CODE:

```
def is_valid_state(state, visited):
    return state not in visited

def dfs(current_state, target, jug1_capacity, jug2_capacity, visited, solution):
    visited.add(current_state)
    solution.append(current_state)

    if current_state[0] == target or current_state[1] == target:
        return True

    jug1, jug2 = current_state

    possible_states = [
        (jug1_capacity, jug2),
        (jug1, jug2_capacity),
        (0, jug2),
        (jug1, 0),
        (max(jug1 - (jug2_capacity - jug2), 0), min(jug2 + jug1, jug2_capacity)),
        (min(jug1 + jug2, jug1_capacity), max(jug2 - (jug1_capacity - jug1), 0)),
    ]

    for state in possible_states:
        if is_valid_state(state, visited):
            if dfs(state, target, jug1_capacity, jug2_capacity, visited, solution):
                return True

    solution.pop()
    return False
```

```
def water_jug_problem(jug1_capacity, jug2_capacity, target):
    visited = set()
    solution = []

    if dfs((0, 0), target, jug1_capacity, jug2_capacity, visited, solution):
        return solution
    else:
        return "No solution found."

jug1_capacity = 4
jug2_capacity = 3
target = 2

solution = water_jug_problem(jug1_capacity, jug2_capacity, target)
print("Solution steps:")
for step in solution:
    print(step)
```

OUTPUT:

```
Solution steps:
(0, 0)
(4, 0)
(4, 3)
(0, 3)
(3, 0)
(3, 3)
(4, 2)
```


RESULT:

Thus, the water jug program has been implemented successfully.