0225. 用队列实现栈

ITCharge 大约 2 分钟

- 标签：栈、设计、队列
- 难度：简单

题目链接

- 0225. 用队列实现栈 - 力扣

题目大意

要求：仅使用两个队列实现一个后入先出（LIFO）的栈，并支持普通栈的四种操作：`push` 、 `top` 、 `pop` 和 `empty` 。

要求实现 `MyStack` 类：

- `void push(int x)` 将元素 `x` 压入栈顶。
- `int pop()` 移除并返回栈顶元素。
- `int top()` 返回栈顶元素。
- `boolean empty()` 如果栈是空的，返回 `True` ；否则，返回 `False` 。

说明：

- 只能使用队列的基本操作 —— 也就是 `push to back` 、 `peek/pop from front` 、 `size` 和 `is empty` 这些操作。
- 所使用的语言也许不支持队列。 你可以使用 `list` （列表）或者 `deque` （双端队列）来模拟一个队列 , 只要是标准的队列操作即可。

示例：

- 示例 1：

```py
输入：
["MyStack", "push", "push", "top", "pop", "empty"]
[[], [1], [2], [], [], []]
输出：
[null, null, null, 2, 2, false]
```

```
解释：
MyStack myStack = new MyStack();
myStack.push(1);
myStack.push(2);
myStack.top(); // 返回 2
myStack.pop(); // 返回 2
myStack.empty(); // 返回 False
```

解题思路

思路 1：双队列

使用两个队列。 pushQueue 用作入栈， popQueue 用作出栈。

- push 操作：将新加入的元素压入 pushQueue 队列中，并且将之前保存在 popQueue 队列中的元素从队头开始依次压入 pushQueue 中，此时 pushQueue 队列中头节点存放的是新加入的元素，尾部存放的是之前的元素。 而 popQueue 则为空。再将 pushQueue 和 popQueue 相互交换，保持 pushQueue 为空， popQueue 则用于 pop 、 top 等操作。
- pop 操作：直接将 popQueue 队头元素取出。
- top 操作：返回 popQueue 队头
- empty ：判断 popQueue 是否为空。

思路 1：代码

py

```
class MyStack:

    def __init__(self):
        """
        Initialize your data structure here.
        """
        self.pushQueue = collections.deque()
        self.popQueue = collections.deque()


    def push(self, x: int) -> None:
        """
        Push element x onto stack.
        """
```

```
        self.pushQueue.append(x)
        while self.popQueue:
            self.pushQueue.append(self.popQueue.popleft())
        self.pushQueue, self.popQueue = self.popQueue, self.pushQueue

    def pop(self) -> int:
        """
        Removes the element on top of the stack and returns that element.
        """
        return self.popQueue.popleft()


    def top(self) -> int:
        """
        Get the top element.
        """
        return self.popQueue[0]


    def empty(self) -> bool:
        """
        Returns whether the stack is empty.
        """
        return not self.popQueue


# Your MyStack object will be instantiated and called as such:
# obj = MyStack()
# obj.push(x)
# param_2 = obj.pop()
# param_3 = obj.top()
# param_4 = obj.empty()
```

思路 1：复杂度分析

- **时间复杂度**：入栈操作的时间复杂度为 $O(n)$。出栈、取栈顶元素、判断栈是否为空的时间复杂度为 $O(1)$。
- **空间复杂度**：$O(n)$。