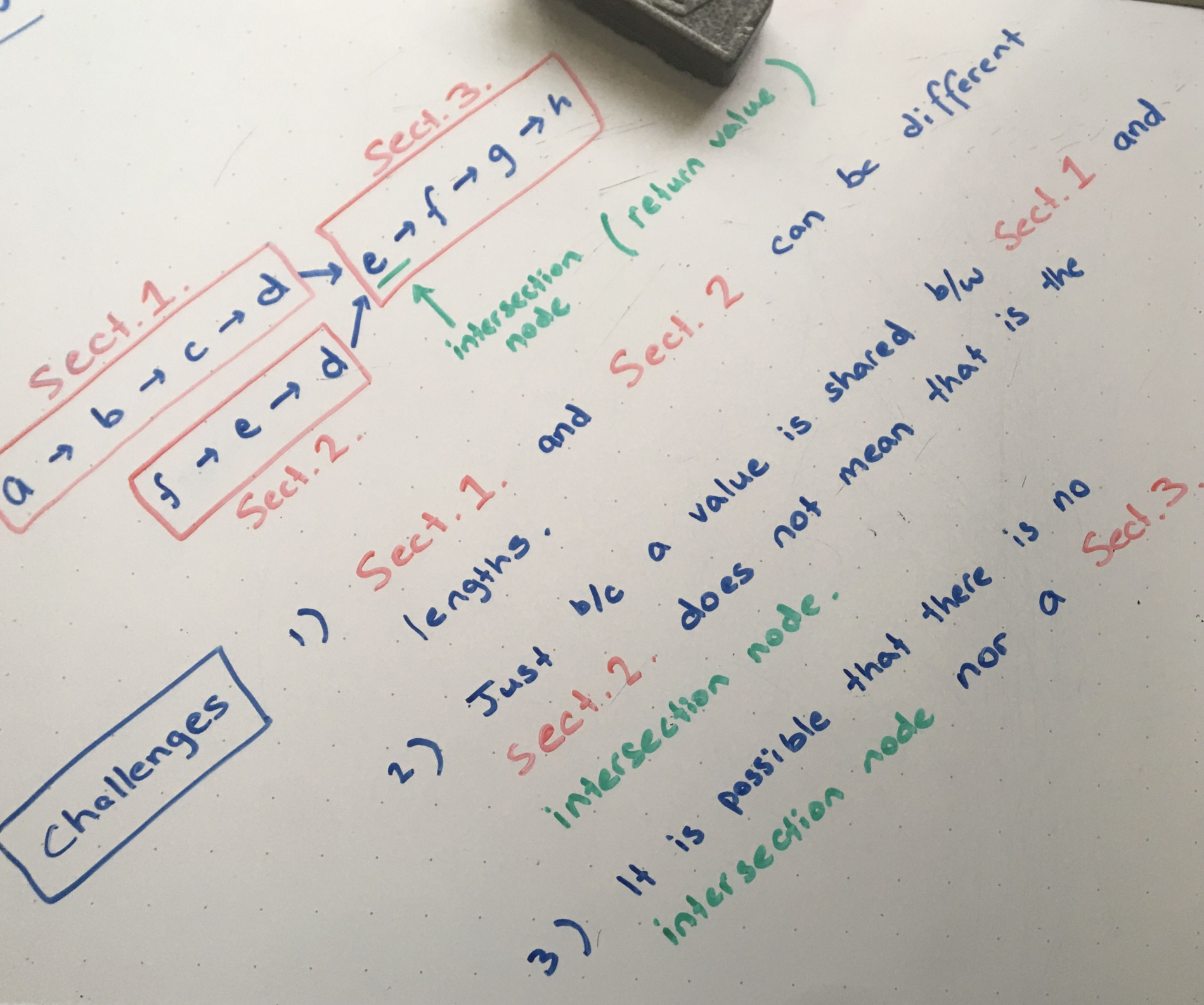
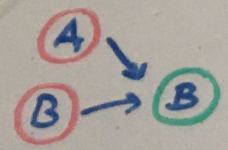
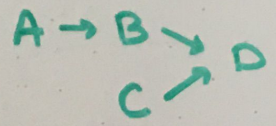

Sect. 1.
a → b → c → d
Sect. 2.
f → e → d
Sect. 3.
e → f → g → h
intersection node (return value)
Challenges
1) Sect. 1. and Sect. 2 can be different lengths.
2) Just b/c a value is shared b/w Sect. 1 and Sect. 2. does not mean that is the intersection node.
3) It is possible that there is no intersection node nor a Sect. 3.

How do we verify an intersection point assuming there is one?



A.next == B.next

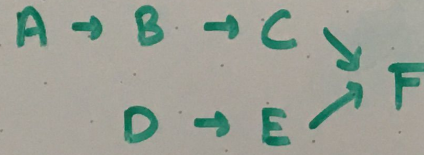
└ (node reference comparison)



∴ B.next == C.next

? A.next == C.next

? B.next == C.next



? A.next == D.next

? B.next == D.next

? C.next == D.next

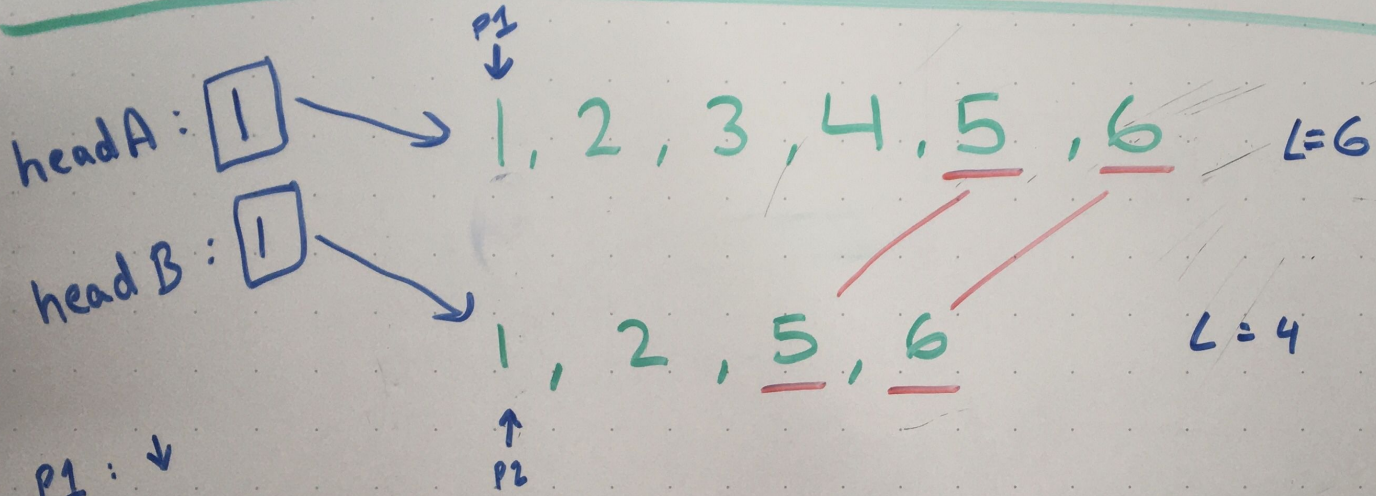
? A.next == E.next

? B.next == E.next

? C.next == E.next

However this yields $O(n^2)$ ...

How can I interate through the lists in $O(n)$ time ?



depending on which list is longer ...

P1 : $L_1 - L_2$: [2] = 3 4 5

P2 : x x 5