Quiz 02:

Duration: 40 minutes
Total Problems: 2

Problem 1: Stack-Based Directory Traversal and Path Simplification (20 minutes)

You're building a file system utility that helps simplify and validate a given path in a directory. The path may include:

- Directory names (e.g., `docs`, `images`)
- `".."` (indicating moving up one level in the directory hierarchy)
- `"."` (indicating the current directory, which should be ignored)

Your task is to use a stack to handle these components and simplify the path to an absolute form. If the path starts with `/`, consider it as the root directory. Assume directory names only contain alphanumeric characters.

- **Requirements**:
  1. Write a function that simplifies paths like `"/home/../user/docs/./photos"` to `"/user/docs/photos"`.
  2. Handle cases where `".."` appears at the root directory (e.g., `"../../file.txt"` should return `"/file.txt"`).
  3. Use the stack to push directory names, pop when encountering `".."`, and ignore `"."`.
- **Input Examples**:
  - Input: *"/projects/../user/./docs/notes"*
    - Expected Output: *"/user/docs/notes"*
  - Input: *"../../work/files/./"*
    - Expected Output: *"/files"*

Problem 2: Queue-Based Task Scheduler (Array based queue)(20 minutes)

You're designing a task scheduler for a system that manages tasks in a specific order. The scheduler uses a circular queue to manage tasks and rotates the queue after each task is removed. The goal is to ensure that tasks cycle through consistently.

The rules are as follows:

1. **Enqueue**: Add a task to the queue. If it's full, display `"Queue is full"`—no rotation is necessary after enqueue.
2. **Dequeue**: Remove a task from the front of the queue. If the queue becomes empty, display `"Queue is empty"`.
3. **Rotation**: After each dequeue (but not after enqueue), rotate the queue to the left by one position. This will keep the queue in a cycle and ensure tasks cycle through consistently without causing additional rotations during enqueue operations.

Input Example

- **Operations**: *Enqueue: Task1, Task2, Task3, Dequeue, Enqueue: Task4, Enqueue: Task5*

Expected Output

Step 1 - Queue after Enqueue Task1: Task1
Step 2 - Queue after Enqueue Task2: Task1, Task2
Step 3 - Queue after Enqueue Task3: Task1, Task2, Task3
Step 4 - Queue after Dequeue and rotation: Task2, Task3
Step 5 - Queue after Enqueue Task4: Task2, Task3, Task4
Step 6 - Queue after Enqueue Task5: Queue is full