**Ex No: 07**

**TRANSPORTATION PROBLEM**

**AIM:**

To perform Transportation problem in python for the given equations with the constraints and get the optimized values.

**PROCEDURE:**

1. Import the necessary library functions.
2. If pulp is not available use pip install method and install pulp library and import the entire package
3. Give the required constraints and minimization function to the model
4. View the model constraints and verify it.
5. Solve the equations using PULP\_CBC\_CMD()
6. View the status of the model
7. Print the results which are calculated by the model
8. Get the optimized values of the given equation and constraints.

**PROGRAM:**

!pip install pulp

from pulp import \*

# Creates a list of all the supply nodes

supply\_nodes = ["S1","S2","S3"]

# Creates a dictionary for the number of units of supply for each supply node

supply = {"S1": 11,

      "S2": 13,

      "S3": 19}

# Creates a list of all demand nodes

demand\_nodes = ["D1","D2","D3" ,"D4"]

# Creates a dictionary for the number of units of demand for each demand node

demand = {"D1": 6,

      "D2": 10,

      "D3": 12,

      "D4":15}

# Creates a list of costs of each transportation path

costs = [# Demand

     #D1 D2 D3 D4

     [21,16,25,13], #S1

     [17,18,14,23], #S2  Supply

     [32,27,18,41]  #S3

     ]

costs = makeDict((supply\_nodes, demand\_nodes),costs)

print(costs)

# Creates the prob variable to contain the problem data

prob = LpProblem("Product Distribution Problem",LpMinimize)

# Creates a list of tuples containing all the possible routes for transport

Routes = [(s,d) for s in supply\_nodes for d in demand\_nodes]

# A dictionary called route\_vars is created to contain the referenced variables (the routes)

route\_vars = LpVariable.dicts("Route",(supply\_nodes,demand\_nodes),0,None,LpInteger)

# The objective function is added to prob first

prob += lpSum([route\_vars[s][d]\*costs[s][d] for (s,d) in Routes]), "Sum of Transporting Costs"

# The supply maximum constraints are added to prob for each supply node (warehouse)

for s in supply\_nodes:

    prob += lpSum([route\_vars[s][d] for d in demand\_nodes]) <= supply[s], "Sum of Products out of supply %s"%s

# The demand minimum constraints are added to prob for each demand node (bar)

for d in demand\_nodes:

    prob += lpSum([route\_vars[s][d] for s in supply\_nodes]) >= demand[d], "Sum of Products into demand %s"%d

prob.solve()

print("Status:",LpStatus[prob.status] )

total=0

for v in prob.variables():

    if(v.varValue != None):

        if (v.varValue > 0):

            print(v.name, "=", v.varValue)

            total+=v.varValue

total=0

for v in prob.variables():

    if(v.varValue != None):

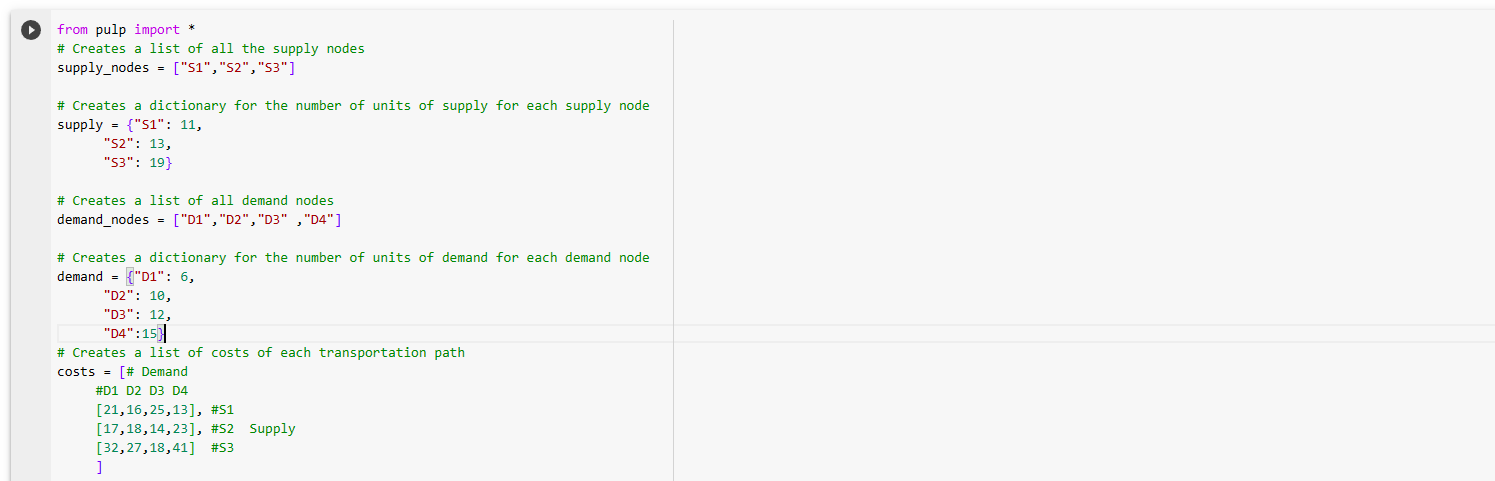
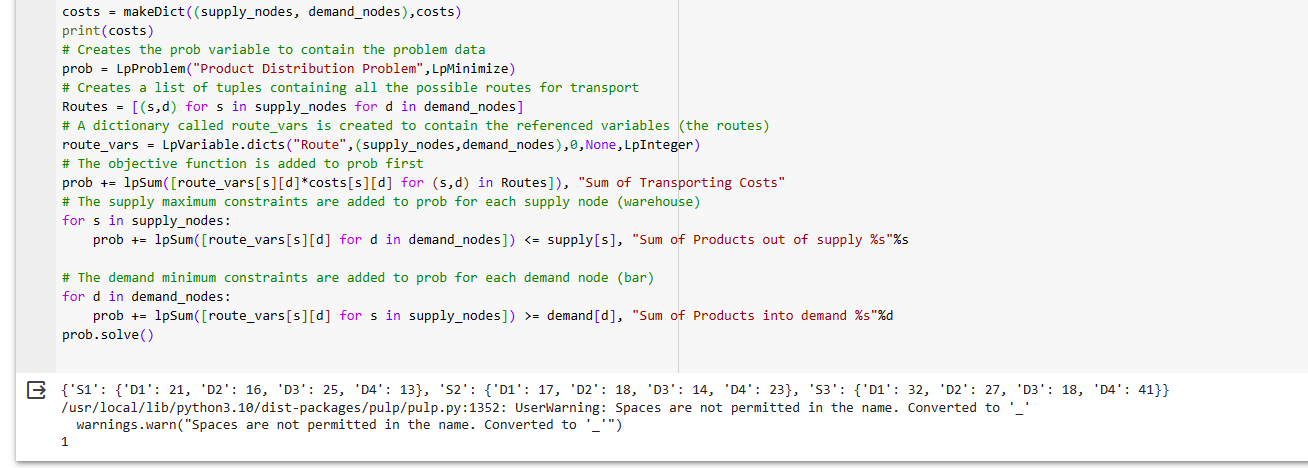
        if (v.varValue > 0):

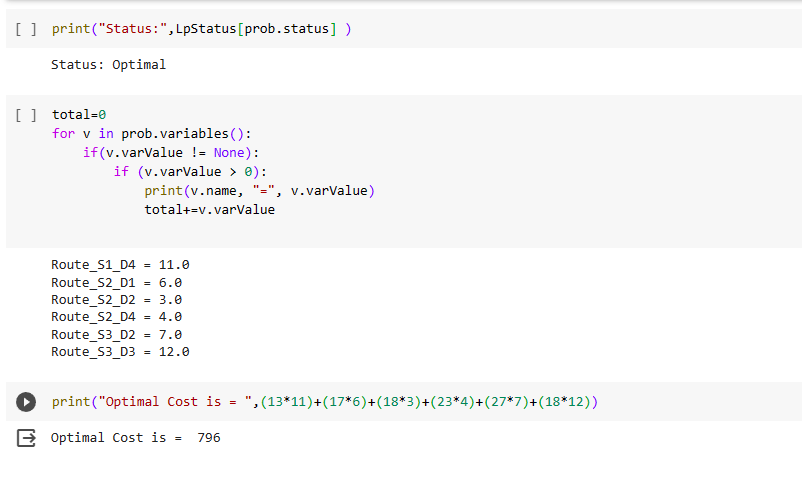
            print(v.name, "=", v.varValue)

            total+=v.varValue

print("Optimal Cost is = ",(13\*11)+(17\*6)+(18\*3)+(23\*4)+(27\*7)+(18\*12))

**OUTPUT:**





**CONCLUSION:**

Thus the transportation problem method using python was implemented and the results of various equations and optimized values was verified successfully.