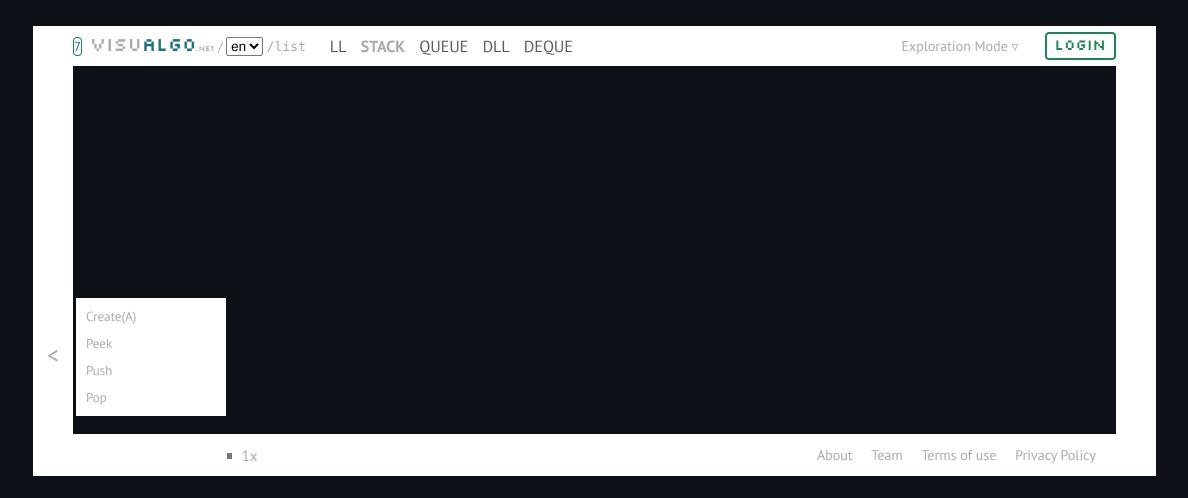
# Estructura de datos y algoritmos

Rodrigo Alvarez

rodrigo.alvarez2@mail.udp.cl

#### Stack



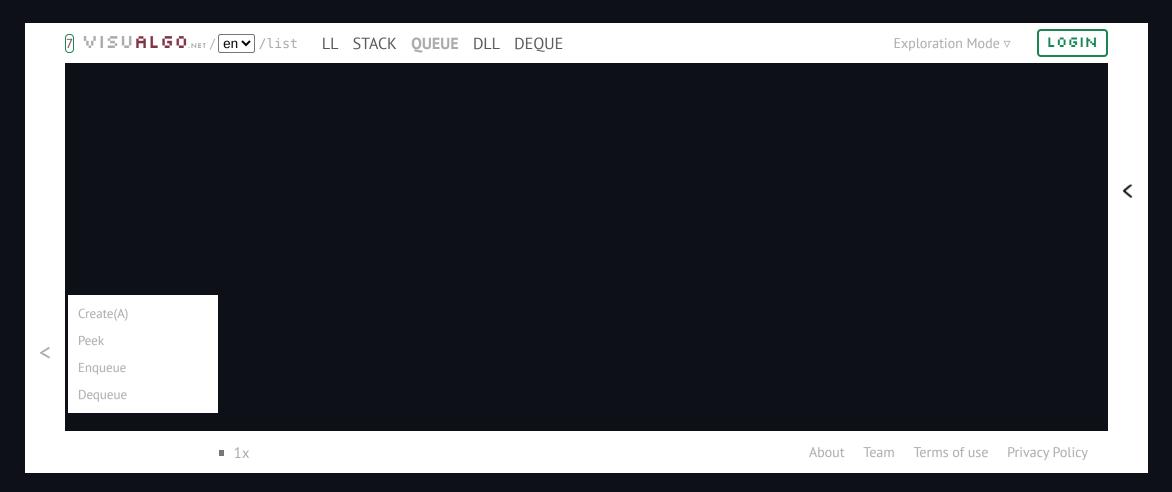
2

```
class Stack {
 class Node {
   int data;
   Node next;
   public Node(int data) {
     this.data = data;
     this.next = null;
 Node head;
 public Stack() { this.head = null; }
 public void push(int e) {
   Node newNode = new Node(e);
   if (head != null) {
     newNode.next = head;
   head = newNode;
 public int pop() {
   int e = head.data;
   head = head.next;
   return e;
 public int peek() { return head.data; }
```

### Complejidad:

- push : O(1)
- pop : O(1)
- peek : O(1)

### Queue



```
class Queue {
 class Node {
   int data;
   Node next;
   public Node(int data) {
     this.data = data;
     this.next = null;
Node head, tail;
 public Queue() {
   this.head = this.tail = null;
 public void enqueue(int e) {
  Node newNode = new Node(e);
  if (head == null) {
     head = tail = newNode;
  } else {
     tail.next = newNode;
     tail = newNode;
 public int dequeue() {
  int e = head.data;
  head = head.next;
   return e;
 public int peek() {
   return head.data;
```

## Complejidad:

- enqueue : O(1)
- dequeue : O(1)
- peek : O(1)

- Stack using two queues
- Balanced brackets
- Reverse a doubly linked list
- Add two numbers
- Merge two sorted lists