Reversing Linked List in Groups

```
class Node {
    constructor(data){
        this.data = data
        this.next = null
    }
}

class LinkedList{
    constructor(data){
        this.head = null;
    }
    addLast(data){
        const newNode = new Node(data)
        if(!this.head){
            this.head = newNode
            return
        }
        let current = this.head
        while(current.next){
            current = current.next
        }
        current.next = newNode
    }
    print(head){
        let current = head;
        while(current){
            console.log(current.data)
            current = current.next
        }
    }

    reverseGroup(head , k){
        if(!head) return null
        let count = 0
        let prevPointer = null;
        let nextPointer = null;
        let currentPointer = head;
        while(currentPointer!== null && count<k){

                nextPointer = currentPointer.next
                currentPointer.next = prevPointer
                prevPointer = currentPointer
                currentPointer = nextPointer
                count++


        }
        if(nextPointer!== null){
            head.next = this.reverseGroup(nextPointer, k)
        }
       return  prevPointer
     }
}

const linkedlist = new LinkedList();
linkedlist.addLast(3)
linkedlist.addLast(13)
linkedlist.addLast(8)
linkedlist.addLast(5)
linkedlist.addLast(10)
linkedlist.print(linkedlist.head)
console.log("================================")
linkedlist.head = linkedlist.reverseGroup(linkedlist.head, 2);
linkedlist.print(linkedlist.head)
```