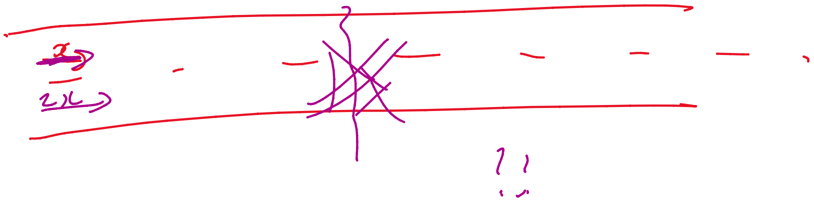
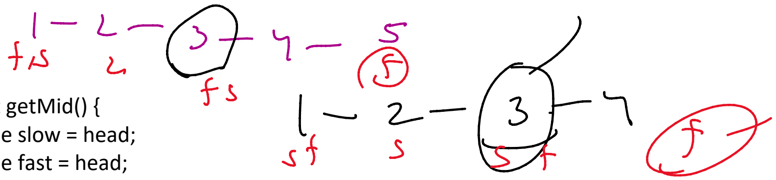
05 August 2023 13:51





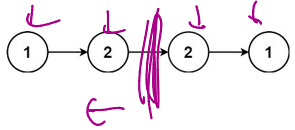
```
public int getMid() {
        Node slow = head;
        Node fast = head;
        while(true) {
            slow = slow.next;
            fast = fast.next.next;
        }
        return slow.data;
    }
```

fast != null

fast. next != null → odd

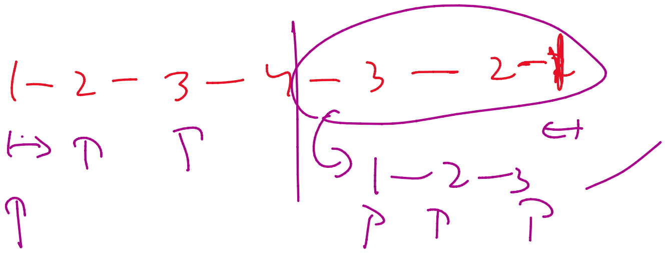
fast. next != null

Q1) M1) → 2 point s→ e→



O(n) T
O(n) S

M2) → O(n) time O(1) space



K reverse

K = 3

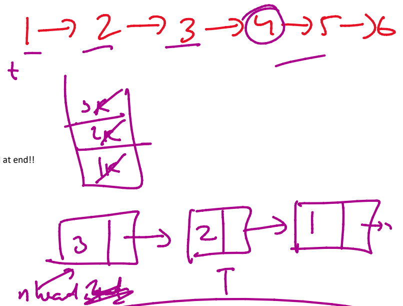
1 → 2 → 3 → 4 → 5 → 6 → 7 → 8 — 9

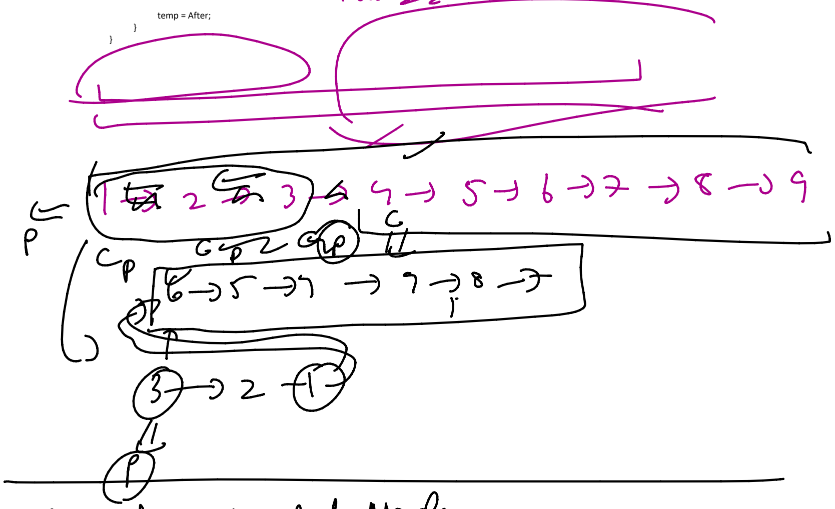
3 → 2 → 1 → 6 → 5 → 4 → 9 → 8 → 7

3 → 2 → 1 → 6 → 5 — 4 — 9 — 8 — 7

```
public void kReverse(int K) {
		Stack<Node> S = new Stack<>();
		Node temp = head;
		Node nhead = null;
		Node ntail = null;
		while (temp != null) {
			Node After = temp.next;
			S.add(temp);
			if (S.size() == K) {
				while (!S.isEmpty()) {
					Node nn = S.pop();
//						Linked List ke add at end!!
					if (nhead == null) {
						nhead = nn;
						ntail = nn;
						ntail.next = null;
					} else {
						ntail.next = nn;
						ntail = nn;
						ntail.next = null;
					}
				}
			}

			temp = After;
		}
	}
```





Q Find Kth last Node

1 → 2 → 3 → 4 → 5 → 6

M1) LL→ reve

x x

K-steps

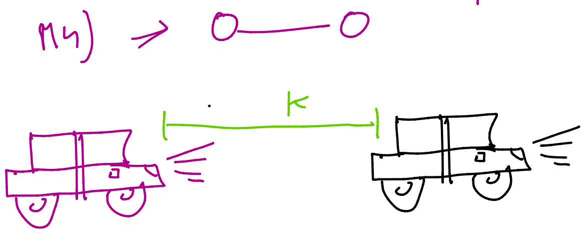
M2) O(n)

O(n) X

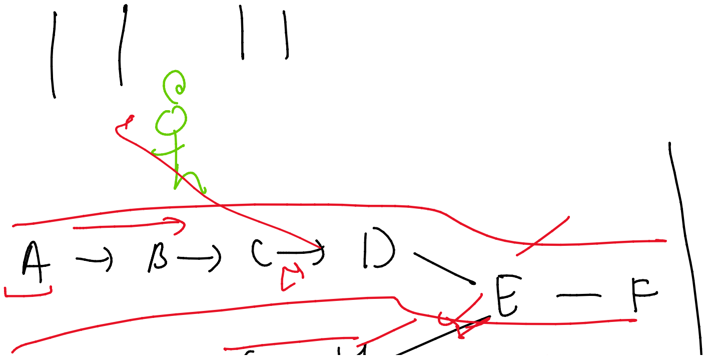
M3) Size

x

M4) →



O?

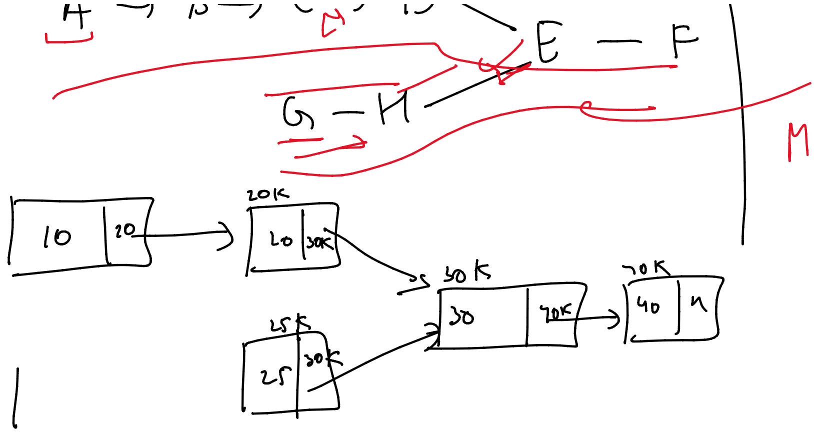


M1) Using stack

S → O(n+n)

T → O(n+n)

M4) Bollywood method



S → O(n+m)

T → O(n+n)

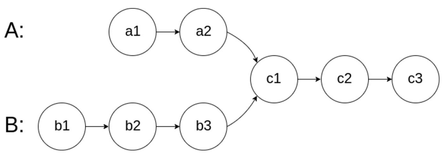
method

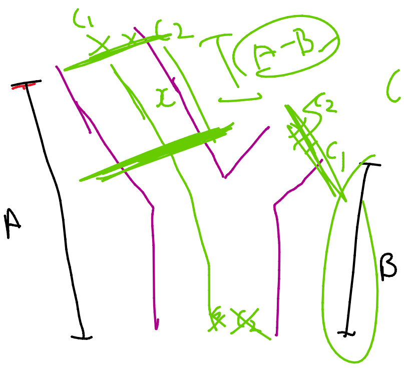
M2) Size ✓

Larges → diff

M3) O(n.m)

O(1)





C1 → A

C2 → B+x

B + x = A

x = A - B