Comparison of Stack Representation Using Queues

1. Time Complexity

Operation	Using One Queue	Using Two Queues
Push	O(n)	O(n)
Pop	O(1)	O(1)
Peek (Top)	O(1)	O(1)

- 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

Approach	Space Complexity
One Queue	O(n)
Two Queues	O(2n)

- 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 Queue	Two Queues
Space-Constrained Systems	More efficient (O(n) space).	Less efficient (O(2n) space).
Heavy Push Operations	Less efficient (O(n) per push).	Less efficient (O(n) per push).
Heavy Pop Operations	Equally efficient (O(1)).	Equally efficient (O(1)).
Overall Simplicity	Slightly complex.	Simpler to understand.

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.