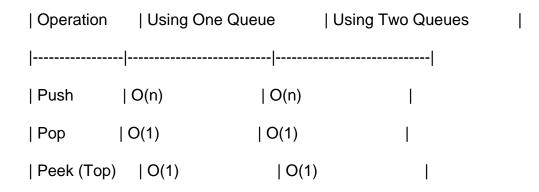
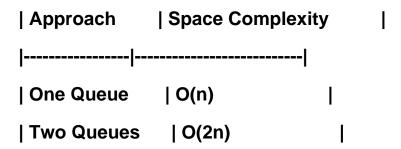
# **Comparison of Stack Representation Using Queues**

# 1. Time Complexity



- Push:
  - One Queue: Rotates the queue to ensure the last inserted element is always at the front (O(n)).
  - Two Queues: Transfers elements from one queue to another, which takes O(n).
- Pop & Peek:
- Both approaches allow O(1) complexity for these operations since the element at the "top" of the stack is directly accessible.

# 2. Space Complexity



- One Queue: Only one queue is used, so the space is proportional to the number of elements.
- Two Queues: Double the space is required since both queues exist

simultaneously.

# 3. Ease of Implementation

- One Queue:
  - Slightly trickier due to the need to rotate the queue on every push operation.
  - Requires understanding of rotation mechanics to simulate stack behavior.
- Two Queues:
- Conceptually simpler since push and pop are straightforward enqueue and dequeue operations.
  - Easier to debug as it involves standard queue transfers.
- 4. Efficiency in Real Scenarios

Scenario	One Que	ue	Two Queue	s	I
		-			
Space-Constrain	ned Systems	More efficie	nt (O(n) space	e).   Less	efficient
(O(2n) space).					
Heavy Push Op	erations	Less efficier	nt (O(n) per pu	ush).  Less	efficient
(O(n) per push).					
Heavy Pop Ope	rations	Equally effici	ient (O(1)).	Equally	efficient
(O(1)).					
Overall Simplici	ty   Slight	ly complex.	Simple	r to unders	tand.
1					

### 5. Use Cases

- One Queue:
  - Best when space is limited.

- Suitable for small-scale or memory-constrained applications.

#### - Two Queues:

- Useful in applications where simplicity and maintainability are more important.
  - Ideal for teaching or situations where clarity is critical.

## 6. Pros and Cons

#### One Queue:

#### Pros:

- Saves memory (O(n) space).
- Efficient for memory-constrained scenarios.

### Cons:

- Slower push (O(n) due to rotation).
- Harder to implement and debug.

### **Two Queues:**

#### Pros:

- Easier to implement and debug.
- Clear separation of roles between queues.

#### Cons:

- Doubles memory usage (O(2n) space).
- Same O(n) push complexity as one-queue.

### 7. Conclusion

- Use the one-queue approach when memory is a primary concern.
- Use the two-queue approach when simplicity and clarity are more important.