|  |  |
| --- | --- |
| **EX.NO: 02** | **WATERJUGPROGRAMUSINGBFS** |
| **DATE:06.03.2024** |

**PROBLEM STATEMENT:**

Youaregiventhe2 jugs,A4-gallonjugand3-gallonjug neitherhasanymeasuring markerson it. There is a pump that can be used to fillthe jug with water how can you get two gallons of waterinto the 4-gallon jug.

**AIM:**

To solve the Water jug Problem using Breadth First Search.

**ALGORITHM:**

Step 1: Start.

Step 2: Get the capacity of A jug and B jug and Target.

Step 3: Create water jug problem function.

Step 4: In Function A, B, Target are parameter.

Step 5: In Function initialize state = (0, 0).

Step 6: Create parent empty set.

Step 7: Create frontier isn’t there is all the possible states to will be stored.

Step 8: Using while loop frontier is the condition.

Step 9: Assign state = frontier pop.

Step 10: If state is reaching the target.

**PROGRAM:**

fromcollectionsimportdeque def BFS(a, b, target):

m={}

isSolvable=False path = []

q = deque() q.append((0,0))

while len(q)>0:

u=q.popleft()#Usepopleft togetthefirst element(breadth-first) if (u[0], u[1]) in m:

continue

ifu[0] >aoru[1]>boru[0]<0or u[1]<0: continue

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

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

ifu[0]==targetoru[1]==target: isSolvable = True

ifu[0]==target: if u[1] != 0:

path.append([u[0],0]) else:

if u[0] != 0: path.append([0,u[1]])

sz=len(path)

forIinrange(sz):

print(“(“,path[i][0],“,”,path[i][1],“)”)

return#Exitingthefunctionafterfindingthesolution q.append([u[0], b])

q.append([a, u[1]])

forapinrange(max(a,b)+1):

c=u[0]+ap

d= u[1]–ap

ifc==aor(d==0andd>=0): q.append([c, d])

c=u[0] - ap

d= u[1]+ap

if(c== 0andc >=0)ord==b:

q.append([c,d])

q.append([a, 0])

q.append([0,b])

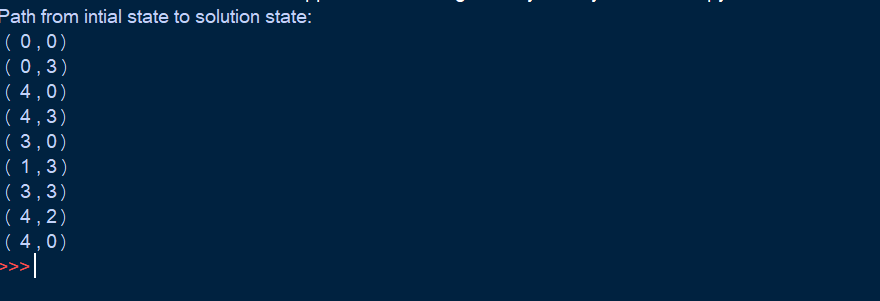
if not isSolvable: print("Nosolution")

ifname =='main':

Jug1,Jug2,target=4,3,2

print("Pathfrominitialstatetosolutionstate:") BFS(Jug1, Jug2, target)

**OUTPUT:**

****

**RESULT:**

Thus, to solve the Water jug Problem using Breadth First Search has been executed successfully.