**Practical: 3**

**Aim: Implement the WaterJug Problem. The state of the system is represented as (*x*,*y*) where *x* is the quantity of water in the 4-gallon jug and *y* is the quantity of water in the 3-gallon jug. Assume the initial state to be (0,0) and goal state to be (2,0).**

**Output the sequence of steps to go from initial to goal state.**

**Program:**

from collections import deque

# Define the capacities of the two jugs

capacity\_4 = 4

capacity\_3 = 3

# Define the goal state

goal\_state = (2, 0)

# Function to print the sequence of steps

def print\_steps(steps):

for step in steps:

print(f"Jug 4: {step[0]}, Jug 3: {step[1]}")

# Function to get the possible next states from the current state

def get\_next\_states(state):

x, y = state

next\_states = []

# Fill jug 4

if x < capacity\_4:

next\_states.append((capacity\_4, y))

# Fill jug 3

if y < capacity\_3:

next\_states.append((x, capacity\_3))

# Empty jug 4

if x > 0:

next\_states.append((0, y))

# Empty jug 3

if y > 0:

next\_states.append((x, 0))

# Pour water from jug 3 to jug 4

if y > 0 and x < capacity\_4:

transfer = min(y, capacity\_4 - x)

next\_states.append((x + transfer, y - transfer))

# Pour water from jug 4 to jug 3

if x > 0 and y < capacity\_3:

transfer = min(x, capacity\_3 - y)

next\_states.append((x - transfer, y + transfer))

return next\_states

# Function to solve the Water Jug Problem

def water\_jug\_solver(initial\_state, goal\_state):

# Perform BFS to find the solution

queue = deque([(initial\_state, [])])

visited = set([initial\_state])

while queue:

current\_state, path = queue.popleft()

# If we reach the goal state, return the path

if current\_state == goal\_state:

return path + [current\_state]

# Explore the next states

for next\_state in get\_next\_states(current\_state):

if next\_state not in visited:

visited.add(next\_state)

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

return []

# Initial state (0, 0)

initial\_state = (0, 0)

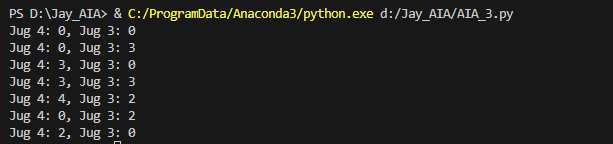
# Find the sequence of steps to reach the goal state

steps = water\_jug\_solver(initial\_state, goal\_state)

# Print the sequence of steps

print\_steps(steps)

**Output:**

****