‘’’ Title : Design a program to solve the Water jug problem

Roll no: 3946

‘’’

# This function is used to initialize the

# dictionary elements with a default value.

from collections import defaultdict

# jug1 and jug2 contain the value

# for max capacity in respective jugs

# and aim is the amount of water to be measured.

#jug1, jug2, aim = 4, 3, 2

jug1 = int(input("Enter quantitiy in jug 1:"))

jug2 = int(input("Enter quantity in jug 2:"))

aim=  int(input("Enter amount of water in jug 1 at the end:"))

aim2=int(input("Enter amount of water in jug 2 at the end:"))

# Initialize dictionary with

# default value as false.

visited = defaultdict(lambda: False)

# Recursive function which prints the

# intermediate steps to reach the final

# solution and return boolean value

# (True if solution is possible, otherwise False).

# amt1 and amt2 are the amount of water present

# in both jugs at a certain point of time.

def waterJugSolver(amt1, amt2):

    # Checks for our goal and

    # returns true if achieved.

    if (amt1 == aim and amt2 == aim2):

        print(amt1, amt2)

        return True

    # Checks if we have already visited the

    # combination or not. If not, then it proceeds further.

    if visited[(amt1, amt2)] == False:

        print(amt1, amt2)

        # Changes the boolean value of

        # the combination as it is visited.

        visited[(amt1, amt2)] = True

        # Check for all the 6 possibilities and

        # see if a solution is found in any one of them.

        return (

                waterJugSolver(0, amt2) or

                waterJugSolver(amt1, 0) or

                waterJugSolver(jug1, amt2) or

                waterJugSolver(amt1, jug2)or

                waterJugSolver(amt1 + min(amt2, (jug1-amt1)),

                amt2 - min(amt2, (jug1-amt1))) or

                waterJugSolver(amt1 - min(amt1, (jug2-amt2)),

                amt2 + min(amt1, (jug2-amt2)))

                )

    # Return False if the combination is

    # already visited to avoid repetition otherwise

    # recursion will enter an infinite loop.

    else:

        return False

print("Steps: ")

# Call the function and pass the

# initial amount of water present in both jugs.

waterJugSolver(0, 0)

'''

1st method:

Enter quantitiy in jug 1:5

Enter quantity in jug 2:3

Enter amount of water at the end:2

Steps:

(0, 0)

(5, 0)

(2, 3)

(0, 3)

(3, 0)

(3, 3)

(5, 1)

(0, 1)

(1, 0)

(1, 3)

(4, 0)

(4, 3)

(5, 2)

(0, 2)

'''

''' 2nd method :

Enter quantitiy in jug 1:5

Enter quantity in jug 2:3

Enter amount of water at the end:2

Steps:

(0, 0)

(5, 0)

(5, 3)

(0, 3)

(3, 0)

(3, 3)

(5, 1)

(0, 1)

(1, 0)

(1, 3)

(4, 0)

(4, 3)

(5, 2)

(0, 2)

'''