```python
V = 6  # number of Vertices

# ****************************************************************************************** #

def findMinWeightVertex(value, MST_set):
    # Initialize variables to store the minimum weight and corresponding vertex
    minWeight = float('inf')
    vertex = -1

    # Iterate over all vertices in the graph
    for i in range(V):
        # Check if the vertex i is not yet included in the MST and its value is less than the current minimum weight
        if MST_set[i] == False and value[i] < minWeight:
            # Update the minimum weight and corresponding vertex
            minWeight = value[i]
            vertex = i

    # Return the vertex with the minimum weight
    return vertex

# ****************************************************************************************** #


def findMST(graph):
    # value array of size v initialized to infinity (This array stores weight of each vertex)
    value = [float('inf')] * V
    # Parent array of size v initialized to -1 (This array stores parent of each node in resulting MST which will be used in printing final MST)
    parent = [-1] * V
    # MST_set array of size V initialized to false that keeps track of vertices in current MST
    MST_set = [False] * V

    # Assuming start point as Node-0
    parent[0] = -1   # Start node has no parent
    value[0] = 0     # Start node has value[0] = 0 because we want this node to get pick first



    # MST will have V-1 edges , So run loop V-1 times to form complete MST
    for i in range(V - 1):
        # Select vertex with minimum weight
        U = findMinWeightVertex(value , MST_set)
        # Add this vertex U in MST
        MST_set[U] = True

        # Relax adjacent vertices (not yet included in MST)
        for j in range(V):
            # Three cases
            # 1] Edge is present from U to j
            # 2] Vertex j is not included in MST
            # 3] Edge weight is smaller than current node weight
            if graph[U][j] != 0   and   MST_set[j] == False   and   graph[U][j] < value[j]:
                value[j] = graph[U][j]
                parent[j] = U


    # Print MST
    for i in range(1, V):
        print("U->V:", parent[i], "->", i, "  wt =", graph[parent[i]][i])

# ****************************************************************************************** #


# Adjacency matrix of graph
graph = [
    [0, 4, 6, 0, 0, 0],
    [4, 0, 6, 3, 4, 0],
    [6, 6, 0, 1, 8, 0],
    [0, 3, 1, 0, 2, 3],
    [0, 4, 8, 2, 0, 7],
    [0, 0, 0, 3, 7, 0]
]

# call findMST() function to find minimum spanning tree
findMST(graph)
```