KRUSKAL ALGORITHM

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

I] Introduction

- It uses greedy approach to find MST.

Property:

- Set A is a forest whose vertices are all of those of the given graph.
- The safe edge added to A is always least weight edge in the graph that connects two distinct components.

Working:

- Algorithm starts with set A which is forest of |V| trees with every tree having one single vertex.
- At each step it finds a safe edge to add to the growing forest which should satisfy 2 condition.
  (i) It connects two distinct trees in forest
  (ii) It should least weight edge.
- "This strategy qualifies as greedy because at each step it adds to the forest an edge of least possible weight".

Imp

II] Algorithm:

I]

```
MST - KRUSKAL (G, w)
{
    1.  A = ϕ                          # Empty spanning tree

Creating forest of (v) trees →
    2.  for each v ∈ G.V
    3.      MAKE-SET (v)               # Creates a new set.

    4.  Sort the edges of G.E into nondecreasing order by weight w.

Selecting least wt edges →
    5.  for each edge (u,v) ∈ G.E, taken in nondecreasing order by weight.
        {
edge separates connecting 2 different components →
    6.          if FIND-SET(u) ≠ FINDSET(v)
                {
    7.              A = A U {(u,v)}
    8.              UNION (u,v)
                }
        }
    9.  return A.
}
```

2] $\boxed{\text{MAKE\_SET}(x)}$: Creates a new set with only one member i.e x.

3] $\boxed{\text{FIND\_SET}(x)}$: Returns a pointer to the representative of the set containing x.

4] **UNION (x, y) :** unites dynamic sets that contain x and y into a new set.
In practice, element of one set is absorb into another set.

III] **Analysis**

- Depends on how we implement the disjoint-set data structure.

- It can be implemented using. Arrays, min-Heap or linked list representation of disjoint sets.

- Creating forest i.e line 2 & 3 will take O(V) time, since MAKE_SET (V) in any implementation will take constant time.

- line 4 i.e sorting will take O(E log E).

	ARRAY	min-Heap	linked list
FIND_SET & UNION OPERATION	O(|V|)	O(log|V|)	O(log|V|)

- Running Time = Sorting + edges * (FIND_SET + UNION OPERATION)
complexity

For Arrays,

Running Time complexity $= O(E \log E + |E||V|)$

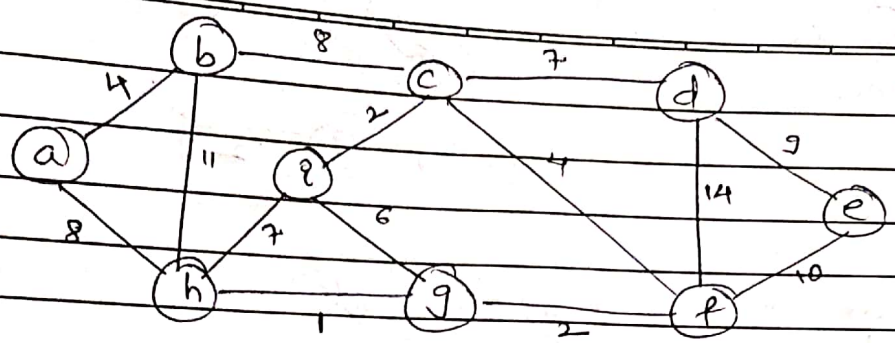
$= O(|E||V|)$

For Min Heap or Linked list,

Running Time complexity $= O(|E| \log |E| + |E| * \log |V|)$

$= O(|E| \log |E|)$

Ex 1



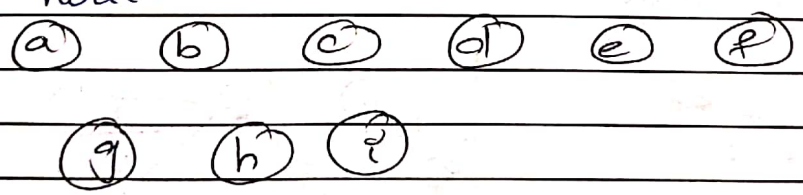
Find MST using Kruskal.

$|V| = 9$ $|E| = 14$

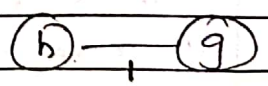
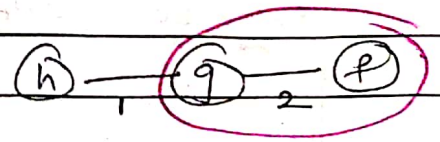
Step 1: Create the Edge table in non-decreasing order.

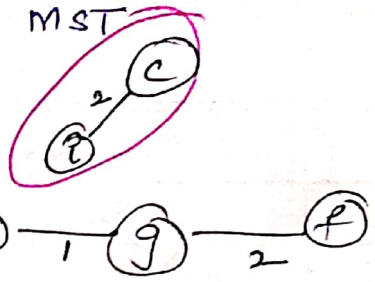
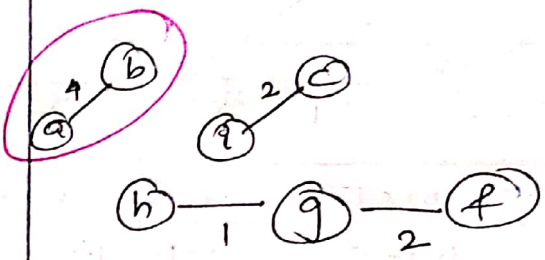
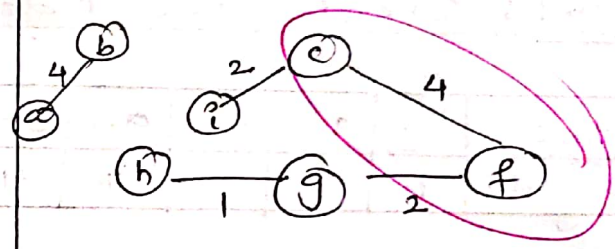
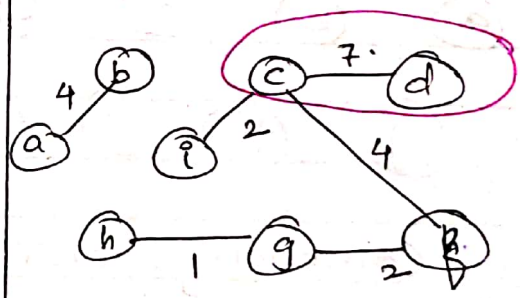
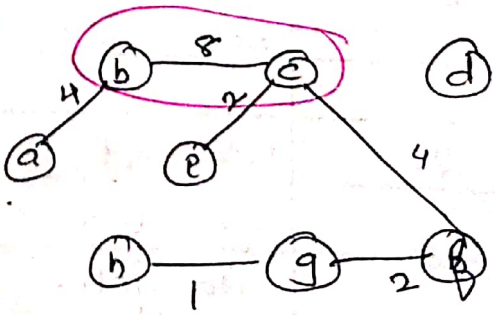
Edges	hg	gf	ci	ab	cf	ig	cd	ih	bc	ah	de	ef	bh	df
Weights	1	2	2	4	4	6	7	7	8	8	9	10	11	14

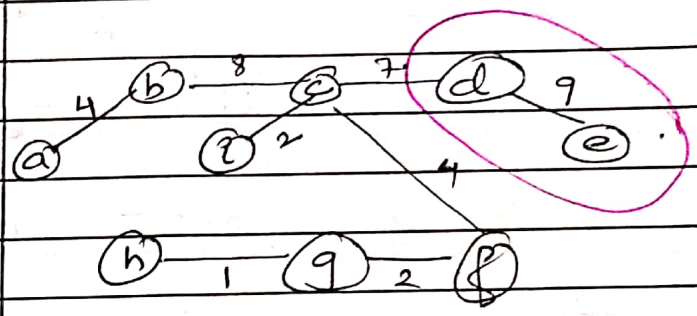
Step 2: Create Forest of $|V| = 9$ trees with one node.



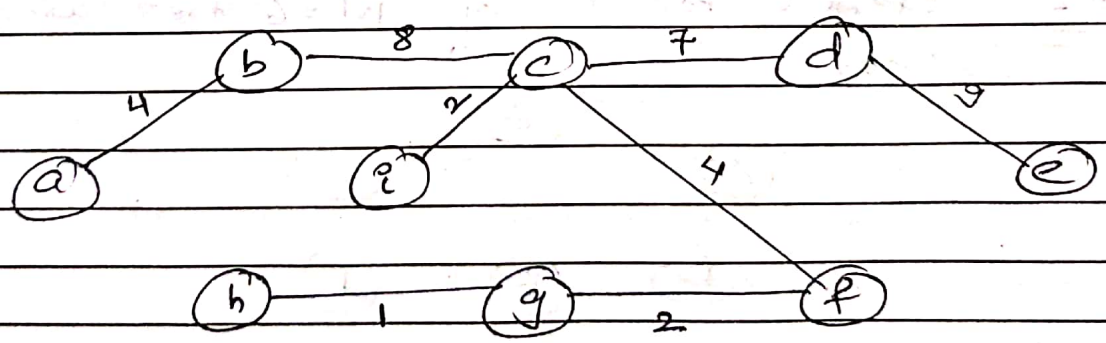
Step 3: Constructing MST

Iteration	Edge	MST
1	Edge hg is selected.	
2.	Edge gf is selected since No cycle	

Iteration	Edge	MST
3.	Edge ci is selected	
4.	Edge ab is selected.	
5.	Edge ef is selected.	
6.	Edge ig is not selected, since it make a circuit.	
7.	Edge cd is selected.	
8.	Edge gh is not selected, since it makes a circuit	
9.	Edge bc is selected.	

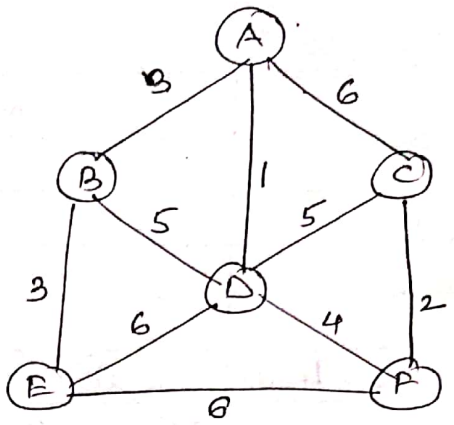
Iteration	Edge	MST
10.	Edge ah is not selected.	
11.	Edge de is selected.	
12.	Edge ef Not selected	
13.	Edge bh Not selected	
14.	Edge df Not selected.	

∴



Cost of MST = 37

Ex



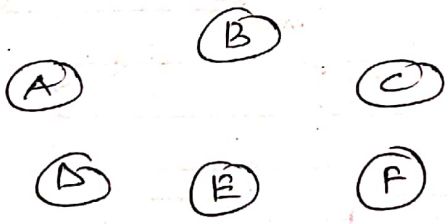
Find MST using Kruskal.

⇒ $|V| = 6$ $|E| = 10$

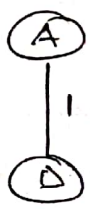
Step 1 : Create the Edge table in non-decreasing order

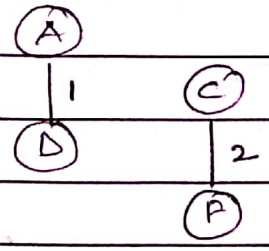
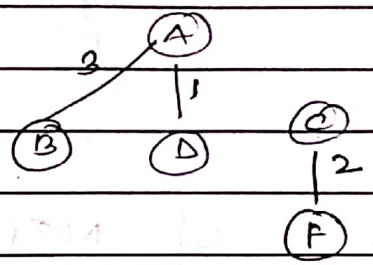
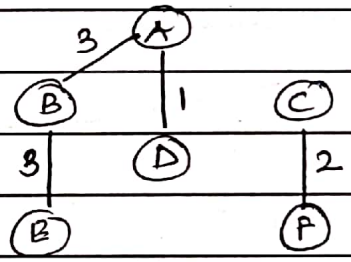
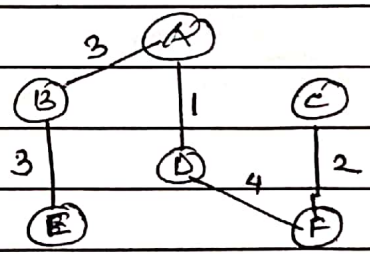
Edges	AD	CF	AB	BE	DF	BD	DC	AC	DE	EF
Weights.	1	2	3	3	4	5	5	6	6	6

Step 2 : Create Forest of $|V| = 6$ trees with one node.

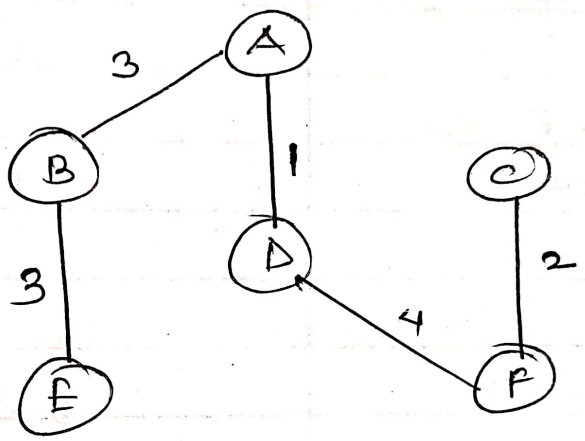


Step 3 : Constructing MST

Iteration	Edge	MST
1	Edge AD is selected	

Iteration	Edge	MST
2.	Edge CF is selected	
3.	Edge AB is selected.	
4.	Edge BE is selected	
5.	Edge DF is selected.	
6.	Edge BD is not selected	
7.	Edge DC, AC, DE and EF is not selected.	

∴ MST is



∴ Cost of MST is : 13