Recall the following elementary data structures from CS 225.

- **A stack** supports the following operations.
  - `Push` pushes an element on top of the stack.
  - `Pop` removes the top element from a stack.
  - `IsEmpty` checks if a stack is empty.

- **A queue** supports the following operations.
  - `Push` adds an element to the back of the queue.
  - `Pull` removes an element from the front of the queue.
  - `IsEmpty` checks if a queue is empty.

- **A deque**, or double-ended queue, supports the following operations.
  - `Push` adds an element to the back of the queue.
  - `Pull` removes an element from the back of the queue.
  - `Cut` adds an element from the front of the queue.
  - `Pop` removes an element from the front of the queue.
  - `IsEmpty` checks if a queue is empty.

Suppose you have a stack implementation that supports all stack operations in constant time.

1. Describe how to implement a queue using two stacks and $O(1)$ additional memory, so that each queue operation runs in $O(1)$ amortized time.

2. Describe how to implement a deque using three stacks and $O(1)$ additional memory, so that each deque operation runs in $O(1)$ amortized time.