In [1]:

```
# Roll No. : 3362
# Name : Shweta Santosh Phatate
# Class : TE - IT

# Assignment No. 1 - Travelling Salesman Problem

# Problem Statement :- Identify and Implement heuristic and search strategy fo
```

In [2]:

```python
# Giving the matrix as a implicit input

import sys

def nearest_neighbor(curr, unvisited, dist_matrix):
    nearest = sys.maxsize
    neighbor = None

    for city in unvisited:
        if dist_matrix[curr][city] < nearest:
            nearest = dist_matrix[curr][city]
            neighbor = city

    return neighbor, nearest

def tsp_nn(dist_matrix):
    n = len(dist_matrix)
    tour = [0] * (n+1)                          # Initialize the tour
    unvisited = set(range(1, n))                # Set of unvisited cities
    curr_city = 0                               # Starting city

    for i in range(1, n):
        next_city, dist = nearest_neighbor(curr_city, unvisited, dist_matrix)
        tour[i] = next_city
        curr_city = next_city
        unvisited.remove(next_city)

    # Return to the starting city

    tour[0] = 0

    # Calculate total cost of the tour

    cost = sum(dist_matrix[tour[i]] [tour[i+1]] for i in range(n-1) )
    cost += dist_matrix[tour[n-1]] [tour[0]]

    return tour, cost


# Giving Matrix input in form of List

dist_matrix = [
    [0,  5,  15,  4],
    [5,  0,  35,  25],
    [15, 35, 0,   30],
    [4,  25, 30,  0]
]

tour, cost = tsp_nn(dist_matrix)

print("The Distance Matrix is :\n\n", dist_matrix)
print("\n\nTour \t    : ", tour)
print("Total Cost  : ", cost)
```

```
The Distance Matrix is :

 [[0, 5, 15, 4], [5, 0, 35, 25], [15, 35, 0, 30], [4, 25, 30, 0]]


Tour        :  [0, 3, 1, 2, 0]
Total Cost  :  79
```

In [3]:

```
# Taking matrix input from user

import sys

Rows = int(input("Enter the number of Rows : "))
Columns = int(input( "Enter the number of Columns : "))

dist_matrix = []


print("\nEnter the values of Distance Matrix Row-wise : ")

for i in range(Rows):
    a =[]
    for j in range(Columns):
        a.append(int(input()))

    dist_matrix.append(a)

print( "\nThe Distance Matrix is as below : \n" )

for i in range(Rows) :
    for j in range(Columns) :
        print( dist_matrix[i][j], end = "   " )

    print()


def nearest_neighbor(curr, unvisited, dist_matrix):
    nearest = sys.maxsize
    neighbor = None

    for city in unvisited:
        if dist_matrix[curr][city] < nearest:
            nearest = dist_matrix[curr][city]
            neighbor = city

    return neighbor, nearest

def tsp_nn(dist_matrix):
    n = len(dist_matrix)
    tour = [0] * (n+1)
    unvisited = set(range(1, n))
    curr_city = 0

    for i in range(1, n):
        next_city, dist = nearest_neighbor(curr_city, unvisited, dist_matrix)
        tour[i] = next_city
        curr_city = next_city
        unvisited.remove(next_city)

    tour[0] = 0

    cost = sum(dist_matrix[tour[i]] [tour[i+1]] for i in range(n-1) )
    cost += dist_matrix[tour[n-1]] [tour[0]]

    return tour, cost
```

```
tour, cost = tsp_nn(dist_matrix)

print("\n\nTour \t    : ", tour)
print("Total Cost  : ", cost)
```

```
Enter the number of Rows : 4
Enter the number of Columns : 4

Enter the values of Distance Matrix Row-wise :
0
4
7
8
4
0
6
2
7
6
0
5
8
2
5
0

The Distance Matrix is as below :

0   4   7   8
4   0   6   2
7   6   0   5
8   2   5   0


Tour        :  [0, 1, 3, 2, 0]
Total Cost  :  18
```

In [4]:

```
import sys

def nearest_neighbor(curr, unvisited, matrix):
    nearest = sys.maxsize
    neighbor = None

    for city in unvisited:
        if dist_matrix[curr][city] < nearest:
            nearest = dist_matrix[curr][city]
            neighbor = city
    return neighbor, nearest


def tsp_nn(dist_matrix, num):
    print('\n*********************************\n\nStarting Point  : ',num)
    n = len(dist_matrix)
    tour = [0] * (n+1)                              # Initialize the tour
    unvisited = set(range(0, n))                    # Set of unvisited cities
    unvisited.discard(num)
    curr_city = num                                 # Starting city

    for i in range(1, n):
        next_city, dist = nearest_neighbor(curr_city, unvisited, dist_matrix)
        tour[i] = next_city
        curr_city = next_city
        unvisited.remove(next_city)
    tour[n] = num
    tour[0] = num                                    # Return to the starting ci

    cost = sum( dist_matrix[tour[i]] [tour[i+1]] for i in range(n-1) )
    cost += dist_matrix[tour[n-1]] [tour[0]]

    return tour, cost                                # Calculate total cost of

Rows = int(input("Enter the number of Rows : "))
Columns = int(input( "Enter the number of Columns : "))

dist_matrix = []

print("\nEnter the values of Distance Matrix i.e. one row in one line by givin

for i in range(Rows):
    a = list(map(int, input().split()))
    dist_matrix.append(a)

print("\nThe Distance Matrix is : \n")
for i in range(Rows):
    for j in range(Columns):
        print(dist_matrix[i][j], end = "  ")
    print()

for num in range(len(dist_matrix)):
    tour, cost = tsp_nn(dist_matrix, num)

    print("\nTour \t    : ", tour)
    print("Total Cost  : ", cost)
```

```
Enter the number of Rows : 4
Enter the number of Columns : 4

Enter the values of Distance Matrix i.e. one row in one line by giving space
between 2 values & after completing one row press enter and type next row in
same manner:
0 8 9 4
8 0 3 6
9 3 0 5
4 6 5 0

The Distance Matrix is :

0  8  9  4
8  0  3  6
9  3  0  5
4  6  5  0

*********************************

Starting Point  :  0

Tour        :  [0, 3, 2, 1, 0]
Total Cost  :  20

*********************************

Starting Point  :  1

Tour        :  [1, 2, 3, 0, 1]
Total Cost  :  20

*********************************

Starting Point  :  2

Tour        :  [2, 1, 3, 0, 2]
Total Cost  :  22

*********************************

Starting Point  :  3

Tour        :  [3, 0, 1, 2, 3]
Total Cost  :  20
```

In [ ]: