

队列

@M了个J

<https://github.com/CoderMJLee>

<http://cnblogs.com/mjios>

码拉松



实力IT教育 www.520it.com

队列 (Queue)

■ 队列是一种特殊的线性表，只能在**头尾两端**进行操作

□ 队尾 (rear)：只能从**队尾添加**元素，一般叫做**enQueue**，**入队**

□ 队头 (front)：只能从**队头移除**元素，一般叫做**deQueue**，**出队**

□ 先进先出的原则，First In First Out, FIFO

队尾 (rear)



队头 (front)



队列的接口设计

- `int size();` // 元素的数量
- `boolean isEmpty();` // 是否为空
- `void enqueue(E element);` // 入队
- `E dequeue();` // 出队
- `E front();` // 获取队列的头元素

■ 思考

□ 队列的内部实现是否可以直接利用以前学过的数据结构?

✓ 动态数组

✓ 链表

队尾 (rear)



队头 (front)

练习 – 用栈实现队列

■ <https://leetcode-cn.com/problems/implement-queue-using-stacks/>

■ 准备2个栈: inStack、outStack

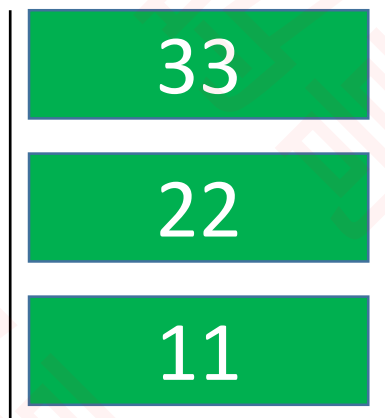
□ 入队时, push到inStack中

□ 出队时

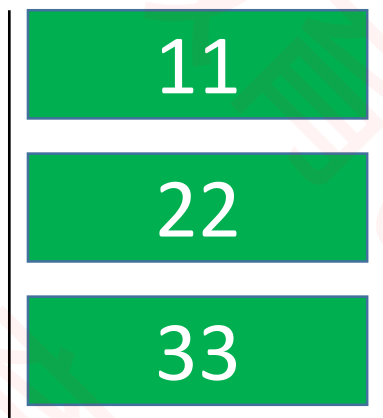
✓ 如果outStack为空, 将inStack所有元素逐一弹出, push到outStack, outStack弹出栈顶元素

✓ 如果outStack不为空, outStack弹出栈顶元素

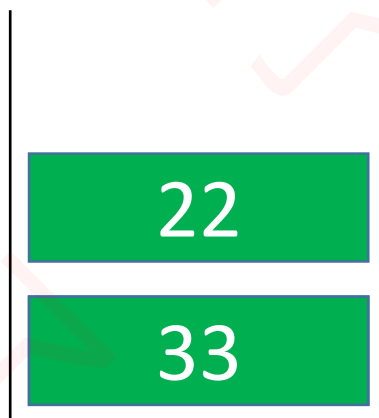
inStack



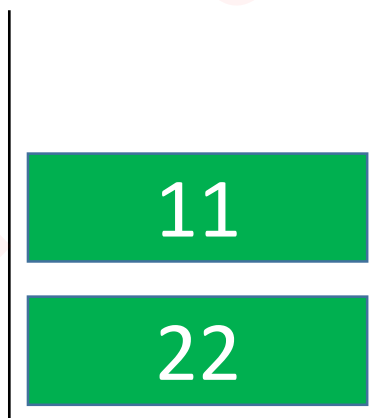
outStack



inStack



outStack



双端队列 (Deque)

■ 双端队列是能在头尾两端添加、删除的队列

□ 英文deque是double ended queue的简称

队尾 (rear)

44

33

22

11

队头 (front)

■ `int size();` // 元素的数量

■ `boolean isEmpty();` // 是否为空

■ `void enqueueRear(E element);` // 从队尾入队

■ `E dequeueFront();` // 从队头出队

■ `void enqueueFront(E element);` // 从队头入队

■ `E dequeueRear();` // 从队尾出队

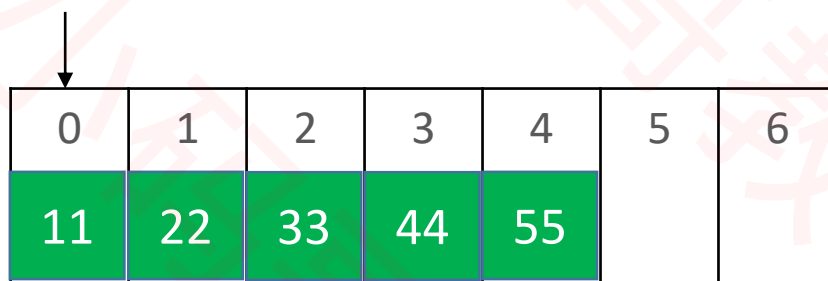
■ `E front();` // 获取队列的头元素

■ `E rear();` // 获取队列的尾元素

循环队列

■ 循环队列底层用数组实现

队头 (front)



0	1	2	3	4	5	6
11	22	33	44	55		

■ 循环双端队列：可以进行两端添加、删除操作的循环队列

作业 – 用队列实现栈

- <https://leetcode-cn.com/problems/implement-stack-using-queues/>