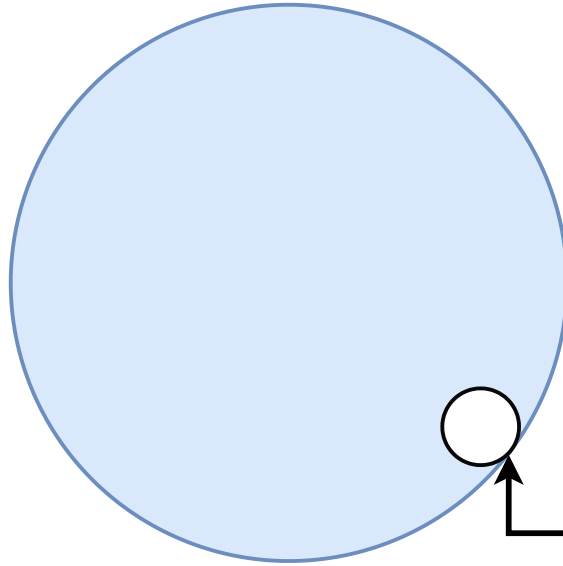


## Data Structures

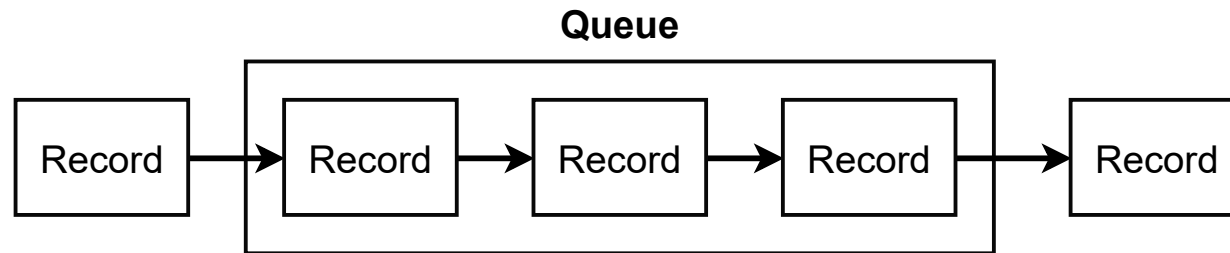
Ways of organizing information with optimal 'runtime complexity' for adding or removing records

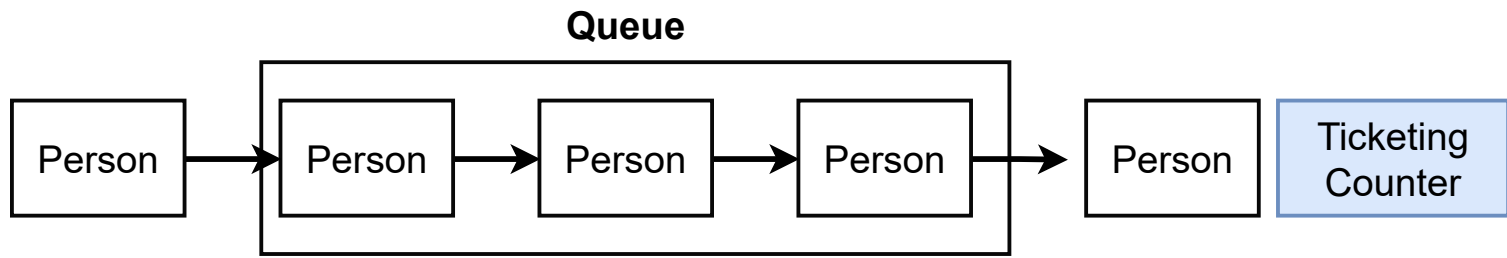
Javascript natively implements several data structures. **You will still be asked about 'inferior' data structures.**

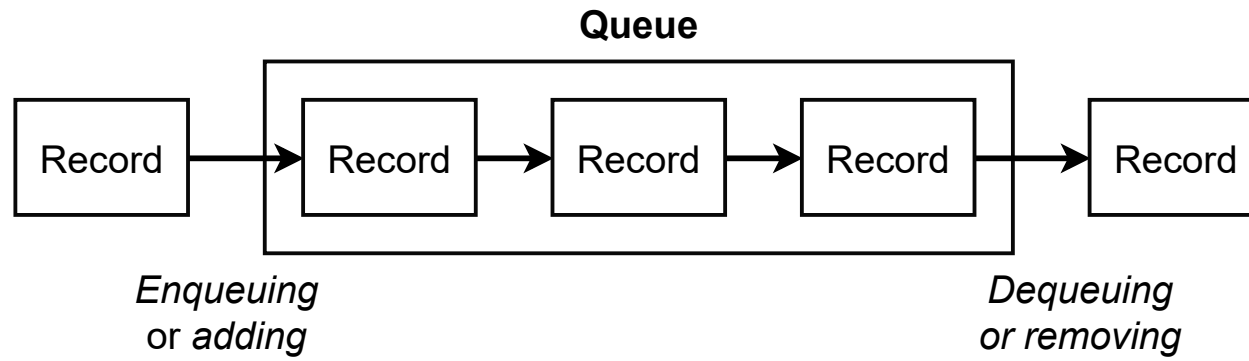
Functionality of  
a JS Array



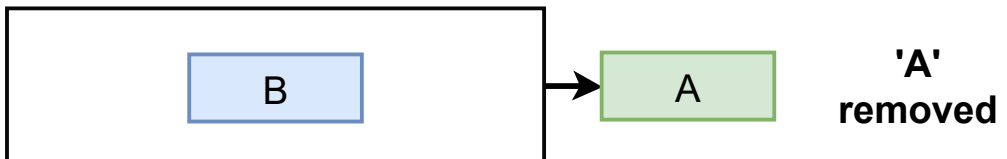
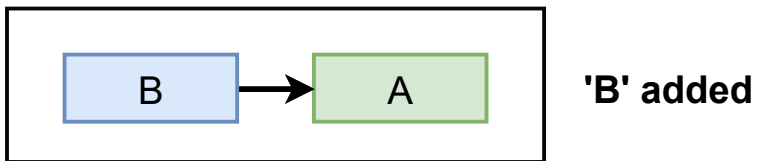
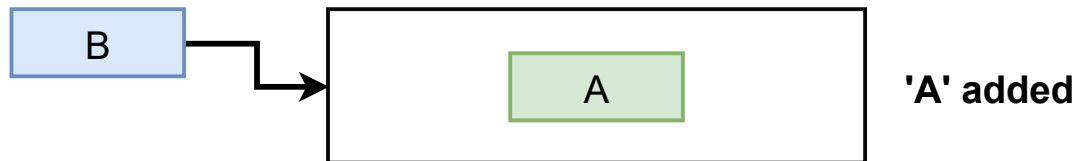
What a  
queue does





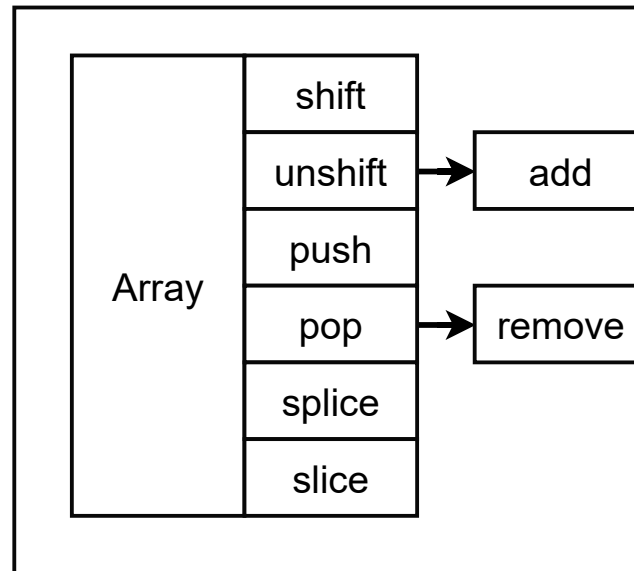


**First In First Out  
F-I-F-O**



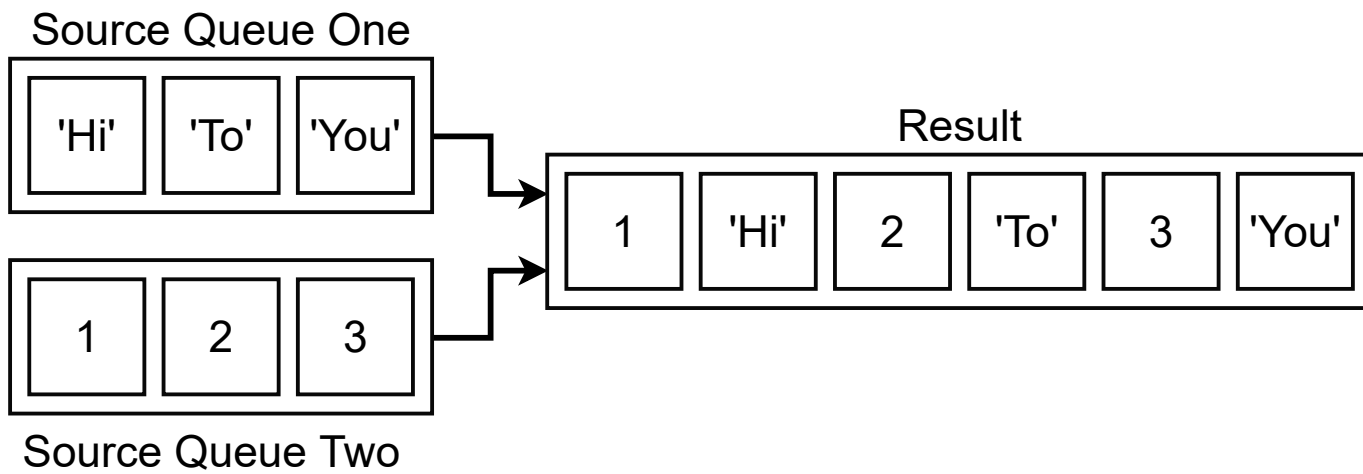
Queue	Array Equivalent
Add to queue	array.unshift();
Remove from queue	array.pop();

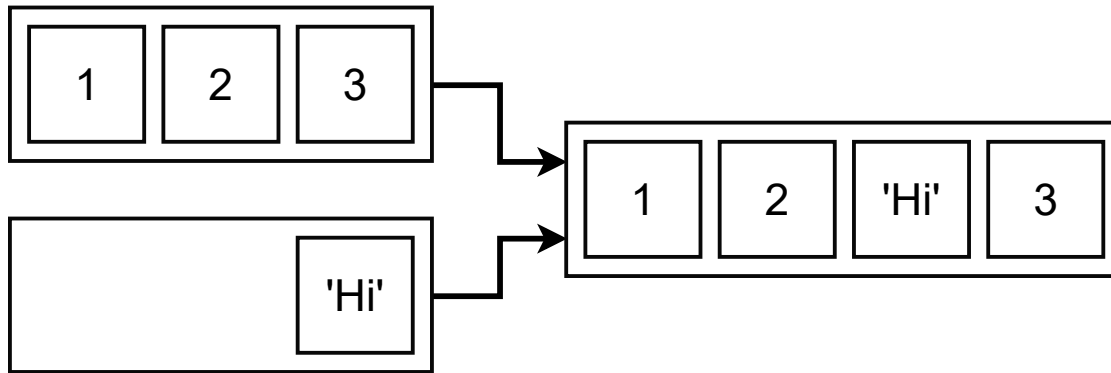
## Queue Class

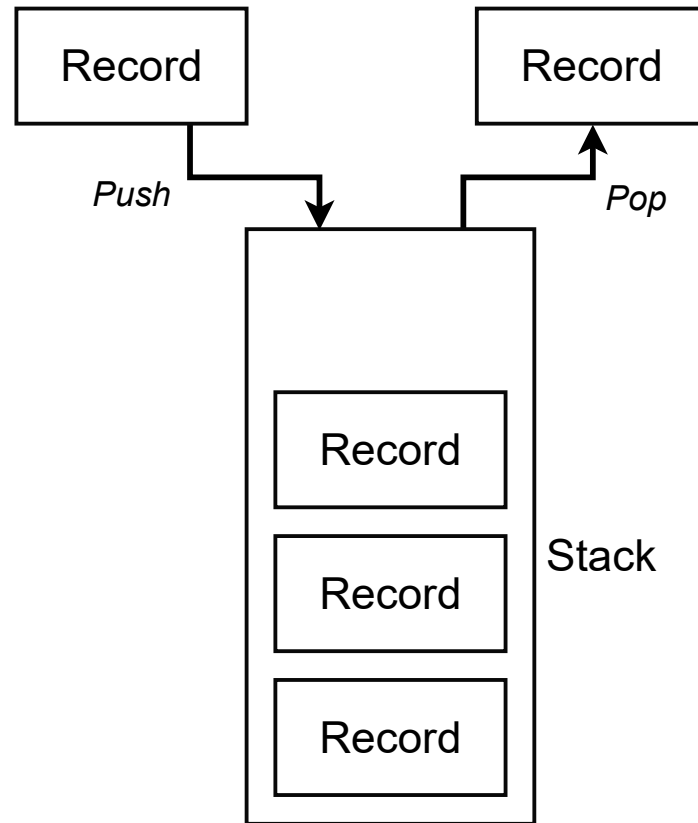


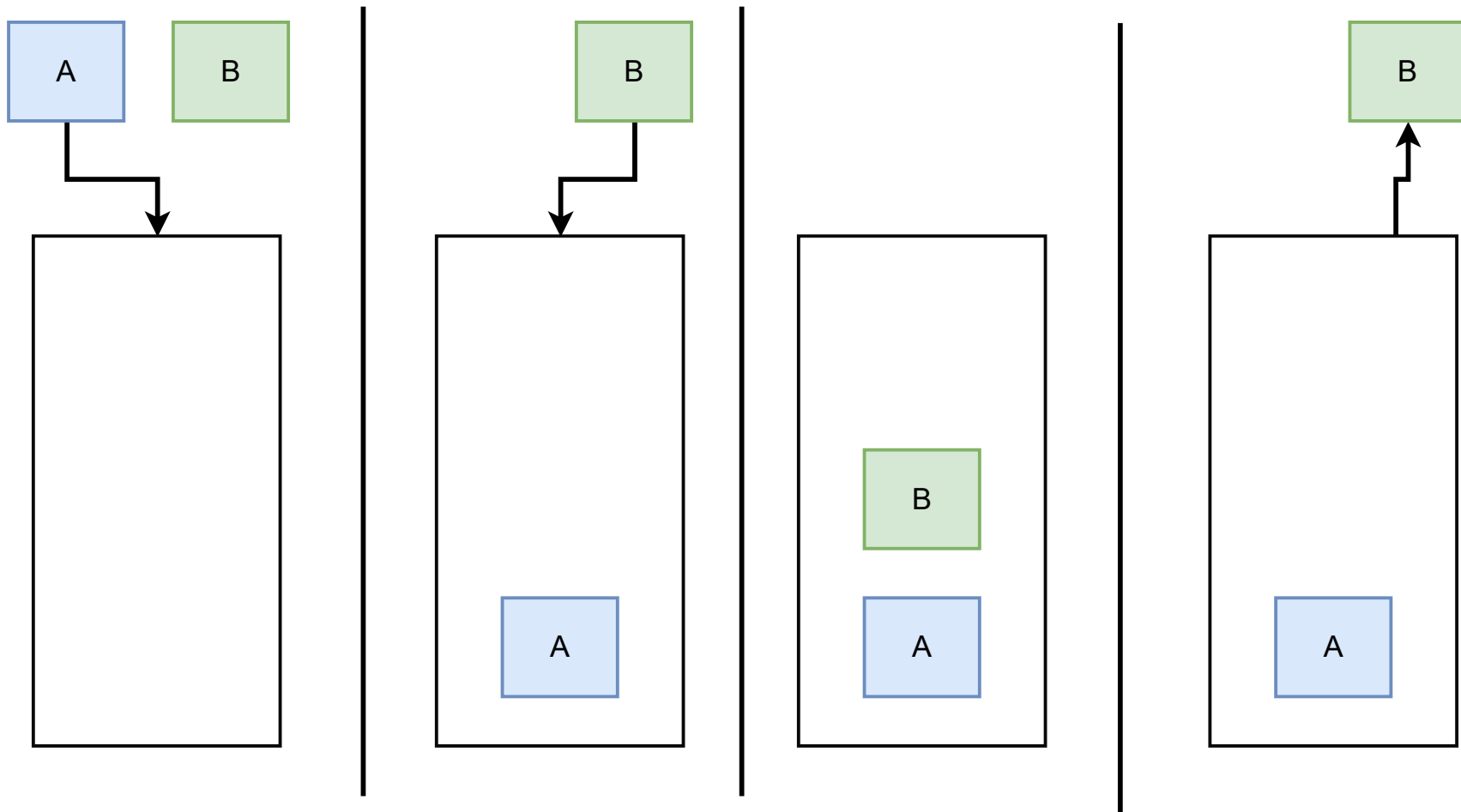


To...	Run This
Create a new, empty queue	<code>const q = new Queue();</code>
Add a record to a queue	<code>q.add(1);</code>
Remove record at the end of a queue	<code>q.remove();</code>



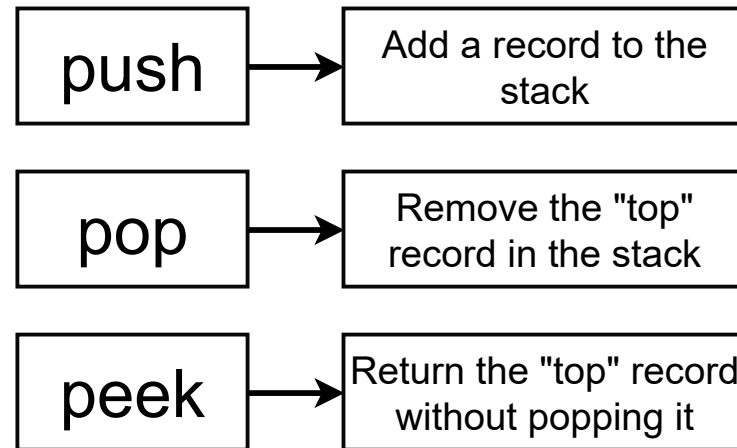


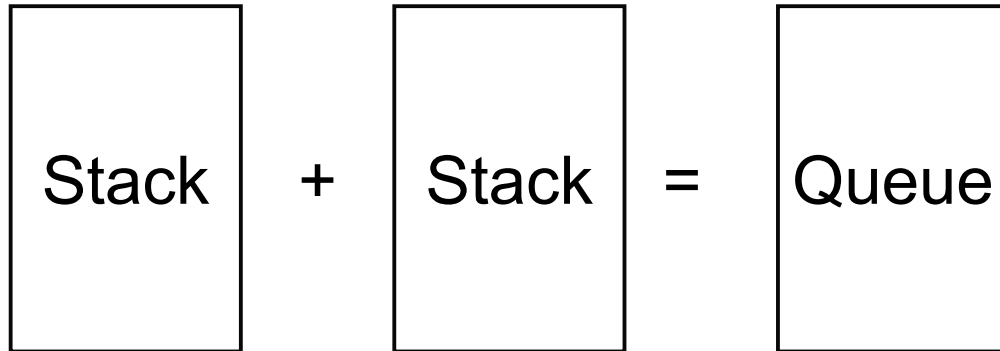




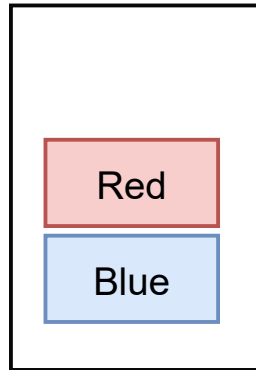
**First In Last Out**  
**FILO**

### Stack Methods





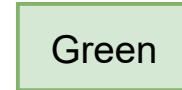
**Add**



Stack A



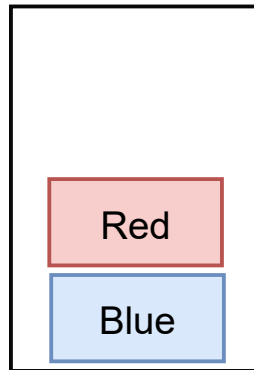
Stack B



*Remember, we want a Queue, so follow  
'First In First Out'*



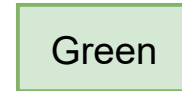
**Remove**



Stack A

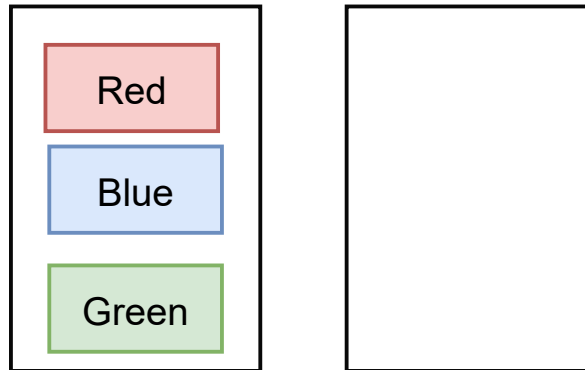


Stack B



*Remember, we want a Queue, so follow  
'First In First Out'*

**Peek**



Stack A

Stack B

*Remember, we want a Queue, so follow  
'First In First Out'*