Experiment-3

Water Jug Problem

AIM

To solve the water jug problem using the Depth-First Search algorithm. The goal is to measure exactly 2 liters of water in the 4-liter jug while ensuring that the 3-liter jug is empty, using a series of operations.

ALGORITHM

1. Represent the state of the jugs as a tuple (x, y), where x is the amount of water in the 4-liter jug and y is the amount of water in the 3-liter jug.
2. Start with both jugs empty: $(0, 0)$.
3. Define possible operations.
4. Start from the initial state $(0, 0)$ and use DFS to explore all possible states by applying the operations.
5. Mark each visited state to avoid revisiting.
6. If the state $(2, 0)$ is reached, where the 4-liter jug has 2 liters and the 3-liter jug is empty, the solution is found.
7. If a state leads to no further valid states, backtrack and try a different approach.

PROGRAM

```
def is_valid_state(x, y):
    return 0<=x<=4 and 0<=y<=3

def dfs(x, y, visited, path):
    if (x, y) in visited:
        return False

    visited.add((x, y))
    path.append((x, y))
    if x == 2 and y == 0:
        return True

    possible_moves = [
        (4, y),
        (x, 3),
        (0, y),
        (x, 0),
        (x - min(x, 3-y), y + min(x, 3-y)),
        (x + min(y, 4-x), y - min(y, 4-x))
    ]

    for (next_x, next_y) in possible_moves:
        if is_valid_state(next_x, next_y) and
                dfs(next_x, next_y, visited, path):
            return True

    path.pop()
    return False

def solve_water_jug_problem():
    initial_state = (0, 0)
    visited = set()
    path = []
    if dfs(initial_state[0], initial_state[1], visited,
                                                path):
        print("Solution found!")
        for step in path:
            print(step)
    else:
        print("No solution exists.")
```

```
if __name__ == "__main__":
    solve_water_jug_problem()
```

OUTPUT

```
Solution found!
(0,0)
(0, 3)
(3,0)
(3,3)
(4, 2)
(0,2)
(2,0)
```

Result

Thus, the water jug problem was effectively solved using Depth First Search algorithm. The output is verified.