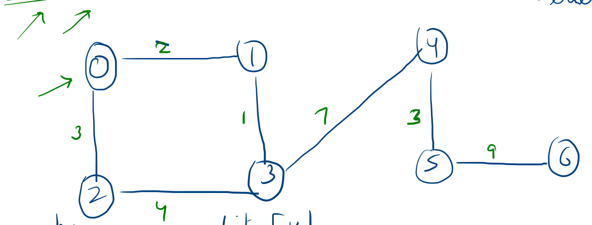
graph

src, des

Vertex => V

8 Vertex

11 nodes



Arraylist<Edge>[] graph = new Arraylist [V]

$O(E)$

Adjacency list =>

0 → {(0,1,2), (0,2,3)

1 → {(1,0,2), (1,3,1)

2 → {(2,0,3), (2,3,4)

3 → {(3,1,1). (3,2,4), (3,4,7)

4 → {(4,3,7) (4,5,3)

5 → {(5,4,3) (5,6,9)

6 → {(6,5, 9)}

0 (0,1,2)
(0,2,3)

0 →(1,2)
(2,3)

Edge

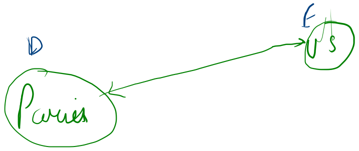
V

	0	1	2	3	4	5	6
0		2	✓				
1	2			✓			
2	✓						
3		✓			✓		
4				✓		✓	
5					✓		✓
6						✓	

=> adjacency matrix
↓
not efficient

$O(V^2)$

$O(V)$

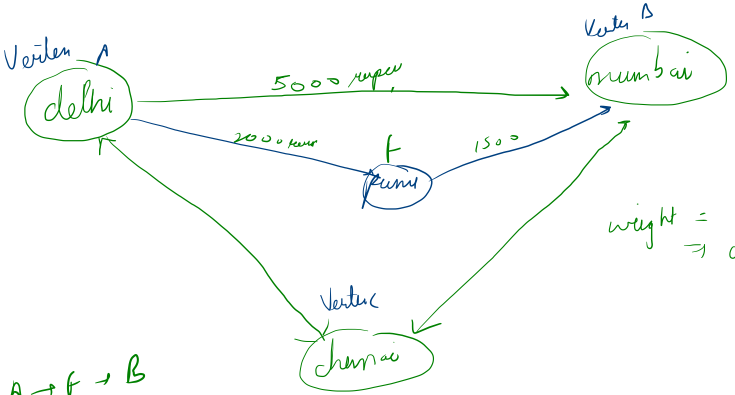


Vertex

edges

weight

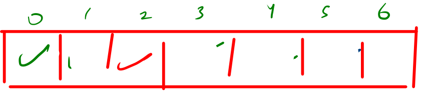
car

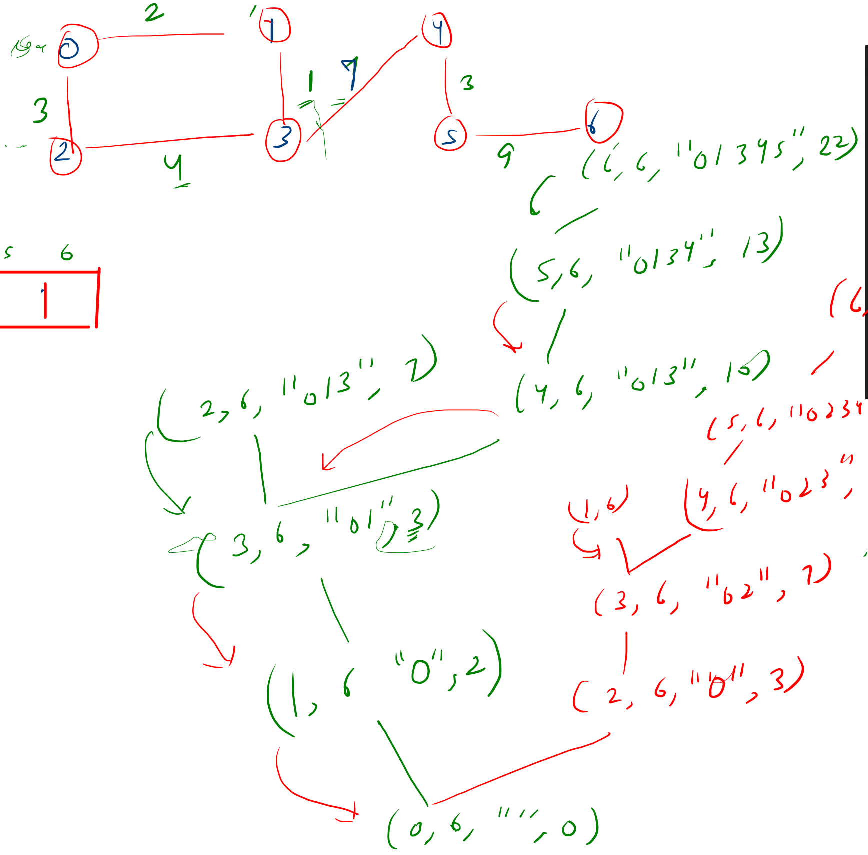


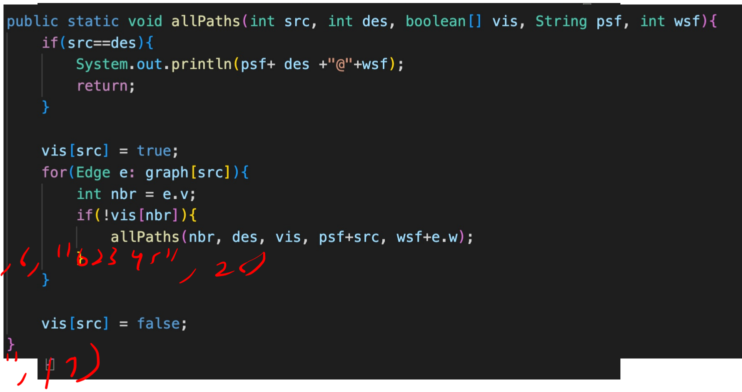
weight = cost
=> distance

A → F → B

0, 6







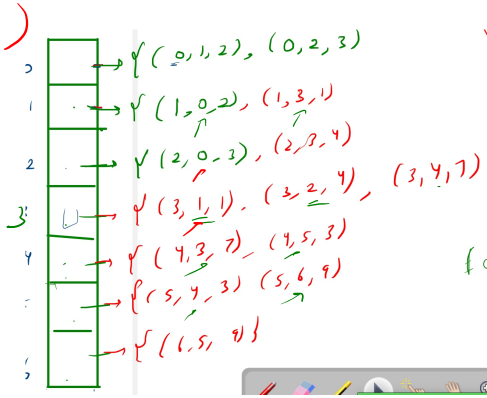
```
public static void allPaths(int src, int des, boolean[] vis, String psf, int wsf){
    if(src==des){
        System.out.println(psf+ des +"@"+wsf);
        return;
    }

    vis[src] = true;
    for(Edge e: graph[src]){
        int nbr = e.v;
        if(!vis[nbr]){
            allPaths(nbr, des, vis, psf+src, wsf+e.w);
        }
    }

    vis[src] = false;
}
```

0 1 3 4 5 6 @ 22

0 2 3 4 5 6 @ 26



0 → {(0, 1, 2), (0, 2, 3)

1 → {(1, 0, 2), (1, 3, 1)

2 → {(2, 0, 3), (2, 3, 4)

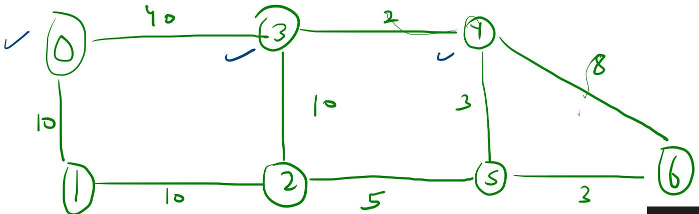
3 → {(3, 1, 1), (3, 2, 4), (3, 4, 7)

4 → {(4, 3, 7), (4, 5, 3)

5 → {(5, 4, 3), (5, 6, 9)

6 → {(6, 5, 9)}

{ edge }



src = 0

dest = 6

criteria = 40

k = 4

0 3 4 6 @ 50

0 3 4 5 6 @ 48

k th

0 3 2 5 4 6 @ 66

0 3 2 5 6 @ 58

0 3 4 6 @ 50

0 1 2 5 6 @ 28

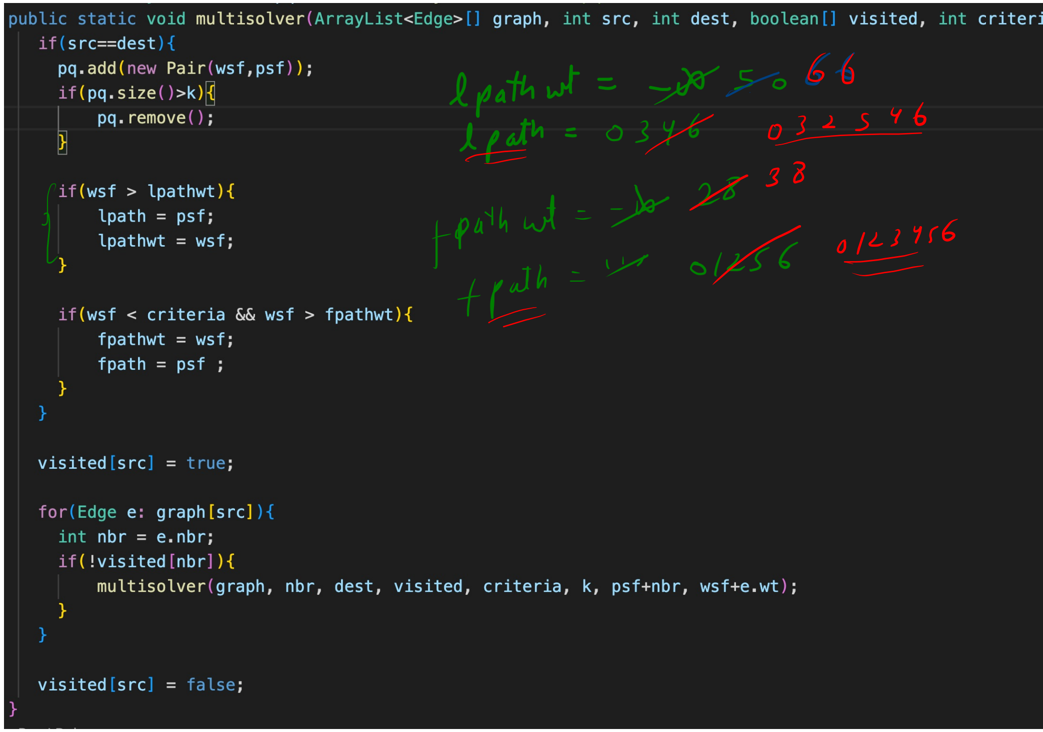
0 1 2 3 4 5 6 @ 38

0 3 2 5 4 6 @ 66

0 1 2 3 4 6 @ 40

0 3 2 5 6 @ 58

0 3 4 5 6 @ 48



```
public static void multisolver(ArrayList<Edge>[] graph, int src, int dest, boolean[] visited, int criteri
   if(src==dest){
     pq.add(new Pair(wsf,psf));
     if(pq.size()>k){
         pq.remove();
     }

     if(wsf > lpathwt){
         lpath = psf;
         lpathwt = wsf;
     }

     if(wsf < criteria && wsf > fpathwt){
         fpathwt = wsf;
         fpath = psf ;
     }
   }

   visited[src] = true;

   for(Edge e: graph[src]){
     int nbr = e.nbr;
     if(!visited[nbr]){
         multisolver(graph, nbr, dest, visited, criteria, k, psf+nbr, wsf+e.wt);
     }
   }

   visited[src] = false;
}
```

lpath wt = ~~-∞~~ ~~50~~ 66

lpath = 0 3 ~~4 6~~ 0 3 2 5 4 6

fpath wt = ~~-∞~~ ~~28~~ 38

fpath = ~~-∞~~ 0 ~~1 2 5 6~~ 0 1 2 3 4 5 6