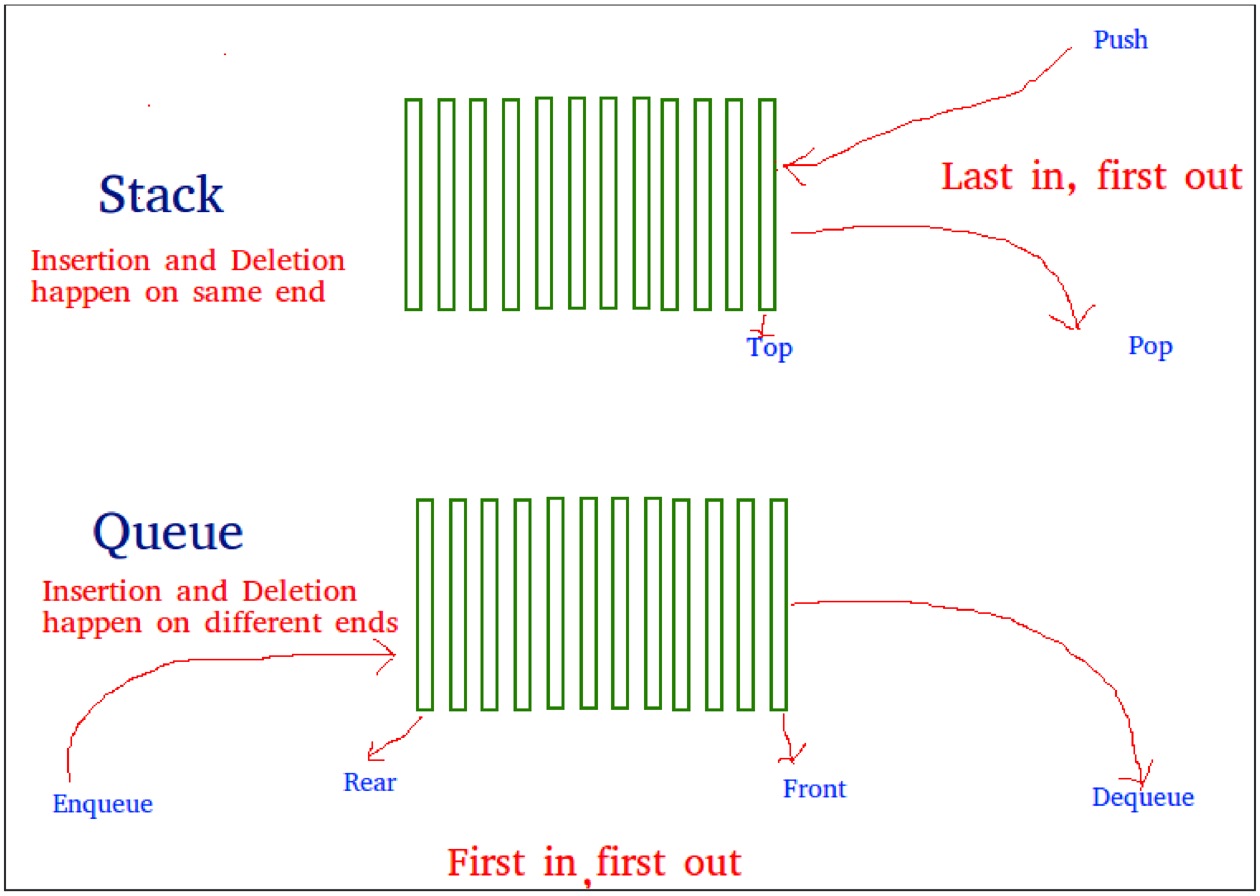
Implementing Queues using Stacks


Push
Stack
Insertion and Deletion happen on same end
Last in, first out
Top
Pop
Queue
Insertion and Deletion happen on different ends
Enqueue
Rear
Front
Dequeue
First in, first out

Question:

https://leetcode.com/problems/implement-queue-using-stacks/

Implement a first in first out (FIFO) queue using only two stacks. The implemented queue should support all the functions of a normal queue (`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.*

Notes:

- *You must use only standard operations of a stack, which means only push to top, peek/pop from top, size, and is empty operations are valid.*
- *Depending on your language, the stack may not be supported natively. You may simulate a stack using a list or deque (double-ended queue) as long as you use only a stack's standard operations.*

The one approach of implementing queues using stacks is to use 2 stacks.

NOTE: We can use a single stack and implement a queue but the reason we preferably use 2 stacks is to separate the read and write of a queue in multi-processing. One stack is for reading and the 2nd stack is for writing. They only interfere with each other only when the former is full or the latter is empty.

Here is a detailed Stack Overflow explanation:
https://stackoverflow.com/questions/2050120/why-use-two-stacks-to-make-a-queue/2050402#2050402

Approach:

- Elements in the first stack follow FIFO Order while the elements in the second stack follow LIFO Order.
- Enqueue to the queue by prepending to the second stack.
- De queue from the queue by taking the first element of the first stack.
- If the first list is empty: reverse the second list and replace the first list with it, and replace the second list with an empty list

Solution:

https://gist.github.com/vermaayush680/f2b7f3db234498bd2c957fe28c033579

```
class Queue(object):
    def __init__(self):
        self.inStack, self.outStack = [], []

    def push(self, x):
        self.inStack.append(x)

    def pop(self):
        self.move()
        self.outStack.pop()

    def peek(self):
        self.move()
        return self.outStack[-1]

    def empty(self):
        return (not self.inStack) and (not self.outStack)

    def move(self):
        if not self.outStack:
            while self.inStack:
                self.outStack.append(self.inStack.pop())
```

I could only find 1 approach but after going through GeeksForGeeks, I got 2 more approaches.

Link for GFG approaches: https://www.geeksforgeeks.org/queue-using-stacks/