Page

Exp No:03
Date

Water Jug Problem using DFS

Aim:

To write a program to solve the water jug problem dynamically using DFS

Algorithm:

Step 1: Start

Step 2: Input Jug capacities and target level

Step 3: Start the initial state (0,0) where both jugs are empty

Step 4: Use DFS to explore all possible States and check for target Solution

Step 5: Use a set to store all Visited states (x,y) to avoid revisiting

Step 6: The program terminates when target is reached

Step 7: Stop

Program:

```
def dfs (jug1, jug2, target, visited, path):

        x, y = path [-1]
        if (x == target or y == target):
            Print (path)
            return True
      if (x, y) is visited
            return False
      Visited.add ((x, y))
      next states = [
        (jug1, y)
        (x, jug2)
         (0, y)
         (x, 0)
        (x - min (x, jug2 - y), y + min (x, jug2 - y)),
        (x + min (y, jug1 - x), y - min (y - jug - x)),
      ]
   return any (dfs (jug1, jug2, target, visited,
         path + [state]) for state in next_states)
def water_jug_problem (jug1, jug2, target)
  if not dfs (jug1, jug2, target, set(), [(0,0)])
        Point ("No Solution")
jug1 = int (input ("Enter the jug1 capacity:"))
Jug2 = int (Input ("Enter the jug2 capacity:"))
target = Int (input ("Target amount:"))
 Water jug_problem (jug1, jug2, target)
```

Date
Page

Output:

```
Enter the capacity of Jug 1 : 4
Enter the capacity of Jug2 : 3
Enter the target amount : 2

Jug1 : 0 litre , Jug2 : 0 litre
       4                0
       1                3
       1                0
       0                1
       4                1
Jug 1: 2 litre    Jug 2: 3 litre
```

Result:

Thus the program to solve water jug problem has been executed successfully.

[illegible]