#LeetCode225

https://leetcode.com/problems/implement-stack-using-queues/description/

Yifeng Zeng

#题目描述

Implement Stack using Queues

#思路报告

拿到这个题，先分析Stack和Queue的性质，stack是FILO，queue是FIFO，那么只放一个数（假设为1）进queue和进stack其实没有区别。那么再放第二个数（假设为2）的时候，先进的那个数1其实需要放到现在放的这个数2的后面，那么下一次从queue里面拿出来就可以先拿出2了。所以做push操作的时候我们用另外一个queue（help）把要push的数（前面提到的2）先暂存一下，然后把queue（真正的data）里面的数全部放入help queue，这样的顺序就是FILO了，再把help queue里面的数放回data queue以便下一次操作。那么做pop，top的时候只需要做data queue的poll，peek操作了。

代码如下

```
class MyStack {

    Deque<Integer> dataQ;
    Deque<Integer> helpQ;
    /** Initialize your data structure here. */
    public MyStack() {
        dataQ = new LinkedList<>();
        helpQ = new LinkedList<>();
    }

    /** Push element x onto stack. */
    public void push(int x) {
        helpQ.offer(x);
        while (!dataQ.isEmpty()) {
            helpQ.offer(dataQ.poll());
        }
        while (!helpQ.isEmpty()) {
            dataQ.offer(helpQ.poll());
```

```
            }
        }

        /** Removes the element on top of the stack and returns that element. */
        public int pop() {
            return dataQ.poll();
        }

        /** Get the top element. */
        public int top() {
            return dataQ.peek();
        }

        /** Returns whether the stack is empty. */
        public boolean empty() {
            return dataQ.isEmpty();
        }
    }
```

#套路总结

- 这个貌似没感觉到有什么套路，既然题目是用queue implement stack，而一个queue明显不够用，就用两个queue，在想想stack queue操作顺序的不同就可以解出来了