Kruskal(G):

Define set S(v) = {v} for each vertex v in G  
Initialize dict D that contains all edges in G with edges as keys and weights as values

Sort edges by their weights in ascending order

Create empty set T that will contain edges and weights of the MST

For each edge (u, v) in dict D  
 Let S(u) be the set containing u, and S(v) be the set containing v

If S(u) != S(v) then

Add edge (u, v) and weight w to T

Merge S(u) and S(v) into one set

Delete S(u) and S(v)

Return tree T

Heap is complete when tree is filled completely and again filled with left to right.

Remove items based on priority-cost,vertex lower to higher

1.heap is a tree-directed acyclic tree

2. it’s a complete heap

3. it’s a max heap

4. multiplying with 2 to get the index is because of base 2