225 Implement Stack using Queues

2018年4月5日 15:28

Question:

Implement the following operations of a stack using queues.

- push(x) -- Push element x onto stack.
- pop() -- Removes the element on top of the stack.
- top() -- Get the top element.
- empty() -- Return whether the stack is empty.

Notes:

- You must use *only* standard operations of a queue -- which means only push to back, peek/pop from front, size, and is empty operations are valid.
- Depending on your language, queue may not be supported natively. You may simulate a queue by using a list or deque (double-ended queue), as long as you use only standard operations of a queue.
- You may assume that all operations are valid (for example, no pop or top operations will be called on an empty stack).

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

Solution for Python3:

```python
class MyStack:

    def __init__(self):
        """
        Initialize your data structure here.
        """
        from collections import deque
        self.deq = deque()


    def push(self, x):
        """
        Push element x onto stack.
        :type x: int
        :rtype: void
        """
        self.deq.append(x)
        return


    def pop(self):
        """
        Removes the element on top of the stack and returns that element.
        :rtype: int
        """
        return self.deq.pop()


    def top(self):
        """
        Get the top element.
        :rtype: int
        """
        return self.deq[-1]
```

```
35
36
37      def empty(self):
38          """
39          Returns whether the stack is empty.
40          :rtype: bool
41          """
42          return not bool(len(self.deq))
43
44
45
46  # Your MyStack object will be instantiated and called as such:
47  # obj = MyStack()
48  # obj.push(x)
49  # param_2 = obj.pop()
50  # param_3 = obj.top()
51  # param_4 = obj.empty()
```

Solution for C++:

```
1   class MyStack {
2   public:
3       queue<int> q;
4       /** Initialize your data structure here. */
5       MyStack() {
6       }
7
8       /** Push element x onto stack. */
9       void push(int x) {
10          q.push(x);
11          for (int i = 0; i < q.size() - 1; i++) {
12              q.push(q.front());
13              q.pop();
14          }
15      }
16
17      /** Removes the element on top of the stack and returns that element. */
18      int pop() {
19          int t = q.front();
20          q.pop();
21          return t;
22      }
23
24      /** Get the top element. */
25      int top() {
26          return q.front();
27      }
28
29      /** Returns whether the stack is empty. */
30      bool empty() {
31          return q.empty();
32      }
33  };
```