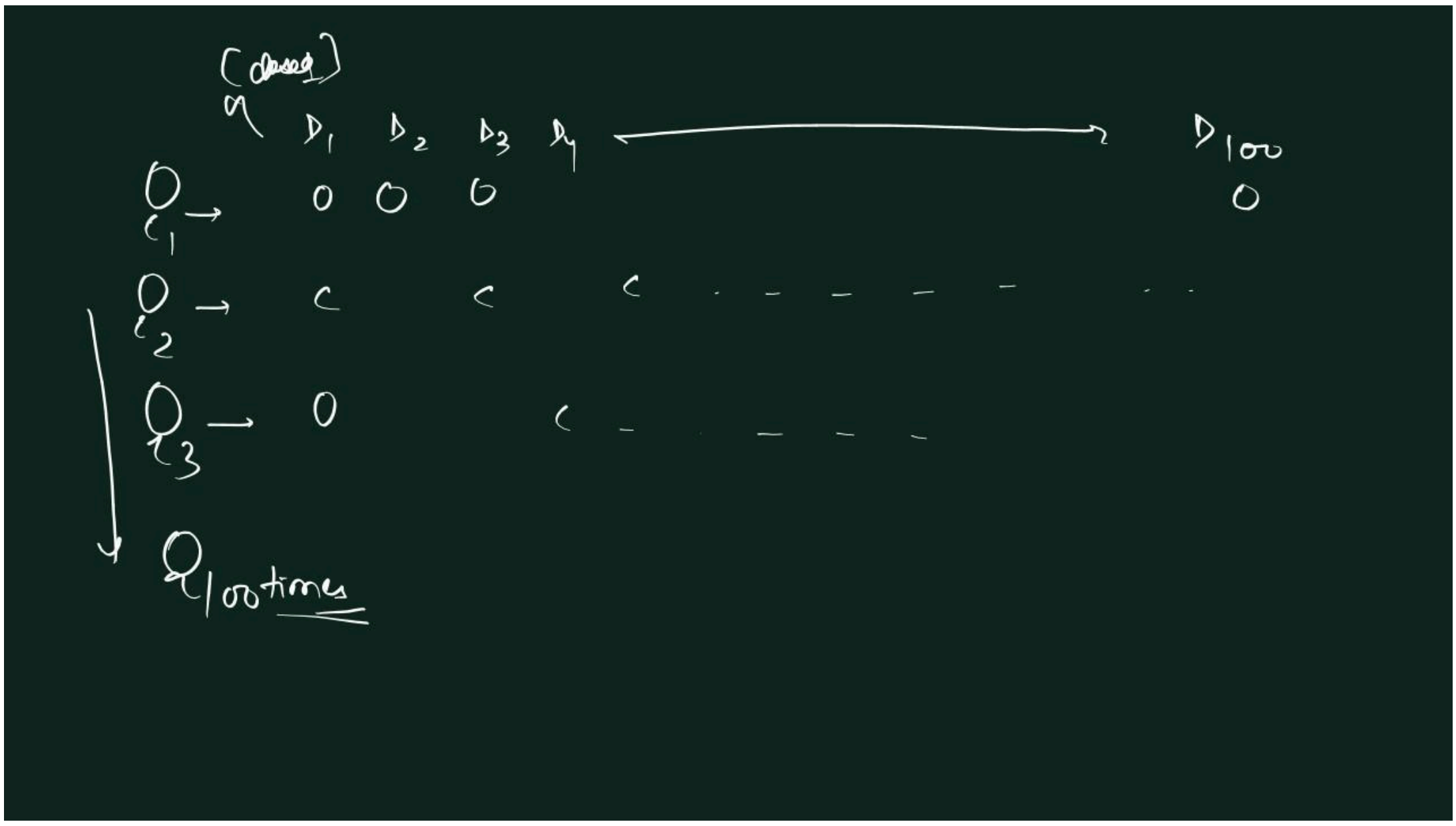
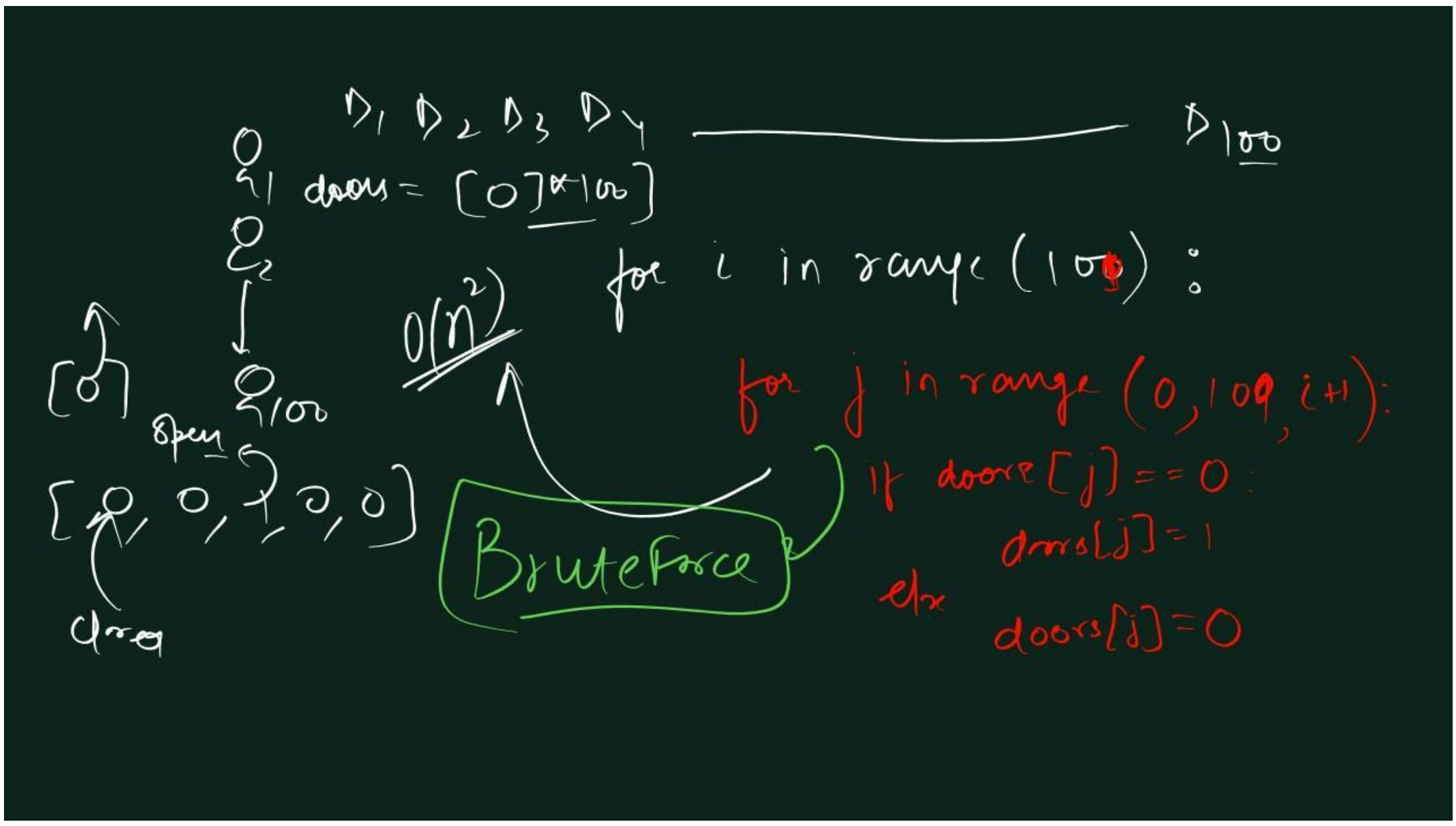
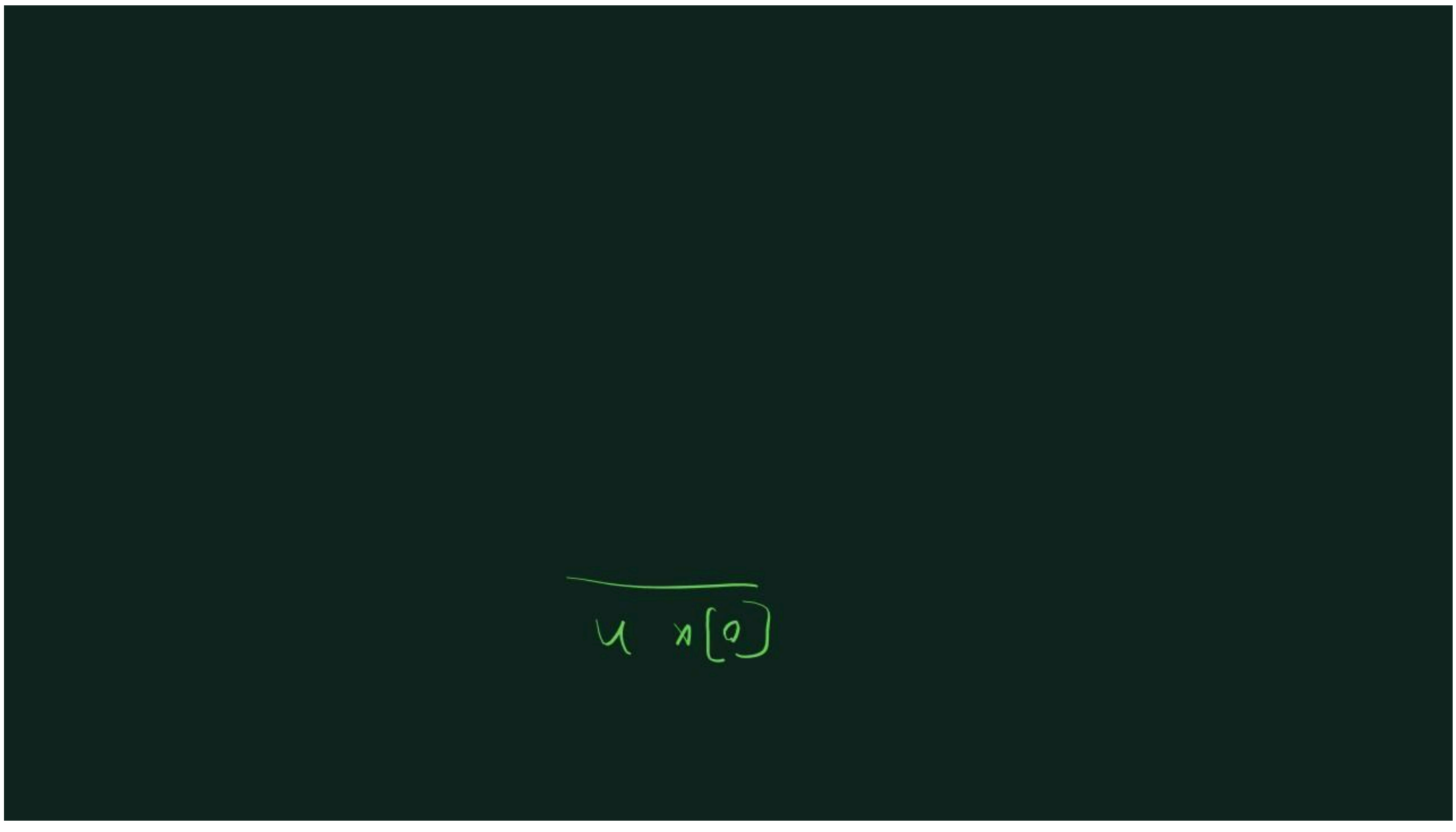

[closed]
Q
D1 D2 D3 D4
D100
C1
C2
C3
Q 100 times


D1 D2 D3 D4 D100
doors = [0]*100
for i in range(100):
for j in range(0,100,i+1):
if doors[j]==0:
doors[j]=1
else
doors[j]=0
O(n²)
BruteForce
[0]
open
[0,0,1,0,0]
door

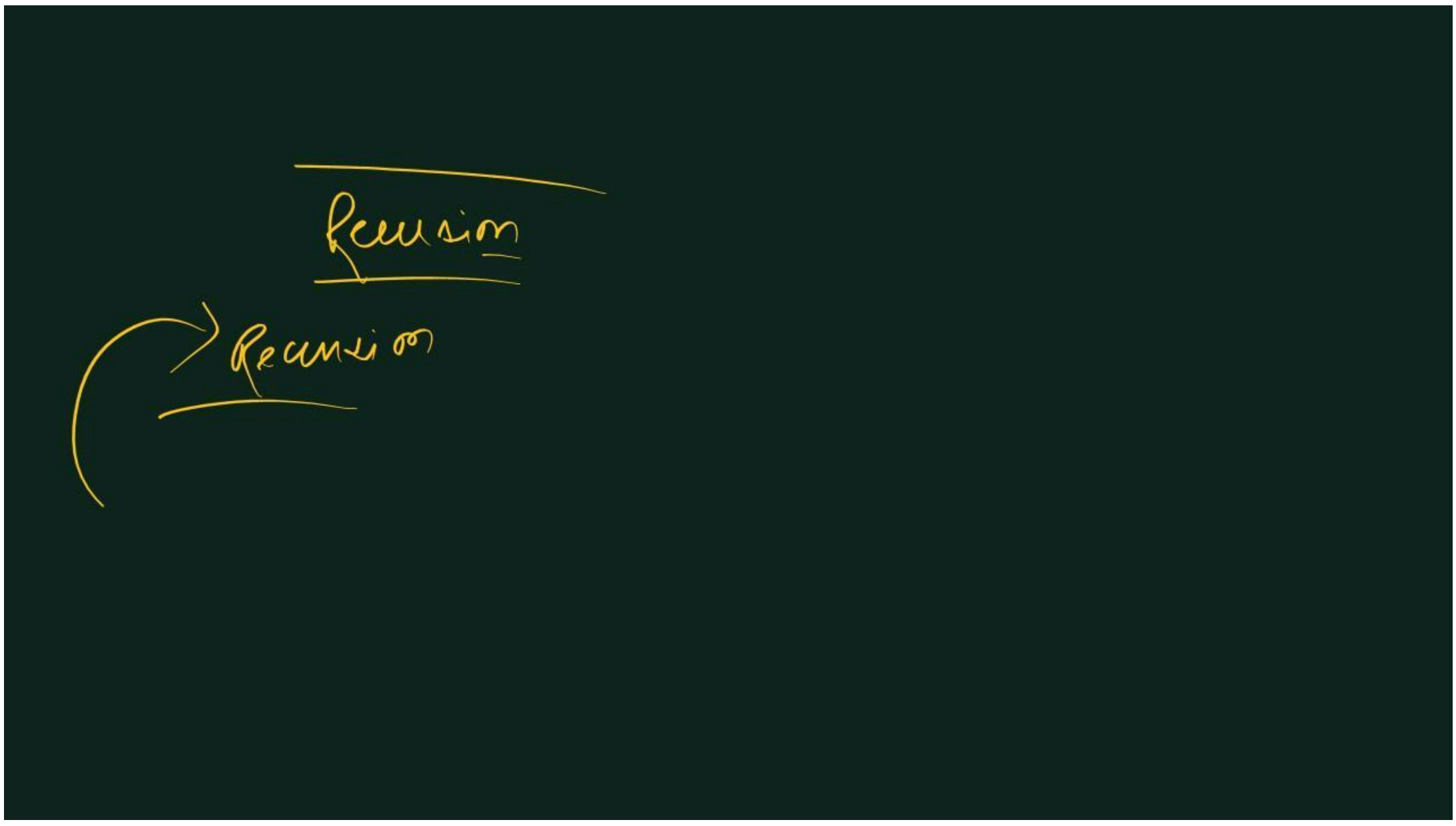

u x[0]

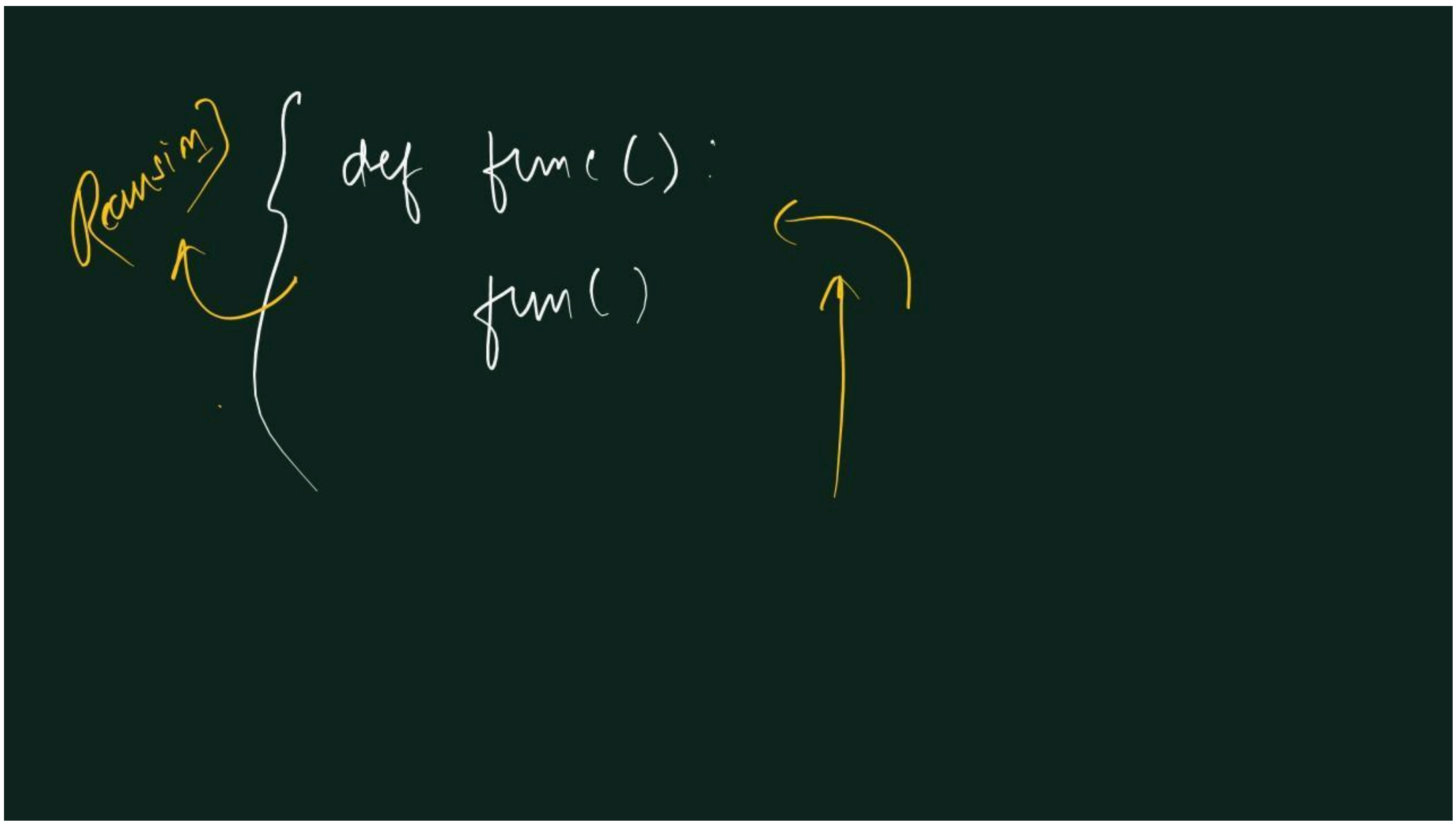
→ (fn) list = [1, 2, 3] → fn { print all the elements }

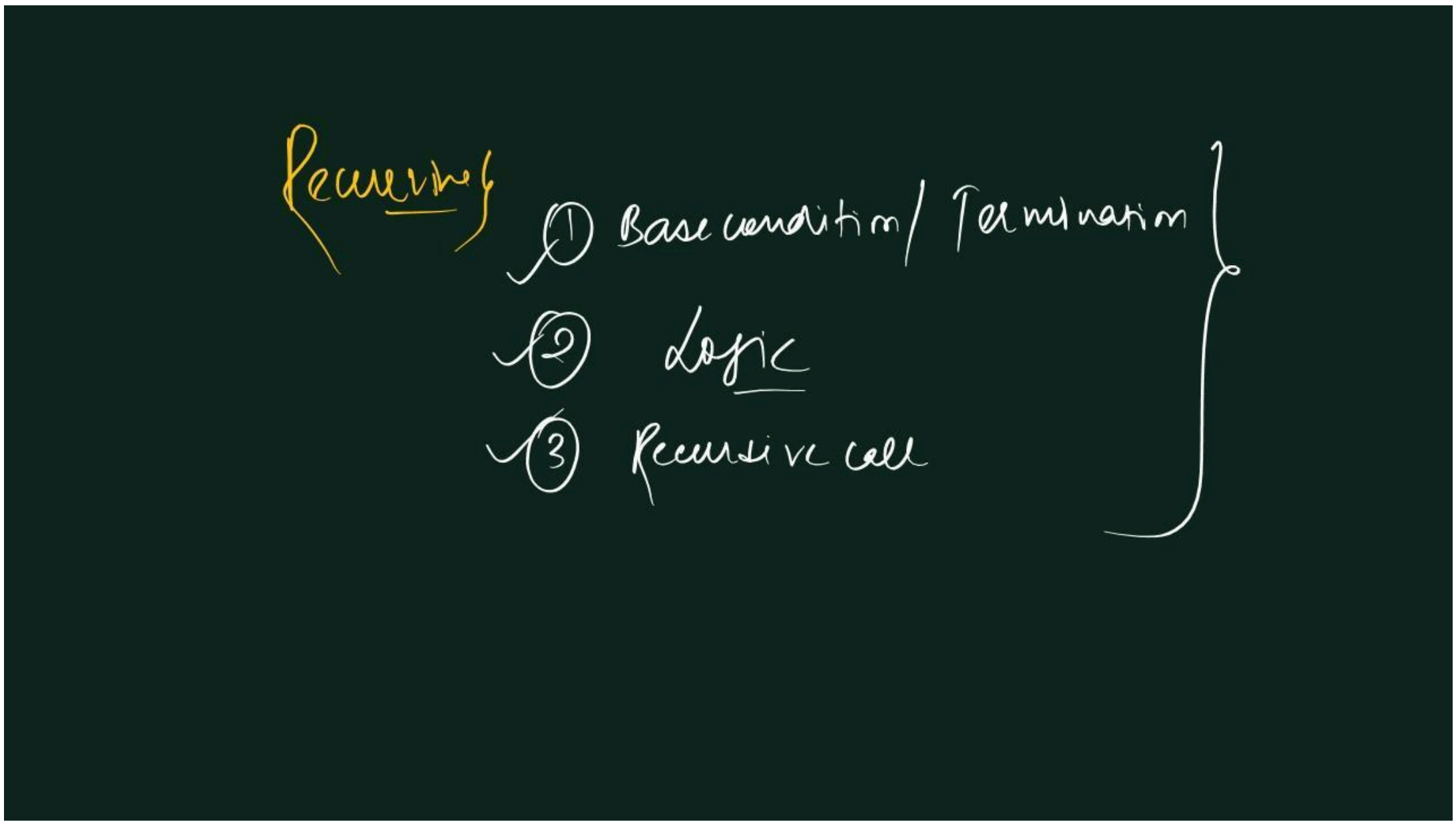
How ?

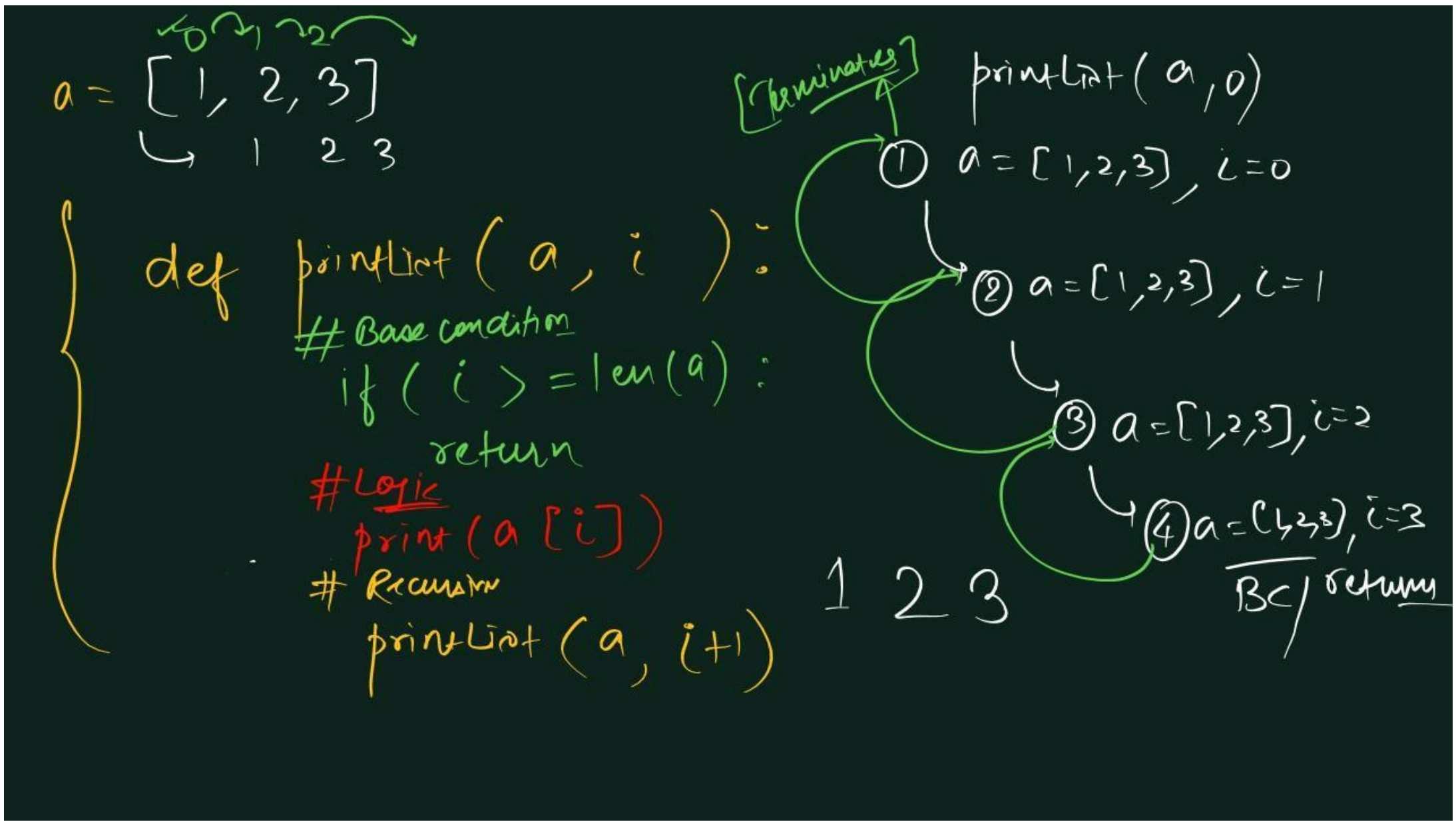
90% { 10%

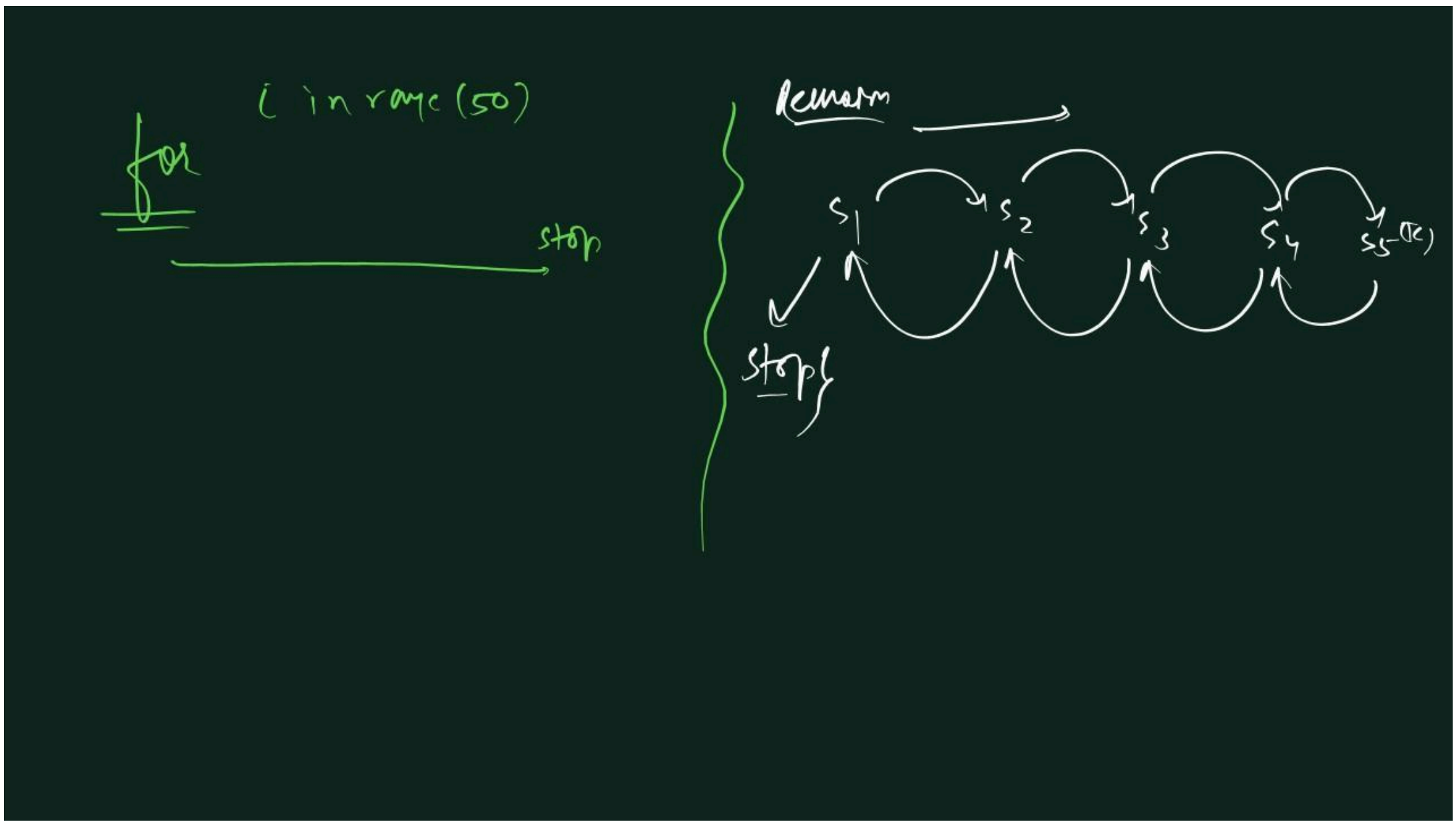
```
def printList ( list) :
    for i in list:
        print (i)
```

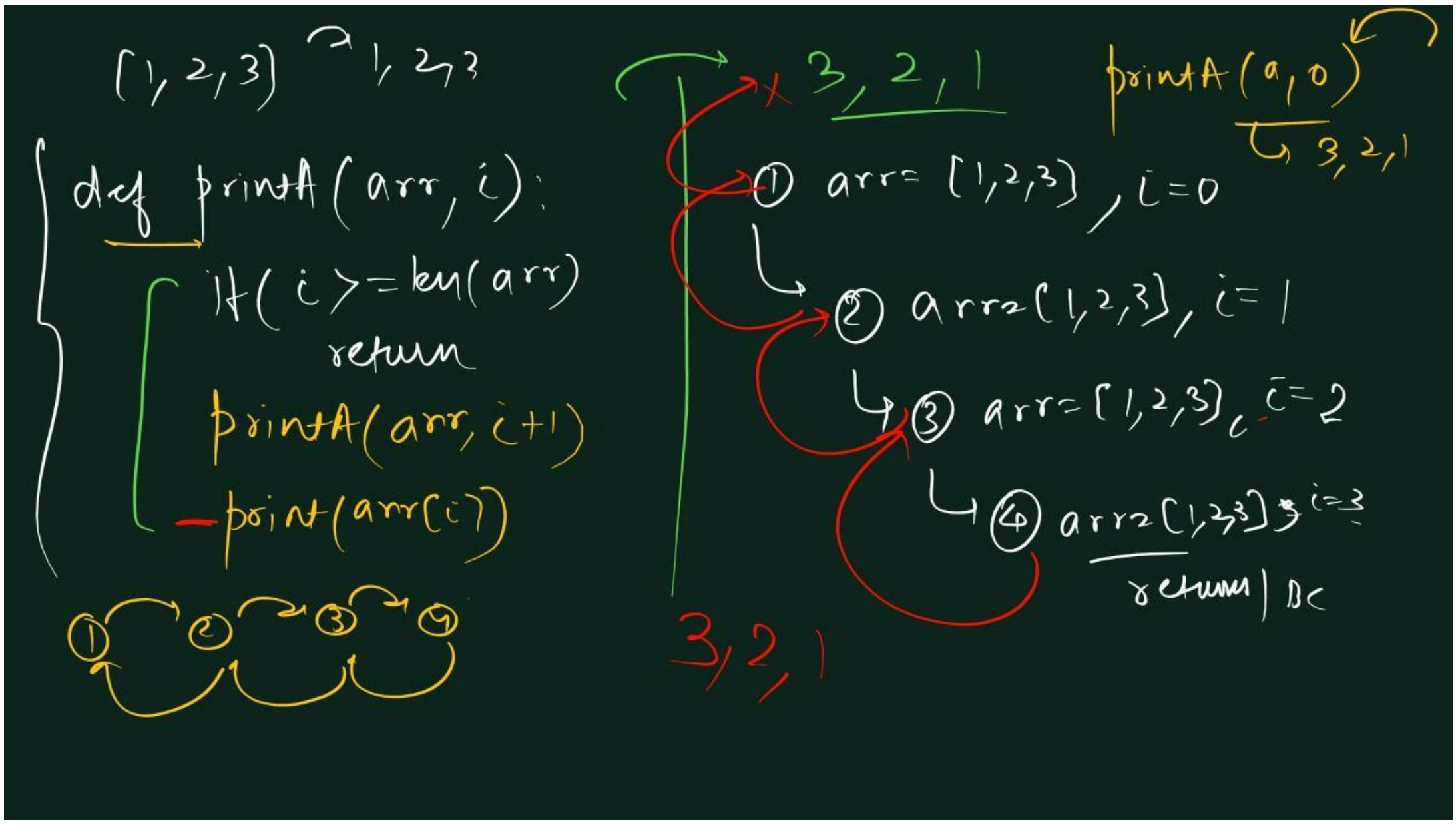

Recursion
Recursion

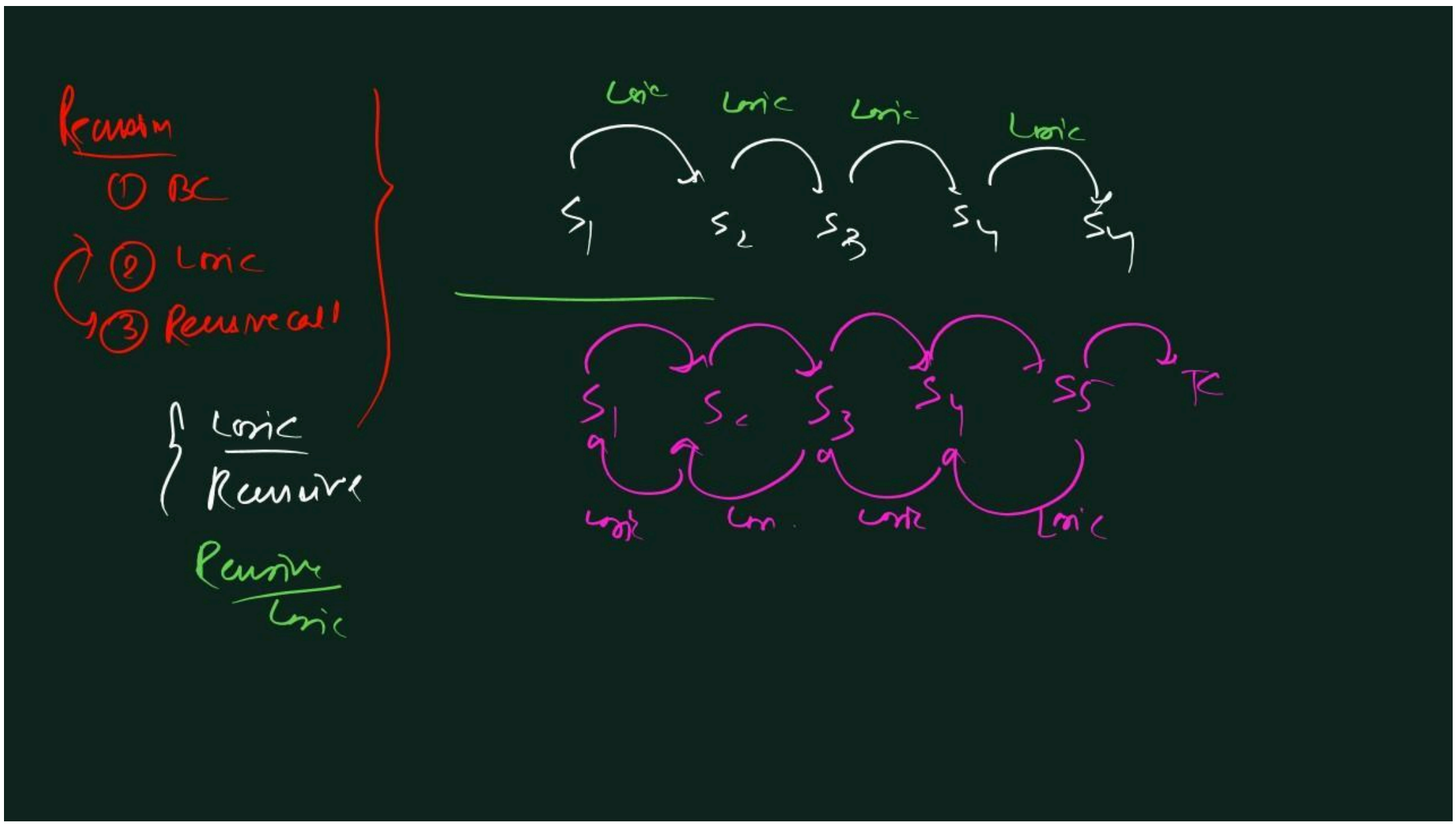

Recursion
def func():
func()


Recursion
1 Base condition / Termination
2 Logic
3 Recursive call


a = [1, 2, 3]
0 1 2
1 2 3
def printList(a, i):
# Base condition
if (i >= len(a)):
return
#Logic
print(a[i])
# Recursion
printList(a, i+1)
[terminates]
printList(a, 0)
① a=[1,2,3], i=0
② a=[1,2,3], i=1
③ a=[1,2,3], i=2
④ a=[1,2,3], i=3
BC return
1 2 3


for
i in range(50)
stop
Recursion
S1
S2
S3
S4
S5-(TC)
Stop


[1, 2, 3]
1, 2, 3
def printA(arr, i):
if(i >= len(arr)
return
printA(arr, i+1)
print(arr[i])
1 2 3 4
3, 2, 1
3, 2, 1
printA(a, 0)
3, 2, 1
① arr=[1,2,3], i=0
② arr=[1,2,3], i=1
③ arr=[1,2,3], i=2
④ arr=[1,2,3], i=3
return BC

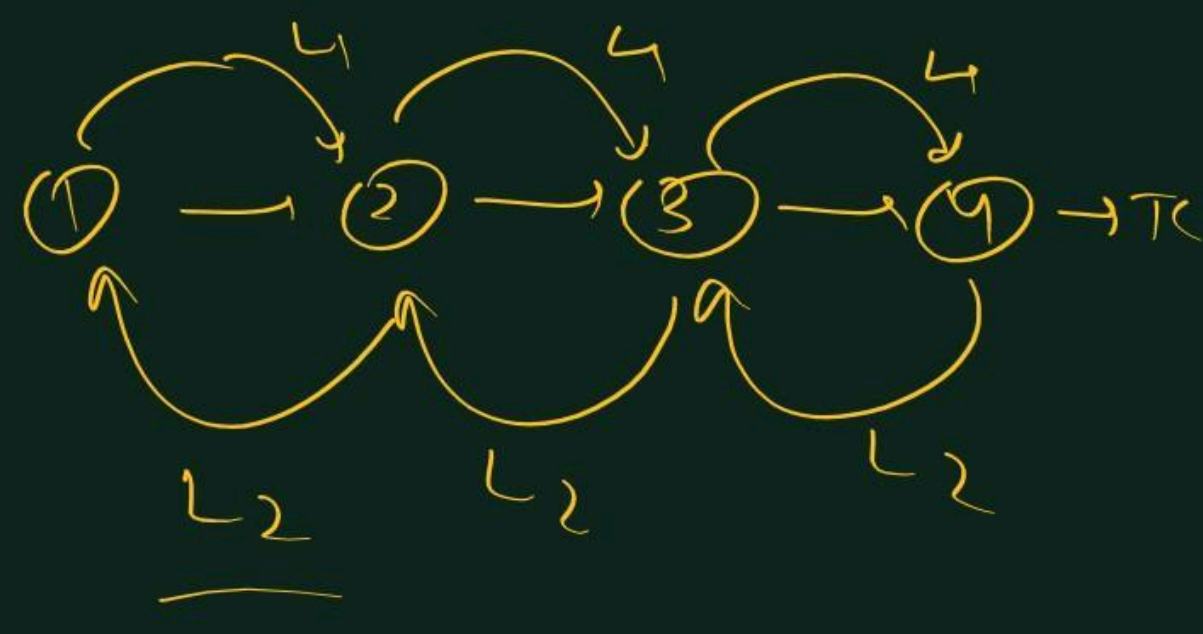

Recursion
① BC
② Logic
③ Recursive call
Logic
Recursive
Recursive
Logic
Logic
Logic
Logic
Logic
S1
S2
S3
S4
S4
S1
S2
S3
S4
S5
TC
Logic
Logic
Logic
Logic

① Base

Logic

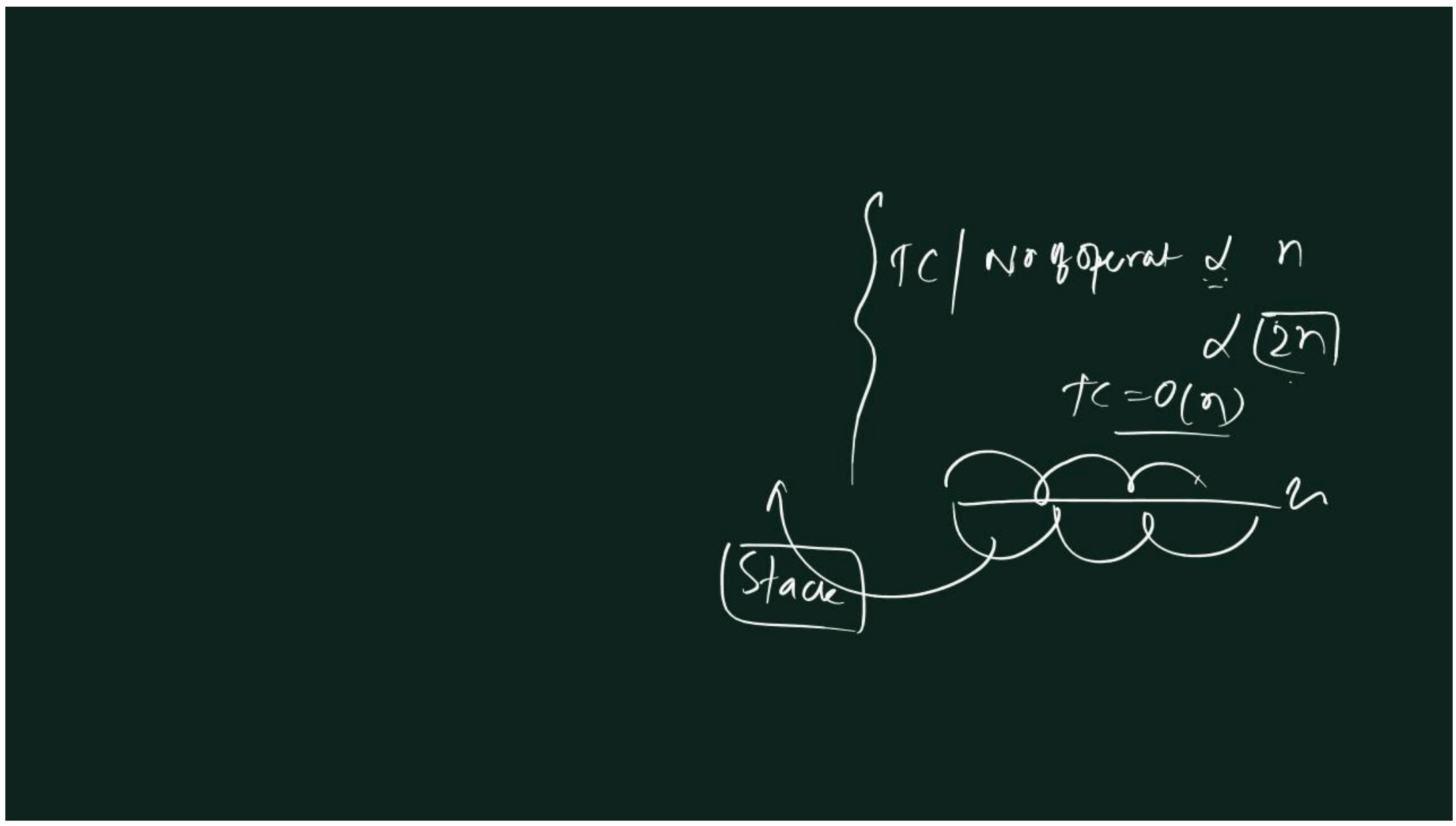
Recursion call

Logic

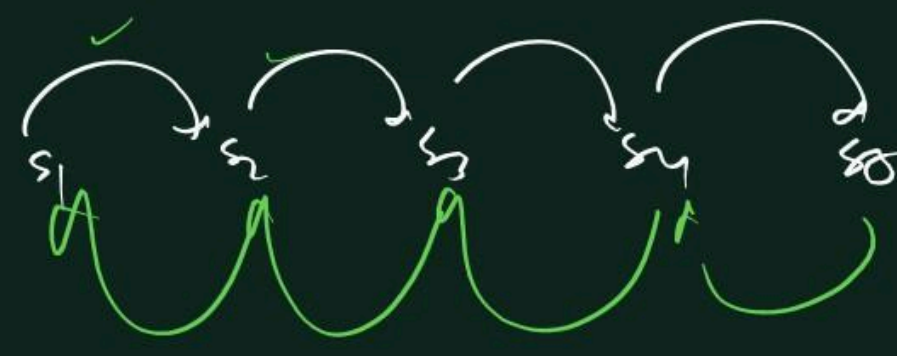



Problem
Loops
Recursion

```
def printL(a, i):
    if(i) = len(a):
        return
    print(a[i])
    printL(a, i+1)
```


TC / No of operat ∝ n
∝ (2n)
TC = O(n)
2n
Stack

arr = [ ] 5



Tc = O(n)

No of operations ∝ n

n

No of operations = 2×n

x ∝ n

x ∝ 2n