

```

In [4]: 1 import sys
2
3 class Graph:
4     def __init__(self, vertices):
5         self.V = vertices
6         self.graph = [[0 for _ in range(vertices)] for _ in range(vertices)]
7
8     def add_edge(self, u, v, weight):
9         self.graph[u][v] = weight
10        self.graph[v][u] = weight
11
12    def min_key(self, key, mst_set):
13        min_val = sys.maxsize
14        min_index = -1
15
16        for v in range(self.V):
17            if key[v] < min_val and not mst_set[v]:
18                min_val = key[v]
19                min_index = v
20
21        return min_index
22
23    def prim_mst(self):
24        key = [sys.maxsize] * self.V
25        parent = [-1] * self.V
26        mst_set = [False] * self.V
27
28        key[0] = 0
29        parent[0] = -1
30
31        for _ in range(self.V):
32            u = self.min_key(key, mst_set)
33            mst_set[u] = True
34
35            for v in range(self.V):
36                if self.graph[u][v] > 0 and not mst_set[v] and key[v] > self.graph[u][v]:
37                    key[v] = self.graph[u][v]
38                    parent[v] = u
39
40        return [(parent[i], i) for i in range(1, self.V)]
41
42    # Example usage:
43    g = Graph(5)
44    g.add_edge(0, 1, 2)
45    g.add_edge(0, 3, 6)
46    g.add_edge(1, 2, 3)
47    g.add_edge(1, 3, 8)
48    g.add_edge(3, 4, 5)
49    g.add_edge(2, 4, 7)
50    g.add_edge(3, 4, 9)
51
52    mst = g.prim_mst()
53    print("Edges in the Minimum Spanning Tree:")
54    for u, v in mst:
55        print(f"{u} - {v}")
56

```

57

Edges in the Minimum Spanning Tree:

0 - 1

1 - 2

0 - 3

2 - 4

In []:

1