**Program:**

import numpy as np

# Distance matrix

d = np.array([

[0, 10, 12, 11, 14],

[10, 0, 13, 15, 8],

[12, 13, 0, 9, 14],

[11, 15, 9, 0, 16],

[14, 8, 14, 16, 0]

])

n\_ants = 5

n\_citys = 5

iterations = 100

e = 0.5 # Evaporation rate

alpha = 1 # Pheromone factor

beta = 2 # Visibility factor

# Inverse distance = visibility

visibility = 1 / (d + np.diag([np.inf]\*n\_citys))

visibility[np.isinf(visibility)] = 0 # Avoid division by zero on diagonals

# Initialize pheromone matrix between cities (n x n)

pheromone = 0.1 \* np.ones((n\_citys, n\_citys))

# Best route tracking

best\_route = None

best\_cost = np.inf

for ite in range(iterations):

rute = np.ones((n\_ants, n\_citys + 1), dtype=int) # Ant routes

for ant in range(n\_ants):

unvisited = list(range(1, n\_citys)) # Exclude city 0 (starting point)

for j in range(1, n\_citys):

current\_city = rute[ant, j - 1]

prob = np.zeros(n\_citys)

for k in unvisited:

prob[k] = (pheromone[current\_city][k] \*\* alpha) \* (visibility[current\_city][k] \*\* beta)

prob\_sum = np.sum(prob)

if prob\_sum == 0:

next\_city = np.random.choice(unvisited)

else:

prob = prob / prob\_sum

next\_city = np.random.choice(range(n\_citys), p=prob)

rute[ant, j] = next\_city

unvisited.remove(next\_city)

rute[ant, -1] = rute[ant, 0] # Return to starting city

# Cost for each ant

dist\_cost = np.zeros(n\_ants)

for ant in range(n\_ants):

for j in range(n\_citys):

from\_city = rute[ant, j]

to\_city = rute[ant, j + 1]

dist\_cost[ant] += d[from\_city][to\_city]

# Best tour update

min\_idx = np.argmin(dist\_cost)

if dist\_cost[min\_idx] < best\_cost:

best\_cost = dist\_cost[min\_idx]

best\_route = rute[min\_idx].copy()

# Pheromone evaporation

pheromone = (1 - e) \* pheromone

# Pheromone update

for ant in range(n\_ants):

for j in range(n\_citys):

from\_city = rute[ant, j]

to\_city = rute[ant, j + 1]

pheromone[from\_city][to\_city] += 1.0 / dist\_cost[ant]

pheromone[to\_city][from\_city] += 1.0 / dist\_cost[ant] # assuming undirected graph

# Final output

print("Final routes taken by ants:")

print(rute)

print("\nBest route found:", best\_route)

print("Cost of best route:", int(best\_cost))

**output:**

**A screenshot of a computer program

AI-generated content may be incorrect.**