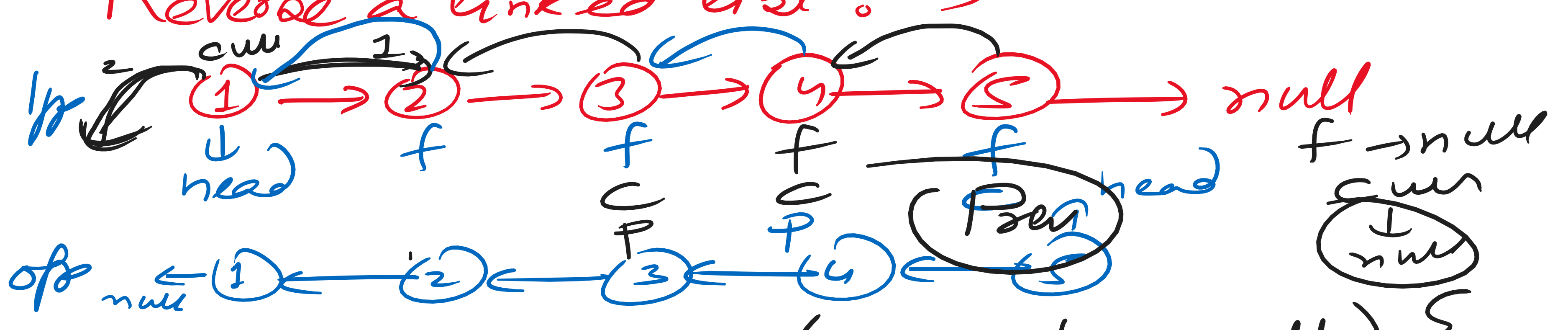
Reverse a linked list :→



prev = null
curr = head
forward = null

⟹ while (curr != null) {
forward = curr → next;
curr → next = prev;
prev = curr;
curr = forward;
} return prev;

*** Important Questions :→ (Arrays) (Heap)

1. Find (K largest) elements in an array : 3 largest element (ex)

f → add O(n)
f → di O(n)

arr = ~~20~~, ~~10~~, 60, ~~30~~, 50, 40

k = 3 → 40, 50, 60

min Heap

6
5 > 3
4 > 3 O(n)
3 > 3 stop
10 pop()

if minHeap.size() > K
6 > 3

K^{th} Smallest Element → max heap

7, ~~10~~ (3rd), 4, 3, ~~20~~ (1st), ~~15~~ (2nd) → size = 6

3rd Smallest :→ 7

→ top → print

7, 4, 3
3rd 2nd 1st

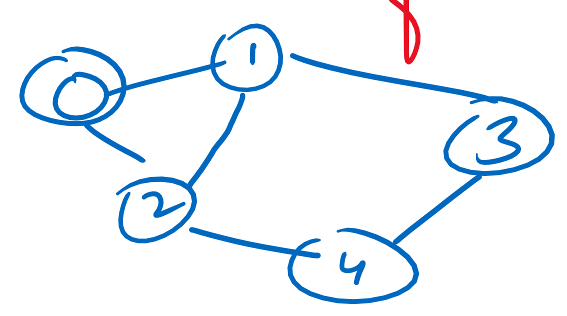
6 > K
6 > 3
5 > 3
4 > 3
3 > 3 no → stop

Graphs interview questions :→

Given an adjacency list of a graph, count :→

① No of nodes?
② No of edges? 12/2 = 6

0 → 1
0 ← 1
②



Adjacency List

Node	Neighbours
0	1, 2
1	0, 2, 3
2	0, 1, 4
3	1, 4
4	2, 3

Convert a given adjacency matrix into an adjacency list :→

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

o/p

java.util Queue

Adj list :

Node		Neighbours
0	→	1, 2
1	→	0, 2, 3
2	→	0, 1, 4
3	→	1, 4
4	→	2, 3

[i][j] == 1
adjList(i).push_back(j);