PRIM's MST algorithm

- Start with an arbitrary vertex r
- Grow MST by repeatedly adding the smallest edge connecting a vertex in the tree with a vertex not in the tree
- To find the smallest edge we use a priority queue containing the *vertices* not in the tree yet:
  - The key/priority of a vertex v is the weight of the smallest edge connecting v to the tree (impl.note: we also maintain a pointer from v to v's location in the priority queue)
  - For each vertex v in the PQ we store the edge that connects it to the tree; we call the other vertex of this edge by $pred(v)$

PRIM
/* initialize */
Pick arbitrary vertex r
For each vertex $u \in V, u \neq r$: PQ.INSERT$(u, prio = \infty)$
For vertex r: PQ.INSERT$(r, prio = 0)$, $pred(r) = NULL$
/* main loop */
WHILE PQ not empty

    u = PQ.DELETE-MIN()

    For each $(u, v) \in E$:

        IF $v \in PQ$ and $w_{uv} <$ prio(v):
            pred$[v] = u$
            PQ.DECREASE-KEY$(v, prio = w_{uv})$

Output the edges $(u, pred(u))$ as the MST.

Kruskal's MST algorithm

KRUSKAL
/* initialize */
For each vertex $v \in V$: MAKE-SET(v)
Sort edges of E in increasing order by weight
/* main loop */
FOR each edge $e = (u, v) \in E$ in order of weight:

    IF FIND-SET$(u) \neq$ FIND-SET(v) THEN

        output edge e as part of MST
        UNION-SET(u, v)