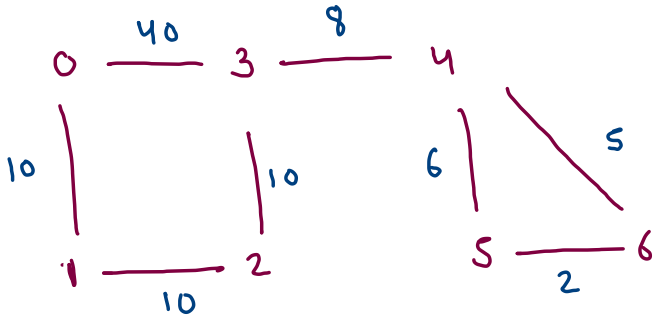
Construct & display



Adjacency matrix

	0	1	2	3	4	5	6
0	∞	10	∞	40	∞	∞	∞
1	10	∞	10	∞	∞	∞	∞
2	∞	10	∞	10	∞	∞	∞
3	40	∞	10	∞	8	∞	∞
4	∞	∞	∞	8	∞	6	5
5	∞	∞	∞	∞	6	∞	2
6	∞	∞	∞	∞	5	2	∞

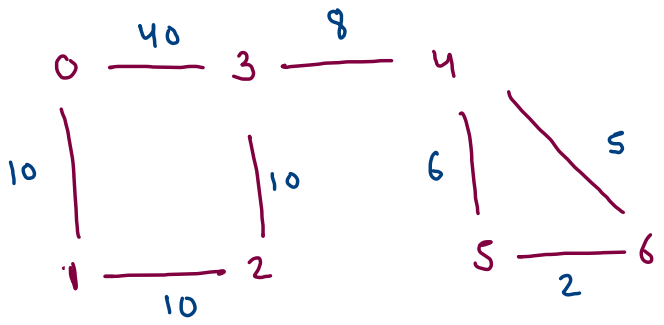
7 vertices

8 edges

	0	1	2	3	4	5	6
0	∞	10	∞	40	∞	∞	∞
1	10	∞	10	∞	∞	∞	∞
2	∞	10	∞	10	∞	∞	∞
3	40	∞	10	∞	8	∞	∞
4	∞	∞	∞	8	∞	6	5
5	∞	∞	∞	∞	6	∞	2
6	∞	∞	∞	∞	5	2	∞

→ space is not efficient used.

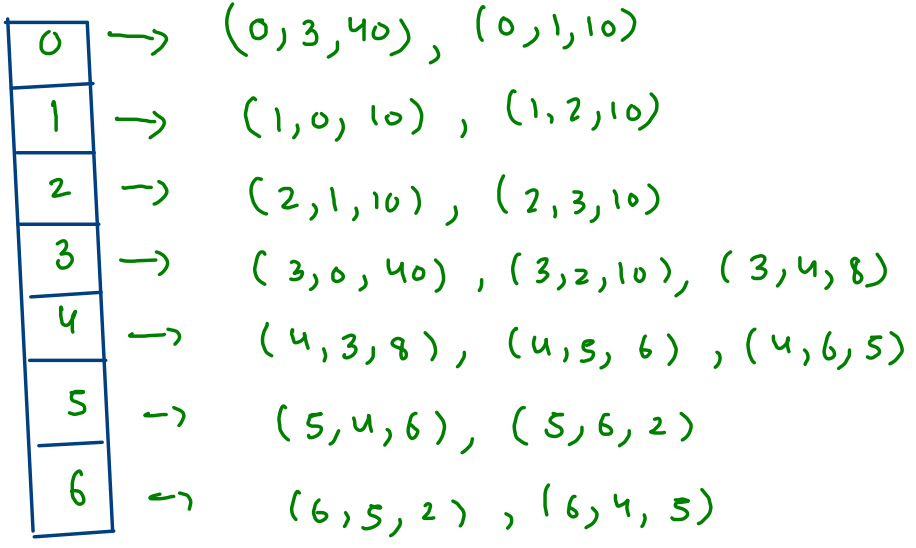
we will not use adjacency matrix representation.



Edge : (u, v, wt)

ArrayList < Edge > [ ] graph;

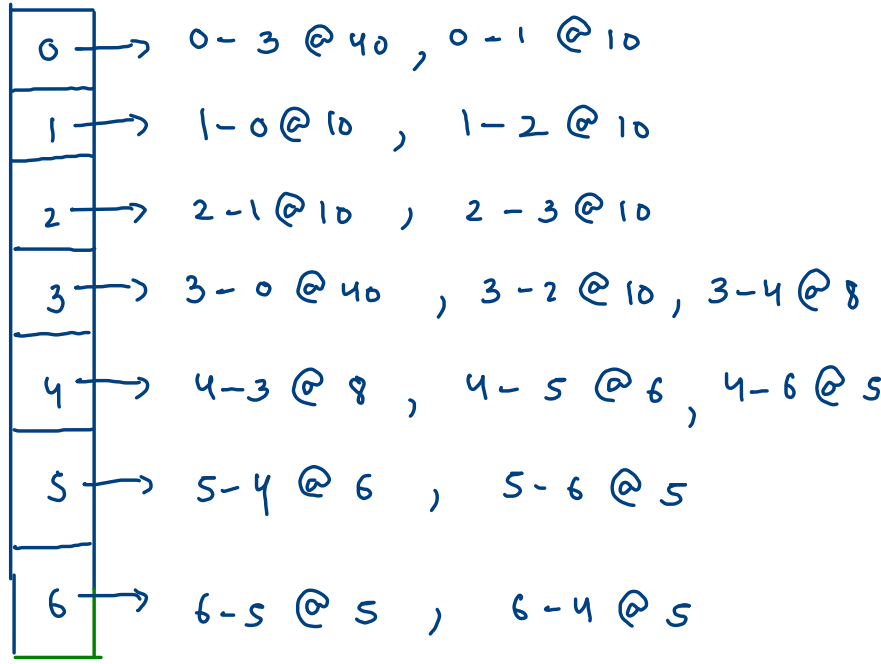
Adjacency list :



```
0 -> 0-3@40, 0-1@10,

1 -> 1-0@10, 1-2@10,

2 -> 2-1@10, 2-3@10,

3 -> 3-0@40, 3-2@10, 3-4@8,

4 -> 4-3@8, 4-5@6, 4-6@5,

5 -> 5-4@6, 5-6@2,

6 -> 6-5@2, 6-4@5,
```

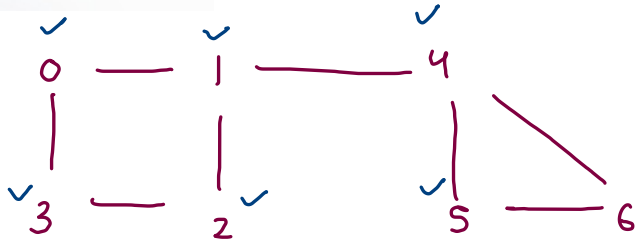
```
addEdge(graph,0,3,40);
addEdge(graph,0,1,10);
addEdge(graph,1,2,10);
addEdge(graph,2,3,10);
addEdge(graph,3,4,8);
addEdge(graph,4,5,6);
addEdge(graph,5,6,2);
addEdge(graph,4,6,5);

display(graph);
```

```
ArrayList<Edge>[]graph = new ArrayList[vtx];

for(int i=0; i < graph.length;i++) {
    graph[i] = new ArrayList<>();
}
```

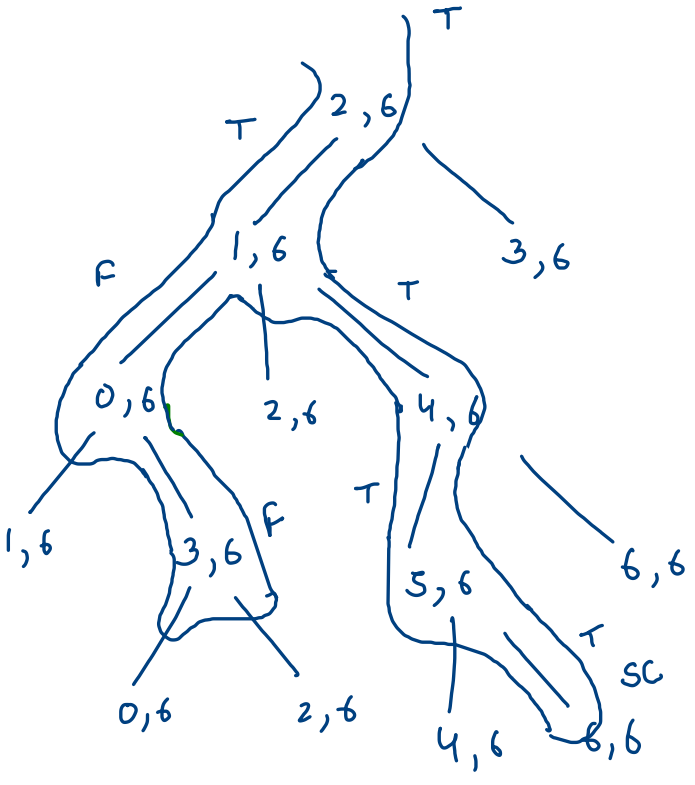



```
public static void addEdge(ArrayList<Edge>[]graph,int u,int v,int wt) {
    Edge e1 = new Edge(u,v,wt);
    Edge e2 = new Edge(v,u,wt);

    graph[u].add(e1);
    graph[v].add(e2);
}


public static void display(ArrayList<Edge>[]graph) {
    for(int i=0; i < graph.length;i++) {
        System.out.print(i + " -> ");
        for(int j=0; j < graph[i].size();j++) {
            Edge edge = graph[i].get(j);

            System.out.print(edge.u + "-" + edge.v + "@" + edge.wt+ ", ");
        }
        System.out.println();
    }
}
```

Has Path?



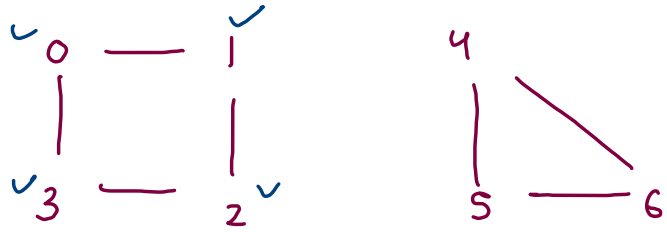

```
S  if(src == dest) {
       return true;
   }

M  vis[src] = true;

   for(Edge edge : graph[src]) {
       int nbr = edge.nbr;

nbr    if(vis[nbr] == false) {
           boolean sa = hasPath(graph,nbr,dest,vis);

           if(sa == true) {
               return true;
           }
       }
   }

   return false;
```

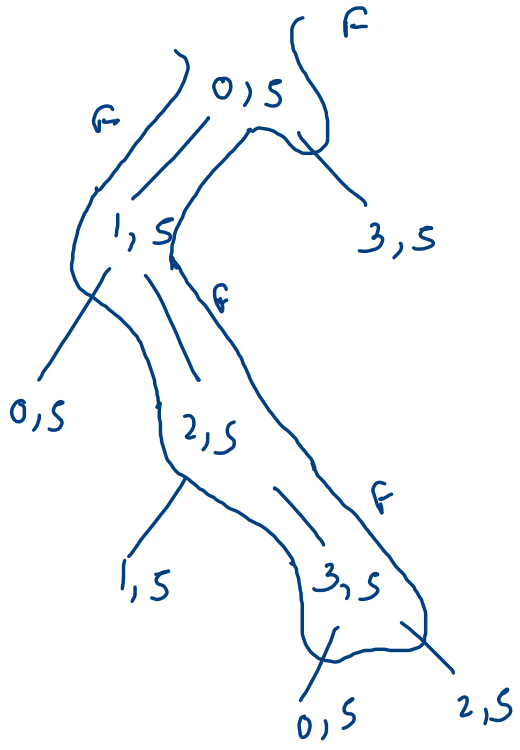


src = 2

dest = 6

```
if(src == dest) {
    return true;
}

vis[src] = true;

for(Edge edge : graph[src]) {
    int nbr = edge.nbr;

    if(vis[nbr] == false) {
        boolean sa = hasPath(graph,nbr,dest,vis);

        if(sa == true) {
            return true;
        }
    }
}

return false;
```





Src = 0

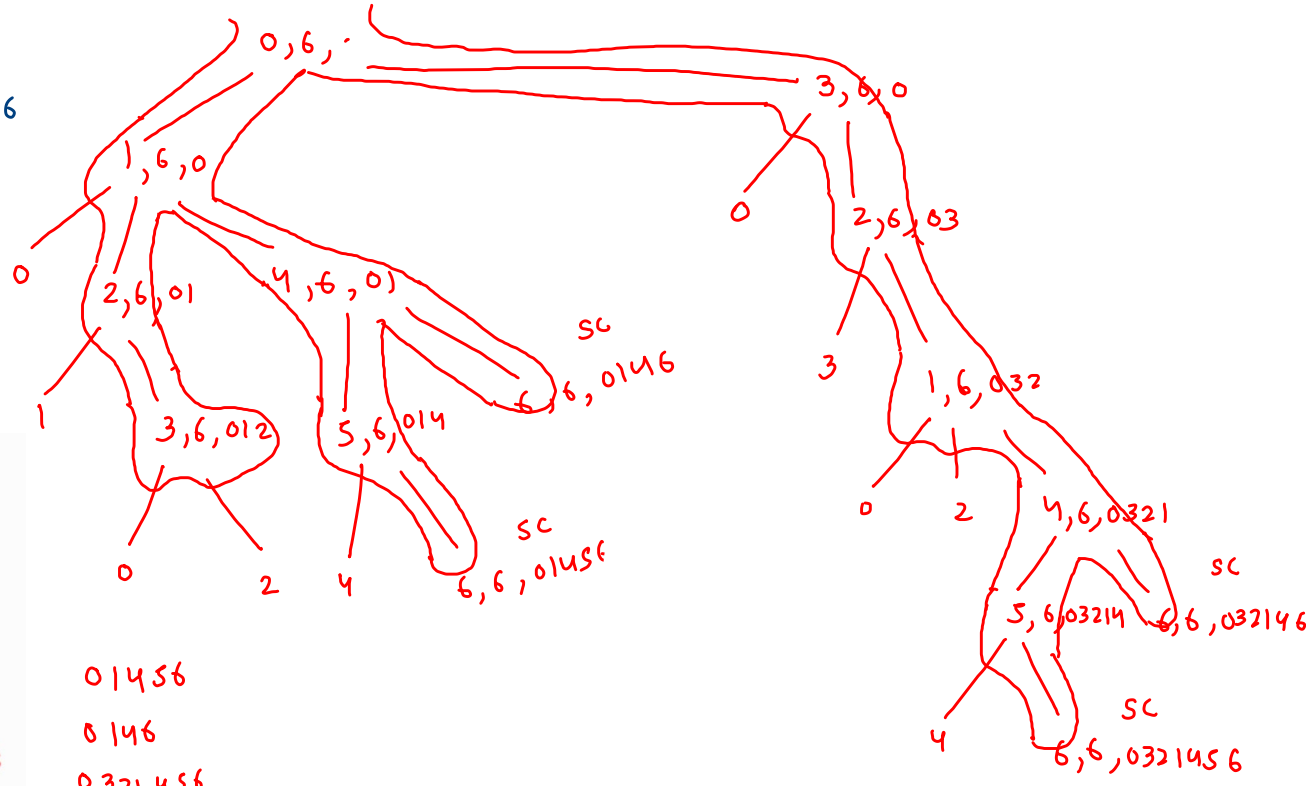
dest = 5

Print All Paths

src = 0

dest = 6


```
if(src == dest) {
    psf += dest;
    System.out.println(psf);
    return;
}

vis[src] = true;

for(Edge edge : graph[src]) {
    int nbr = edge.nbr;

    if(vis[nbr] == false) {
        printAllPaths(graph,nbr,dest,vis,psf + src);
    }
}

vis[src] = false;
```



01456

0146

0321456

032146