# **DFS Water Jug Problem**

**Program:**

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

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([a,0])

else:

if(u[0]!=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])

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 ) Path from initial state to solution state:

( 0 , 0 )

( 0 , 3 )

( 4 , 0 )

( 4 , -4 )

( 4 , 3 )

( 3 , 0 )

( 4 , -1 )

( 7 , 0 )

( 3 , 3 )

( 7 , 3 )

( 4 , 2 )

( 0 , 2 )

( 4 , -4 )

( 4 , 3 )

( 3 , 0 )

( 4 , -1 )

( 7 , 0 )

( 3 , 3 )

( 7 , 3 )

( 4 , 2 )

( 0 , 2 )