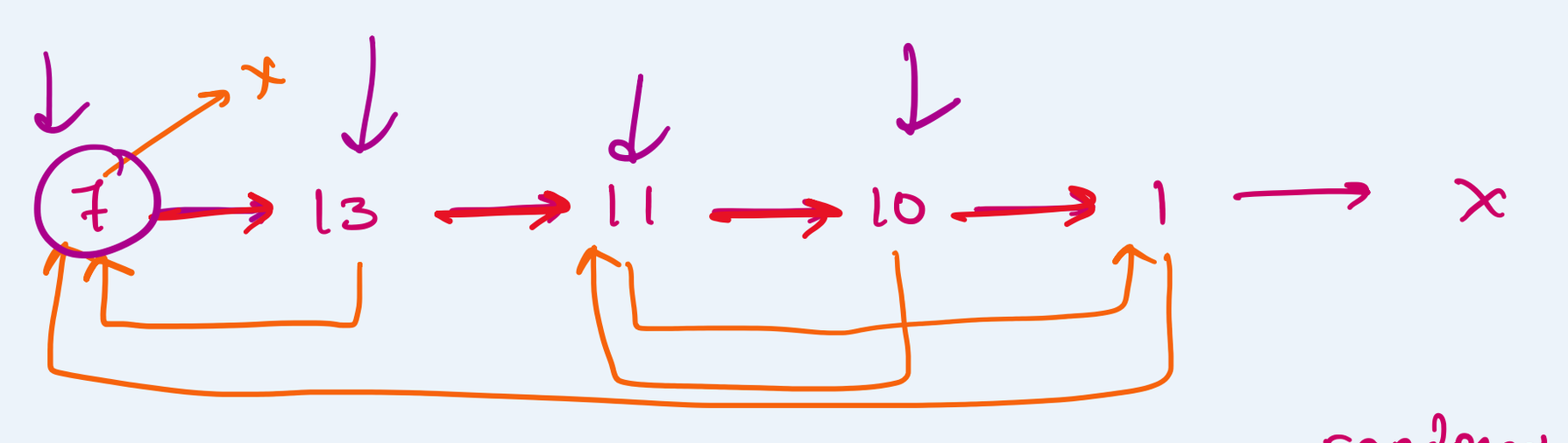
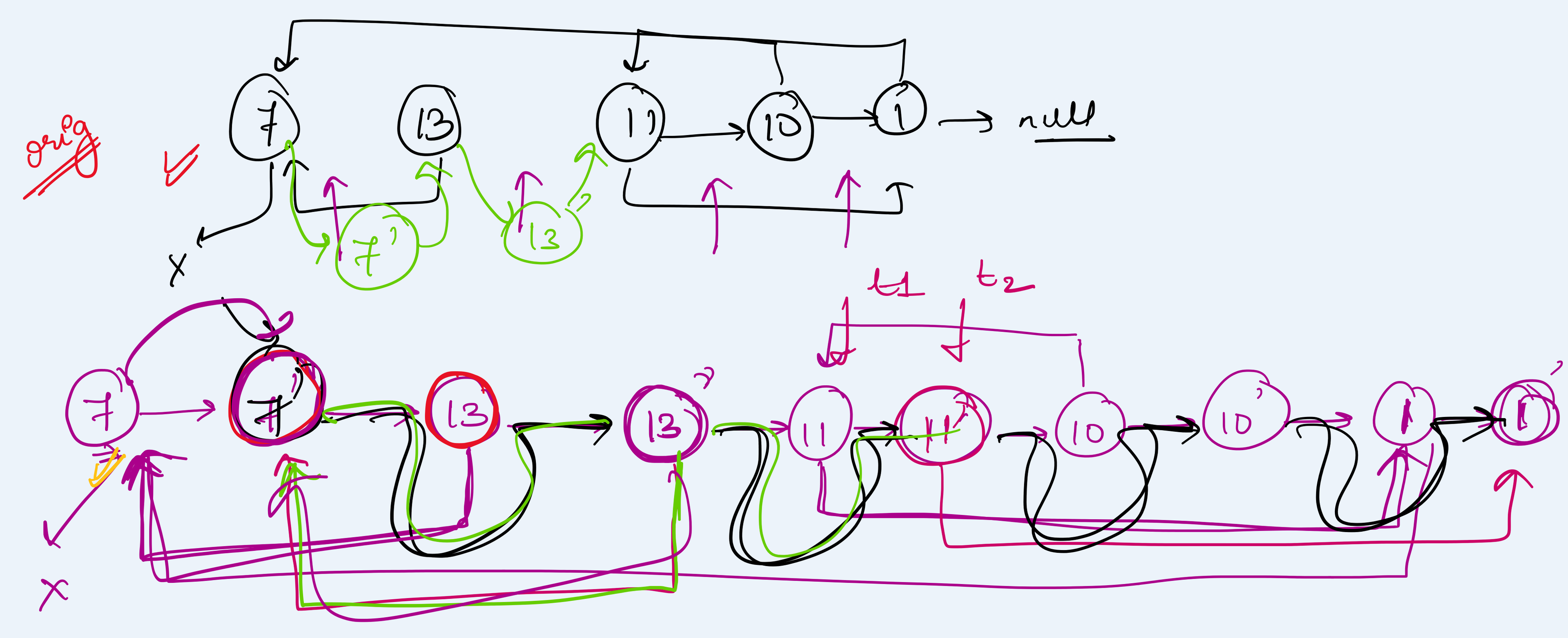
Thursday, 30 September 2021 9:08 PM

Q1 Given a linked list, create a copy (deep) of the linked list.

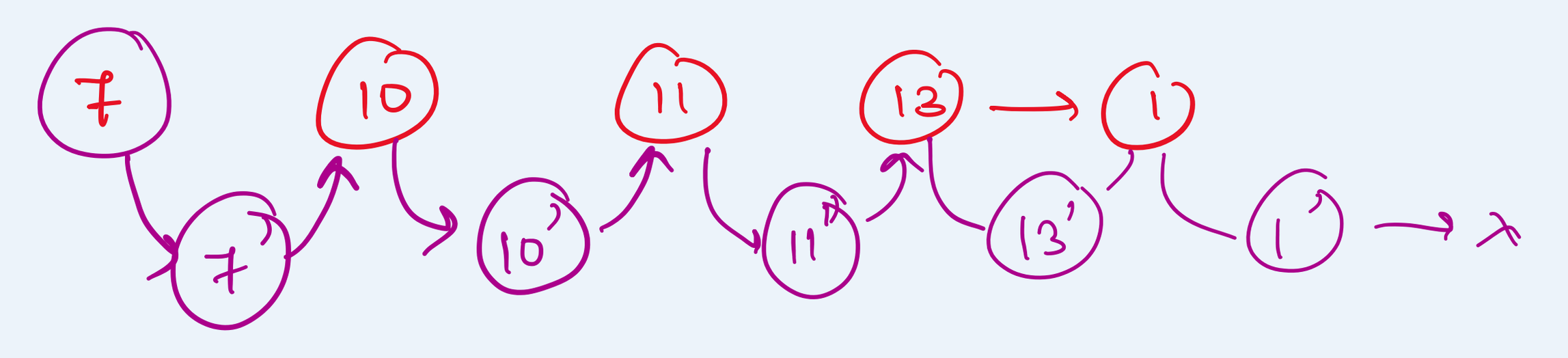


next	
7	13
13	11
11	10
10	1
1	null

	random
7	null
13	7
11	1
10	11
1	7



temp2. random = temp1. random . next;
(1). next → 1'



→ Random
→ next

Q Given an array of integers, containing n+1 integers, where each integer is of the range [1, n].
There is only one repeated number, find it.

Eg: arr [1, 3, 4, 2, 2] → n=4
output → 2

Eg: arr [1, 1, 2] → n = 2
output → 1

arr [1, 1, 1, 2, 3] → n = 4

All the integers in the array appear only once except for precisely one integer which can appear two or more times.

Approach 1: store the frequency of all elements in the array.
[1, 1, 1, 2, 3]

1	3
2	1
3	1

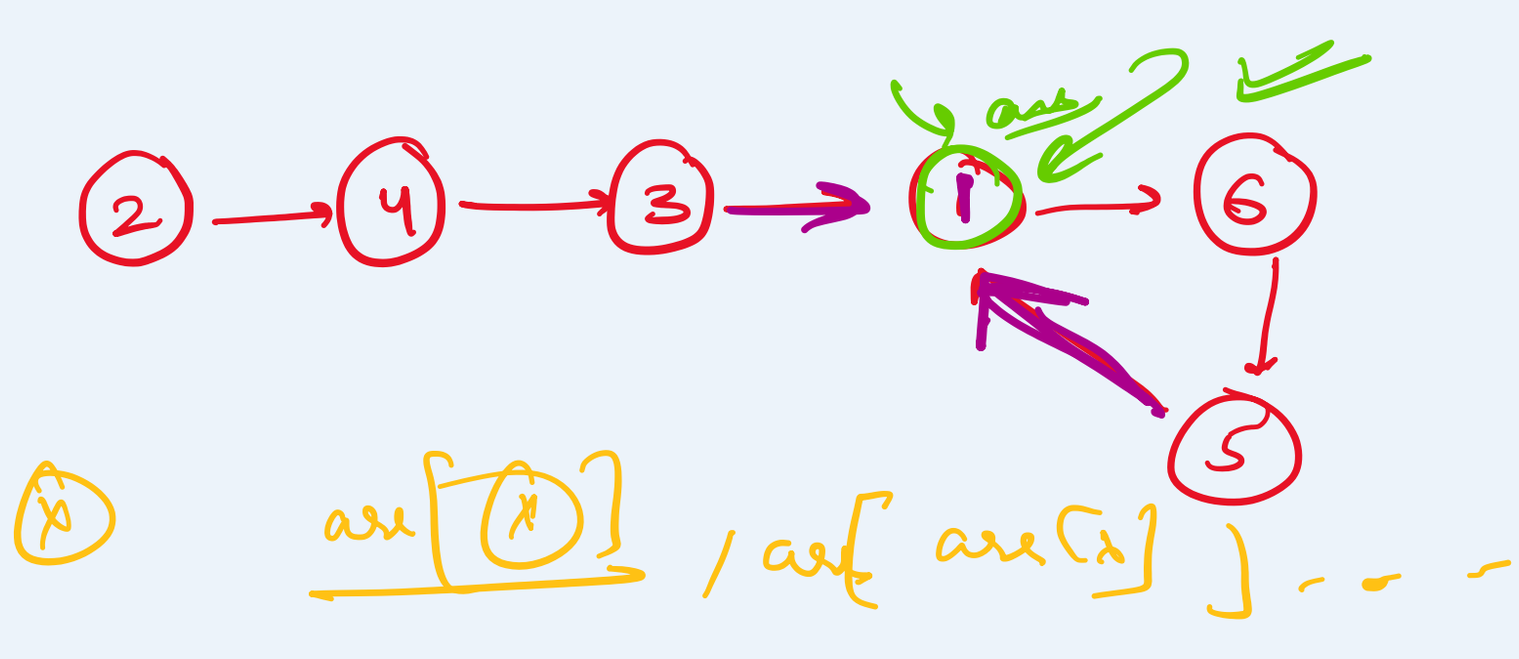
ans → 1

TC → O(n)
SC → O(n)

Approach 2: Cycle detection problem

Given a LL, return the node where the cycle begins.

x → arr[x], arr[arr[x]] ...



x arr[x] / arr[arr[x]] ....

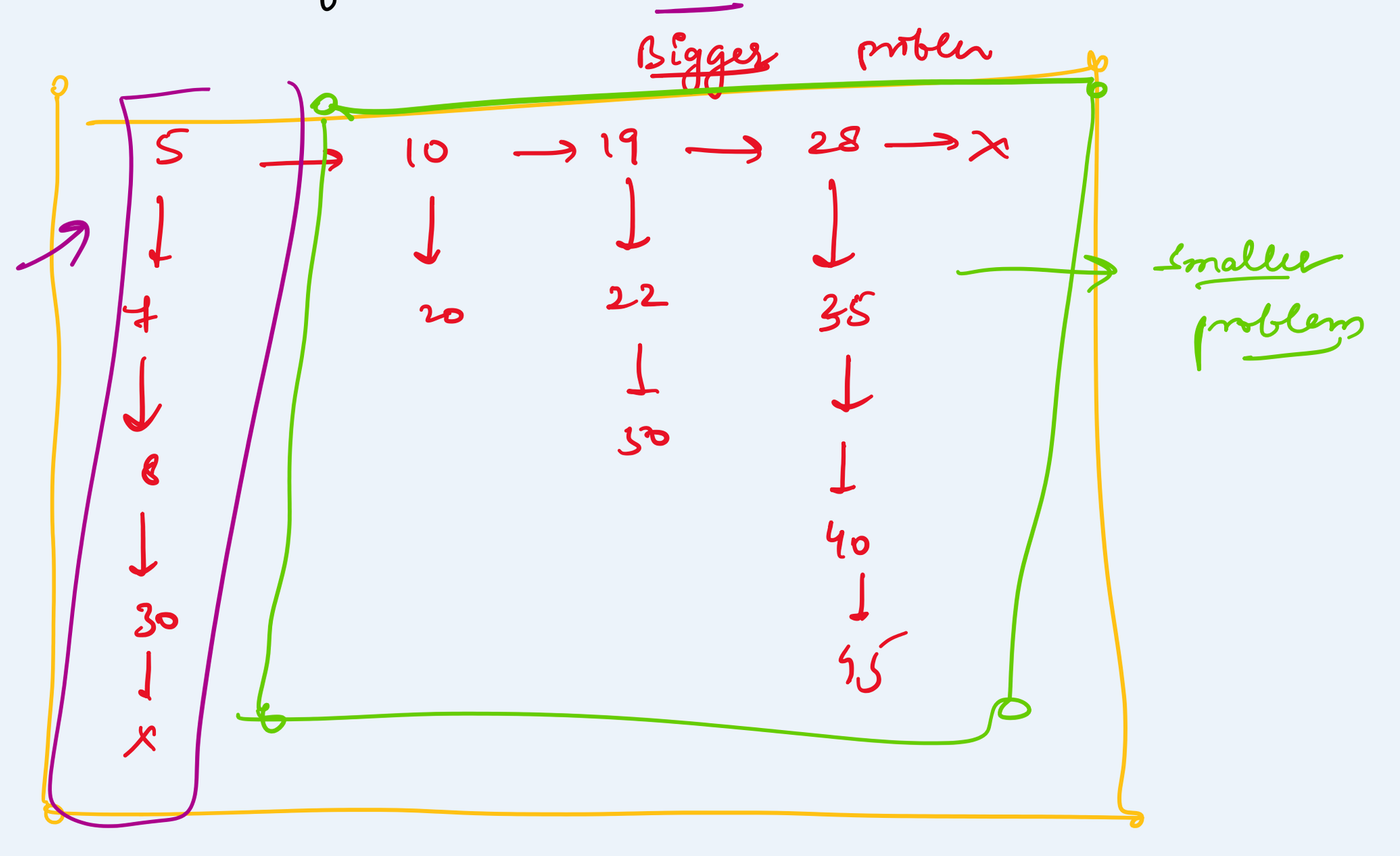
```
find duplicate ( int[] nums) {
    int slow = nums[0];
    int fast = nums[0];
    do {
        slow = nums[slow];
        fast = nums[nums[slow]];
    while ( slow != fast )
        slow = nums[0];
        while ( slow != fast) {
            slow = nums[slow];
            fast = nums [fast];
        }
    return slow;
}
```

TC → O(n)
SC → O(1)

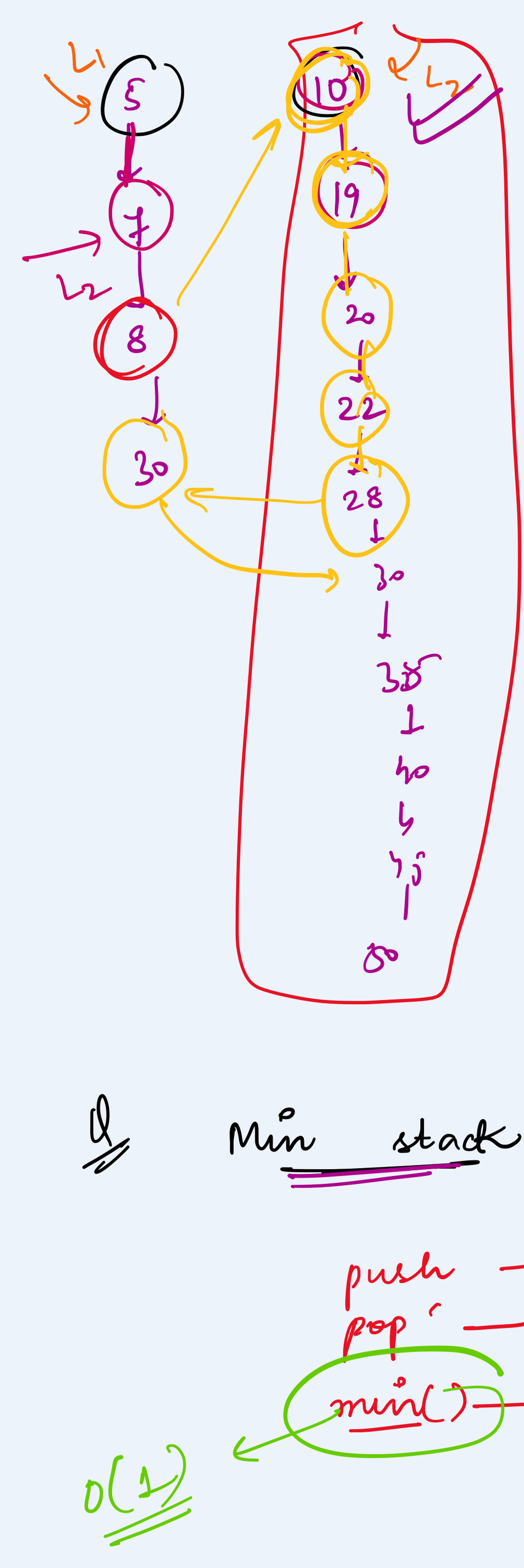
Q Given a LL, (all are sorted)

```
class Node {
    int data;
    Node right;
    Node down;
}
```

flatten the LL.



output 5 → 7 → 8 → 10 → 19 → 20 → 22 → 28 → 30 → 35 → 40 → 45 → 50 → x

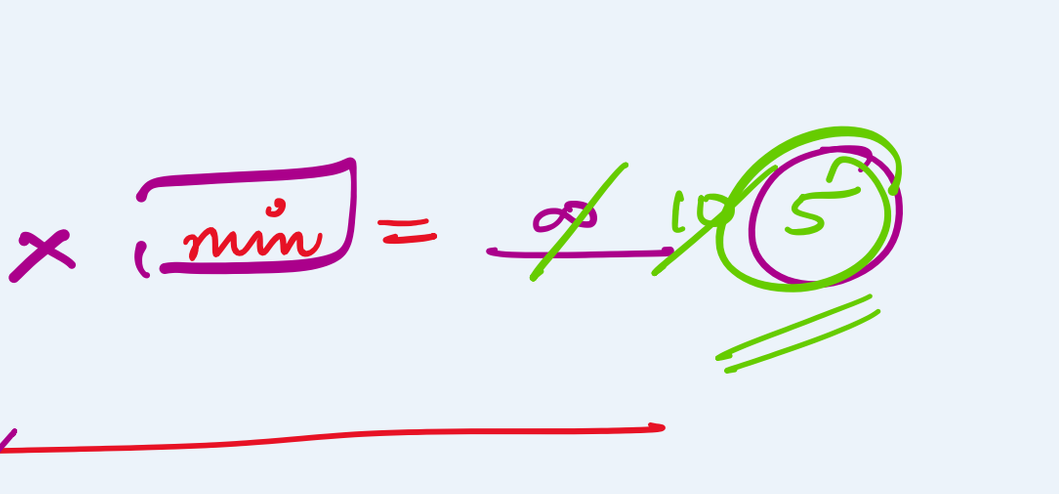


L1 and L2 are sorted
Merge them

Flatten(head);
if (head == null) return;
Node t = Flatten(head. next);
// Merge

Q Min stack

push →
pop →
min() → gets the min ele
O(1)



push (10);
push (20);
push (5)
push (35);
getmin() → 5
pop();
pop();
getmin() → 10

push (10);
push (5);
push(20);
getmin → min → 5
pop(); → 20
min → 5
pop ();