Input: Graph $G = \langle V, E \rangle$
Output: Edge set T forming a min. span tree
▷ Here, the $MakeSet, FindSet, Union$ make use of a disjoint-set structure.

```
procedure KRUSKAL(G)
    T ← ∅                                          ▷ set of edges
    S ← ∅                                          ▷ set of disjoint sets

    for v ∈ V do
        S ← S ∪ MakeSet(v)

    for (u, v) ∈ G.E (ordered by weight) do
        set_u ← FindSet(u)
        set_v ← FindSet(v)
        if set_u ≠ set_v then
            T ← T ∪ {(u, v)}
            Union(set_u, set_v)
return T
```

Input: Graph $G = \langle V, E \rangle$
Output: E of G containing only the edges belonging to a min. span tree
▷ Connectivity can be checked via graph traversal algorithms, namely BFS or DFS.

```
procedure REVERSE_DELETE(G)
    Sort G.E in descending order
    i ← 0
    while i < |E| do
        e ← E[i]
        Delete e                                  ▷ i is now at the next edge
        if G not connected then
            E[i] ← e                              ▷ Re-add edge
            i ← i + 1                             ▷ Move on to next edge
return G.E
```