As always, sit with a partner and work through these together.

Activity 1: Prim-Jarnik

while PQ not empty:
    N = removeMin and circle it
    connect N to MST (except start node)
    for each neighbor M of N:
        update distance to M if smaller

Activity 2: Runtime of Prim-Jarnik's

Find the runtime of Prim-Jarnik's Algorithm based on the following pseudocode by filling in the runtime for each appropriate line of code.

```python
function prim(G):
    for all v in V:
        v.cost = ∞
        v.prev = null
    source = a random v in V
    source.cost = 0
    MST = []
    PQ = PriorityQueue(V)
    while PQ is not empty:
        v = PQ.removeMin()
        if v.prev != null:
            MST.append((v, v.prev))
        for all incident edges (v,u) of v:
            if u.cost > (v,u).weight:
                u.cost = (v,u).weight
                u.prev = v
                PQ.replaceKey(u, u.cost)
    return MST
```

Runtime of Prim-Jarnik's: __________________________