**Explanation on Water Jug Problem using DFS**

1. **Introduction**

The water jug problem is a classic problem in artificial intelligence and algorithm design. It involves two jugs with fixed capacities and aims to determine a sequence of operations to measure a specific amount of water using **depth-first search (DFS).**

1. **Problem Statement**

Given two jugs of capacities **X** and **Y** liters and an initial empty state, find a sequence of operations (fill, empty, pour) that results in exactly **Z** liters of water in either jug.

1. **Implementation Details**

* The provided notebook implements a **DFS**-based solution:
* Uses a stack to simulate the recursive depth-first traversal.
* Maintains a set of visited states to avoid infinite loops.
* Defines a set possible operations:
* Filling a jug to its full capacity.
* Emptying a jug completely.
* Pouring water from one jug to another until either the source is empty or the target is full.
* The algorithm continues exploring states until the goal amount (Z) is reached or all possibilities are exhausted.

1. **Code Summary**

* **Function Definitions:** The main function utilizes a stack to perform **DFS**, checking all possible moves and avoiding revisited states.
* **State Representation:** Each state is represented as a tuple **(X, Y)** corresponding to the current water levels in the jugs.
* **Condition Checks:** Ensures that a valid solution exists by verifying **(Z <= max (X, Y))** and whether **Z** is a multiple of the **GCD** of **X** and **Y**.
* **Output:** if a solution exists, it prints the sequence of steps taken; otherwise, it reports that no solution.

1. **Conclusion**

This notebook successfully implements the Water Jug Problem using **DFS**. While effective, an alternative approach using **BFS** would be more optimal for finding the shortest solution.