**Deadline: 8th Nov 2022**

[Implement Stack using Queues(Amazon and Microsoft)](https://leetcode.com/problems/implement-stack-using-queues/)

Implement a LIFO stack using only two queues. The implemented stack should support all the functions of a usual stack (push, pop, top, empty)

Implement MyStack class:

* void push(int x) Pushes element x to the top of the stack.
* int pop() Removes the element on the top of the stack and returns it.
* int top() Returns the element on the top of the stack.
* boolean empty() Returns true if the stack is empty, false otherwise.

Input

["MyStack", "push", "push", "top", "pop", "empty"]

[[], [1], [2], [], [], []]

Output

[null, null, null, 2, 2, false]

Explanation

MyStack myStack = new MyStack();

myStack.push(1);

myStack.push(2);

myStack.top(); // return 2

myStack.pop(); // return 2

myStack.empty(); // return False

[Implement Queue using Stack](https://leetcode.com/problems/implement-queue-using-stacks/) (Amazon)

Implement a first in first out (FIFO) queue using only two stacks. The implemented queue should support a normal queue's functions (push, peek, pop, and empty).

Implement the MyQueue class:

* void push(int x) Pushes element x to the back of the queue.
* int pop() Removes the element from the front of the queue and returns it.
* int peek() Returns the element at the front of the queue.
* boolean empty() Returns true if the queue is empty, false otherwise.

Input

["MyQueue", "push", "push", "peek", "pop", "empty"]

[[], [1], [2], [], [], []]

Output

[null, null, null, 1, 1, false]

Explanation

MyQueue myQueue = new MyQueue();

myQueue.push(1); // queue is: [1]

myQueue.push(2); // queue is: [1, 2] (leftmost in front of the queue)

myQueue.peek(); // return 1

myQueue.pop(); // return 1, the queue is [2]

myQueue.empty(); // return false