

# LinkedList Based Stack

```
public class LinkedListStack {  
    private Node head; // This will be the top of the stack  
  
    // Constructor  
    public LinkedListStack() {  
        this.head = null;  
    }  
  
    // Node inner class  
    private static class Node {  
        int data;  
        Node next;  
  
        Node(int data) {  
            this.data = data;  
            this.next = null;  
        }  
    }  
  
    // Method to push an item onto the stack  
    public void push(int data) {  
        Node newNode = new Node(data);  
        newNode.next = head;  
        head = newNode;  
        System.out.println(data + " pushed to stack");  
    }  
}
```

// Method to check if the stack is empty

```
public boolean isEmpty() {  
    return head == null;  
}
```

// Method to return the top of the stack without removing it

```
public int peek() {  
    if (!isEmpty()) {  
        return head.data;  
    } else {  
        throw new IllegalStateException("Stack is empty");  
    }  
}
```

// Method to remove the top of the stack

```
public int pop() {  
    if (head == null) {  
        throw new IllegalStateException("Stack is empty, cannot pop");  
    }  
    int popped = head.data;  
    head = head.next;  
    System.out.println(popped + " popped from stack");  
    return popped;  
}
```

// Utility method to print the contents of the stack

```
public void printStack() {  
    if (head == null) {  
        System.out.println("Stack is empty");  
    }  
}
```

```

        return;
    }

    System.out.println("Elements in stack : ");

    Node temp = head;
    while (temp != null) {
        System.out.print(temp.data + " ");
        temp = temp.next;
    }
    System.out.println();
}

// Main method to run the program
public static void main(String[] args) {
    LinkedListStack stack = new LinkedListStack();

    stack.push(10);
    stack.push(20);
    stack.push(30);
    stack.printStack();
    stack.peek();
    stack.pop();
    stack.printStack();
}
}

```

## Output

10 pushed to stack

20 pushed to stack

30 pushed to stack

Elements in stack :

30 20 10

30 popped from stack

Elements in stack :

20 10