write a program to find minimum spanning tree of a given graph using prim's algorithm

```
import sys

def minKey(key, mstSet, n):
    min_value = sys.maxsize
    min_index = -1
    for v in range(n):
        if mstSet[v] == False and key[v] < min_value:
            min_value = key[v]
            min_index = v
    return min_index

def printMST(parent, c, n):
    totalWeight = 0
    print("Edge  Weight")
    for i in range(1, n):
        print(f"{parent[i] + 1} - {i + 1}    {c[i][parent[i]]}")
        totalWeight += c[i][parent[i]]
    return totalWeight

def primMST(c, n):
    parent = [None] * n
    key = [sys.maxsize] * n
    mstSet = [False] * n
    key[0] = 0
    parent[0] = -1
    for count in range(n):
        u = minKey(key, mstSet, n)
        mstSet[u] = True
        for v in range(n):
            if c[u][v] > 0 and mstSet[v] == False and c[u][v] < key[v]:
                parent[v] = u
                key[v] = c[u][v]
    totalWeight = printMST(parent, c, n)
    print("Total cost of the minimum spanning tree: " + str(totalWeight))

# Main code
```

```
n = int(input("Enter the number of vertices: "))
c = []
print("Enter the cost adjacency matrix:")
for i in range(n):
    c.append(list(map(int, input().split())))

primMST(c, n)
```

output:

```
Enter the number of vertices: 5
Enter the cost adjacency matrix:
0 11 9 7 8
11 0 15 14 13
9 15 0 12 14
7 14 12 0 6
8 13 14 6 0

Edge  Weight
1 - 2    11
1 - 3    9
1 - 4    7
4 - 5    6

Total cost of the minimum spanning tree: 33
```