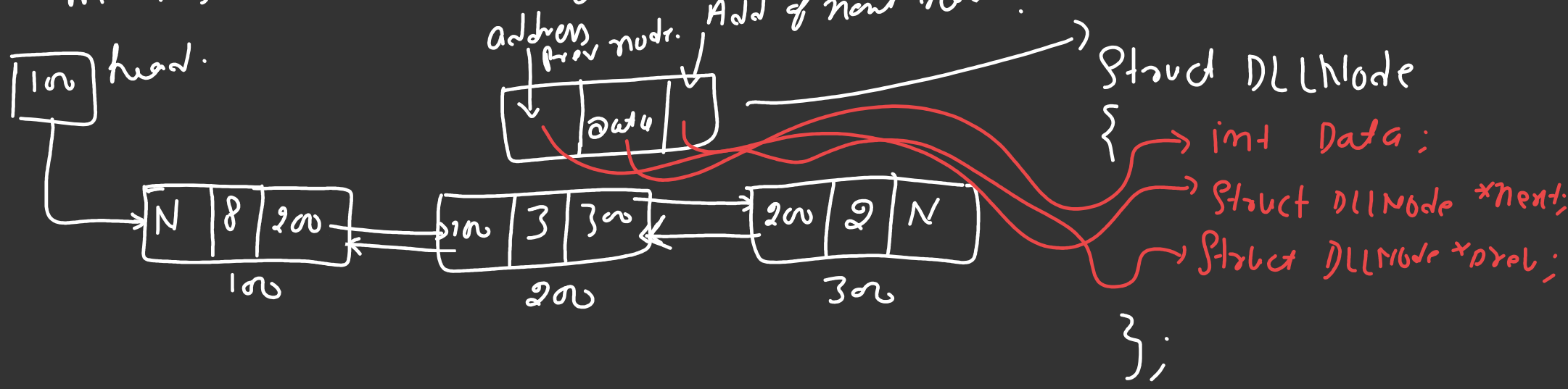
Date
02/01/23

§ Data Structures §

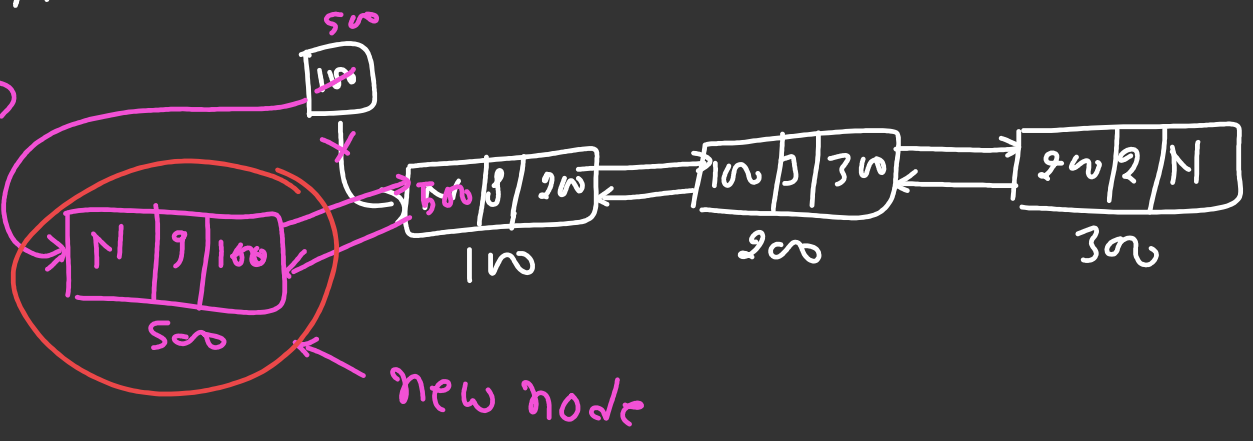
Linked Lists (Singly) ⟹ Insertion
(DLL) ⟹ Deletion

§ Doubly linked lists: It is also called two way linked lists.
On this we can navigate in both directions.



Insertion in DLL

- At beginning
- At end.
- At intermidate

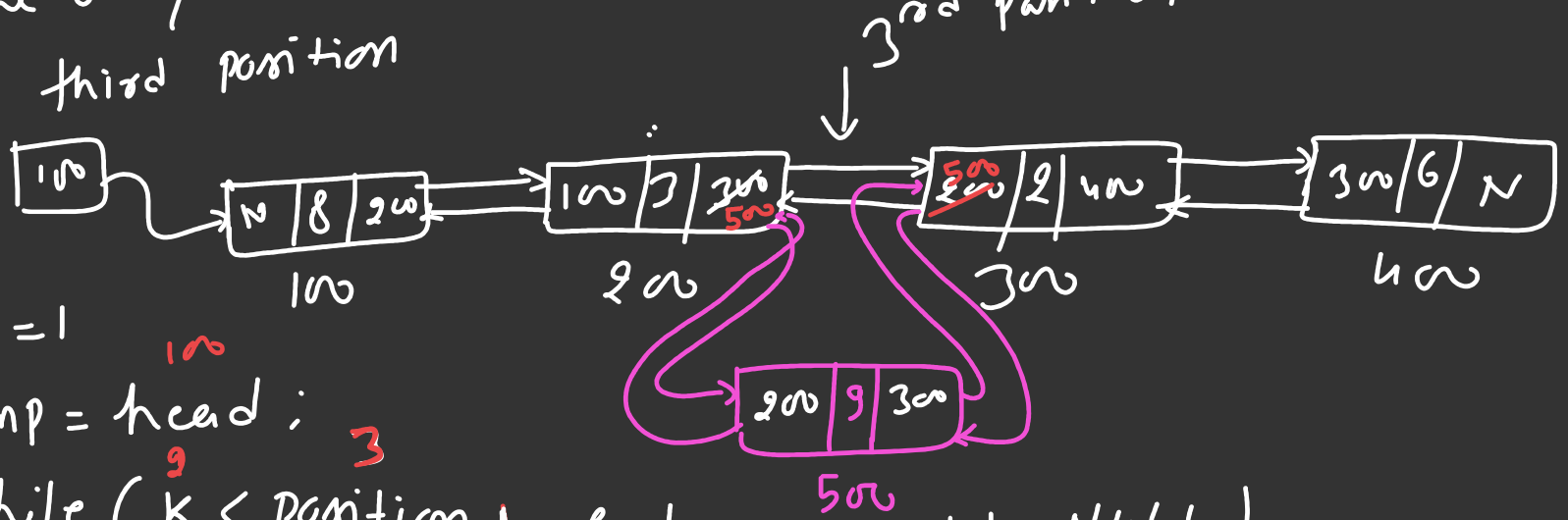


Create new_node.

```
new_node → Data = Data
new_node → next = head
new_node → prev = NULL
head → prev = new_node;
head = new_node;
```

At the End / Intermidiate location -

At third position

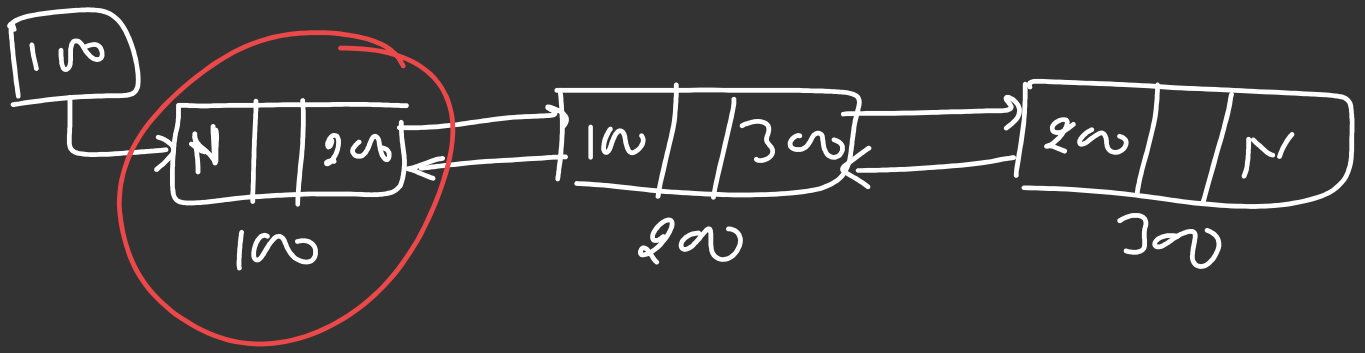



```
k = 1
temp = head;
while ( k < position-1 && temp->next != NULL )
{
    temp = temp->next;
    k++;
}
newnode->next = temp->next;
newnode->Prev = temp;
temp->next = newnode;
temp->next->prev = newnode;
```

§ Deletion in DLL:

At beginning

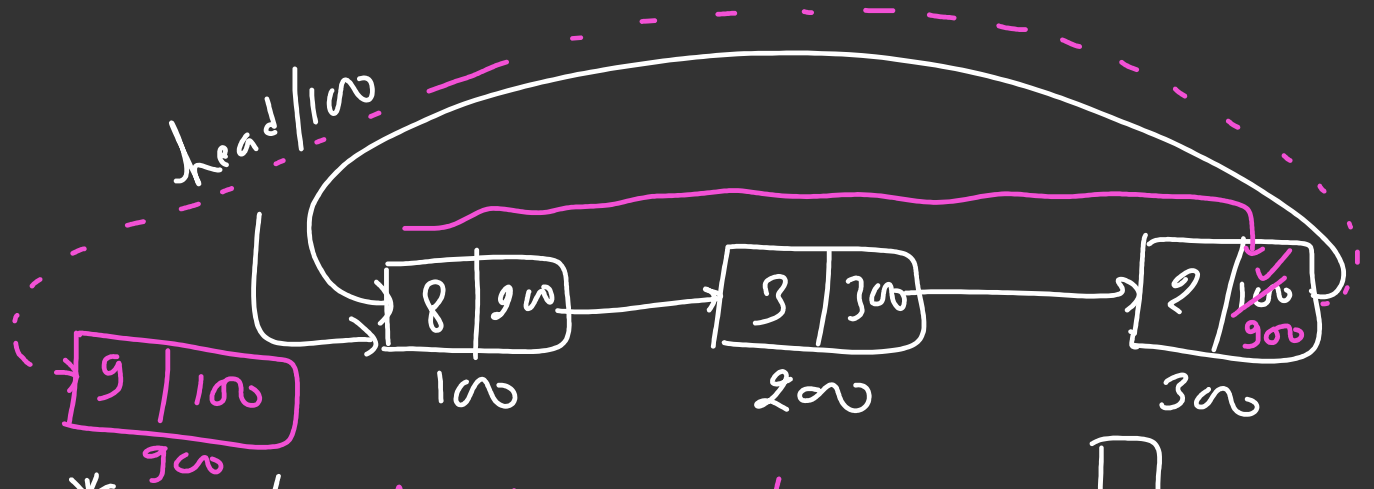
At End / Intermidiate ✓



§ Circular linked lists:

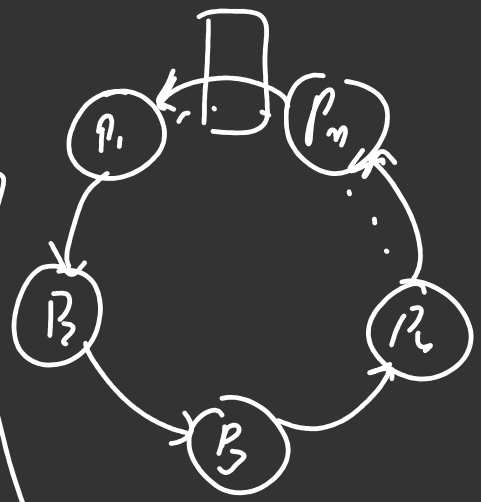
+ elete

```
Struct CLLNode
{
    int data;
    Struct CLLNode *next;
};
```



head = newnode.

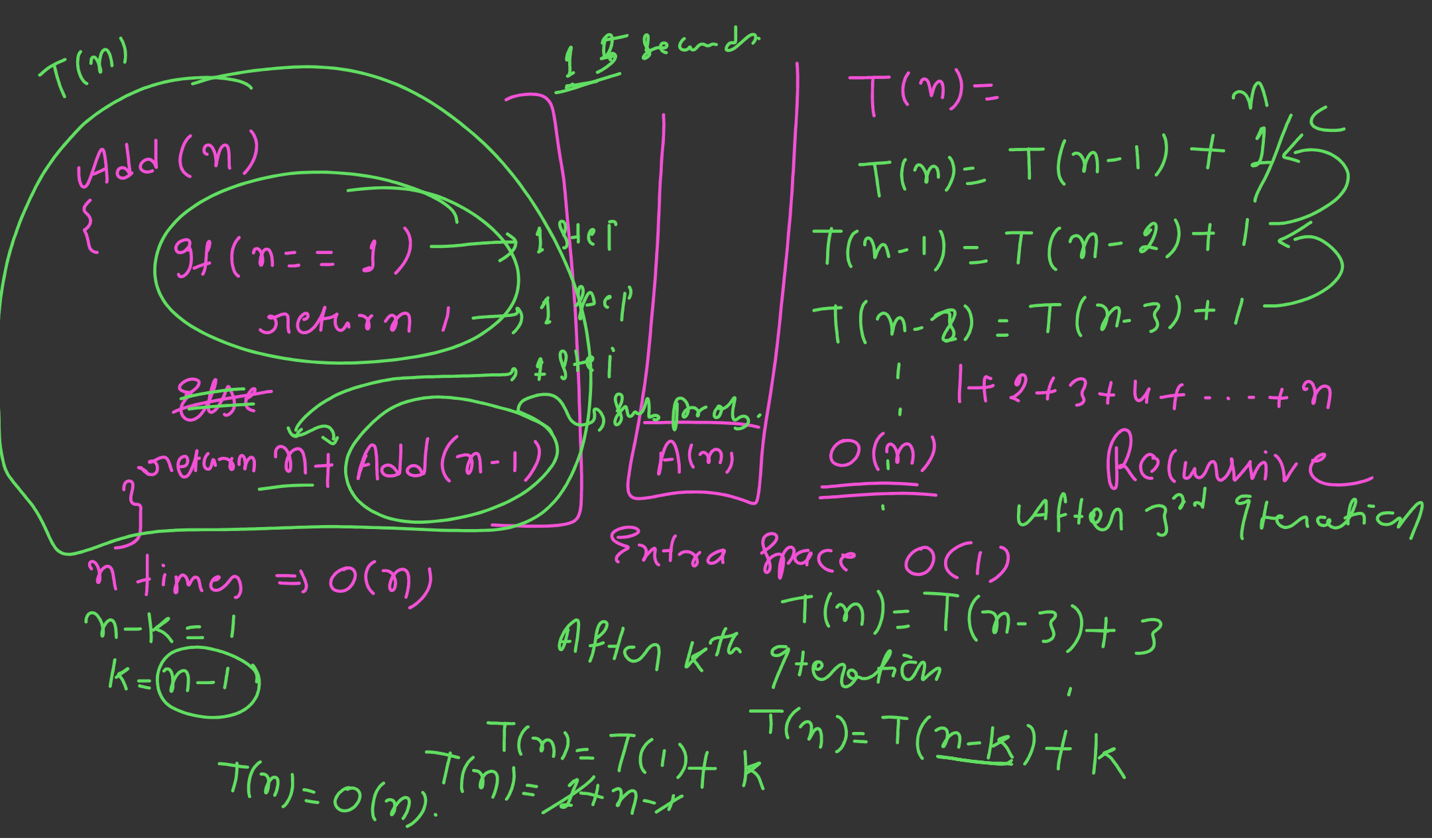
```
Circular linked list ( struct Node *head )
{
```



```
temp = head
newnode -> data = data

while ( temp -> next != head )
{
    temp = temp -> next
}

newnode -> next = head
temp -> next = newnode;
head = newnode
```


T(n)
Add(n)
{
if(n==1)
return 1
else
return n + Add(n-1)
}
1 second
1 step
1 step
+ step
Sub prob.
A(n)
n times ⇒ O(n)
n-K = 1
K = n-1
T(n) = O(n).
T(n) = T(n-1) + 1
T(n-1) = T(n-2) + 1
T(n-2) = T(n-3) + 1
1+2+3+4+....+n
O(n)
Extra space O(1)
After kth iteration
T(n) = T(n-3) + 3
T(n) = T(n-k) + k
T(n) = T(1) + k
T(n) = 1 + n - 1
n
1/c
Recursive
After 3rd iteration

$$T(n) = T(n-1) + n$$

$T(n) = T(n-1) + n$

$T(n-1) = T(n-2) + (n-1)$

$T(n-2) = T(n-3) + (n-2)$

$\vdots$

$\vdots$

$T(n) = T(n-3) + (n-2) + (n-1) + n$

After k^{th} iteration

$T(n) = T(n-k) + (n-(k-1)) + \cdots + (n-2) + (n-1) + n$

$n-k = 1$

$k = n-1$

$T(n) = T(1) + \cdots + (n-(n-1-1)) + \cdots + (n-2) + (n-1) + n$

$= 1 + 2 + 3 + \cdots + (n-2) + (n-1) + n$

```
A(n)
{
    if(  )
        return
    return A(n-1) + f(n)
}
```

$O(n^2)$

$f(n)$ ↑ n times