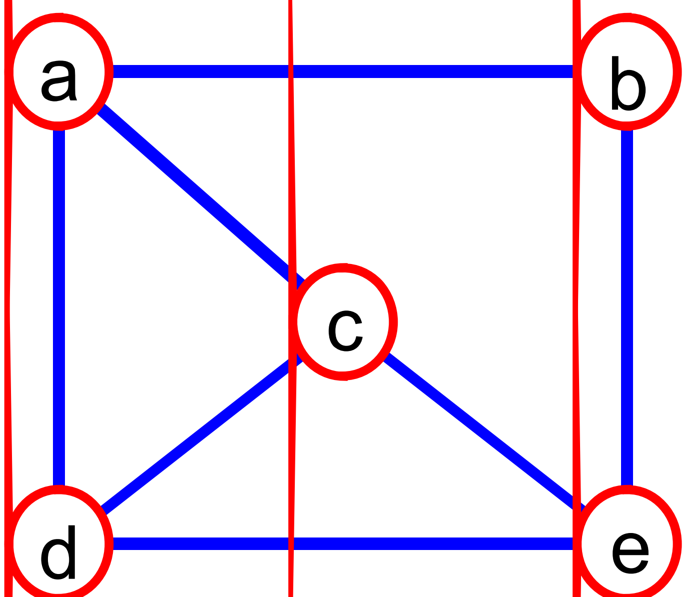
Applications of graphs

- **SHORTEST PATH ALGORITHMS**
  - **DIJKSTRA'S ALGORITHM**
    (SINGLE SOURCE SHORTEST PATH)
  - **FLOYD'S ALGORITHM / FLOYD WARSHALL'S ALGORITHM**
    (ALL PAIR SHORTEST PATH)



Dijkstra's Algorithm

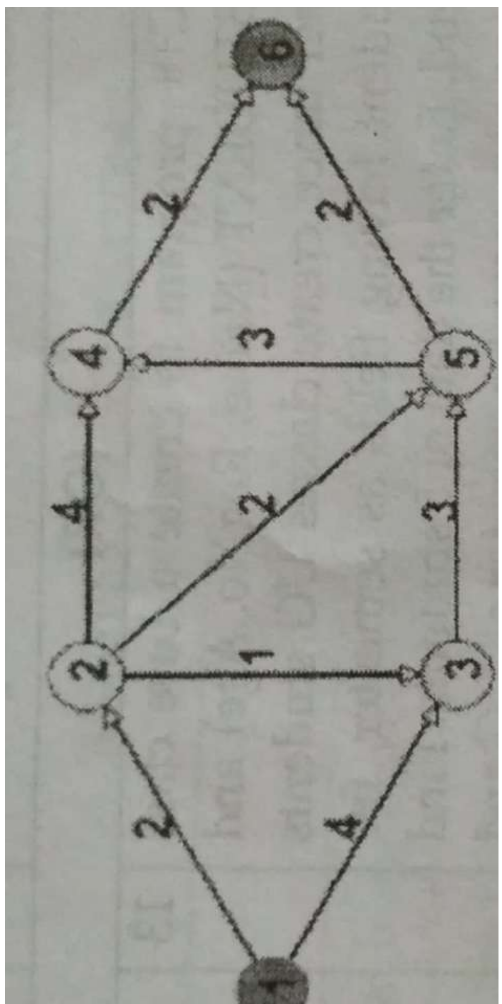
Used to find the **shortest path between source node and every other node**

Vertex	*Known*	d_v	p_v
A			
B			
C			
D			
E			
F			
G			
H			

Known* – *T / F

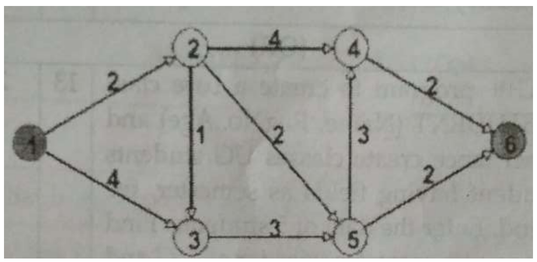
d_v - ***Weight of the shortest edge connecting v to a known vertex***

p_v - ***Last vertex to cause a change in*** d_v


1
2
3
4
5
6
2
4
1
2
4
3
1
2
2

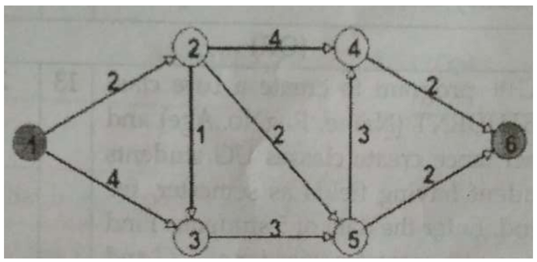
Step-1

Initialize Configuration



V	K	d_v	p_v
1	F	∞	–
2	F	∞	–
3	F	∞	–
4	F	∞	–
5	F	∞	–
6	F	∞	–

Step-2



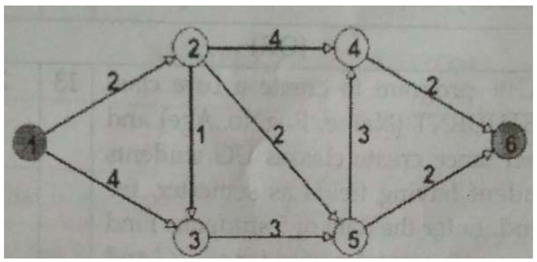
Start with source node 1

After 1 is declared known

V	*K*	d_v	p_v
1	T	0	–
2	F	2	1
3	F	4	1
4	F	∞	–
5	F	∞	–
6	F	∞	–

Step-3

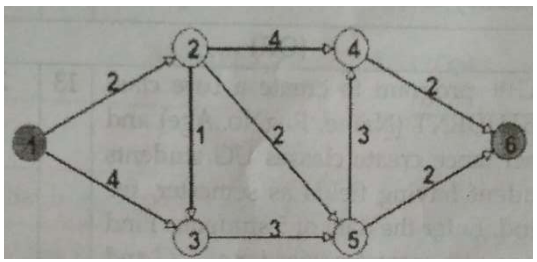
After 2 is declared known



V	K	d_v	p_v
1	T	0	–
2	T	2	1
3	F	3	2
4	F	6	2
5	F	4	2
6	F	∞	–

Step-4

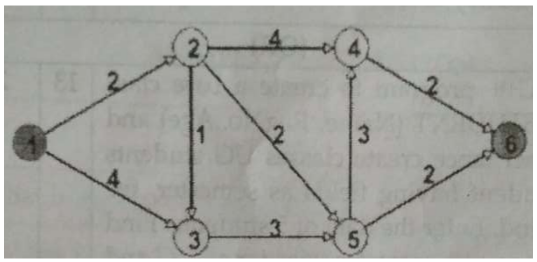
After 3 is declared known



V	K	d_v	p_v
1	T	0	–
2	T	2	1
3	T	3	2
4	F	6	2
5	F	4	2
6	F	∞	–

Step-5

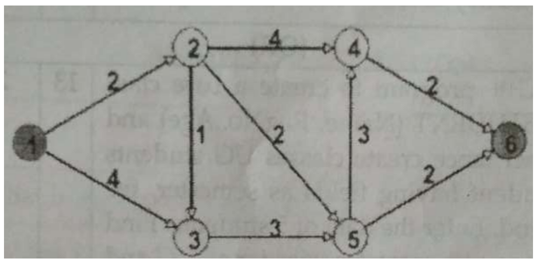
After 5 is declared known



V	*K*	d_v	p_v
1	T	0	–
2	T	2	1
3	T	3	2
4	F	6	2
5	T	4	2
6	F	6	5

Step-6

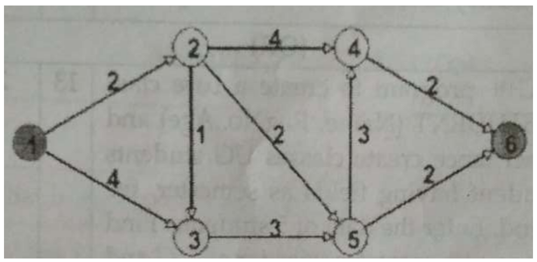
After 4 is declared known



V	K	d_v	p_v
1	T	0	–
2	T	2	1
3	T	3	2
4	T	6	2
5	T	4	2
6	F	6	5

Step-7

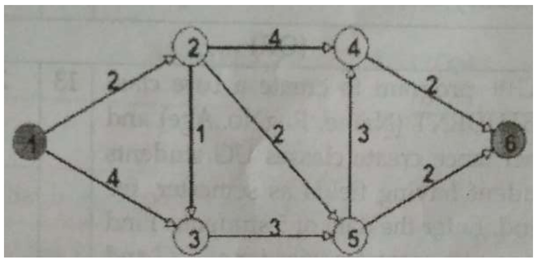
After 6 is declared known



V	*K*	d_v	p_v
1	T	0	–
2	T	2	1
3	T	3	2
4	T	6	2
5	T	4	2
6	T	6	5

Step-8

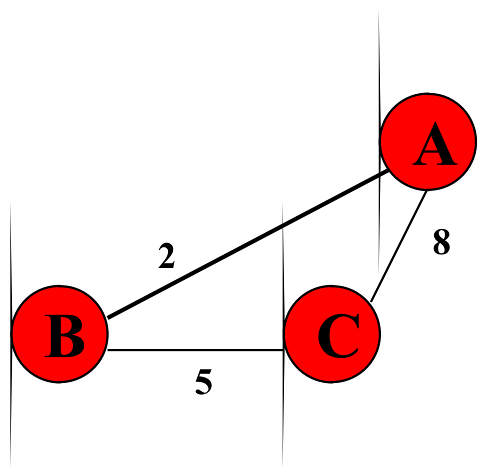
Shortest path from source vertex 1



V	*PATH*	d_v
2	1→2	2
3	1→2→3	3
4	1→2→4	6
5	1→2→5	4
6	1→2→5→6	6

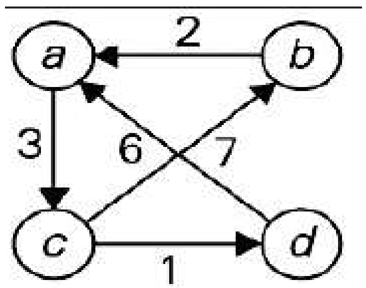
Floyd-Warshall Algorithm

Used to find the **distance between every pair of vertices in a weighted graph G**



	A	B	C
A	0	2	8
B	2	0	5
C	8	5	0

Floyd-Warshall Algorithm



$$D^{(0)} = \begin{array}{c} \\ a \\ b \\ c \\ d \end{array} \begin{array}{c} \begin{array}{cccc} a & b & c & d \end{array} \\ \begin{bmatrix} 0 & \infty & 3 & \infty \\ 2 & 0 & \infty & \infty \\ \infty & 7 & 0 & 1 \\ 6 & \infty & \infty & 0 \end{bmatrix} \end{array}$$

$$D^{(1)} = \begin{array}{c} \\ a \\ b \\ c \\ d \end{array} \begin{array}{c} \begin{array}{cccc} a & b & c & d \end{array} \\ \begin{bmatrix} 0 & \infty & 3 & \infty \\ 2 & 0 & \mathbf{5} & \infty \\ \infty & 7 & 0 & 1 \\ 6 & \infty & \mathbf{9} & 0 \end{bmatrix} \end{array}$$

$$D^{(2)} = \begin{array}{c} \\ a \\ b \\ c \\ d \end{array} \begin{array}{c} \begin{array}{cccc} a & b & c & d \end{array} \\ \begin{bmatrix} 0 & \infty & 3 & \infty \\ 2 & 0 & 5 & \infty \\ \mathbf{9} & 7 & 0 & 1 \\ 6 & \infty & 9 & 0 \end{bmatrix} \end{array}$$

$$D^{(3)} = \begin{array}{c} \\ a \\ b \\ c \\ d \end{array} \begin{array}{c} \begin{array}{cccc} a & b & c & d \end{array} \\ \begin{bmatrix} 0 & \mathbf{10} & 3 & \mathbf{4} \\ 2 & 0 & 5 & \mathbf{6} \\ 9 & 7 & 0 & 1 \\ 6 & \mathbf{16} & 9 & 0 \end{bmatrix} \end{array}$$

$$D^{(4)} = \begin{array}{c} \\ a \\ b \\ c \\ d \end{array} \begin{array}{c} \begin{array}{cccc} a & b & c & d \end{array} \\ \begin{bmatrix} 0 & 10 & 3 & 4 \\ 2 & 0 & 5 & 6 \\ \mathbf{7} & 7 & 0 & 1 \\ 6 & 16 & 9 & 0 \end{bmatrix} \end{array}$$

Floyd-Warshall Algorithm

```
ALGORITHM  Floyd(W[1..n, 1..n])
    //Implements Floyd's algorithm for the all-pairs shortest-paths problem
    //Input: The weight matrix W of a graph with no negative-length cycle
    //Output: The distance matrix of the shortest paths' lengths
    D ← W //is not necessary if W can be overwritten
    for k ← 1 to n do
        for i ← 1 to n do
            for j ← 1 to n do
                D[i, j] ← min{D[i, j], D[i, k] + D[k, j]}
    return D
```