2.10 - Linked List -1

Sunday, June 22, 2025 10:36 AM

public static void main (--) {

static → belongs class

void → return

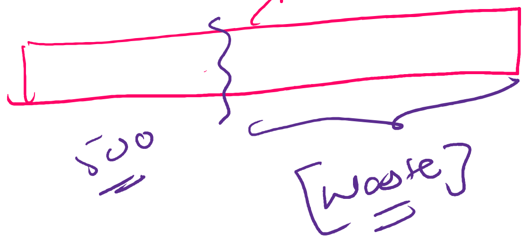
↓ object

}

linked list ↳ Data structure

→ arrays → 1000 size int

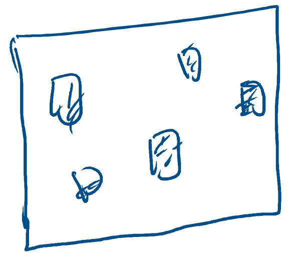
1000 x 4 = 4000 Bytes



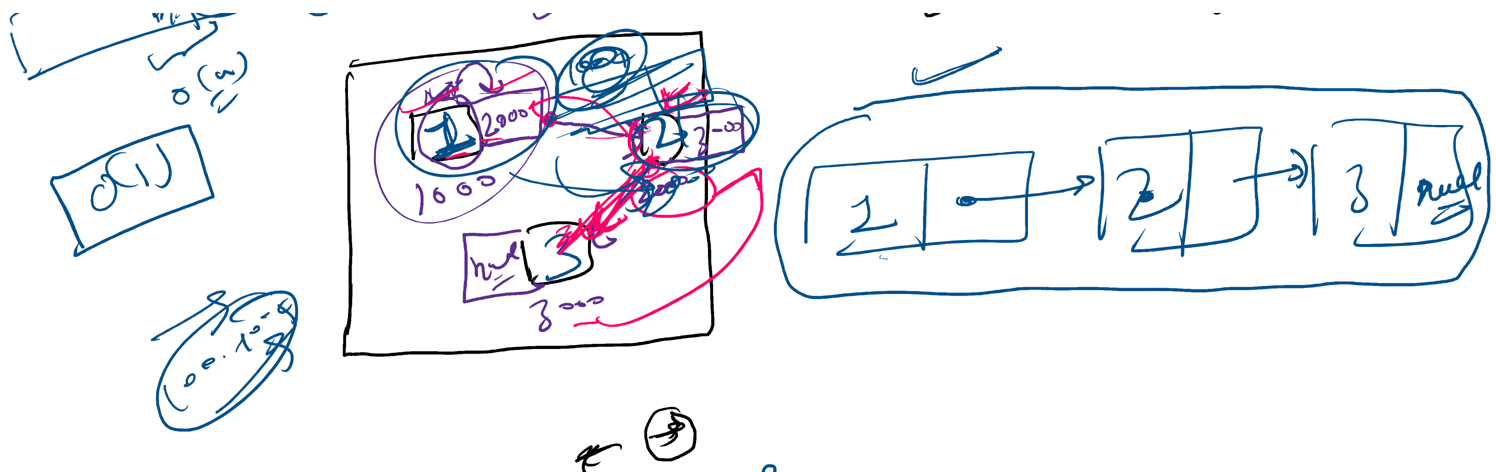
→ array list ↳ dynamic
↳ contiguous

500 x 4 = 2000 Bytes

linked list
↓
non contiguous memory locations

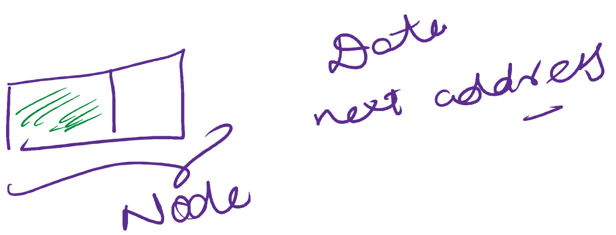


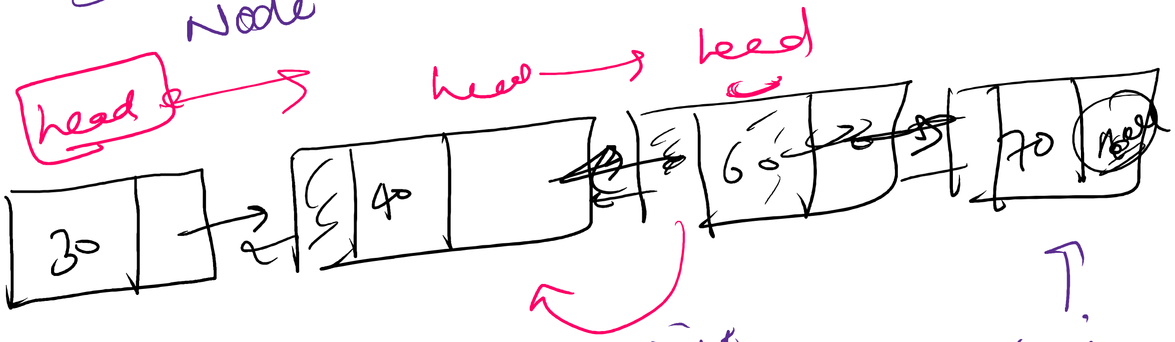
3 x 4 = 12 bytes



doubly

Node → building block of a linked list.

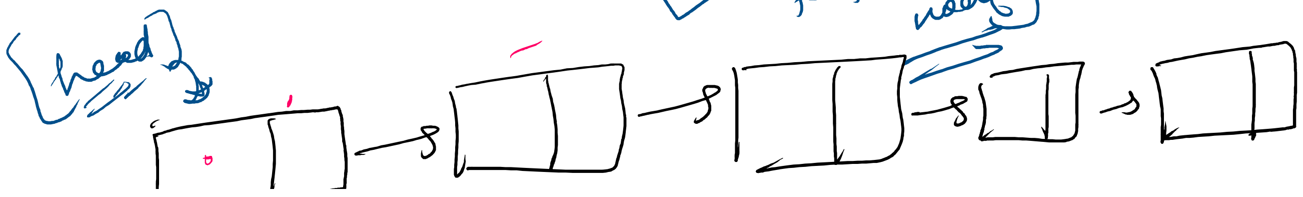


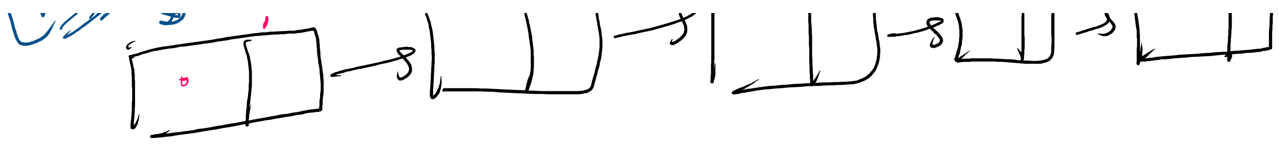


Singly linked list → move only in a single direction

Single LL collection of nodes

[data + address to the next node]

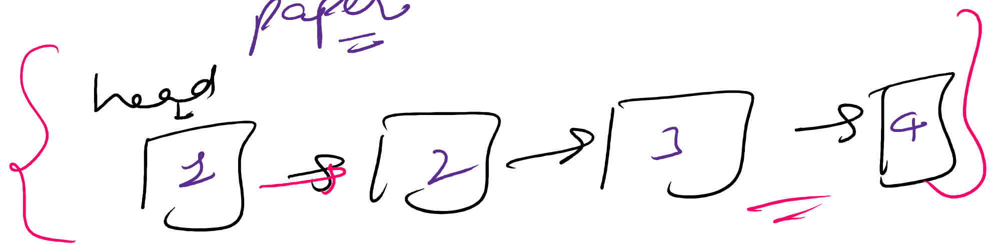




→ Linked list is dynamic in size.

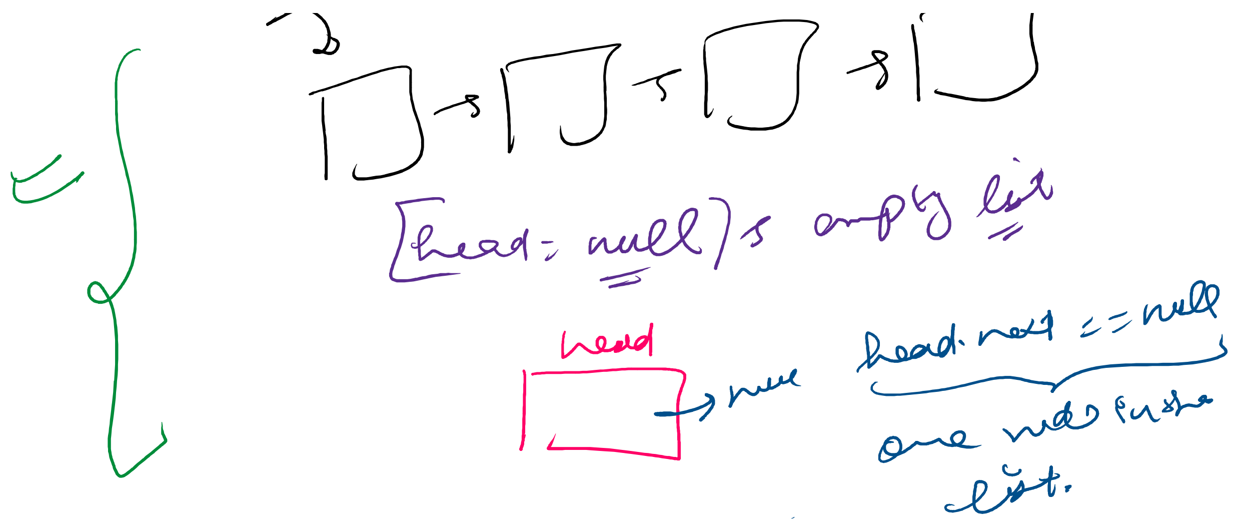
→ There is no concept of indexes in LL, because it is not continuous

* To solve the problems of LL, always draw a diagram on the paper



→ In LL, always think of two cases

1) empty list] ← mandatory
2) only one node in the list
specially in case of delete / remove



* While doing operations on a LL, we donot directly move head, instead we make another ref variable pointing to head.

Because we cannot afford to loose head as it is the only way to access our LL.

* Approach for solving LL questions

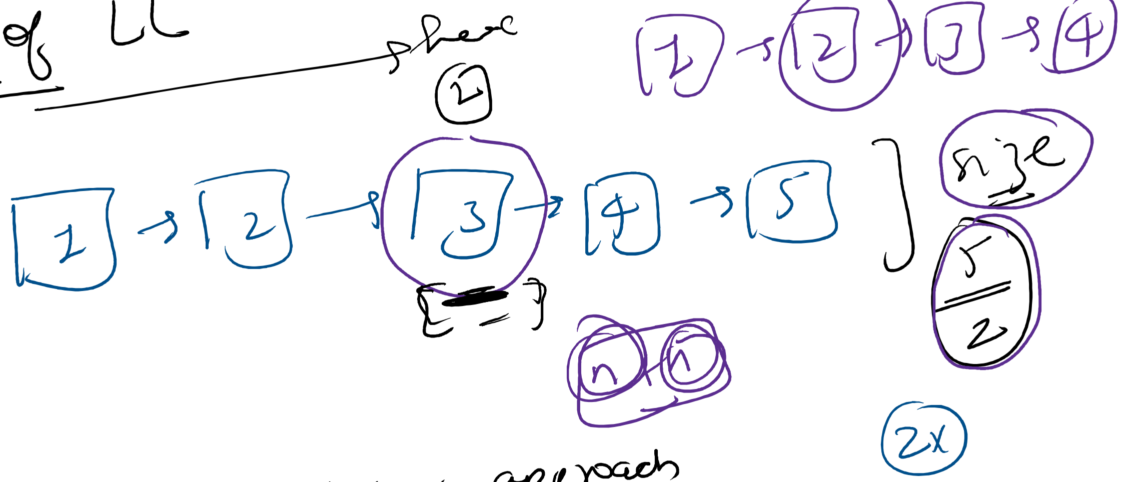
1) Think of a general approach.
Draw a LL of length 4/5/6 ..

2) Think of empty list wala base case

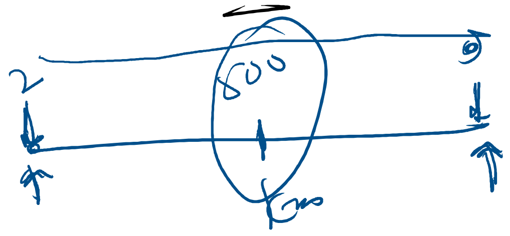
3) Think of single node in the LL

3) Think of single node in

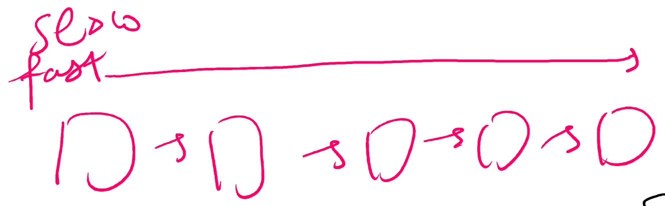
~~Amid~~ of LL

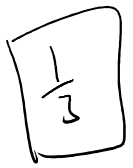


⊕ slow and fast pointer approach



1, 2





[slow = 1
fast = 3]