

```
import pandas as pd

#Load the Excel files
file_path_demand = 'HW 4S - Demand.xlsx'
file_path_DCs = 'HW 4S - DCs.xlsx'
data_demand = pd.read_excel(file_path_demand)
data_DCs = pd.read_excel(file_path_DCs)

#Extract the data columns from the files and convert them into lists
D = data_demand['Demand'].tolist(); n = len(D)
F = data_DCs['Fixed Cost'].tolist(); m = len(F)
La_DCs = list(map(float, data_DCs['Latitude']))
Lo_DCs = list(map(float, data_DCs['Longitude']))
La_demand = list(map(float, data_demand['Latitude']))
Lo_demand = list(map(float, data_demand['Longitude']))

import numpy as np
import math as math

#Calculate all d_ij values and store them into a 2D(dimension) list
def haversine_distance(a_i, b_i, a_j, b_j):
    # Give radius of the Earth in kilometers, 6371.01km
    R = 6371.01

    # Convert degrees to radians
    a_i, b_i, a_j, b_j = map(np.radians, [a_i, b_i, a_j, b_j])

    # Apply Haversine formula
    delta_a = a_j - a_i
    delta_b = b_j - b_i
    haversine_distance = math.sin(delta_a/2)**2 + math.cos(a_i) * math.cos(a_j) * math.sin(delta_b/2)**2
    haversine_distance = math.sqrt(haversine_distance)
    haversine_distance = 2 * R * math.asin(haversine_distance)

    return haversine_distance

d_ij = [[haversine_distance(La_demand[i], Lo_demand[i], La_DCs[j], Lo_DCs[j]) for j in range(m)] for i in range(n)]
```

```

def Lagrange_Relaxation(F, D, c, lambdas):

    num_facilities = len(F)
    num_customers = len(D)

    #Initialize x and y
    x = [0]*num_facilities
    y = [[0]*num_facilities for _ in range(num_customers)]

    #Compute beta_j for each facility j
    beta = [0]*num_facilities
    for j in range(num_facilities):
        total_min_term = 0.0
        for i in range(num_customers):
            cost_diff = D[i]*c[i][j] - lambdas[i]
            total_min_term += min(0, cost_diff)
        beta[j] = total_min_term

    #Decide whether to open each facilities and assign customers by Theorem 8.1
    for j in range(num_facilities):
        if beta[j] + F[j] < 0: #We have potential revenue to open facility j
            x[j] = 1
            for i in range(num_customers):
                cost_diff = D[i]*c[i][j] - lambdas[i]
                if cost_diff < 0: #We have potential revenue to serve demand i
                    y[i][j] = 1

    #Compute the Lagrangian objective
    sum_lambdas = np.sum(lambdas)
    lagrangian_value = sum_lambdas + sum(min(0, beta[j] + F[j]) for j in range(num_facilities))

    return x, y, beta, lagrangian_value

```

```

#In global function, we try to use the function Lagrange_Relaxation

```

```

x = ?

```

```

n = 2.0
c = K*np.array(d_ij)

#We define lambdas to be the same scale of d*c
average_demand = sum(D)/n
average_trancost = sum([sum(row) for row in c])/(n*m)
lambdas = [average_demand*average_trancost/18]*n


x, y, beta, lagrangian_value = Lagrange_Relaxation(F, D, c, lambdas)

for j in range(m):
    if x[j] == 1:
        print(f"Facility {chr(65 + j)} should be open, because x_{j}=1.")

print("\n")
for i in range(n):
    for j in range(m):
        if y[i][j] == 1:
            print(f"Demand zone {i+1} should be served by facility {chr(65 + j)}, because y_{i}{j}=1.")

print("\n", f"The Lagrangian value is {lagrangian_value}. ")

```

 Facility A should be open, because x_0=1.
 Facility B should be open, because x_1=1.
 Facility C should be open, because x_2=1.
 Facility D should be open, because x_3=1.
 Facility E should be open, because x_4=1.
 Facility F should be open, because x_5=1.
 Facility G should be open, because x_6=1.
 Facility H should be open, because x_7=1.
 Facility I should be open, because x_8=1.
 Facility J should be open, because x_9=1.

Demand zone 3 should be served by facility A, because y_20=1.
 Demand zone 3 should be served by facility B, because y_21=1.
 Demand zone 3 should be served by facility C, because y_22=1.
 Demand zone 3 should be served by facility D, because y_23=1.
 Demand zone 3 should be served by facility E, because y_24=1.
 Demand zone 3 should be served by facility F, because y_25=1.
 Demand zone 3 should be served by facility G, because y_26=1.
 Demand zone 3 should be served by facility H, because y_27=1.
 Demand zone 3 should be served by facility I, because y_28=1.
 Demand zone 3 should be served by facility J, because y_29=1.

Demand zone 4 should be served by facility A, because $y_{30}=1$.
Demand zone 4 should be served by facility B, because $y_{31}=1$.
Demand zone 4 should be served by facility C, because $y_{32}=1$.
Demand zone 4 should be served by facility D, because $y_{33}=1$.
Demand zone 4 should be served by facility E, because $y_{34}=1$.
Demand zone 4 should be served by facility F, because $y_{35}=1$.
Demand zone 4 should be served by facility G, because $y_{36}=1$.
Demand zone 4 should be served by facility H, because $y_{37}=1$.
Demand zone 4 should be served by facility I, because $y_{38}=1$.
Demand zone 4 should be served by facility J, because $y_{39}=1$.
Demand zone 7 should be served by facility A, because $y_{60}=1$.
Demand zone 7 should be served by facility B, because $y_{61}=1$.
Demand zone 7 should be served by facility E, because $y_{64}=1$.
Demand zone 19 should be served by facility A, because $y_{180}=1$.
Demand zone 19 should be served by facility B, because $y_{181}=1$.
Demand zone 19 should be served by facility C, because $y_{182}=1$.
Demand zone 19 should be served by facility D, because $y_{183}=1$.
Demand zone 19 should be served by facility E, because $y_{184}=1$.
Demand zone 19 should be served by facility F, because $y_{185}=1$.
Demand zone 19 should be served by facility G, because $y_{186}=1$.
Demand zone 19 should be served by facility H, because $y_{187}=1$.
Demand zone 19 should be served by facility I, because $y_{188}=1$.
Demand zone 19 should be served by facility J, because $y_{189}=1$.
Demand zone 48 should be served by facility A, because $y_{470}=1$.

The Lagrangian value is 146206.9575943991.

