PRINCIPLES OF ARTIFICIAL INTELLIGENCE
LABORATORY PROGRAMS

WATER JUG PROBLEM:

SOURCE CODE:

```
from collections import deque

def waterJugProblem(a, b, target):
    # Dictionary to track visited states
    visited = {}

    # Queue for BFS traversal
    queue = deque([(0, 0)])

    # Path to store the sequence of states
    path = []

    # Flag to indicate if the target is reachable
    isSolvable = False

    while queue:
        # Current state
        state = queue.popleft()

        # If state is already visited, skip
        if state in visited:
            continue

        # Mark state as visited
        visited[state] = True

        # Add state to path
        path.append(state)

        # Check if target is reached
```

```
        if state[0] == target or state[1] == target:
            isSolvable = True
            break

        # Perform possible operations
        for newState in [(a, state[1]), (state[0], b), (0, state[1]), (state[0], 0),
                         (min(state[0] + state[1], a), 0 if state[0] + state[1] <= a else state[1] - (a -
state[0])),
                         (0 if state[0] + state[1] <= b else state[0] - (b - state[1]), min(state[0] +
state[1], b))]:
            if newState not in visited:
                queue.append(newState)

    # If solution is found, print path
    if isSolvable:
        print("Solution Found:")
        for state in path:
            print(state)
    else:
        print("No solution found.")

# Example usage
if __name__ == "__main__":
    Jug1, Jug2, target = 4, 3, 2
    print("Starting water jug problem with Jug1={}, Jug2={}, and target={}".format(Jug1,
Jug2, target))
    waterJugProblem(Jug1, Jug2, target)
```

OUTPUT: