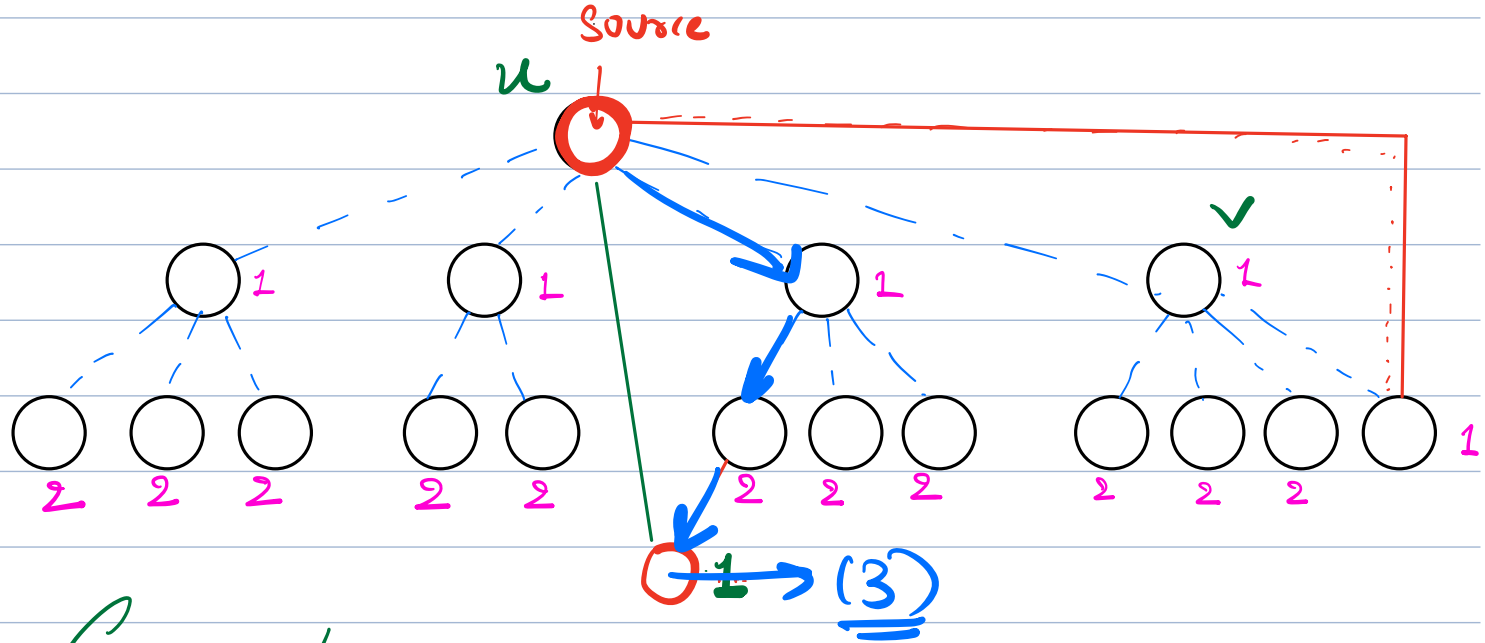
Agenda :

1) BFS
2) Dijikstra's Algorithm.
3) MST

Linked In



Given two users u & v.
find their degree of connection.

Breadth first Traversal (BFS)

Code

```
vst [N+1] = {false};

void bfs (source , ) {
    Queue <int> q;

    q. enqueue (source);
    vst[source] = True;

    while (! q. isEmpty()) {

        u = q. dequeue();

        for (all V connected to u) {

            if (vst[v] == false) {
                vst[v] = True;
                q. enqueue(v);
            }
        }
    }
}
```

$O(\underline{V} + \underline{E})$

```
int findDegree(u, v, N){
    vst[N+1] = {false};
    Queue<Pair<int, int>> q;
```

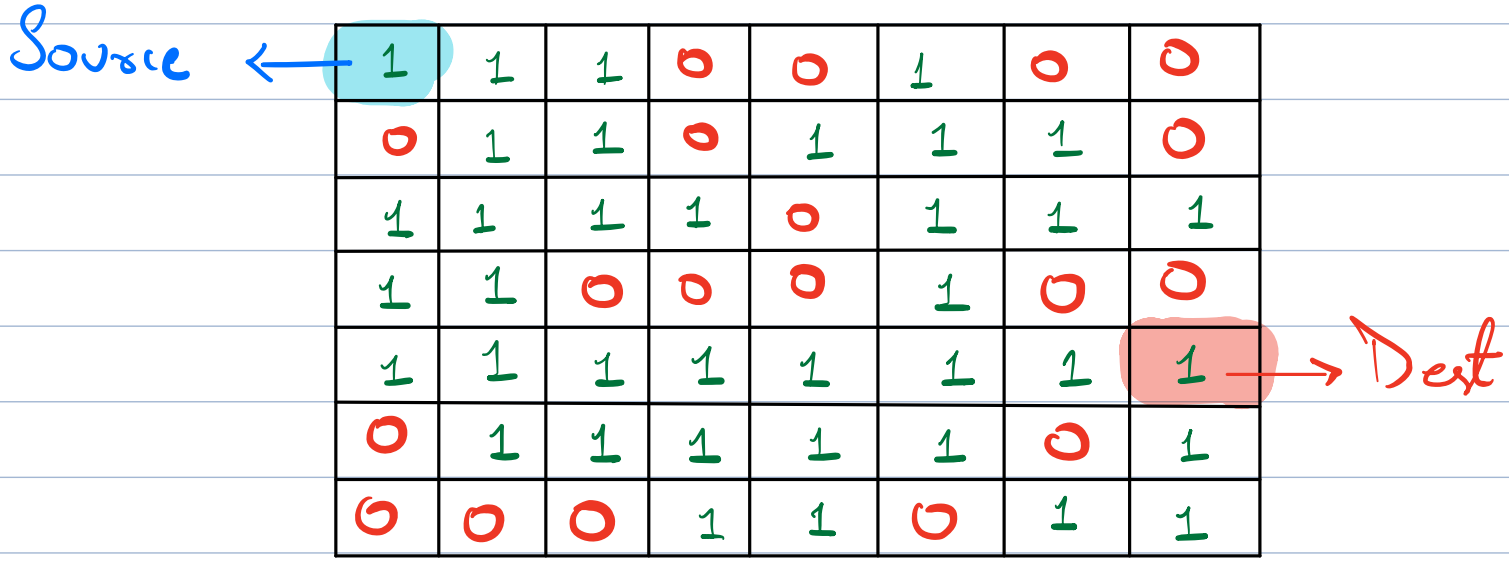
first (Node) — second <Degree>

```
    vst[u] == True;
    q.enqueue(u, 0);
    while(!q.isEmpty()){
        w = q.dequeue();
        node = w.first();
        degree = w.second();
        for(all nodes z connected to node){
            if(vst[z] == false){
                if(z == v){
                    return degree+1;
                }
```

```
            q.enqueue(z, degree+1);
            vst[z] = true;
        }
    }
}
return -1;
}
```

T.C. = O(V+E)

Grid Problems

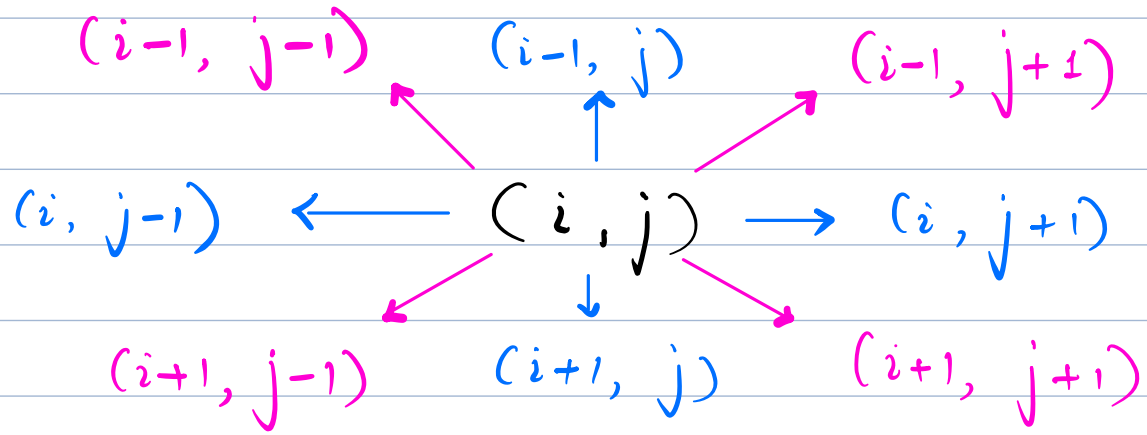


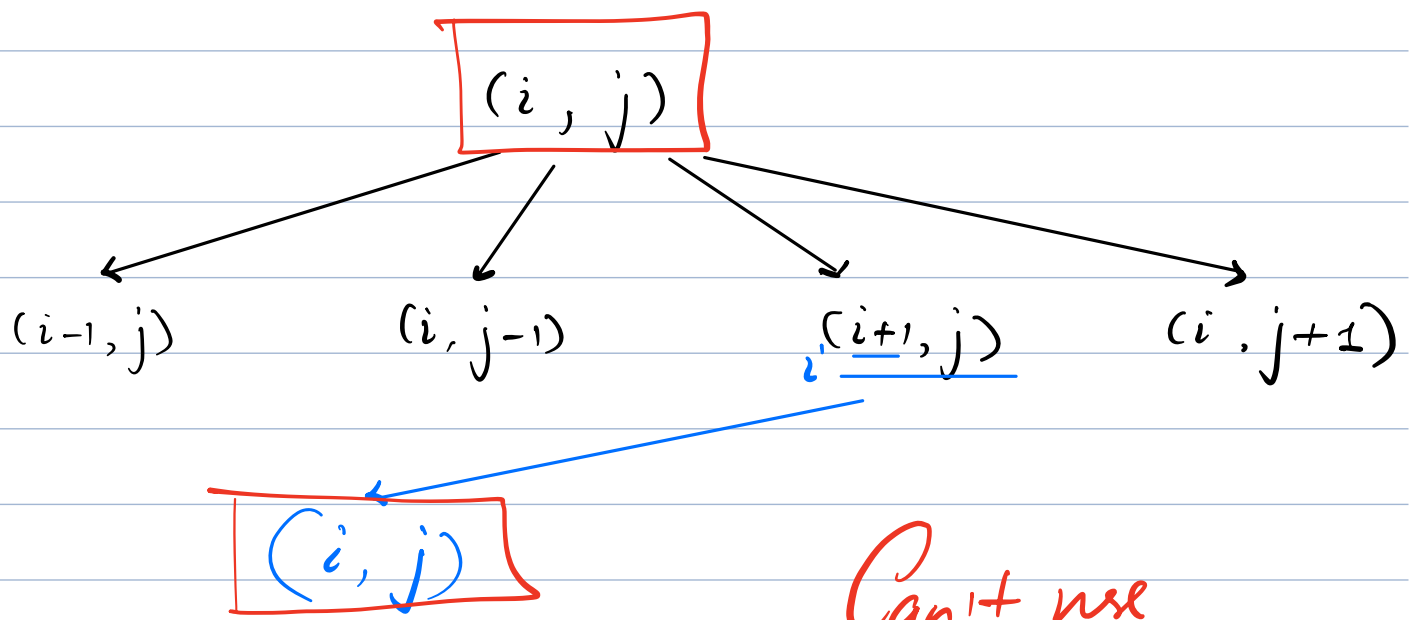
1	1	1	0	0	1	0	0
0	1	1	0	1	1	1	0
1	1	1	1	0	1	1	1
1	1	0	0	0	1	0	0
1	1	1	1	1	1	1	1
0	1	1	1	1	1	0	1
0	0	0	1	1	0	1	1

0 → Blocked

1 → Open

Given source & dest cell.
find the min distance b/w them



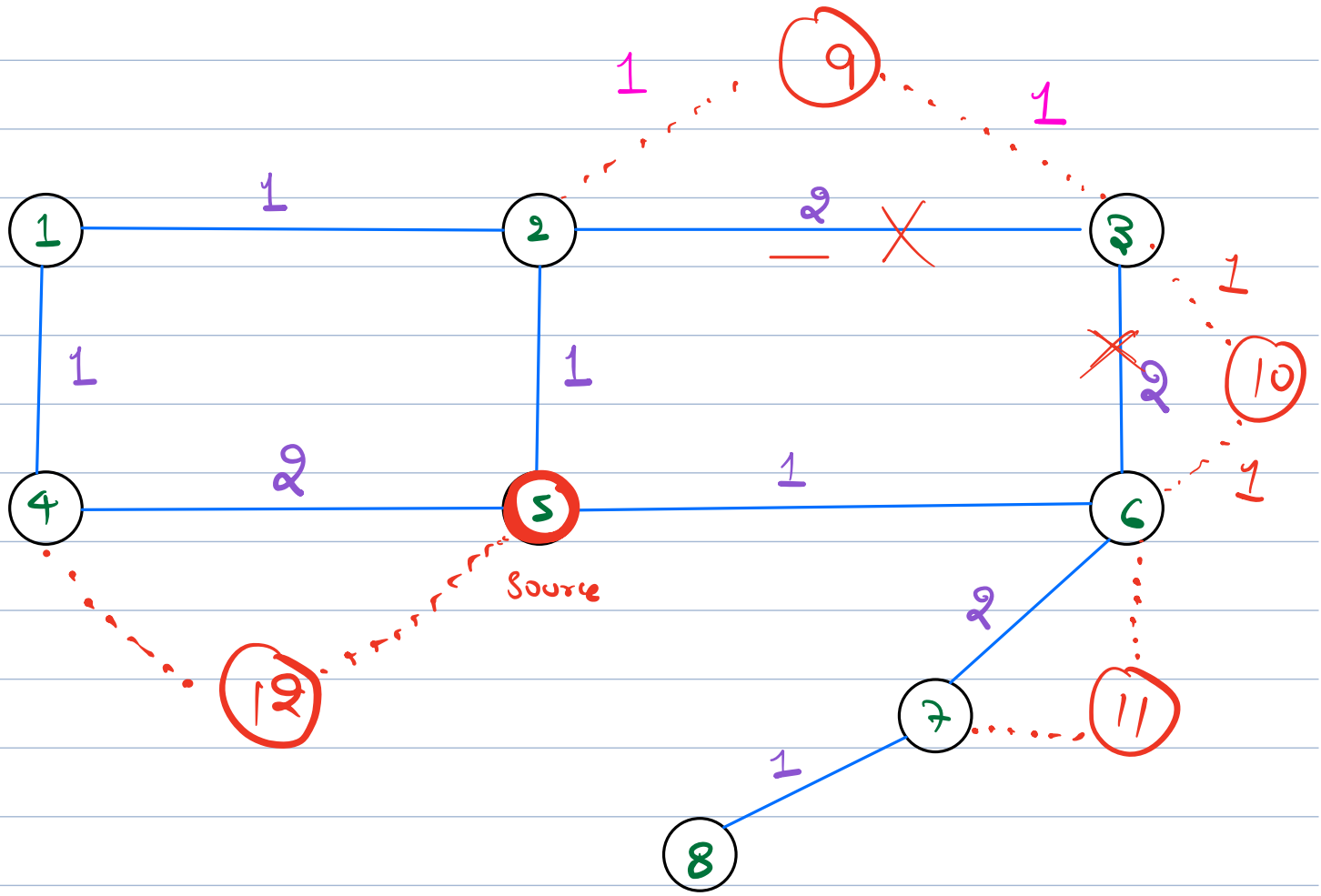


Can't use DP

⇒ BFS algorithm.

Q) Given an undirected weighted graph where the weights of the edges is $\in \{1, 2\}$

Find the shortest distance to every node from a given source node.



1 : <2, 1>, <4, 1>

2 : <1, 1>, ~~<3, 2>~~ <9, 1>

3 : ~~<2, 2>~~, ~~<6, 2>~~ <9, 1>, <10, 1>

4 : <1, 1>, <5, 2>

5 : <2, 1>, <4, 2>, <6, 1>

6 : ~~<3, 2>~~, <7, 2>, <5, 1>

7 : <6, 2>, <8, 1>

8 : <7, 1>

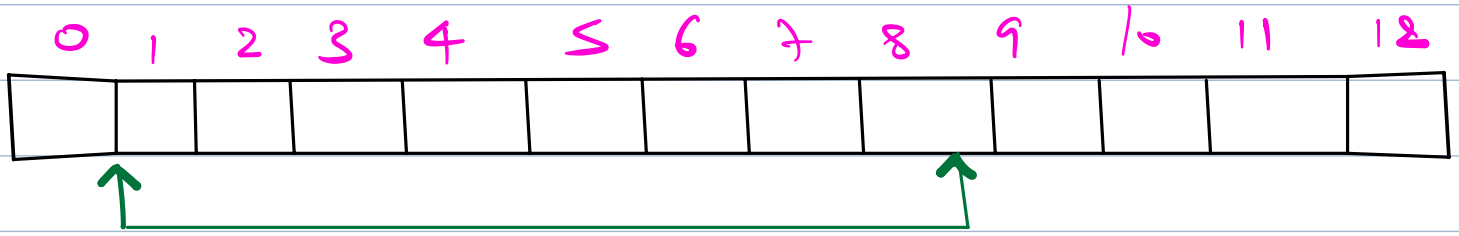
9 : <2, 1>, <3, 1>

10 : <3, 1>, <6, 1>

11 :

12 :

Use Bfs to find shortest distance



Initial Nodes = V

Max. Modified Node Count = V + E (V')

Initial Edges = E

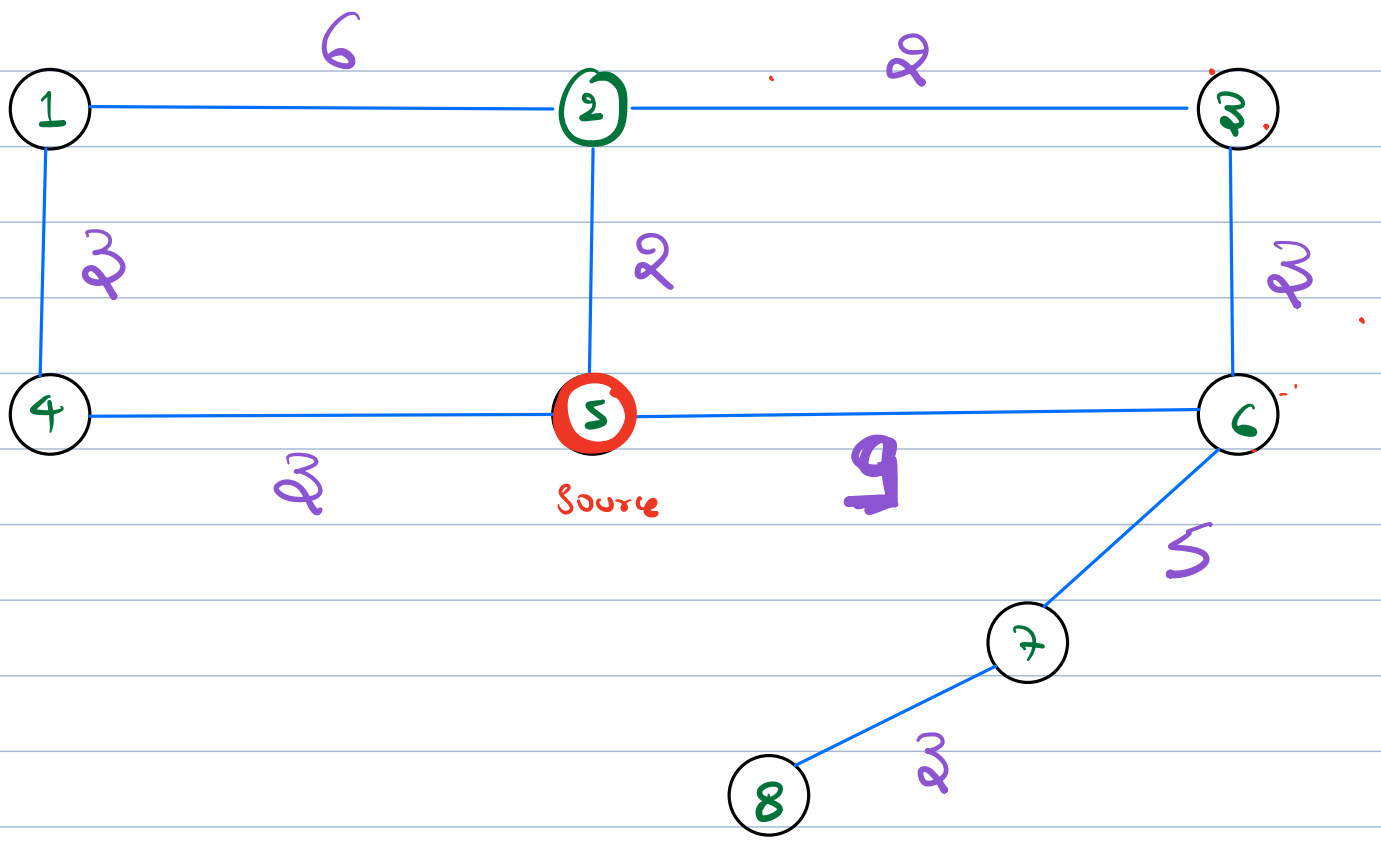
Max. Modified Edge Count = 2E (E')

$$T.C. = O(V' + E')$$

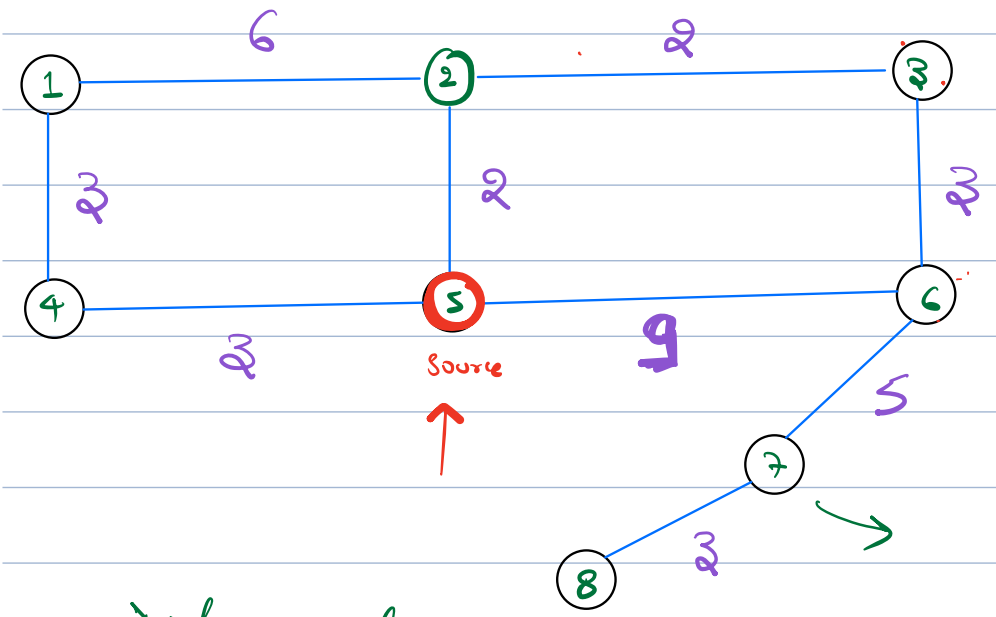
$$= O(V + E + 2E)$$

$$= O(V + 3E)$$

$$\underline{\underline{T.C}} = O(V + E)$$



Min Heap << dist, node >>



1 : ~~∞~~ 8 6
2 : ~~∞~~ 2
3 : ~~∞~~ 4
4 : ~~∞~~ 3
5 : 0
6 : ~~∞~~ 9 7
7 : ~~∞~~ 12
8 : ~~∞~~ 15

Dist Node

<2, 2>
<9, 6>
<3, 4>
<4, 3>
<8, 1>
<6, 1>
<7, 6>
<12, 7>
<15, 8>

ε

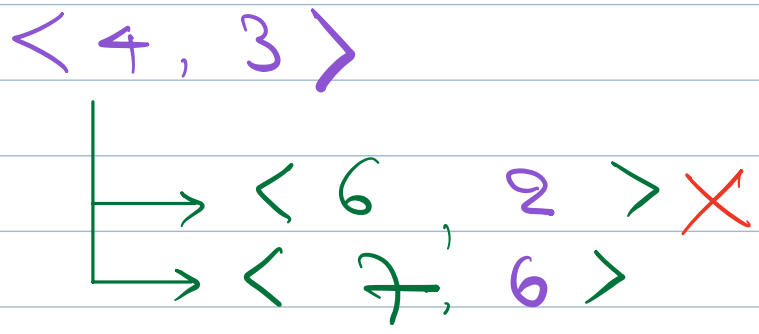
<2, 2>

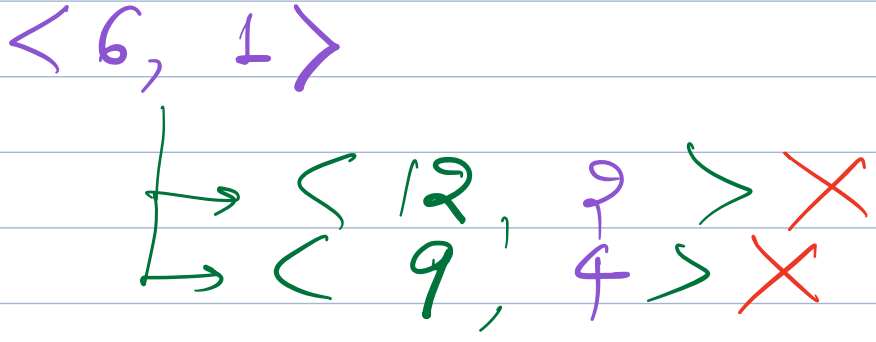
<4, 3>
<8, 1>
<4, 5> X

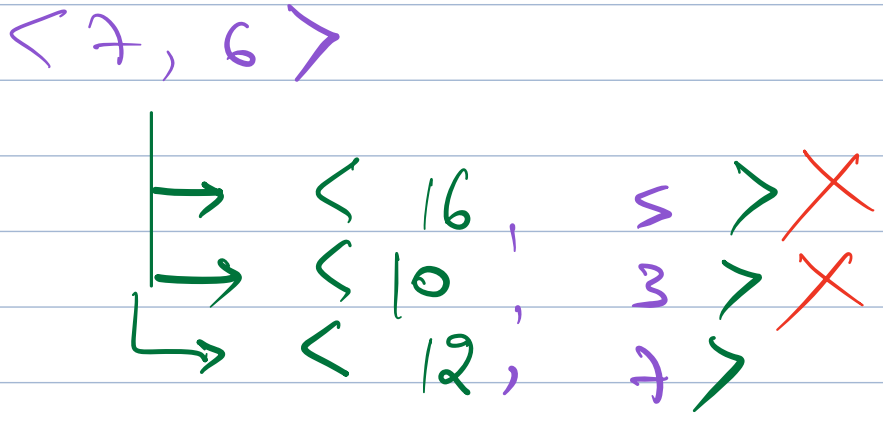
<3, 4>

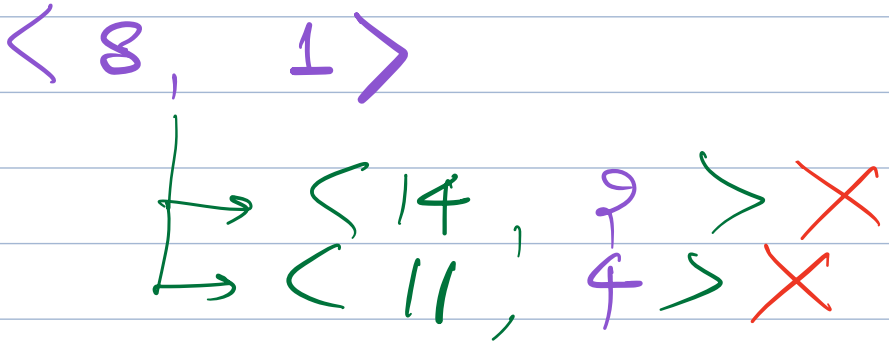
<6, 1>
<6, 5> X

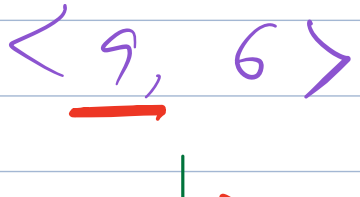
$d_{u-w} = d_{u-v} + \varepsilon_{v-w}$

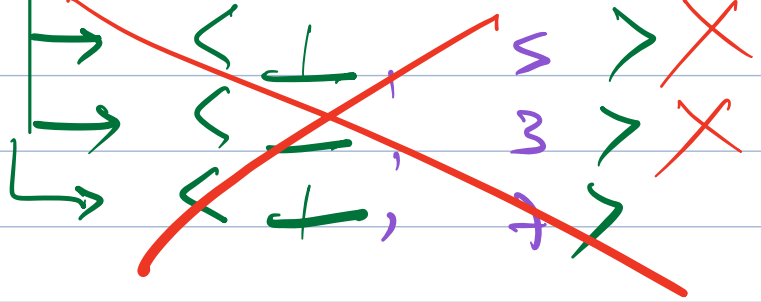

<4, 3>
<6, 2> X
<7, 6>


<6, 1>
<12, 9> X
<9, 4> X


<7, 6>
<16, 5> X
<10, 3> X
<12, 7>


<8, 1>
<14, 9> X
<11, 4> X


<9, 6>



< 19, 7 >

↳ < 17, 6 > X
↳ < 15, 8 >

< 15, 8 >

↳ < 18, 7 > X

Dijikstra's Algorithm

T.C. = $O(V + E \log(E))$

Code

```
int[] dijikstra (source) {
    dist[N+1] = {∞};
    dist[source] = 0
    MinHeap<Pair<int, int>> mh;
                  ↓      ↓
                first  second
              Distance  Node
    mh.insert(<0, source>);
    while (!mh.isEmpty()) {
        x = mh.getMin();
        distance_x = x.first();
        Node_x     = x.second();
        if (distance_x ≤ dist[Node_x]) {
```

```
for (all u connected to Node_x) {
    du = distance_x + W_x-u
    if (du < dist[u]) {
        dist[u] = du
        mh.insert(<du, u>);
    }
}
return dist;
}
```

Interview Probs

- 1) Prim's Algo (MST)
- 2) Multi Source BFS
- 3) Running Median (Heap)

Problems

Extra Session on Interview Problems