**WATER JUG PROGRAM USING BFS**

|  |  |
| --- | --- |
| **EX.N0:4** | **WRITE PROG PROGRAM TO WATER JUG PROGRAM USING BFS** |
| **DATE: 20/03/2024** |

**AIM:**

To solve the Water jug Problem using Breadth First Search.

**ALGORITHM:**

Step1:Start.

Step2:Get the capacity of A jug and B jug and Target.

Step3: Create water jug problem function.

Step4:In Function A,B, Target are parameter.

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

Step6:  Create parent empty set.

Step7:Create frontier isn’t there is all the possible states to will be stored

Step8:  Using while loop frontier is the condition.

Step 9:Assign state =frontier pop.

Step10:If state is reach the target

**PROGRAM:**

from collections import deque

def BFS(a, b, target):

    m = {}

    isSolvable = False

    path = []

    q = deque()

    q.append((0, 0))

    while len(q) > 0:

        u = q.popleft()

        print("Exploring state:", u)

        if (u[0], u[1]) in m:

            print("Already visited state:", u)

            continue

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

            print("Invalid state:", u)

            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[0] != 0:

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

            sz = len(path)

            print("Path:")

            for i in range(sz):

                print("(", path[i][0], ",", path[i][1], ")")

            break

        q.append([u[0], b])  # Fill Jug2

        q.append([a, u[1]])  # Fill Jug1

        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])  # Empty Jug1

        q.append([0, b])  # Empty Jug2

    if not isSolvable:

        print("No solution")

if \_\_name\_\_ == '\_\_main\_\_':

    Jug1, Jug2, target = 4, 3, 2

    print("Path from initial state to solution state:")

    BFS(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

**RESULT:** Thus the experiment to do water jug problem(BFS) by using python has been executed and verified Successfully.