# Python3 program to implement traveling salesman

# problem using naive approach.

from sys import maxsize

from itertools import permutations

V = 4

# implementation of traveling Salesman Problem

def travellingSalesmanProblem(graph, s):

    # store all vertex apart from source vertex

    vertex = []

    for i in range(V):

        if i != s:

            vertex.append(i)

    # store minimum weight Hamiltonian Cycle

    min\_path = maxsize

    next\_permutation=permutations(vertex)

    for i in next\_permutation:

        # store current Path weight(cost)

        current\_pathweight = 0

        # compute current path weight

        k = s

        for j in i:

            current\_pathweight += graph[k][j]

            k = j

        current\_pathweight += graph[k][s]

        # update minimum

        min\_path = min(min\_path, current\_pathweight)

    return min\_path

# Driver Code

if \_\_name\_\_ == "\_\_main\_\_":

    # matrix representation of graph

    graph = [[0, 10, 15, 20], [10, 0, 35, 25],

            [15, 35, 0, 30], [20, 25, 30, 0]]

    s = 0

    print(travellingSalesmanProblem(graph, s))

**OUTPUT**

**80**