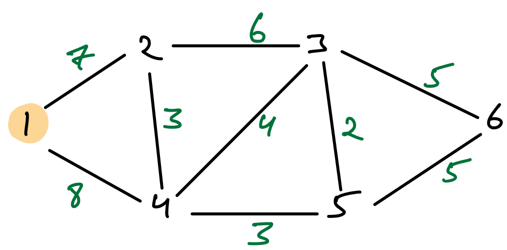
Welcome ☺

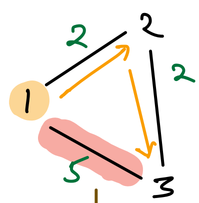
Agenda: Dijkstra's
Floyd Warshall Algo.
Coloring Graph
2 ques^n s.

Dijkstra's algo ⇒ Single source shortest path algo for weighted graph with +ve weights.

Q There are N cities in a country, you are living in city 1. Find min. distance to reach every city from city 1



$d[1] = 0$ (source)

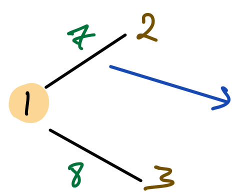


$d[1] = 0$
$d[2] = 2$
$d[3] = 2+2 = 4$

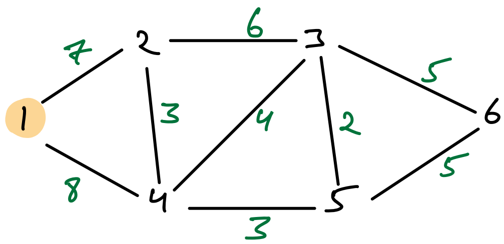
Relaxing an edge.

$$\text{if } (d[u\_w] > d[u\_v] + d[v\_w])$$
$$d[u\_w] = d[u\_v] + d[v\_w]$$

$\forall i > 1 \quad d[i] = \text{INT\_MAX}$

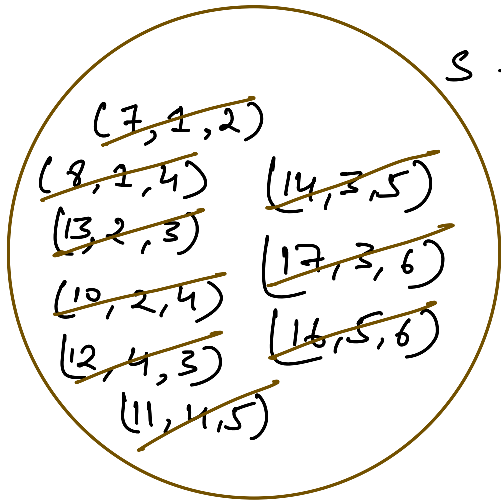


min. wt. edge.



[d[s] + wt , s , d]

s —wt— d



d = [	0	7	12	8	11	16]
	1	2	3	4	5	6

P = [	-1	1	4	1	4	5]
	1	2	3	4	5	6

Find the min. distance path from source (1) to city 6.

Ans ⇒ 6 ← 5 ← 4 ← 1

1) Insert all edges connected to source in min. heap.

2) Get min. distance path from heap, update distance of destination if it is INT_MAX

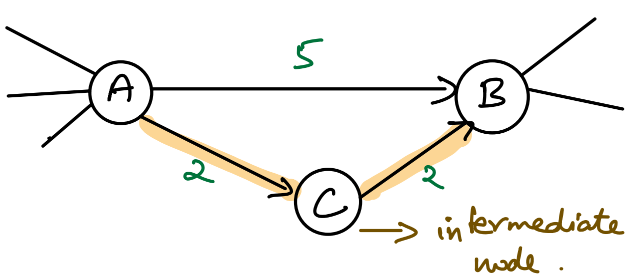
3) Insert all edges connected to the poped out element which are not directed to nodes for which we already have the answer. i.e d[n] = INT_MAX

T.C $O(E \log E)$ SC $O(E)$

Floyd Warshall Algo.

Q Find min. distance from every node to every other node.

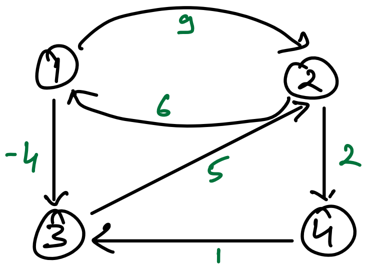
Ans - 2D array.



Is shortest distance from A to B is 5 ?

NO

∀ nodes, consider it as a intermediate node, & try all alternate paths.



Adj. matrix

	1	2	3	4
1	0	9	-4	0
2	6	0	0	2
3	0	5	0	0
4	0	0	1	0

1 is my intermediate.

$d(2-3)$ ∞ $d(2-1) + d(1-3)$ $6 + -4 = 2$

$d(2-4) = 2$ $d(2-1) + d(1-4)$ $6 + \infty = \infty$

$d(3-2) = 5$ $d(3-1) + d(1-2)$ $\infty + 9 = \infty$

Distance

D_1

	1	2	3	4
1	0	9	-4	∞
2	6	0	2	2
3	∞	5	0	∞
4	∞	∞	1	0

D_2 =

	1	2	3	4
1	0	9	-4	11
2	6	0	2	2
3	11	5	0	7
4	∞	∞	1	0

2 is my intermediate.

$d(3-1)$ ∞ $d(3-2) + d(2-1)$ $5 + 6 = 11$

$d(1-4)$ ∞ $d(1-2) + d(2-4)$ $9 + 2 = 11$

$d(3-4)$ ∞ $d(3-2) + d(2-4)$ $5 + 2 = 7$

Similarly consider 3 & 4 as intermediate node to get ans.

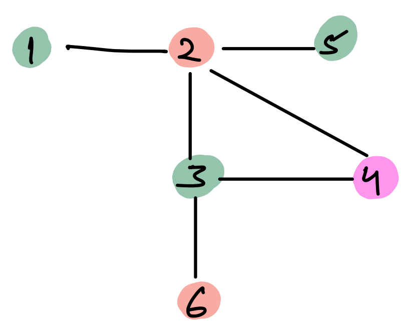
Code

```
for ( k → 1 to N)   ← decides intermediate node
{
    for (i → 1 to N)   ← source
    {
        for ( j → 1 to N)   ← destination.
        {
            if( d[i][j] > (d[i][k] + d[k][j]))
            {
                    d[i][j] = (d[i][k] + d[k][j])
            }
        }
    }
}
return d.
```

T.C $= O(N^3)$ S.C $= O(1)$

Graph Coloring

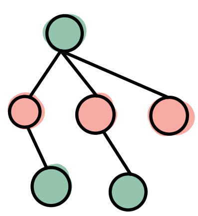
→ Color all nodes of graph, s.t no 2 adjacent nodes are of same color & count of distinct colors is minimum.
→ chromatic numbers. (N-P hard)



Ans = cnt = 3.

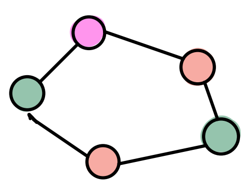
Special cases.

1) Trees

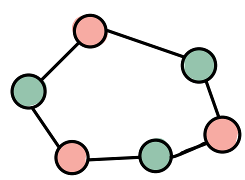


cnt = 2

2) Cycle graph



Ans = 3

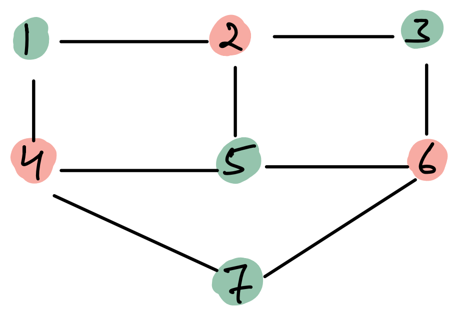


Ans = 2

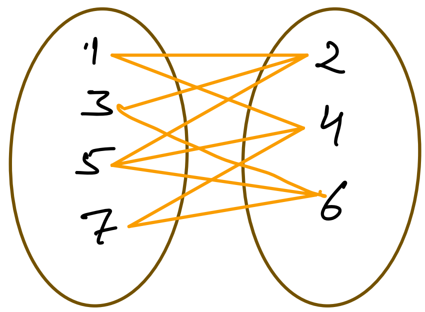
CN = 2 + (N%2)

Bipartite graph ← (CN = 2)

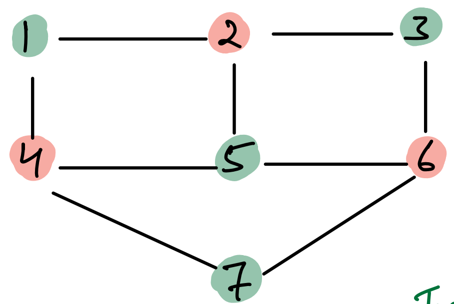
eg:



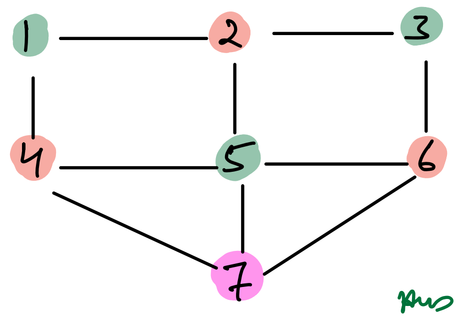
⇒ Graph can be divided into two sets s.t there is no edge b/w nodes in the same set.



Q Check if the graph is bi-partite.



Ans = True



Ans = False.

Soln Travel and color all nodes s.t no 2 adjacent nodes have same color. If not possible, return false.

$\forall i$, col[i] = -1

col[s] = 0   0, 1 → colors

boolean dfs (u) → assumed the graph to be connected and undirected. Otherwise, handle accordingly.

```
boolean dfs ( u )
{
    for ( v: adj[u])
    {
        if ( col[v] == -1)
        {
            col [v] = 1 - col[u]
            if ( !dfs(v)) return false.
```

```
        }
        else if ( col[v] == col[u] )
            return false.
    }
    return true.
}
```

T.C → O(N+E)

S.C → O(N)

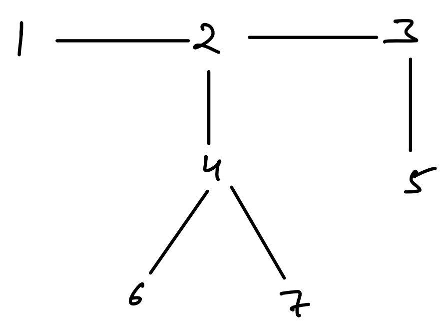
Q A country consist of N cities connected by (N-1) roads. King of that country want to construct max. roads s.t the cities can be divided into 2 sets & there is no road b/w cities in the same set. ← bipartite graph.

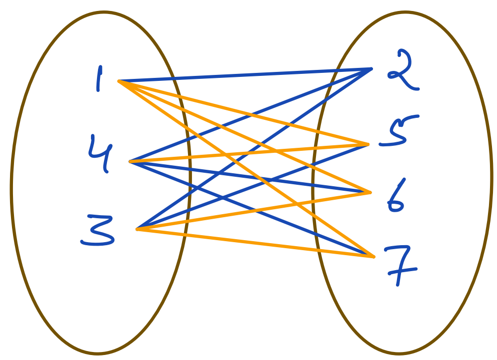
Tree

Find max # new roads king can construct.

eg:

N=7





ans = 6

I/P → count of elements in set 1 & set 2

T.C → O(N+E)

S.C → O(N)

Ans = New roads = $\underbrace{(cnt1 * cnt2)}_{\text{Total roads}} - \underbrace{(N-1)}_{\text{existing roads}}$

Is ans = 0 possible ?