Question:3

WATER JUG PROBLEM

AIM

To solve the water jug problem using python.

ALGORITHM

1. Define a class `State` to represent the state of the two jugs.
2. Implement methods for equality comparison and hashing in the `State` class.
3. Define a function `get_next_states` to generate all possible next states from a given state.
4. Implement a function `solve` to find the solution using breadth-first search (BFS).
5. Initialize a queue with the initial state and an empty path, and a set to track visited states.
6. While the queue is not empty, dequeue a state and its path.
7. If the dequeued state is the goal state, return the path.
8. If the state is already visited, continue to the next state.
9. Add the current state to the visited set and enqueue its next possible states with their paths.
10. If no solution is found, return `None`.
11. Implement a `main` function to call the `solve` function and print the solution if found.

CODE

```
from collections import deque

class State:
    def __init__(self, jug1, jug2):
        self.j1, self.j2 = jug1, jug2

    def __eq__(self, other):
        return self.j1 == other.j1 and self.j2 == other.j2

    def __hash__(self):
        return hash((self.j1, self.j2))

def get_next_states(s):
    return [State(4, s.j2), State(s.j1, 3), State(0, s.j2), State(s.j1, 0),
            State(max(0, s.j1 - (3 - s.j2)), min(s.j1 + s.j2, 3)),
            State(min(s.j1 + s.j2, 4), max(0, s.j2 - (4 - s.j1)))]

def solve():
```

```
    q, v, g = deque([(State(0, 0), [])]), set(), State(2, 0)
    while q:
        (c, p) = q.popleft()
        if c == g: return p
        if c in v: continue
        v.add(c)
        q.extend((n, p + [n]) for n in get_next_states(c))
    return None

def main():
    sol = solve()
    if sol: print("\n".join(f"Step {i+1}: Jug1={s.j1}, Jug2={s.j2}" for i, s in enumerate(sol)))
    else: print("No solution found.")

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

OUTPUT

```
Step 1: (0, 3)
Step 2: (3, 0)
Step 3: (3, 3)
Step 4: (4, 2)
Step 5: (0, 2)
Step 6: (2, 0)
```