```
class ListNode {
  constructor(val, next){
    this.val = val == undefined ? 0 : val;
    this.next = next == undefined ? null : next;
  }
}

var MyLinkedList = function() {
 this.head = null;
};

MyLinkedList.prototype.get = function(index) {
    let curr = this.head;
  let len =this.getLength();
  if(index<0 || index >= len) return -1;
  for(let i =0; i < index; i++){
    curr = curr.next;
  }
  console.log('gett', curr.val);
  return curr.val;
};

MyLinkedList.prototype.addAtHead = function(val) {
  let node = new ListNode(val);
  node.next = this.head;
  this.head = node;
  console.log('headd', this.head)
};

MyLinkedList.prototype.addAtTail = function(val) {
 if (this.head == null){
    this.addAtHead(val);
    return;
 }
  let curr = this.head;
  let node = new ListNode(val);
  while (curr.next !== null){
    curr = curr.next;
 }
  curr.next = node;
  console.log('tailll', this.head);
};

MyLinkedList.prototype.addAtIndex = function(index, val) {
```

```javascript
  let curr = this.head;
    if (index === 0) {
      this.addAtHead(val);
      return;
    }
    if (index === this.getLength()) {
      this.addAtTail(val);
      return;
    }
  if(index >= this.getLength()) return;
  for(let i =0; i < index-1; i++){
    curr = curr.next;
  }
  let node = new ListNode(val);
  let temp = curr.next;
  curr.next = node;
  node.next = temp;
  console.log('added', this.head);
};

/**
 * Delete the index-th node in the linked list, if the index is valid.
 * @param {number} index
 * @return {void}
 */
MyLinkedList.prototype.deleteAtIndex = function(index) {

};

MyLinkedList.prototype.getLength = function() {
  let  len=0, curr = this.head;
  while(curr !== null) {
    curr = curr.next;
    len++;
  }
  console.log('lennn', len)
  return len;
};


var obj = new MyLinkedList()
obj.addAtHead(1);
obj.addAtTail(2);
obj.getLength();
obj.get(2);
obj.addAtIndex(1,4)
```

```
/**
 * Your MyLinkedList object will be instantiated and called as such:
 * var obj = new MyLinkedList()
 * var param_1 = obj.get(index)
 * obj.addAtHead(val)
 * obj.addAtTail(val)
 * obj.addAtIndex(index,val)
 * obj.deleteAtIndex(index)
 */
```