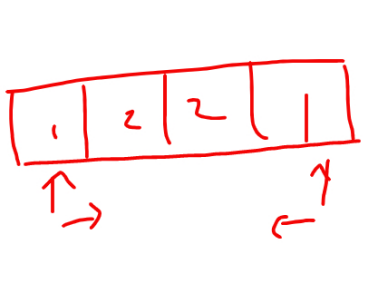
1 → 2 ⋮ 2 → 1

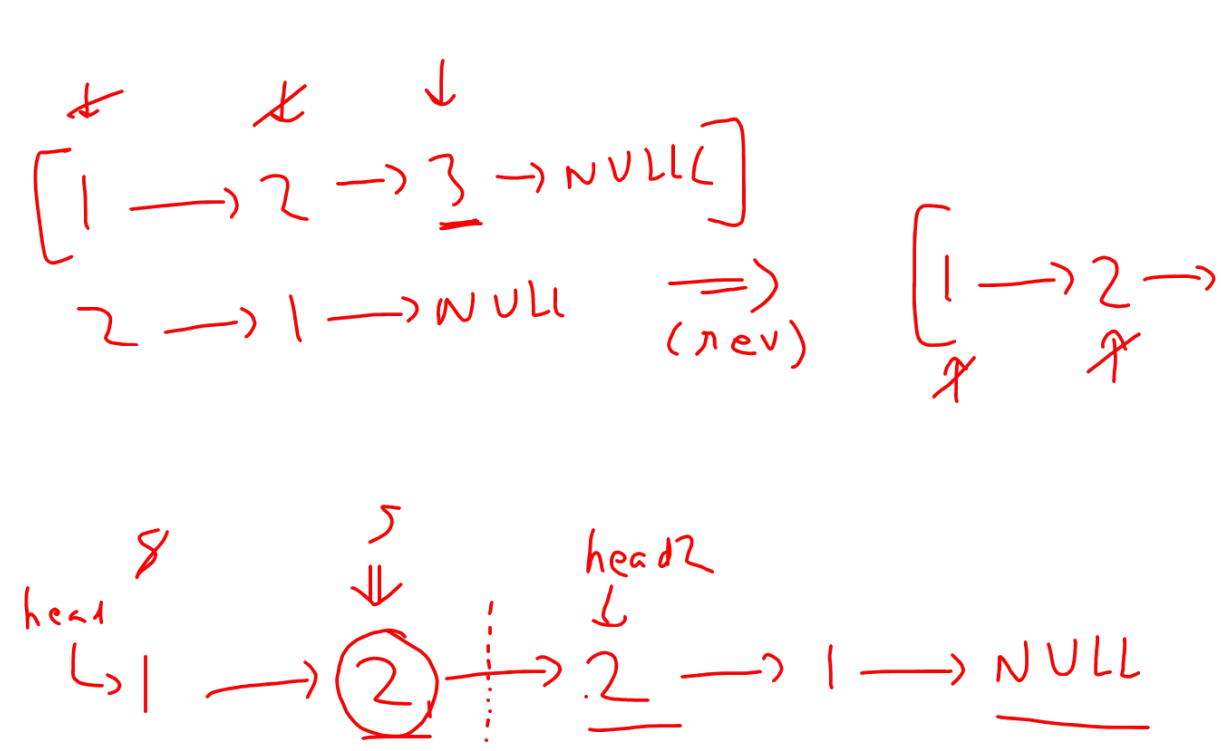


O(n) aux space in rec. space.

[1 → 2 → NULL
2 → 1 → NULL] ⇒ (revs) [1 → 2 → NULL]

1 → 2 → 3 ⋮ 2 → 1 → NULL

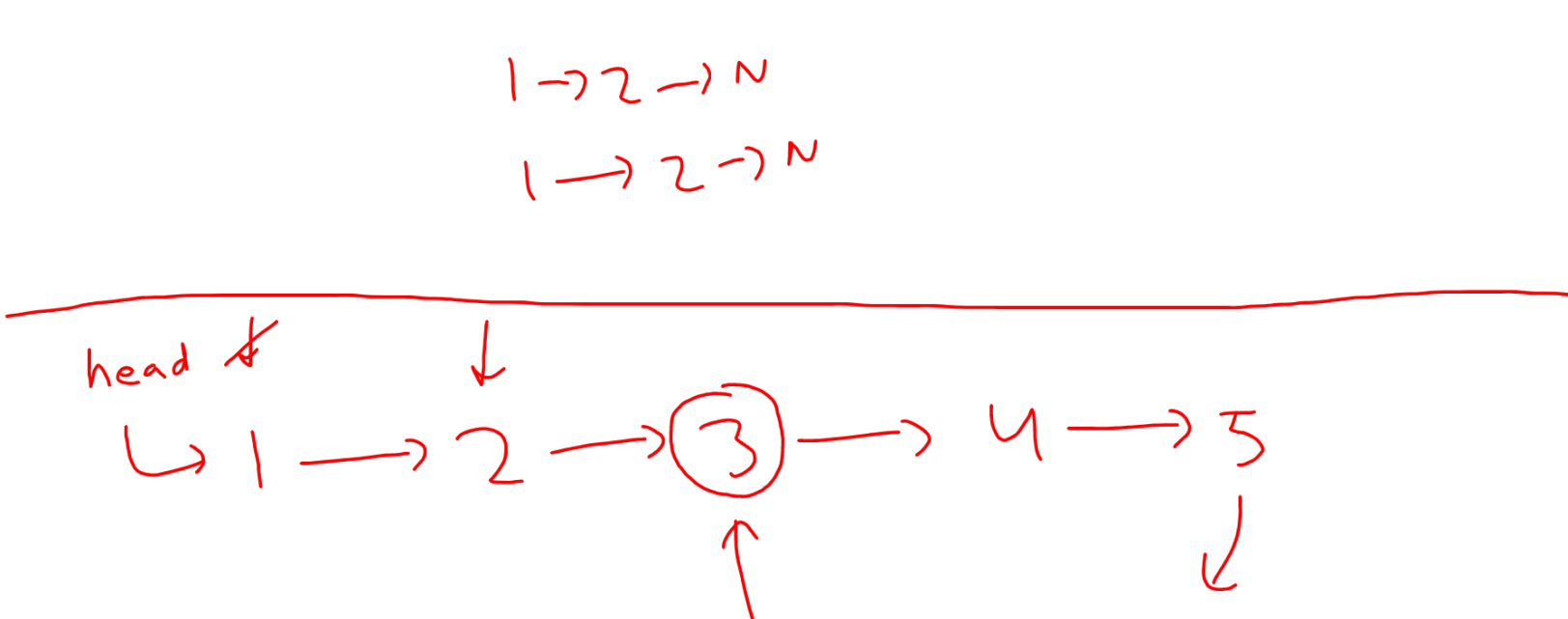
[1 → 2 → 3 → NULL
2 → 1 → NULL] ⇒ (rev) [1 → 2 → NULL]


s
s
head
head2
1
2
2
1
NULL
f
f

1 → 2 → 2 → N
1 → N] X

head
1 → 2 → N
2 → 1 → N] ✓
head2

1 → 2 → N
1 → 2 → N


head
1
2
3
4
5
6
7

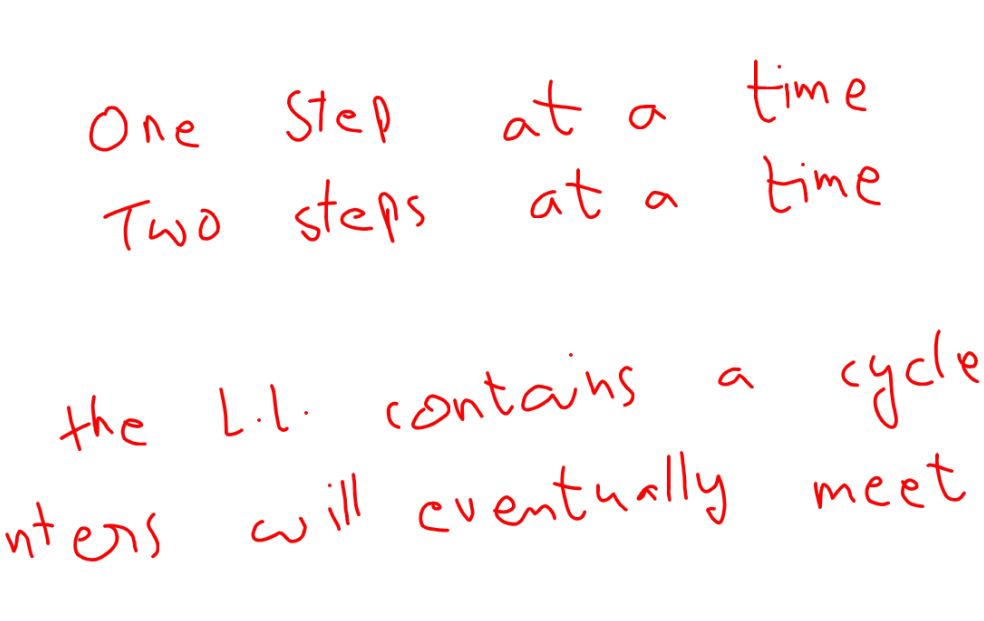
1 → 2 → 3 → 4 → 5 → 6 → 7 → NULL
[3x]

[address → Boolean]
1x → True
2x → True
[3x → T]
4x → T
5x → T
6x → T

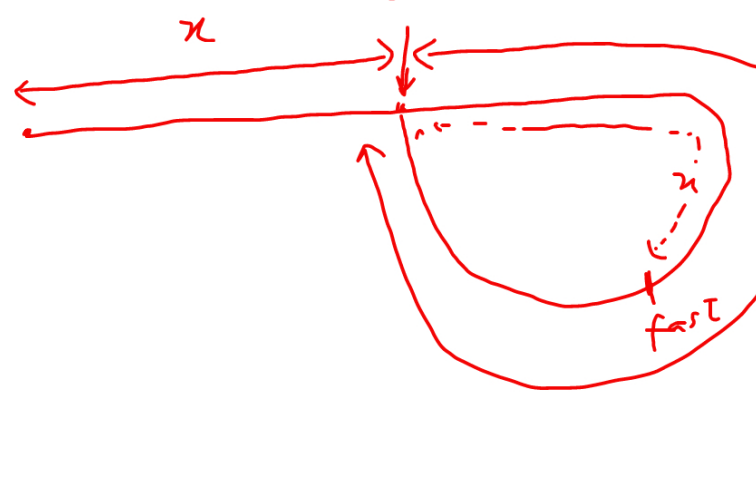
slow : One step at a time
fast : Two steps at a time

Claim: If the L.L. contains a cycle, the slow & fast pointers will eventually meet somewhere.

Proof:


slow
x
x
fast
y

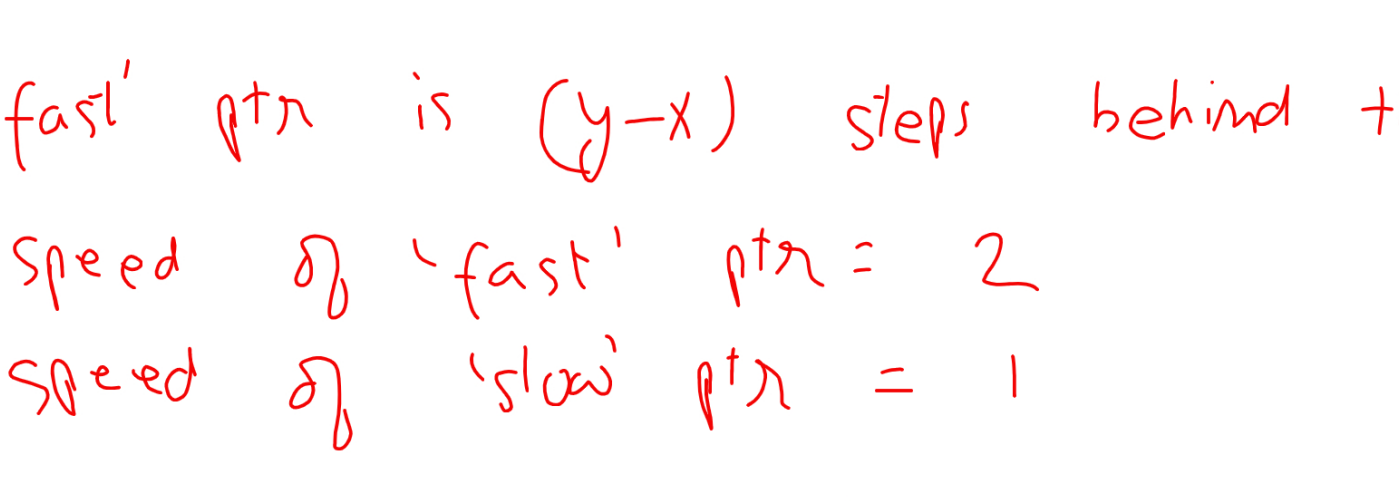
⇒ when 'slow' ptr reaches the intersection point, then the 'fast' ptr will be x steps inside the loop

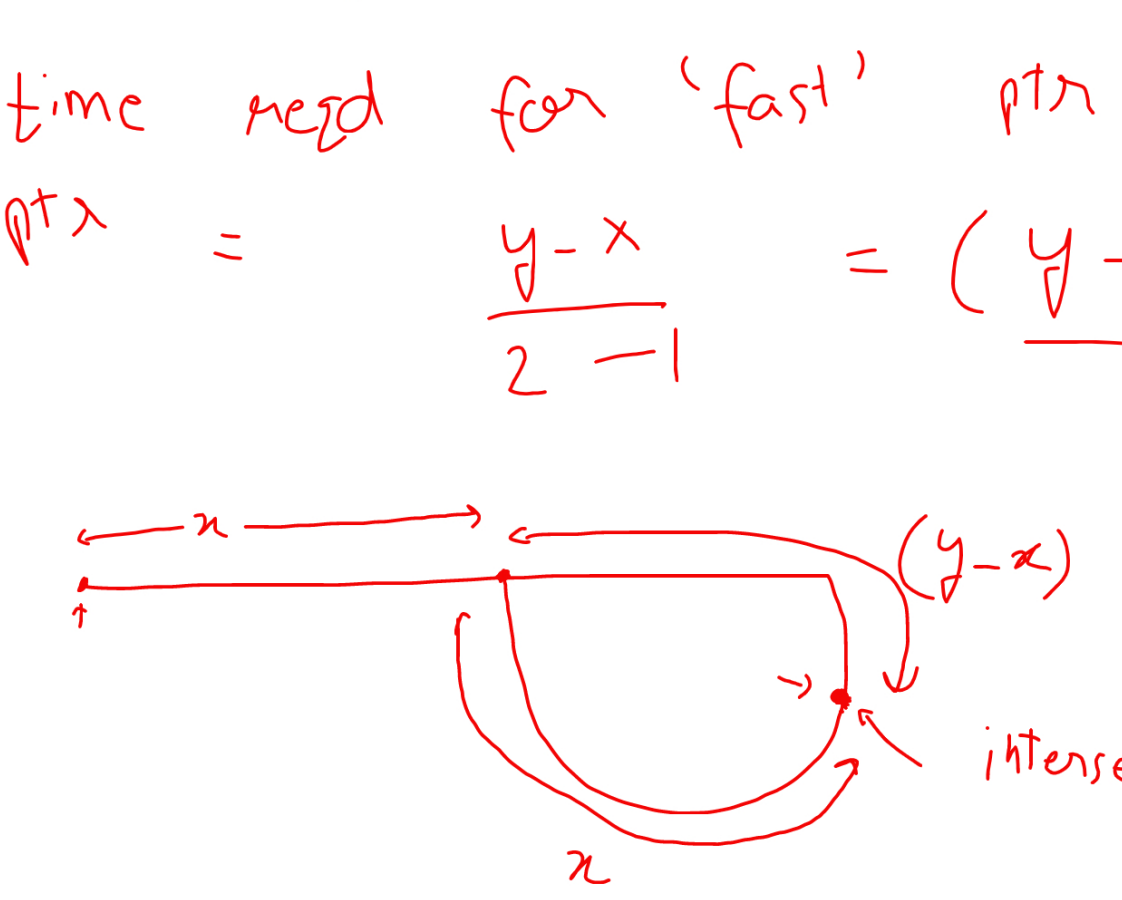

slow
y-x
fast

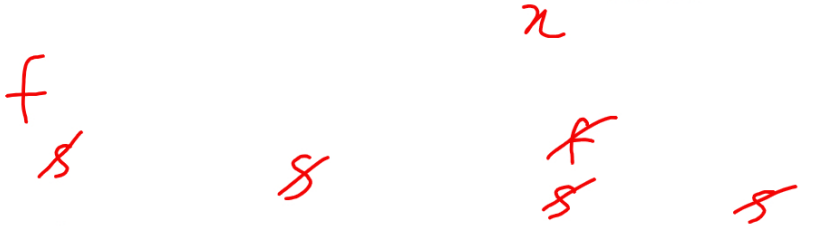
⇒ ['fast' ptr is (y−x) steps behind the 'slow' ptr.
Speed of 'fast' ptr = 2
Speed of 'slow' ptr = 1]

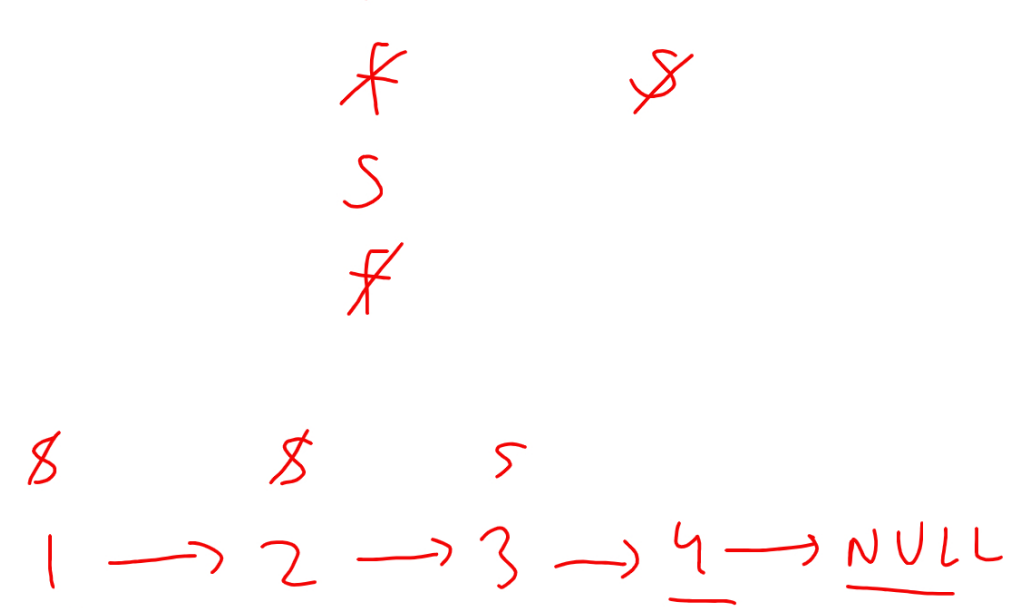
time reqd for 'fast' ptr to catch the 'slow' ptr = $\frac{y-x}{2-1}$ = (y − x)

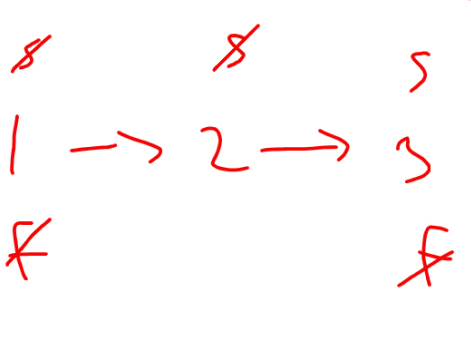
⇒


x
(y-x)
intersection pt
x

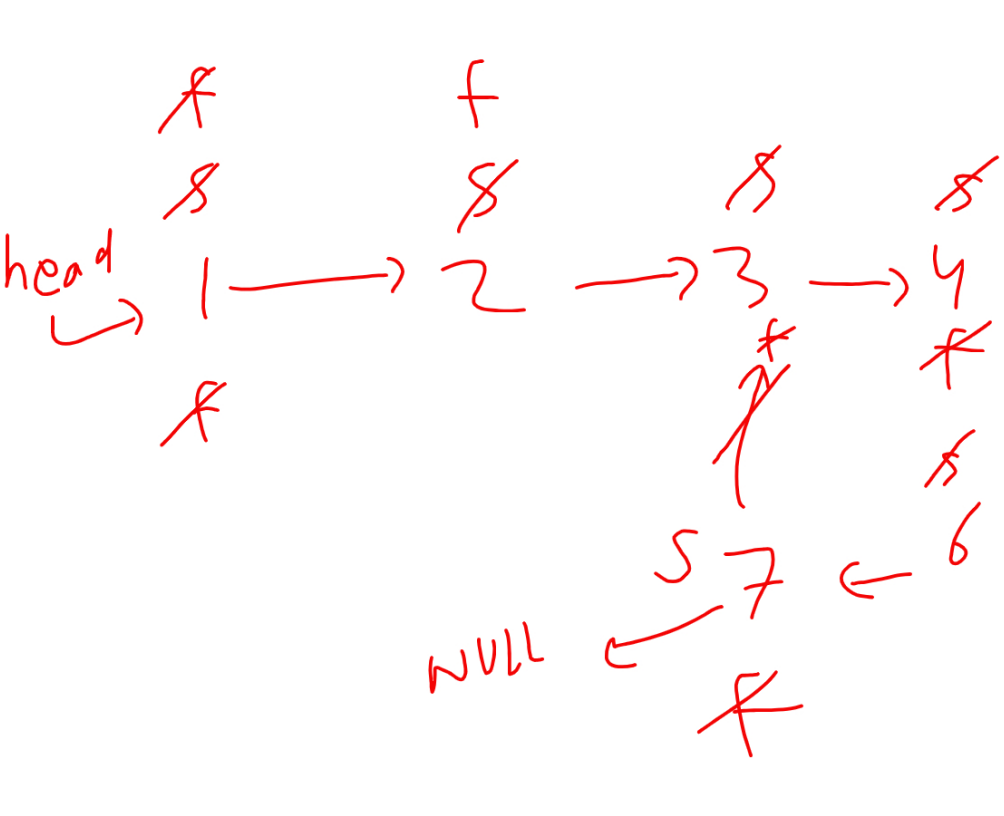

f
s
f
1
2
3
4
5
6
7
s
f
s


s
s
s
1
2
3
4
NULL
f
f
f
s
s
s
1
2
3
4
5
NULL
f
f
f


f
f
s
s
head
1
2
3
4
5
6
7
NULL
s
s
s
f
f



x = 0


(s, f)
1
2
3
4
5
6
s
fast
f
s
f

head
1 → NULL
X

1 → 2 → 3 → 4 → 5 → 6 → NULL
✓