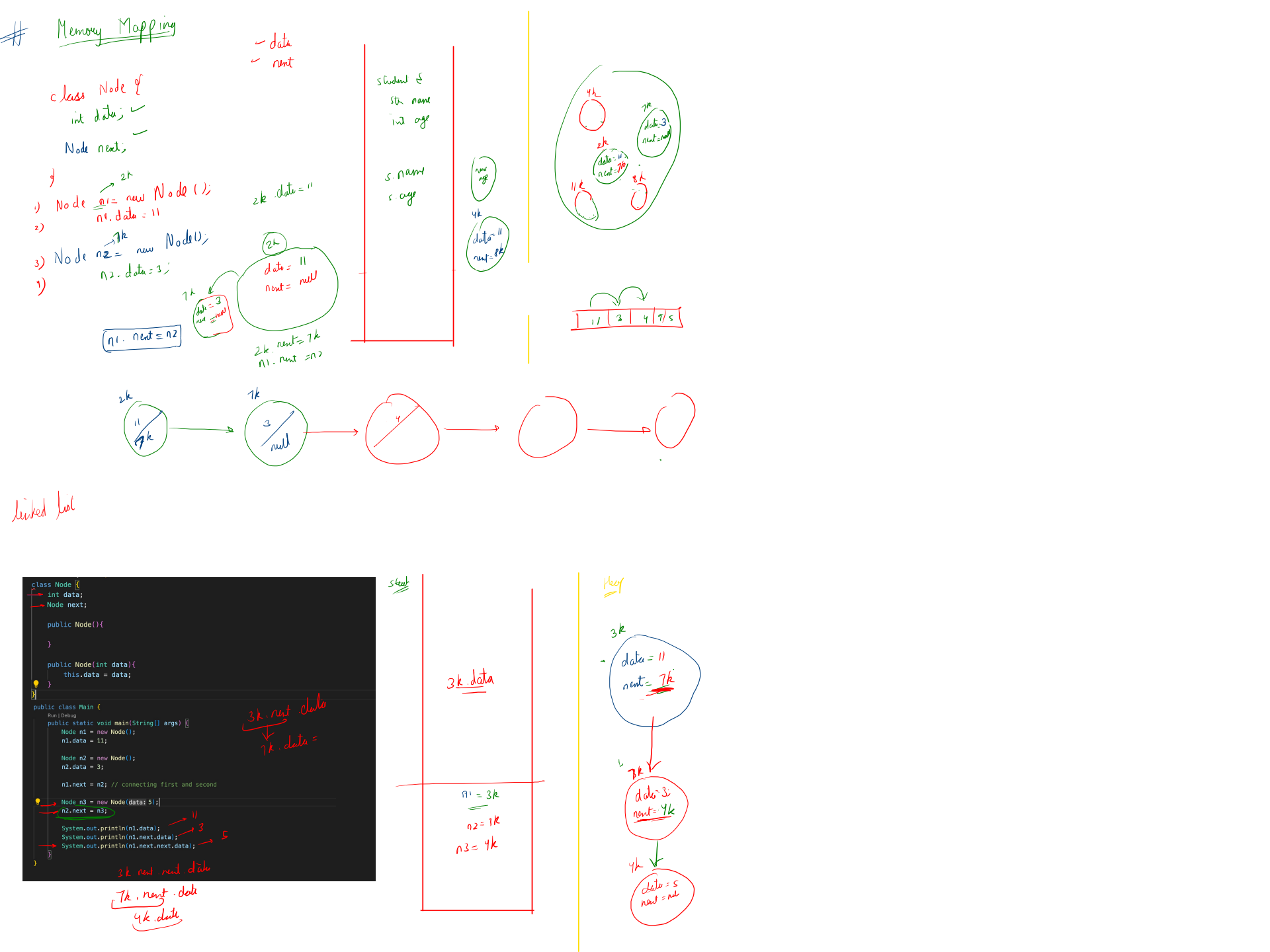
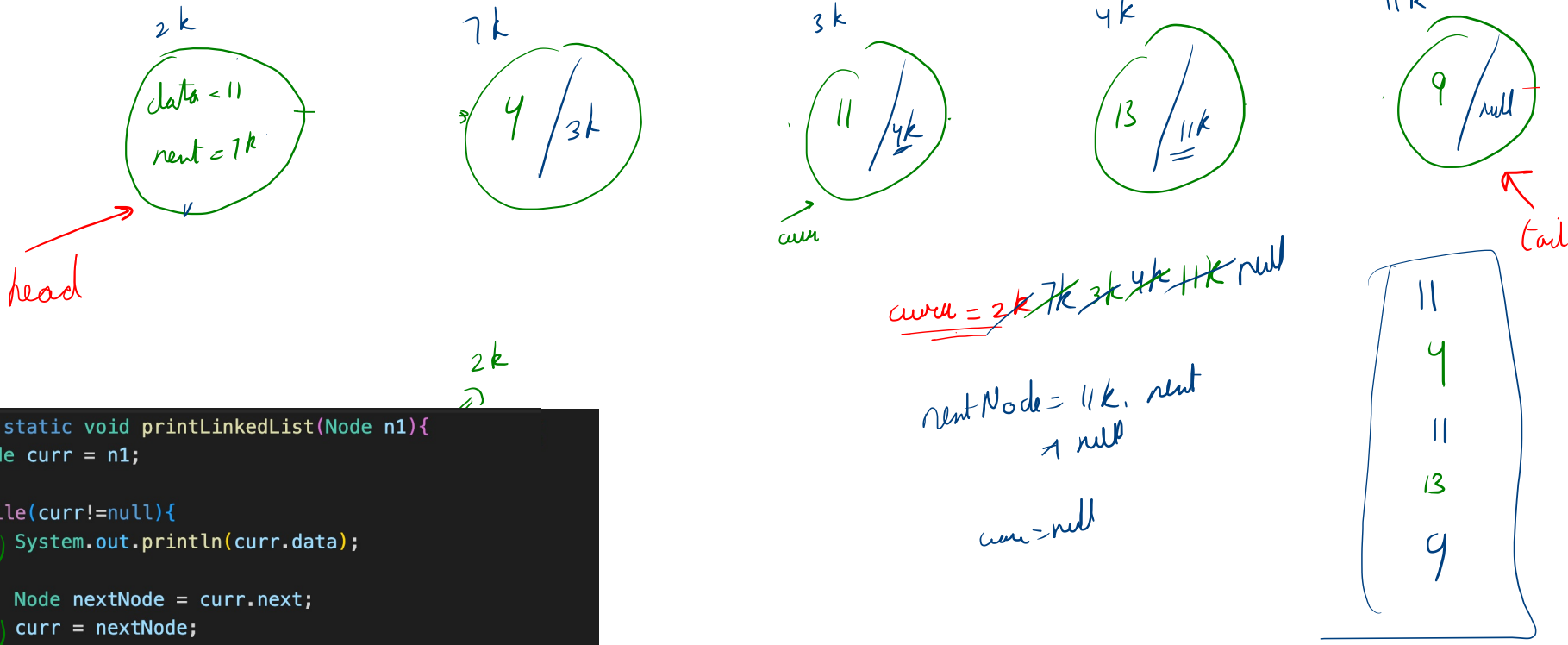
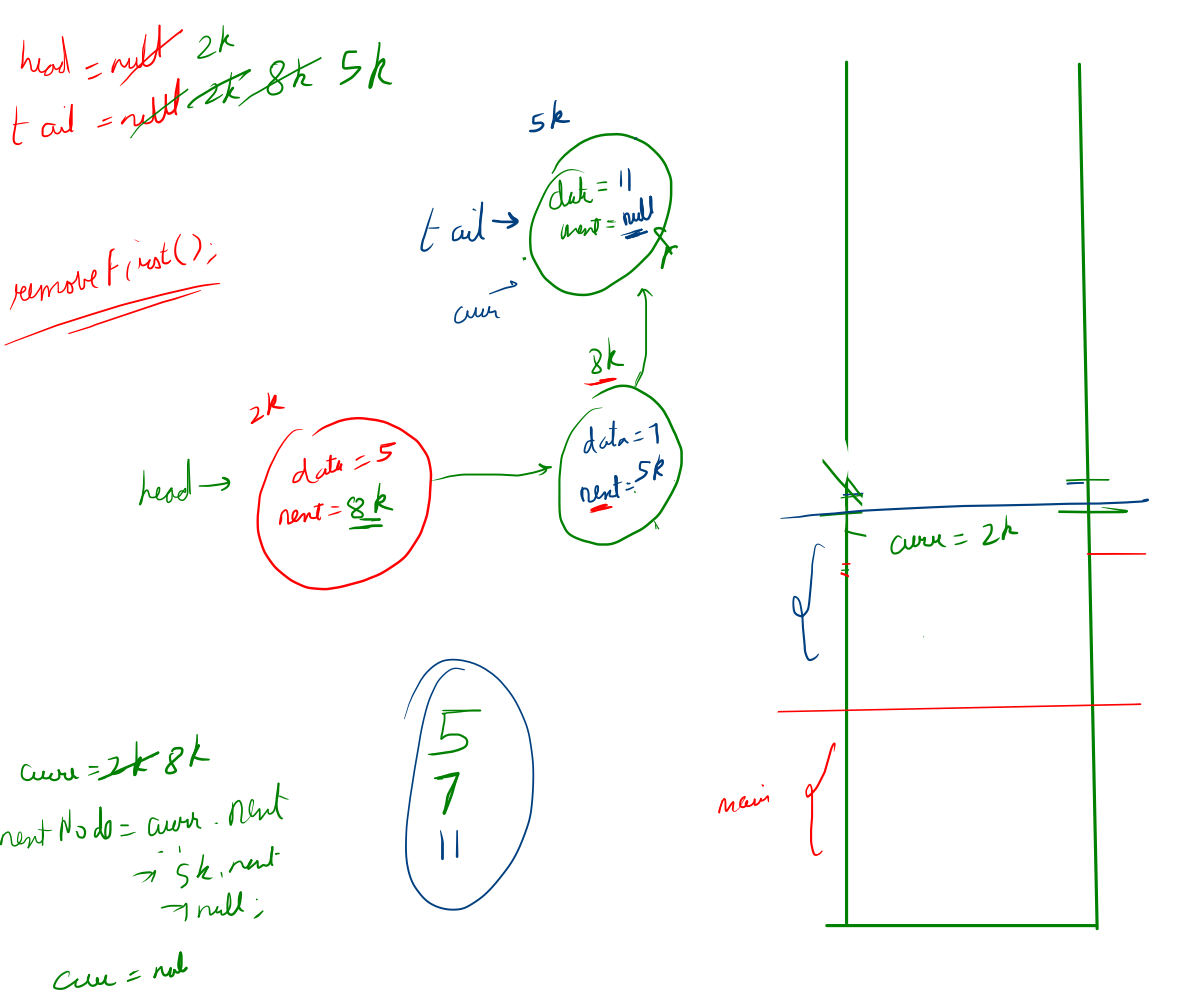
Memory Mapping




```java
class Node {
    int data;
    Node next;

    public Node(){

    }

    public Node(int data){
        this.data = data;
    }
}

public class Main {
    Run | Debug
    public static void main(String[] args) {
        Node n1 = new Node();
        n1.data = 11;

        Node n2 = new Node();
        n2.data = 3;

        n1.next = n2; // connecting first and second

        Node n3 = new Node(data: 5);
        n2.next = n3;

        System.out.println(n1.data);
        System.out.println(n1.next.data);
        System.out.println(n1.next.next.data);
    }
}
```

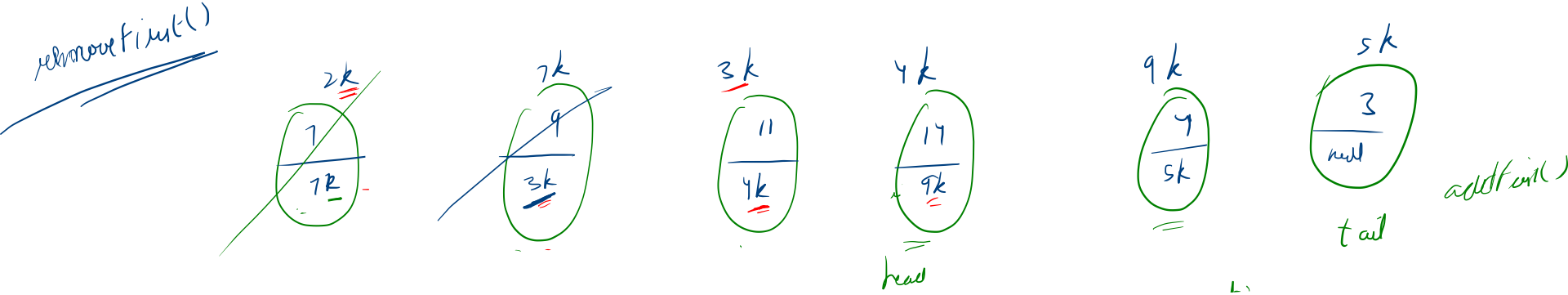



```java
public static void printLinkedList(Node n1){
    Node curr = n1;

    while(curr!=null){
        System.out.println(curr.data);

        Node nextNode = curr.next;
        curr = nextNode;
    }
}
```




```java
public class Main {
    static Node head;
    static Node tail;

    public static void addLast(int value){
        Node nn = new Node(value);

        if(head == null){
            head = nn;
            tail = nn;
        } else {
            tail.next = nn;
            tail = nn;
        }
    }

    public static void printLinkedList(Node n1){
        Node curr = n1;

        while(curr!=null){
            System.out.println(curr.data);

            Node nextNode = curr.next;
            curr = nextNode;
        }
    }
    Run | Debug
    public static void main(String[] args) {
        head = null;
        tail = null;

        addLast(value: 5);
        addLast(value: 7);
    }
}
```

(adding node at last)
(moving tail)
head
print (head)
addLast (11)




```java
public static void removeFirst(){
    if(head==null){
        System.out.println(x: "LinkedList is empty, not possible to remove first");
    } else {
        Node nextOfHead = head.next;
        head = nextOfHead;
    }
}
```

(2) addFirst() → add in front

(1) create linked list with user input