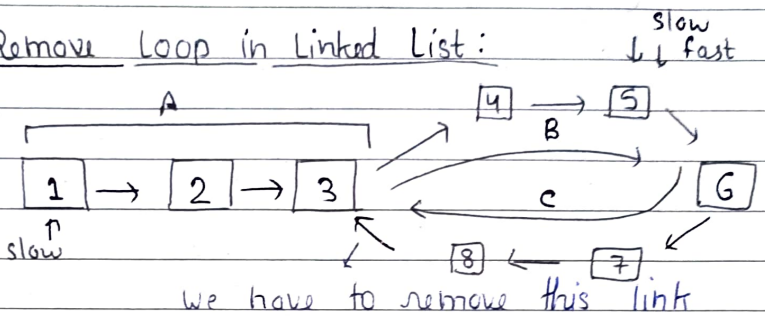
Date _____ Page _____

Day - 122

Linked List - 8

* **Remove Loop in Linked List:**

⇉



⇉ First we have to detect the loop.

⇉ Next step is to move slow to head.

⇉ Now, we will move both slow and fast by 1 step.

⇉ And when the slow and fast meet - is the starting of the loop.

⇉ Slow pointer: $A + (B+C) \times i + B$ (Distance covered)
Fast pointer: $A + (B+C) \times j + B$ (Distance covered)

⇉ Fast = 2 x slow distance

⇉ $A + (B+C) \times j + B = 2A + 2(B+C) \times i + 2B$

$A + B = (B+C) \times j - 2(B+C) \times i$

$A + B = (B+C) \times (j - 2i)$

$A + B = T \times (B+C)$

⇉ Note: $j > 2i$

⇉ $A + B = B + C$ if $T = 1$

⇉ $A + B = 2(B+C)$ if $T = 2$

$A = (B+C) + C$

Date _____ Page _____

⇒ So, A = C

Code

```
Node *slow = head , *fast = head;
while( fast && fast -> next){
        slow = slow -> next;
        fast = fast -> next -> next;
        if (slow== fast)
            break;
}
if( fast == NULL || fast -> next == NULL){
        return ;
}
slow = head;
while (slow != fast ){
    slow = slow -> next;
    fast = fast -> next;
}
while( slow -> next != fast){
        slow = slow -> next;
}
slow -> next = NULL;
```

Second Approach

⇒ First, we find the loop or check for the loop.

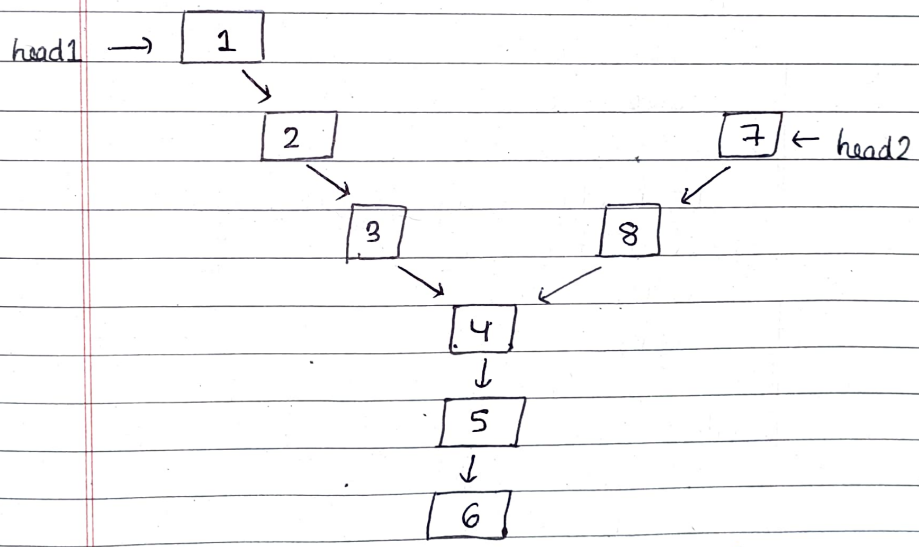
⇒ Then count the length of the loop.

Date _____ Page _____

⇒ After that put both the points count nodes apart.

⇒ Now, we will move both the pointers and then when they meet is the starting of the loop.

* Intersection point in Y shaped LL:



⇒ First, we will count the length of left side and then right side.

⇒ After that, we will check the difference and move the pointer of the bigger count to that difference.

⇒ Now, we will move both the pointers and when they will meet is the intersection point.

Date _____ Page _____

Code

⇒

```
Node *curr1 = head1, *curr2 = head2;
int count1 = 0, count2 = 0;
while(curr1) {
    count1++;
    curr1 = curr1 -> next;
}
while( curr2){
    count2++;
    curr2 = curr2 ->next;
}
curr1= head1 , curr2 = head2;
while( count1 > count2){
    curr1 = curr1 -> next;
    count1--;
}
while( count2 > count 1 ){
    curr2 = curr2 ->next;
    count2--;
}
while(curr1 == != curr2){
    curr1 = curr1 -> next;
    curr2 = curr2 -> next;
}; if( !curr1) return -1;
return curr1 -> data;
```