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.