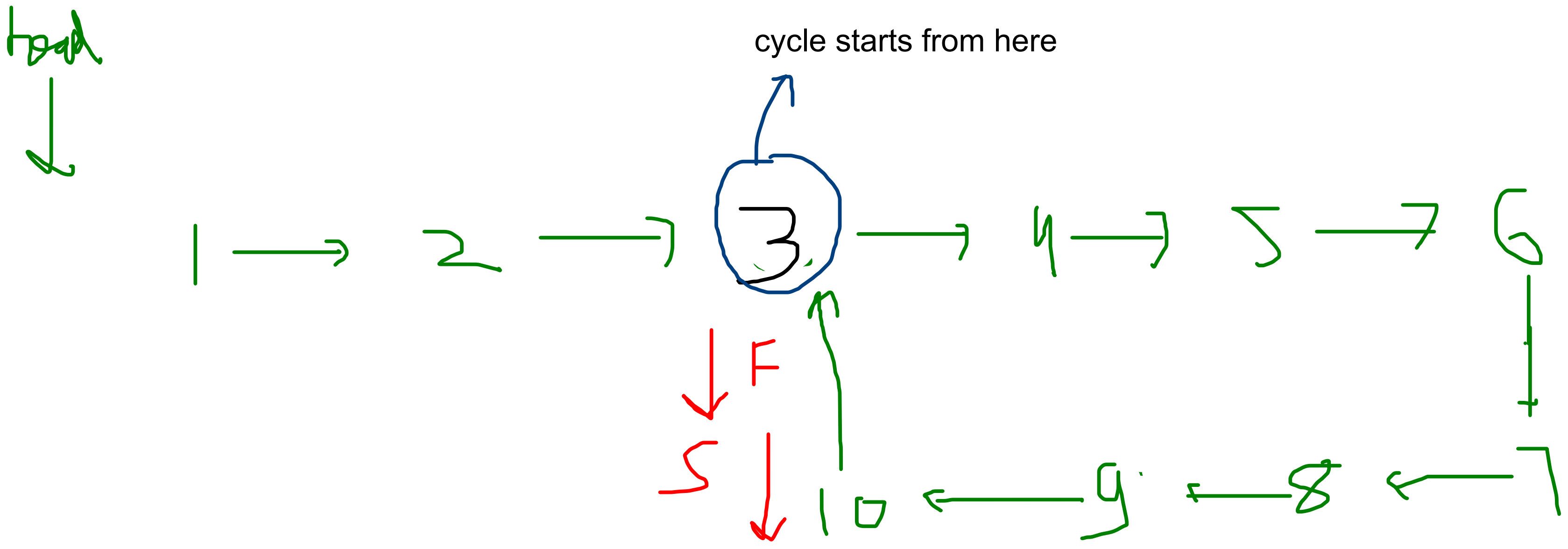
Detect Cycle point in LL



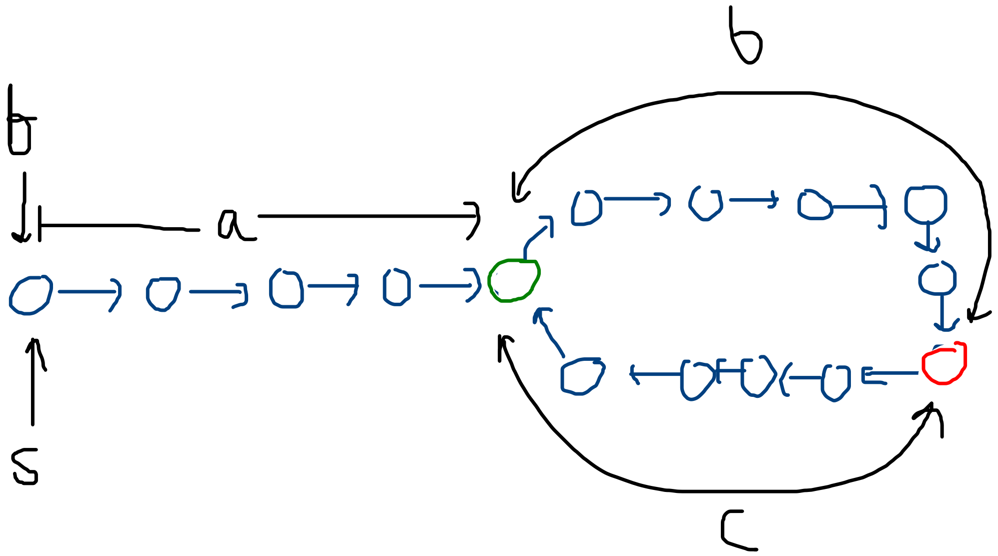
after meet of S and F

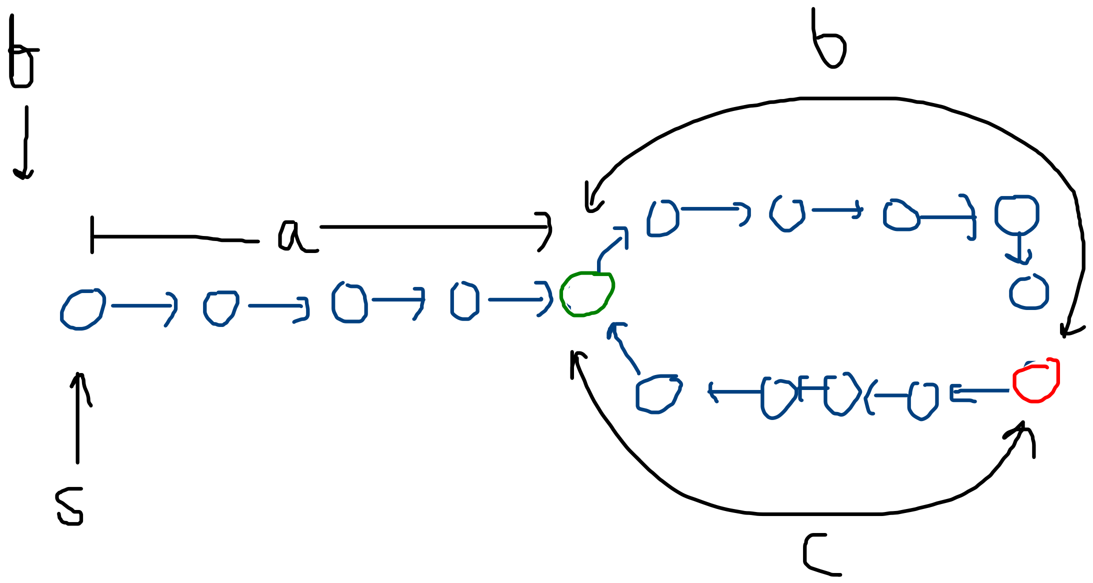
Reset slow to head

move both by one one

proof of cycle detection algo

🟢	Cyclic point
🔴	fast and slow meets first time
a	distance b/w head and green
b	distance from green to red
c	distance from red to green





d_s distance travel by S to reach red from head

d_f distance travel by F to reach red from head

n number of rotations S take before meeting of F pointer

m number of rotations F take before meeting of S pointer

ds = a + n x (b + c) + b

df = a + m x (b + c) + b

dist = velocity x time

time = dist/velocity

T = ds/s , T = df/f

$$\frac{d_s}{s} = \frac{d_f}{f}$$

$$df = \left(\frac{f}{s}\right) ds$$

let $\frac{f}{s} = \pi$

$$df = \pi\, ds$$

$$a + m(b+c) + b = \pi(a+b) + \pi . n (b+c$$

$$(b+c)\left[m - \pi . n\right] = (a+b)(\pi - 1)$$

$$\Rightarrow \boxed{a+b = \frac{(m - \pi n)(b+c)}{(\pi - 1)}}$$

$$\Rightarrow \boxed{a+b = \frac{(m-nn)(b+c)}{(n-1)}}$$

$$n-1 \neq 0$$

$$n-1 > 0$$

$$n > 1$$

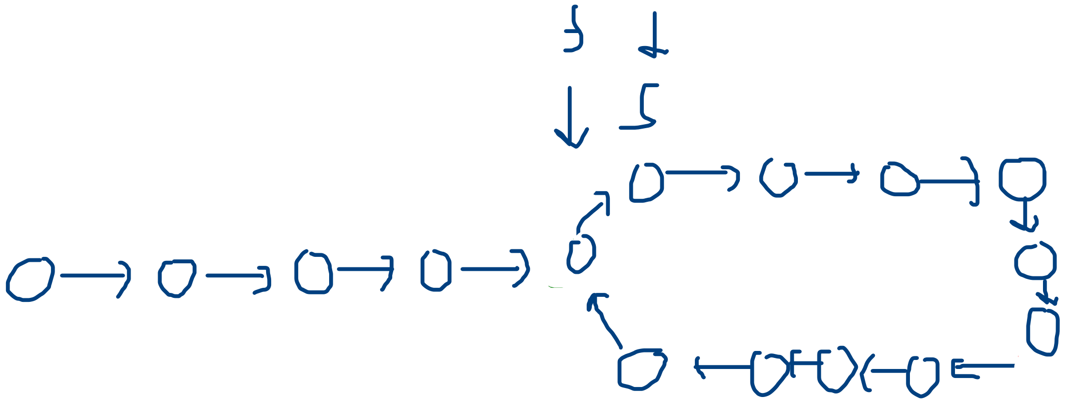
$$\frac{f}{s} > 1$$

$$\boxed{p \in R^{+},\ p \neq 1}$$

$$\boxed{f = ps}$$

$m = 1$

$n = 1$



f = p x s, p = 2, p -> R+, p != 0

$f = ps$

$f = 3$

$s = 2$

best --> s will cover total of 0 rotation

$f = 2s$

$$\boxed{a + b = \frac{(m - 2n)(b + c)}{(n - \cancel{1})} \quad 1}$$

r = 2

$$k$$

$$a + b = (\cancel{m} - 2n)(b + c)$$

$$a + b = (b + c)\, k \quad 1$$

m - 2n

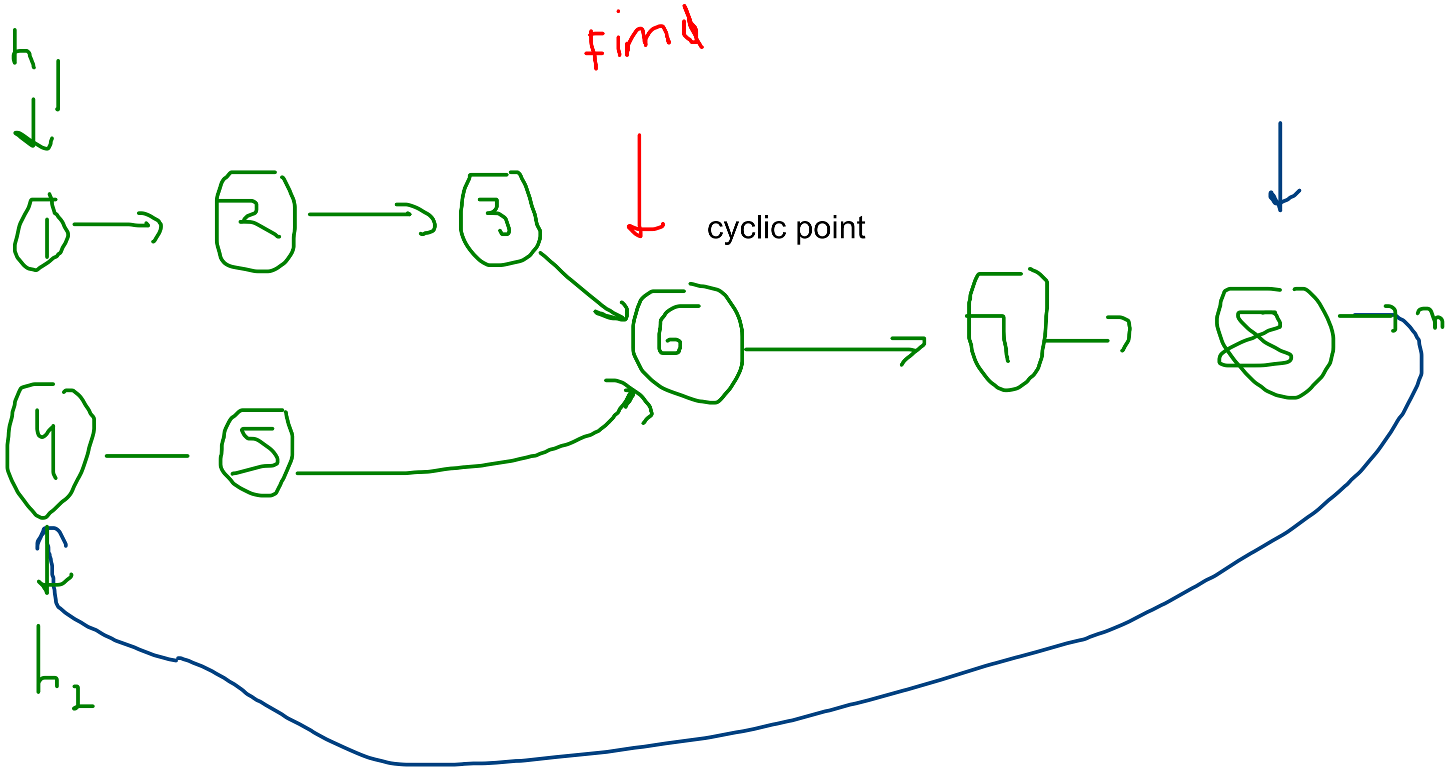
$$a + \cancel{b} = \cancel{b} + c$$

a = c

dist from head to green and red to green is equal

Intersection of LL

M1



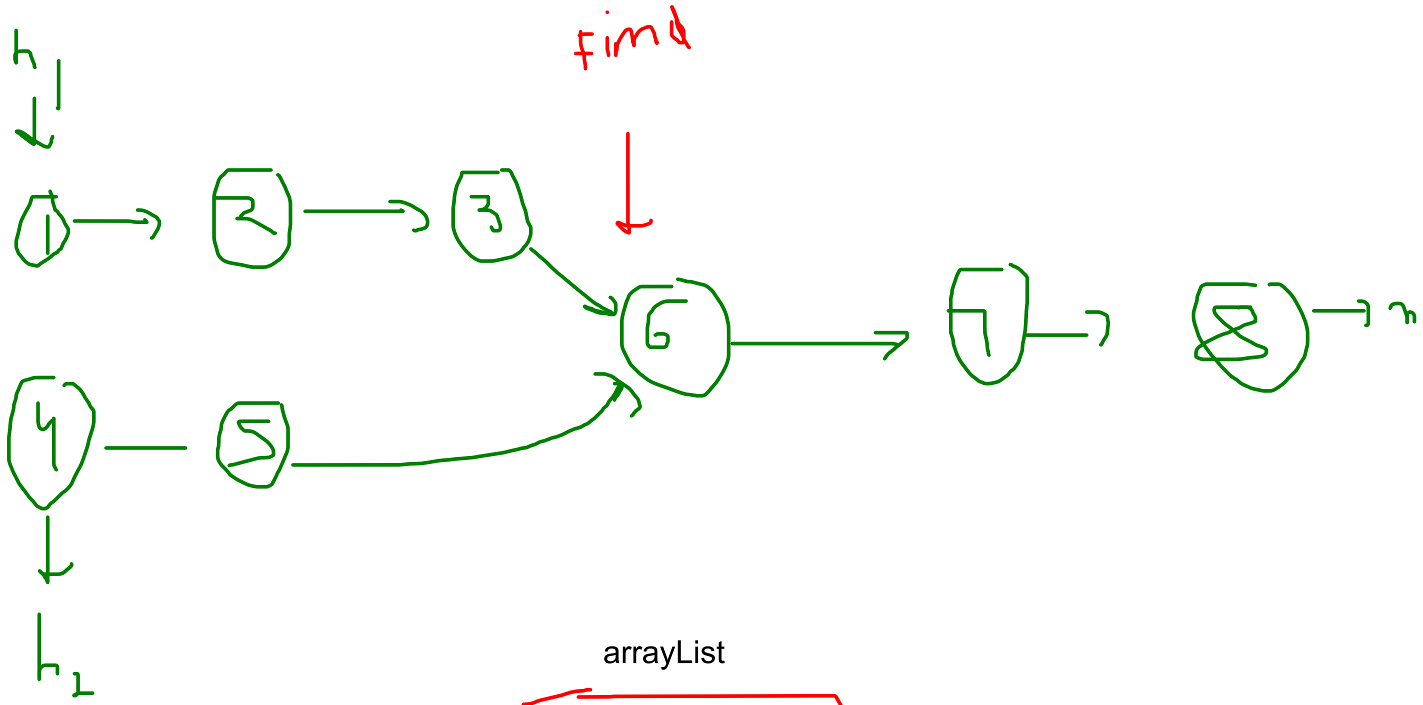
input

head1 and head2

↓ correct input again

Intersection of LL

M2



input

head1 and head2

arrayList

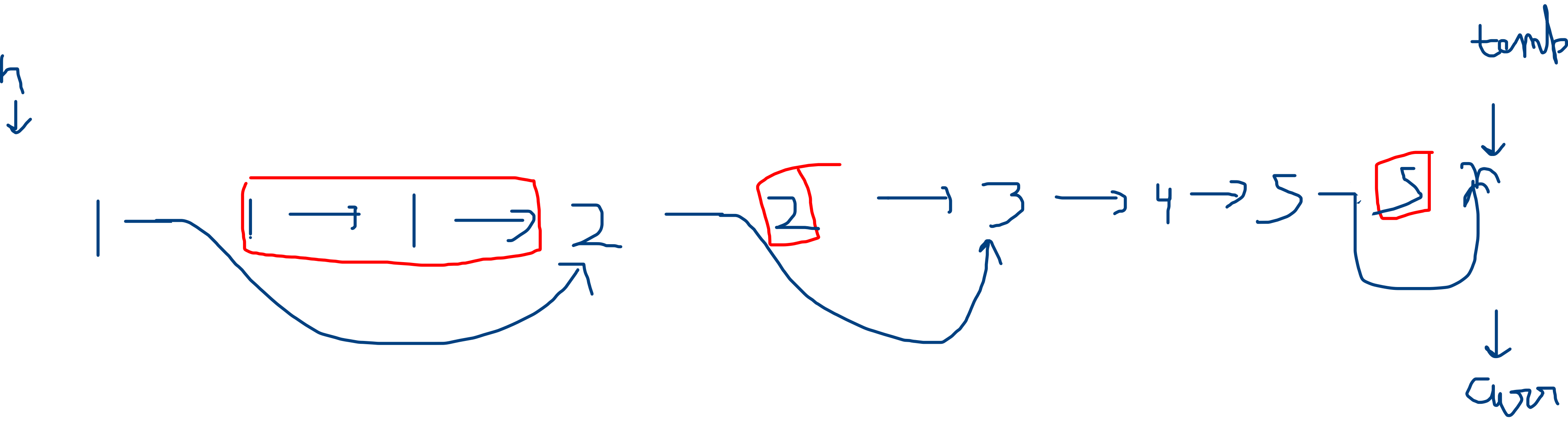
arr1 = 1, 2, 3, 6, 7, 8

arr2 =   4, 5, 6, 7, 8

Remove Duplicates

$$\begin{tikzcd}
h \arrow[d] \\
1 \arrow[r] & 1 \arrow[r] & 1 \arrow[r] & 2 \arrow[r] & 2 \arrow[r] & 3 \arrow[r] & 4 \arrow[r] & 5 \arrow[r] & 5
\end{tikzcd}$$

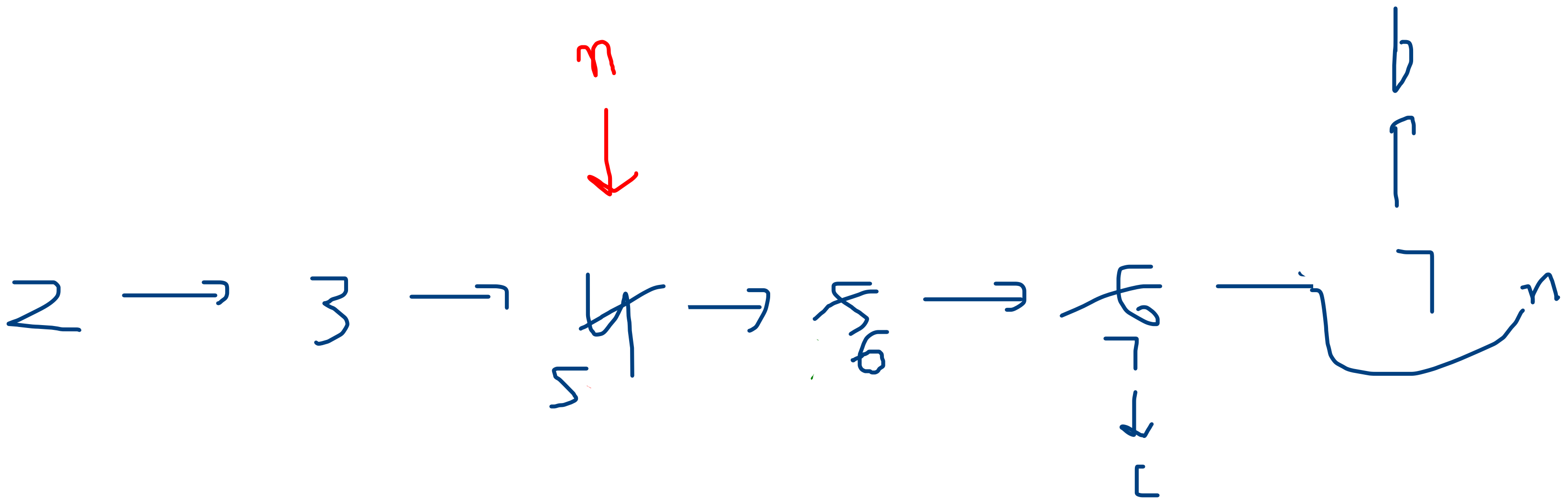
o/p $\Rightarrow$ $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$



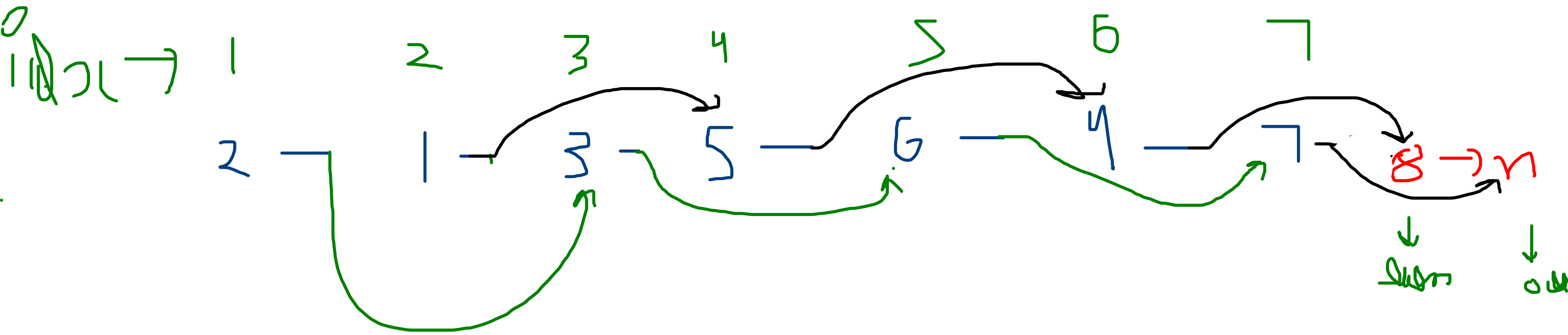
```
curr = head

while(curr != null) {
            temp = curr
}
```

remove a node in LL

head is not given,
remove node is given



Odd Even



odd = head
even = head.next

odd.next = even.next

odd = odd.next

even.next = odd.next

even = even.next

2 [2,1,3,5,6,4,7]

0 [2,3,6,7,1,5,4]

odd even

2 → 3 → 6 → 7 → 1 → 5 → 4