



链滴

[每日 LeetCode] 225. Implement Stack using Queues

作者: [Hanseltu](#)

原文链接: <https://ld246.com/article/1557232739177>

来源网站: [链滴](#)

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

原文链接 [\[每日LeetCode\] 225. Implement Stack using Queues](#)

Description:

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.

Example:

```
MyStack stack = new MyStack();
```

```
stack.push(1);
stack.push(2);
stack.top(); // returns 2
stack.pop(); // returns 2
stack.empty(); // returns false
```

Notes:

- You must use `_only_standard` operations of a queue -- which means only `push to back, peek 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).

思路：本题是[\[每日LeetCode\] 232. Implement Queue using Stacks](#)镜像题，本题要求用队列结构实现队列的基本操作。由于队列先进先出的特性，考虑只使用一个队列完成操作，具体做法为：对每个的 `push` 操作后，对当前队列逆序，让队头始终保存刚刚压入的元素，如此一来即可实现栈的基本操作。

C++代码

```
class MyStack {
public:
    /** Initialize your data structure here. */
    MyStack() {
    }

    /** Push element x onto stack. */
    void push(int x) {
        temp.push(x);
        for (int i=0; i<temp.size() - 1; i++){
            int t = temp.front();
            temp.pop();
            temp.push(t);
        }
    }

    /** Get the top element. */
    int top() {
        return temp.front();
    }

    /** Returns whether the stack is empty. */
    bool empty() {
        return temp.empty();
    }
};
```

```
        temp.push(temp.front());
        temp.pop();
    }
}
/** Removes the element on top of the stack and returns that element.*/
int pop() {
    int tem = temp.front();
    temp.pop();
    return tem;
}
/** Get the top element.*/
int top() {
    return temp.front();
}
/** Returns whether the stack is empty.*/
bool empty() {
    return temp.empty();
}
private:
    queue<int> temp;
};
/**
* Your MyStack object will be instantiated and called as such:
* MyStack* obj = new MyStack();
* obj->push(x);
* int param_2 = obj->pop();
* int param_3 = obj->top();
* bool param_4 = obj->empty();
*/
```

运行时间： 4ms

运行内存： 8.9M