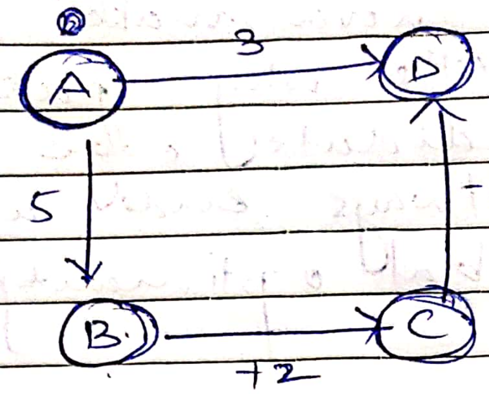
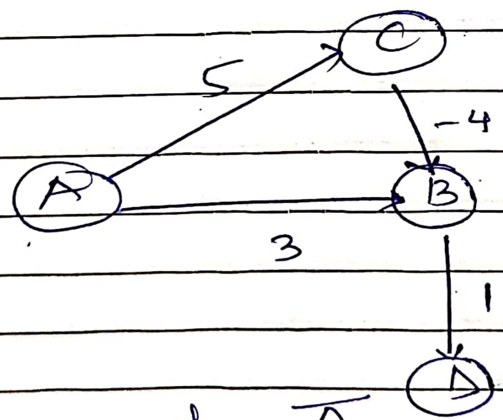
Ex. with Negative edges

1]



	d	π
A	0	–
B	~~∞~~ 5	~~–~~ A
C	~~∞~~ 7	~~–~~ B
D	~~∞~~ 3	~~–~~ A

2]



	d	π
A	0	–
B	~~∞~~ ~~3~~ 1	~~–~~ ~~A~~ C
C	~~∞~~ 5	~~–~~ A
D	~~∞~~ 4	~~–~~ B

Dijkstra fails.

MINIMUM SPANNING TREE

EXPERIMENT : No.

PAGE No.

DATE

(GREEDY ALGORITHM)

I] What is spanning Tree?

- Given an undirected and connected graph $G = (V, E)$,

- "A spanning tree of the graph G is a tree that spans G (i.e includes every vertex of G) and is a subgraph of G (i.e every edge in spanning tree belongs to G)".

- A spanning tree of G is a subgraph T, that is,

  i) Connected.
  ii) Acyclic
  iii) Includes all of the vertices.

II] What is MST?

- Cost of spanning tree is the sum of the weights of all the edges in the tree.
  - There can be many spanning trees.

- MST is the spanning tree with minimum cost.
- There can be many MST,

Teacher's Sign.:

III] Applications

1. Dithering.
2. Medical Image processing.
3. Cluster Analysis
4. Find Road networks in satellite & aerial imagery.
5. Network design
6. Approximation algorithm for NP-hard problems.
7. Hand writing Recognition.
8. Image segmentation.

IV] Generic MST

Problem definition :

Given a connected, undirected graph $G(V,E)$ with weight $w(u,v) \in E$, we Need to find a MST for G.

Both kruskal & prim's algorithm follows greedy approach.

This greedy approach is captured by following generic method, which grows the MST One edge at a time.

Generic method has set A → set of edges in MST
Prior to every iteration, A, subset of some MST.

EXPERIMENT : No. PAGE No. DATE

At each iteration, an edge (u,v) is added to set A without voilating any constraints.
i.e. $A \cup \{(u,v)\}$ is also a subset of MST.

- Such an edge is known as safe edge

Algorithm

GENERIC_MST (G, w)
{

1. $A = \phi$
2. while A does not form a spanning tree
   {
3. find an edge (u,v) i.e safe edge.
4. $A = A \cup \{(u,v)\}$
   }
5. return A.

V] Total Number of ~~MST~~ Spanning Tree.

- For a graph G, with |v| vertices & |E| edges,

- MST will have |V| vertices & |V|-1 edges.

- If 'm' is the no. of cycles then.

Total No. of spanning Tree $= {}^{|E|}C_{|V|-1} - m$.

EXPERIMENT : No.

PRIM'S ALGORITHM

PAGE NO.
DATE

I] INTRODUCTION :

- Prim's algorithm uses greedy approach to find the minimum spanning tree (MST)
- It operates much like dijkstra's algorithm for finding shortest path.

- Property of Prim's Algo

"Edges in set A will always form a single tree".

- Working

- Algorithm starts tree with an arbitrary vertex 'r' as root vertex and grows until the tree spans all the vertices in V.

- At each step, it adds a light edge to the tree A, that connects A to the isolated vertex (ie one on which no edge in A was incident).
  - This rule adds only edges that are SAFE.
- Once algorithm terminates, edges in A will form MST.

Imp → - "This strategy qualifies as greedy because at each step it adds to the tree an edge that contributes the minimum amount possible to the tree's weight"

Sundaram

Teacher's Sign.:

II] Algorithm

A) Notations :

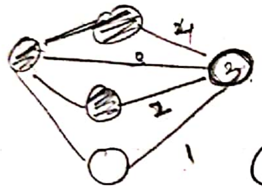
(i) $G(V, E)$ → graph with set of vertices V and set of edges E.

(ii) r → root of MST

(iii) for each vertex $v \in V$,

ⓐ $v.key$ → minimum wt. of any edge that is connecting to a vertex in the tree

$\boxed{v.key = \infty}$ if there is no such incident edge.



ⓑ $v.\pi$ → parent of v in tree

B) Algorithm - MST-PRIMS

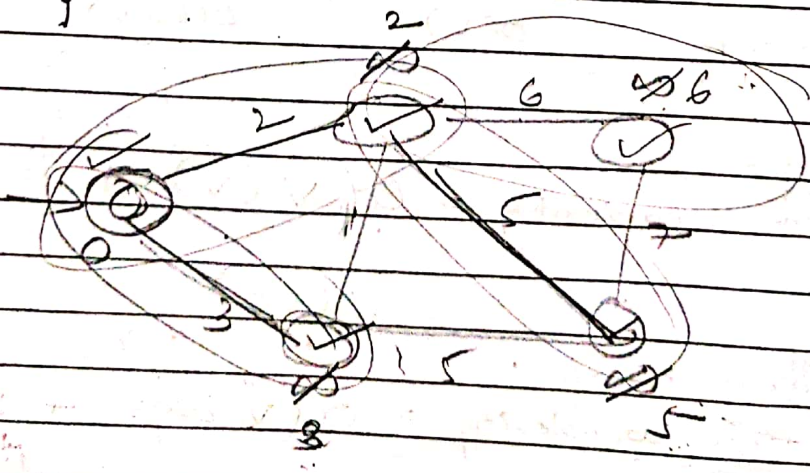
i/p : Graph G and root vertex r.

o/p : Set $A = \{(v, v.\pi) : v \in V - \{r\}\}$

DATE

MST_PRIMS (G, r)

```
{
    for each v ∈ V
    {
            v.key = ∞ .
            v.π   = NIL
    }
    r.key = 0
    Q = G.V          # Minimum priority queue.
    while Q != φ
    {
            u = EXTRACT_MIN (Q)
            for each v ∈ G.Adj [u]
            {
                    if v ∈ Q and w(u,v) < v.key
                    {
                            v.π = u
                            v.key = w(u,v)
                    }
            }
    }
}
```



Teacher's Sign.:

III] Analysis

- Algo maintains MIN-PRIORITY Q by using
  (i) INSERT (line 5)
  (ii) EXTRACT-MIN (line 7)
  (iii) DECREASE-KEY (line 11)

- Line 6, $\boxed{\text{while}}$ loop will executes once per vertex i.e $\boxed{|V|}$ times.
- Line 8, $\boxed{\text{for}}$ loop will executes |E| times altogether.

- Line 9, Testing for membership in Q can be done in constant time i.e O(1) by keeping a bit for each vertex that tells whether vertex is in Q or not, and updating it once it is removed from Q.

- Like Dijkstra's, Running time depends on how min priority queue is implemented.

Data structure	Insert	DecreaseKey	ExtractMin
Array	O(1)	O(1)	O(V)
Binary Heap	O(V)	O(log V)	O(log V)

- For Array,
  Running time complexity $= O(\underset{\text{line 6}}{V}\,(\underset{\text{line 7}}{V}) + \underset{\text{lines 8 - 11}}{E})$

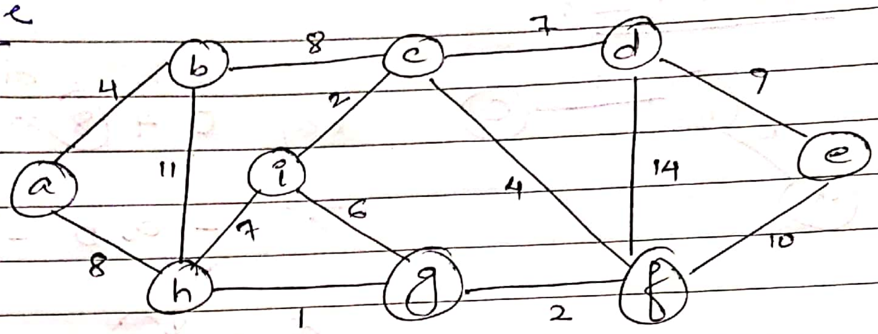
  $= O(V^2)$

- For Binary Heap.
  Running time complexity $= O(\underset{\text{line 6}}{V} \underset{\text{line 7}}{\log V} + \underset{\text{line 8 to 10}}{E} \underset{\text{line 11}}{\log V})$

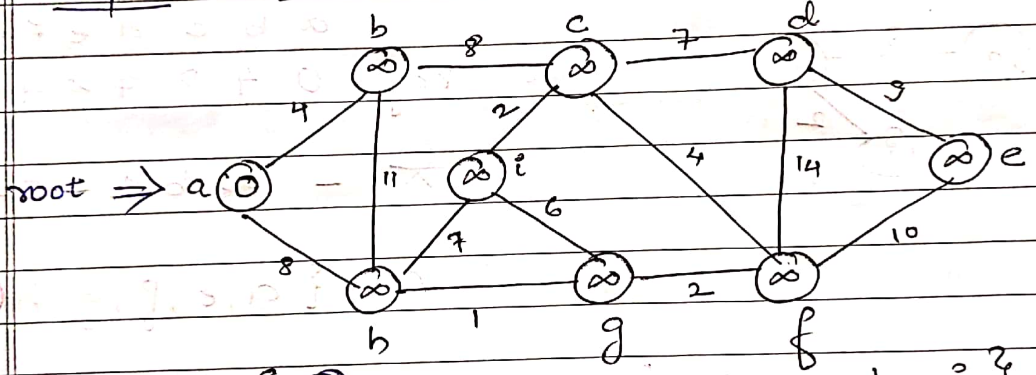
  $= O(E \log V)$

IV Example

I



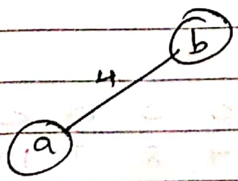
⇒ Step I : Initialization



Q = { a , b , c , d , e , f , g , h , i }

Step II : PRIM'S ALGORITHM

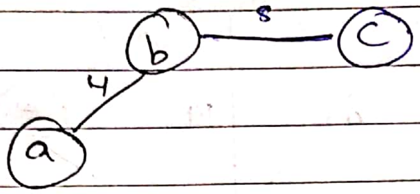
Iteration 1 :



	a	b	c	d	e	f	g	h	i
key	0	4	∞	∞	∞	∞	∞	8	∞
π	–	a	–	–	–	–	–	a	–

Q = { b , c , d , e , f , g , h , i }

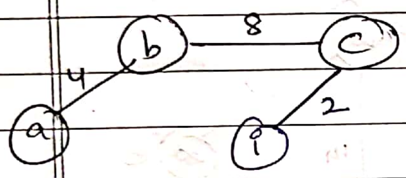
Iteration 2 :



	a	b	c	d	e	f	g	h	i
key	0	4	8	∞	∞	∞	∞	8	∞
π	-	a	b	-	-	-	-	a	-

Q = {c, d, e, f, g, h, i}

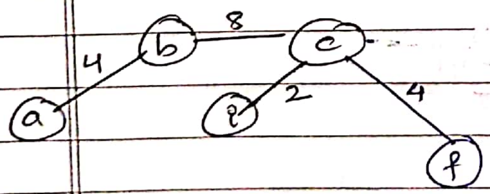
Iteration 3 :



	a	b	c	d	e	f	g	h	i
key	0	4	8	7	∞	4	∞	8	2
π	-	a	b	c	-	c	-	a	c

Q = {d, e, f, g, h, i}

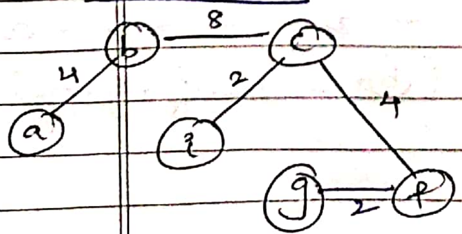
Iteration 4 :



	a	b	c	d	e	f	g	h	i
key	0	4	8	7	∞	4	6	7	2
π	-	a	b	c	-	c	i	i	c

Q = {d, e, f, g, h}

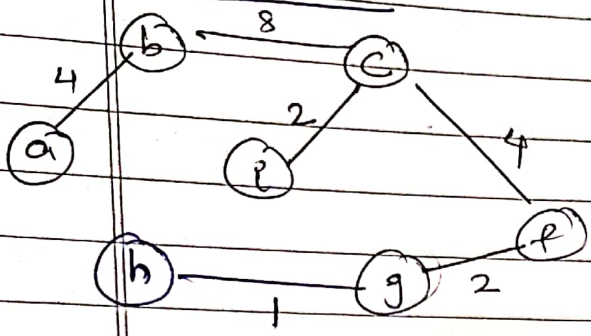
Iteration 5



	a	b	c	d	e	f	g	h	i
key	0	4	8	7	10	4	2	7	2
π	-	a	b	c	f	c	f	i	c

Q = {d, e, g, h}

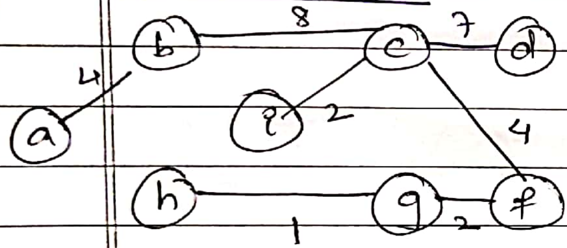
Iteration 6



	a	b	c	d	e	f	g	h	i
Key	0	4	8	7	10	4	2	1	2
π	-	a	b	c	f	c	f	g	c

$Q = \{d, e, h\}$

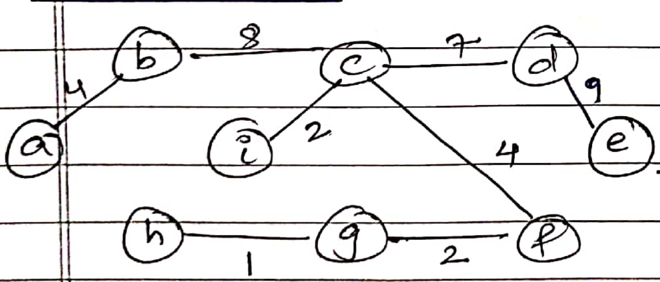
Iteration 7



	a	b	c	d	e	f	g	h	i
Key	0	4	8	7	10	4	2	1	2
π	-	a	b	c	f	c	f	g	c

$Q = \{d, e\}$

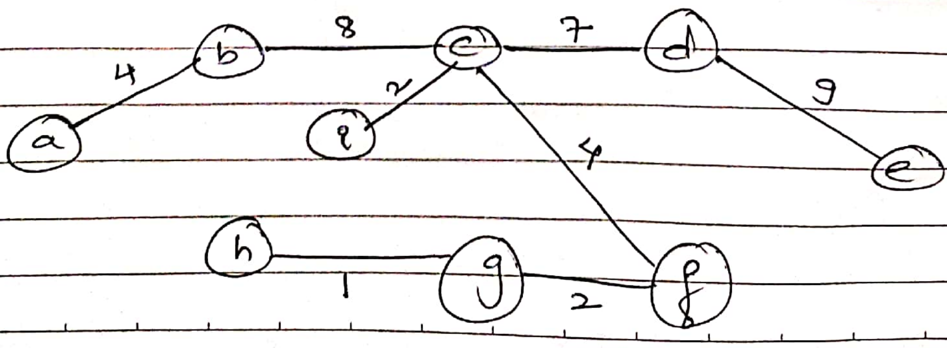
Iteration 8



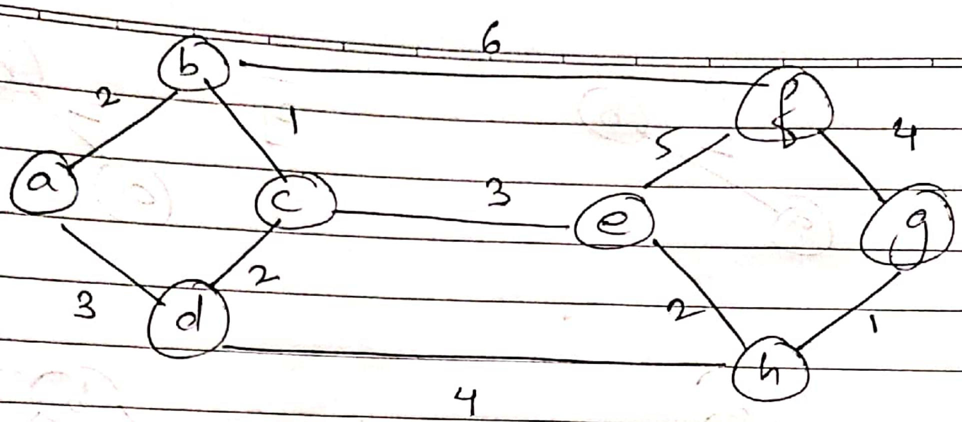
	a	b	c	d	e	f	g	h	i
Key	0	4	8	7	9	4	2	1	2
π	-	a	b	c	~~f~~ d	c	f	g	c

$Q = \{e\}$

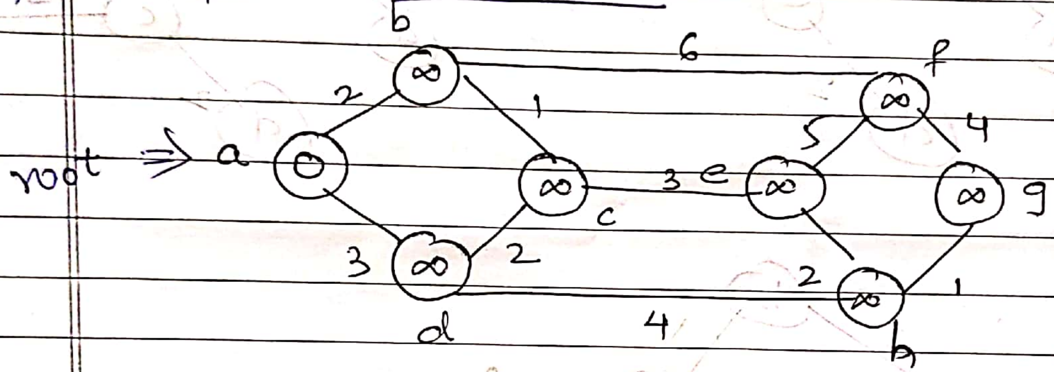
$\therefore$ Final MST is



2]



⇒ Step I Initialization

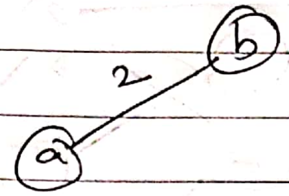


Q = {a, b, c, d, e, f, g, h}

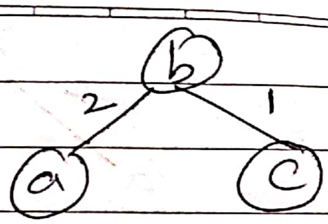
Step II PRIMS ALGORITHM

	key								π							
	a	b	c	d	e	f	g	h	a	b	c	d	e	f	g	h
Iteration 1	0	2	∞	3	∞	∞	∞	∞	–	$\boxed{a}$	–	a	–	–	–	–
Iteration 2	0	2	1	3	∞	6	∞	∞	–	a	$\boxed{b}$	a	–	b	–	–
Iteration 3	0	2	1	2	3	6	∞	∞	–	a	b	$\boxed{c}$	c	b	–	–
Iteration 4	0	2	1	2	3	6	∞	4	–	a	b	c	$\boxed{c}$	b	–	d
Iteration 5	0	2	1	2	3	5	∞	2	–	a	b	c	c	e	–	$\boxed{e}$
Iteration 6	0	2	1	2	3	5	1	2	–	a	b	c	c	e	$\boxed{h}$	e
Iteration 7	0	2	1	2	3	4	1	2	–	a	b	c	c	$\boxed{g}$	h	e
Iteration 8	No change								No change							

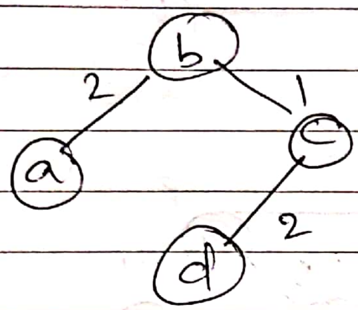
1]



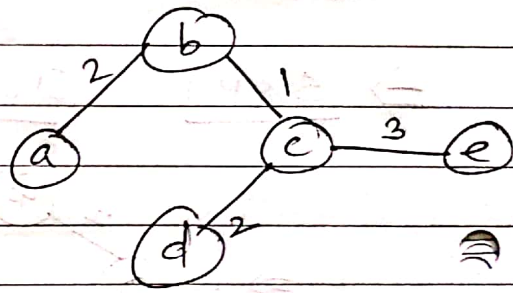
2]



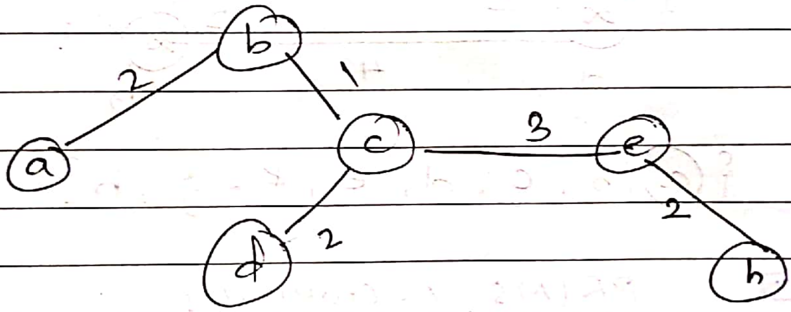
3]



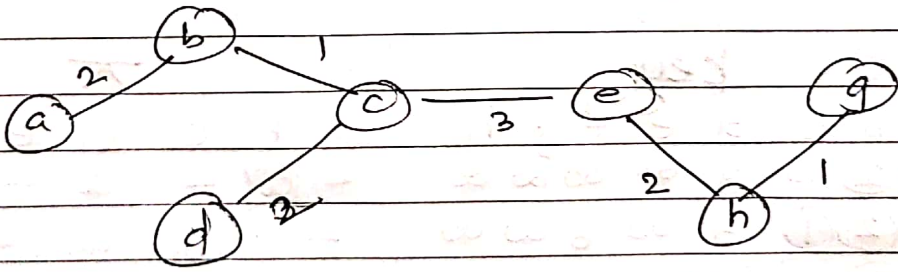
4]



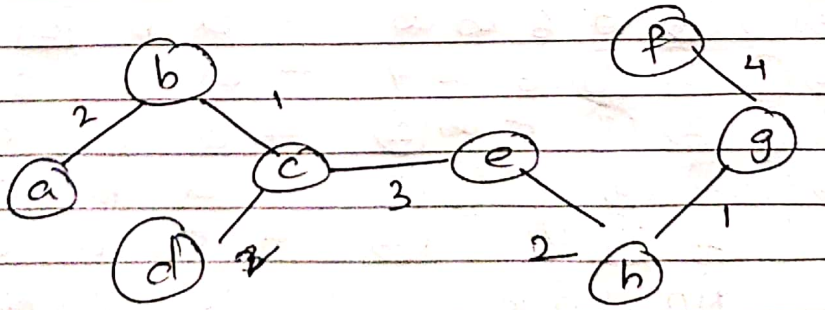
5]



6]



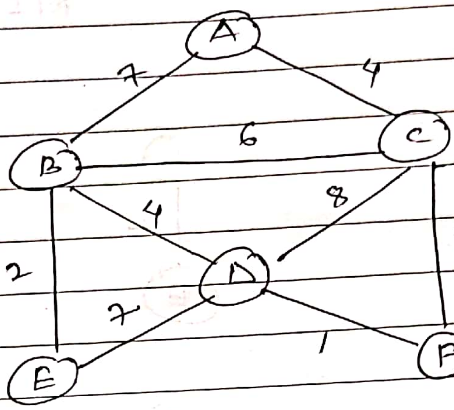
7]



Cost of MST = 15

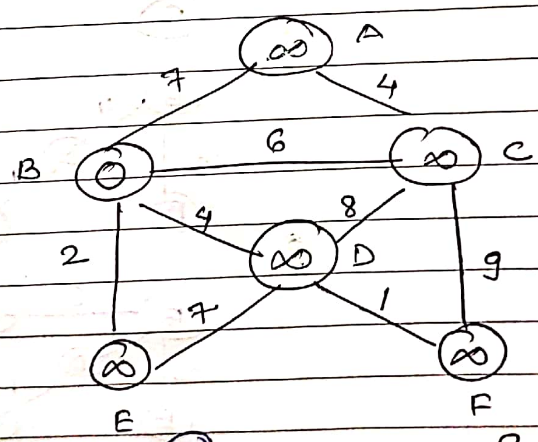
Example

3



Step 1 : Initialization

root ⇒

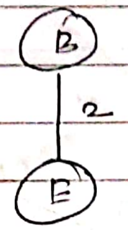


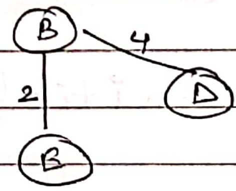
$Q = \{ A, B, C, D, E, F \}$

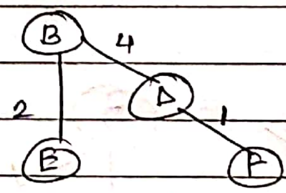
Step 2 : PRIM's ALGO.

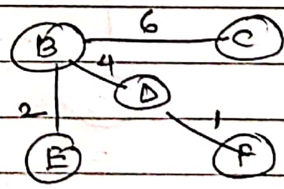
	Key A	Key B	Key C	Key D	Key E	Key F	π A	π B	π C	π D	π E	π F
Iteration 1	7	0	6	4	2	∞	B	–	B	B	$\boxed{B}$	–
Iteration 2	7	0	6	4	2	∞	B	–	B	$\boxed{B}$	B	–
Iteration 3	7	0	6	4	2	1	B	–	B	B	B	$\boxed{D}$
Iteration 4	7	0	6	4	2	1	B	–	$\boxed{B}$	B	B	D
Iteration 5	4	0	6	4	2	1	$\boxed{C}$	–	B	B	B	D
Iteration 6	No change						No change					

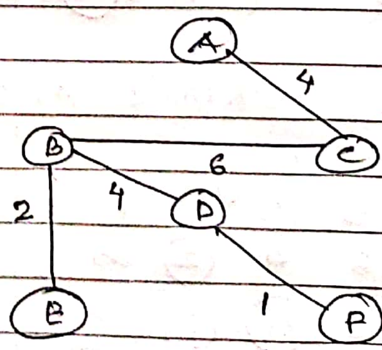
Iteration MST

1. 

2. 

3. 

4. 

5. 

∴ Cost of MST is 17