.next;

        }

    }

// A methods that deletes m nodes after n nodes

**public** Node removeMNodesAfterNNodes(**int** m, **int** n) {

*// Node header = head;*

        Node current = head;

        Node prevNode = null;

        Node nextNode = null;

        while (current.next != null) {

            for (**int** i = 1; i <= n; i++) {

                if (i == n) {

                    prevNode = current;

                } else {

                    current = current.next;

                    continue;

                }

            }

            for (**int** j = 1; j <= m; j++) {

                if (j == m && current.next != null) {

                    nextNode = current.next;

                } else {

                    if (current != null && current.next != null)

                        current = current.next;

                }

            }

            if (current.next != null) {

                current = current.next;

            }

            prevNode.next = current.next == null ? null : nextNode.next;

            tail = prevNode;

        }

        return head;

    }

}