function LinkedList() {

this.head = null;

this.tail = null;

}

function Node(value, next, prev) {

this.value = value;

this.next = next;

this.prev = prev;

}

LinkedList.prototype.addToHead = function(value) {

var newNode = new Node(value, this.head, null);

if (this.head) this.head.prev = newNode;

else this.tail = newNode;

this.head = newNode;

};

LinkedList.prototype.addToTail = function(value) {

var newNode = new Node(value, null, this.tail);

if (this.tail) this.tail.next = newNode;

else this.head = newNode;

this.tail = newNode;

};

LinkedList.prototype.removeHead = function() {

if (!this.head) return null;

var val = this.head.value;

this.head = this.head.next;

if (this.head) this.head.prev = null;

else this.tail = null;

return val;

};

LinkedList.prototype.removeTail = function() {

if (!this.tail) return null;

var val = this.tail.value;

this.tail = this.tail.prev;

if (this.tail) this.tail.next = null;

else this.head = null;

return val;

};

LinkedList.prototype.search = function(searchValue) {

var currentNode = this.head;

while (currentNode) {

if (currentNode.value === searchValue) return currentNode.value;

currentNode = currentNode.next;

}

return null;

};

LinkedList.prototype.indexOf = function(value) {

var indexes = [];

var currentIndex = 0;

var currentNode = this.head;

while(currentNode) {

if (currentNode.value === value) indexes.push(currentIndex);

currentNode = currentNode.next;

currentIndex++;

}

return indexes;

};

var myLL = new LinkedList();

myLL.addToHead(123);

myLL.addToHead(70);

myLL.addToHead('hello');

myLL.addToTail(19);

myLL.addToTail('world');

myLL.addToTail(20);