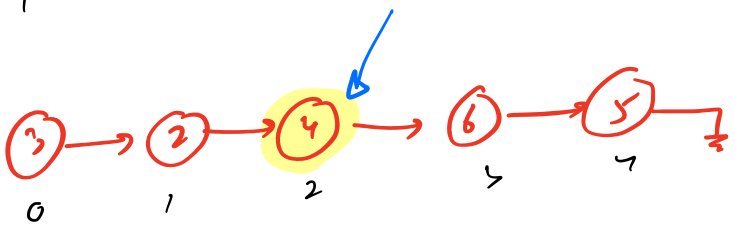
Q Given LL. Find the middle most node!



N = 5 : ODD

$(5-1)/2$

2

N = 6 : EVEN

$(6-1)/2$

2

find $(N-1)/2$ th Node!

```
N = getLen(h);
Node x = findKthNode(h, (N-1)/2);
get x;
```

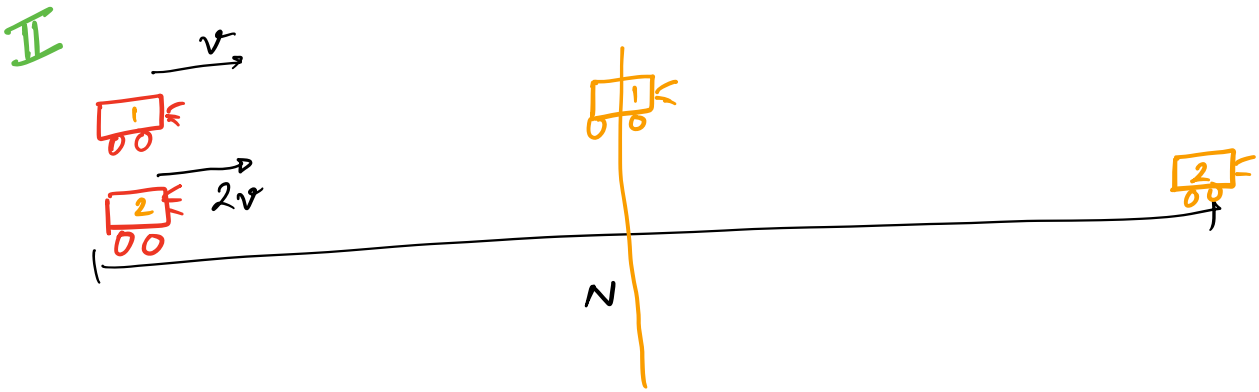
N

N/2

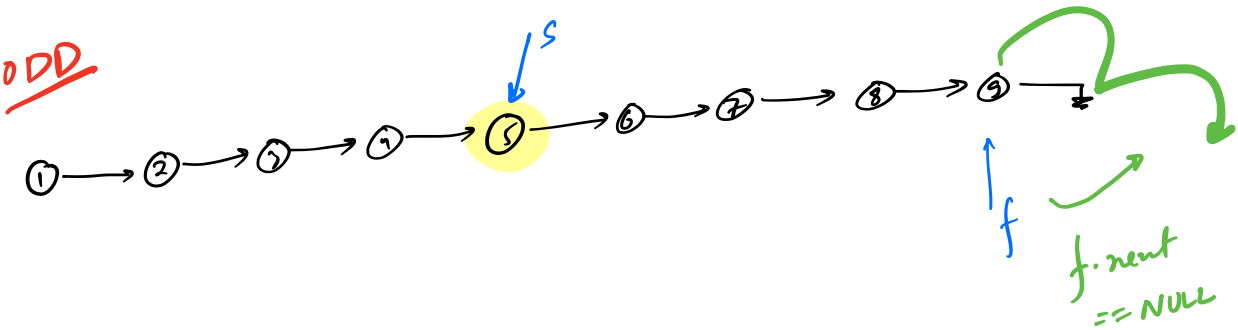
TC = O(N)

SC = O(1)

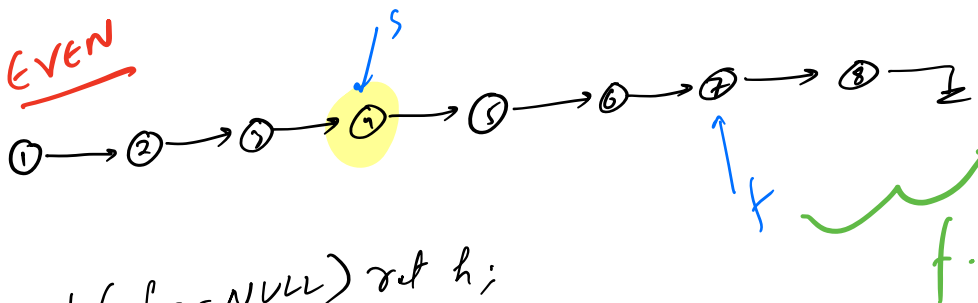
II



ODD



EVEN



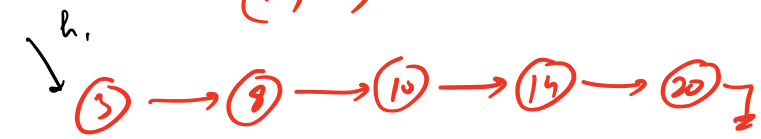

```
if ( h == NULL ) ret h;
Node s = h, f = h;
while( f.next != NULL && f.next.next != NULL ) {
        s = s.next;
        f = f.next.next;
}
ret s;
```

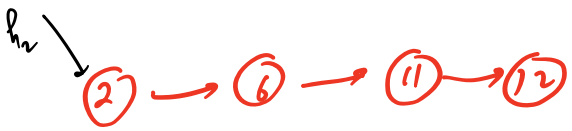
TC = O(N)

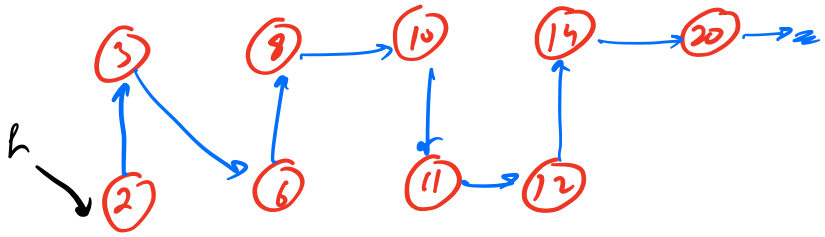
SL = O(1)

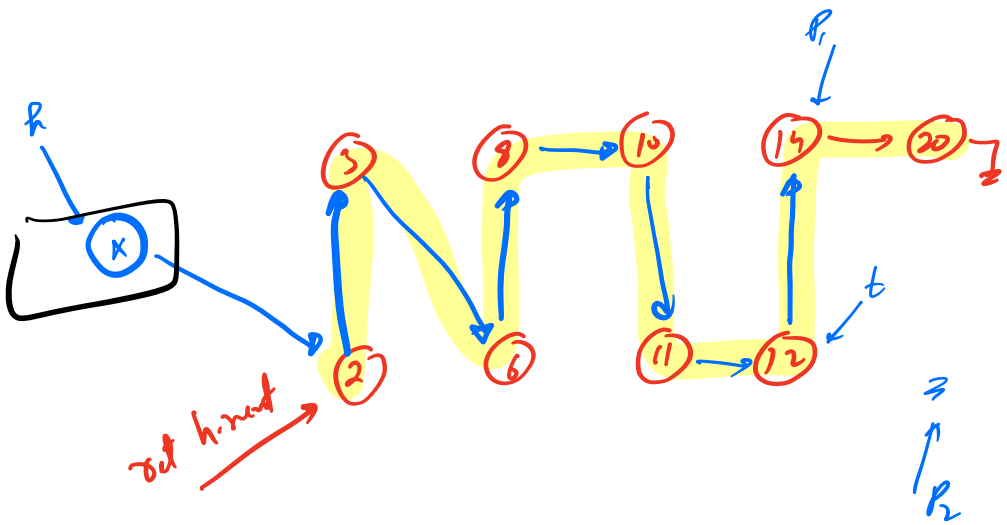
Q Given 2 sorted LL. Merge them into a single sorted LL.

(N, M)

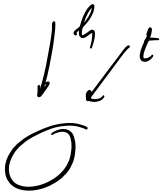








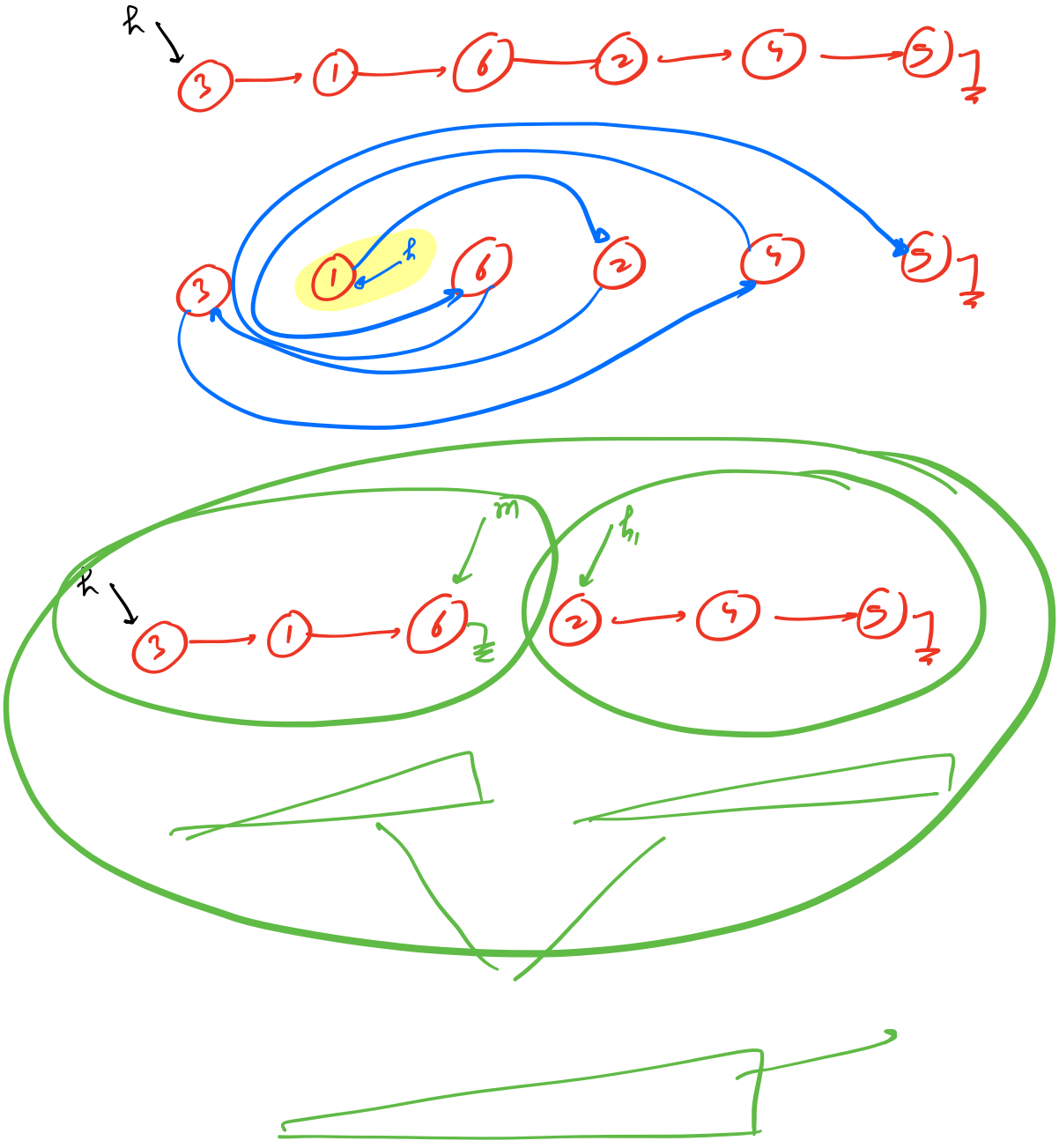
```
Node h = new Node(-1);
Node t = h;
Node p1 = h1
Node p2 = h2
while( p1 != NULL && p2 != NULL)
    if( p1.data < p2.data ) {
        t.next = p1;
        t = t.next;
        p1 = p1.next;
    }
    else {
        t.next = p2;
        t = t.next;
        p2 = p2.next;
    }
}
if( p1 == NULL) {
    t.next = p2;
}
else {
    t.next = p1;
}
ret h.next;
```



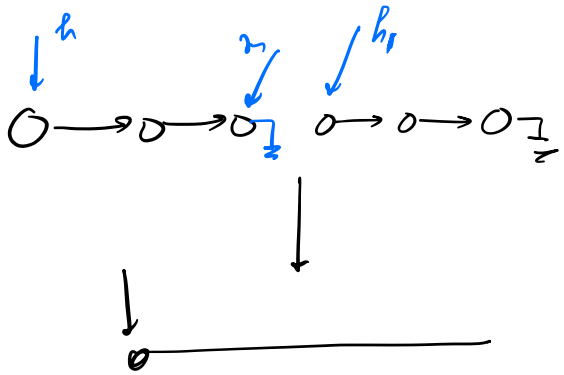
TC = O(N+M)

SC = O(1)

Given a LL. Sort it!



```
Node mergeSort(Node h) {
    if (h == NULL || h.next == NULL) {
        ret h;
    }
    Node m = getMiddle(h);
    Node h1 = m.next;
    m.next = NULL;
    h  = mergeSort(h);
    h1 = mergeSort(h1);
    ret merge(h, h1);
}
```



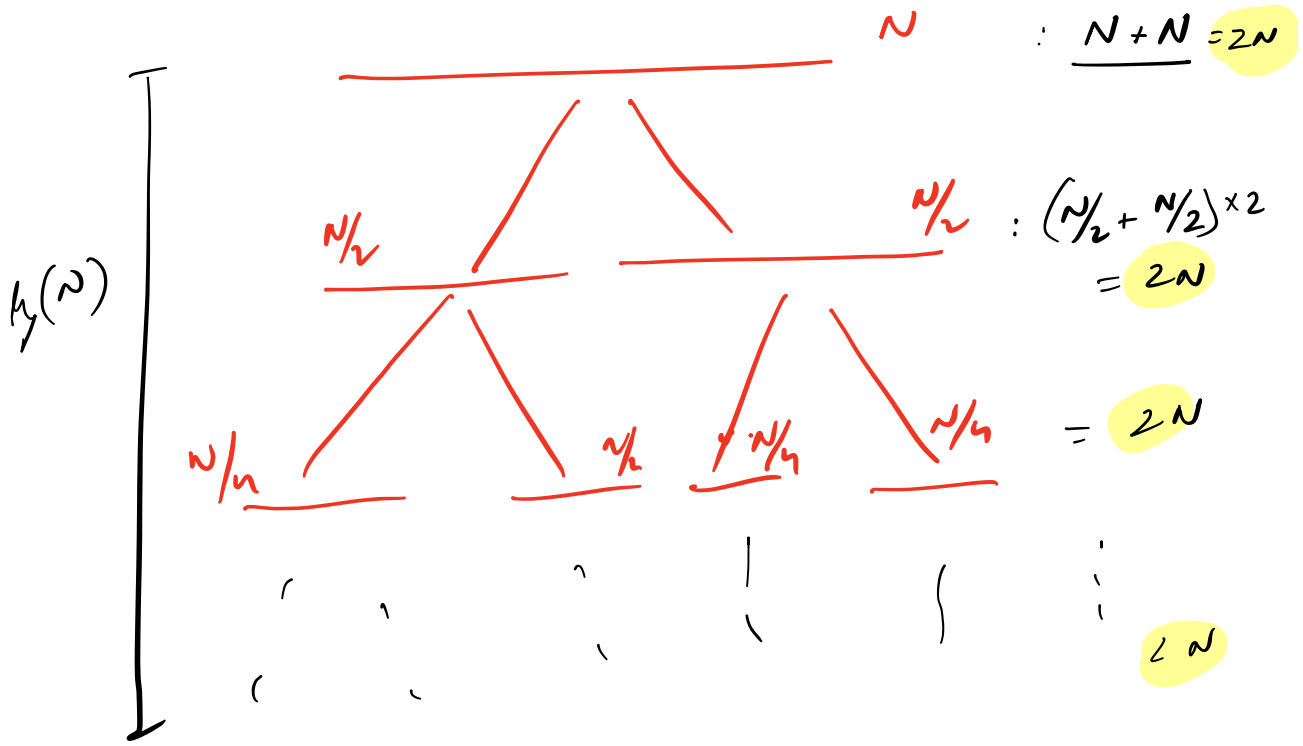
$$T(N) = N + N + 2T(N/2)$$

$$T(N) = 2T(N/2) + 2N$$

$$T(N) = 2[T(N/2 + N)] \rightarrow N \log N$$

$$TC = O(N \log N)$$

$$SC = O(\log N)$$

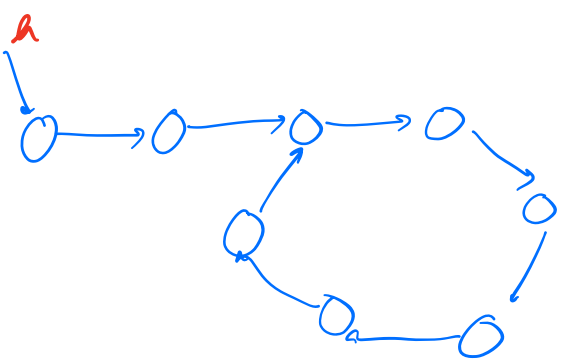


$TC = O(N \log N)$

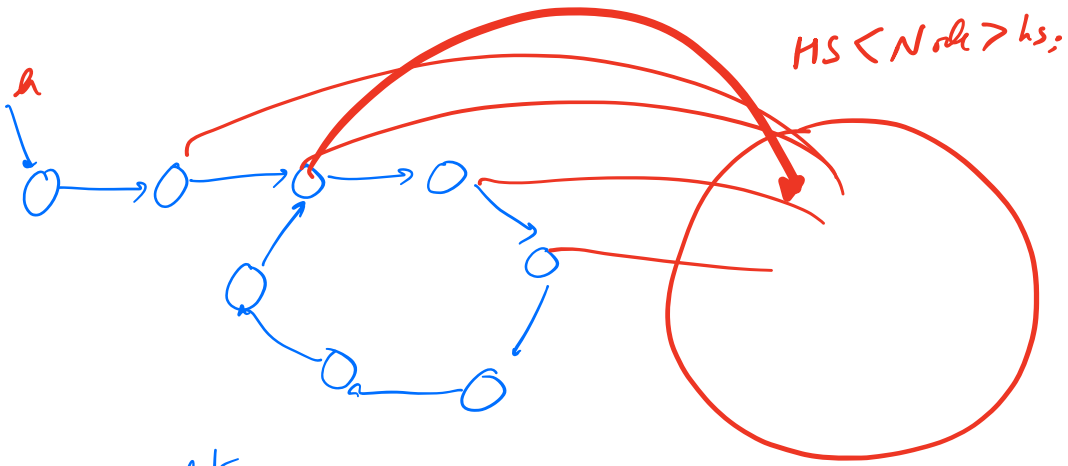
Q Given a LL. Detect if there is a cycle!

I

X



✓



Stopping Conditions

→ If Node ref is already in HS

→ CYCLE ✓

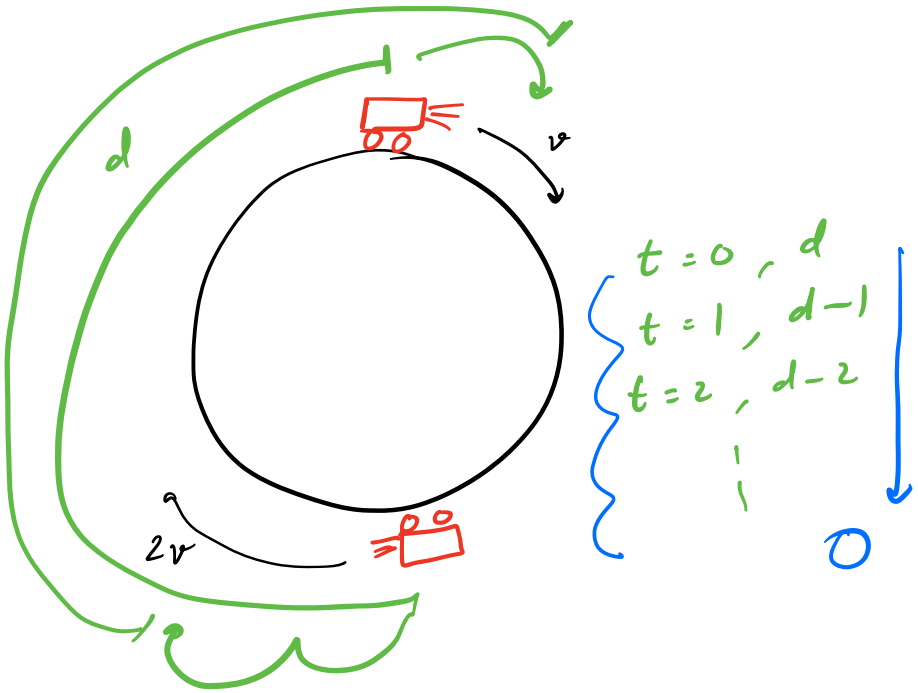
→ Node is NULL

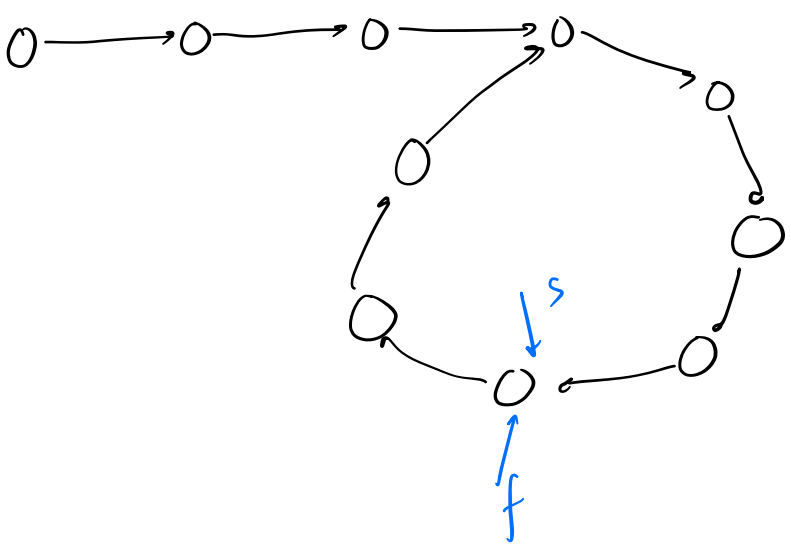
→ CYCLE X

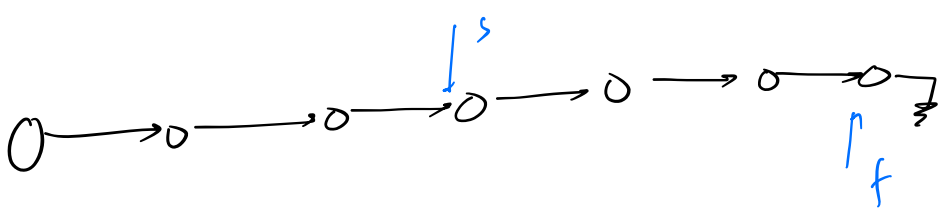
TC = O(N)

SC = O(N)

II



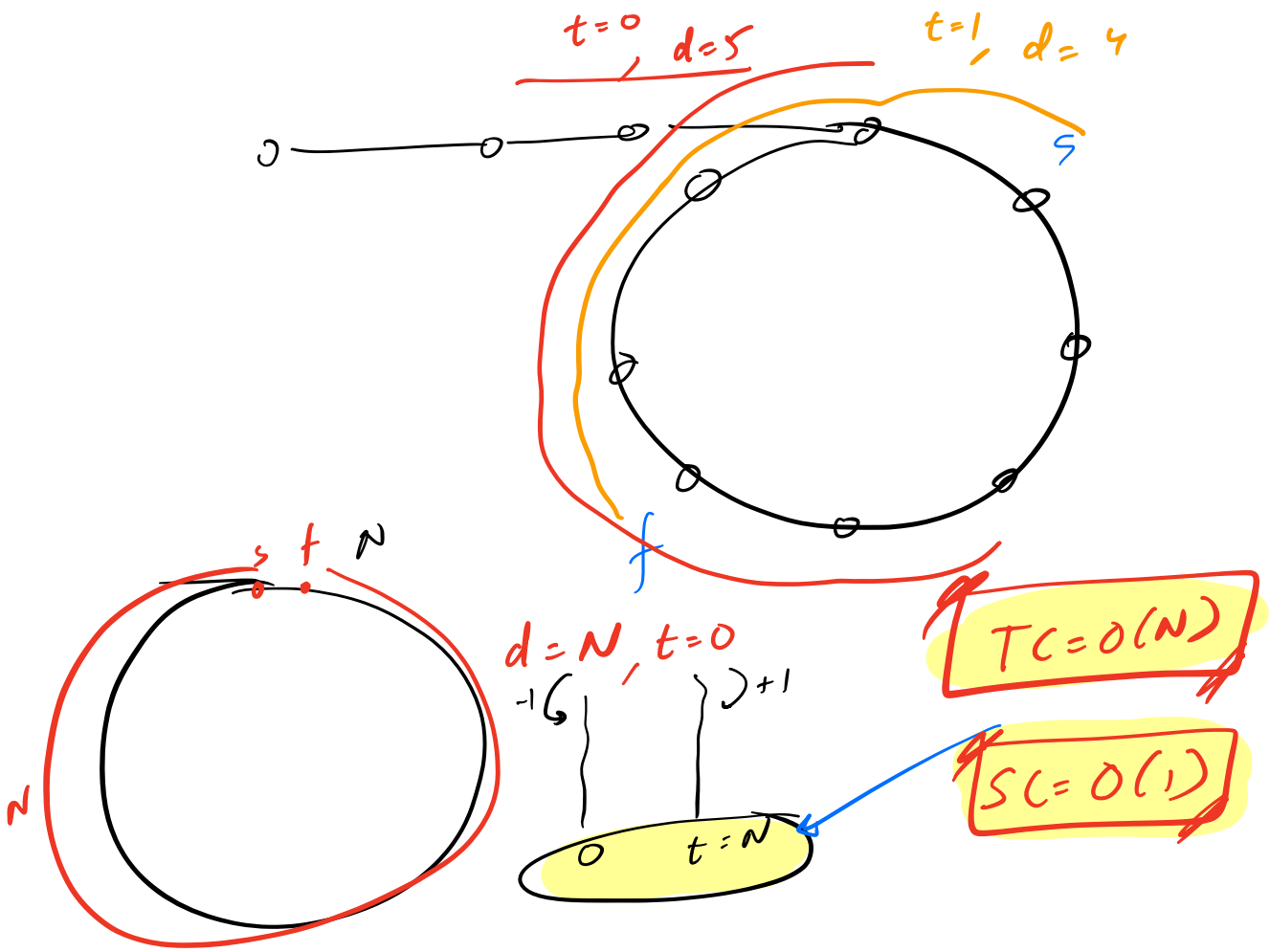




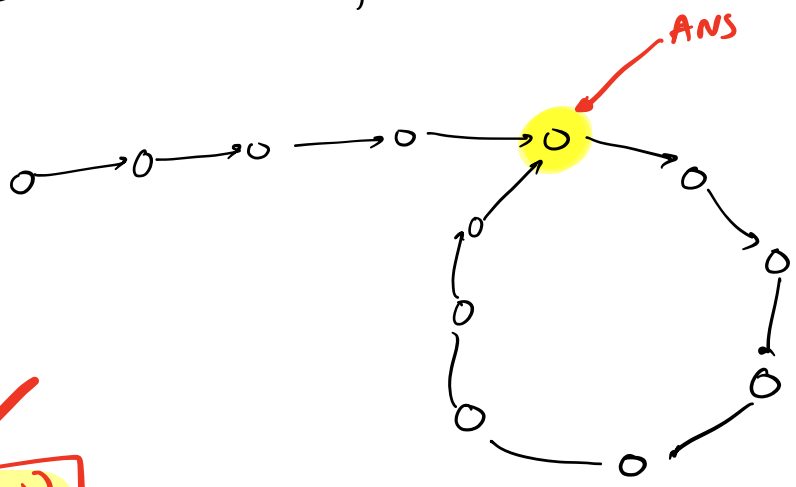
```
Node s = h, f = h;
while( f.next != NULL && f.next.next != NULL) {
    s = s.next;
    f = f.next.next;
    if (f == s) {
        ret true;
    }
}
ret false;
```

cycle ✓

X



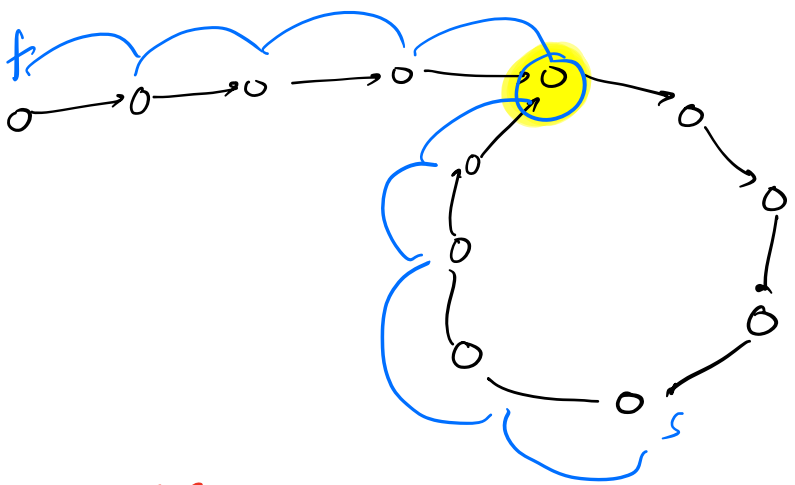
Q SAME Q. find the starting node of the loop!



1) HS ✓

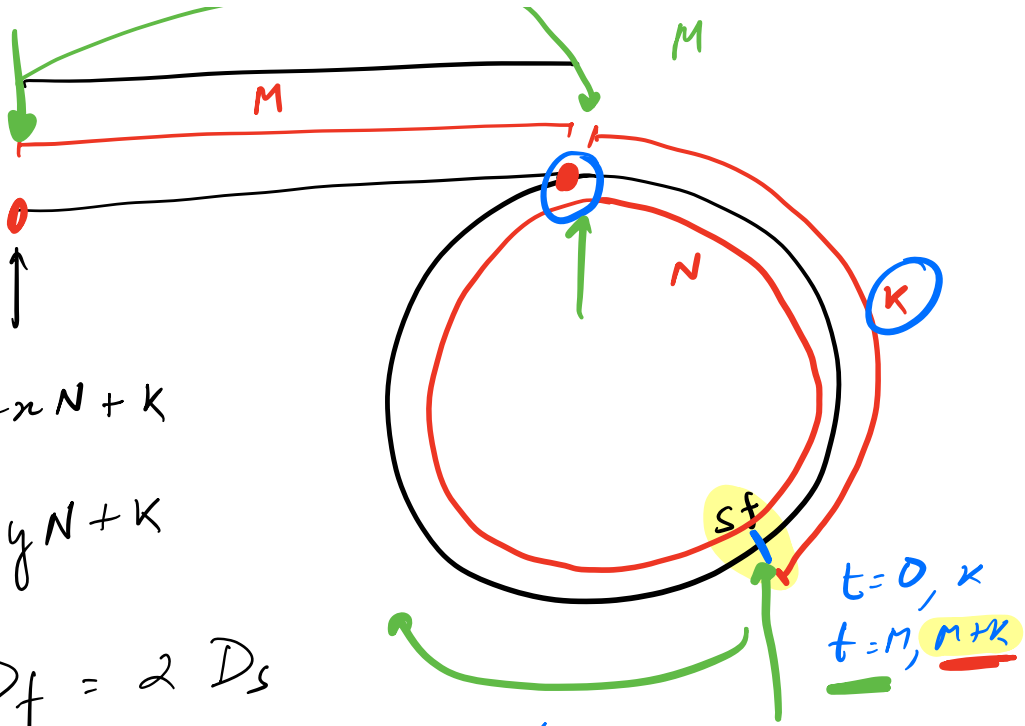
TC = O(N)

SC = O(N)



STEPS:

1) Do slow & fast ptr thing.
2) Once they meet, take f and put it as head.
3) Move them with 1 step at a time speed
4) Wherever they meet first → ANS!



$D_s = M + xN + K$

$D_f = M + yN + K$

$D_f = 2\,D_s$

$M + yN + K = 2M + 2xN + 2K$

$M + K = yN - 2xN$

$\boxed{M + K = (y - 2x).N}$

M+K is a MULTIPLE of N

Node x = new Node(s);