**EXPERIMENT 3:**

from collections import deque

def is\_goal(state):

return state[0] == 2 or state[1] == 2

def get\_next\_states(state, max\_a, max\_b):

a, b = state

states = []

states.append((max\_a, b))

states.append((a, max\_b))

states.append((0, b))

states.append((a, 0))

pour = min(a, max\_b - b)

states.append((a - pour, b + pour))

pour = min(b, max\_a - a)

states.append((a + pour, b - pour))

return states

def bfs(max\_a, max\_b):

start = (0, 0)

queue = deque()

queue.append((start, [start]))

visited = set()

while queue:

(current, path) = queue.popleft()

if current in visited:

continue

visited.add(current)

if is\_goal(current):

return path

for next\_state in get\_next\_states(current, max\_a, max\_b):

queue.append((next\_state, path + [next\_state]))

return None

max\_a = 4

max\_b = 3

solution = bfs(max\_a, max\_b)

if solution:

print("Steps to reach the goal:")

for step in solution:

print(f"Jug A: {step[0]}L, Jug B: {step[1]}L")

else:

print("No solution found.")

**OUTPUT:**

