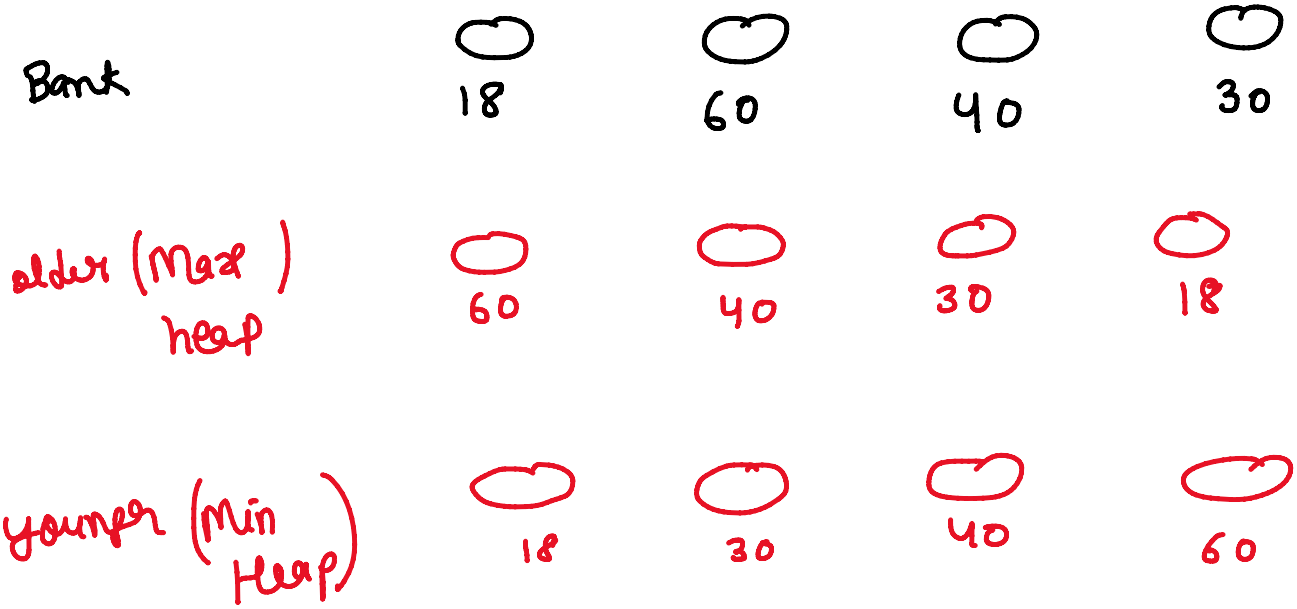
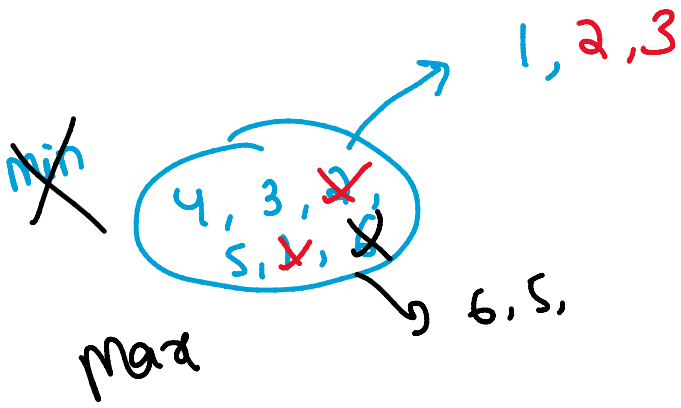
Heap/Priority Queue

22 October 2022 13:03

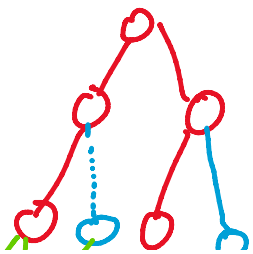


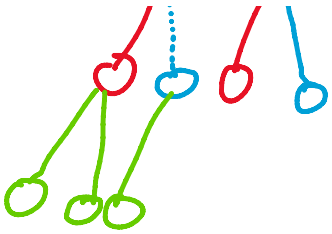


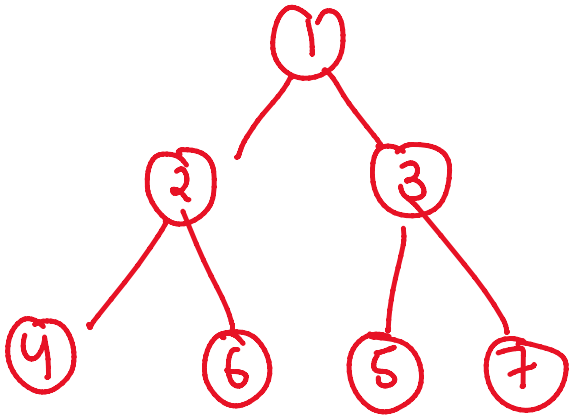
1. Binary tree (at max 2 children)
2. Complete binary tree. (left to right direction).
3. Heap order property → Min
   → Max

all levels will be completely filled. Except the last level. and last level will filled from L to R.

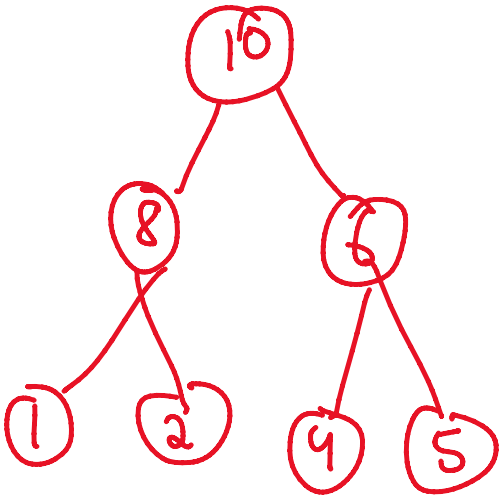
yes







Min Heap



Max Heap

Insertion - log N
Deletion - log N
top - O(1)

insertion
deletion

C++
push
pop
top

Java
add / offer
poll / remove
peek

max heap

priority_queue <int> pq; max

priority_queue <int, vector<int>, greater<int>> pq; min

priorityQueue <Integer> pq = new priorityQueue<>(); //min heap

//max heap. collections.Array

Kth largest element in an array

22 October 2022 13:22

① Sorting. O(N logN).

②

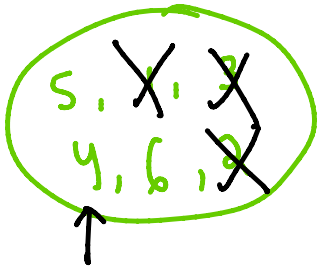
Min Heap

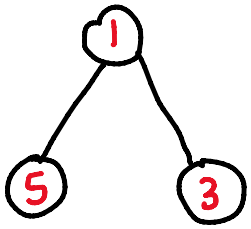
5, 1, 3, 4, 6, 2

K = 3

output = 4

minHeap (K)





.top

Min Heap

time: N log K

space: O(K)

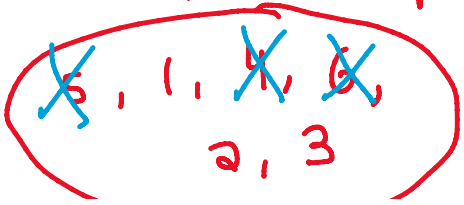
→ 4

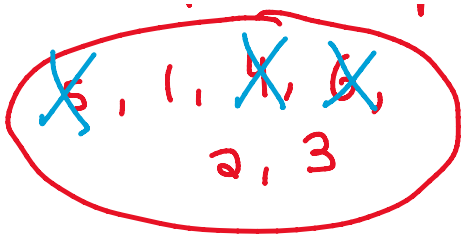
2. Max Heap

5, 1, 4, 6, 2, 3

K = 3

Max Heap N





6 5 4

longest 1 2 3

space O(N)

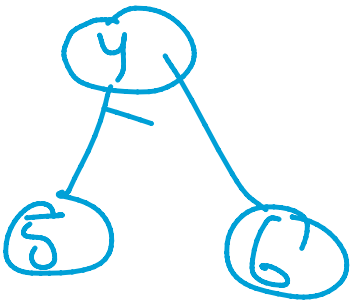
timer (N logN)

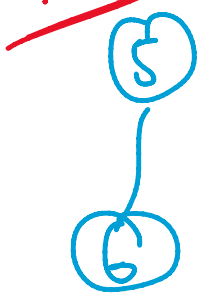
1, 5, 6, 4, 2, 3

min heap

k = 3

k = 2



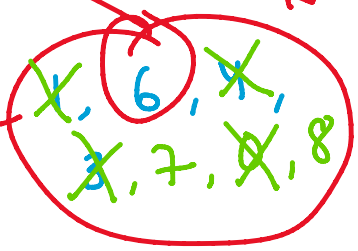


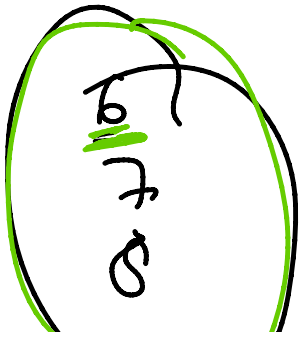
1, 6, 4, 3, 7, 0, 8

k = 3

min heap (k)

output = 6

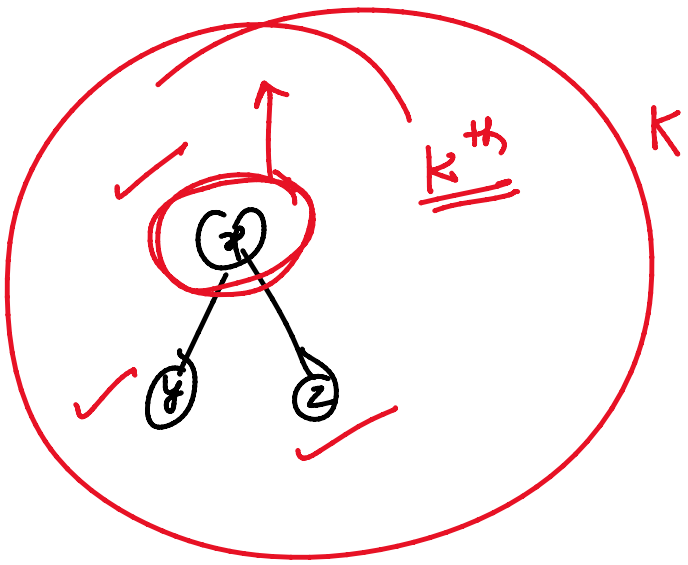




k^{th} largest → min Heap

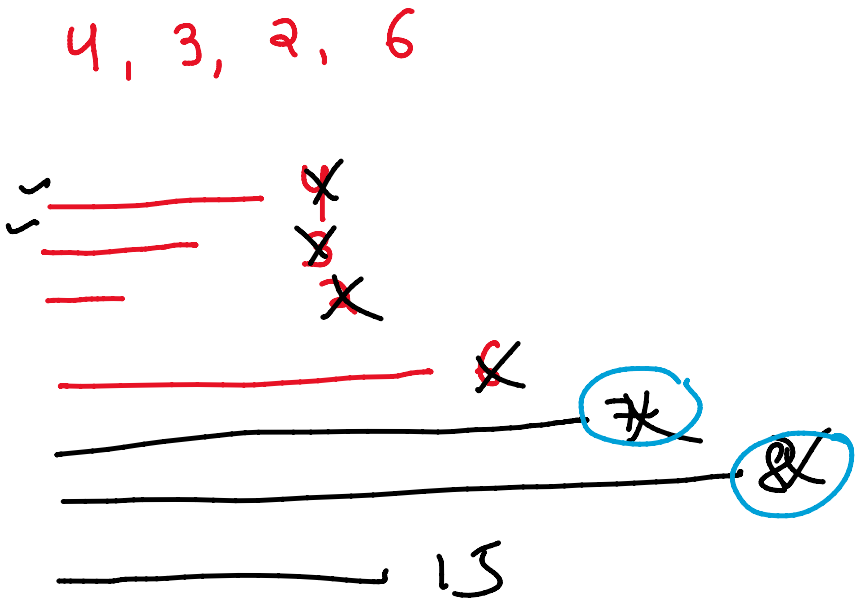
k^{th} smallest → max Heap

closest



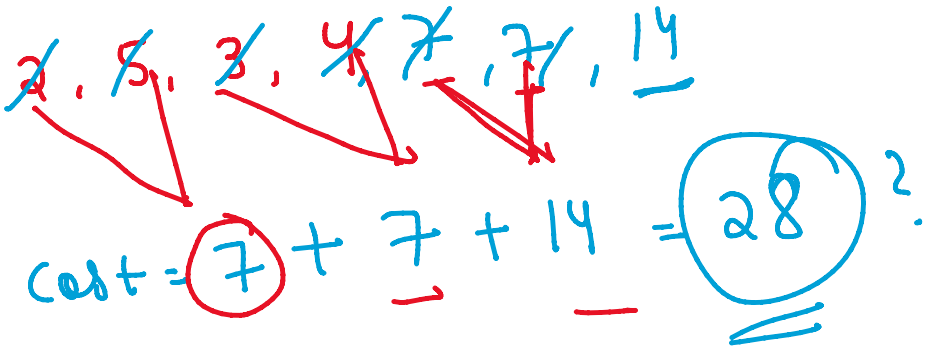
Min cost ropes

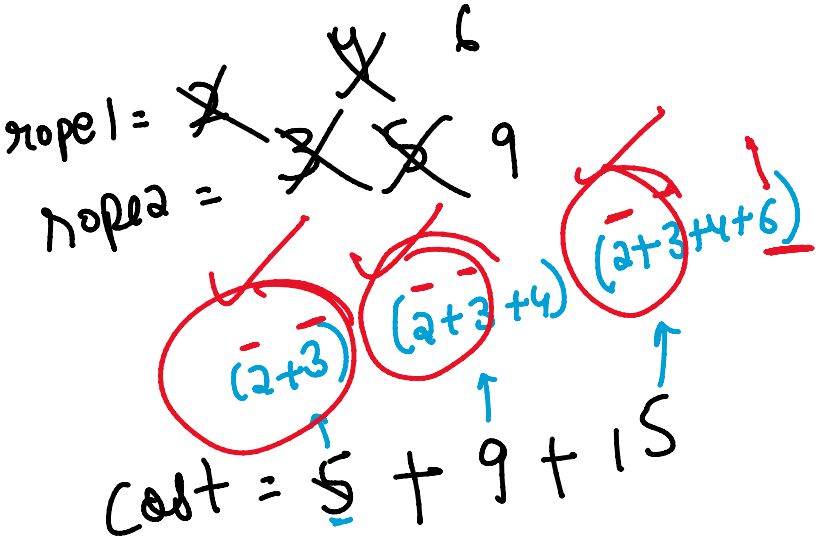
22 October 2022 13:46



cost = 7 + 8 + 15

= 30





why 2 min ropes?

= 29

3, 6, 2, 5

Min heap.

rope1 =

rope2 =

(2+3) (2+3+5) (2+3+5+6)

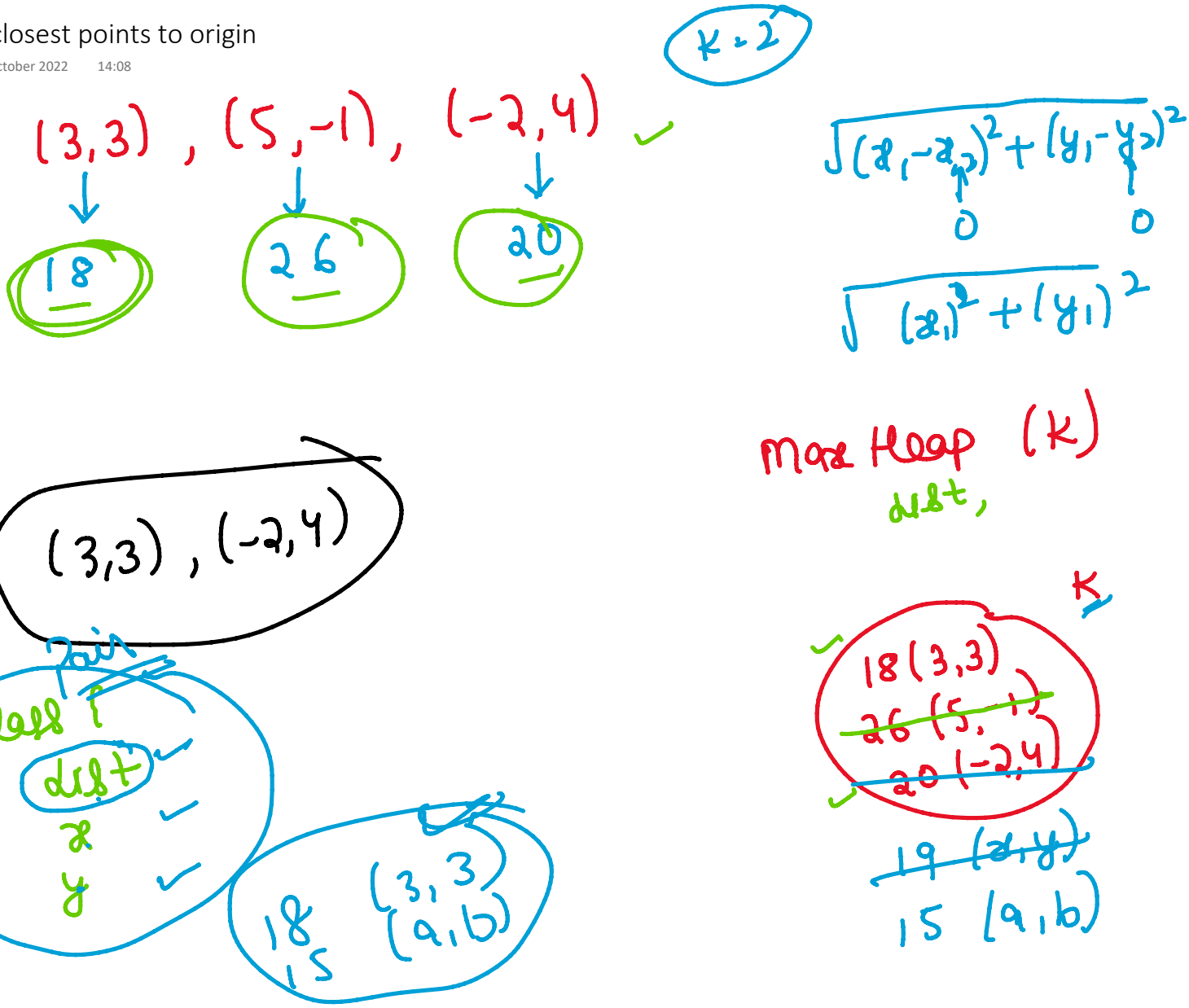
Cost = 5 + 10 + 16 = 31

space → O(N)

time → O(N log N)

K closest points to origin

22 October 2022 14:08



Merge K sorted arrays

22 October 2022 14:25

$K = 3$

$$\begin{bmatrix} 1 & 5 & 6 \\ 2 & 7 & 9 \\ 3 & 4 & 8 \end{bmatrix} \Rightarrow [1, 2, 3, 4, \ldots, 9]$$

space = $O(K^2)$

time = $K^2 \log K^2$

unsorted

Min Heap (K^2)

$9!$ (N)

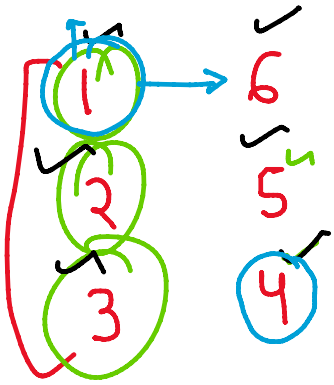
1

~~1~~, 5, 6, ~~7~~,

7, 9, ~~3~~, ~~4~~, 8

1, 2, 3, ---- 9

K^2 $\log K^2$ $\log N$



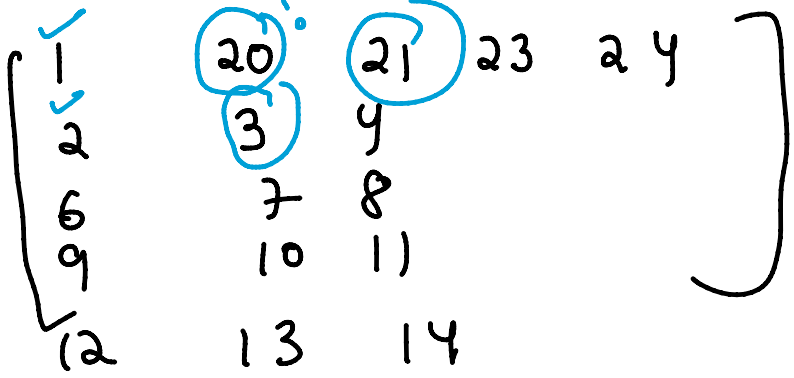
1, 2, 3, 4

min Heap (k)

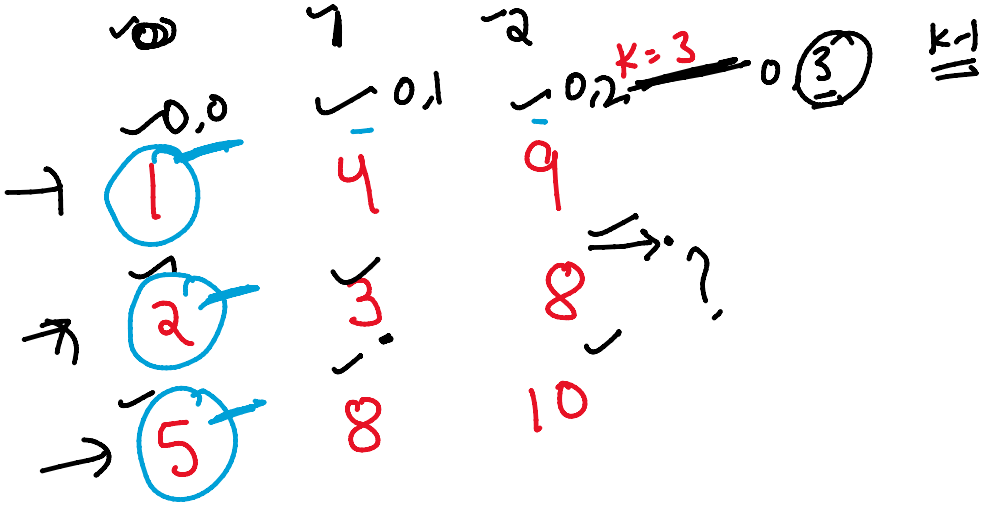
~~1~~, ~~2~~, ~~3~~

6, 5, ~~4~~

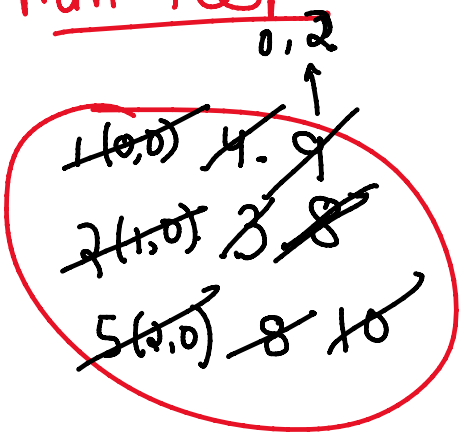
8



1(0,0), 6(0,1), 7(0,2)



min Heap K



return

[1, 2, 3, 4, 5, 8, 8, 9, 10]

space : O(k)

time : $K^2 \log K$

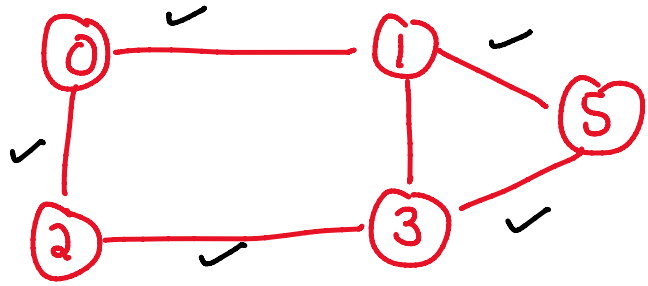
1. Merge K sorted lists ✓
2. Top K frequent elements ✓
3. Median of running stream.

1

Graph Intro

22 October 2022 15:18

1. Vertices / Nodes
2. Edges

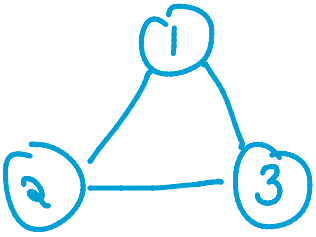


vertices: 0, 1, 2, 3, 5

Edges: 0-1, 1-5, 0-2, 2-3, 3-5, 1-3

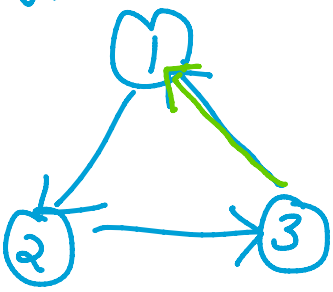
① Undirected and directed

undirected

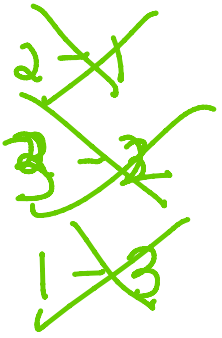


1-2, (2-1)
2-3, (3-2)
3-1, (1-3)

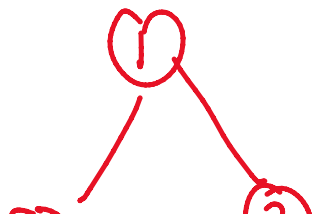
directed

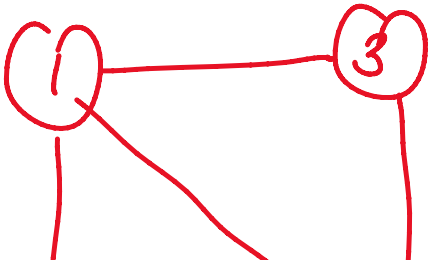


1-2
2-3
3-1

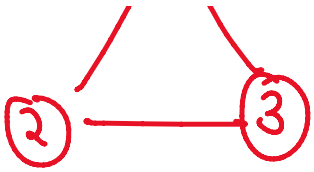


② Cyclic and Acyclic

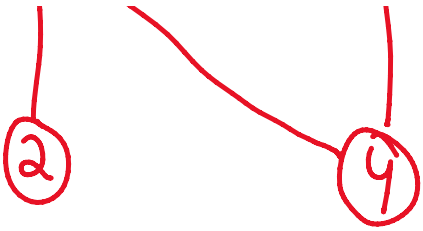




1-3-4

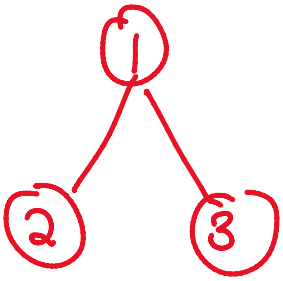


cyclic

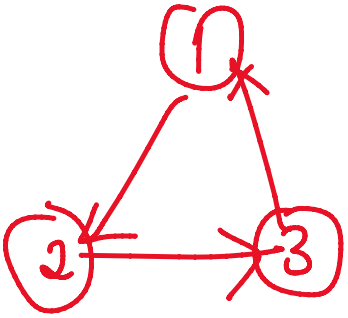


1-3-4

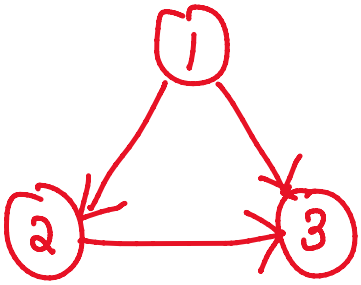
cyclic



acyclic



cyclic



× acyclic

DAG

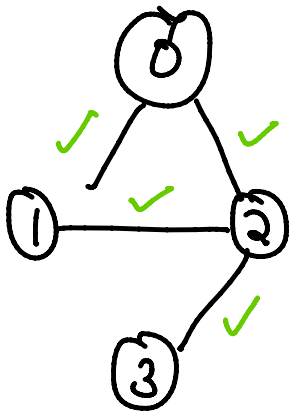
Representation of graph

22 October 2022 15:26

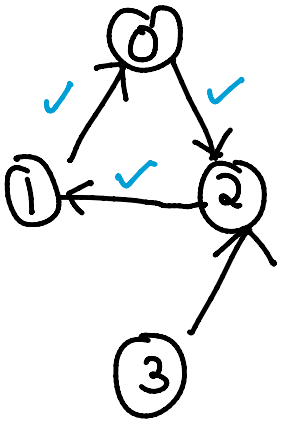
① Adjacency matrix

② Adjacency list

①

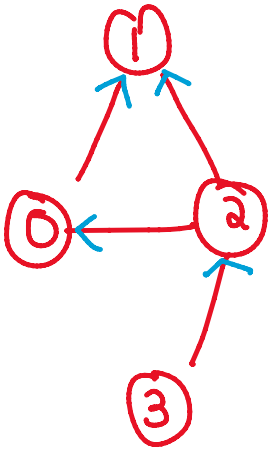


	0	1	2	3
0	0	1	1	0
1	1	0	1	0
2	1	1	0	1
3	0	0	1	0



	0	1	2	3
0	0	0	1	0
1	1	0	0	0
2	0	1	0	1
3	0	0	1	0

②



0 → (1, 2)

1 → (0, 2)

2 → (0, 1, 3)

3 → (2)

(3)

0 → 1

1 → -

2 → 0, 1

3 → 2

list < list >

vector <int> adj [ ]

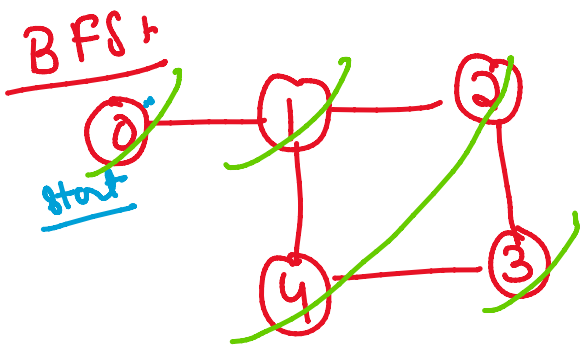
BFS

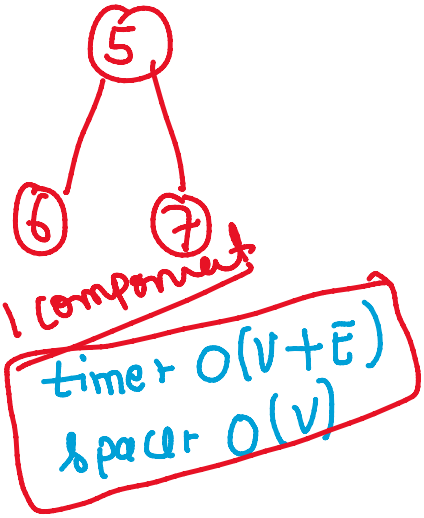
22 October 2022 15:35

2 traversals.

1. BFS - Breadth first search
   iterative

2. DFS - Depth first search
   recursive





0 → 1
1 → 0, 4, 2
2 → 1, 3
3 → 4, 2
4 → 1, 3
5 → 6, 7
6 → 5
7 → 5

for(0 to 7)

DFS

22 October 2022 15:57

Depth first search

- recursive

0 → 1,3

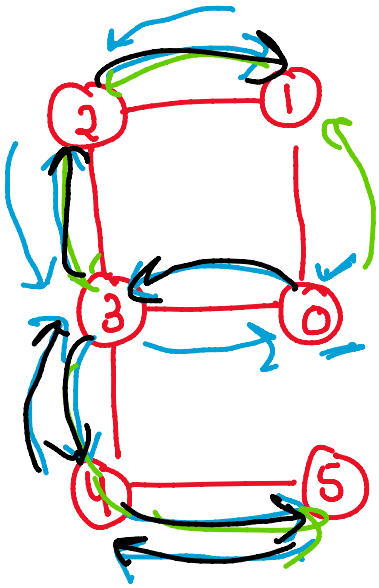
1 → 2,0

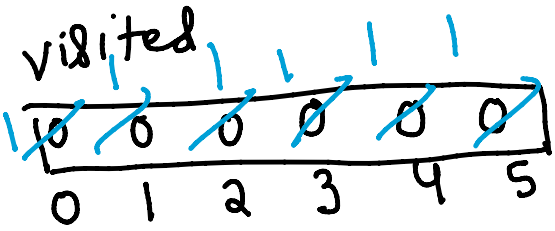
2 → 3,1

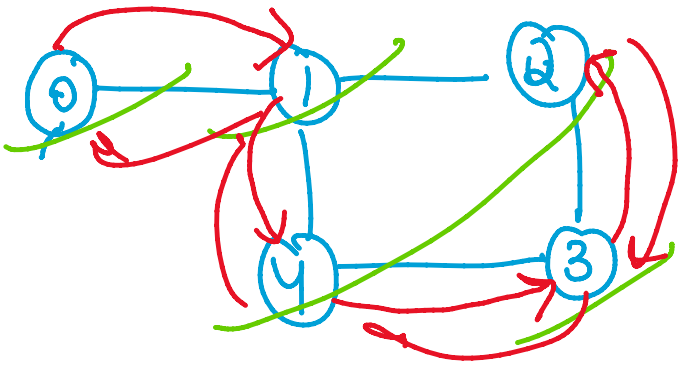
3 → 0,2,4

4 → 5,3

5 → 4







0, 3, 4, 5, 2, 1

0, 1, 2, 3, 4, 5

BFS: 0, 1, 4, 2, 3 / 0, 1, 2, 4, 3

DFS: 0, 1, 4, 3, 2 / 0, 1, 2, 3, 4