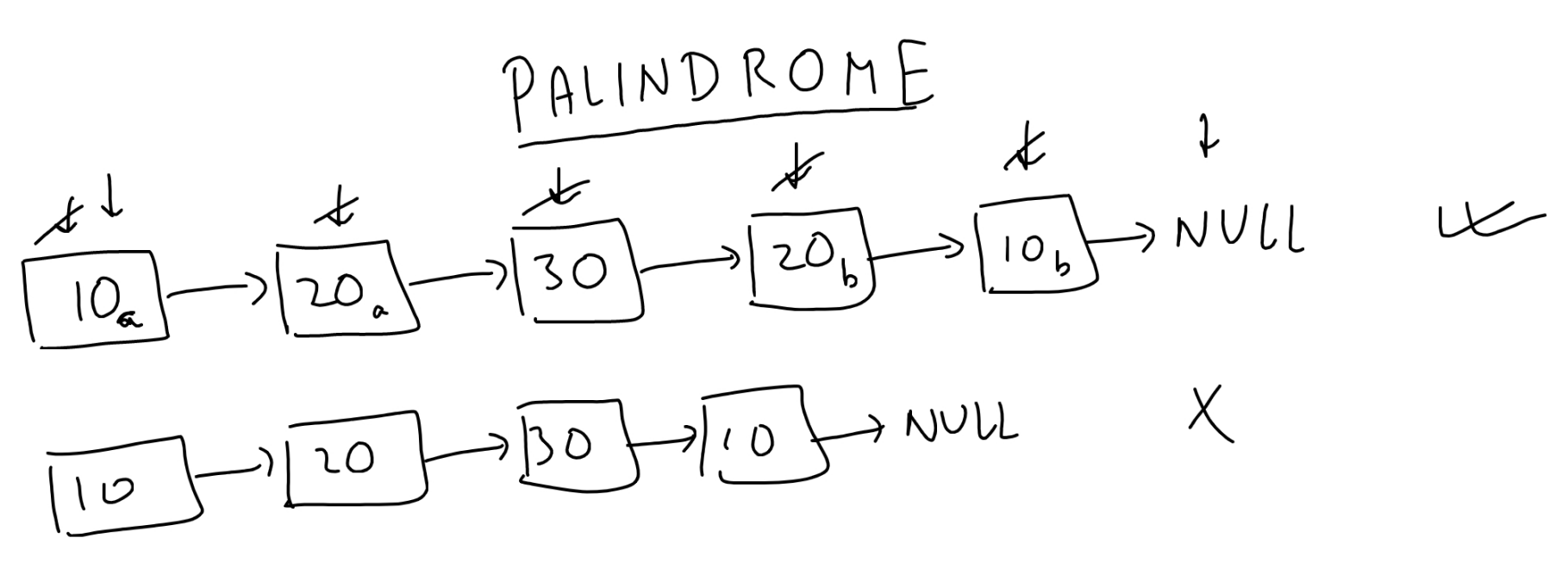
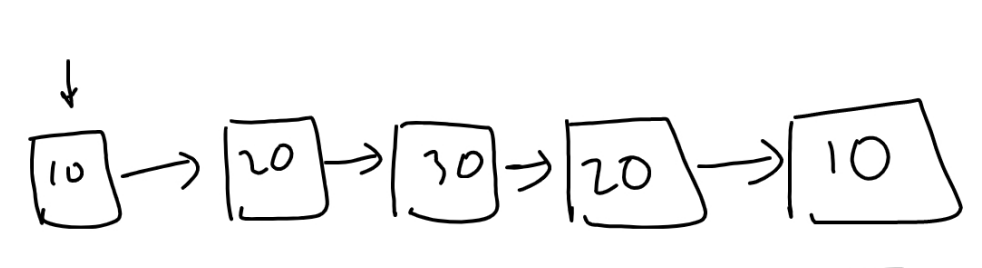
PALINDROME

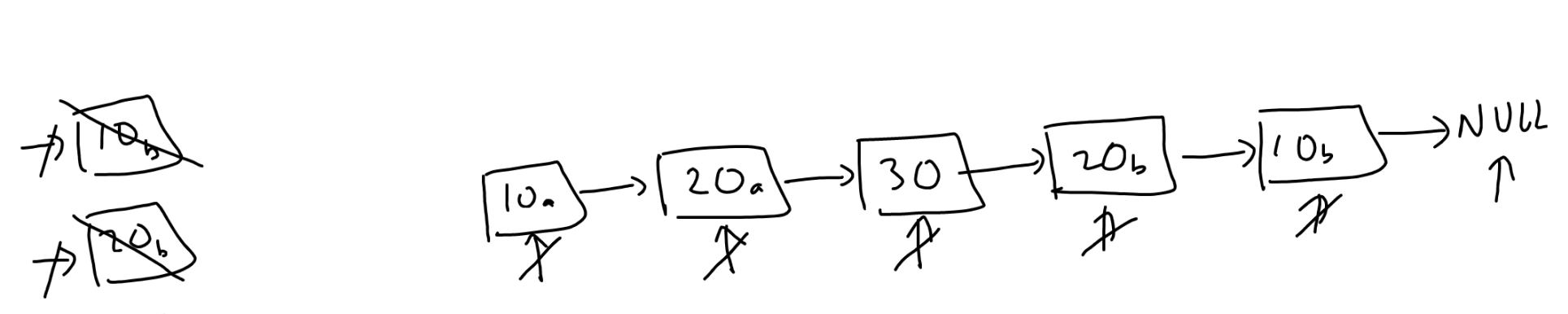

10a
20a
30
20b
10b
NULL
10
20
30
10
NULL
X

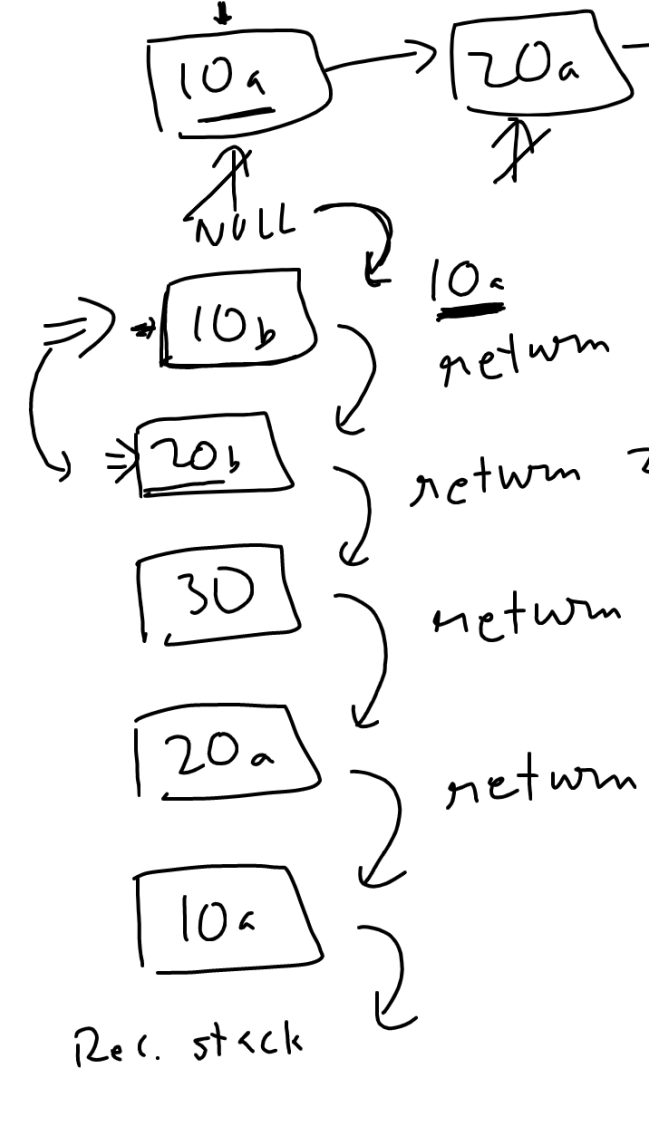

10
20
30
20
10

Create duplicate reverse:

TC: O(n) (2 traversals)

Aux.space: O(n)


10a
20a
30
20b
10b
NULL


head
10a
20a
30
20b
10b
NULL
Node*
10b
20b
30
20a
10a
Rec. stack

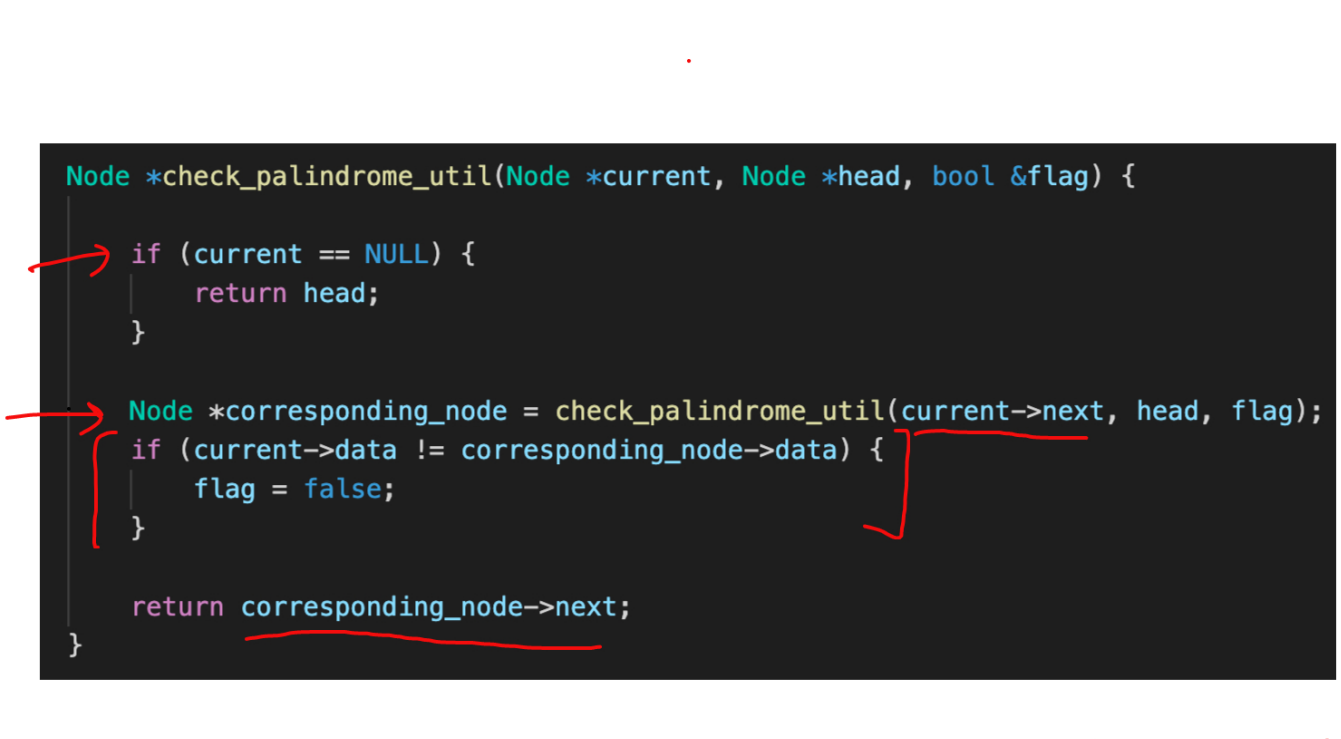
10a

return 10a → next = 20a

return 20a → next = 30

return 30 → next = 20b

return 20b → next = 10b

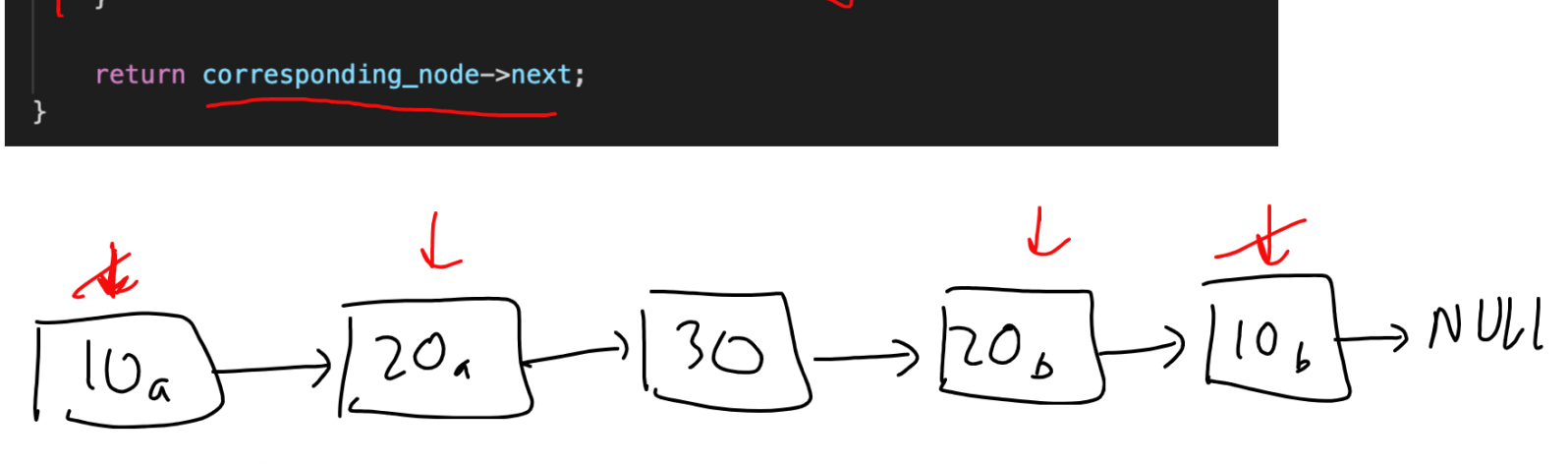


```
Node *check_palindrome_util(Node *current, Node *head, bool &flag) {

    if (current == NULL) {
        return head;
    }

    Node *corresponding_node = check_palindrome_util(current->next, head, flag);
    if (current->data != corresponding_node->data) {
        flag = false;
    }

    return corresponding_node->next;
}
```


10a
20a
30
20b
10b
NULL

current: NULL → corr. node = 10a

current = 10b → corr. node = 20a

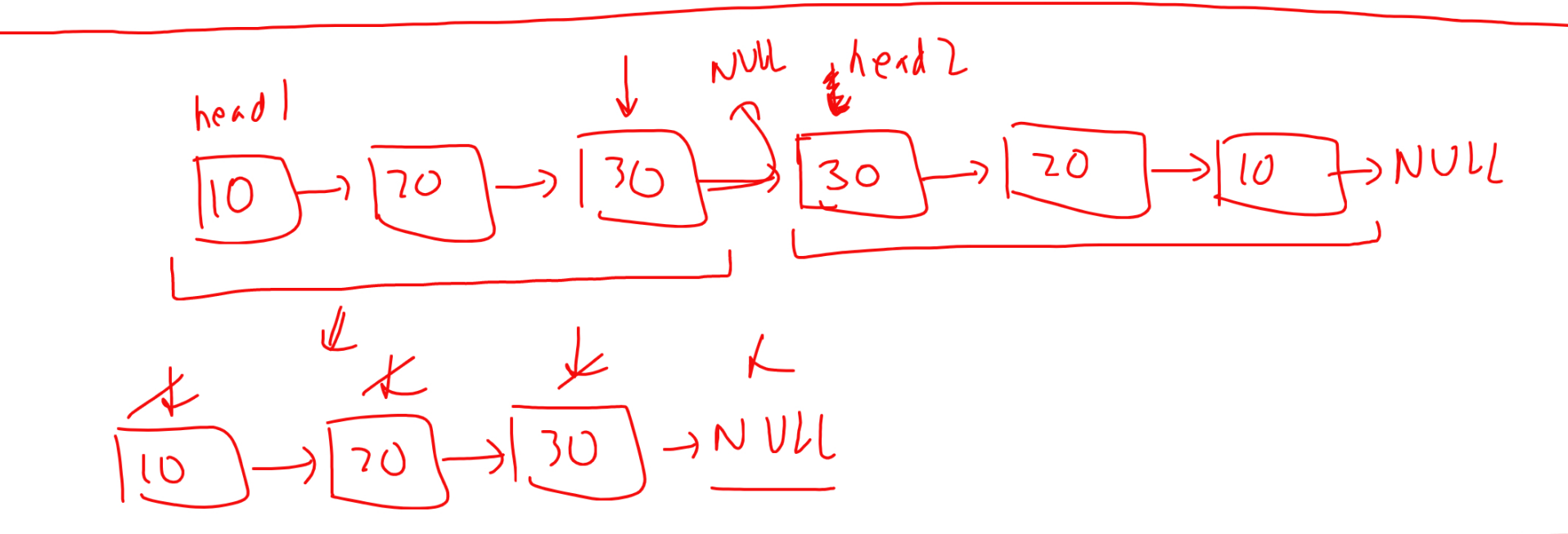
current = 20b → corr. node = 30

current = 30

→ corr. node = 20b

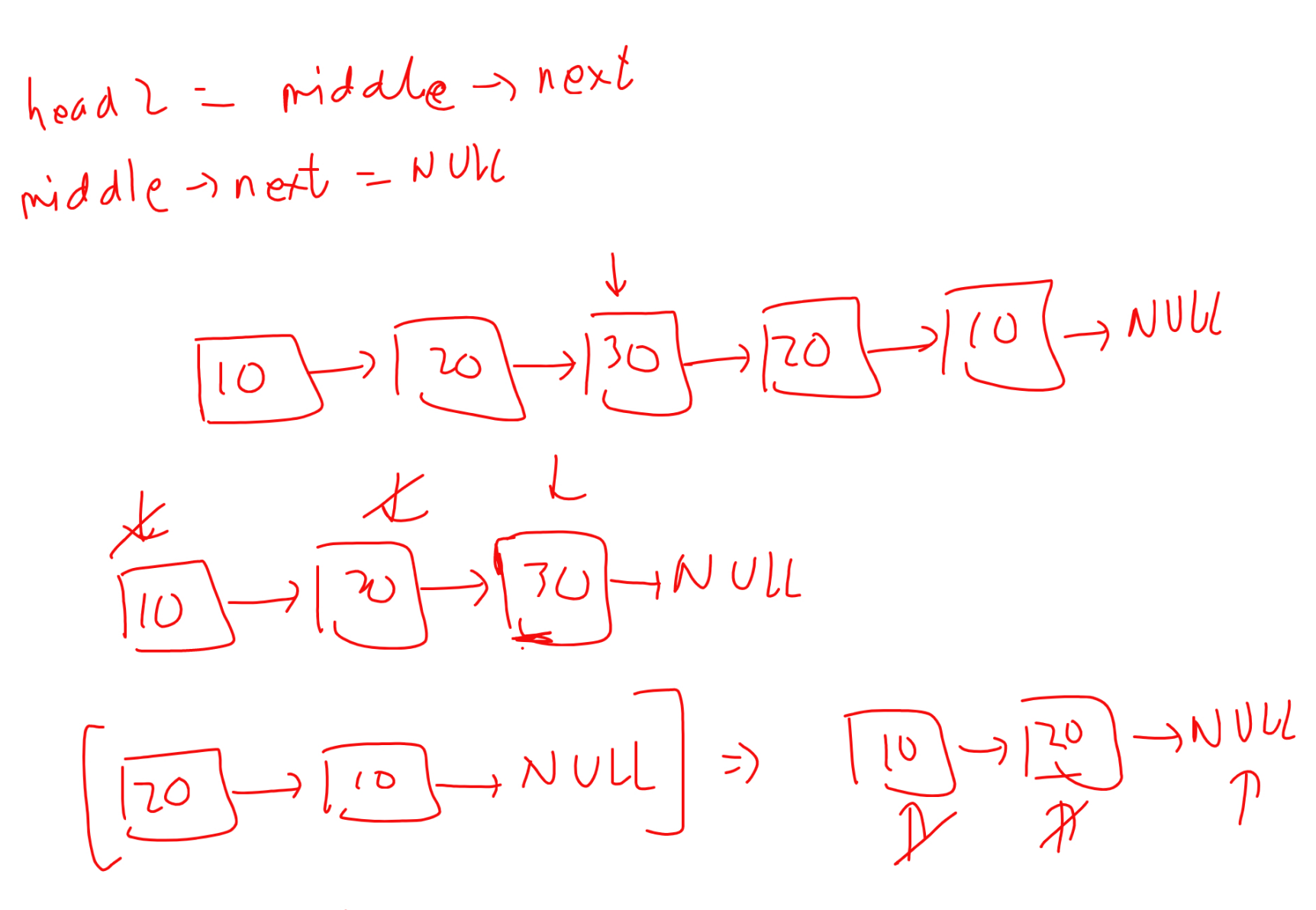
current : 20a → corr. node = 10b

current = 10a

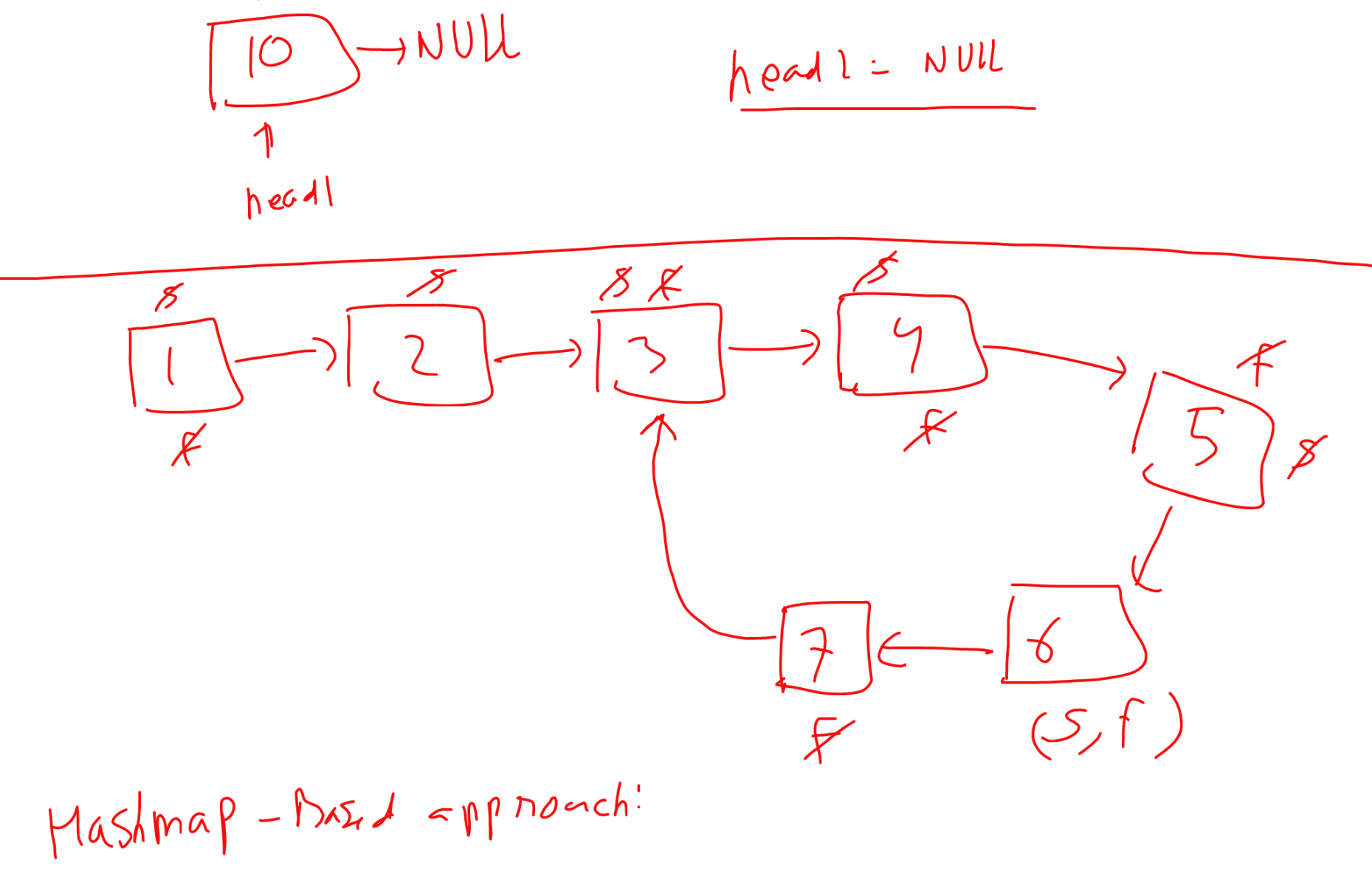

head1
10
20
30
NULL
head2
30
20
10
NULL
10
20
30
NULL
30
20
10
NULL
prev
10
20
30
NULL

head2 = middle → next

middle → next = NULL


10
20
30
20
10
NULL
10
20
30
NULL
20
10
NULL
10
20
NULL
middle
10
NULL
head1

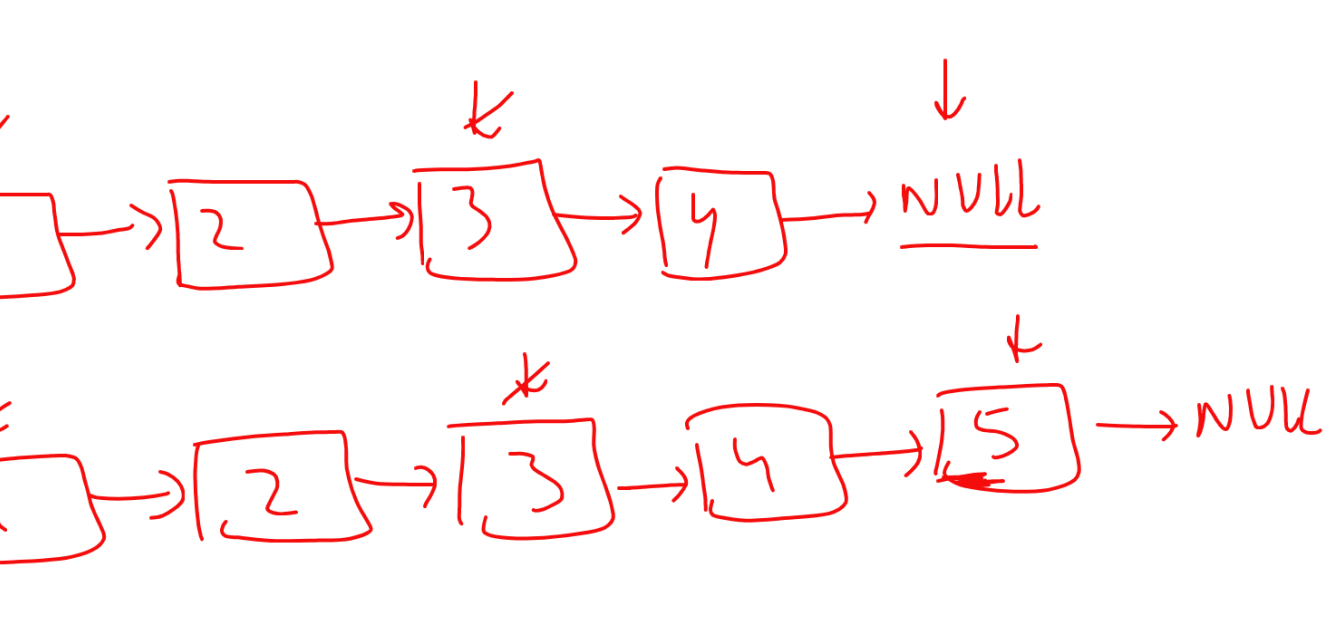
head1 = NULL

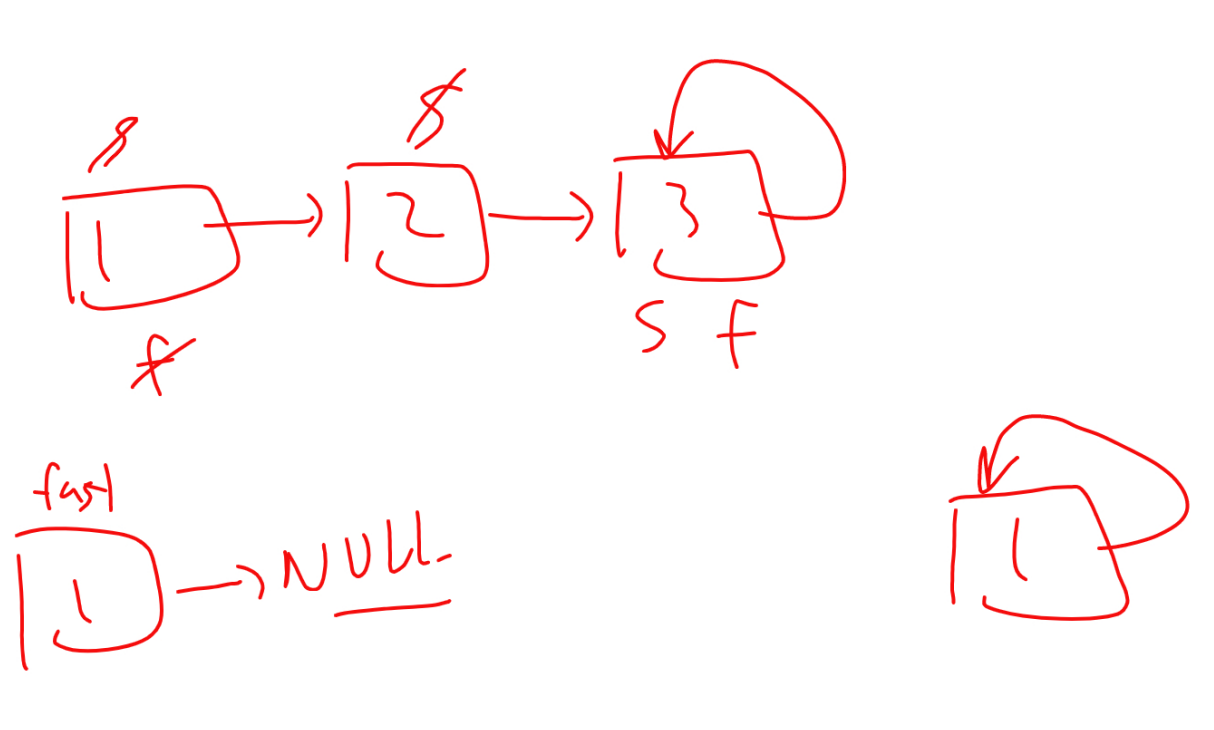


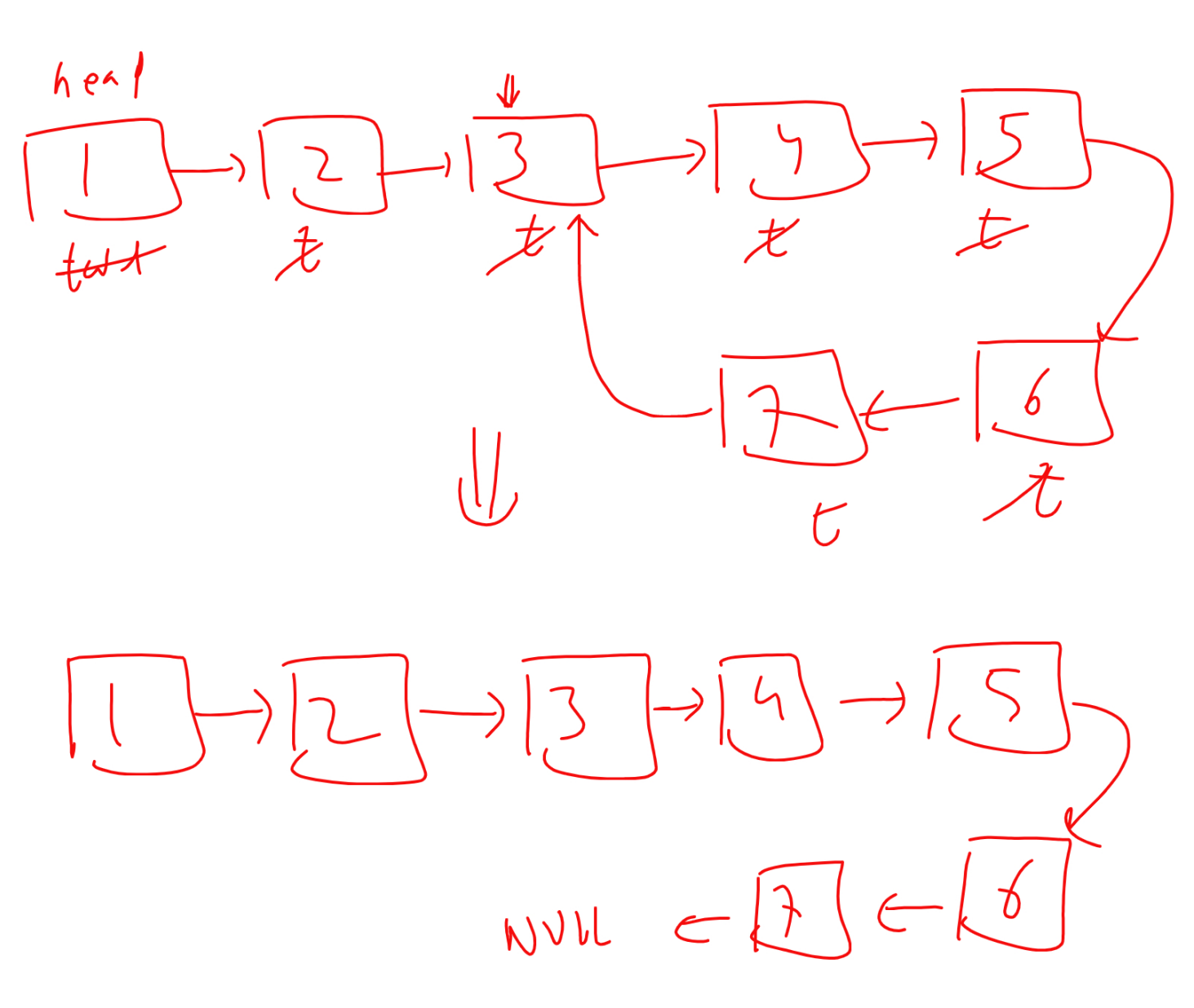
#1 Hashmap-Based approach:

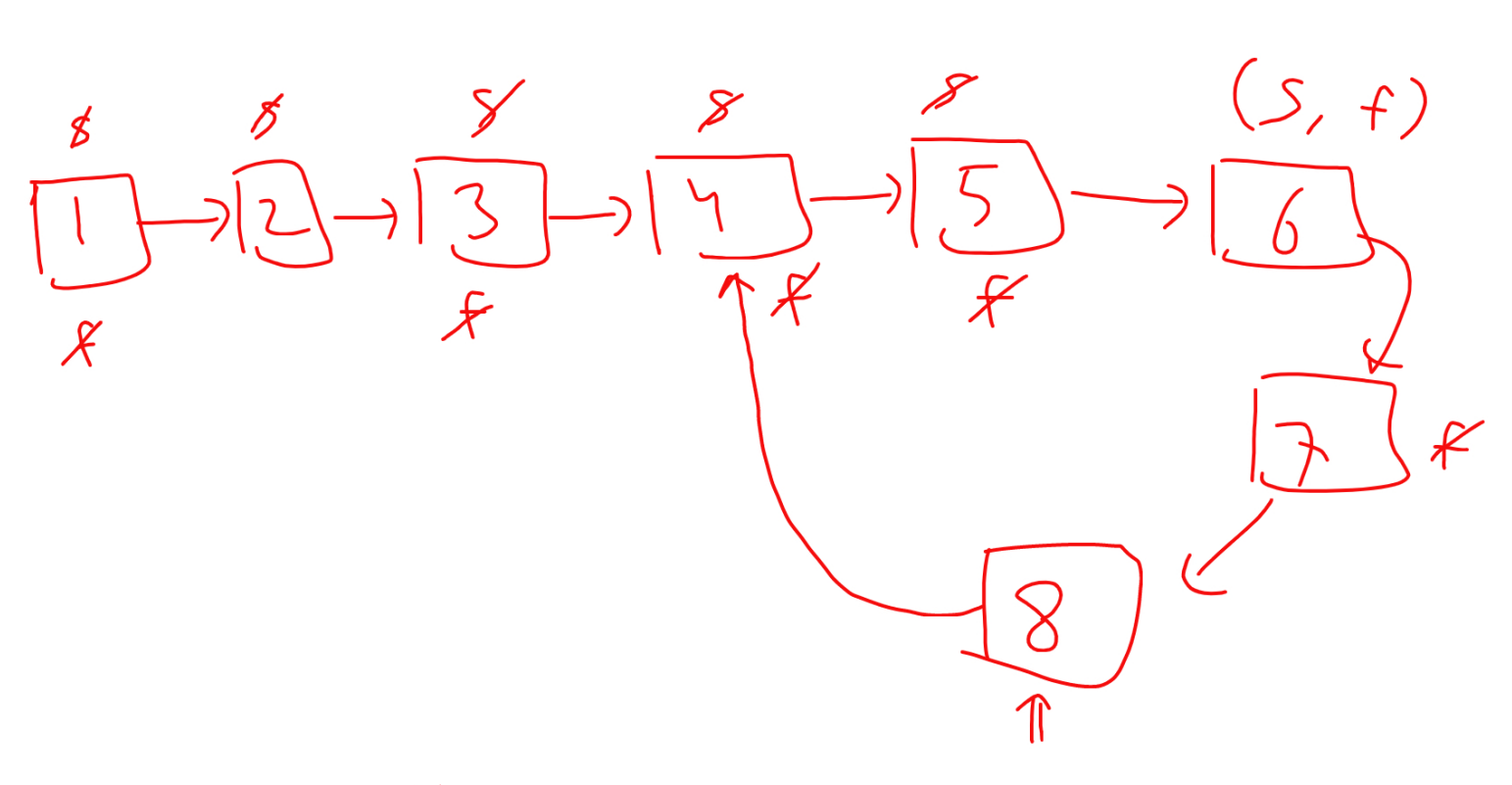
TC: O(n)

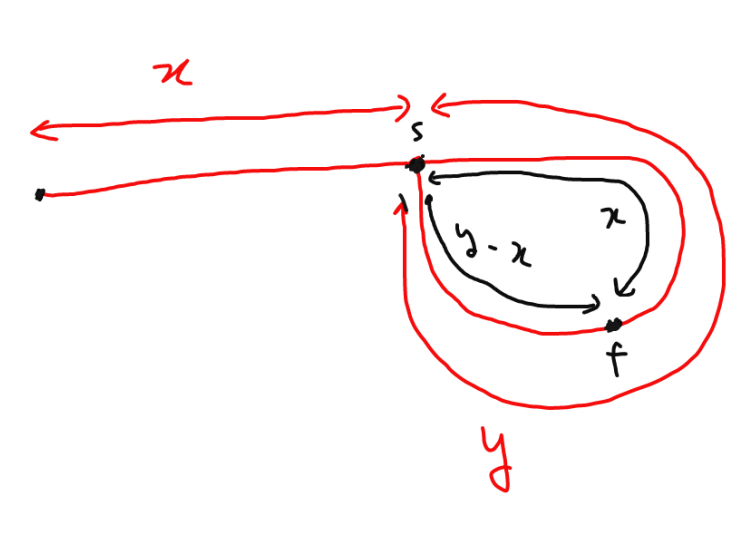
Aux.space: O(n)


1
2
3
4
NULL
1
2
3
4
5
NULL


1
2
3
s f
fast
1
NULL
1


head
1
2
3
4
5
6
7
NULL


1
2
3
4
5
6
7
8
(s, f)



[Speed of slow ptr = 1
Speed of fast ptr = 2]

→ when slow reaches the start of the loop, fast will be x-steps inside the loop.

→ At this time, fast is (y − x) steps behind slow.

→

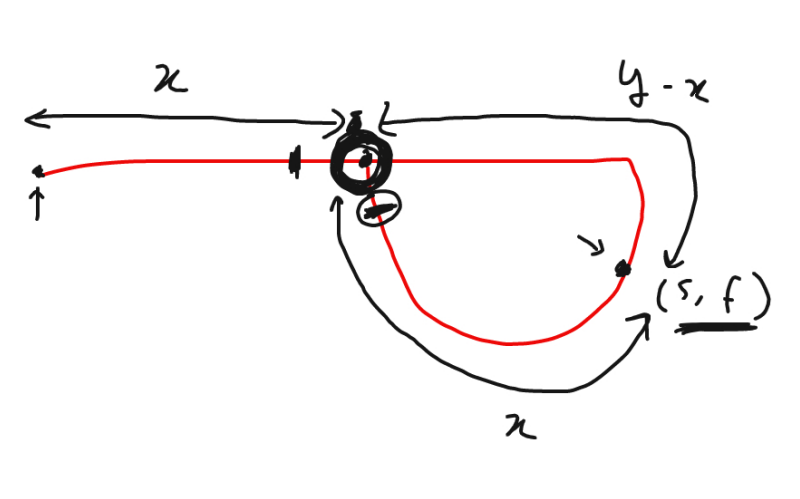
Speed = $\frac{\text{Distance}}{\text{Time}}$ ⇒ Time = $\frac{\text{Distance}}{\text{Speed}}$

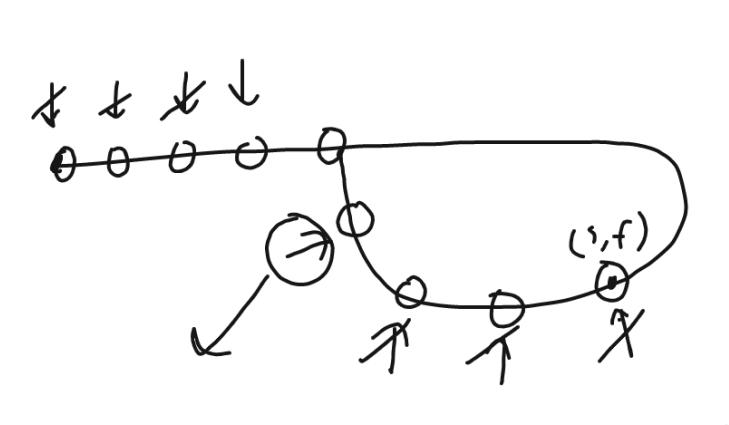
Relative distance b/w fast & slow = (y − x)

Relative speed = (2 − 1)

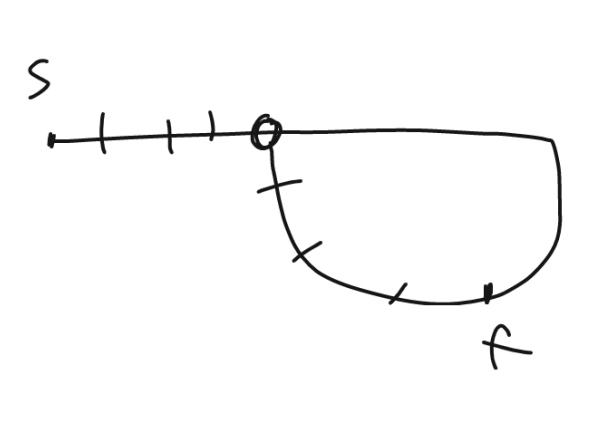
∴ Time taken by fast ptr to catch the slow ptr:

$$\Rightarrow \frac{(y-x)}{(2-1)} = \underline{(y-x)}$$

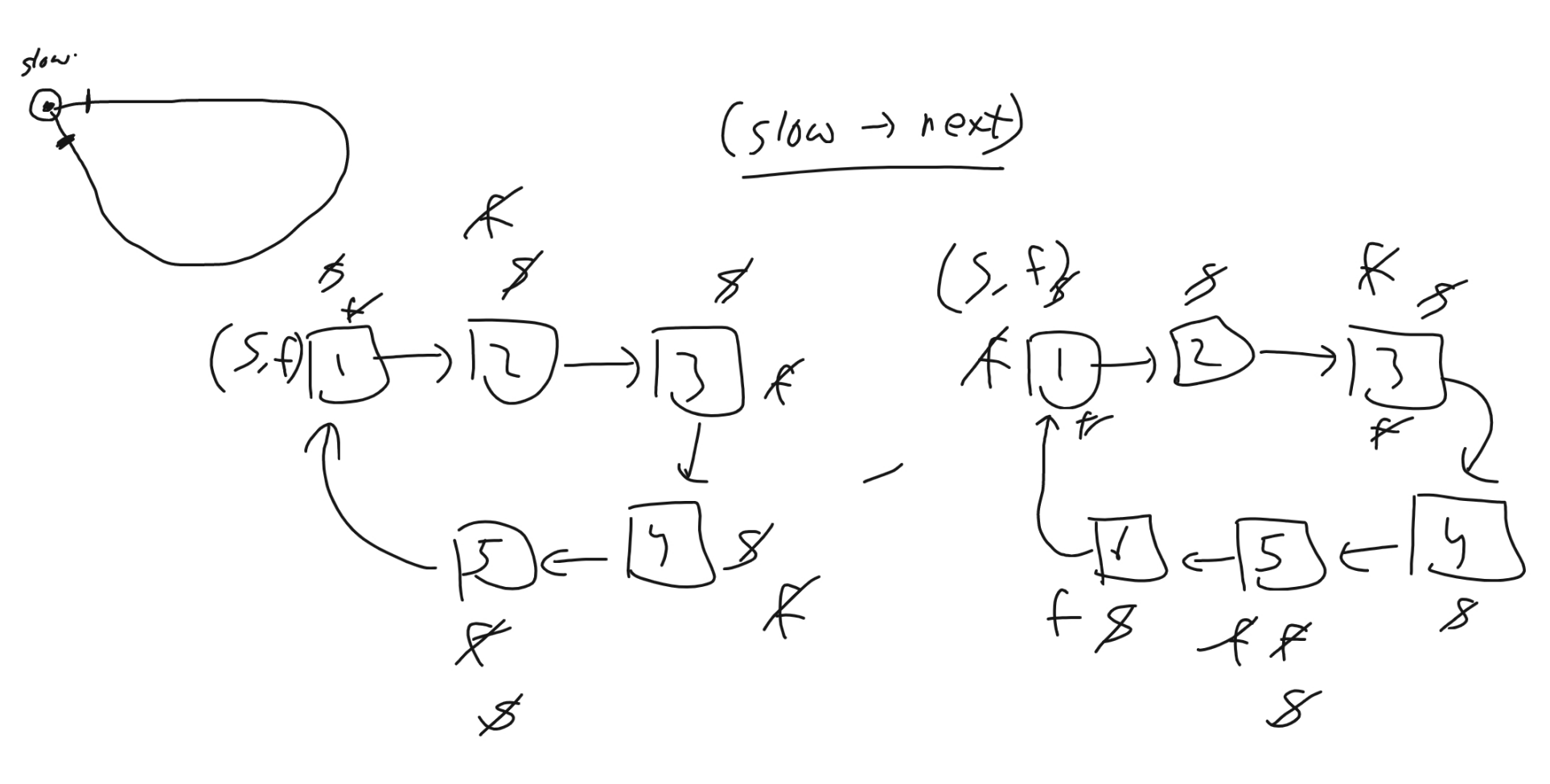




(mark the next of this ptr as NULL.)



(slow → next)


slow
(s, f)
1
2
3
4
5
(s, f)
1
2
3
4
5
1