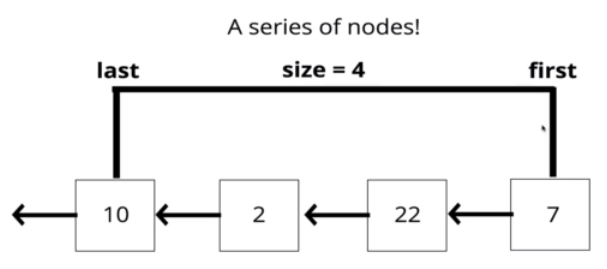
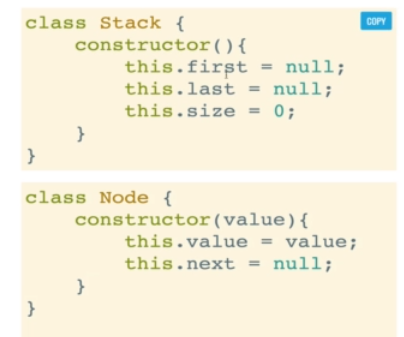
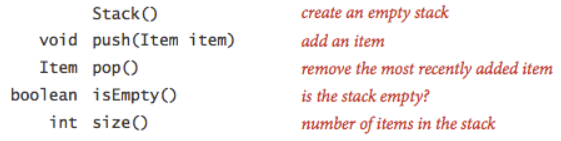
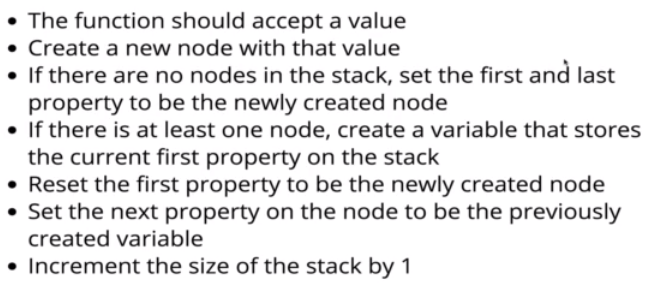
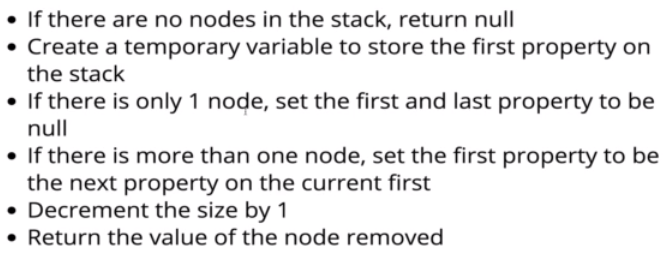
JavaScript Algorithms and Data Structures Masterclass

# Section 21: Stacks and Queues

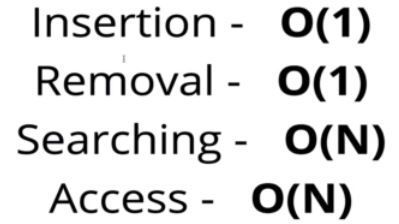
# Stacks:

* What is a Stack?
  + Last In First Out (**LIFO**) Data Structure (singly linked list)
    - Last element added will be first removed
  + **Diagram**:
    - 
  + **Examples**:
    - Recursion Call Stack
    - Manage function invocation
    - Undo/Redo
    - Routing (history object) in React is a stack
* You can use Array to implement a stack
  + This is not efficient since unshift() causes ALL indices to be changed per item
* Better to implement a stack with Singly Linked List

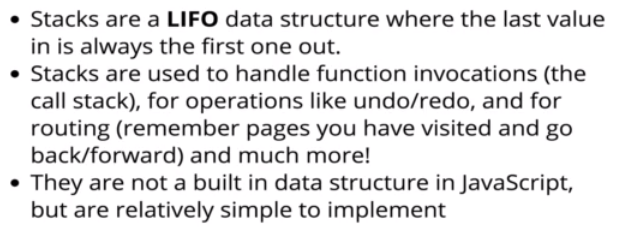
### Implementation

* Stack Class
  + 
* API for Stack
  + 
* **PUSHING**()
  + Add a value to the top of the stack (the ‘first’ node) -> pushes into the ‘front’/left of the stack
  + **Pseudo-code**:
    - 
* **POP**()
  + Pop off the end (Last one in, first one out)
  + **Psuedo-code**:
    - 

### Big O of Stacks

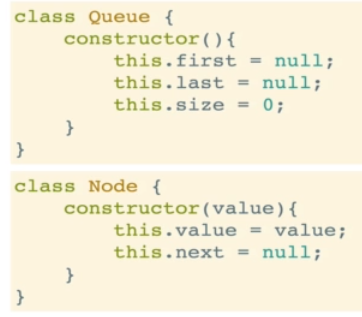
* Stacks prioritize insertion and removal
  + 

### Recap

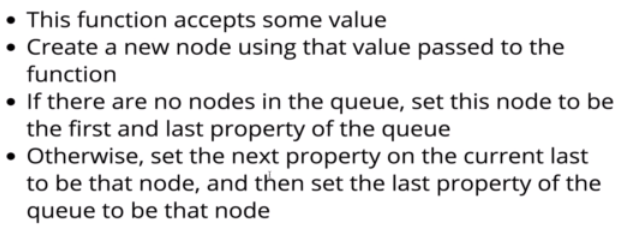
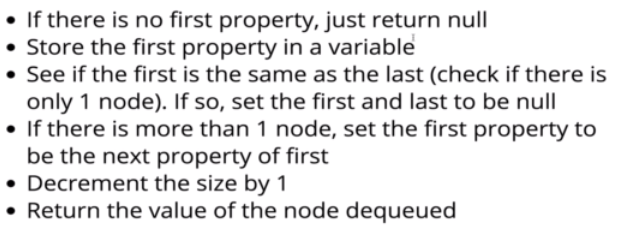
* 

# Queues:

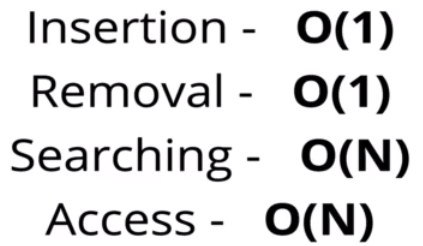
## What is a Queue?

* First In First Out (**FIFO**) -> a line up
  + **Examples**:
    - Background tasks
    - Printing queue
* **Queue Class**, but can also implement with array (not recommended)
  + 

## Implementation

* **Enqueue**()
  + Add a node at the end
    - 
* **Dequeue**()
  + Remove node from beginning of the line
    - 

## Big O of Queues

* 

### Recap:

* 