Challenge Question

Sequence: Push `["X", "Y", "Z"]`, then 1 pop, then push `"W"`

Algorithmic Steps:

1. Initialize an empty stack.
2. Push `"X"`, `"Y"`, `"Z"` → Stack becomes `["X", "Y", "Z"]`
3. Pop once → `"Z"` is removed → Stack becomes `["X", "Y"]`
4. Push `"W"` → Stack becomes `["X", "Y", "W"]`

Code:

```
stack = []

 # Step 1: Push elements

stack.append("X")

stack.append("Y")

 stack.append("Z")

 # Step 2: Pop one element

 stack.pop()

 # Step 3: Push "W"

 stack.append("W") # Final stack state print(stack)
```

<u>Output:</u>

```
['X', 'Y', 'W']
```

Reflection Question

Why do stack models handle undo/redo efficiently?

Stacks follow the Last-In, First-Out (LIFO**)** principle, which perfectly mirrors how undo/redo operations work:

- Undo**:** The most recent action is undone first. This is exactly what popping from a stack does.
- Redo: If undo actions are stored in a separate stack, redoing simply means popping from that redo stack.

Stacks are ideal for managing history in text editors, drawing apps, and command-based systems because they preserve the exact order of operations and allow quick reversal.

2.Challenge: Queue vs Stack for Ordering Meals

Scenario:

In a restaurant, customers place orders and expect to be served in the order they arrived.

Algorithmic Comparison:

| Feature | Stack (LIFO) | Queue (FIFO) |
|---|---|---|
| Order of Serving Last | person served first | First person served first |
| Fairness | ✖ Unfair (newcomers served first) | ✓ Fair (earliest served first) |
| Real-world analogy | Like stacking plates | Like a line at a food counter |

```
from collections import deque


orders = deque(["Order1", "Order2", "Order3"])

orders.append("Order4")  # New order


# Serve first order

served = orders.popleft()

print(f"Served: {served}")
```

Reflection: Why FIFO Ensures Customer Happiness

FIFO (First-In, First-Out) keeps customers happy because:

- Fairness: Everyone is served in the order they arrived—no cutting in line.
- Predictability**:** Customers can estimate their wait time based on position.
- Trust**:** Systems that honor arrival order build confidence and reduce frustration.
- Efficiency: FIFO avoids chaos and ensures smooth flow in high-traffic environments like banks, hospitals, and transport hubs.

In short, FIFO respects time and order—two things people value deeply when waiting.