큐를 이용해 다음 연산을 지원하는 스택을 구현하라.

- void push(int x) Pushes element x to the top of the stack.
- int pop() Removes the element on the top of the stack and returns it.
- int top() Returns the element on the top of the stack.
- boolean empty() Returns true if the stack is empty, false otherwise.

1.deque

```
class MyStack:

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

    def push(self, x: int) -> None:
        """
        Push element x onto stack.
        """
        self.q.append(x)
        for _ in range(len(self.q) - 1):
            self.q.append(self.q.popleft())

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

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

    def empty(self) -> bool:
        """
        Returns whether the stack is empty.
        """
        return len(self.q) == 0


# 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()
```