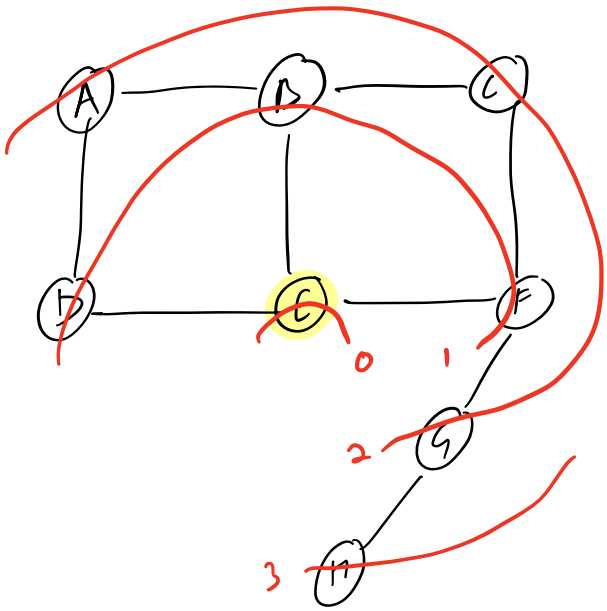
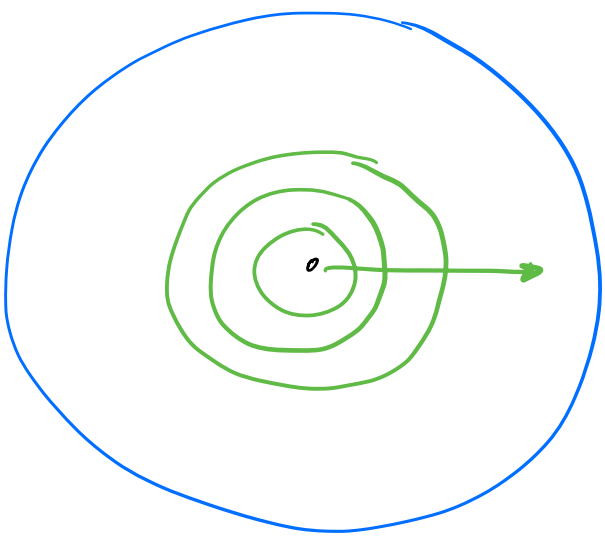
Given a graph (Unweighted).

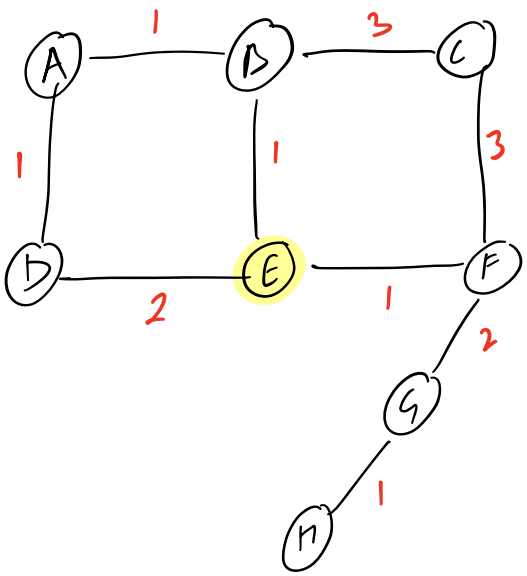
Find the min. distance from a source node to all other nodes!



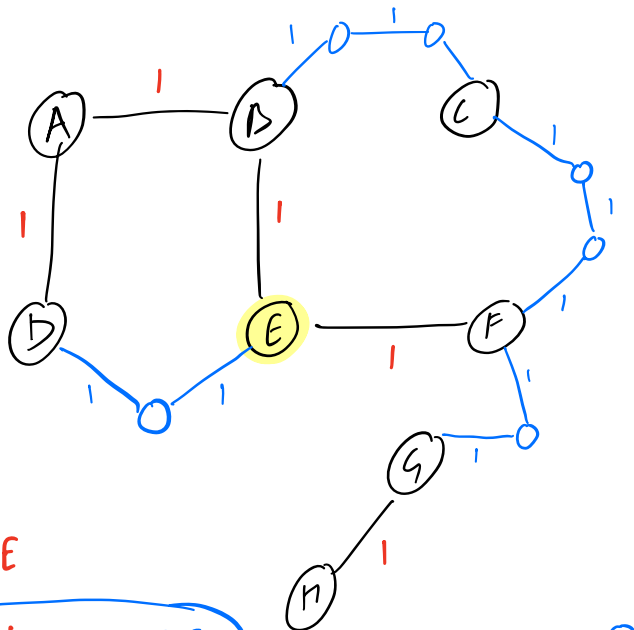
bfs



8 SAME QUES. Weighted → $W_i \in \{1, 2, 3\}$



BFS

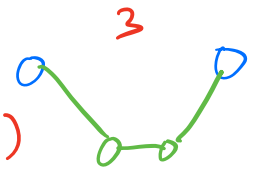


BFS ✓

G → V, E

G' → V' = V + 2E

E' = 3E

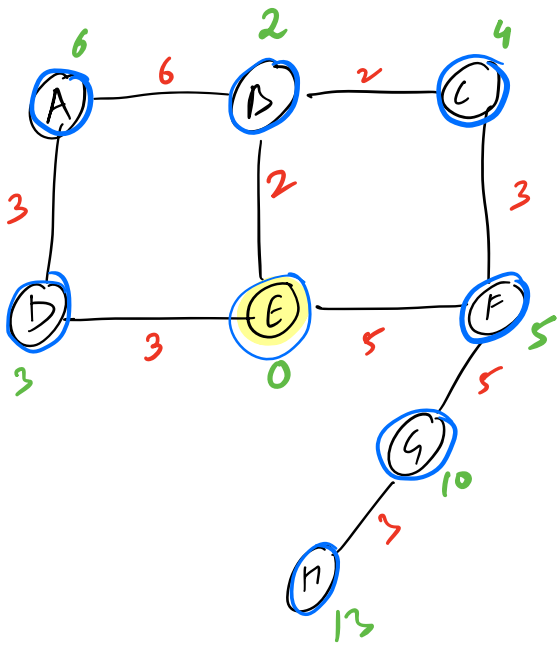


TC = O(V' + E')

TC = O(V + E)

SC = O(V') → O(V + E)

Q SAME QUES $w_i \in [1 \ldots 10^9]$



~~<D, 3>~~
~~<B, 2>~~
~~<F, 5>~~
~~<A, 8>~~
~~<C, 9>~~
~~<A, 6>~~
~~<F, 7>~~
~~<G, 10>~~
~~<H, 15>~~

Dijkstra's Algorithm

[Single Source Shortest path]

[edge weights → +ve]

→ find the shortest distance from a source node to all other nodes!

I/P → G, S

O/P → d[] // distance Array!

Requirement of DS →

- INSERT
- GET MIN
- DEL MIN

} → MIN HEAP

```
int d[N+1] → {∞}
MinHeap < pair<int, int> > mh;
```

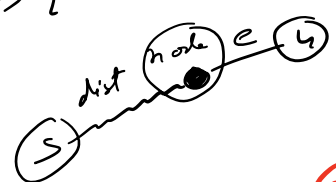
.first → dist from source

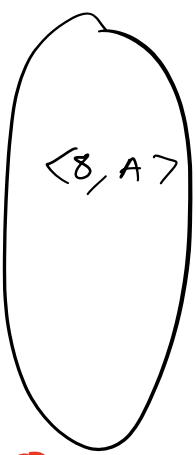
.second → node

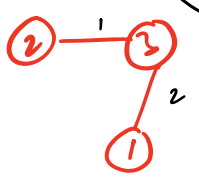
<8, A>

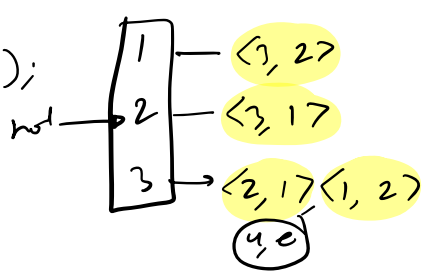
<0, S>

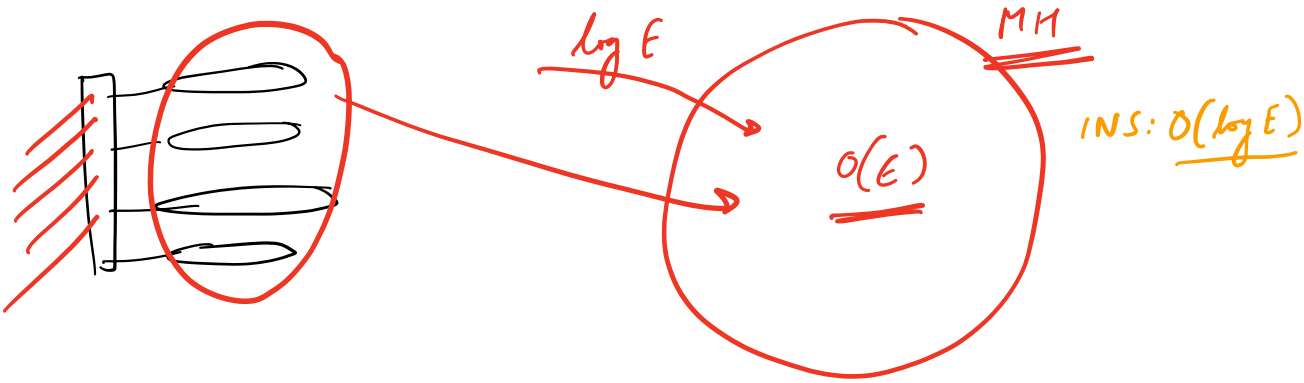
```
mh.insert({0, s});
while( !mh.isEmpty()){
    pair<int, int> p = mh.getMin();
    mh.delMin();
    dist = p.first,   node = p.second;
    if( d[node] != ∞) {
        continue;
    }
    d[node] = dist;
    for( (u, e) : adj[node]) {
        if( d[u] == ∞){
            Mh.insert({dist+e, u});
        }
    }
}
→ ret d[]
```











TC: $O(V + E \log(E))$

SC = $O(E)$

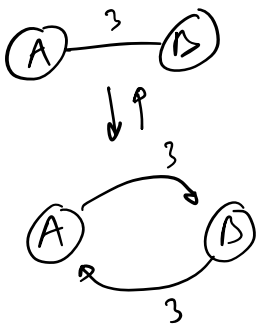
Floyd Warshall Algo

[-ve weights]

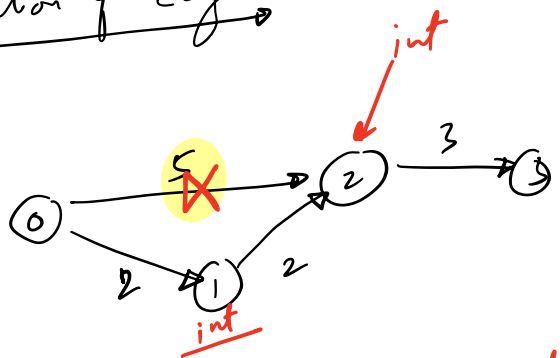
[All pair shortest path]

[Directed edges]

→ find the shortest dist b/w every pair of node



① Relaxation of edge



$d(0,2) = \min(d(0,2),\ d(0,1) + d(1,2))$

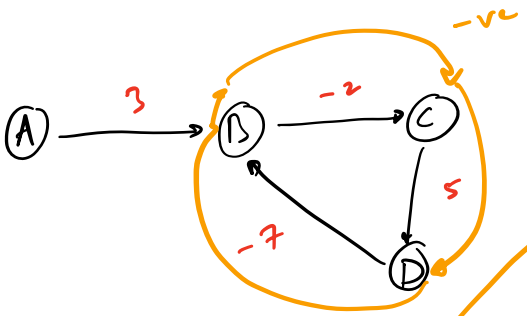
$\min(5,\ 2+2)$

$d(0,2) = 4$

$d(0,3) = \min(d(0,3),\ d(0,2) + d(2,3))$

$\min(\infty,\ 4+3)$

$= 7$



d(A, B)

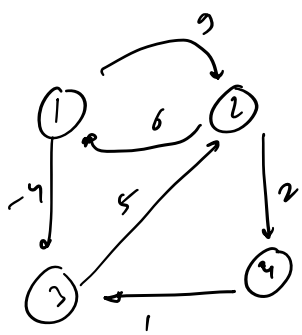
cycle → -ve total weight

3
-1
-5
-9
-13
⋮
$-\infty$

there is NO shortest dist
A → B

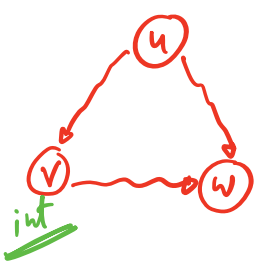
→ floyd W. Can detect -ve weight cycle!

Idea: Consider every node as intermediate node & relax as many edges as possible!



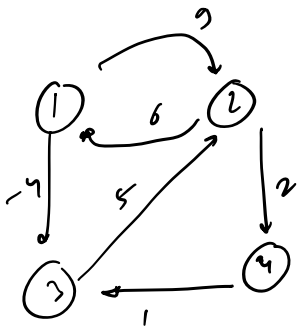
Adj MAT

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

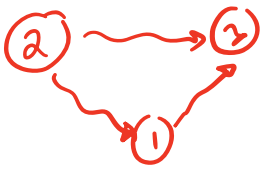


$$d(u,w) = \min\left(d(u,w),\ d(u,v) + d(v,w)\right)$$

1) Consider Node 1 as intermediate node



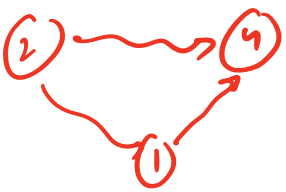
	1	2	3	4
1	0	6	-4	∞
2	6	0	2	2
3	∞	5	0	∞
4	∞	∞	1	0



$d(2,3) = \min(d(2,3),\ d(2,1)+d(1,3))$

$\min(\infty,\ 6+(-4))$

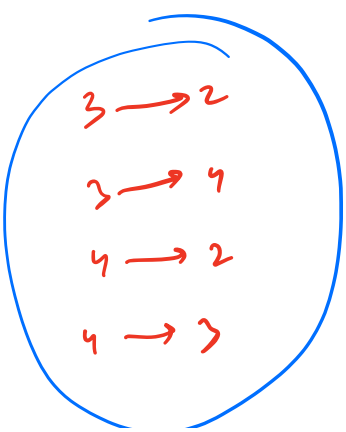
$= 2$ //



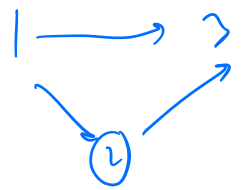
$d(2,4) = \min(d(2,4),\ d(2,1)+d(1,4))$

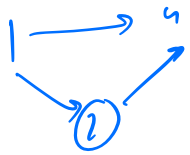
$\min(2,\ 6+\infty)$

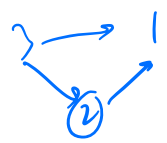
2 //

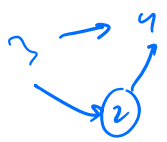


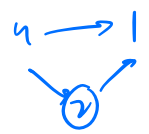
2) Consider Node 2 as intermediate node.

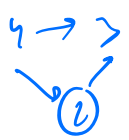








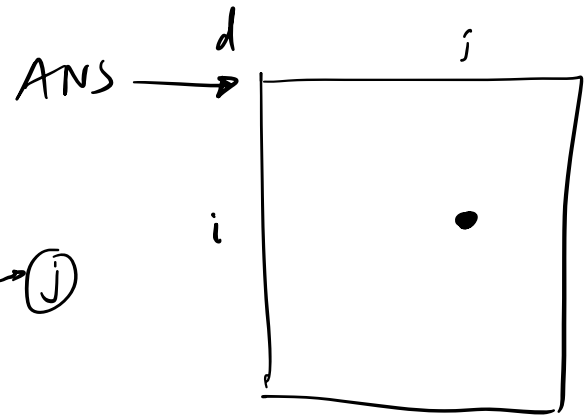




3) Consider node 3 as intermediate node

4)

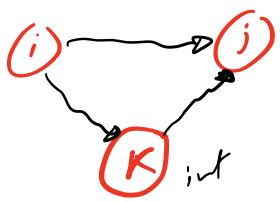
5)



d[i][j]
→ min dist from (i) → (j)

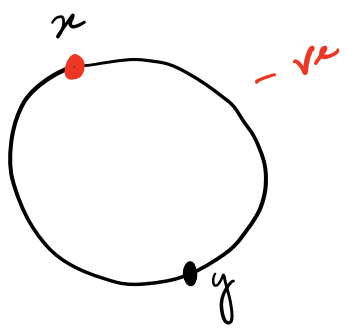
I/P → // Adj Mat ⟶ d[][]

```
for (k = 1; k <= V; k++) {   // k is int. node
    for (i = 1; i <= V; i++) {
        for (j = 1; j <= V; j++) {
            if (i != k && j != k && i != j) {
                d[i][j] = min(d[i][j],
                              d[i][k] + d[k][j]);
            }
        }
    }
}
ret d[][];
```



TC = O(V³)

SC = O(1)

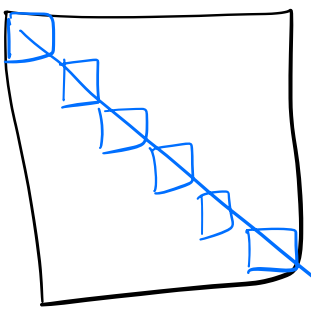


$d(x,x)$ via y $= \min(d(x,x), d(x,y)+d(y,x))$

$d(x,y)+d(y,x)$ → $-ve$

$\min(0, -ve)$

$-ve$



if ANY

$-ve$: negative cycle present

if $(d[i][i] < 0)$

for any i :

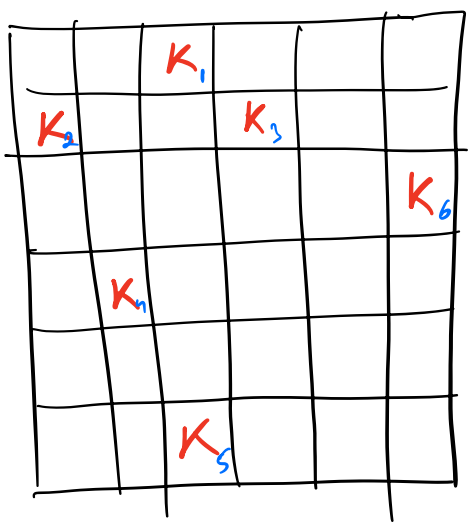
Q

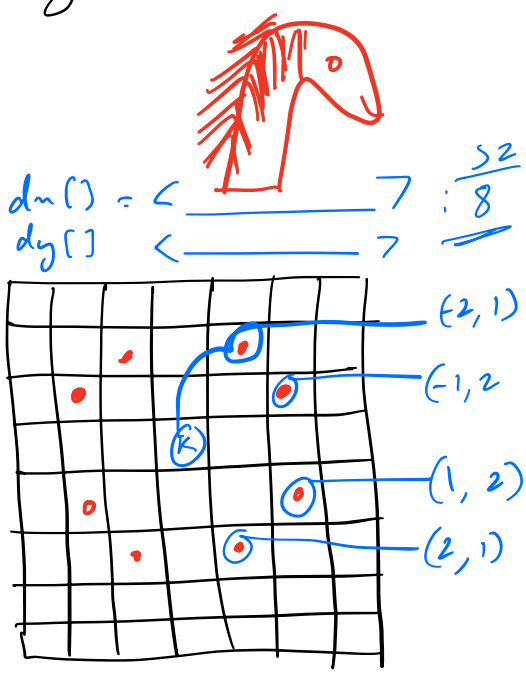
Given a chessboard (N × N)

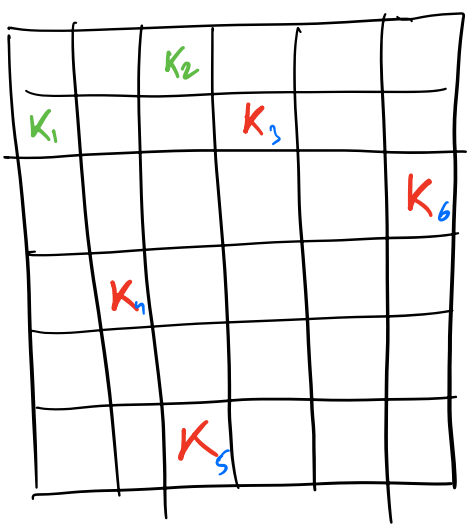
You are given K Knights placed on it !

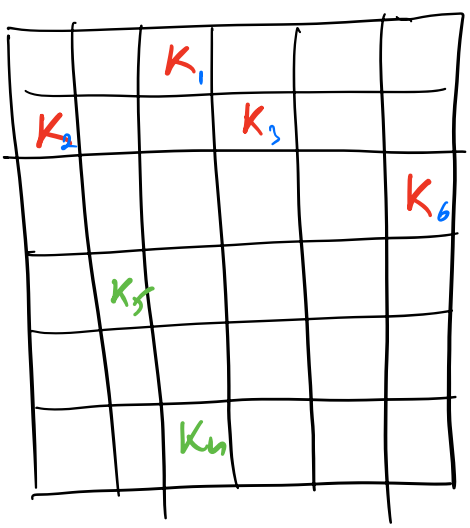
If a Knight is reachable from other Knight in 1 step → you can swap them.

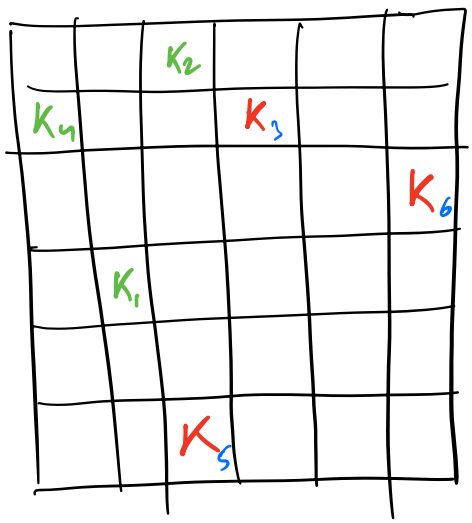
Find the no. of ways of arranging the Knights

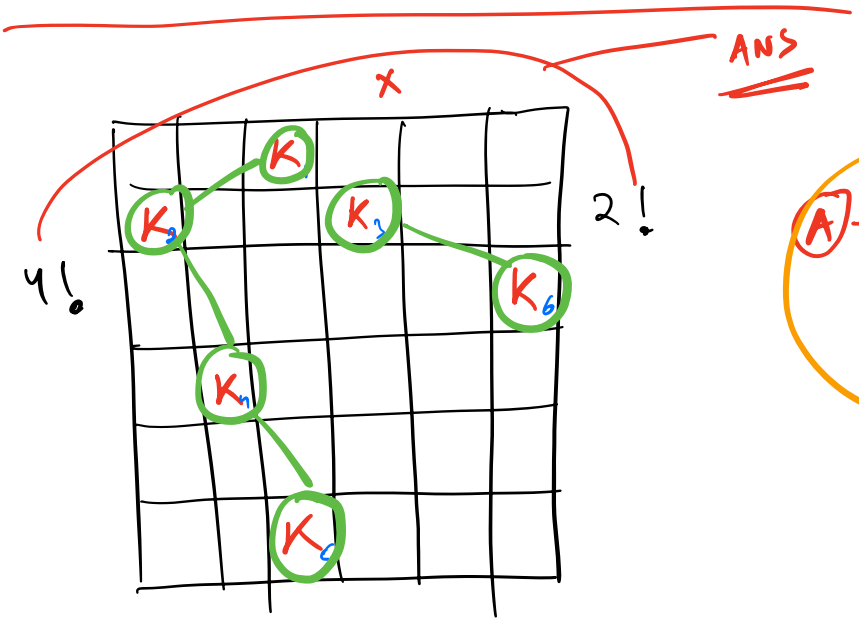


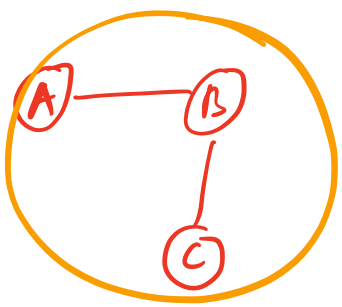






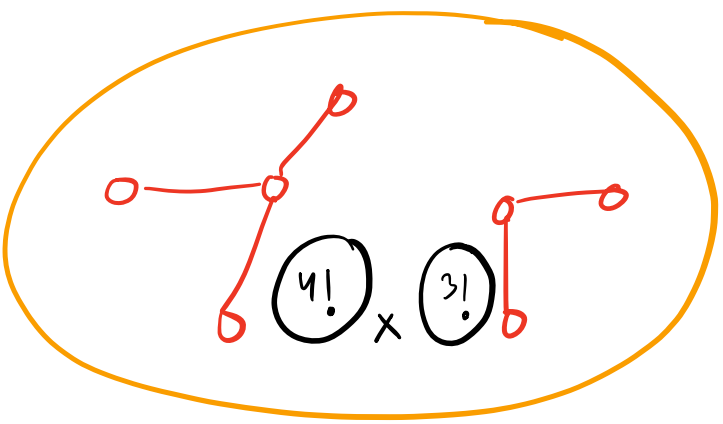


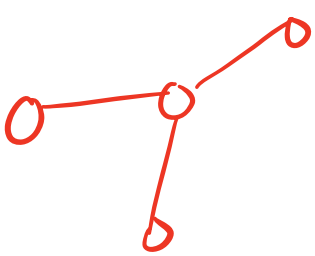




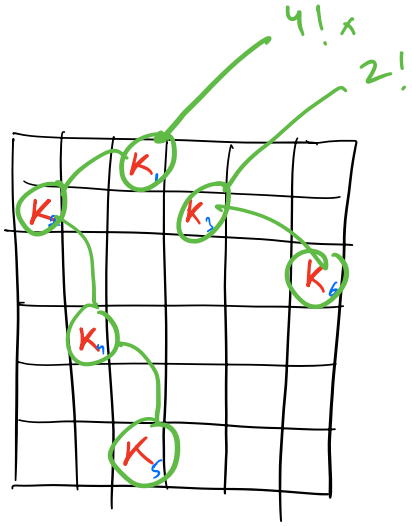
3!

= 6





= 4!



$V = N^2$

$E = \sim N^2$

$TC = O(N^2)$

$SC = O(N^2)$