**D.7** Vervollständigen Sie den Algorithmus `kruskal` unter Verwendung der *UnionFind*-Datenstruktur.

```python
# MINIMUM SPANNING TREE

# Vervollstaendigung von kruskal mit UnionFind
# verwendet die Klasse UnionFind
from vl.lek07.union_find import UnionFind

def kruskal(G):
    m = len(G)
    # Kantenmenge ohne doppelte Eintraege, in O(m+k)
    E = { (G[u][v],frozenset({u,v})) for u in range(m) for v in G[u] }
    # Ergebnis: Spannbaum und dessen Kosten
    T = set()
    c = 0

    # UnionFind-Datenstruktur
    uf = UnionFind(m)

    # Kanten nach Kosten sortiert durchlaufen
    for (cost,(u,v)) in sorted(E):
        contains_u = uf.find(u)
        contains_v = uf.find(v)

        # Überprüfe ob Knoten in gleicher Menge
        # -> falls nein, dann kreisfrei
        if  contains_u != contains_v:
            # Fasse Mengen zusammen
            uf.union(contains_u, contains_v)
            T.add(frozenset((u,v)))
            c += cost
    return T, c

```