unacademy

Stack and Queue - Part I

Course on Data Structure

Pankaj Sharma • Lesson 14 • July 24, 2024

unacademy

CS & IT Engineering

Data Structure

Linked List

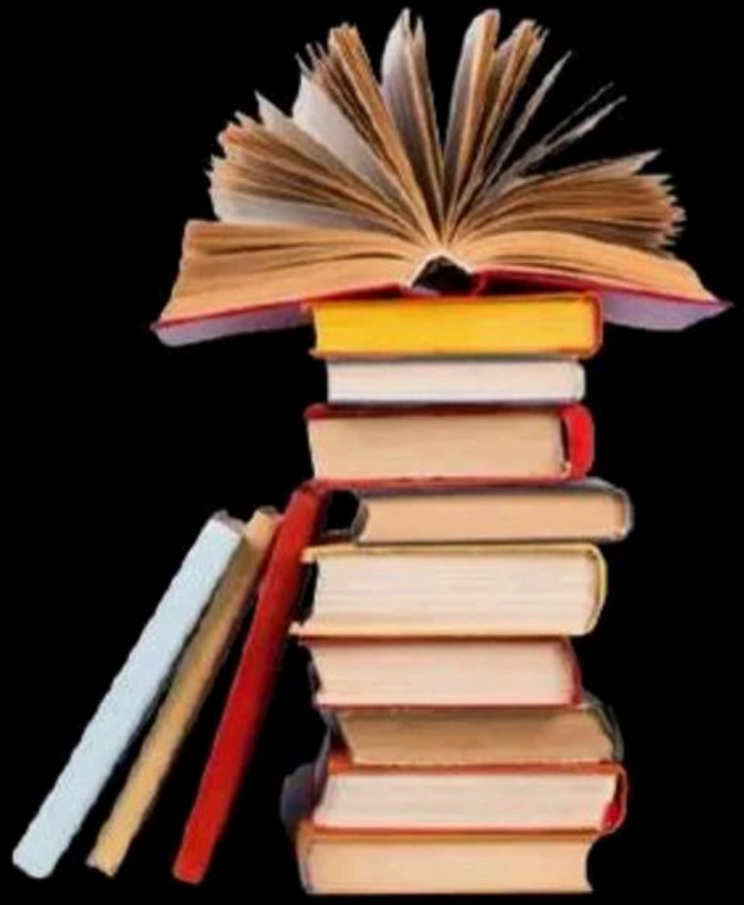
Lecture Number- 14

By- Pankaj Sir

unacademy

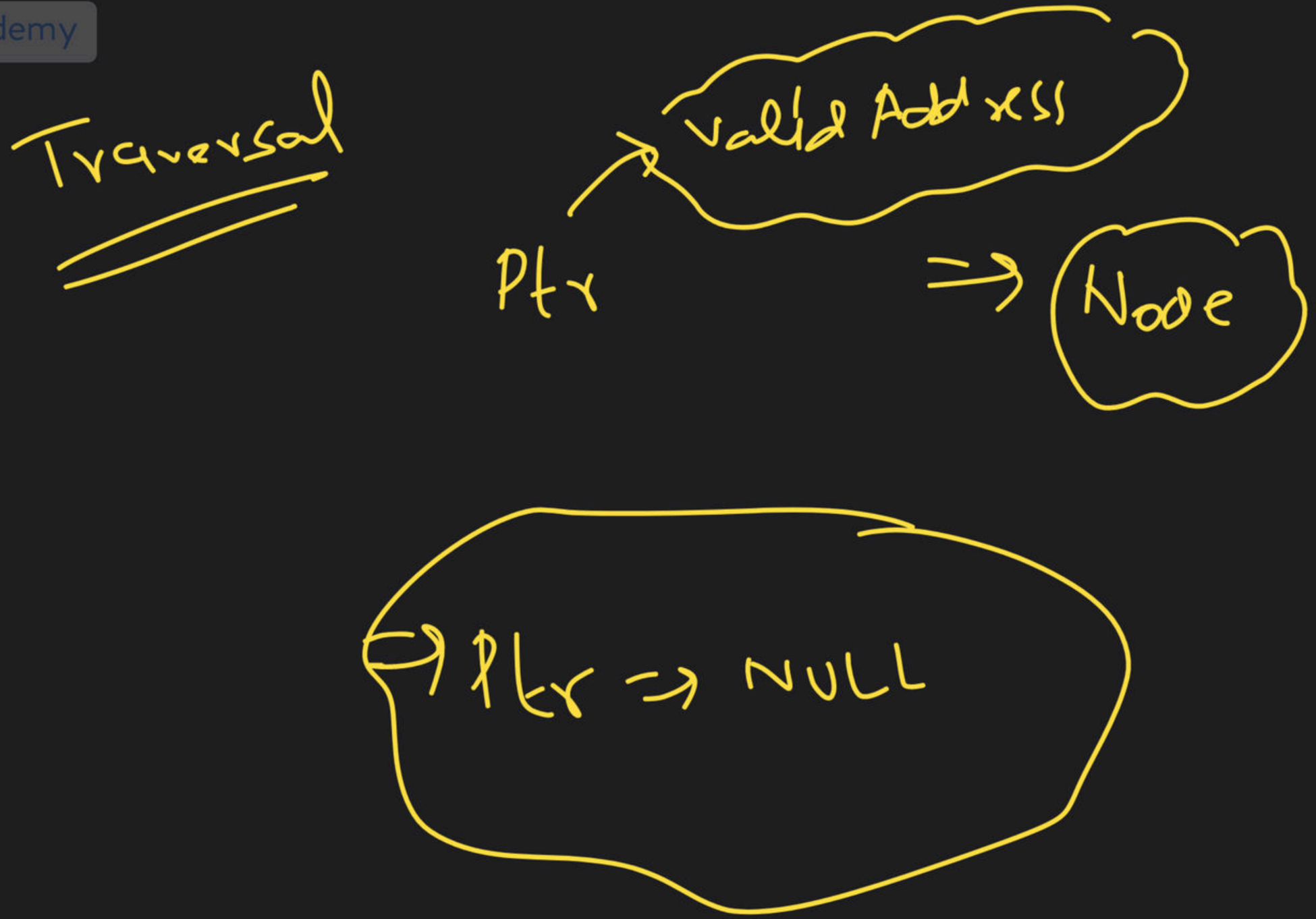
Topics

to be covered



1. Linked List

Traversal



Last node

while(ptr->Next != NULL)

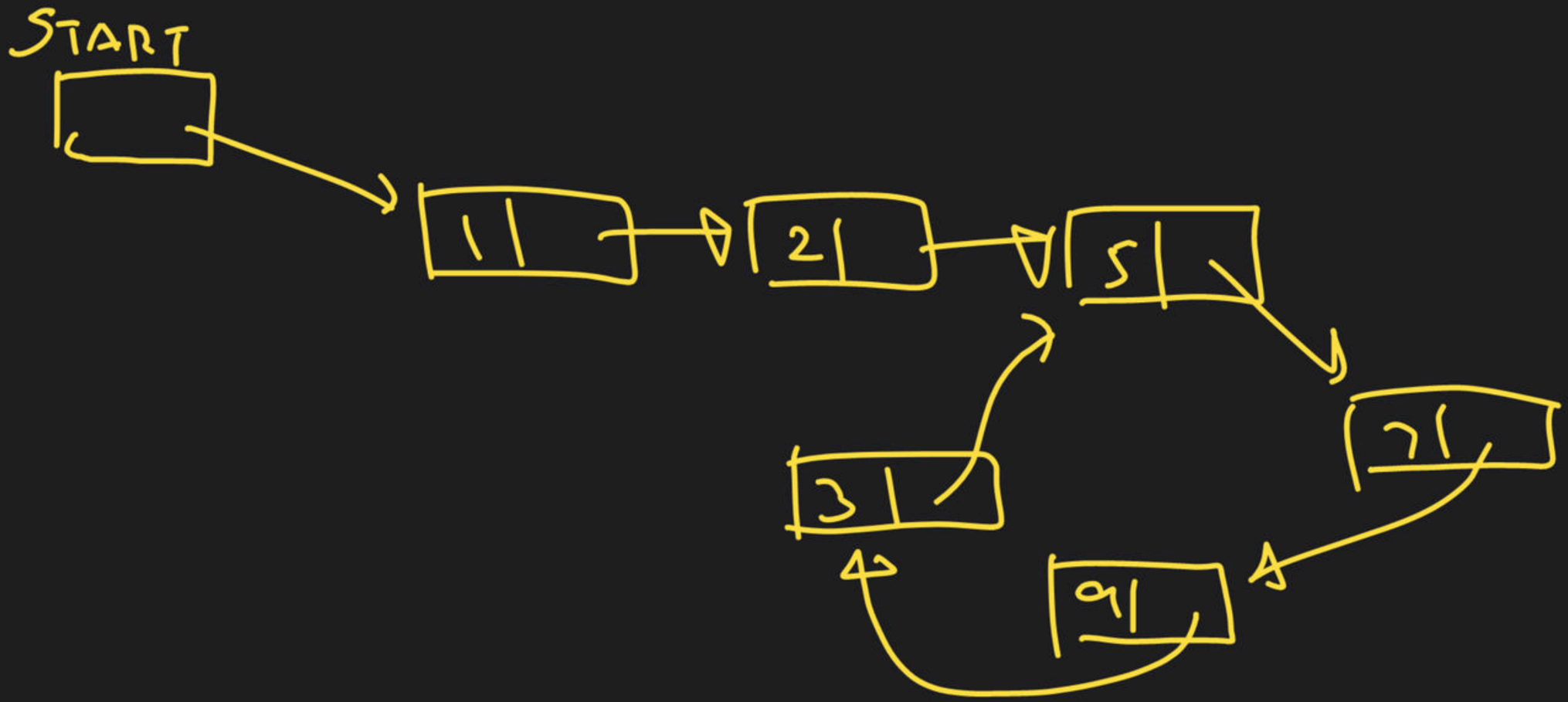
{

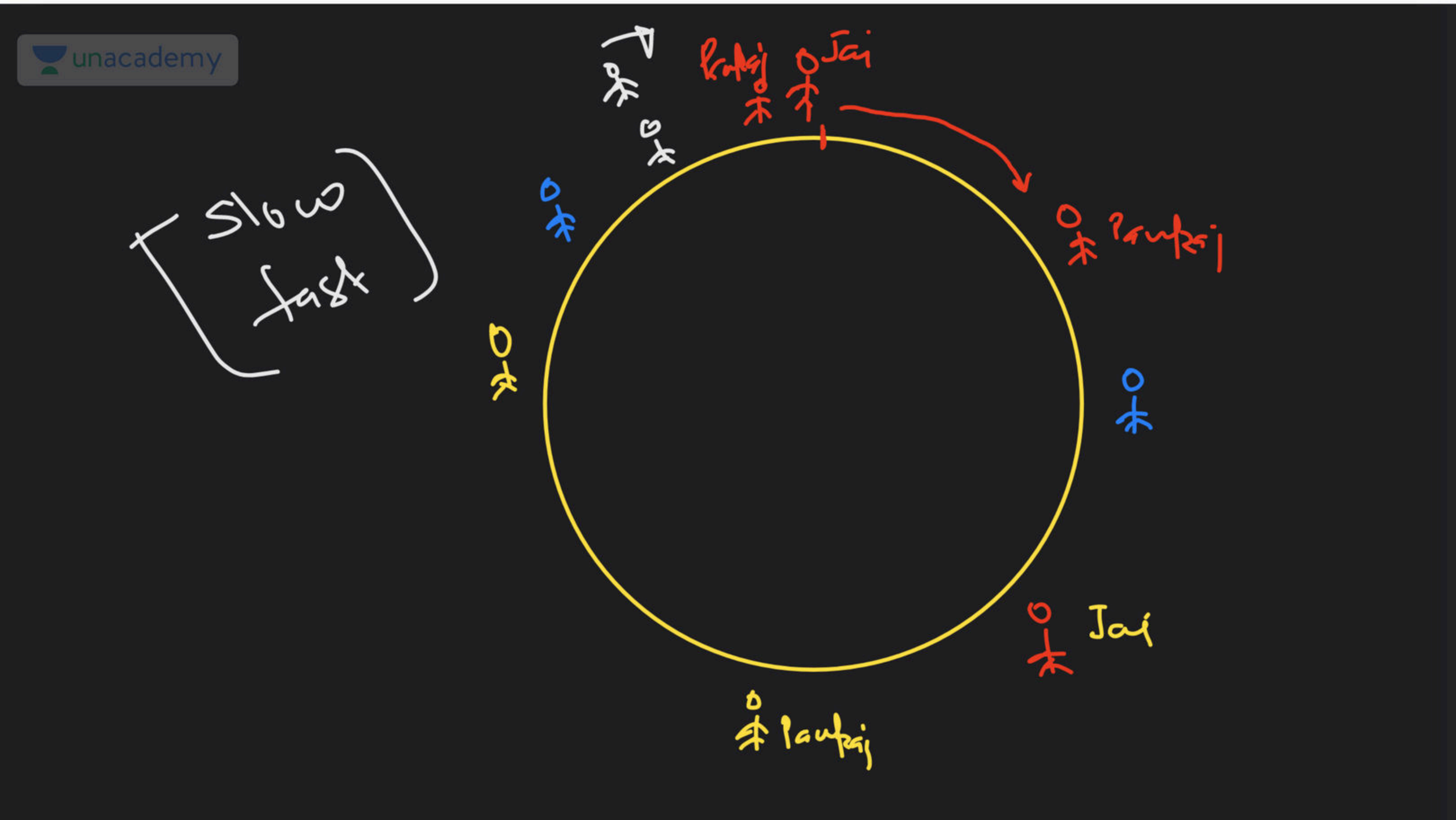
LinkedList

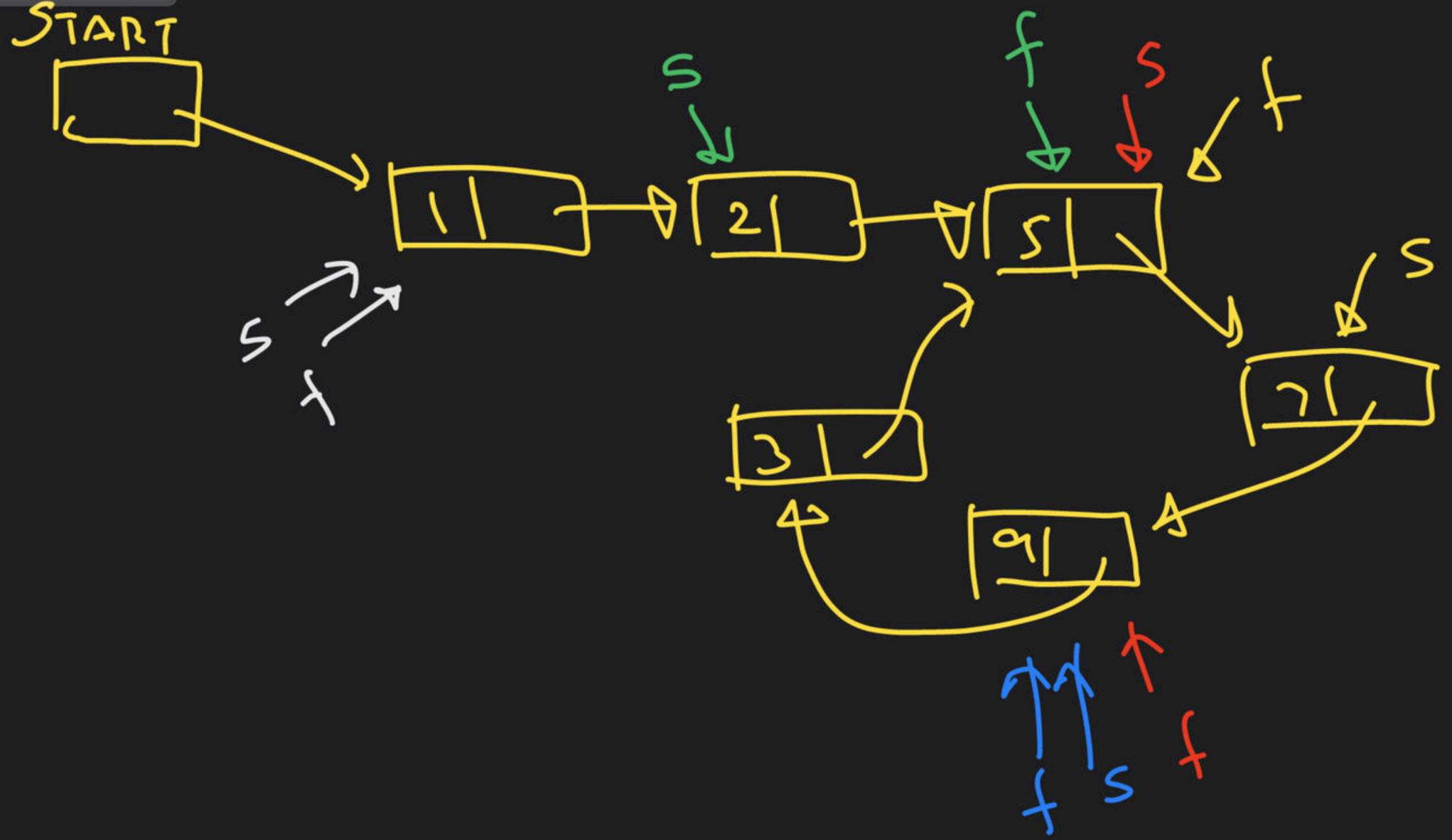
trees

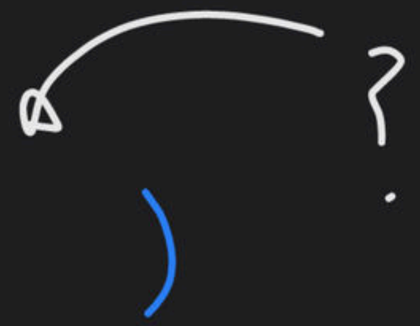
Elevate C++ x STL-er

Given a linked list, detect whether it contains a loop or not.



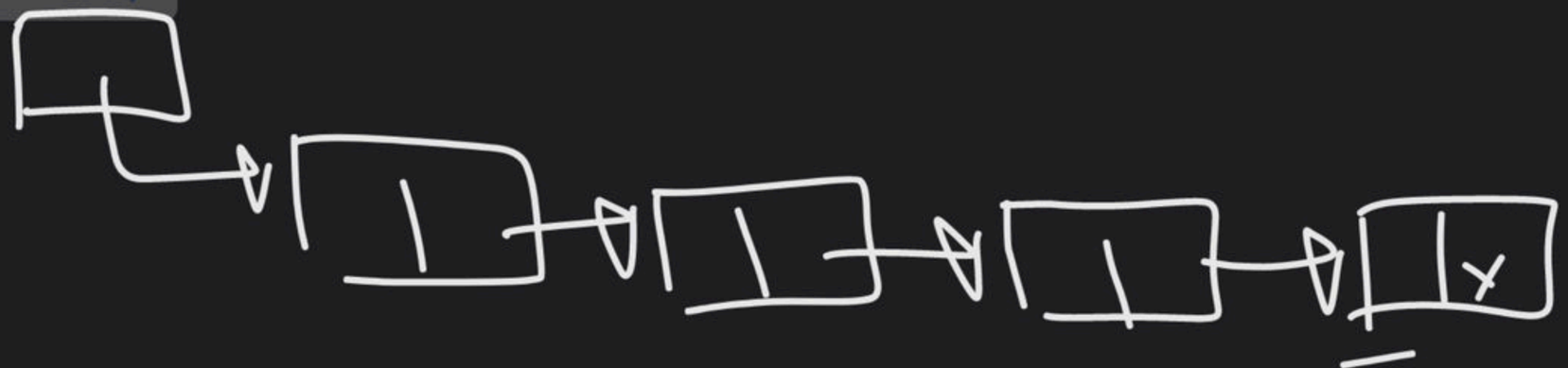

unacademy
Slow
fast
Pankaj
Jai
Pankaj
Jai
Pankaj


START
1
2
5
3
7
9
s
f
s
f
s
f
s
f
s
f



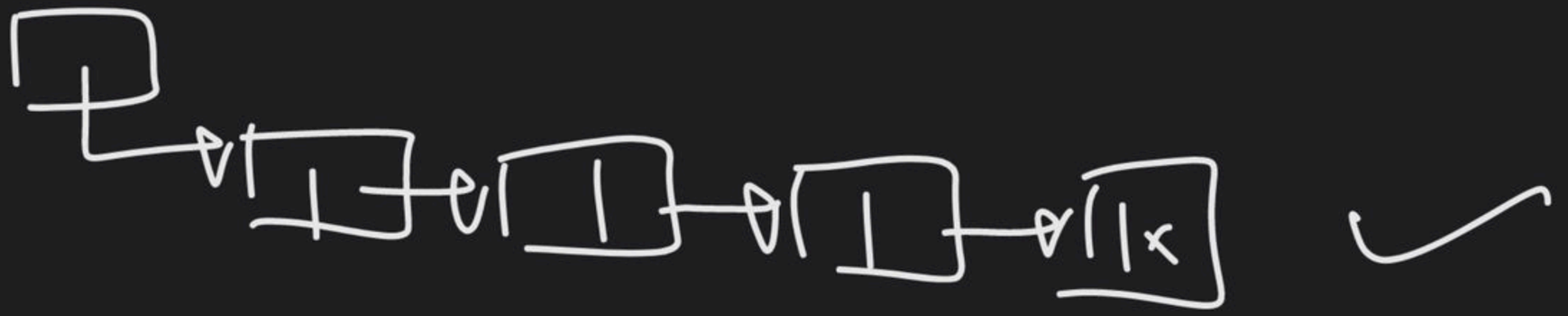
```
while (
    {
        s = s -> Next;
        f = f -> Next -> Next;
        if (s==f)
            return 1;
    }
```

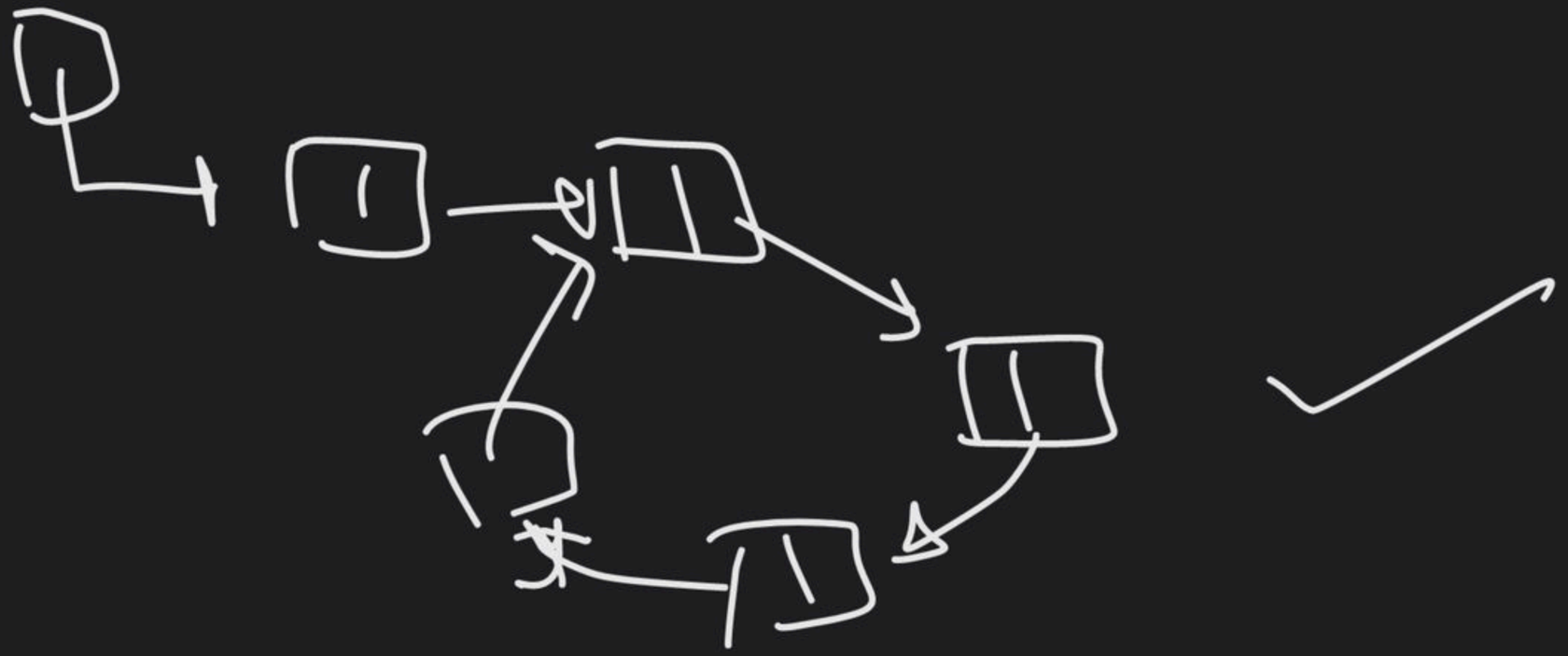
Cycle in L.L.

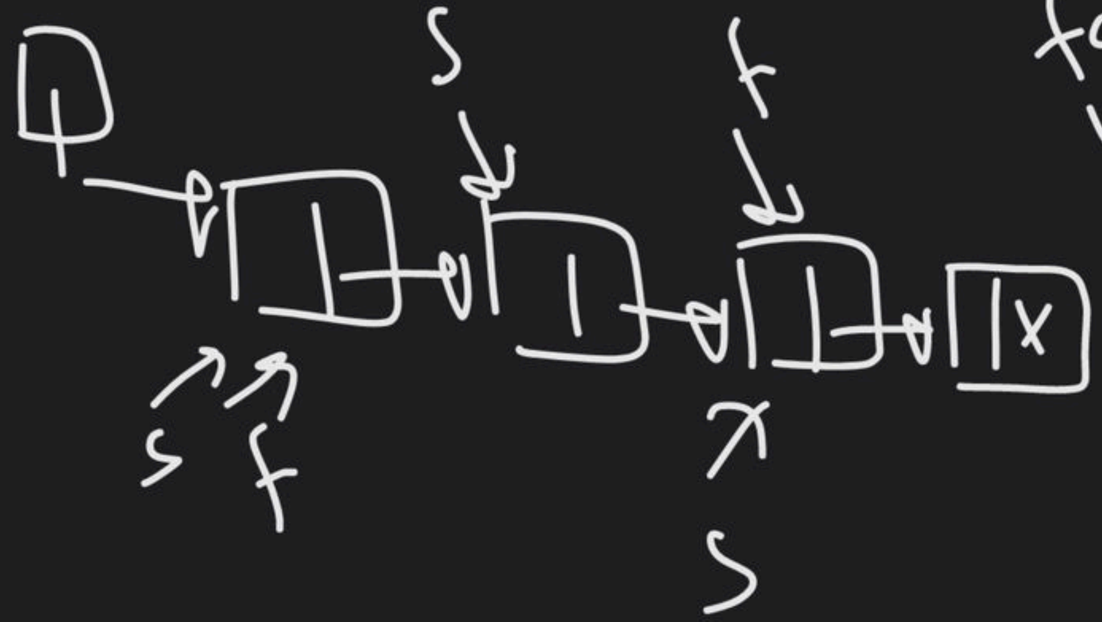


```
while ( fast != NULL && fast->Next != NULL )
    {
        s = s->Next;
        f = f->Next->Next;
        if (s == f) return 1;
    }
return 0;
```

Pinnacle







while( )

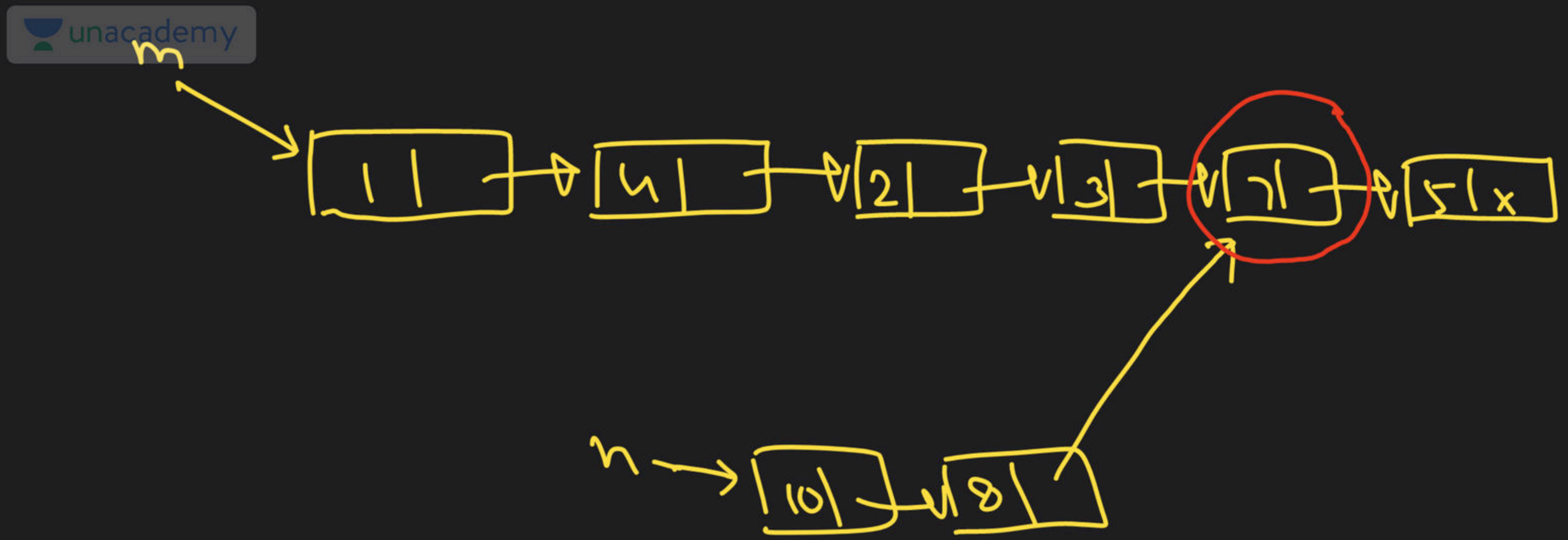
{

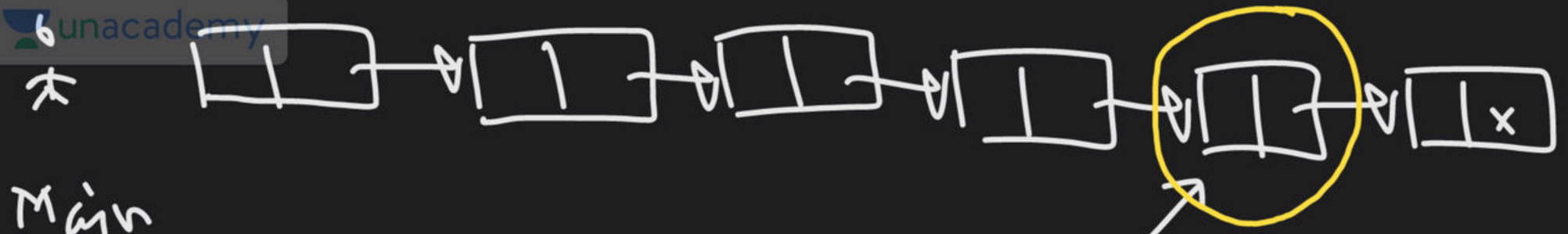
s = s → Next;

f = f → Next → Next

if(s == f) return 1;

}

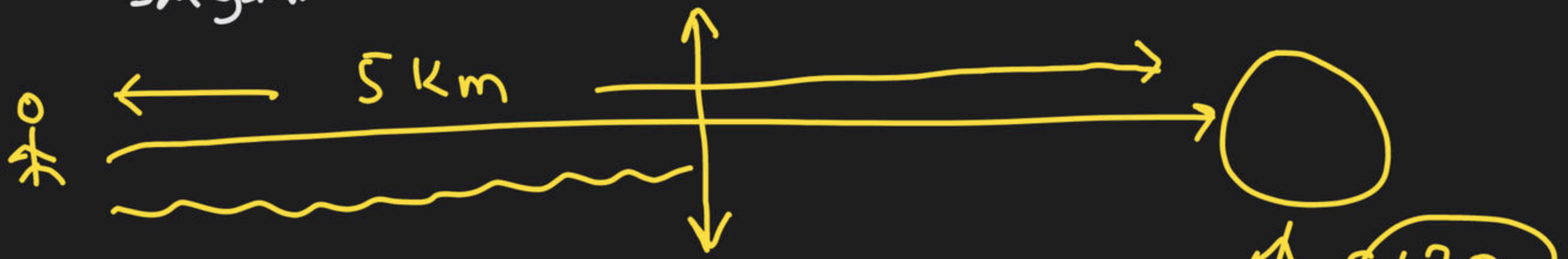

m
1
4
2
3
6
5
x
n
10
8



Main

Satyam

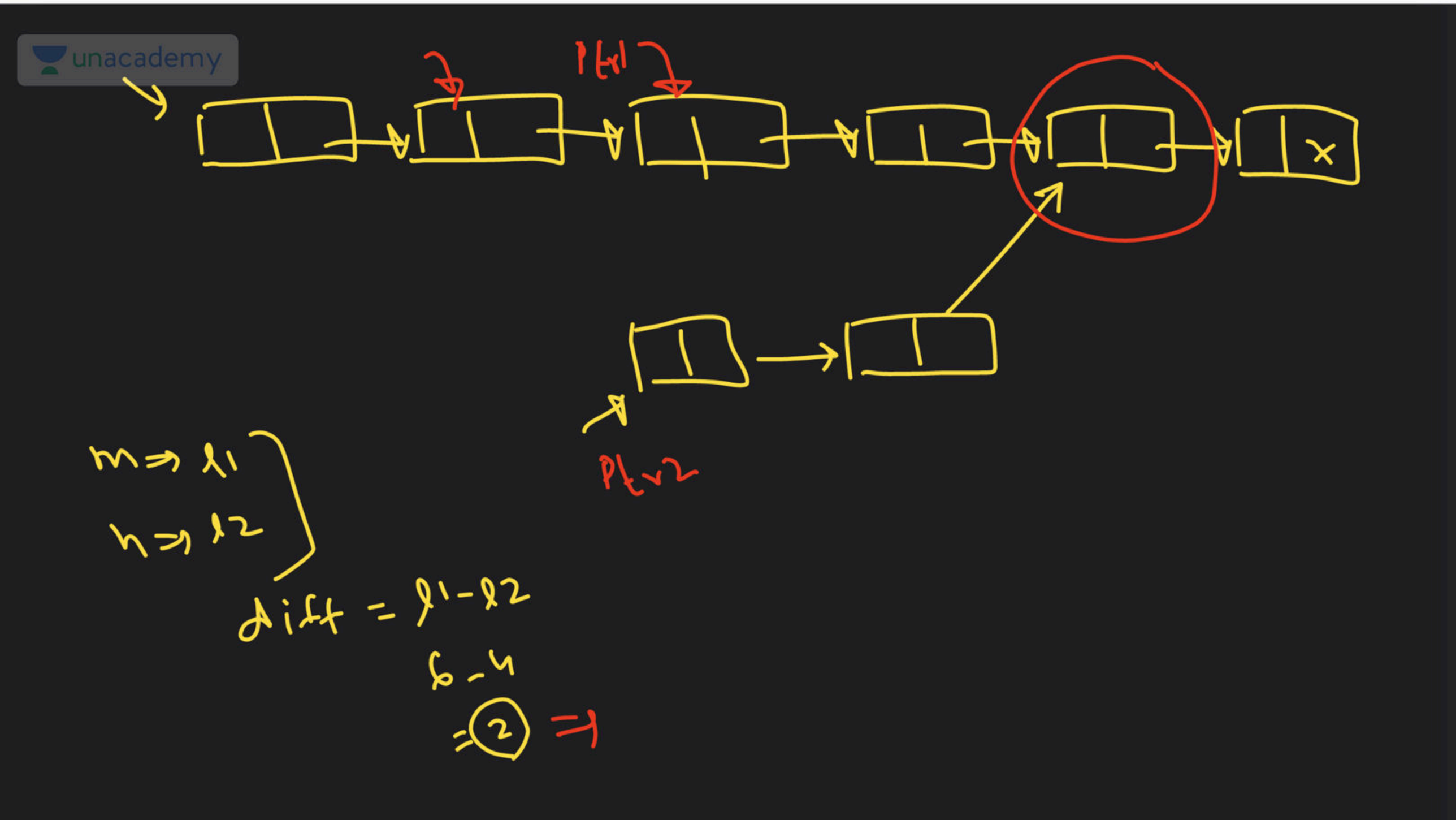
16 min



8:30

9:00

9:20

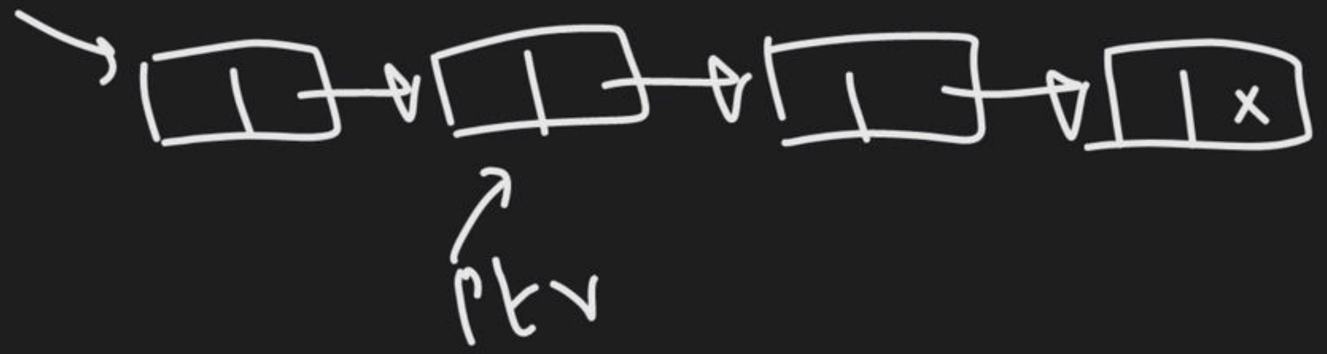

unacademy
x
m ⇒ l1
h ⇒ l2
diff = l1 - l2
6 - 4
= 2 ⇒ 1

```
struct Node *ptr1 = head1;
struct Node *ptr2 = head2;
int l1=0, l2=0;
while (ptr1 != NULL)
{
    l1++;
    ptr1 = ptr1->Next;
}
while(ptr2 != NULL { l2++;
                     ptr2=ptr2->Next;
}
ptr1=head1; ptr2 = head2;
int diff;
diff = abs(l1-l2);
if(l1>l2){
for(i=1;i<=diff;i++)
    ptr1 = ptr1->Next;
}
else{
}
```

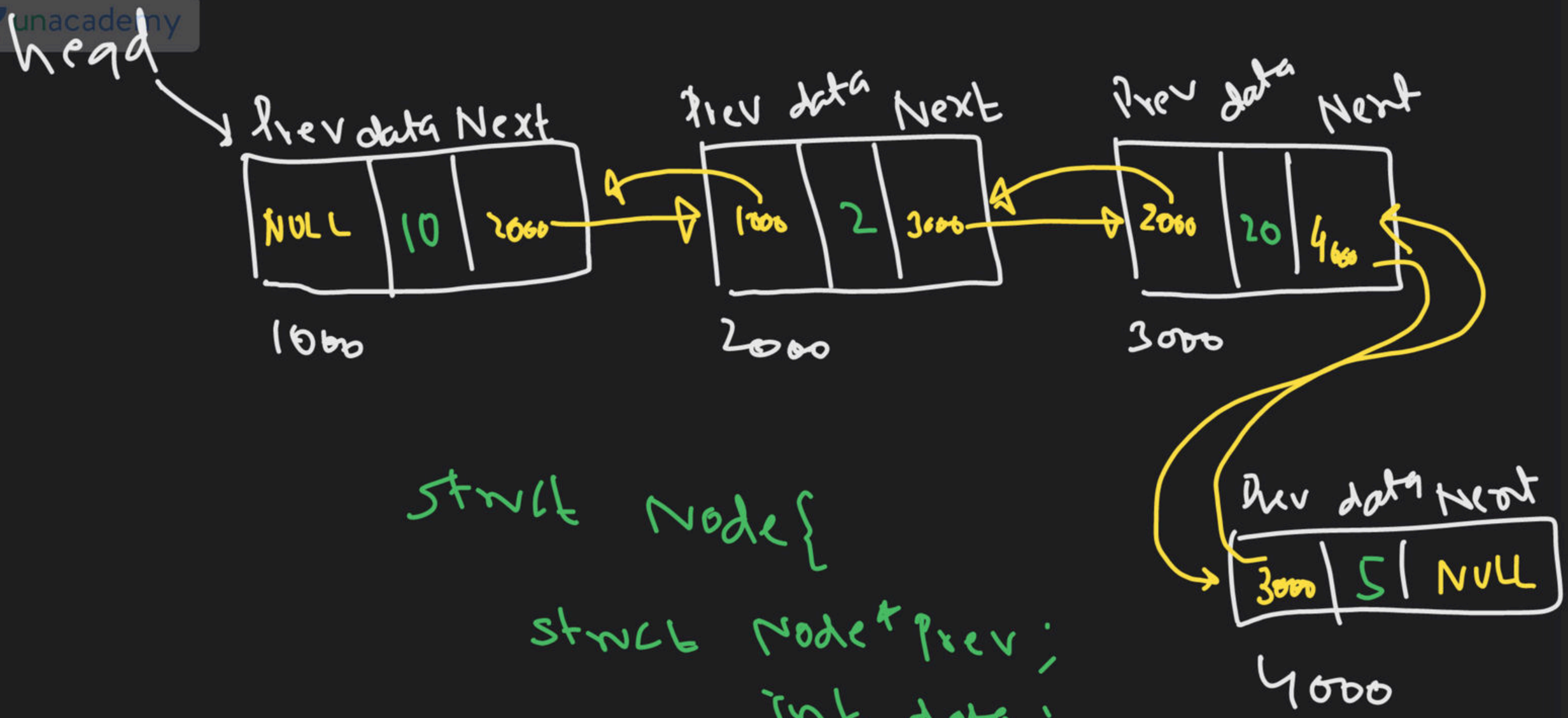
unacademy

Types of Linked List

① Singly L.L:



② Doubly Linkedlist: Node → Prev, Next, data



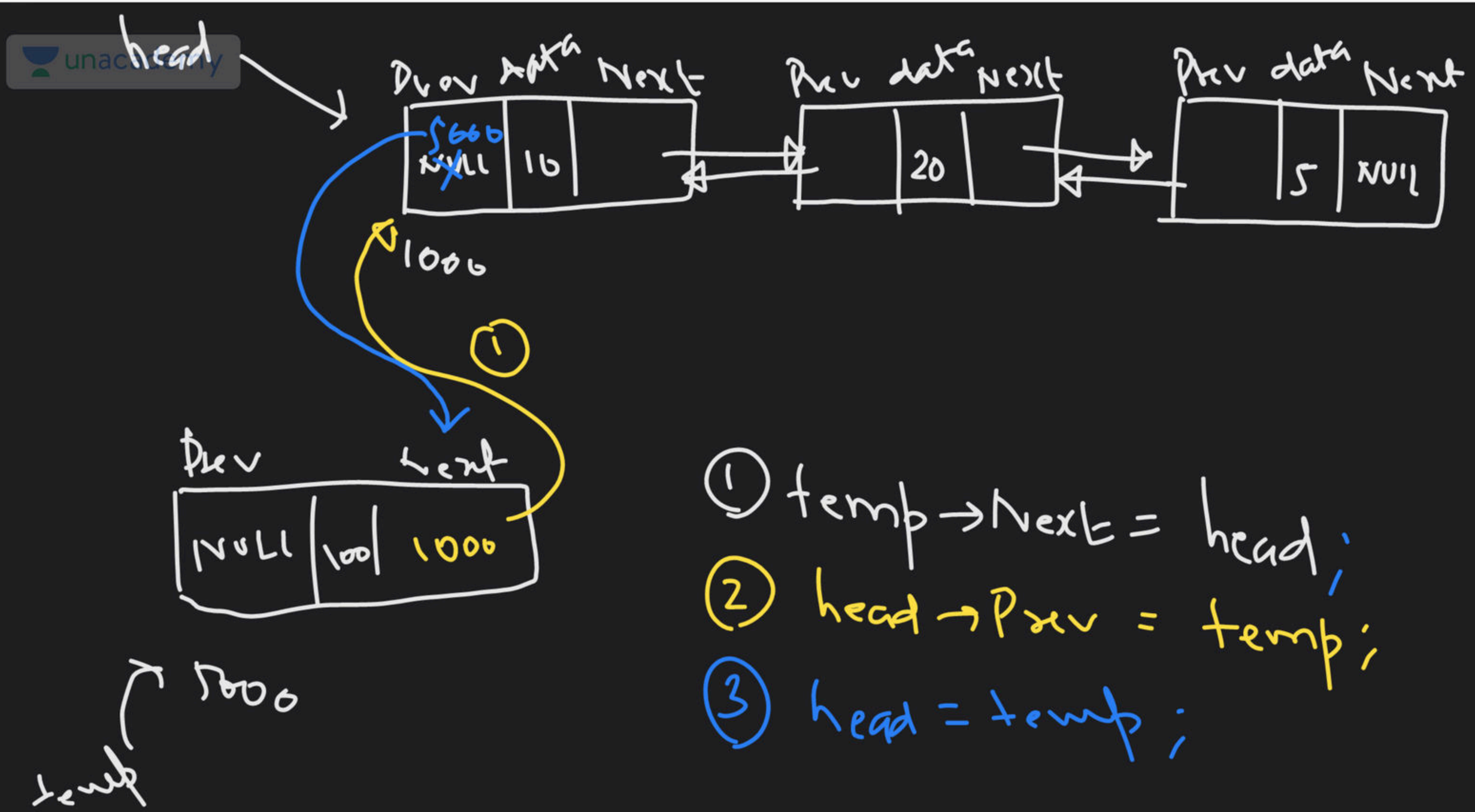

```
struct Node{
    struct Node* prev;
    int data;
    struct node * Next;
};
```

Insert - begin → create a Node

```
struct Node *temp;
temp = malloc(sizeof(struct Node));
temp->data = key;
temp->Prev = NULL;  (1st Node)
```

```
if (START == NULL) {
    temp->Next = NULL;
    START = temp;
    return;
}
```

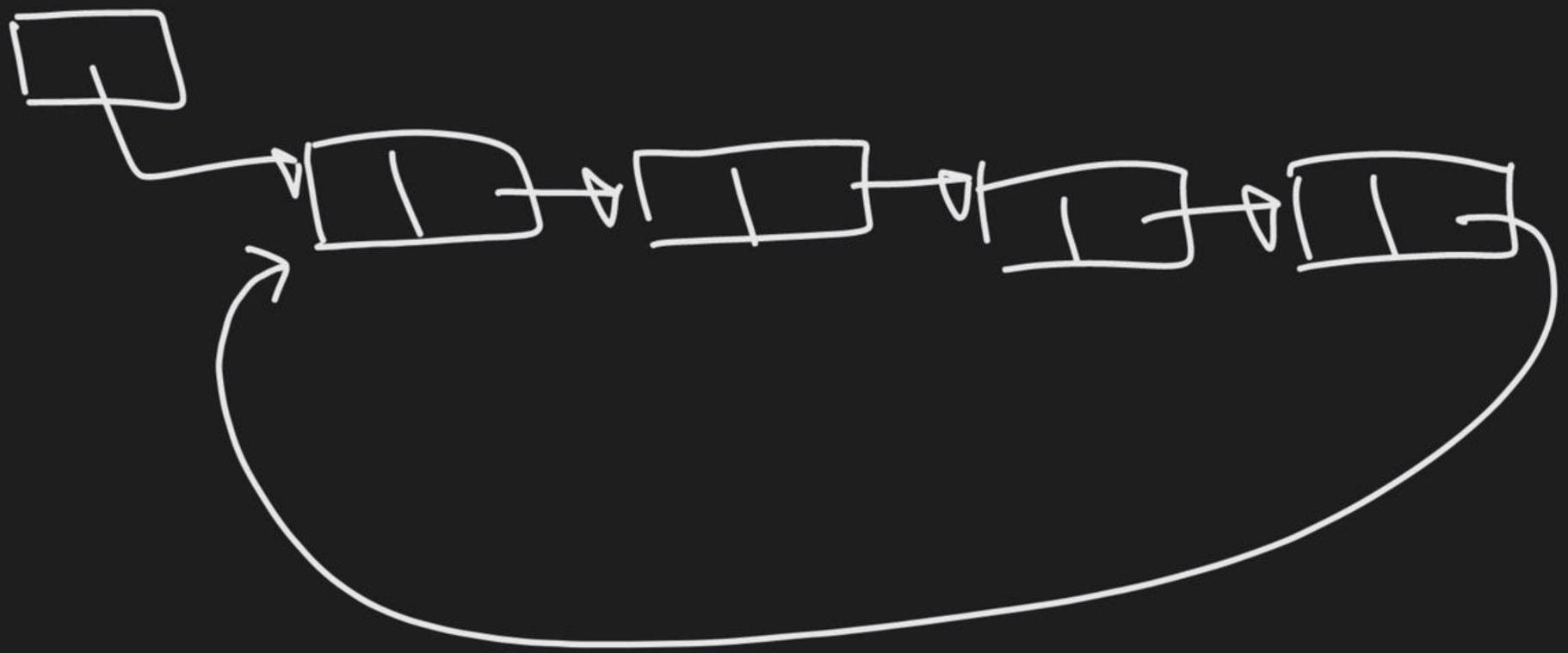
atleast 1 node is already present in L.L.

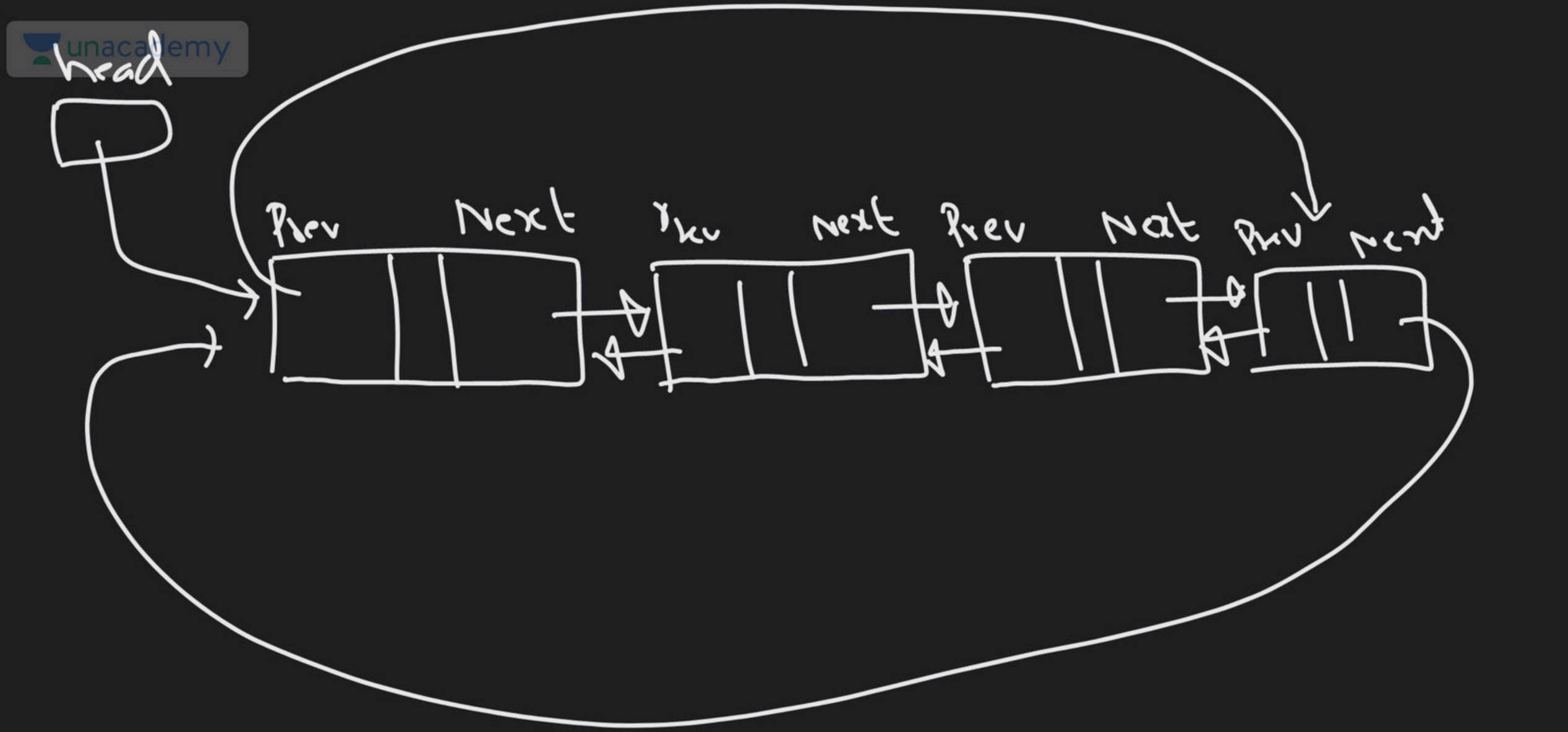


1. temp->Next = head;
2. head->Prev = temp;
3. head = temp;

unacademy

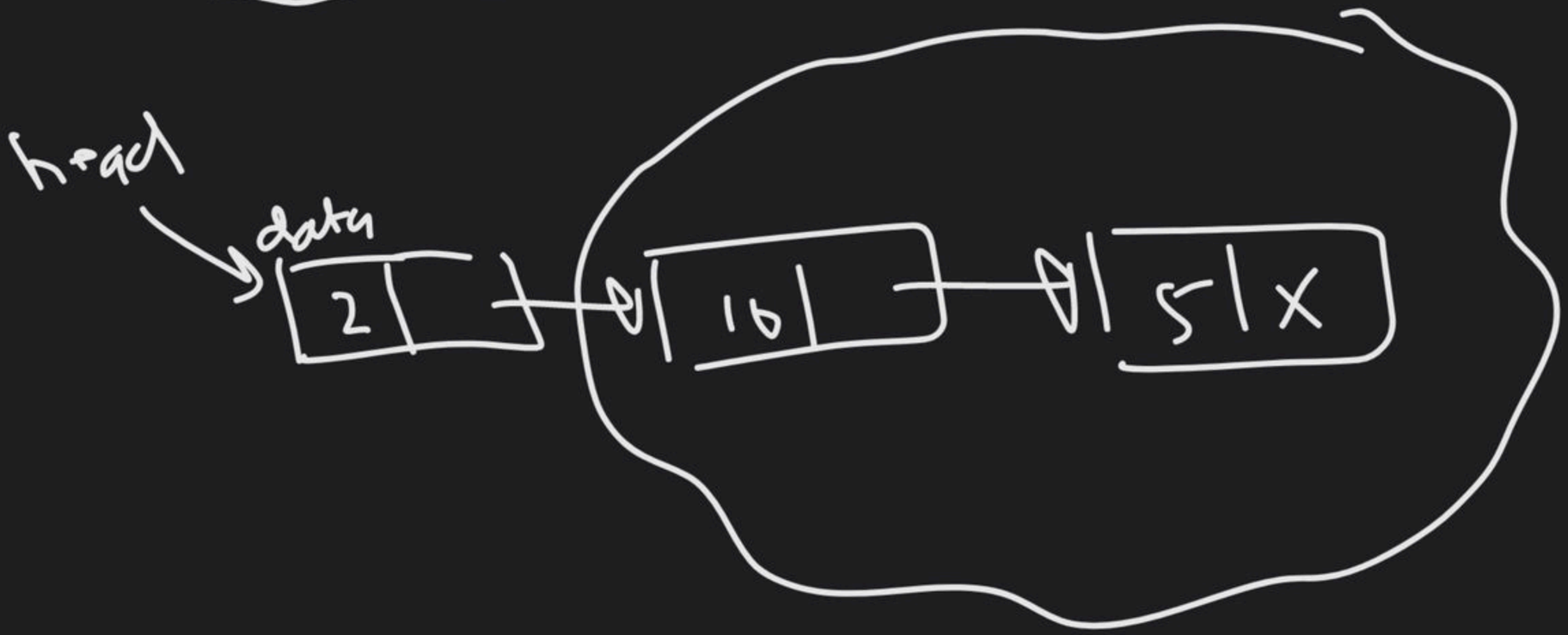
Circular Linked List




head
Prev
Next
Prev
Next
Prev
Nat
Prev
Next

header L.L

⇒ header node



Doubt?

Given a pointer to a node

LL

→ implement

Types of Linked List

unacademy

THANK YOU!

Here's to a cracking journey ahead!