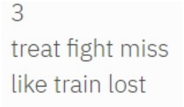
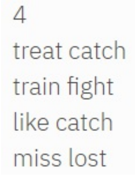


```
3
treat fight miss
like train lost
```

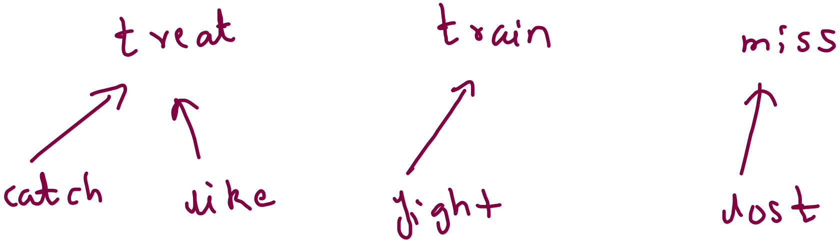
s1 : treat fight miss

s2 : like train lost

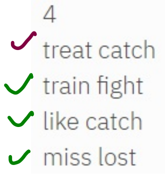


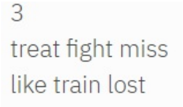
```
4
treat catch
train fight
like catch
miss lost
```

→ create DSU →



```
Hashmap < string, string > parent;
Hashmap < string, integer > rank;
```


4
✓ treat catch
✓ train fight
✓ like catch
✓ miss lost


3
treat fight miss
like train lost

parent

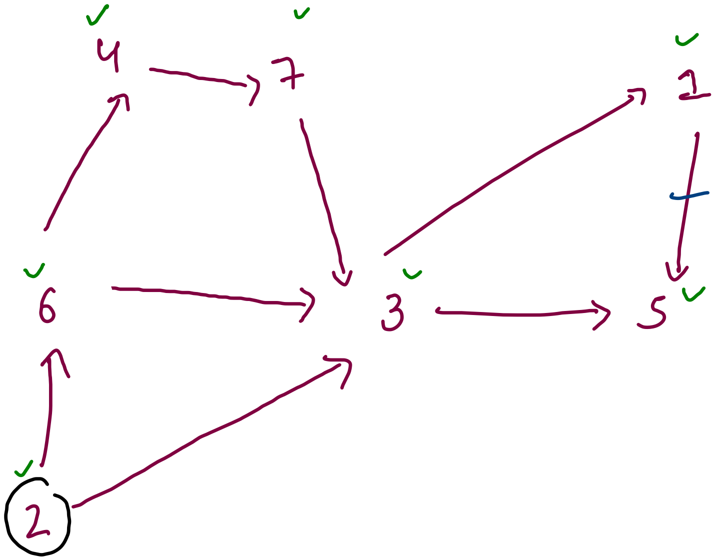
treat ->	treat
catch ->	treat
train ->	train
fight ->	train
like ->	treat
miss ->	lost
lost ->	lost

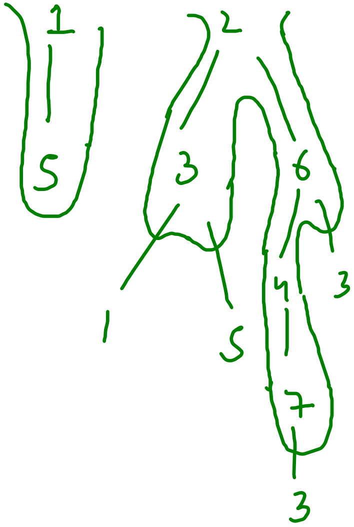
rank

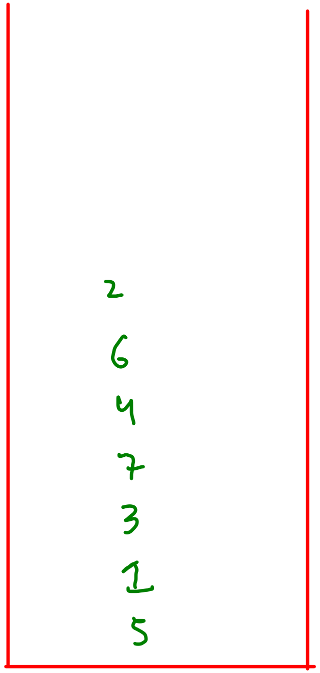
treat ->	1
catch ->	0
train ->	1
fight ->	0
like ->	0
miss ->	0
lost ->	1

Kahn' algo : topological sort (DAG)

directed acyclic graph







u → v.

u should come before v

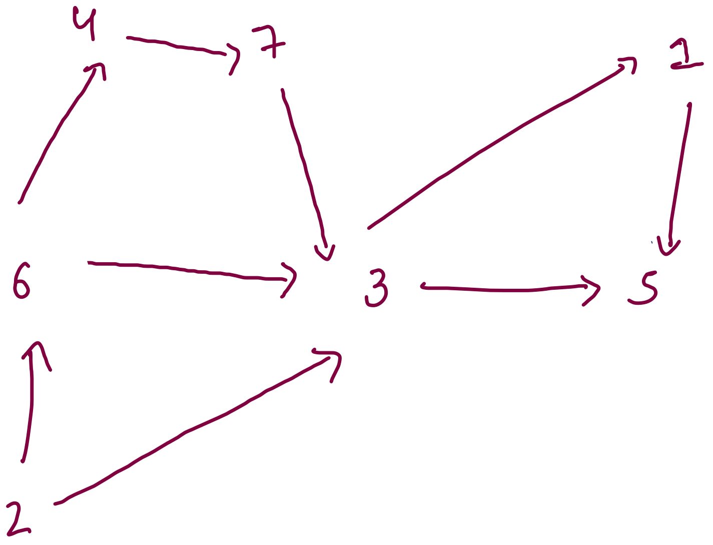
u → v, u depends on v

ts: 2 6 4 7 3 1 5

ordn of comp: rev(ts)

↳ 5 1 3 7 4 6 2

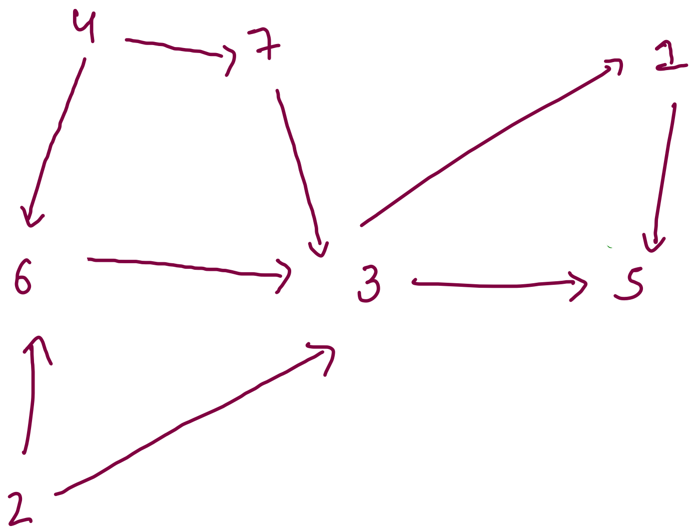
Kahn's algo



indegree	0 ~~1~~	0	~~2~~ ~~1~~ 0 ~~3~~	0 ~~1~~	~~1~~ 0 ~~2~~	~~1~~ 0	0 ~~1~~
	1	2	3	4	5	6	7

~~2~~	~~6~~	~~4~~	~~7~~ ~~3~~	~~1~~	~~5~~	

2 6 4 7 3 1 5



Kahn's algo

① . fill indegree array.

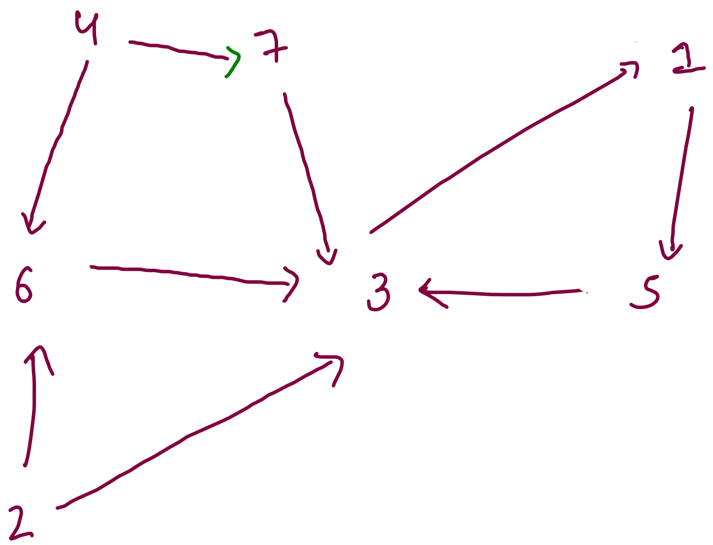
~~1~~ 0	0	~~1~~ 0 ~~2~~ ~~3~~	0	~~1~~ 0 ~~2~~	0 ~~2~~ ~~1~~	0 ~~1~~
1	2	3	4	5	6	7

② . Start with '0' indegree vertices

~~2~~	~~4~~	~~7~~	~~6~~	~~3~~	~~1~~	~~5~~

2 4 7 6 3 1 5

cycle detect:



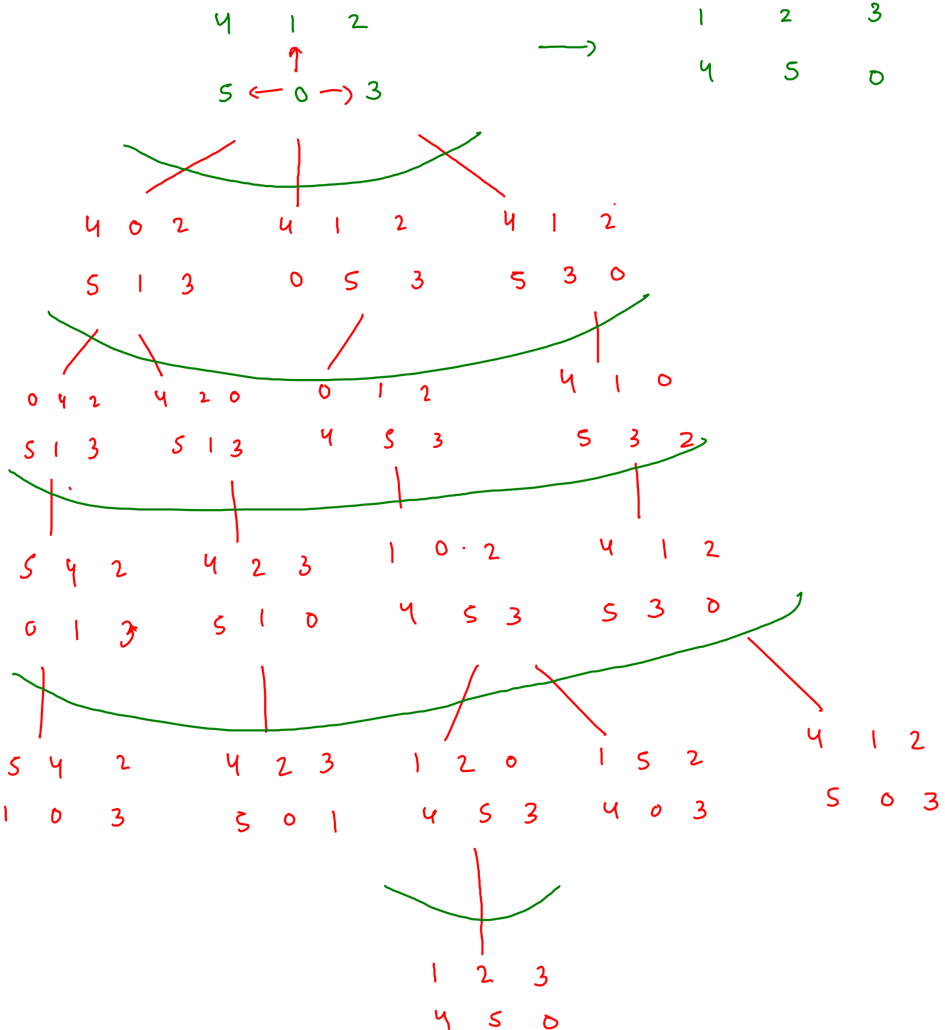
1	0	~~4~~ ~~3~~ ~~2~~ 1	0	1	~~2~~ ~~1~~ 0	~~1~~ 0
1	2	3	4	5	6	7

~~4~~	~~2~~	~~7~~	~~6~~

(4, 2, 7, 6)

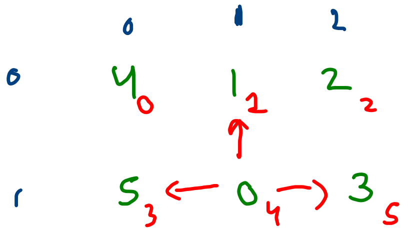
count = 0 + 1 + 1 + 1 + 1

Sliding puzzle



9K

4 0 2
5 1 3



0	1	2	3	4	5
4	1	2	5	0	3

m -> total cols

bno = i*m+j;

dest -> 1 2 3 4 5 0

i = bno / m;

j = bno % m;

~~412503~~	402513	412053	412530

2d -> string ;

m = 3

0	1	2	3	4	5
4	1	2	5	0	3

0 { i = 1, j = 1

$\xrightarrow{t}$ (0,1) $\xrightarrow{\text{bno.}}$ 1

$\xrightarrow{d}$ (1,0) $\xrightarrow{\text{bno.}}$ 3

$\xrightarrow{r}$ (1,2) $\xrightarrow{\text{bno.}}$ 5

(i) easy to maintain visited , bfs.

(ii) difficult to find nbrs .

map : [[1, 3] , [0, 2, 4] , [1, 5] , [0, 4] , [1, 3, 5], [2, 4]]

0	1	2	3	4	5

	0	1	2
0	0	1	2
1	3	4	5

~~412503~~	402513	412053	412530

str = $\overset{0}{4}\,\underline{\overset{1}{1}}\,\overset{2}{2}\,\overset{3}{5}\,\underline{\overset{4}{0}}\,\overset{5}{3}$

↓

sb = 4 0 2 5 1 3

↓

str = 4 1 2 5 0 3

$\overset{0}{4}\ \overset{1}{1}\ \overset{2}{2}\ \overset{3}{5}\ \boxed{\overset{4}{0}}\ \overset{5}{3}$

map (4) ⟶ nbr : 1, 3, 5

402513 412053 412530

1 0 2

4 5 3

→

1 2 3

4 5 0

~~102453~~	~~012453~~	~~120453~~	152403	412053

102453

012453

120453

412053

3 0

1 2 ~~0~~ 4 5 ~~3~~

0 1 2 3 4 5

ans → 2

[[1, 3] , [0, 2, 4] , [1, 5] , [0, 4] , [1, 3, 5] , [2, 4]]

0 1 2 3 4 5