SUPERIOR UNIVERSITY

Programming For Artificial Intelligence (Lab)

Assignment - 3

Name:

Ali Maqsood.

Roll no:

SU92-BSAIM-F23-050.

Department:

Software Engineering Department.

Program:

Artificial Intelligence.

Section:

BSAI-4A

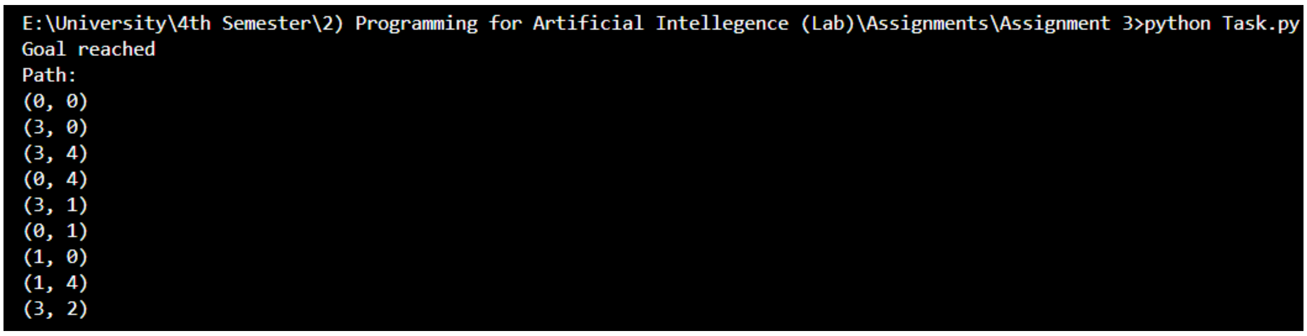
Question # 1:

Implement the Water Jug Problem using Depth-First Search (DFS) and print the rules applied at each step.

Code:

```
def water_jug_dfs(x,y,visited,path):
    if x==2 or y==2:
        print("Goal reached")
        print("Path:")
        for i in path+[(x,y)]:
            print(i)
        return True
    visited.add((x,y))
    moves=[]
    moves.append((3,y))
    moves.append((x,4
    moves.append((0,y))
    moves.append((x,0))
    if x+y<=4:
        moves.append((0,x+y))
    else:
        moves.append((x-(4-y),4))
    if x+y<=3:
        moves.append((x+y,0))
    else:
        moves.append((3,y-(3-x)))
    for move in moves:
        if move not in visited:
            if water_jug_dfs(move[0],move[1],visited,path+[(x,y)]):
                return True
    return False
visited=set()
water_jug_dfs(0, 0, visited, [])
```

Output:



```
E:\University\4th Semester\2) Programming for Artificial Intellegence (Lab)\Assignments\Assignment 3>python Task.py
Goal reached
Path:
(0, 0)
(3, 0)
(3, 4)
(0, 4)
(3, 1)
(0, 1)
(1, 0)
(1, 4)
(3, 2)
```

Documentation:

This code solves the **Water Jug Problem** using a **Depth First Search (DFS)**. It starts from the initial state (0, 0) and explores all possible states of the two jugs, one with a **capacity of 3-liters** and the other with a **capacity of 4- liter**, to measure **2 liter** as that is our goal. The algorithm uses a recursive DFS function that checks if either jug contains 2 liters, then it prints the complete path taken to reach the goal. From each state, it generates all possible next moves until one is full or empty. Each visited state is recorded to prevent it from looping. The search continues depth-first until a valid solution is found, and the sequence of steps is displayed in the end.