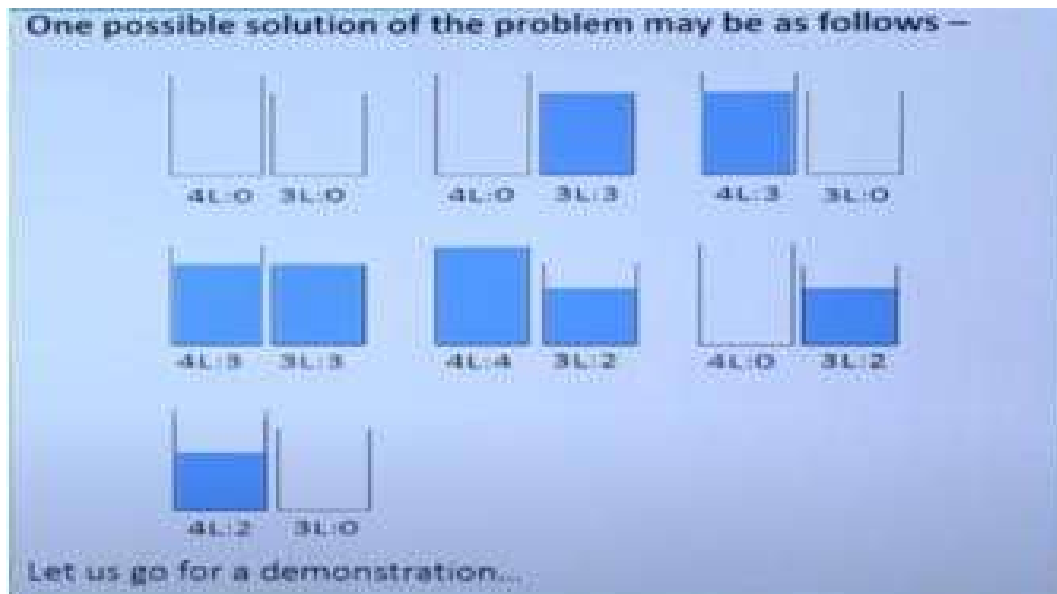


EX.NO:DATE:**DEPTH-FIRST SEARCH – WATER JUG PROBLEM**

In the **water jug problem in Artificial Intelligence**, we are provided with two jugs: one having the capacity to hold 3 gallons of water and the other has the capacity to hold 4 gallons of water. There is no other measuring equipment available and the jugs also do not have any kind of marking on them. So, the agent's task here is to fill the 4-gallon jug with 2 gallons of water by using only these two jugs and no other material. Initially, both our jugs are empty.



CODE:

```

from collections import deque

def DFS(a, b, target):

    m = {}
    isSolvable = False
    path = []
    q = deque()

    q.append((0, 0))

    while (len(q) > 0):
        u = q.popleft()
        if ((u[0], u[1]) in m):
            continue

        if ((u[0] > a or u[1] > b or
            u[0] < 0 or u[1] < 0)):
            continue

        path.append([u[0], u[1]])

        m[(u[0], u[1])] = 1

        if (u[0] == target or u[1] == target):
            isSolvable = True

            if (u[0] == target):
                if (u[1] != 0):
                    path.append([u[0], 0])
            else:
                if (u[1] != 0):
                    path.append([0, u[1]])
                sz = len(path)
                for i in range(sz):
                    print("(", path[i][0], ", ", path[i][1], ")")
                break
            q.append([u[0], b])
            q.append([a, u[1]])
        for ap in range(max(a, b) + 1):
            c = u[0] + ap
            d = u[1] - ap

            if (c == a or (d == 0 and d >= 0)):
                q.append([c, d])

```

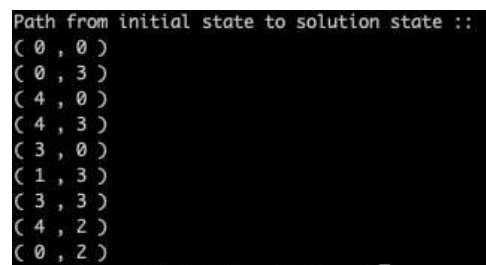
```
c = u[0] - ap
d = u[1] + ap

if ((c == 0 and c >= 0) or d == b):
    q.append([c, d])

q.append([a, 0])
q.append([0, b])
if (not isSolvable):
    print ("No solution")

Jug1, Jug2, target = 4, 3, 2
print("Path from initial state ""to solution state ::")
DFS(Jug1, Jug2, target)
```

OUTPUT:



```
Path from initial state to solution state ::
( 0 , 0 )
( 0 , 3 )
( 4 , 0 )
( 4 , 3 )
( 3 , 0 )
( 1 , 3 )
( 3 , 3 )
( 4 , 2 )
( 0 , 2 )
```

RESULT:

Thus, the code has been successfully executed, and the output has been verified successfully.