94) Minimum Spanning Tree

CODE:

```
from collections import defaultdict import heapq
 def prim_mst(graph):
mst = []
    visited = set()
    start_node = next(iter(graph))
visited.add(start_node)
    edges = [(cost, start_node, to)
 for to, cost in graph[start_node]]
     heapq.heapify(edges)
     while edges:
         cost, frm, to = heapq.heappop(edges)
        if to not in visited:
             visited.add(to)
            mst.append((frm, to, cost))
             for next_to, next_cost in graph[to]:
                 if next_to not in visited:
                    heapq.heappush(edges, (next_cost, to, next_to))

    return mst

graph = {
    'A': [('B', 2), ('C', 3)],
    'B': [('A', 2), ('C', 1), ('D', 1)],
    'C': [('A', 3), ('B', 1), ('D', 2)],
    'D': [('B', 1), ('C', 2)]
}
mst = prim_mst(graph)
print(mst) OUTPUT:
```

TIME COMPLEXITY : O(nlogn)