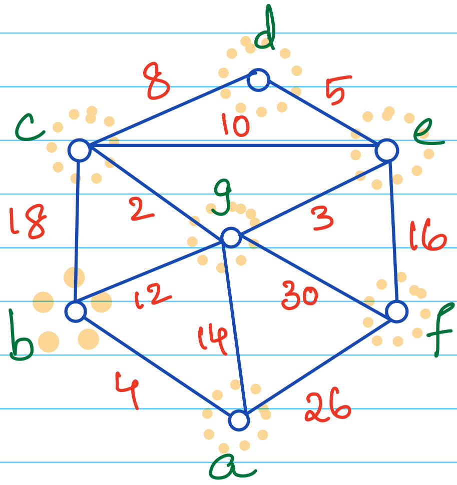
Prim's algorithm

Maintain a spanning forest, and keep growing one tree of the forest using the greedy choices



Starting from a

(a,b), (b,g), (g,c)

(g,e), (e,d), (e,f)

Implementation details:

- Use priority queue to maintain the min-edge weights to the tree that is growing

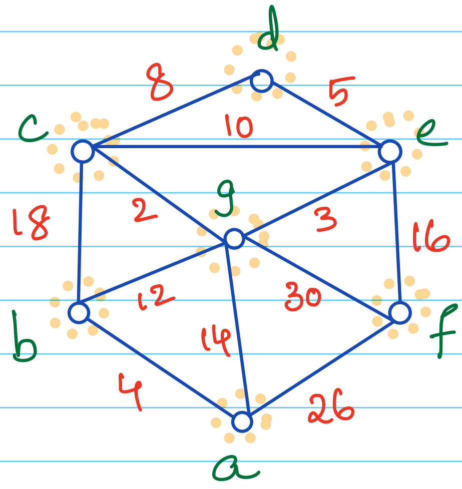
  (Similar to Dijkstra)

Running time: O(E log V) (Binary heaps)

O(V log V + E) (Fibonacci heaps)

Kruskal's Algorithm

Maintain a spanning forest $\mathcal{F}$, and find the least wt edge that connects two trees in the forest



Order of edges added

(c,g), (g,e), (a,b), (e,d)

(b,g), (e,f)

Implementation details:

① Sort the edges in increasing order of edge weights – $O(E \log E)$

② For each $e \in E$, check if the component labels of the end-points are different – $O(1)$

- If so, update the forest by adding the edge
- Change the component labels – $O(V)$

$\Rightarrow$ Running time: $O(V^2)$

(Need to update labels for $O(V)$ edges that belong to the MST)