```
function Node(data) {
  this.data = data;
  this.next = null;
}

class SinglyLinkedList {
  constructor() {
    this.head = null;
    this.tail = null;
    this.numberOfValues = 0;
  }

  add(data) {
    const node = new Node(data);
    if(!this.head) {
      this.head = node;
      this.tail = node;
    } else {
      this.tail.next = node;
      this.tail = node;
    }
    this.numberOfValues++;
  }

  remove(data) {
    let previous = this.head;
    let current = this.head;
    while(current) {
      if(current.data === data) {
        if(current === this.head) {
          this.head = this.head.next;
        }
        if(current === this.tail) {
          this.tail = previous;
        }
        previous.next = current.next;
        this.numberOfValues--;
      } else {
        previous = current;
      }
      current = current.next;
    }
  }

  insertAfter(data, toNodeData) {
    let current = this.head;
    while(current) {
      if(current.data === toNodeData) {
        const node = new Node(data);
        if(current === this.tail) {
          this.tail.next = node;
          this.tail = node;
        } else {
          node.next = current.next;
          current.next = node;
        }
        this.numberOfValues++;
      }
      current = current.next;
    }
```

```
  }

  traverse(fn) {
    let current = this.head;
    while(current) {
      if(fn) {
        fn(current);
      }
      current = current.next;
    }
  }

  length() {
    return this.numberOfValues;
  }

  print() {
    let string = '';
    let current = this.head;
    while(current) {
      string += `${current.data} `;
      current = current.next;
    }
    console.log(string.trim());
  }
}

const singlyLinkedList = new SinglyLinkedList();
singlyLinkedList.print(); // => ''
singlyLinkedList.add(1);
singlyLinkedList.add(2);
singlyLinkedList.add(3);
singlyLinkedList.add(4);
singlyLinkedList.print(); // => 1 2 3 4
console.log('length is 4:', singlyLinkedList.length()); // => 4
singlyLinkedList.remove(3); // remove value
singlyLinkedList.print(); // => 1 2 4
singlyLinkedList.remove(9); // remove non existing value
singlyLinkedList.print(); // => 1 2 4
singlyLinkedList.remove(1); // remove head
singlyLinkedList.print(); // => 2 4
singlyLinkedList.remove(4); // remove tail
singlyLinkedList.print(); // => 2
console.log('length is 1:', singlyLinkedList.length()); // => 1
singlyLinkedList.add(6);
singlyLinkedList.print(); // => 2 6
singlyLinkedList.insertAfter(3, 2);
singlyLinkedList.print(); // => 2 3 6
singlyLinkedList.insertAfter(4, 3);
singlyLinkedList.print(); // => 2 3 4 6
singlyLinkedList.insertAfter(5, 9); // insertAfter a non existing node
singlyLinkedList.print(); // => 2 3 4 6
singlyLinkedList.insertAfter(5, 4);
singlyLinkedList.insertAfter(7, 6); // insertAfter the tail
singlyLinkedList.print(); // => 2 3 4 5 6 7
singlyLinkedList.add(8); // add node with normal method
singlyLinkedList.print(); // => 2 3 4 5 6 7 8
console.log('length is 7:', singlyLinkedList.length()); // => 7
singlyLinkedList.traverse(node => { node.data = node.data + 10; });
singlyLinkedList.print(); // => 12 13 14 15 16 17 18
```

```
121 singlyLinkedList.traverse(node => { console.log(node.data); }); // => 12 13 14 15 16
    17 18
122 console.log('length is 7:', singlyLinkedList.length()); // => 7
123
```