Graph 3 : MST (Prims Algorithm) & Dijkstra Algorithm

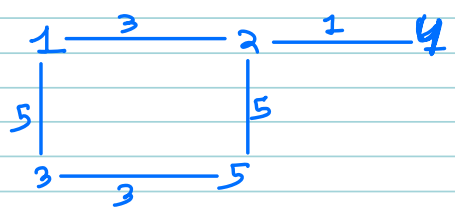
AGENDA

- PRIM'S ALGO
- DIJKSTRA ALGO

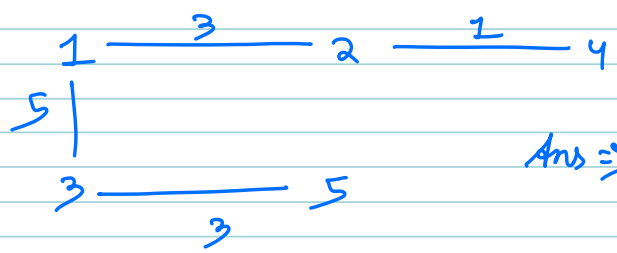
Question :- Given N Islands & Cost of Construction of bridge b/w Multiple pair of Islands. Find the min cost of Construction required such that it is possible to travel from one Island to another Island via bridges. If not possible return -1.

Ex :- N = 5

1 —3— 2
1 —5— 3
2 —1— 4
2 —5— 5
3 —3— 5

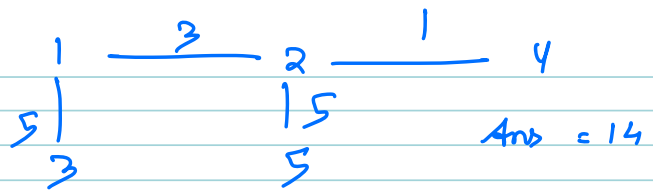


TRY! :-



Ans => 12

TRY 2 :-



Ans = 14

Q How Many Edges I need given N Nodes

↳ N - 1 Ex:- N = 5
edges = N-1 = 4

⇒ SPANNING TREE

↳ A Spanning Tree is a Subgraph of a graph that includes all the nodes, maintains connectivity & has no cycle.

Note:- It does not give Shortest Distance (weight) from one point to other

⇒ MINIMUM SPANNING TREE

Tree generated from a connected graph such that all nodes are connected & Sum of weight of all selected edges is Minimum

SAMPLE PROBLEM STATEMENT

Scenario:
Suppose Flipkart has N local distribution centers spread across a large metropolitan city. These centers need to be interconnected for efficient movement of goods. However, building and maintaining roads between these centers is costly. Flipkart's goal is to minimize these costs while ensuring every center is connected and operational.

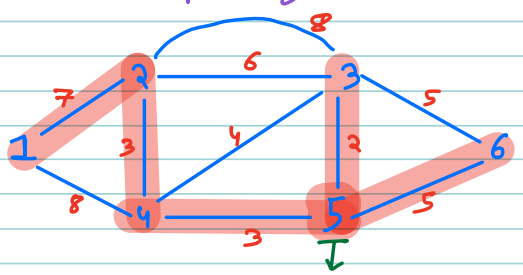
Goal: You are given number of centers and possible connections that can be made with their cost. Find minimum cost of constructing roads between centers such that it is possible to travel from one center to any other via roads.

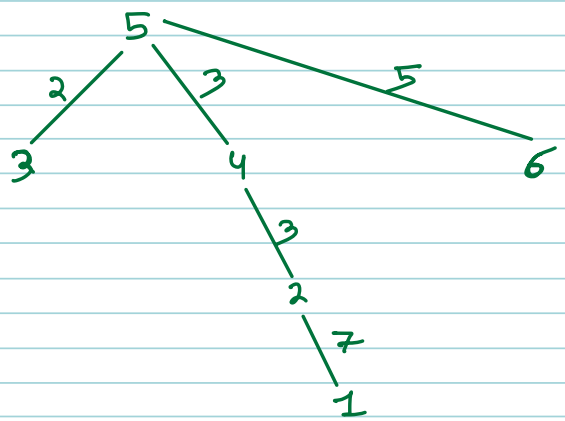
⇒ ALGORITHM WHICH HELS WITH MST

① Prims Algo
② Kruskal Algo ⟶ (DSA 4.2)

PRIMS ALGORITHM

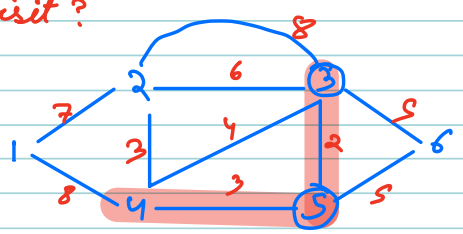
Goal :- Find Minimum Spanning tree





Ans = 20

Quiz 1 :- which is the next node, we need to visit ?

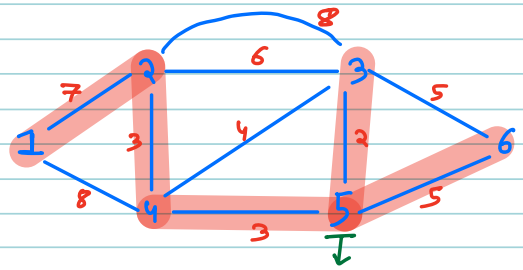


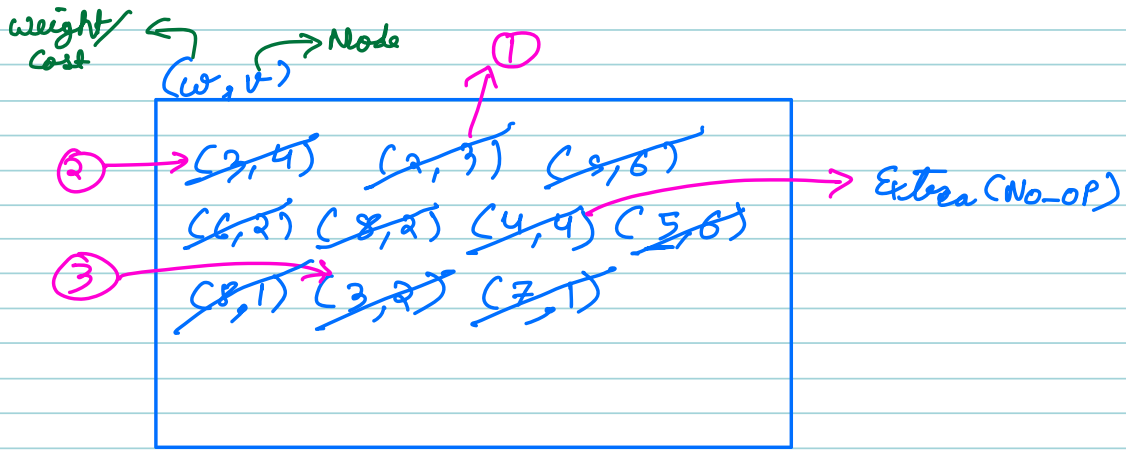
↳ 2 [4 – 2]

Quiz 2 :- what is the most suitable data structure for implementing prim's algorithm ?

↳ Heap.

⇒ One More Iteration





Q why (4,4) got added when their is another Way to reach 4 with lesser weight

↳ 4 was not Visited when we have inserted (4,4)

PSEUDO CODE

Step 1 :- Select any node & add all its neighbour in min heap.

Step 2 :-

```
while ( mh.size() > 1 ) {
    pair p = mh.getMin();
    mh.removeMin();
    if ( vis[u] == true ) {
        continue;
    }
    ans += w;
    vis[u] = true;
    for ( (w, v) in adj[u] ) {
        if ( vis[v] != true ) {
            mh.add( {w, v} );
        }
    }
}
return ans;
```

(w, u)

TC

max size of heap = E

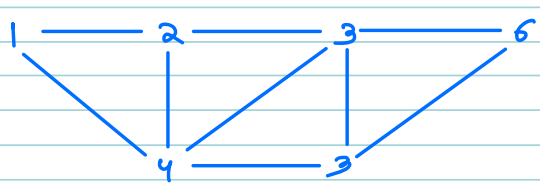
$$TC \rightarrow E \log E$$

$$SC \rightarrow O(E + N + \text{adj list}) \rightarrow (N+E)$$

$$\approx O(N+E)$$

question:- Find min Edges to travel from u to v.

Ex1:-



u = 1
v = 6
} Shortest path

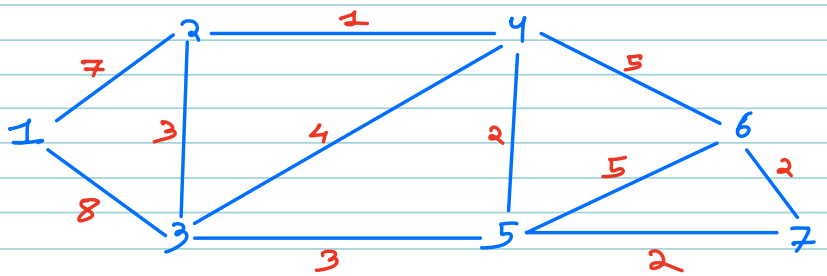
Ans = 3
[Simple BFS]

⇒ The BFS will fail when edges have weight & we want to find the path with min weight.

↳ This can be solved using DIJKSTRA

DIJKSTRA ALGO

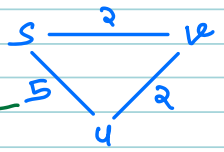
question :- There are n cities in a country, you are living in city 1. Find min distance to reach every other city from city 1.



⇒ Single Source shortest path → Dijkstra

d[] → d[i] min dis to reach from source to ith node.

Ex :-



d[s] = 0

source to u d[u] = 5

d[v] = 2

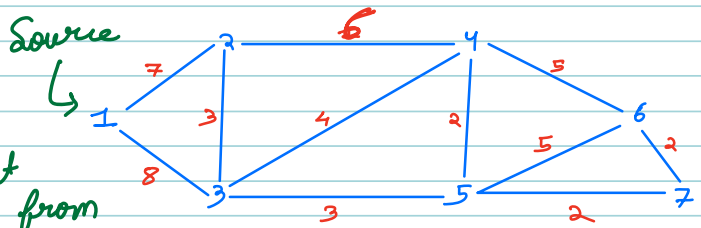
↳ Source to v

if $d[s-u] > d[s-v] + d[v-u]$

then $d[s-u] = d[s-v] + d[v-u]$

Relaxing an edge.

Ex



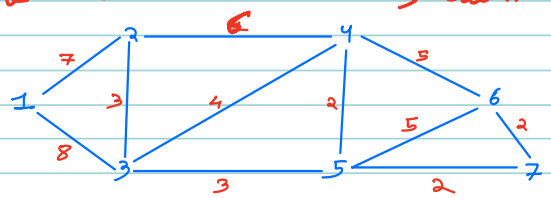
Shortest distance from source to s array

↳ d = [0 ~~7~~ ~~8~~ ~~12~~ ~~11~~ ~~15~~ ~~13~~]
1 2 3 4 5 6 7

(d, v)

~~(7,2)~~ ~~(8,3)~~		~~(15,6)~~
~~(10,3)~~ ~~(13,4)~~		
~~(12,4)~~ ~~(11,5)~~		
~~(14,5)~~ ~~(17,6)~~		
~~(10,6)~~ ~~(13,7)~~		

Quiz 3 :- The current State are, Given Graph



Min Heap = [(10,3) (13,4) (11,5) (12,4)]

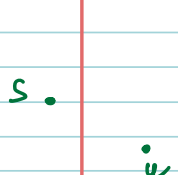
d[] = [0 7 8 ∞ ∞ ∞ ∞]
1 2 3 4 5 6 7

What would be the next step, which is going to take place?

↳ pick (10,3) & Dont update the distance array.

PSEUDO CODE

```
dist -> // TODO [ all ∞ ]
dis[ s ] = 0;
Add all the neighbour of Source to mh.
    ↳ ( d, u)
while ( mh. size ()>0){
(d, u)  pair p =  mh. getmin ()
    if ( dis[u] < d){
        continue;
    }
    dis [u] = d;
    for( (w, v) of neighbours ){
        nbr = {w, v};
        new_distance = dis[u] + w;
        if ( new_ distance < dis[v]){
            mh. add ( {new_distance, v});
        }
    }
}
return dis;
```

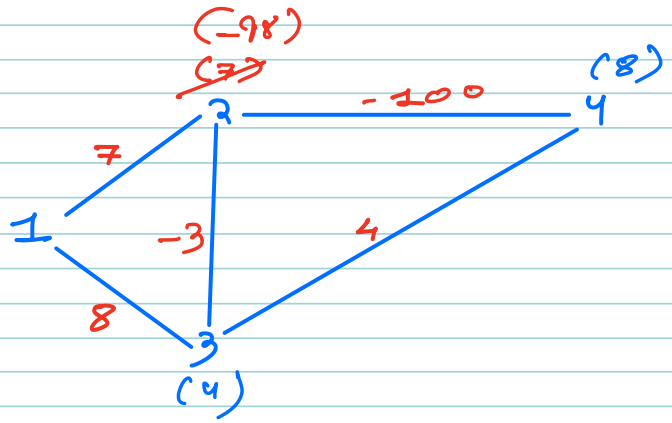


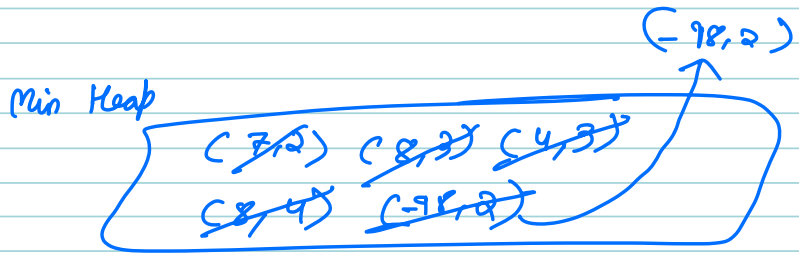
TC → O(E log E)

SC → O(N+E)

IMP NOTE

DIJKSTRA WON'T WORK OVER -ve NOS.





★ For -ve nos ⇒ Bell man Ford Work