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

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

SinglyLinkedList.prototype.add = function(data) {
  var node = new Node(data);
  if(!this.head) {
    this.head = node;
    this.tail = node;
  } else {
    this.tail.next = node;
    this.tail = node;
  }
  this.numberOfValues++;
};
SinglyLinkedList.prototype.remove = function(data) {
  var previous = this.head;
  var 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;
  }
};
SinglyLinkedList.prototype.insertAfter = function(data, toNodeData) {
  var current = this.head;
  while(current) {
    if(current.data === toNodeData) {
      var 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;
  }
};
SinglyLinkedList.prototype.traverse = function(fn) {
  var current = this.head;
```

```
  while(current) {
    if(fn) {
      fn(current);
    }
    current = current.next;
  }
};
SinglyLinkedList.prototype.length = function() {
  return this.numberOfValues;
};
SinglyLinkedList.prototype.print = function() {
  var string = '';
  var current = this.head;
  while(current) {
    string += current.data + ' ';
    current = current.next;
  }
  console.log(string.trim());
};

var 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(function(node) { node.data = node.data + 10; });
singlyLinkedList.print(); // => 12 13 14 15 16 17 18
singlyLinkedList.traverse(function(node) { console.log(node.data); }); // => 12 13 14
15 16 17 18
console.log('length is 7:', singlyLinkedList.length()); // => 7

```