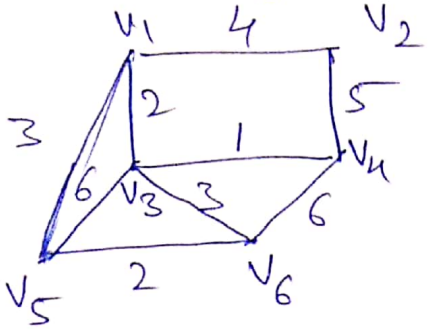
Prim's Algorithm to find Minimal Spanning Tree of a Graph

Let G be a connected weighted graph with vertices $V_1, V_2 \ldots, V_n$. Arrange the given weights of the edges of G in an $(n \times n)$ table. The wts of the non-existent edges are taken as very large. We take it as ∞. Diagonal elements are empty. Following are the steps of Prim's algorithm —

1. Start from the vertex V_1 and connect it to the vertex with smallest entry in row 1 of the table. Let this vertex be V_k.
2. Connect the subgraph with vertices V_1 & V_k to a vertex with smallest entry in rows 1 & K. This vertex must be different from V_1 & V_k. Let it be V_j.
3. Connect the subgraph with vertices V_1, V_k & V_j with the vertex with smallest entry in rows $1, k$ & j.
4. Continue this process until all (n) vertices are connected by $(n-1)$ edges.

Ex→ Consider the graph →



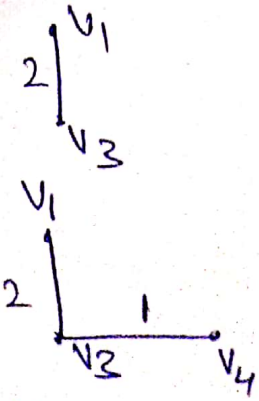
To find minimal spanning tree.
Total vertices = 6 = n
S.T. will have edges
= 6 − 1 = 5

We use Prim's algorithm—
First we make following table → (matrix)
0 = ∞

	v_1	v_2	v_3	v_4	v_5	v_6
v_1	—	4	2	∞	3	∞
v_2	4	—	∞	5	∞	∞
v_3	2	∞	—	1	6	3
v_4	∞	5	1	—	∞	6
v_5	3	∞	6	∞	—	2
v_6	∞	∞	3	4	2	—

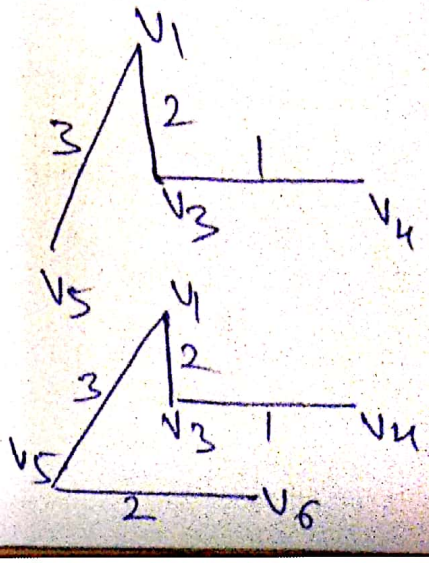
1. Start with vertex v_1. Choose smallest entry in row 1. It is (v_1, v_3). Draw (v_1, v_3).

2. Choose smallest entry in rows 1 & 3. It is (v_3, v_4) & it does not make a circuit with (v_1, v_3). Draw (v_3, v_4)

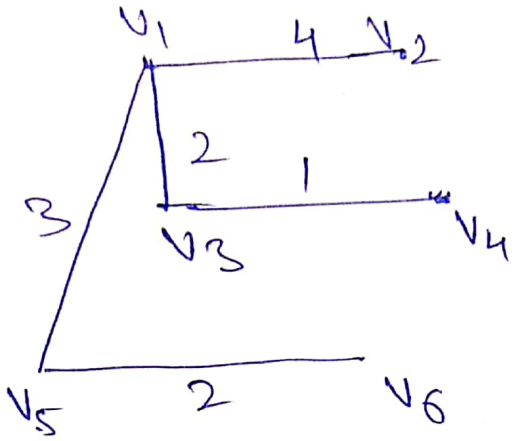


3. Choose smallest entry in rows (1, 3, 4). It is (v_1, v_5) or (v_3, v_6). We choose (v_1, v_5). Draw (v_1, v_5)

4. Chk rows (1, 3, 4, 5). Choose (v_5, v_6). Draw (v_5, v_6)



5. Chk edges (1, 3, 4, 5, 6).
Choose (v_1, v_2). Draw (v_1, v_2).



This is not S.T. as it has all the vertices and 5 edges.

Wt. of S.T. = 4 + 2 + 1 + 3 + 2

= 12 units.
