19.Write the python program for Water Jug Problem

from collections import deque

def water\_jug\_bfs(jug1\_capacity, jug2\_capacity, target):

visited = set()

parent = dict() # For tracking the path

queue = deque()

# Start from (0, 0)

queue.append((0, 0))

visited.add((0, 0))

parent[(0, 0)] = None

while queue:

x, y = queue.popleft()

# If we reach the target in either jug

if x == target or y == target:

print("Solution steps:")

print\_path((x, y), parent)

return True

# Possible next states

next\_states = set()

# Fill Jug1

next\_states.add((jug1\_capacity, y))

# Fill Jug2

next\_states.add((x, jug2\_capacity))

# Empty Jug1

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

# Empty Jug2

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

# Pour Jug1 -> Jug2

transfer = min(x, jug2\_capacity - y)

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

# Pour Jug2 -> Jug1

transfer = min(y, jug1\_capacity - x)

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

for state in next\_states:

if state not in visited:

visited.add(state)

parent[state] = (x, y)

queue.append(state)

print("No solution found.")

return False

def print\_path(state, parent):

path = []

while state:

path.append(state)

state = parent[state]

path.reverse()

for s in path:

print(f"Jug1: {s[0]}L, Jug2: {s[1]}L")

# Example usage

jug1\_capacity = 4

jug2\_capacity = 3

target = 2

water\_jug\_bfs(jug1\_capacity, jug2\_capacity, target)

OUTPUT:

