```
/**
 * 链表节点
 */
class Node {
  constructor(data: number) {
    let data: number
    let next: Node | null
    let size: number = data ? 1 : 0

    this.data = data
    // 头部指针
    this.head = this
    // 尾部指针
    this.last = this
    // 链表实际长度
    this.size = size
    this.next = null
  }

  /**
   * 链表插入元素
   * @param data 插入元素
   * @param index 插入位置
   */
  insert(data: number, index: number) {
    if (index < 0 || index > this.size) {
      throw new Error('超出链表节点范围')
    }
    let insertedNode: Node = new Node(data)
    if (this.size === 0) {
      // 空链表
      this.head = insertedNode
      this.last = insertedNode
    } else if (index === 0) {
      // 插入头部
      insertedNode.next = this.head
      this.head = insertedNode
    } else if (this.size === index) {
      // 插入尾部
      this.last.next = insertedNode
      this.last = insertedNode
    } else {
      // 插入中间
      console.log(index)
      let prevNode: Node = this.get(index - 1)
    insertedNode.next = prevNode.next
    prevNode.next = insertedNode
  }
  this.size++
}

/**
 * 链表删除元素
 * @param index 删除的位置
 * @returns 删除的元素
 */
remove(index: number): Node | null {
  if (index < 0 || index >= this.size) {
    throw new Error('超出链表节点范围')
  }
  let removedNode: Node | null = null
  if (index === 0) {
    // 删除头节点
    removedNode = this.head
    this.head = this.head.next
  } else if (index === this.size - 1) {
    // 删除尾部节点
  } else {
    // 删除中间节点
  }
  this.size--
  return removedNode
}

/**
 * 链表查找元素
 * @param index 查找的位置
 * @returns 查找的节点
 */
get(index: number):Node {
  if (index < 0 || index > this.size) {
    throw new Error('超出链表节点范围')
  }
  let temp:Node = this.head
  for (let i:number = 0; i < index; i++) {
    temp = temp.next
  }
  return temp
}

/**
 * 输出链表
```

```
   */
  output() {
    let temp: Node = this.head
    while (temp !== null) {
      console.log(temp.data)
      temp = temp.next
    }
  }

}

let myLinkedList = new Node()
myLinkedList.insert(3, 0);
myLinkedList.insert(7,1);
myLinkedList.insert(9,2);
myLinkedList.insert(5,3);
myLinkedList.insert(6,1);
myLinkedList.remove(0);
myLinkedList.output();
```