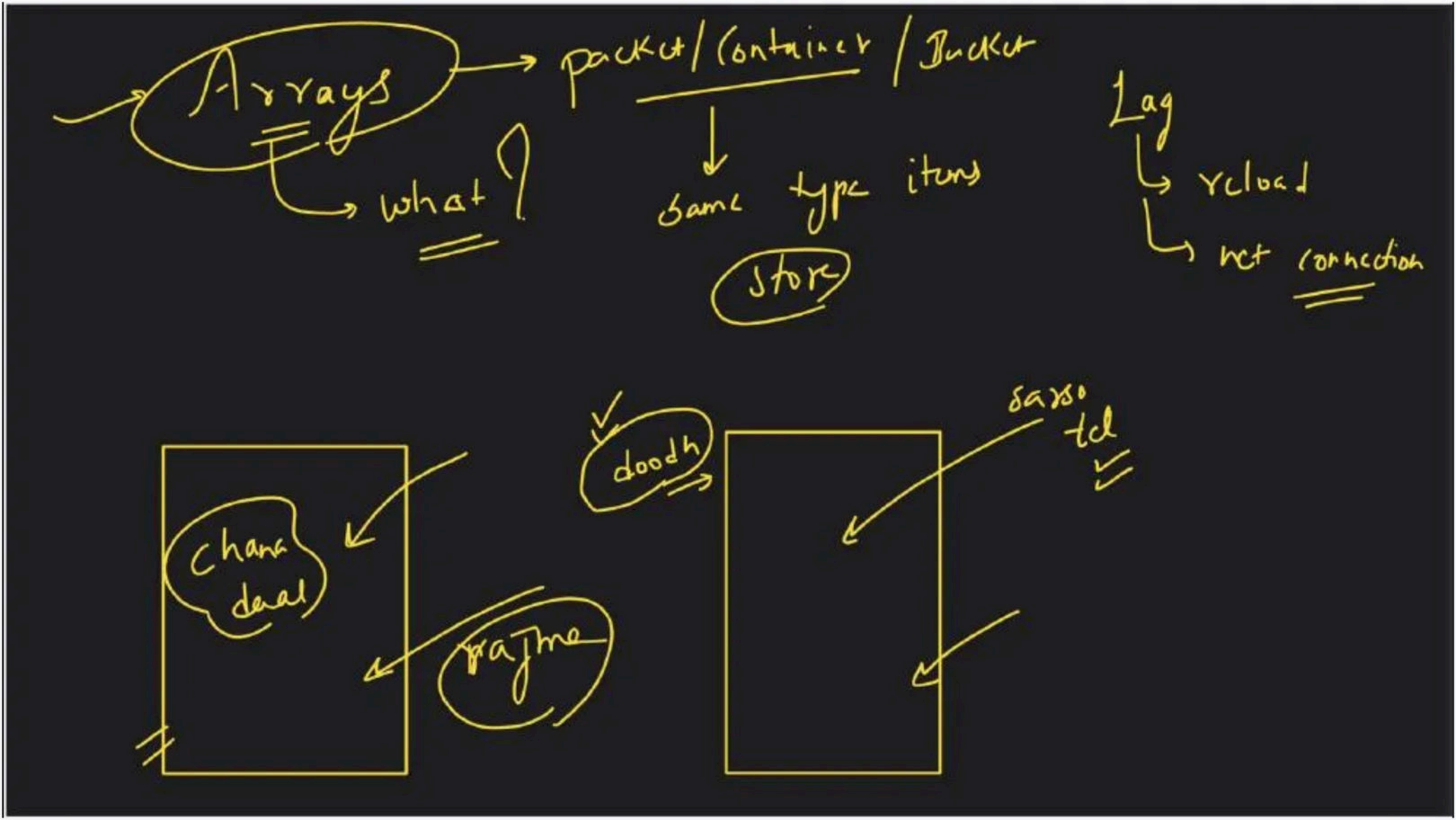
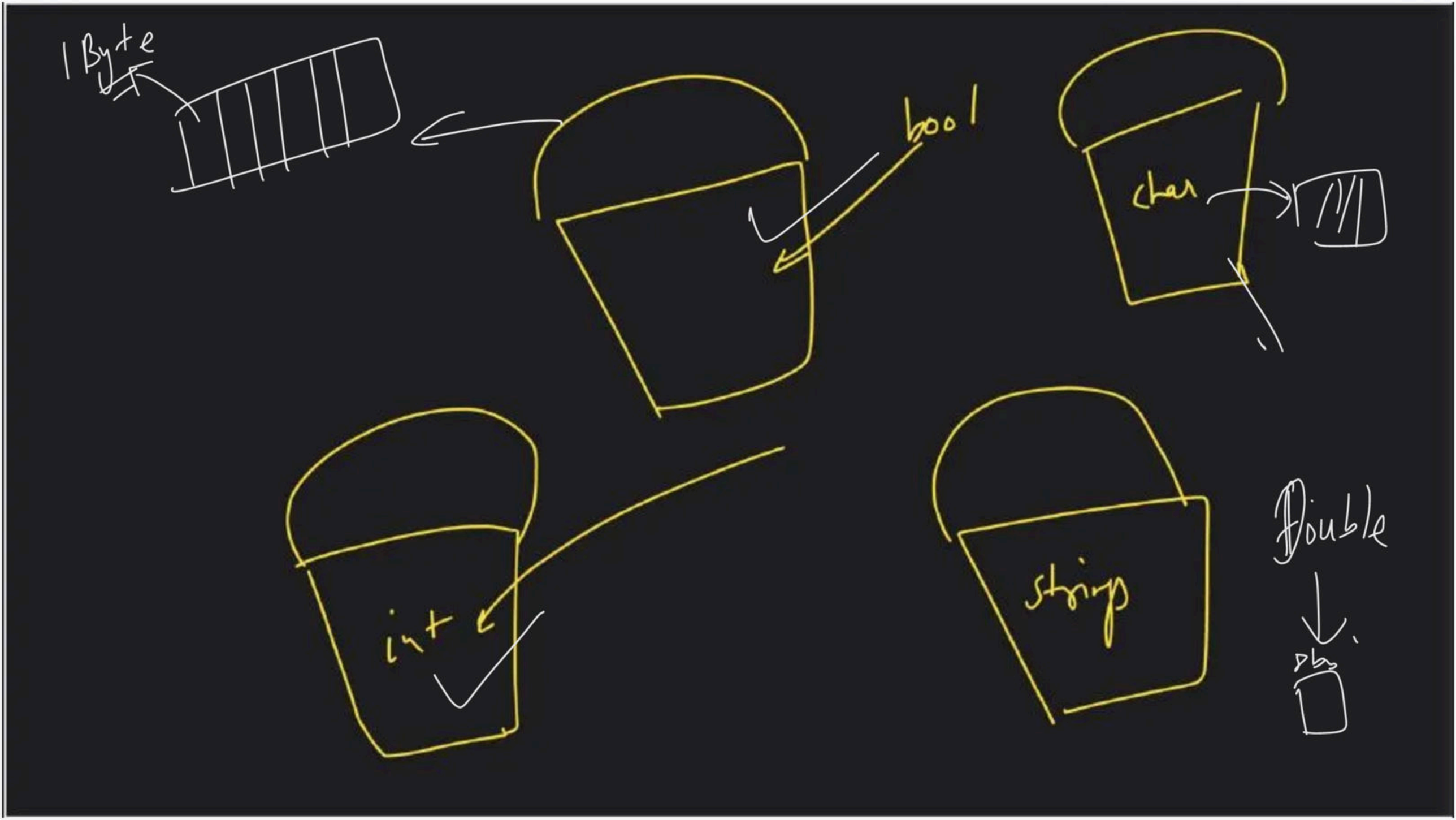
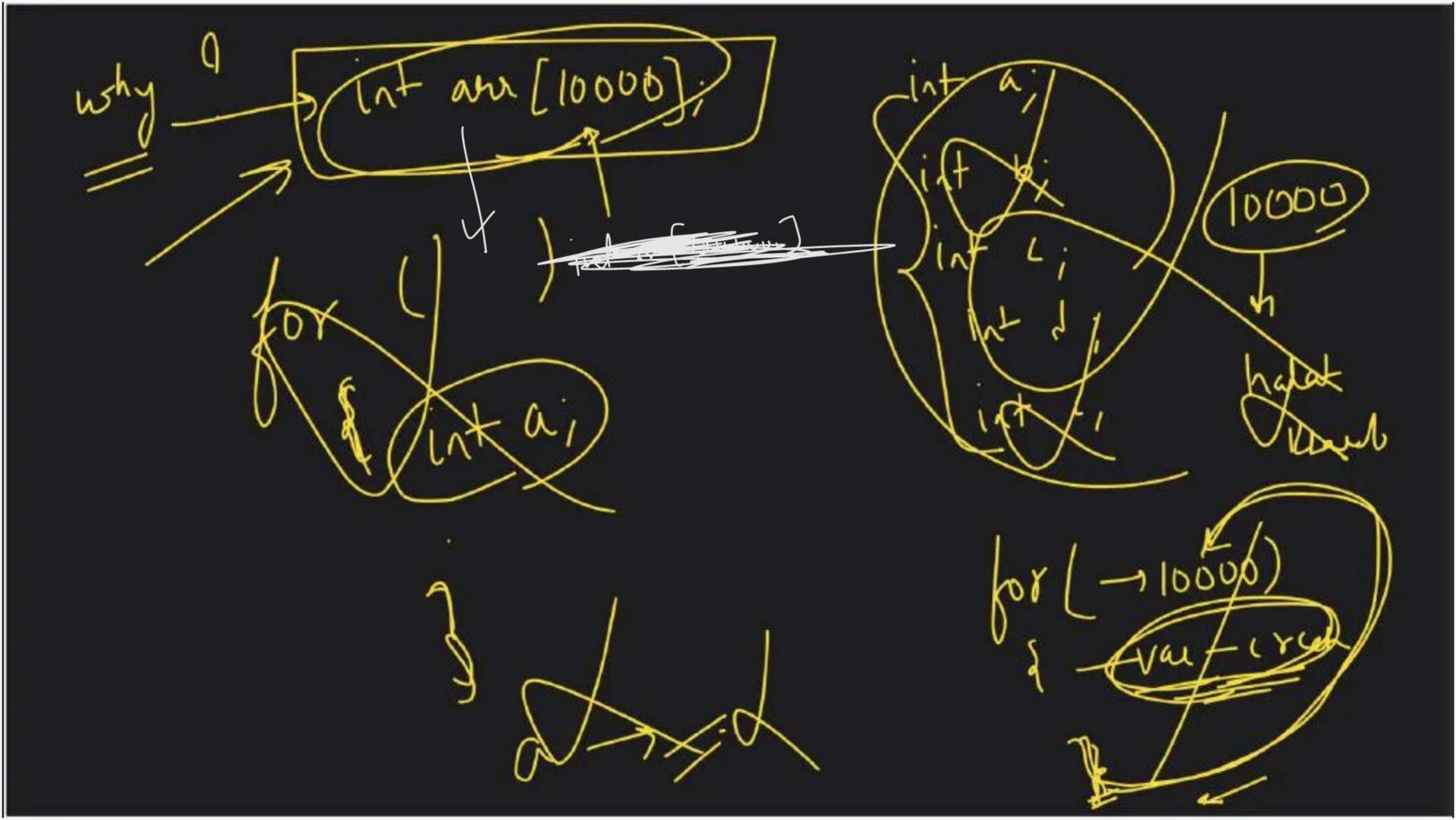
Doubts Class with Lakshay

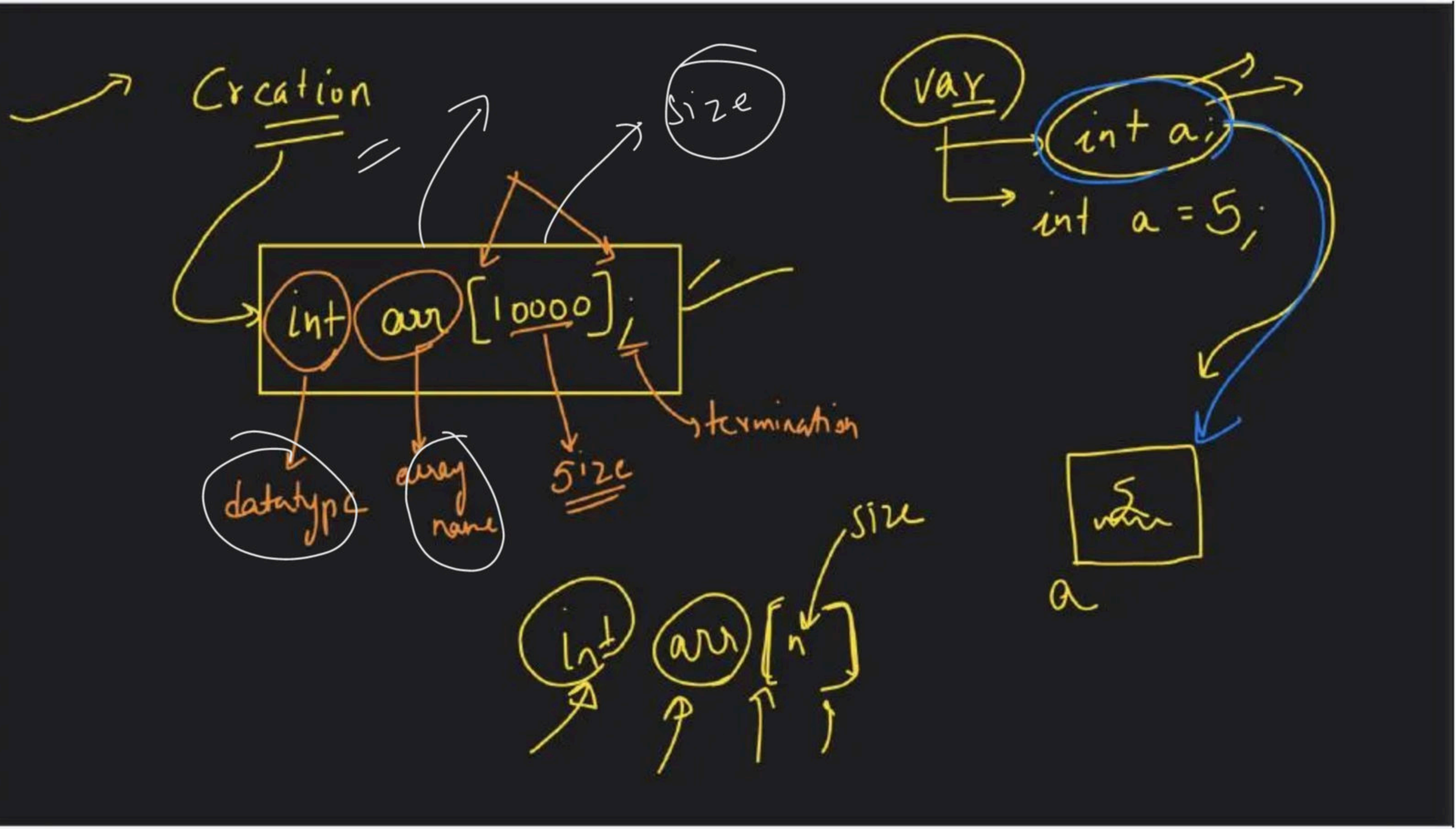
Special class

Love Babbar • May 18, 2024


Arrays
packet / container / Bucket
what ?
same type items
store
Lag
Ycloud
net connection
doodh
saxso
tcl
chana
dal
rajma


1 Byte
bool
char
int
string
Double


why ?
int arr[10000];
for ( )
{
int a;
}
int a;
int i;
int i;
int i;
10000
halt
for (→ 10000)
{


Creation
size
int arr [10000];
termination
datatype
array name
size
int arr [n];
size
var
int a;
int a = 5;
5
a

cin >> size

dynamic memory allocation

int arr[5]

int size;

cin >> size;

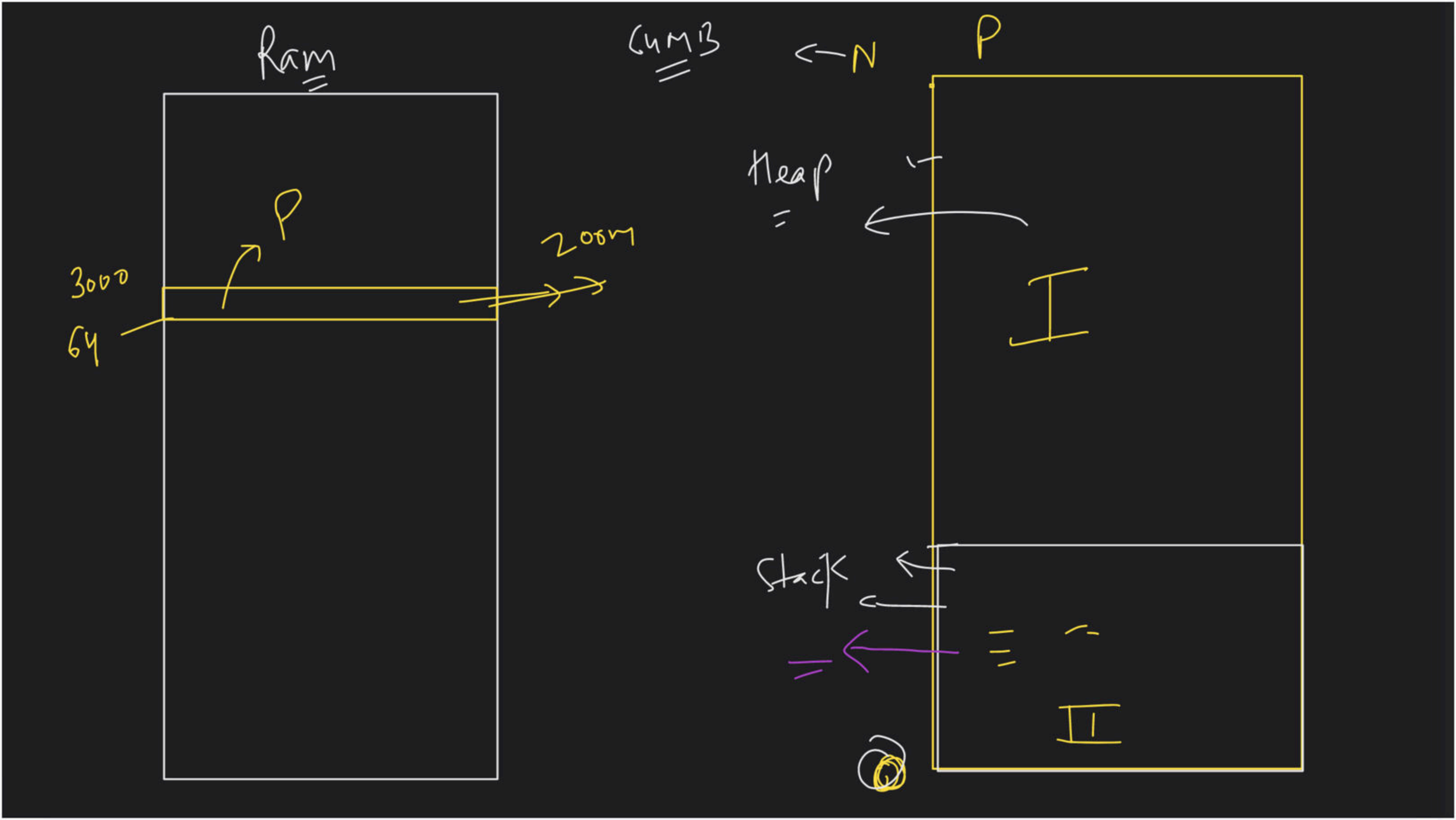
int arr[size];

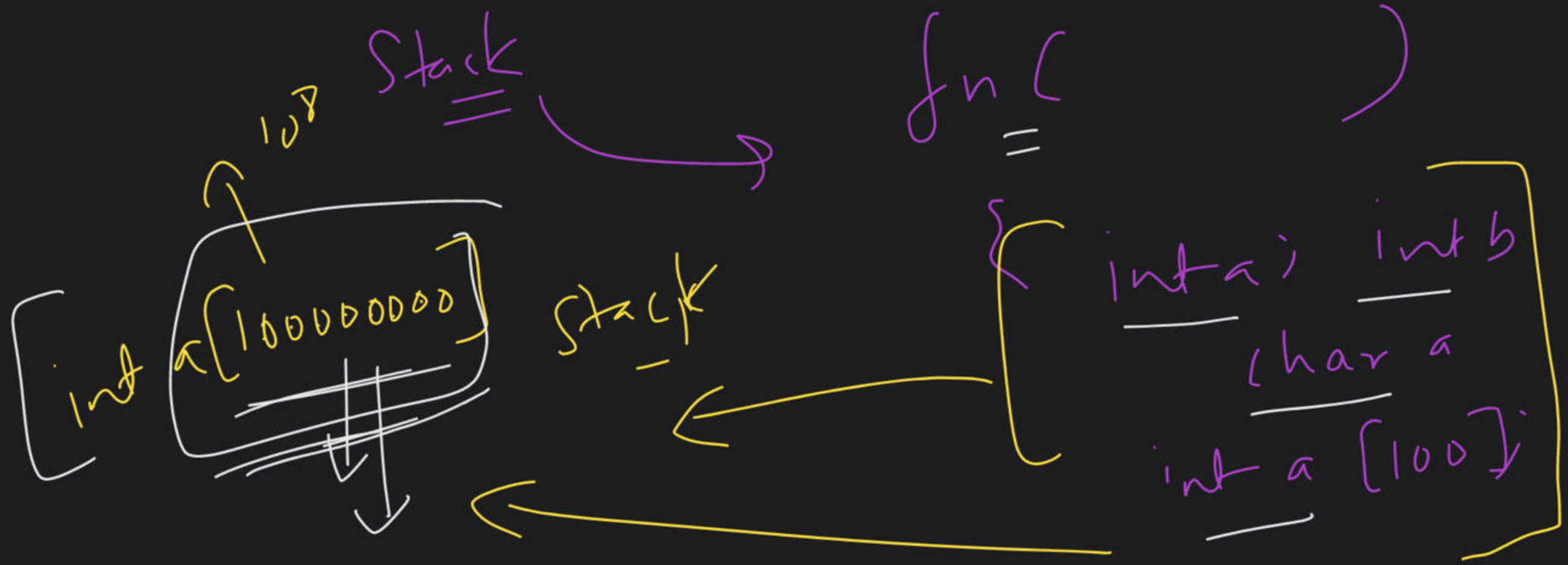
Memory Allocation

why

BAD

Practice

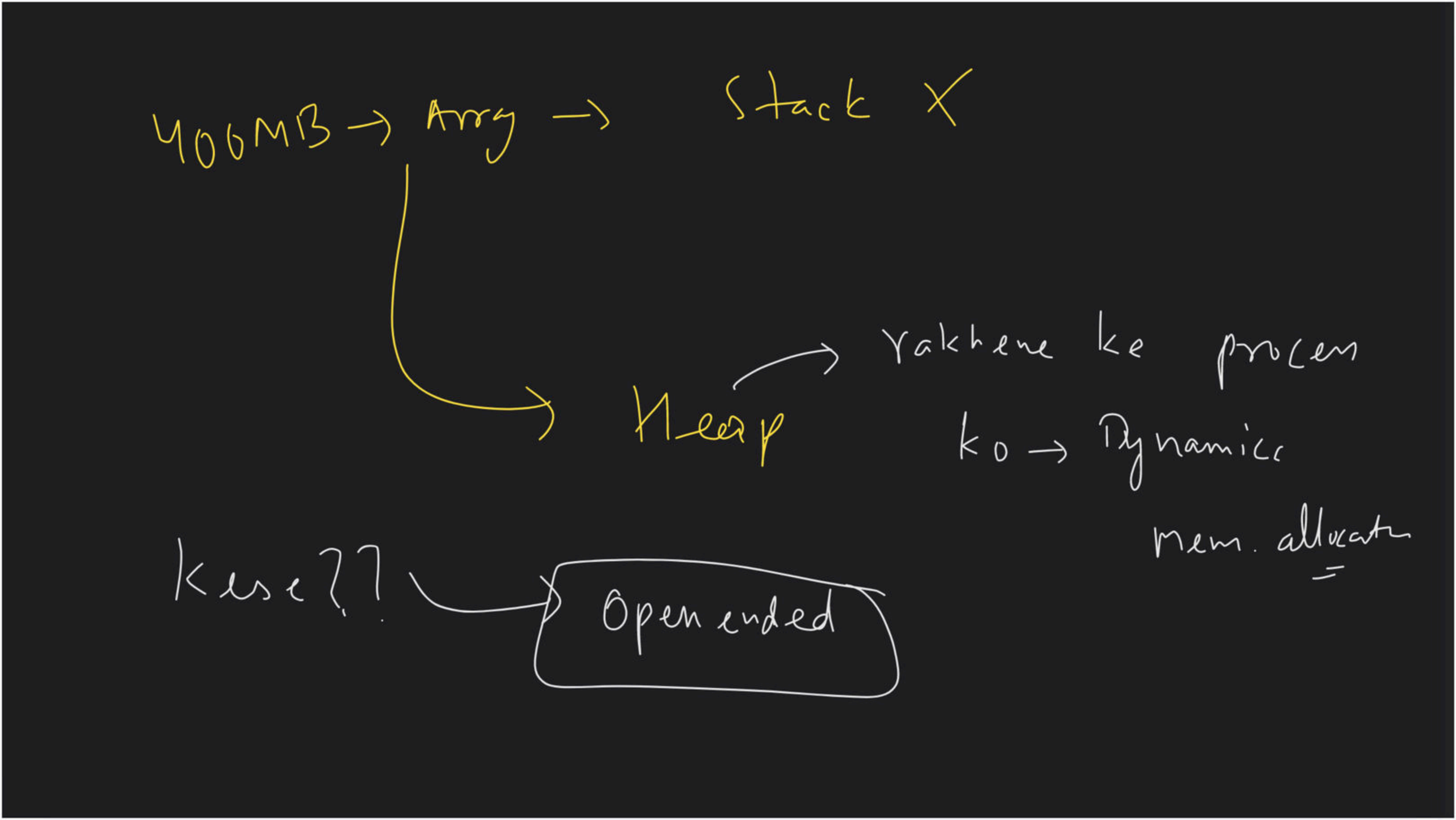

Ram
CUMB
N
P
P
3000
64
200M
Heap
I
Stack
II

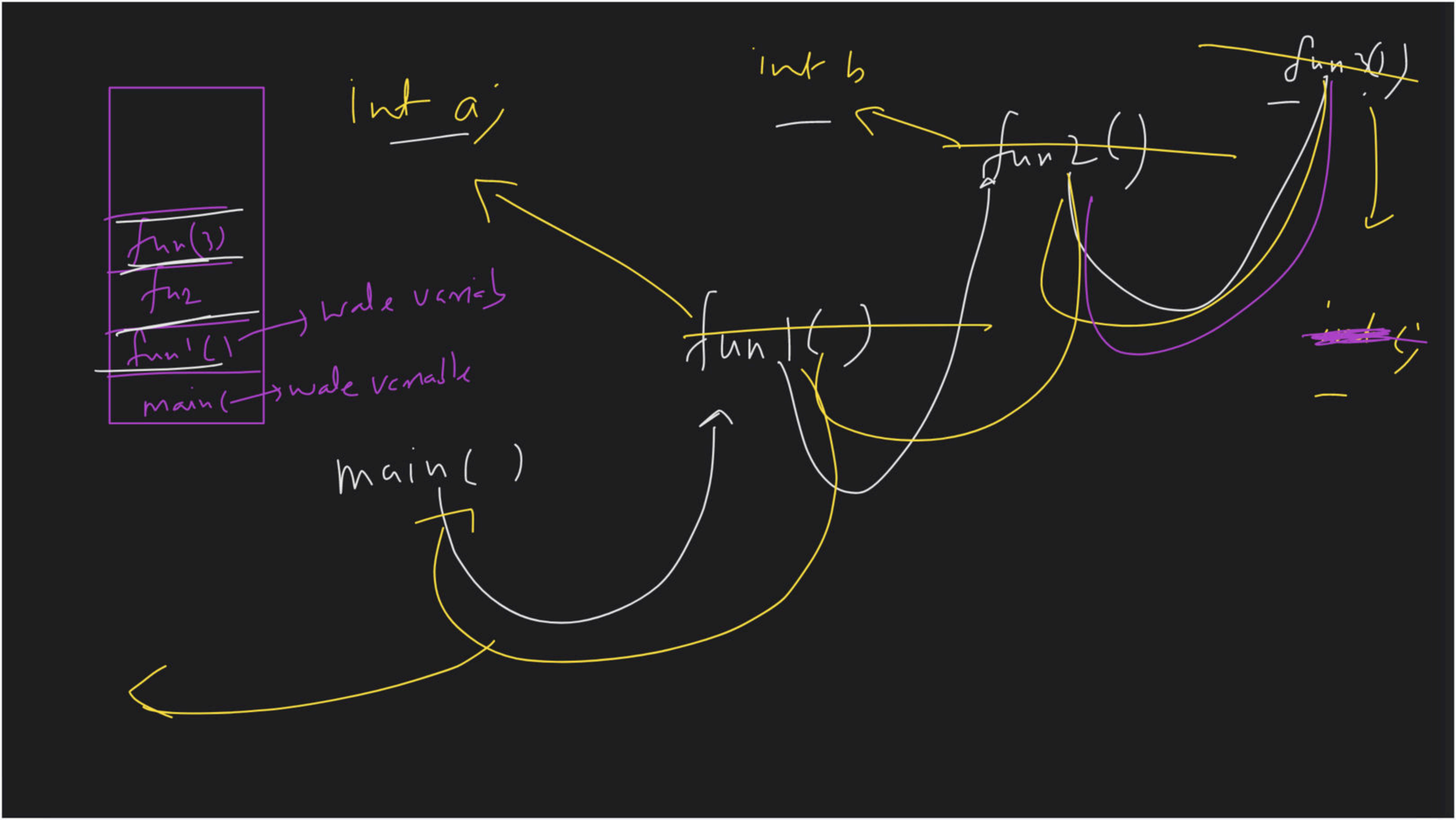


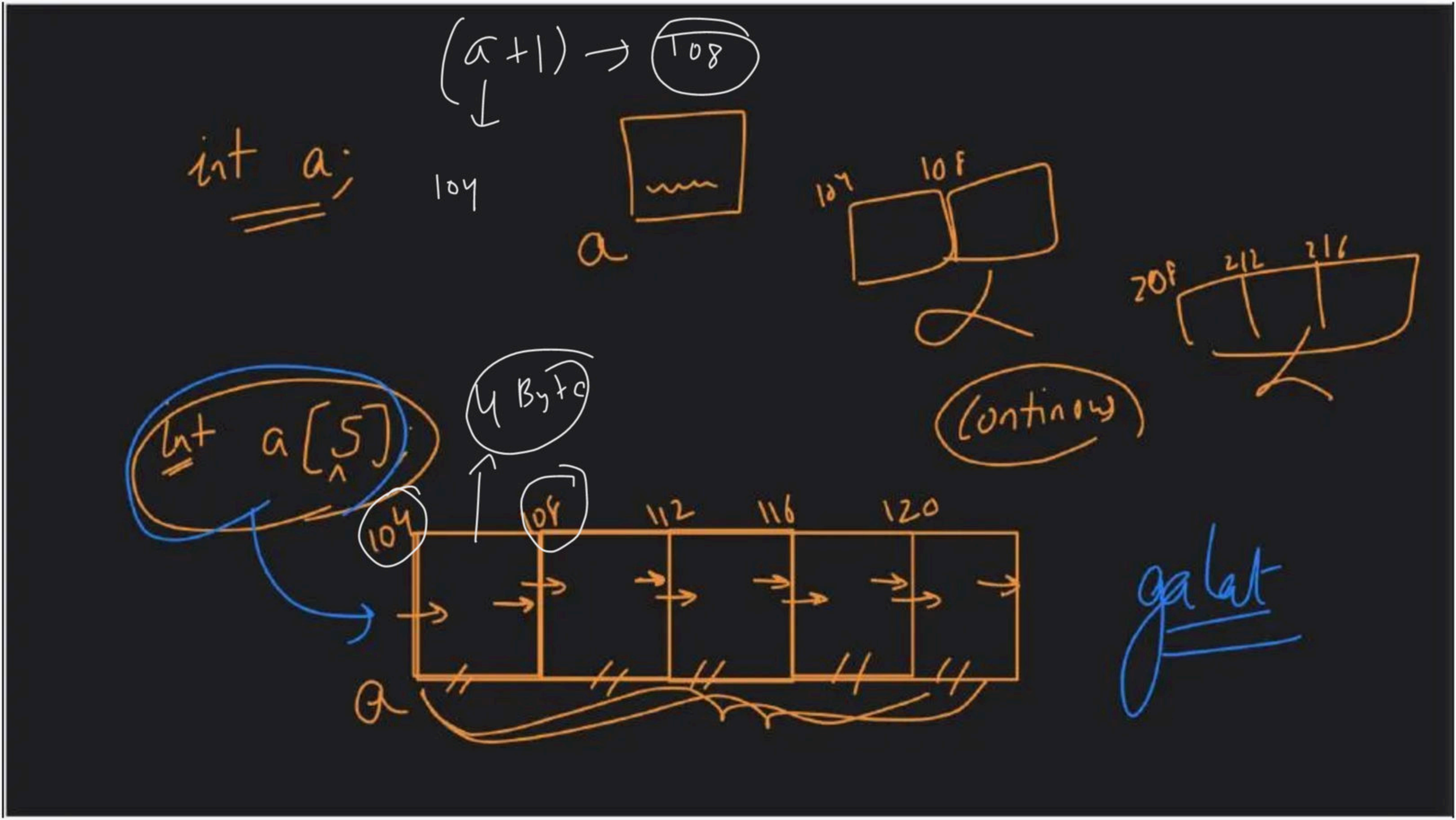
$\Rightarrow$

$100000000 \times 4 \Rightarrow 400,000,000$ Bytes

$\approx$ 400MB

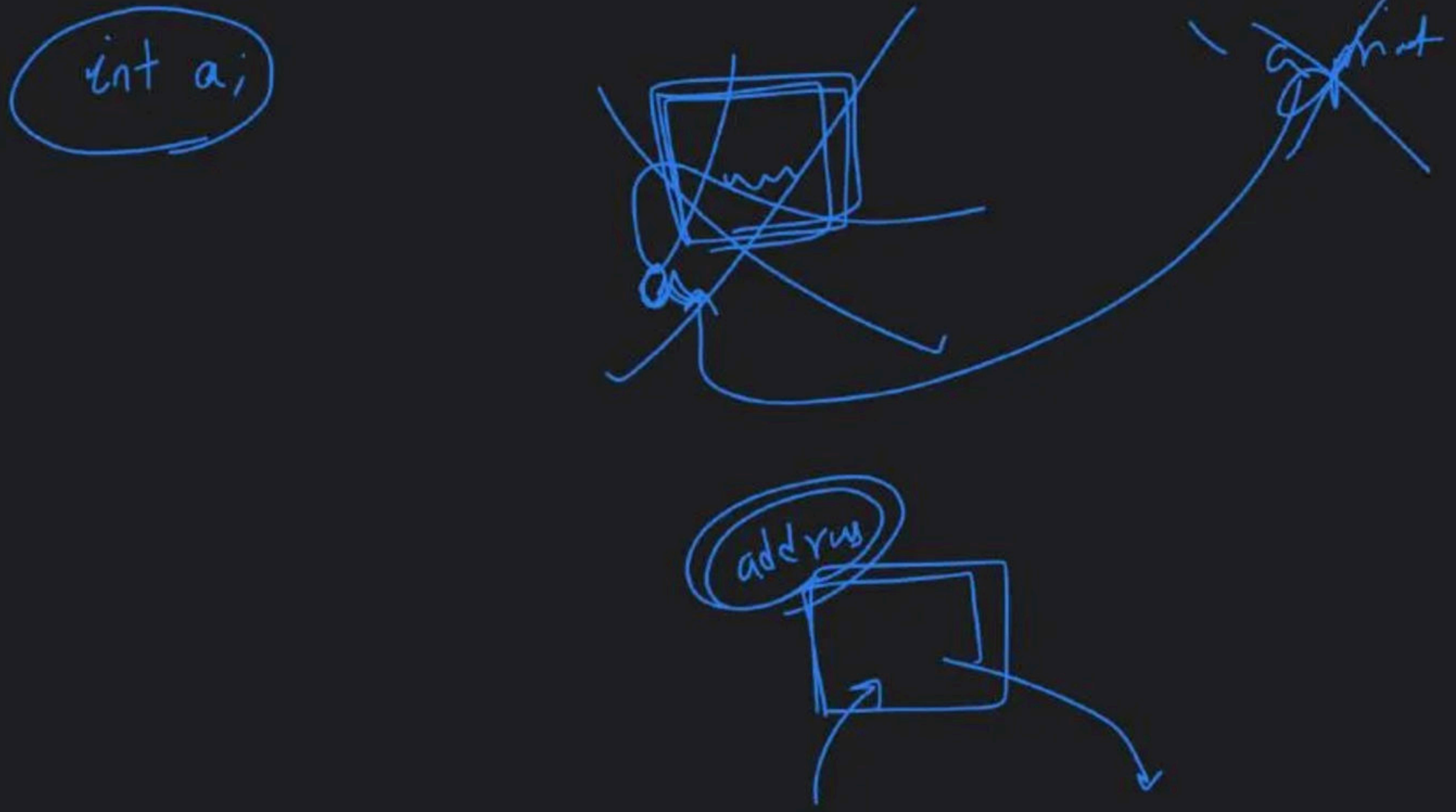

400MB → Array → Stack X
Heap
rakhene ke process
ko → Dynamic
mem. allocate
Kese ??
Open ended

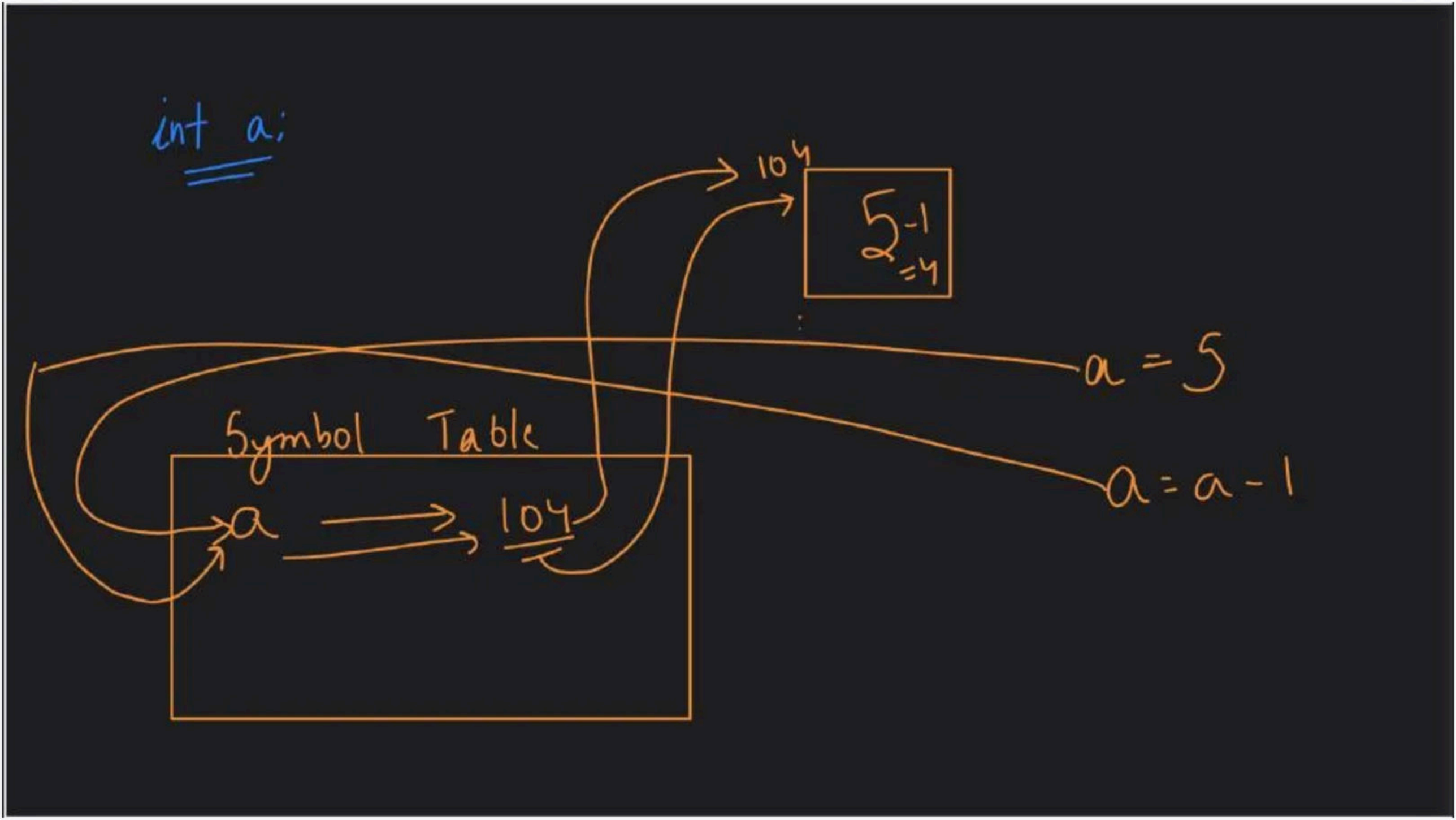

int a;
int b
fun3()
fun2()
fun3()
fun2
fun1()
wade variable
wade variable
fun1()
int i;
main()

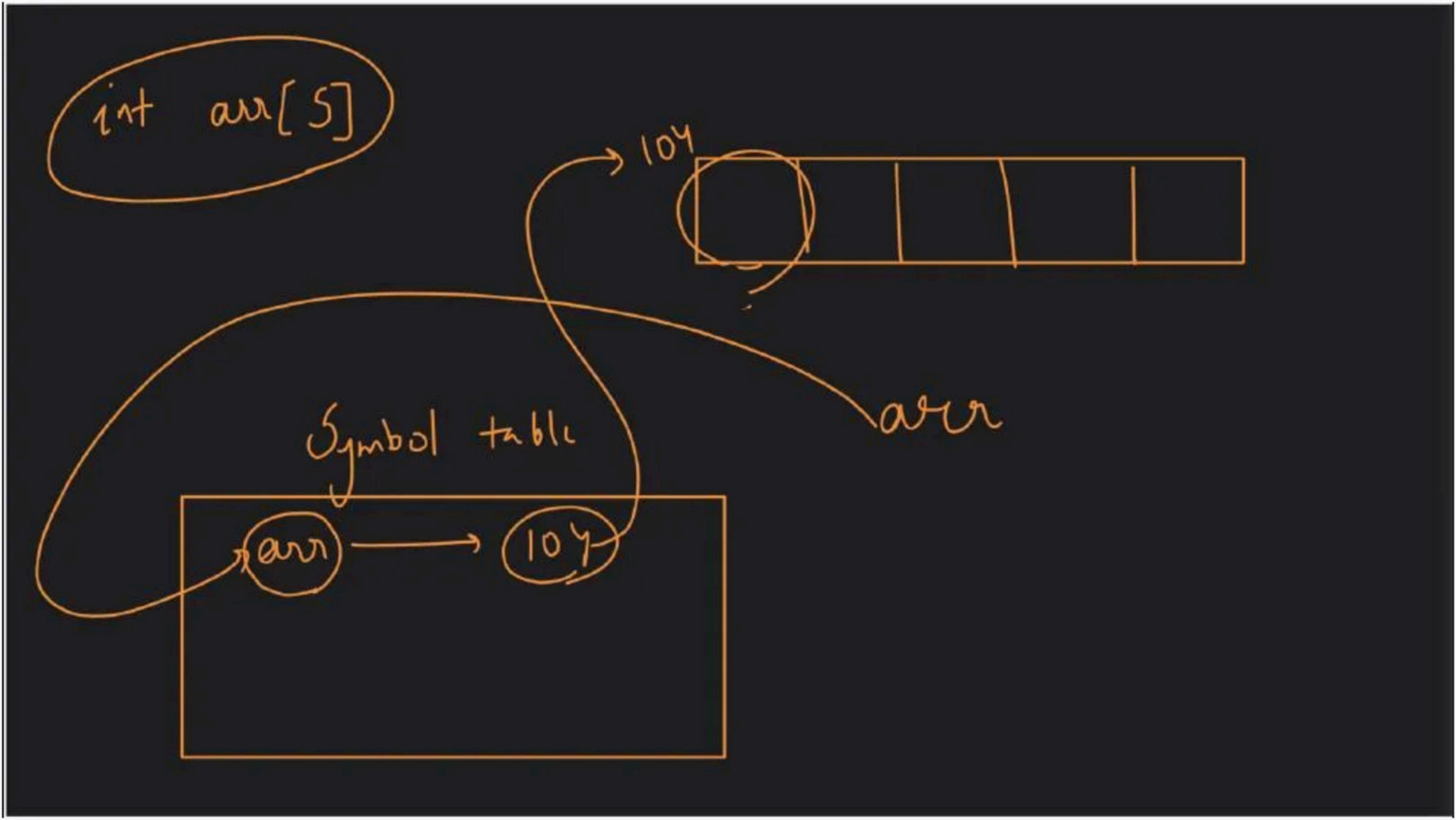

(a+1) → 108
int a;
104
a
104
108
204
208
212
216
int a[5];
4 Byte
104
108
112
116
120
a
Continue
galat

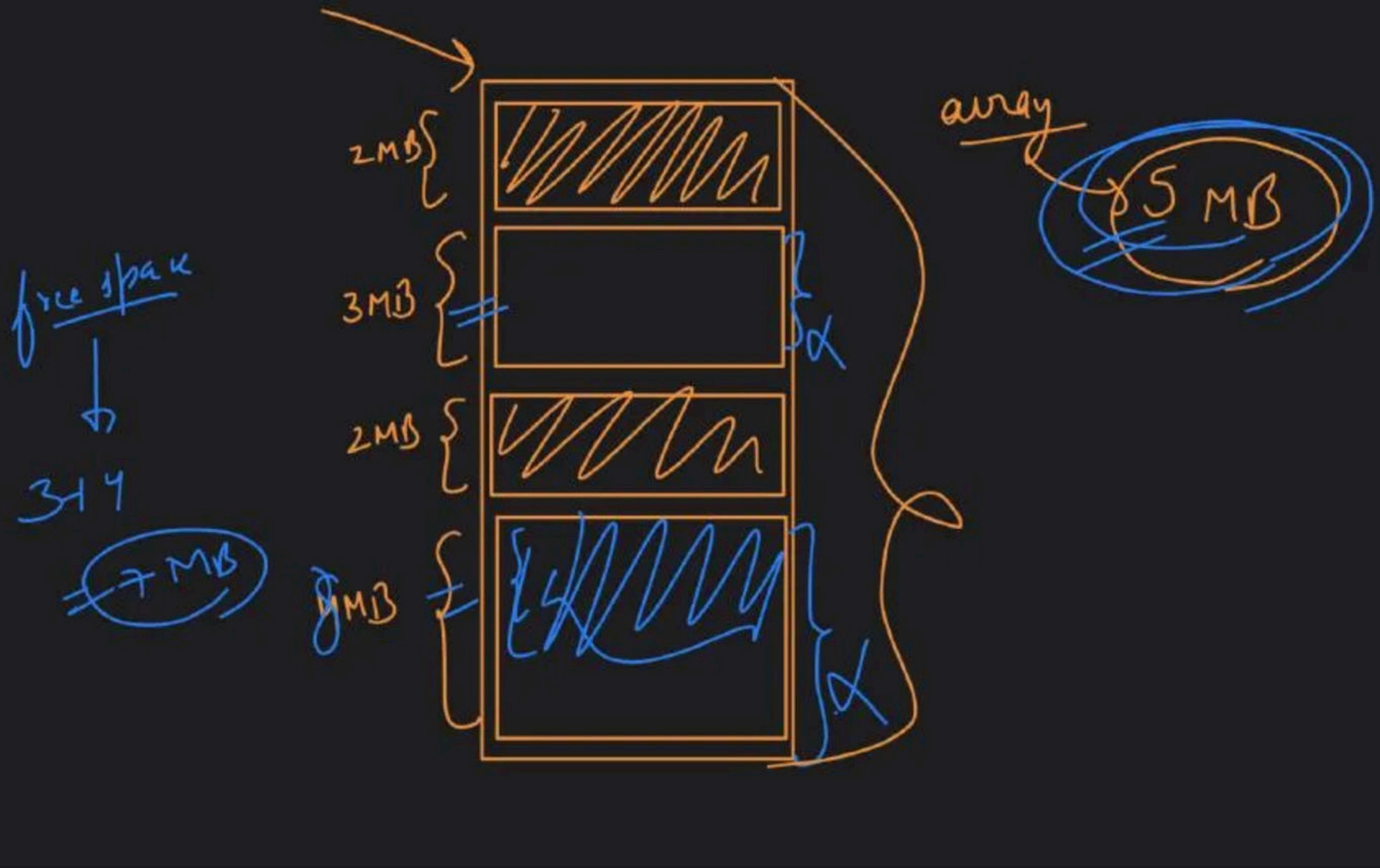
double a[10];

(a+1) → ??

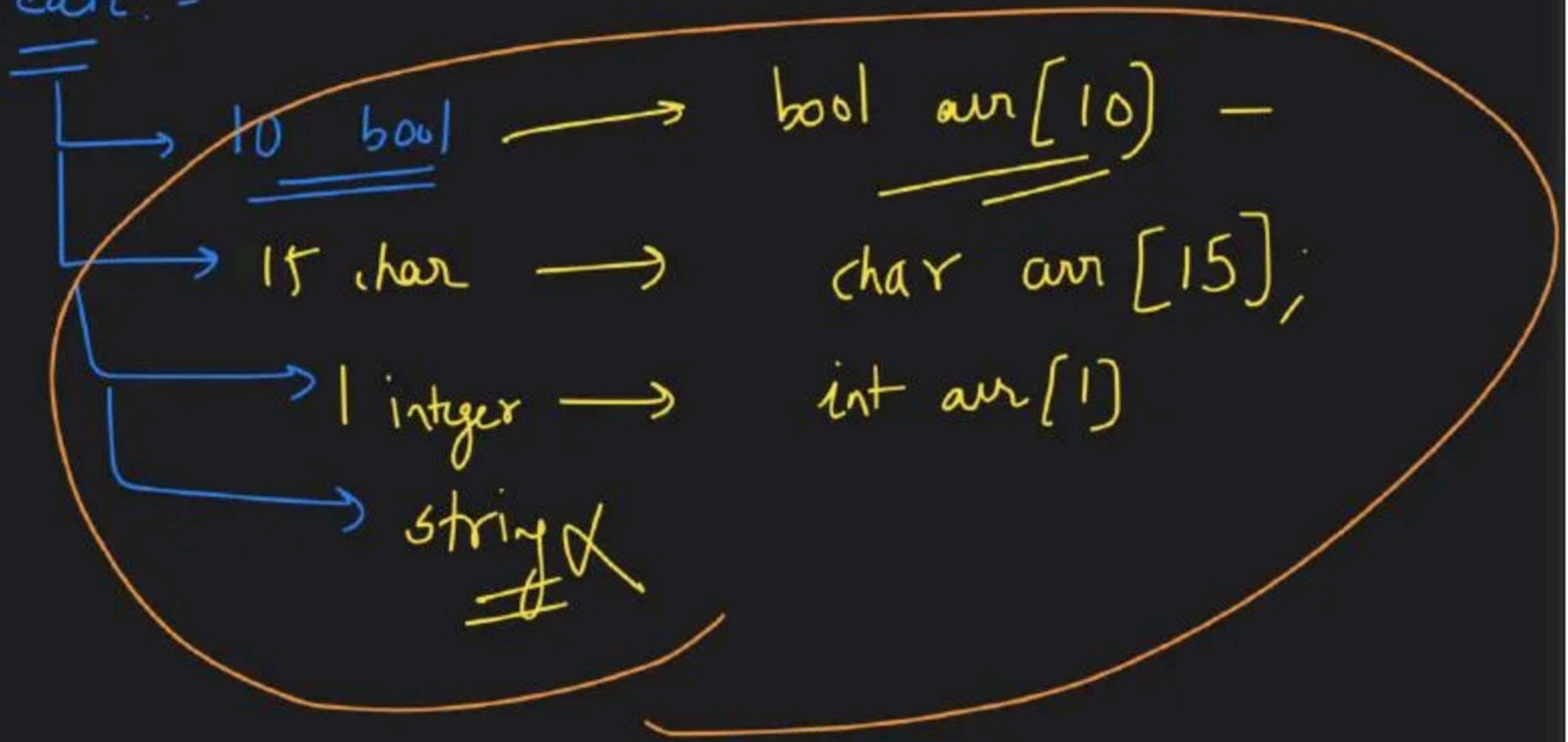

int a;
int
address

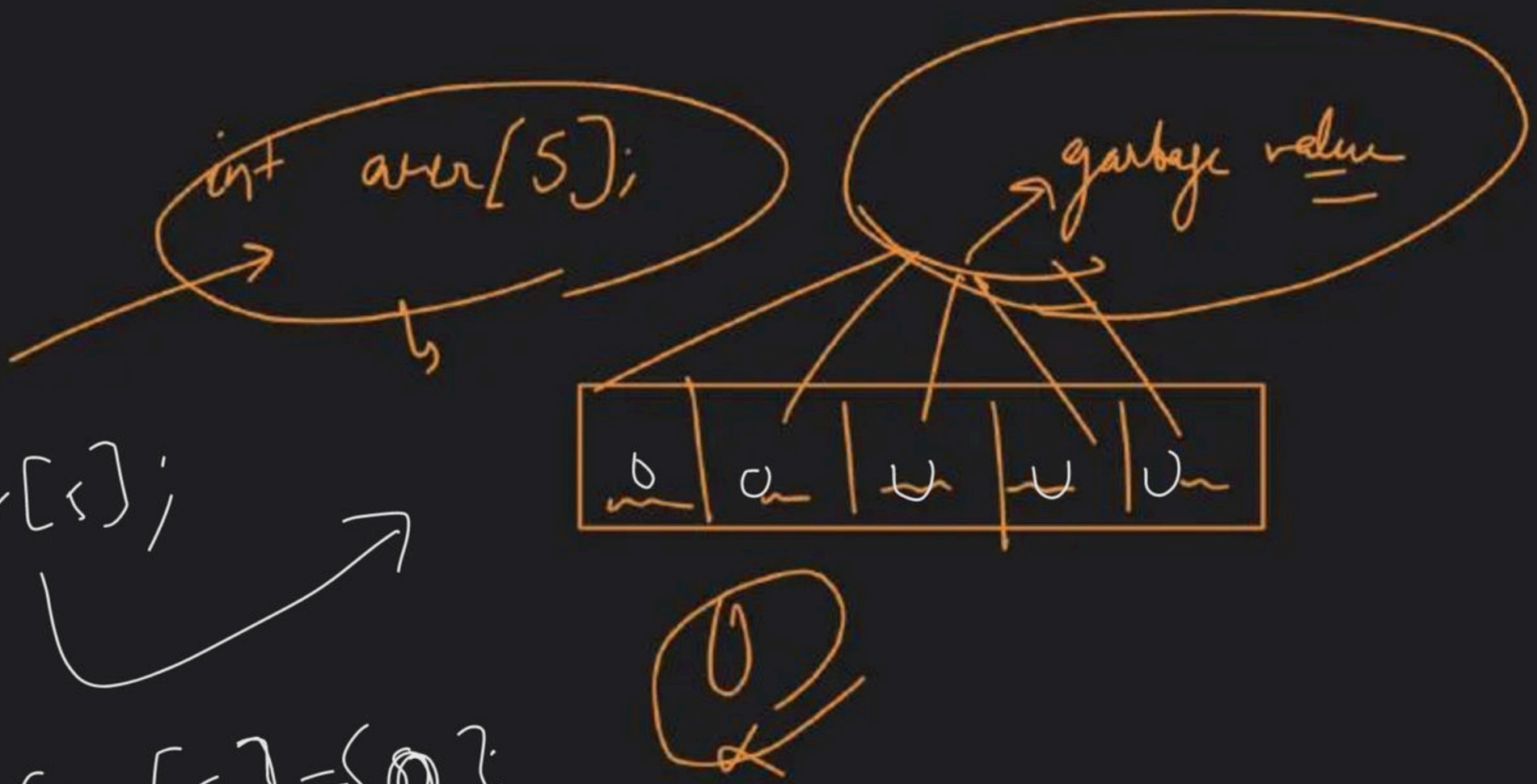

int a;
104
5
-1
=4
a = 5
a = a - 1
Symbol Table
a
104

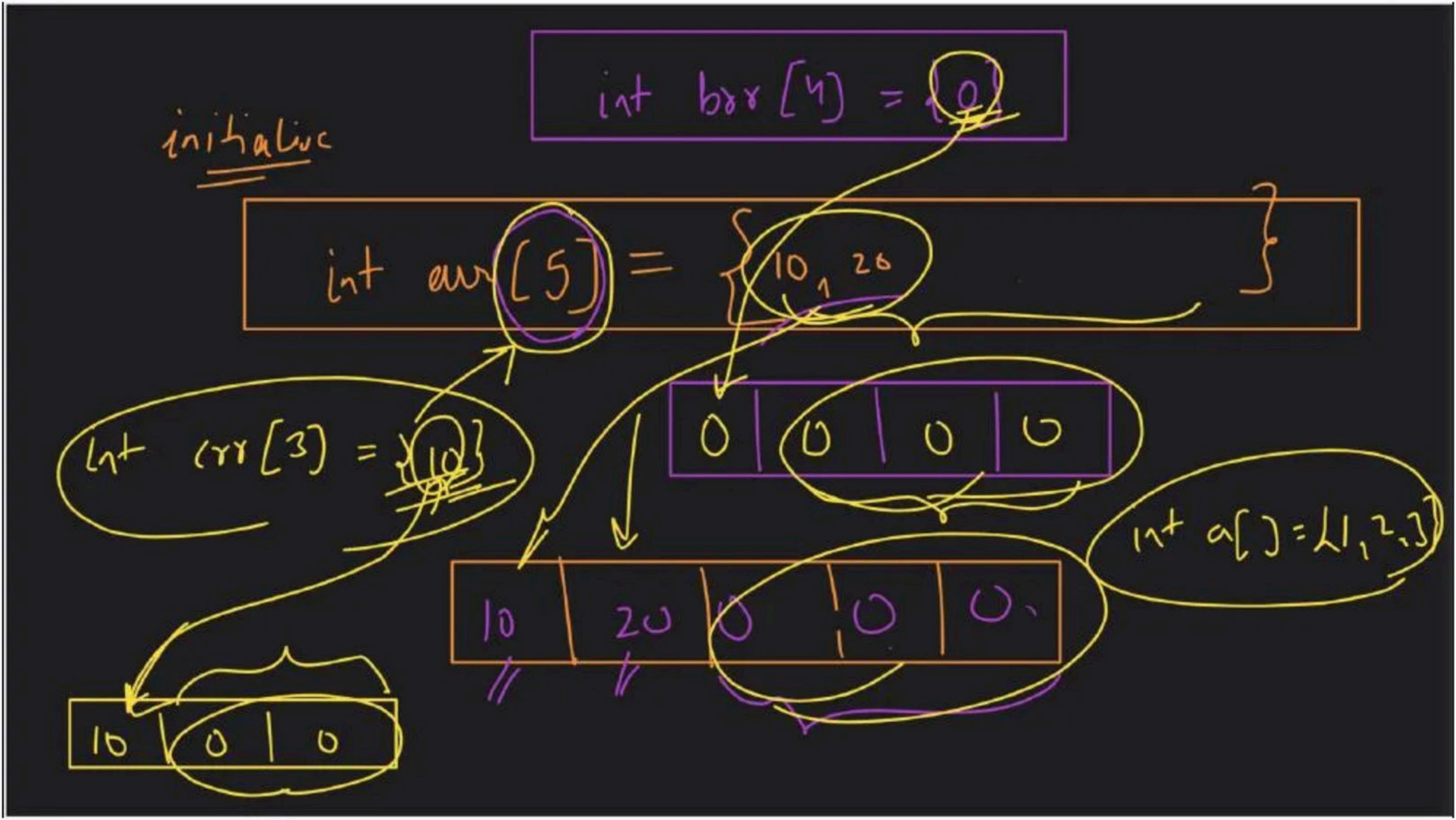

int arr[5]
104
arr
Symbol table
arr
104

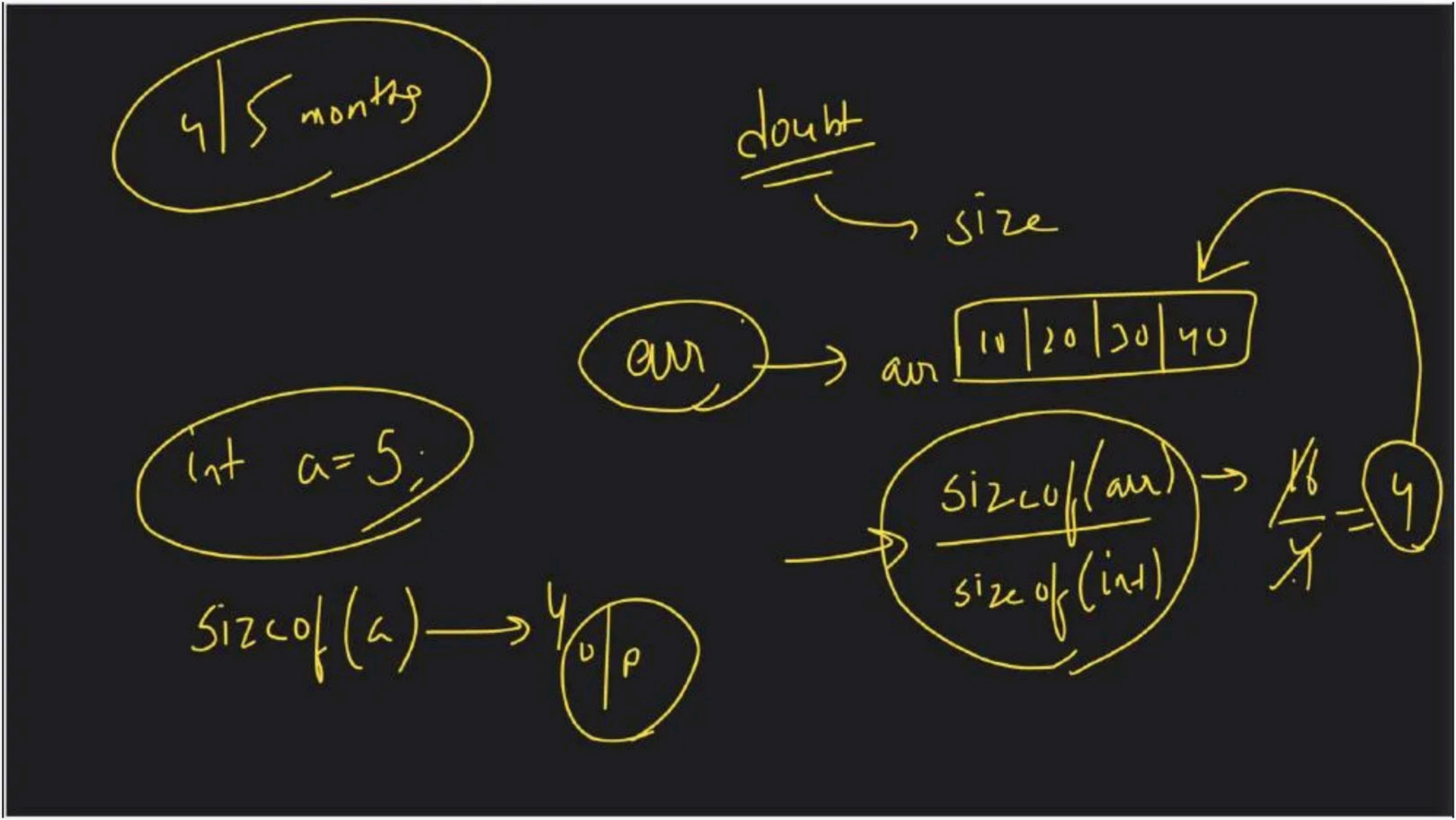

2MB
3MB
2MB
3MB
array
5 MB
free space
314
7 MB

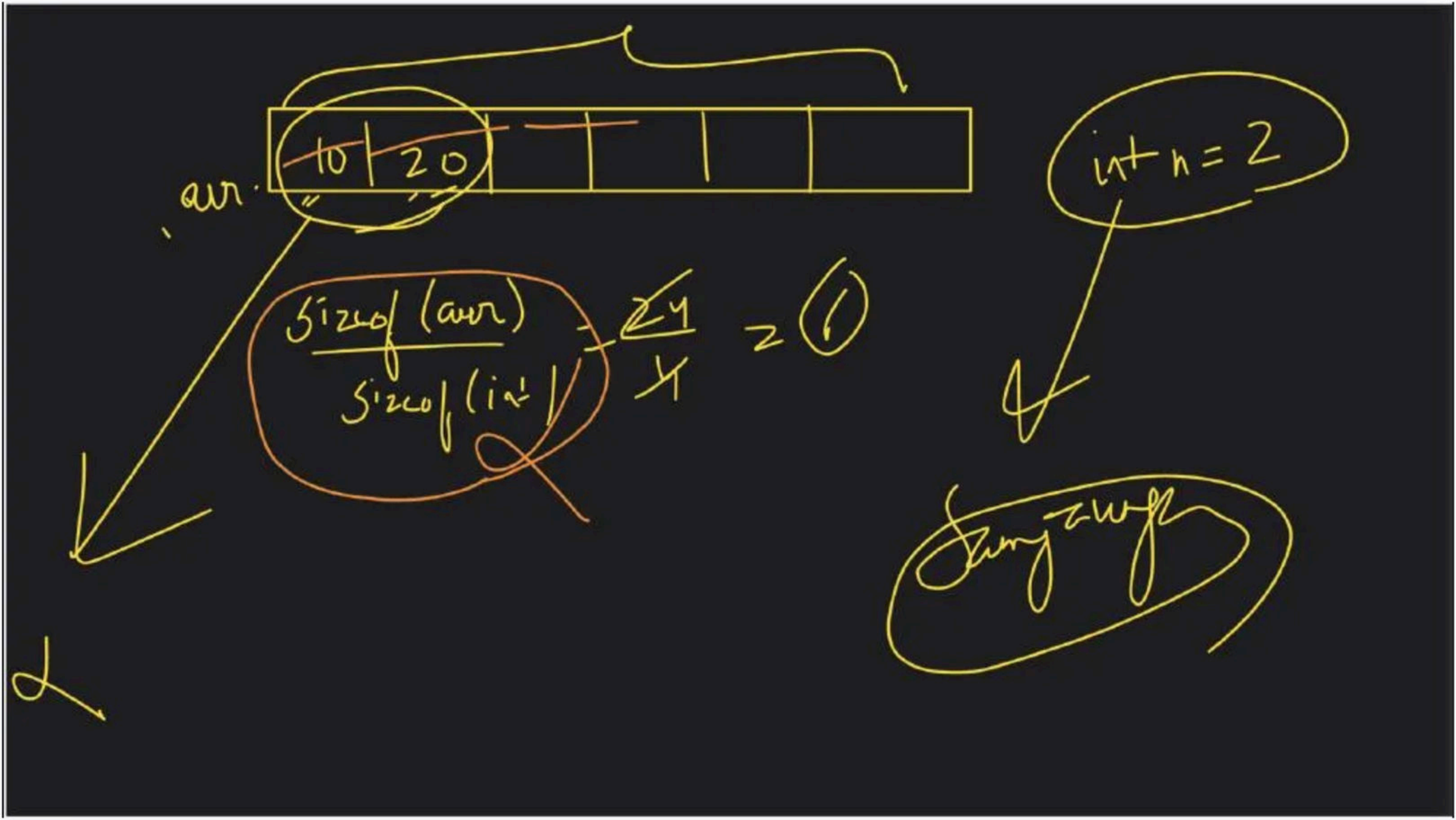
→ Create :-

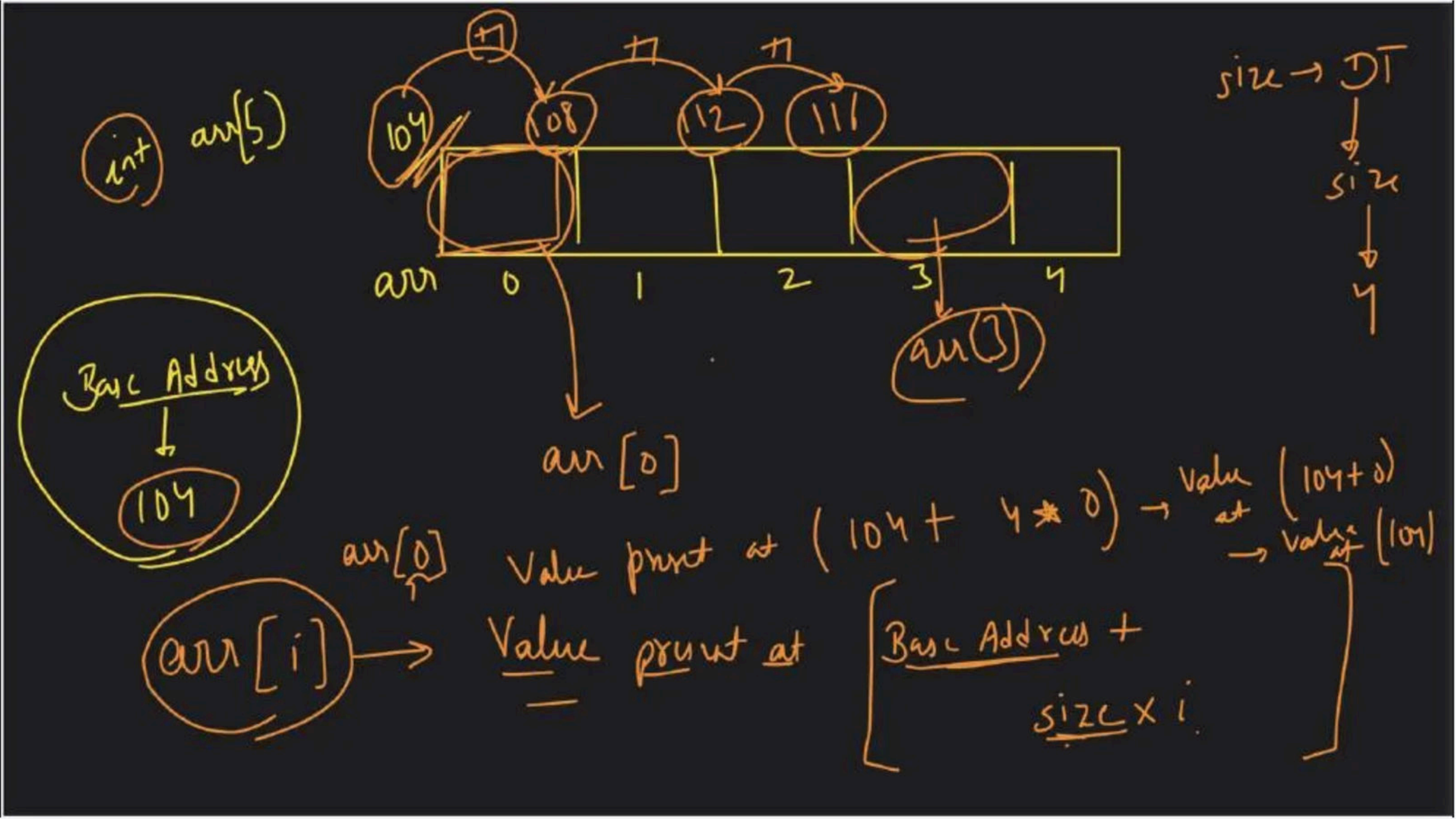

10 bool → bool arr[10] ;
If char → char arr[15];
1 integer → int arr[1]
string ✗

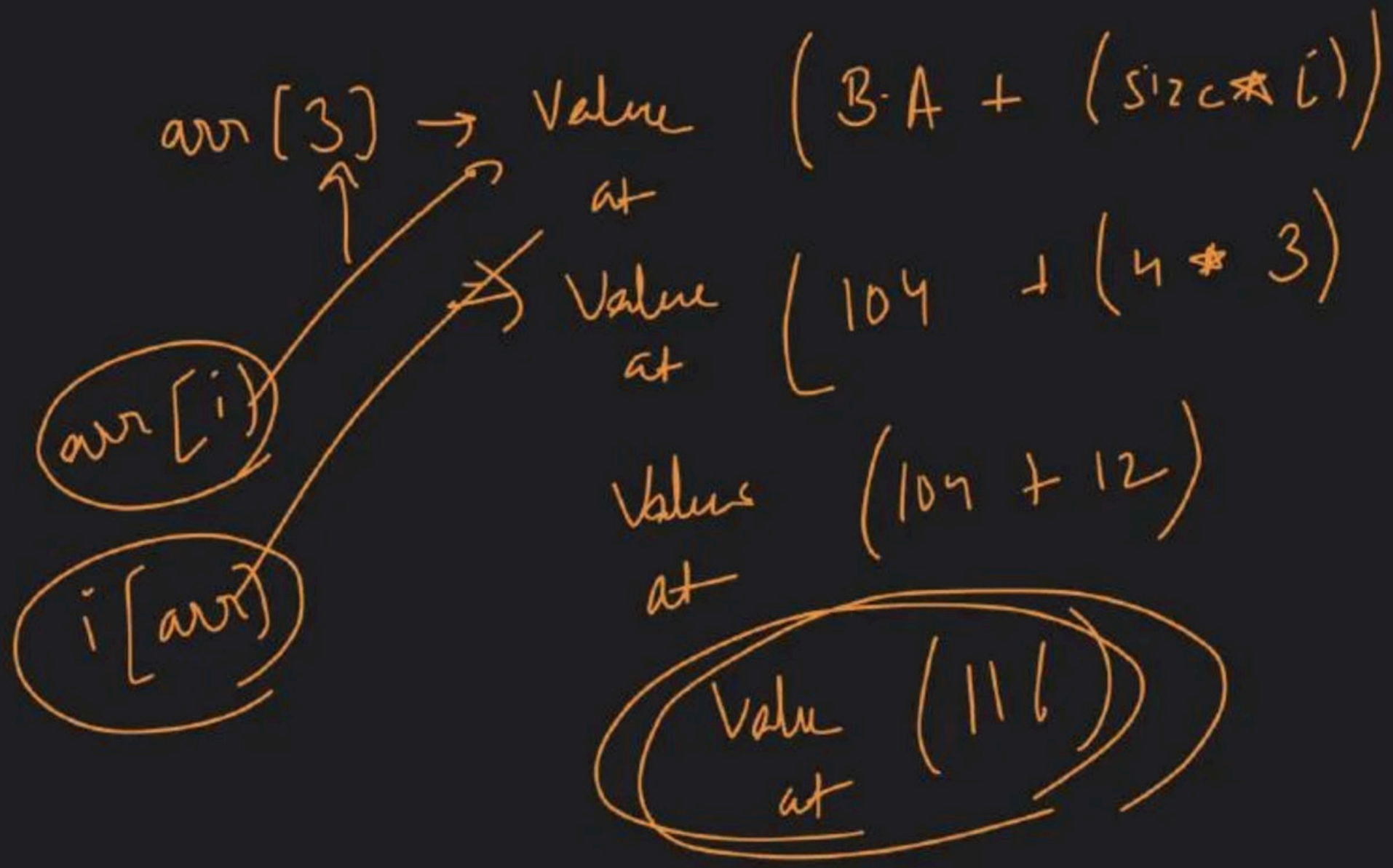

int arr[5];
garbage value
0
int arr[5];
int arr[5]={0};


int bar[4] = {0};
initialize
int arr[5] = {10, 20};
int arr[3] = {10};
0 0 0 0
10 20 0 0 0
10 0 0
int a[] = {1, 2, 3};


4/5 months
doubt
size
arr
arr
10
20
30
40
int a=5;
sizeof(arr)
sizeof(int)
16
4
4
sizeof(a)
4
0
0
P


arr
10
20
sizeof(arr) / sizeof(int)
24 / 4 = 6
int n = 2
Sizeof

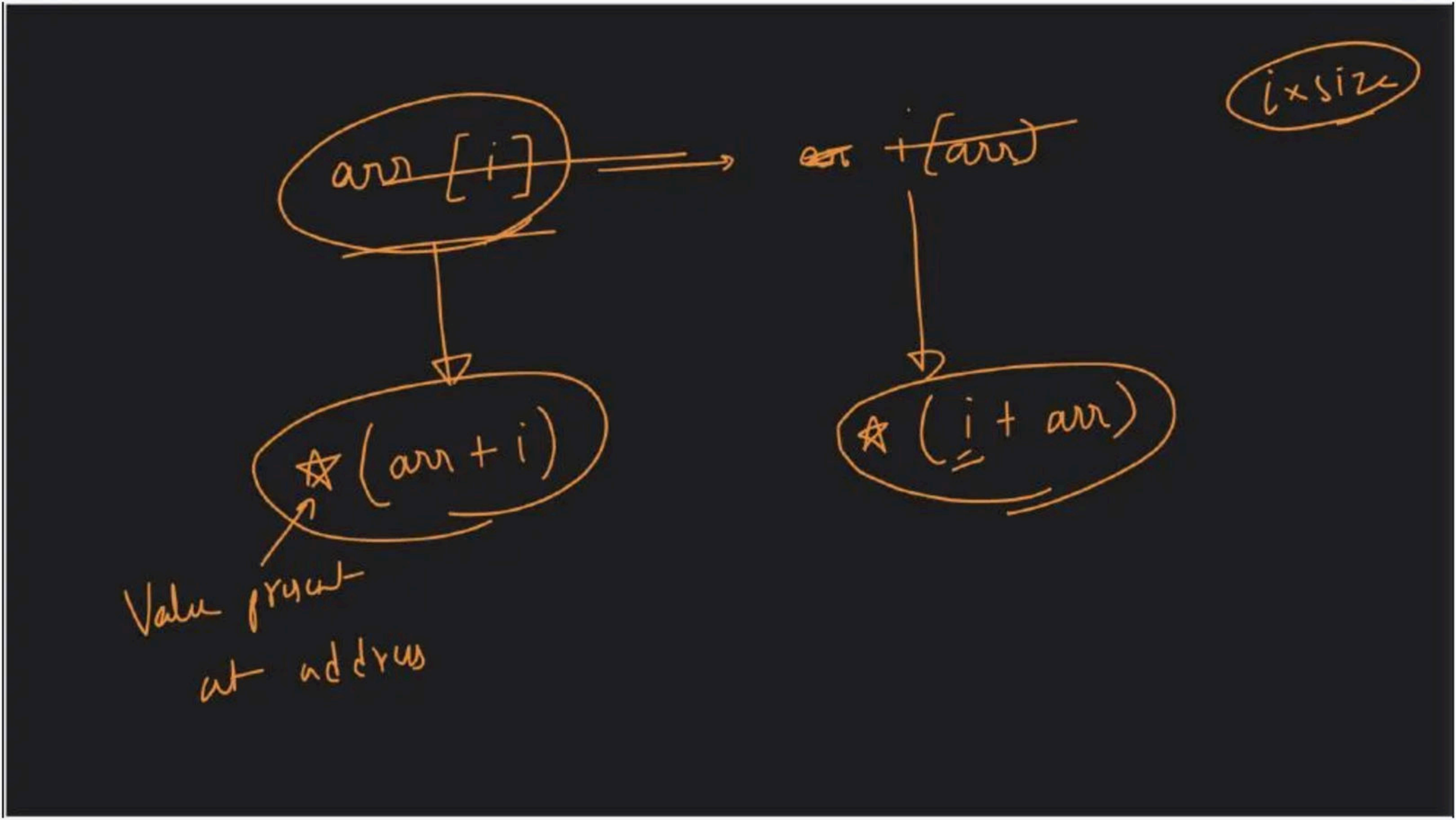

int arr[5]
104
108
112
116
arr
0
1
2
3
4
arr[3]
arr[0]
size → DT
size
4
Base Address
104
arr[0] Value present at (104 + 4 * 0) → Value at (104+0)
→ Value at (104)
arr[i] → Value present at [Base Address + size x i]

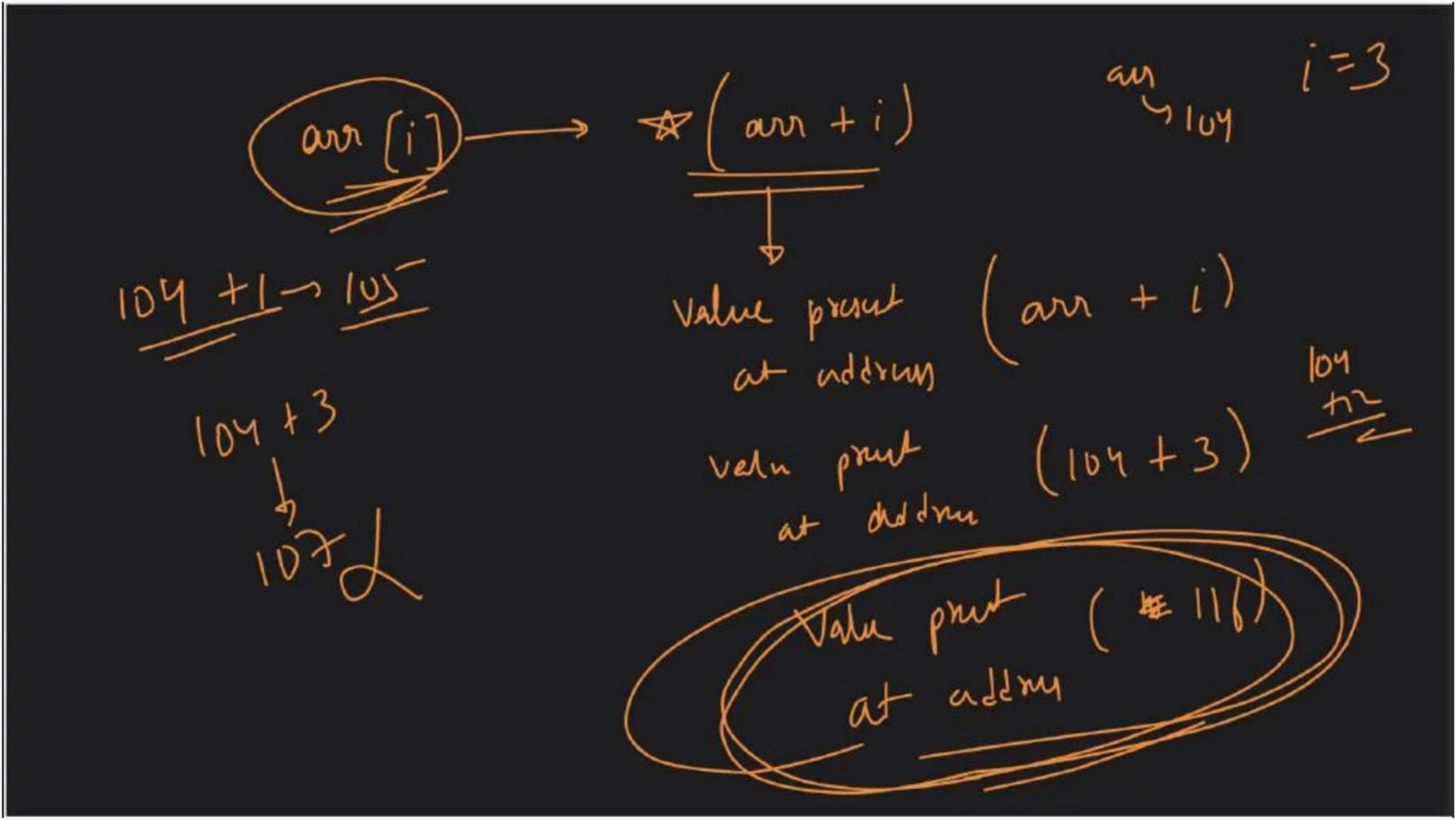

arr[3] → Value at (B.A + (size * i))
Value at (104 + (4 * 3))
Value at (104 + 12)
Value at (116)
arr[i]
i[arr]

arr[i] → Value at (arr + (size * i))

i[arr]

arr[i] → i[arr]


i x size
arr[i]
arr + (arr)
*(arr + i)
*(i + arr)
Value present at address


arr [i]
*(arr + i)
arr → 104
i = 3
104 + 1 → 105
Value present at address (arr + i)
104 + 3
↓
107
Value present at address (104 + 3)
104 + 2
Value present at addrss (*110)

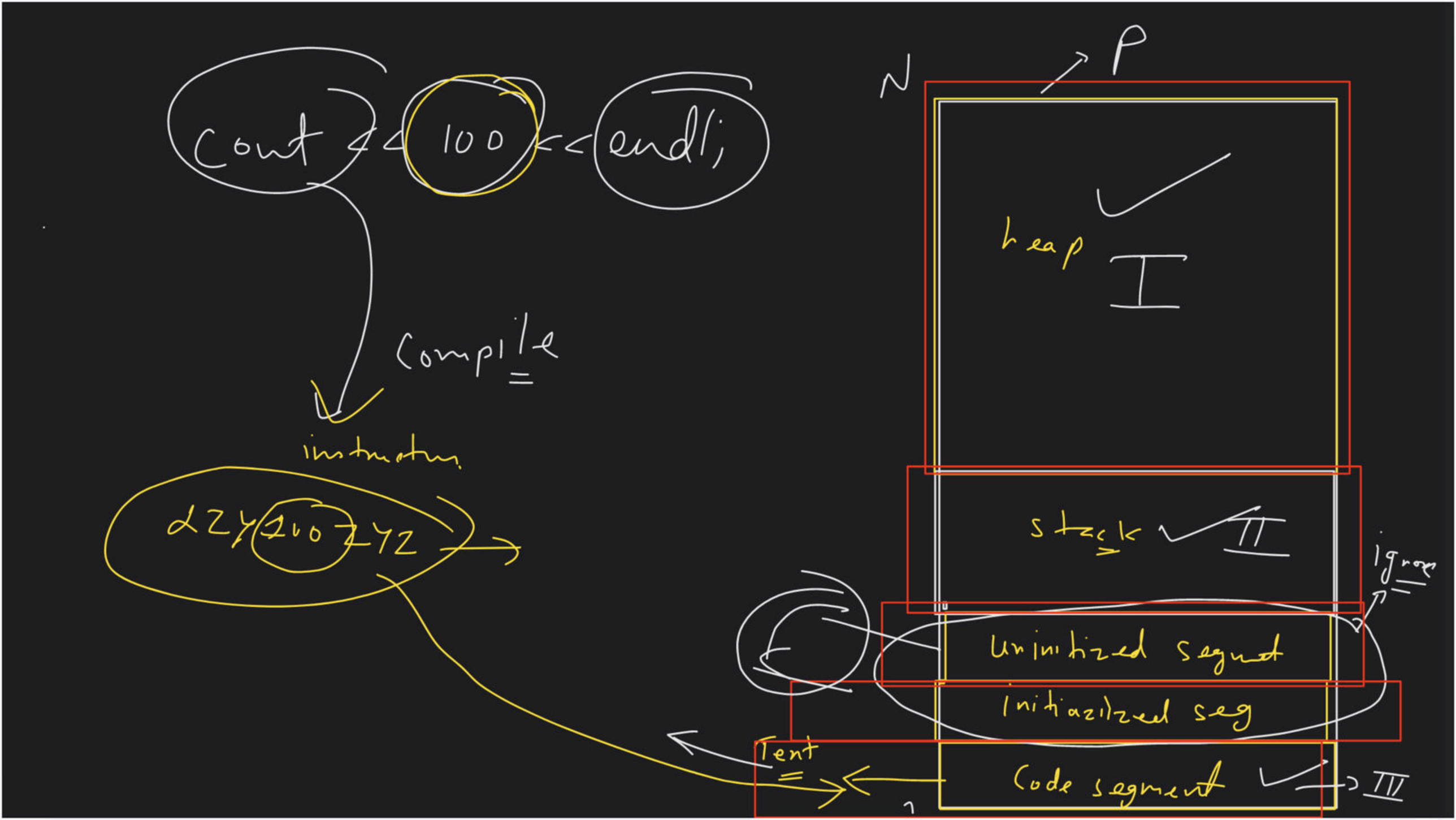
double a[10] = {0};

$\equiv$ {

→ sizeof(a) = ? 80 ⇒ A

→ sizeof(a[0]) = ? 8 ⇒ B

A/B ⇒ 10

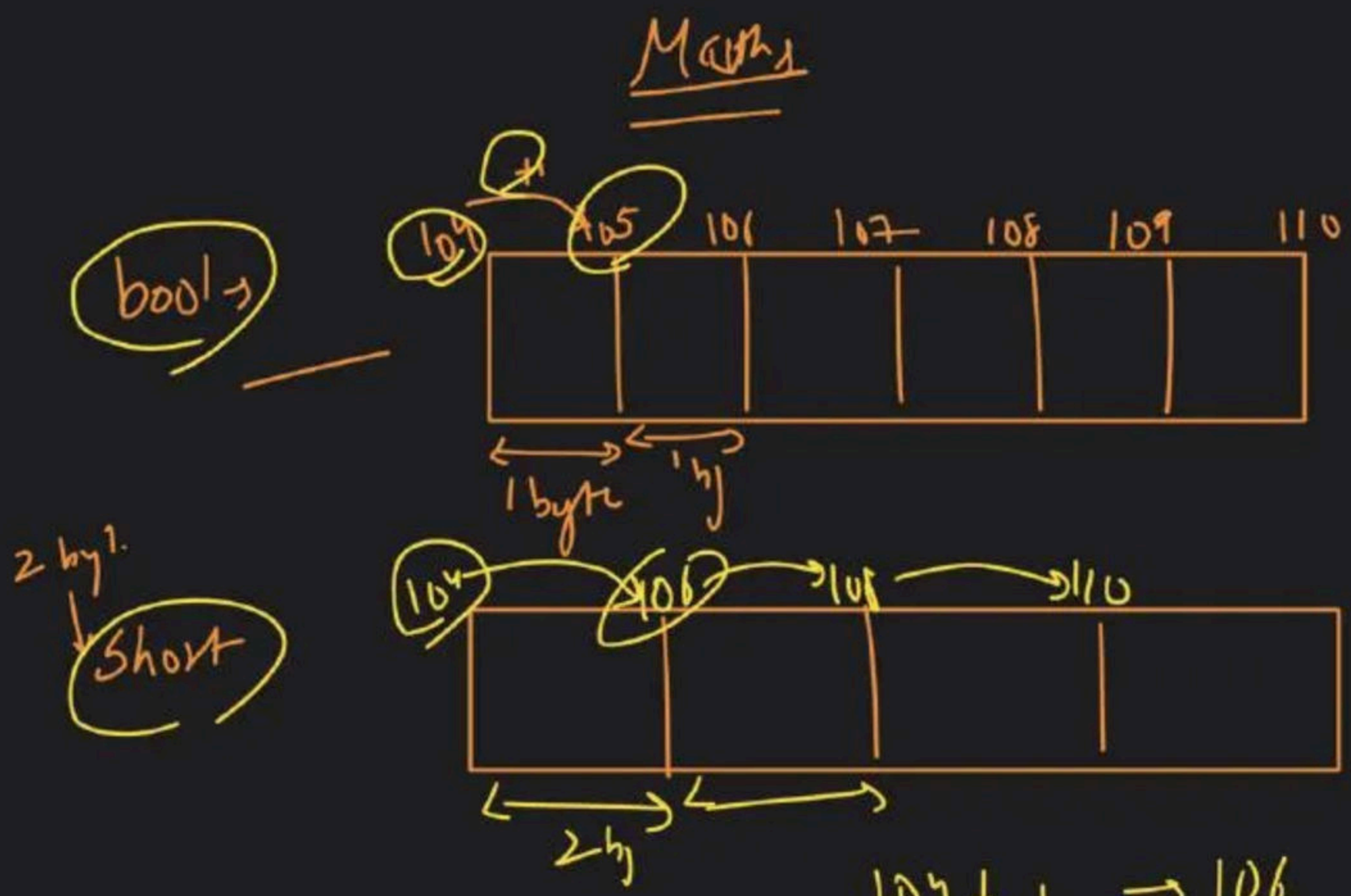

cout << 100 << endl;
Compile
instruction
mov ax, 100
N
P
heap
I
stack
II
Uninitized segment
Initiazilzed seg
Code segment
III
Ignore
Text

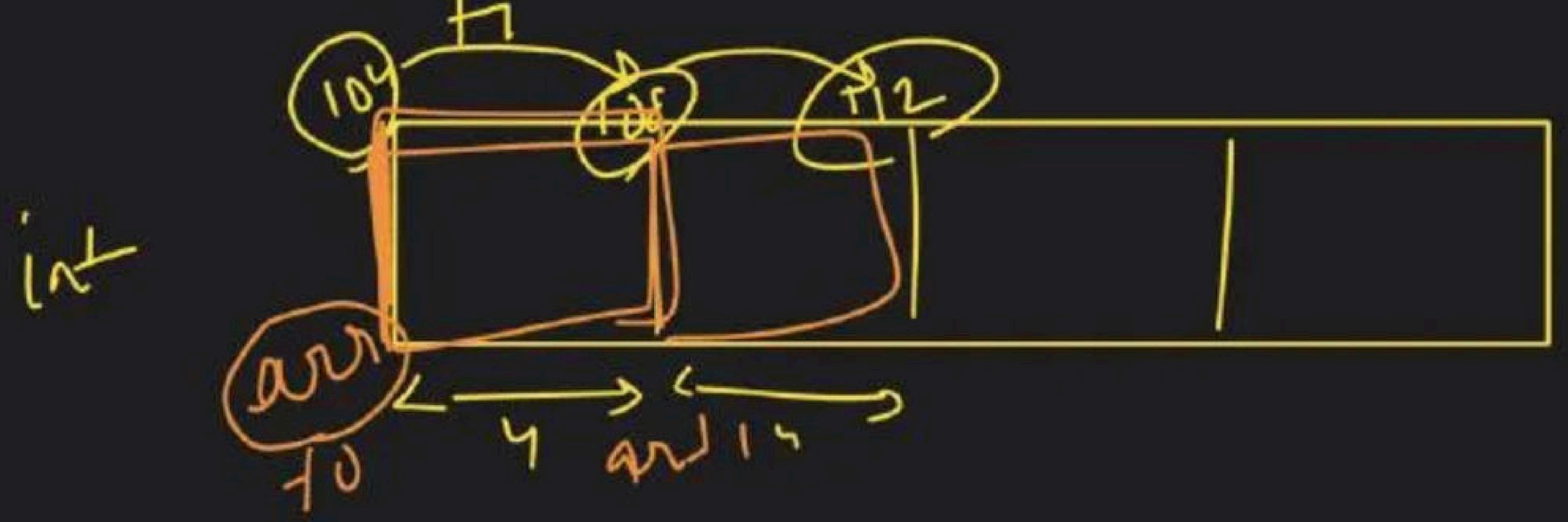
i[arr] → *(i+arr)

value present at address (3 + 104)

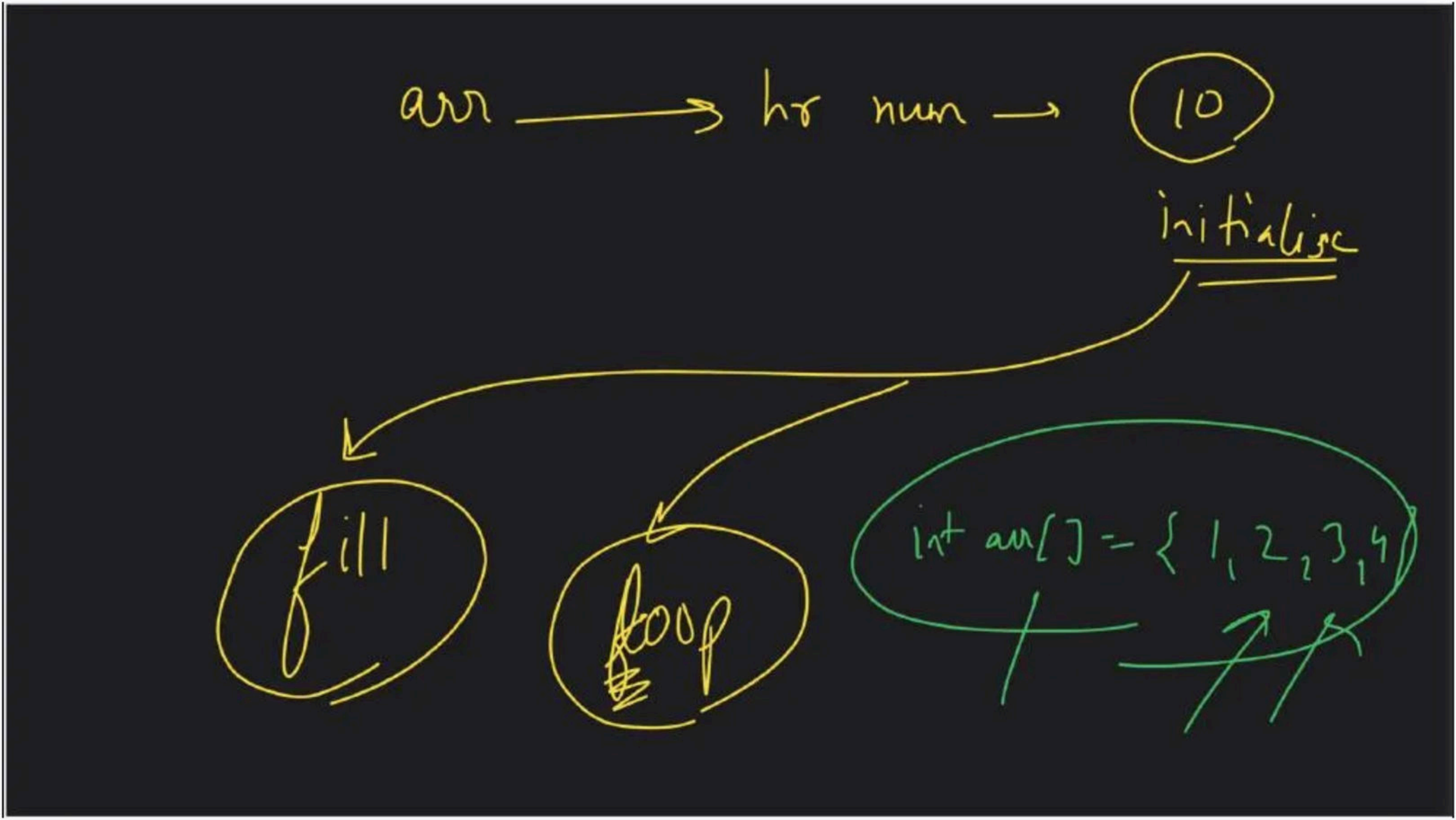
3
104
116

value present (116) at address


Main
bool
104
105
101
107
108
109
110
1byte
1byte
2byte
Short
104
106
108
110
2byte
104+1 → 106



104 +1 → 108


arr
hr
num
10
initialise
fill
loop
int arr[] = { 1, 2, 3, 4 }

pattern

i -> 1 2

i = 0

Indexing

1 -> n

n no

coach

initialization

fill

Indexing

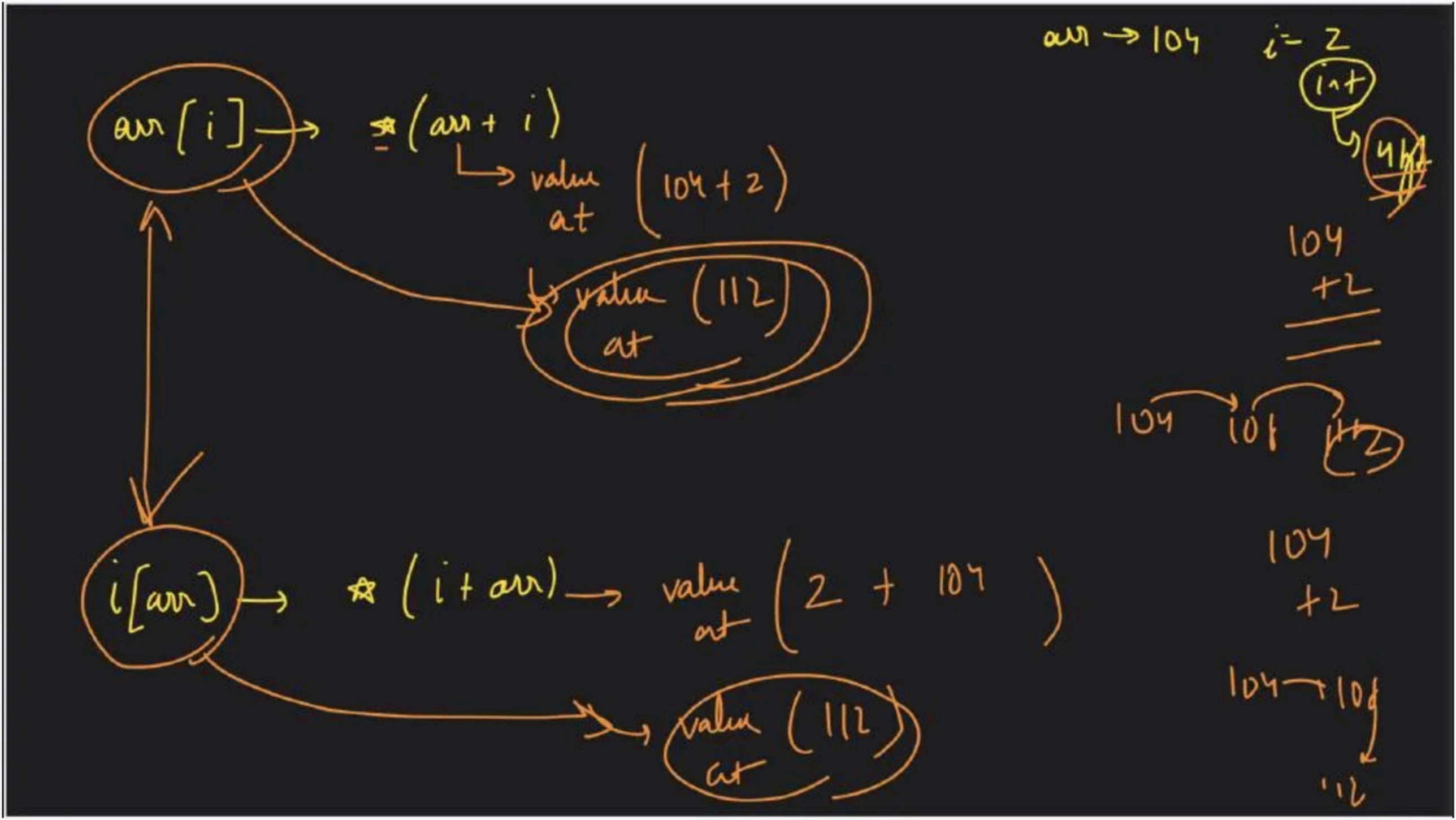
taking input

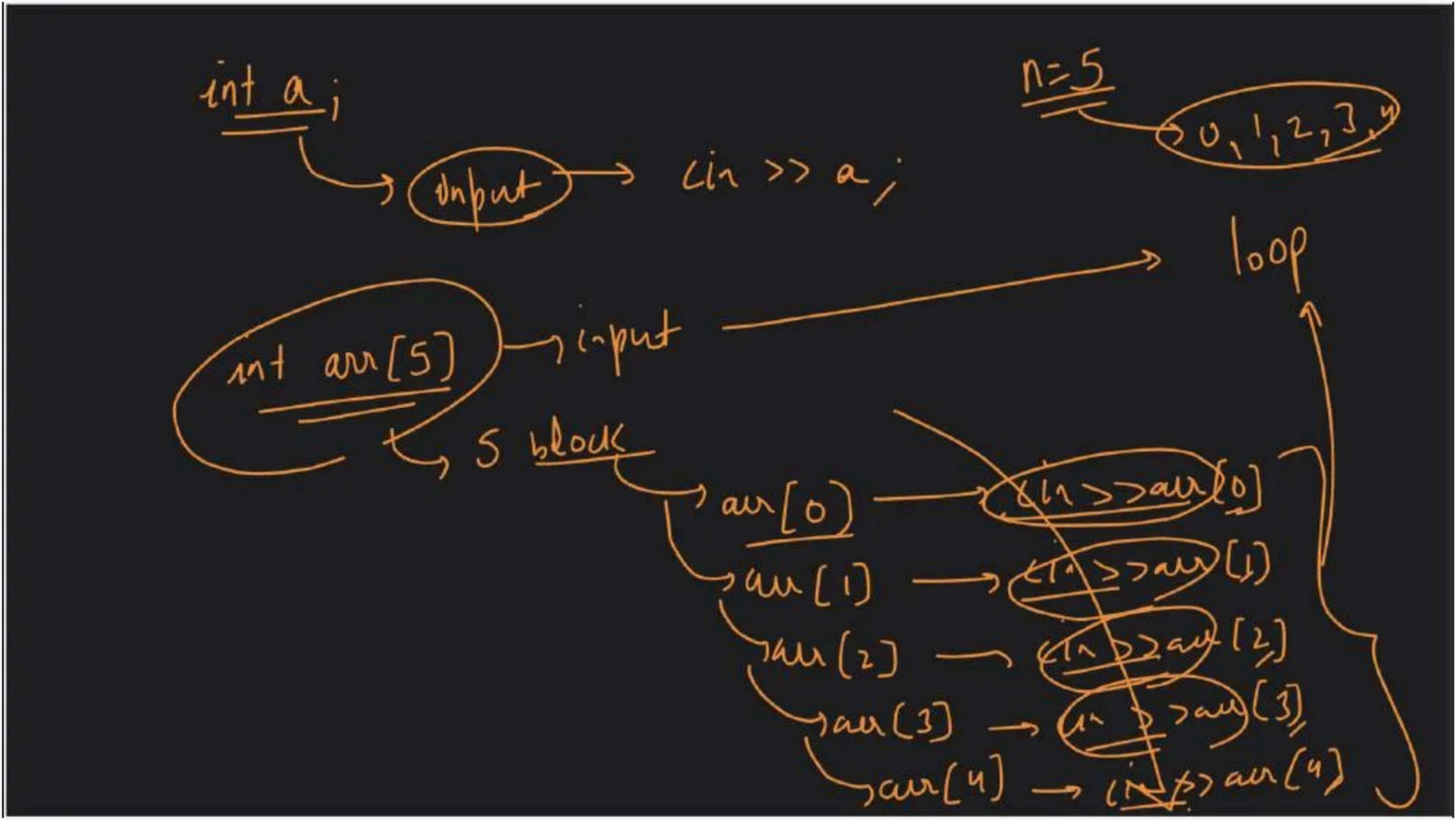
1 -> 5

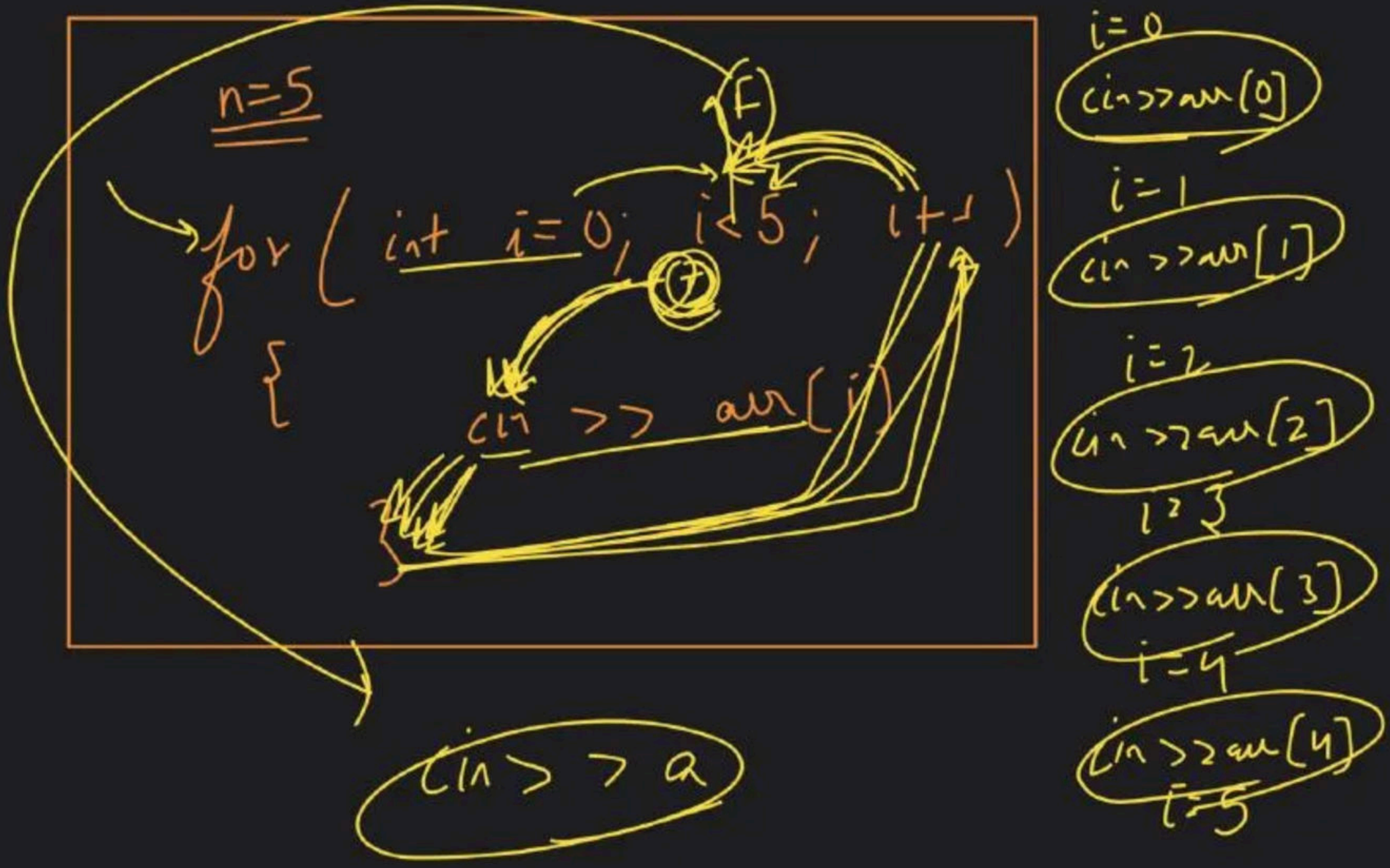
1 -> n

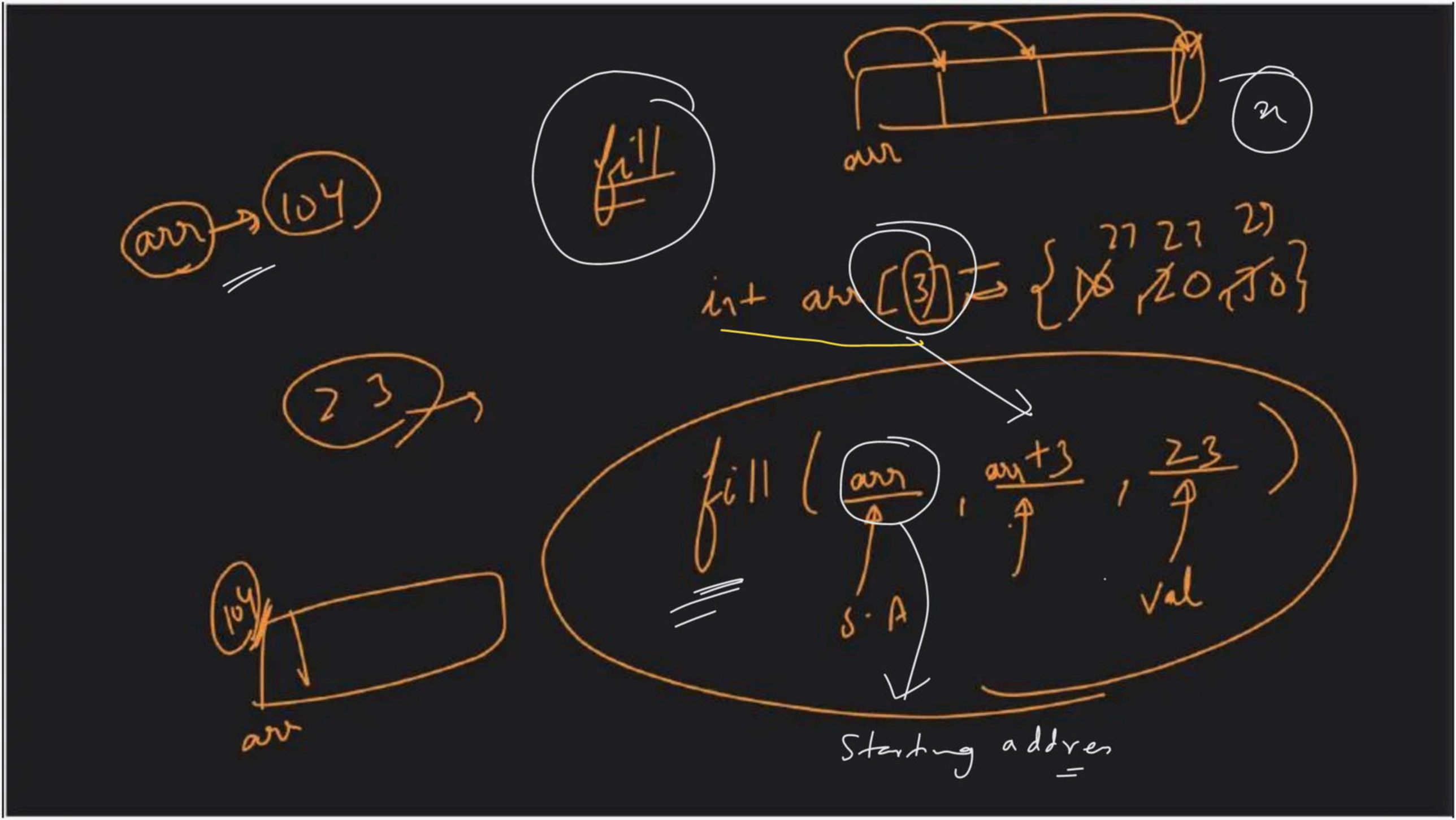
(-1) (-1)

0 -> n-1 -> n terms

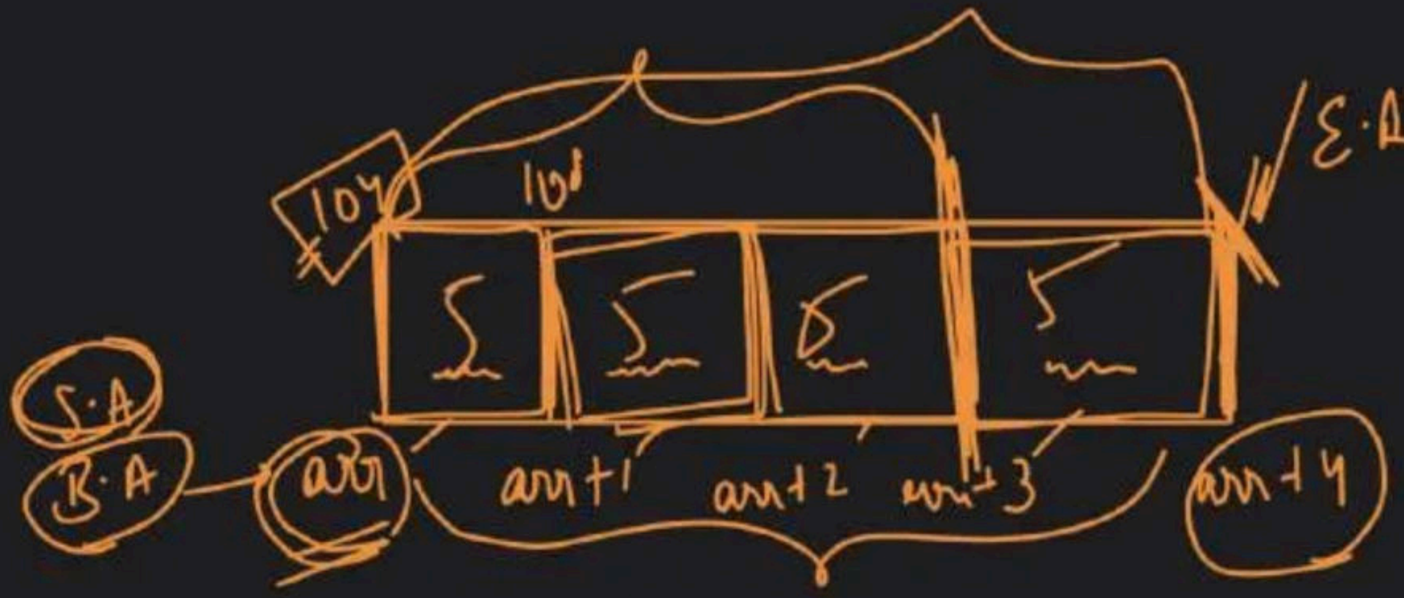

arr → 104
i = 2
int
arr[i] → *(arr + i)
value at (104 + 2)
value at (112)
104
+2
104 106
i[arr] → *(i + arr) → value at (2 + 104)
value at (112)
104
+2
104 → 106
112


int a;
input
cin >> a;
n=5
0, 1, 2, 3, 4
loop
int arr[5]
input
5 block
arr[0]
arr[1]
arr[2]
arr[3]
arr[4]
cin>>arr[0]
cin>>arr[1]
cin>>arr[2]
cin>>arr[3]
cin>>arr[4]


n=5
for (int i=0; i<5; i++)
{
cin >> arr[i]
F
T
cin >> a
i=0
cin>>arr[0]
i=1
cin>>arr[1]
i=2
cin>>arr[2]
i=3
cin>>arr[3]
i=4
cin>>arr[4]
i=5


arr
104
fill
arr
24
int arr[3] = {23, 23, 23}
23
fill (arr , arr+3 , 23)
S.A
Val
104
arr
Starting address

int arr[4]



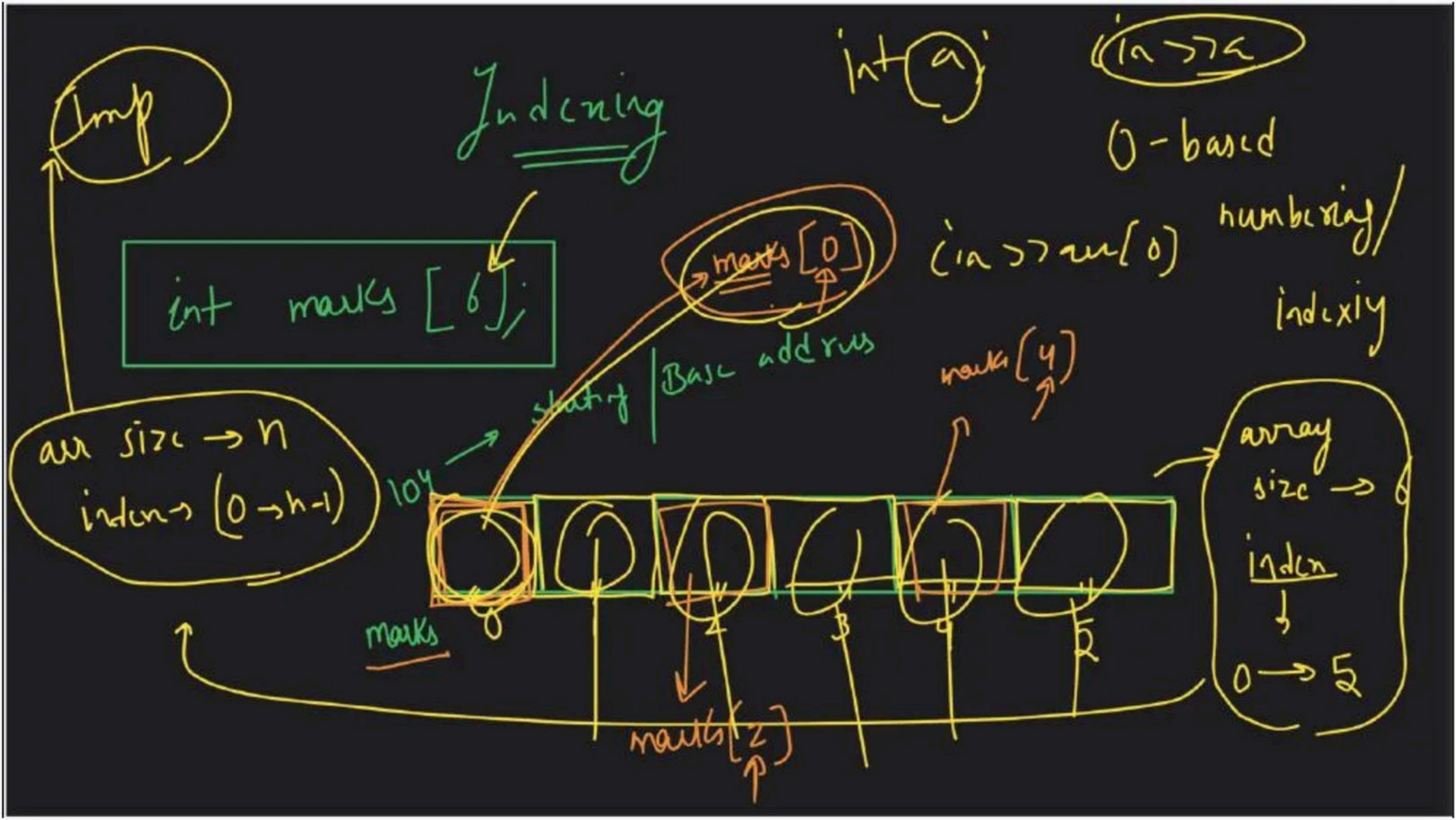
(5) Sc initialize hr block ko

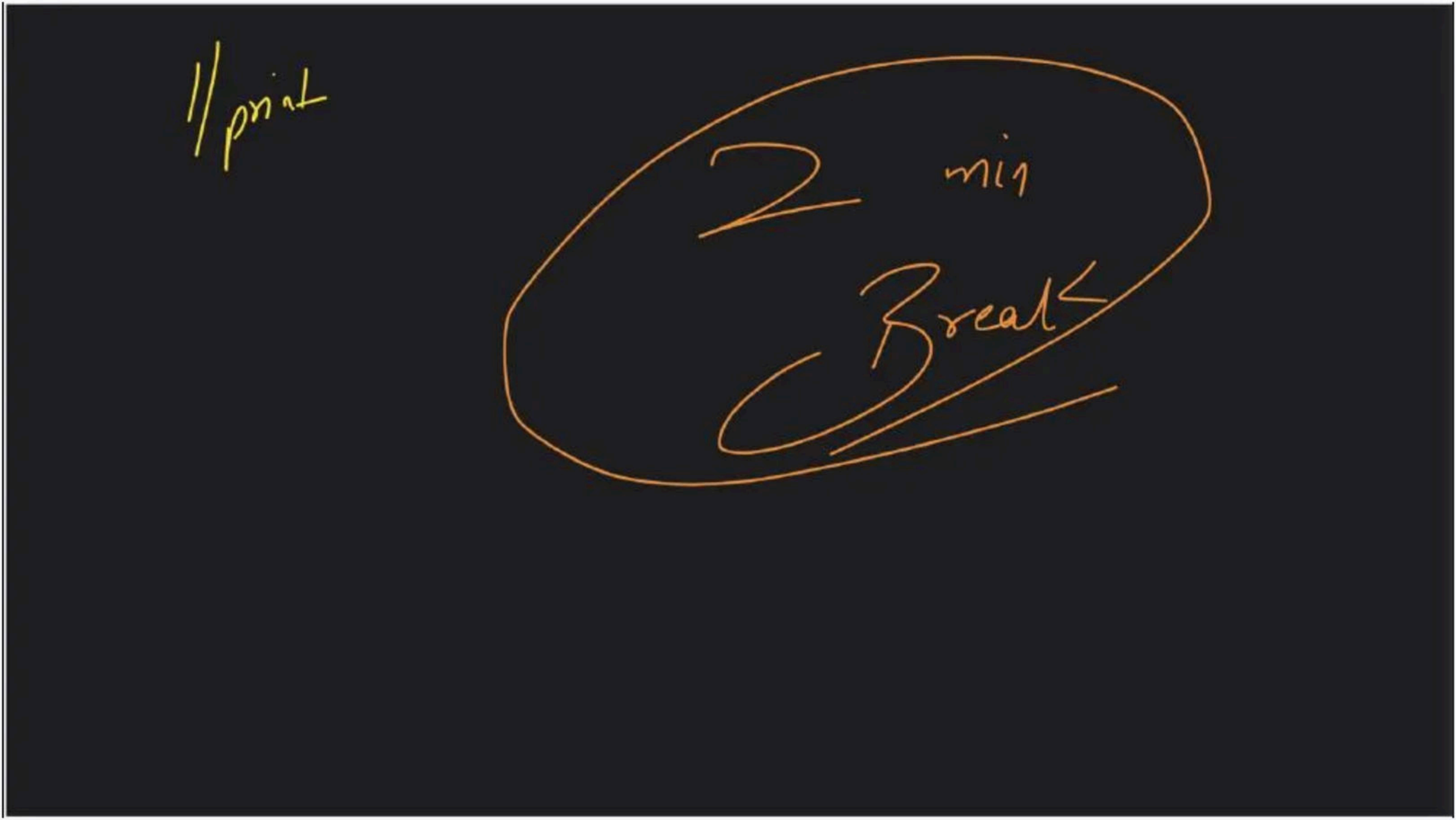
fill (arr , arr+4 , 5)

arr → S.A

arr+4 → E.A

5 → Val


Imp
Indexing
int a;
cin >> a
0-based
numbering/
indexing
int marks [6];
marks [0]
cin >> arr[0]
Base address
starting
104
marks [4]
arr size → n
index (0 → n-1)
array
size →
index
0 → 5
marks
marks [2]


1/print
2 min
Break

→ Functions with Array

```
int main()
{
    int arr[5];
    int a=5;
    solve(a);
}
```

```
solve(arr);
```

```
int solve(int a)
{

}
```

int arr[]

```
int main()
{
    int arr[5] = {0};
    solve(arr, n);
}
```

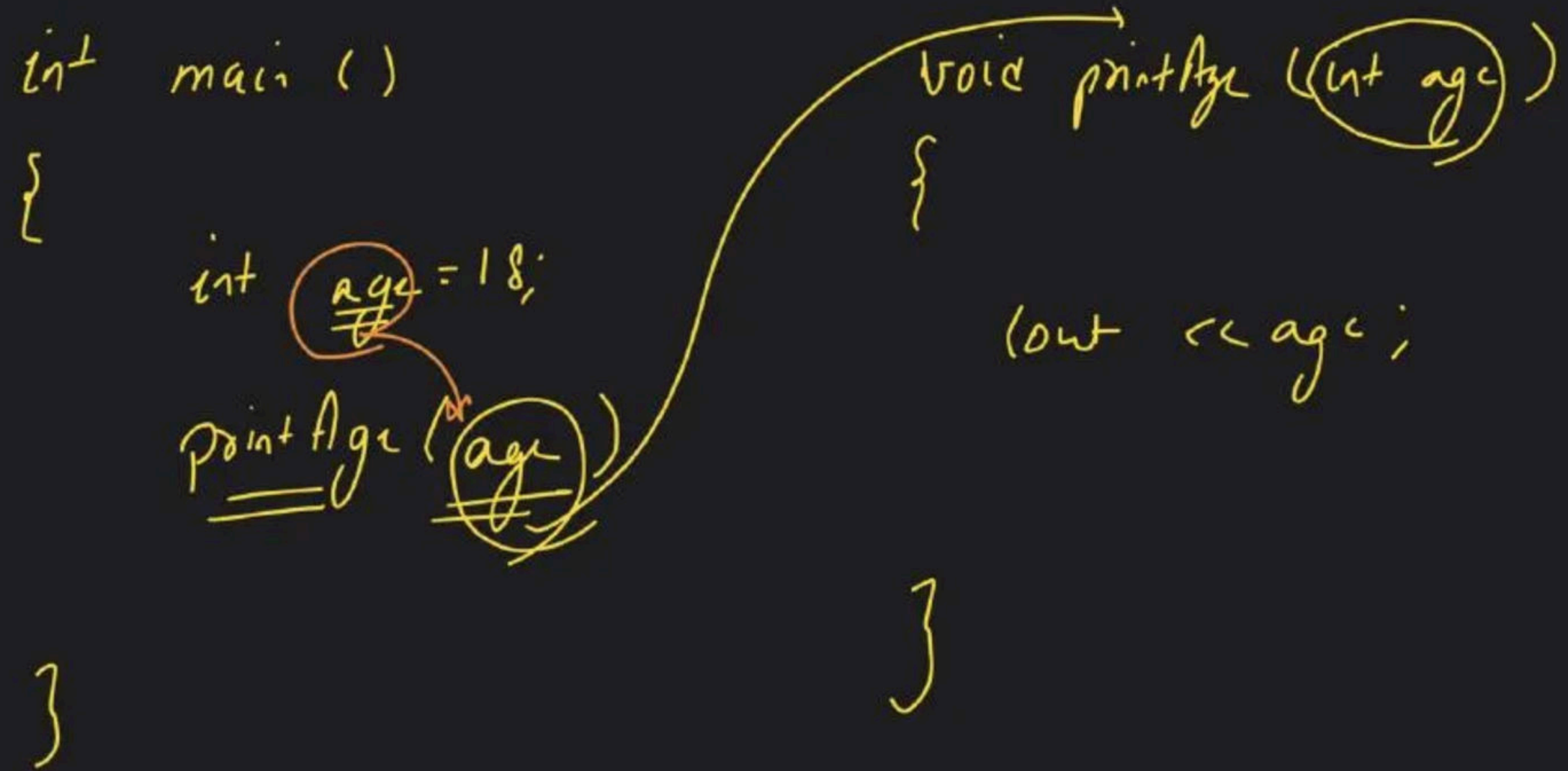
size

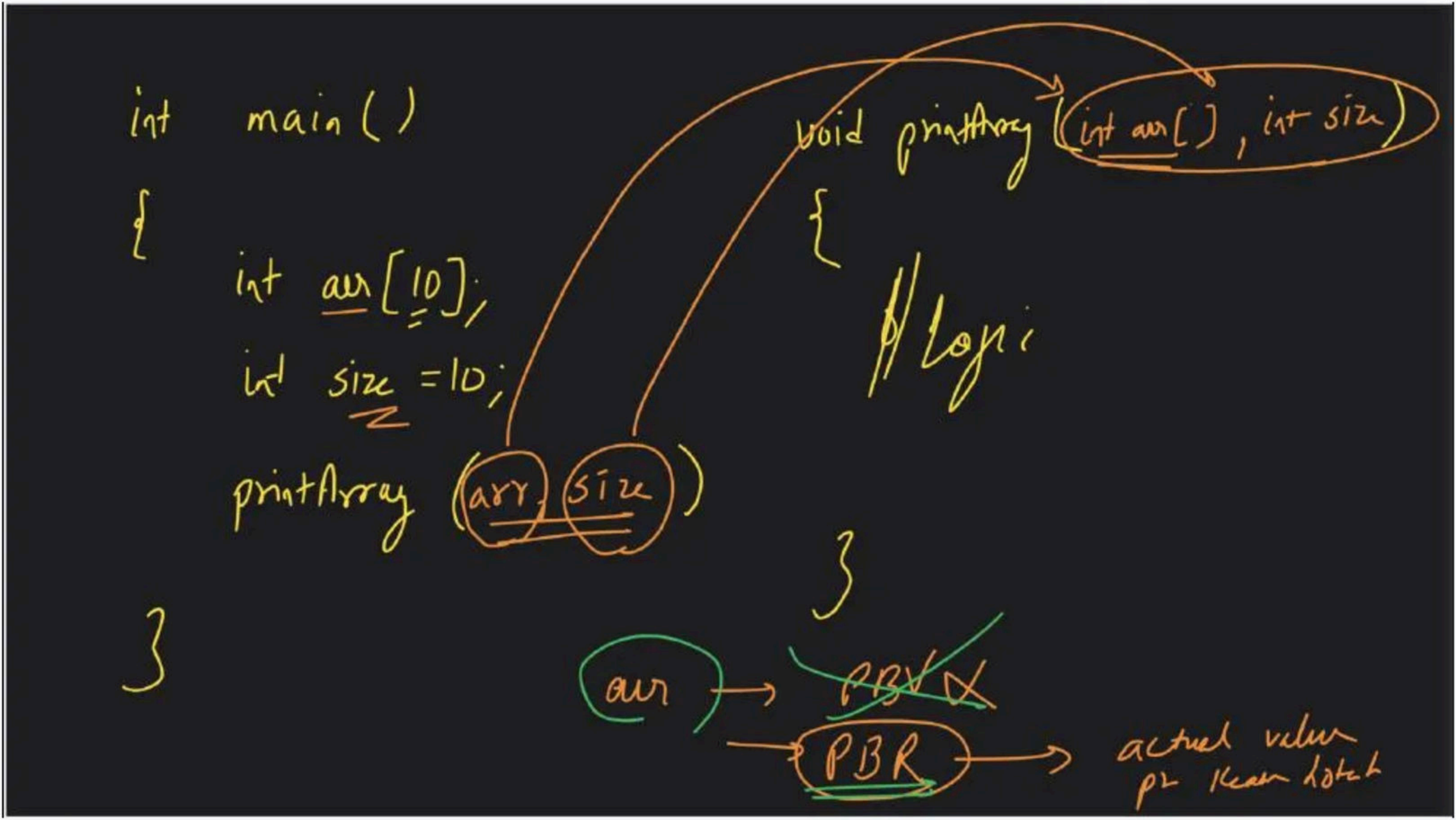
solve(arr, size)

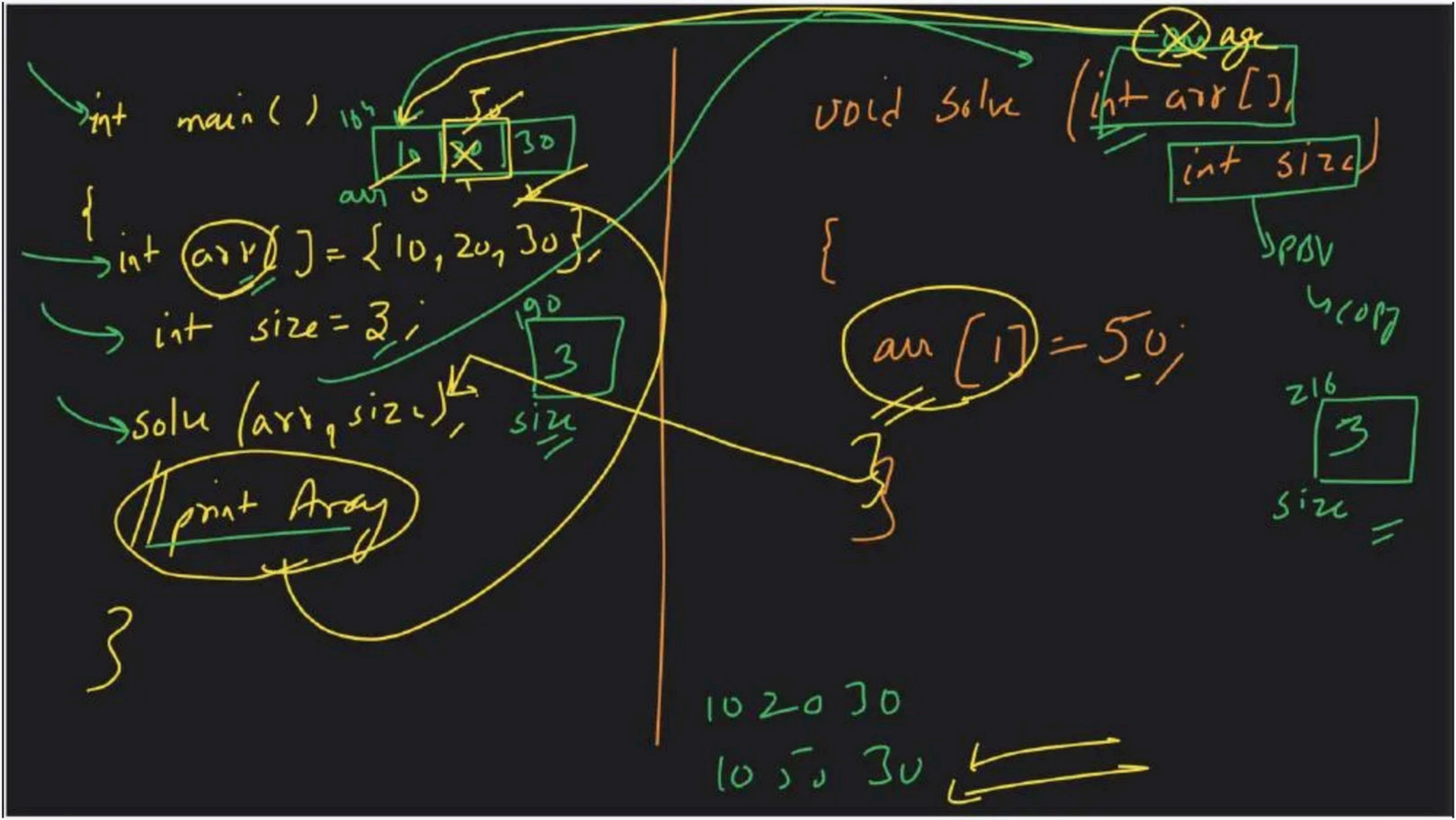
```
int solve(int arr[])
{
}
```

int size

solve(int arr[], int size)


int main ()
{
int age = 18;
printAge (age)
}
void printAge (int age)
{
cout << age;
}

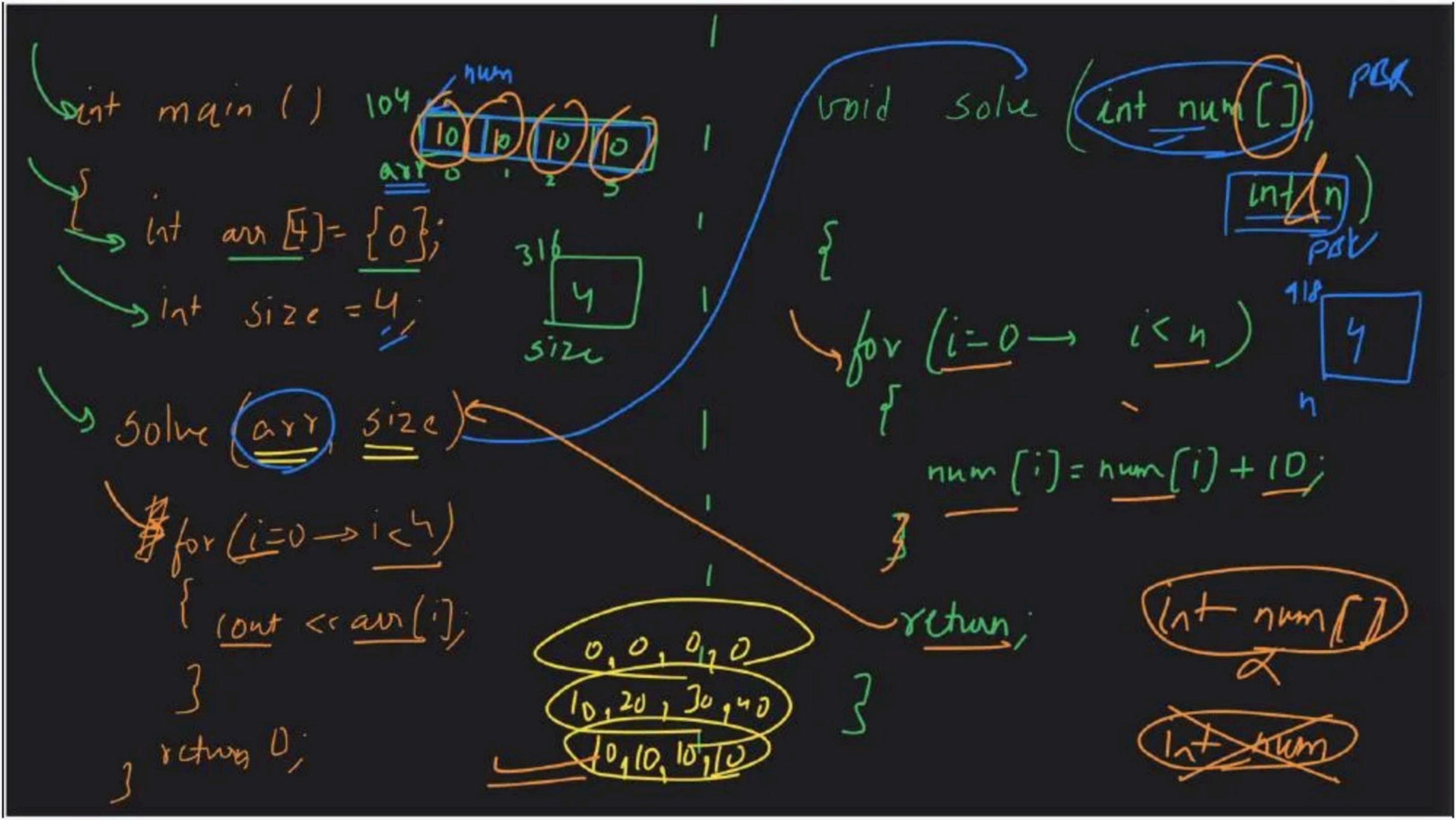

int main()
{
int arr[10];
int size = 10;
printArray (arr, size)
}
void printArray (int arr[], int size)
{
// Logic
}
arr → PBV ✗
PBR → actual value
pt Kaam hoteh


int main ( )
{
int arr[ ] = {10, 20, 30};
int size = 3;
Solu (arr, size);
print Array
}
arr
size
100
3
10 20 30
10 50 30
void Solu (int arr[],
int size)
{
arr[1] = 50;
}
pbv
copy
216
3
size

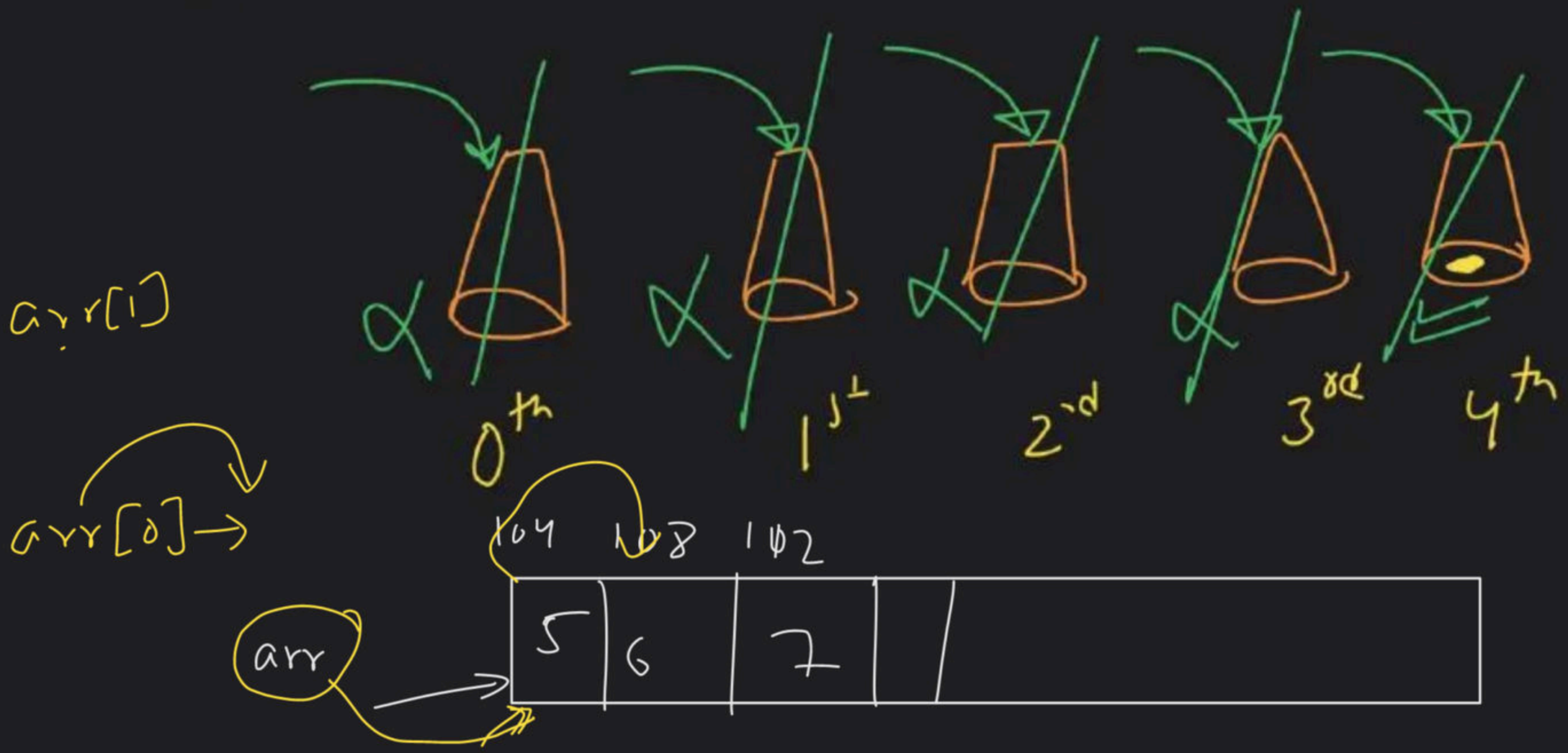
Pass by Reference → Last Class

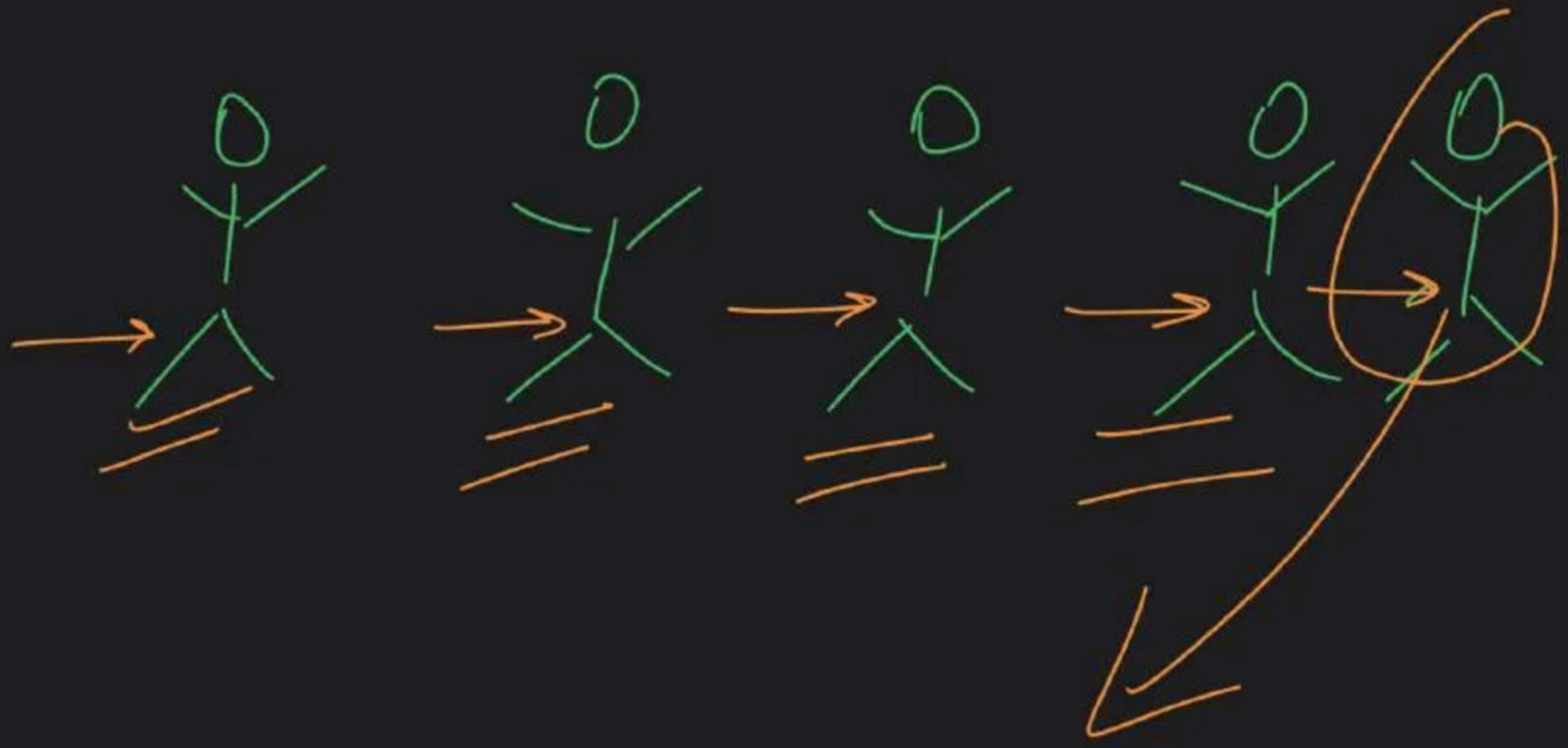
↓

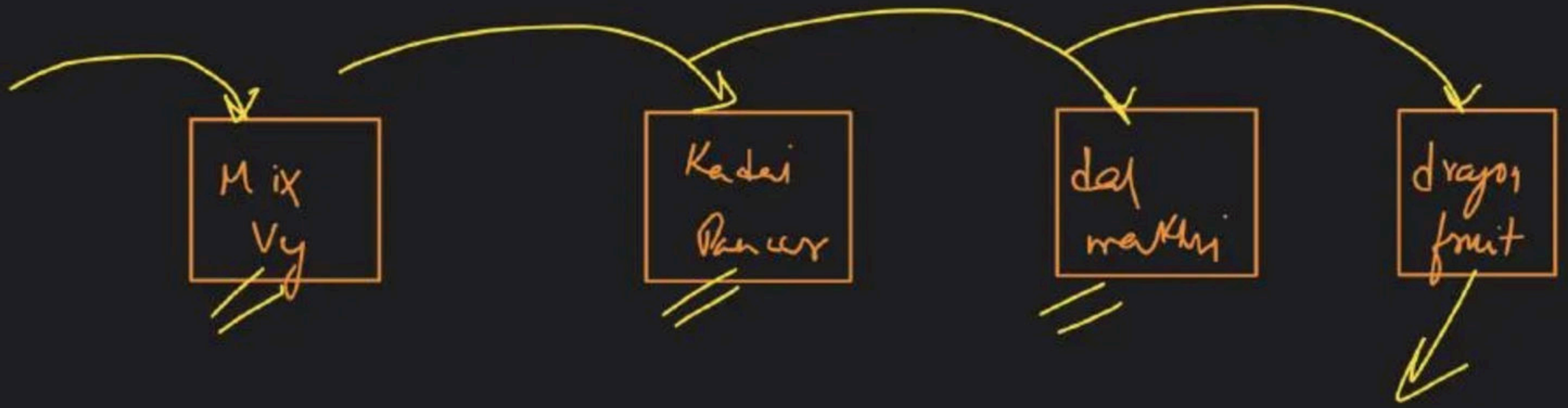
Re-visit


int main ()
{
int arr[4]= {0};
int size =4;
Solve (arr, size)
for (i=0 → i<4)
{ cout << arr[i];
}
return 0;
}
num
104
10 10 10 10
arr 0 1 2 3
316
4
size
void Solve (int num[], int n)
{
for (i=0 → i<n)
{
num[i] = num[i] + 10;
}
return;
}
PBR
PBR
718
4
n
0, 0, 0, 0
10, 20, 30, 40
10, 10, 10, 10
int num[]
int num

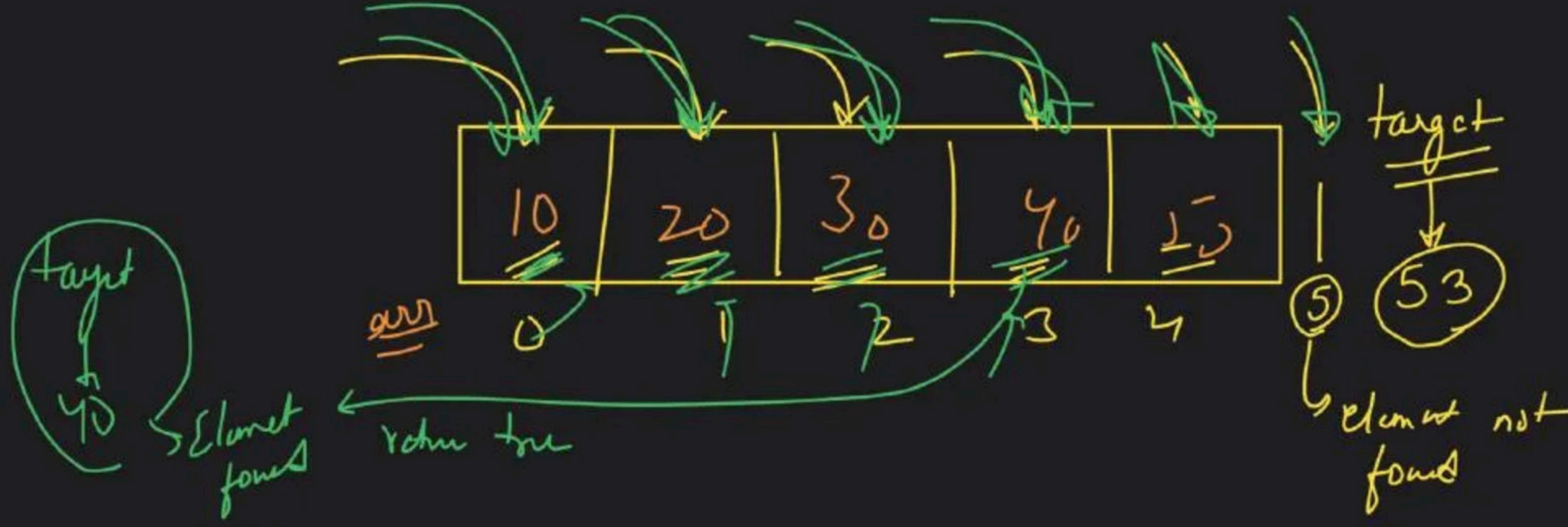
① Algorithm → Linear Search

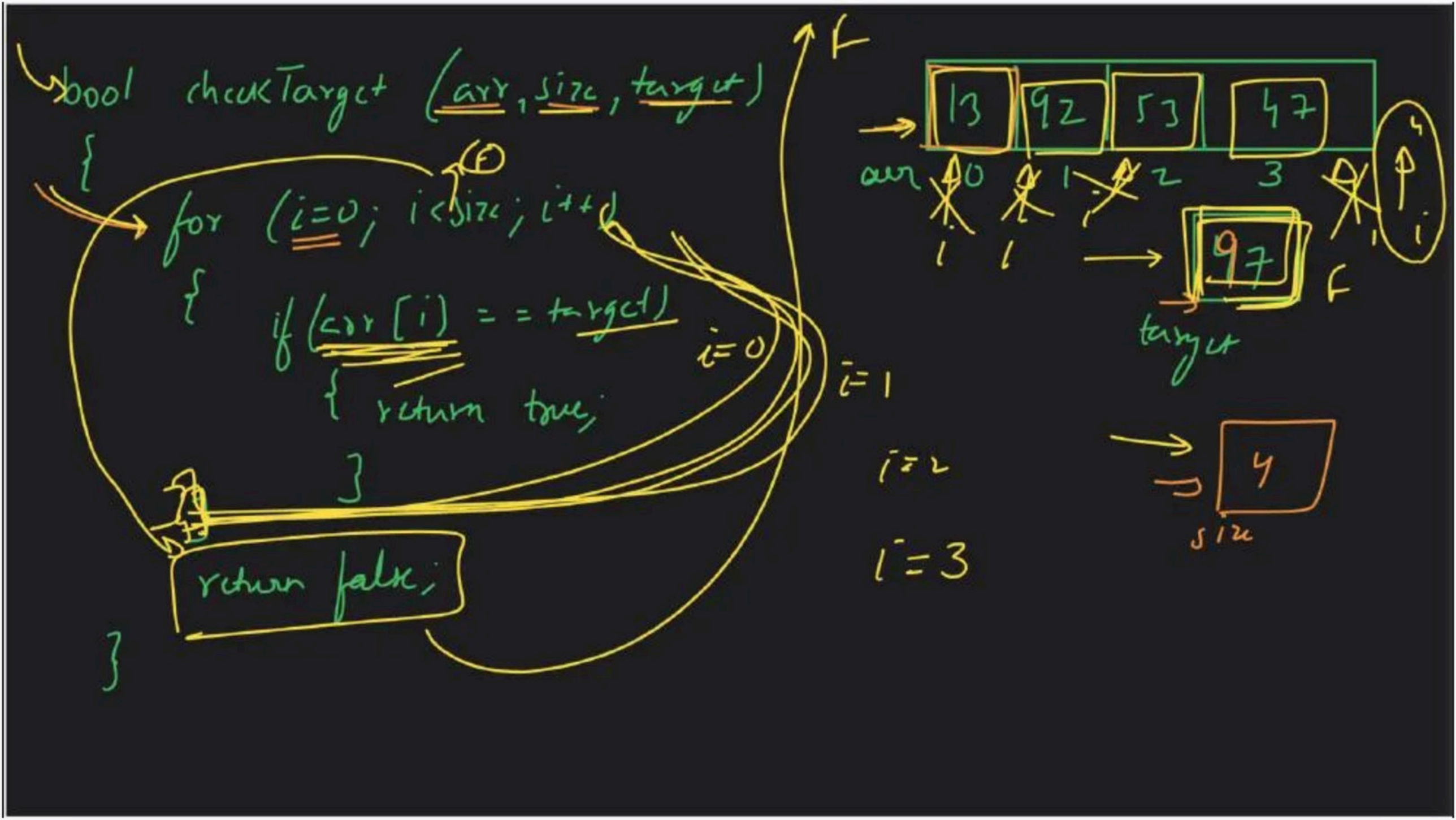


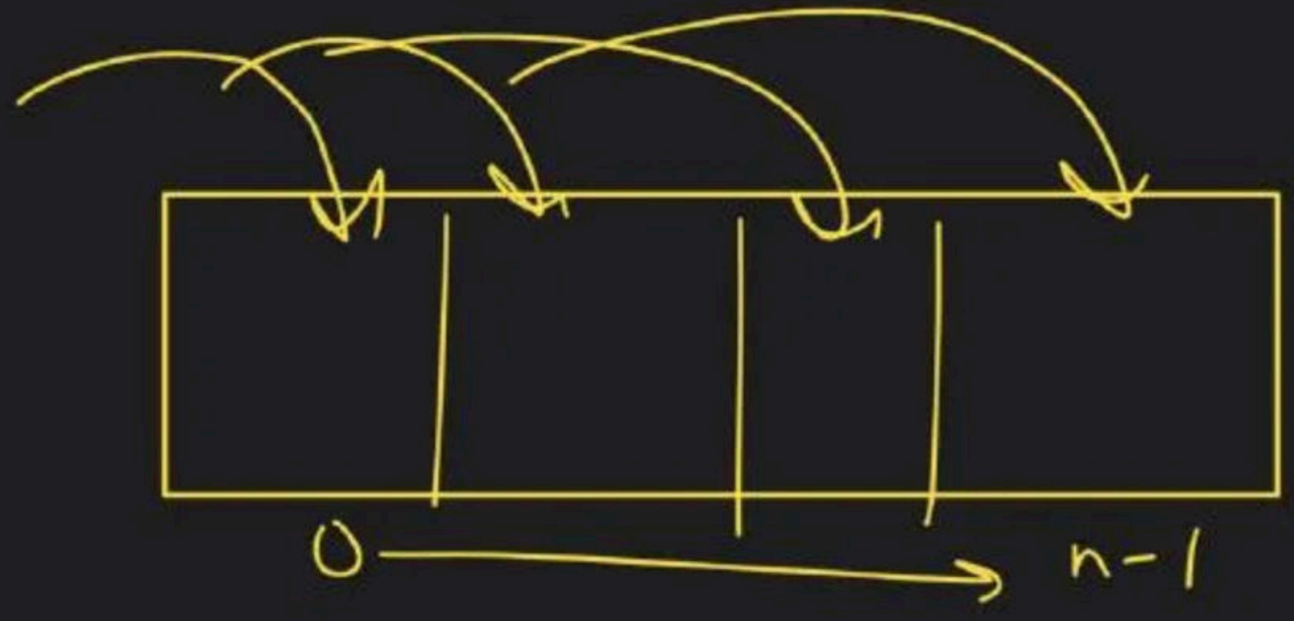


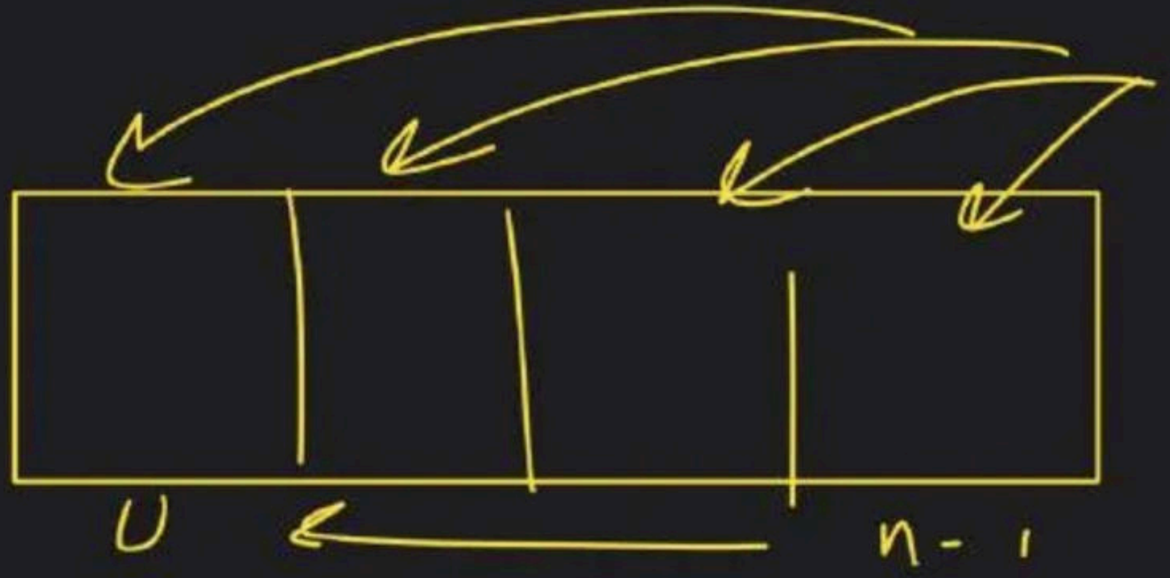

Mix
Veg
Kadai
Paneer
dal
makhni
dragon
fruit

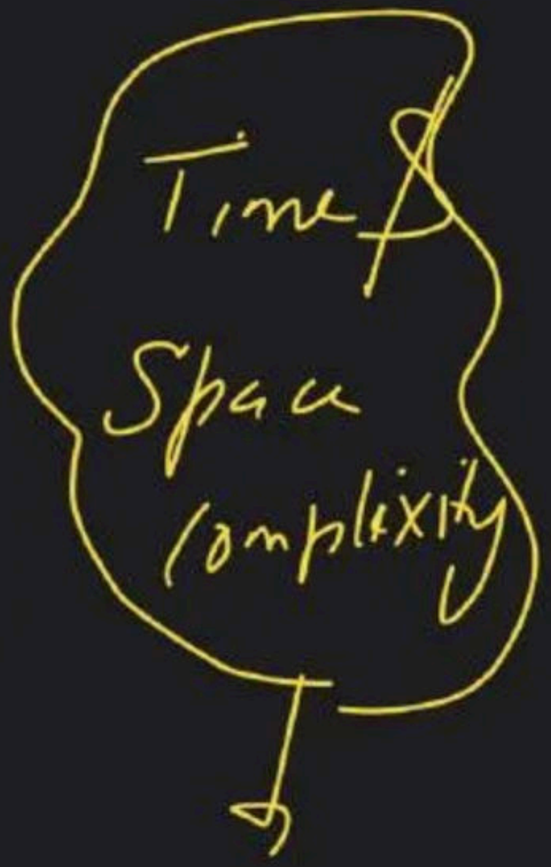
array → int arr[5] → {10, 20, 30, 40, 50}

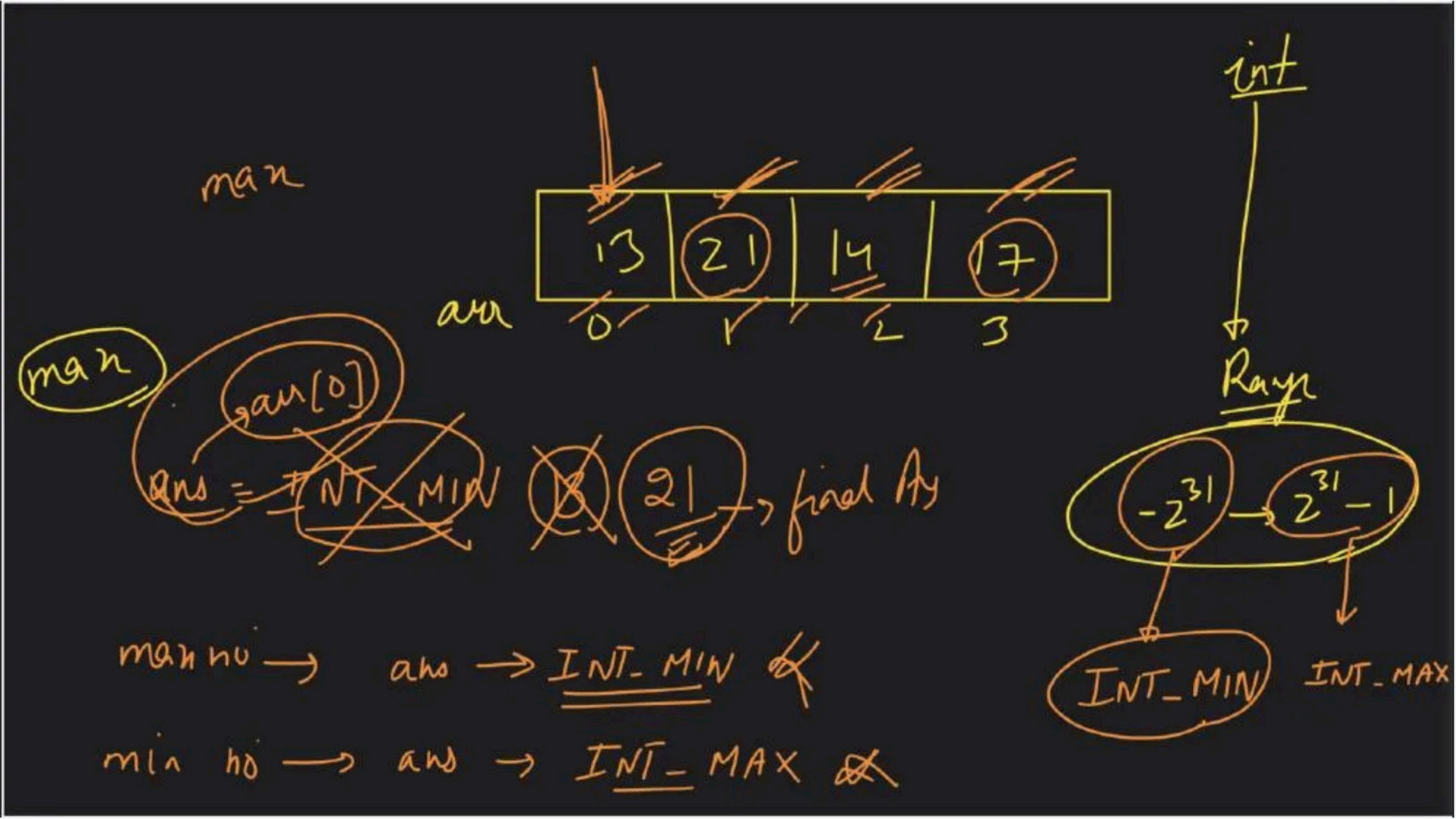



bool checkTarget (arr, size, target)
{
for (i=0; i<size; i++)
{
if (arr[i] == target)
{ return true;
}
return false;
}
i=0
i=1
i=2
i=3
13
92
53
47
arr
0
1
2
3
i
i
97
target
F
4
size


0
n-1

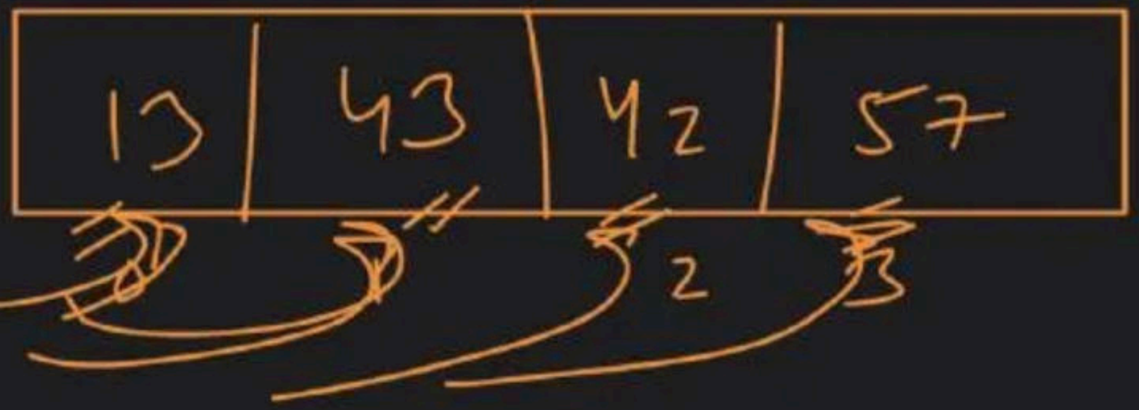

0
n-1


Time &
Space
complexity


int
Range
-2^{31}
$2^{31}-1$
INT_MIN
INT_MAX
max
arr
13
21
14
7
0
1
2
3
max
arr[0]
ans = INT_MIN
21
final Ans
max no → ans → INT_MIN
min no → ans → INT_MAX

arr[i] > ans ⟶ ans = arr[i]

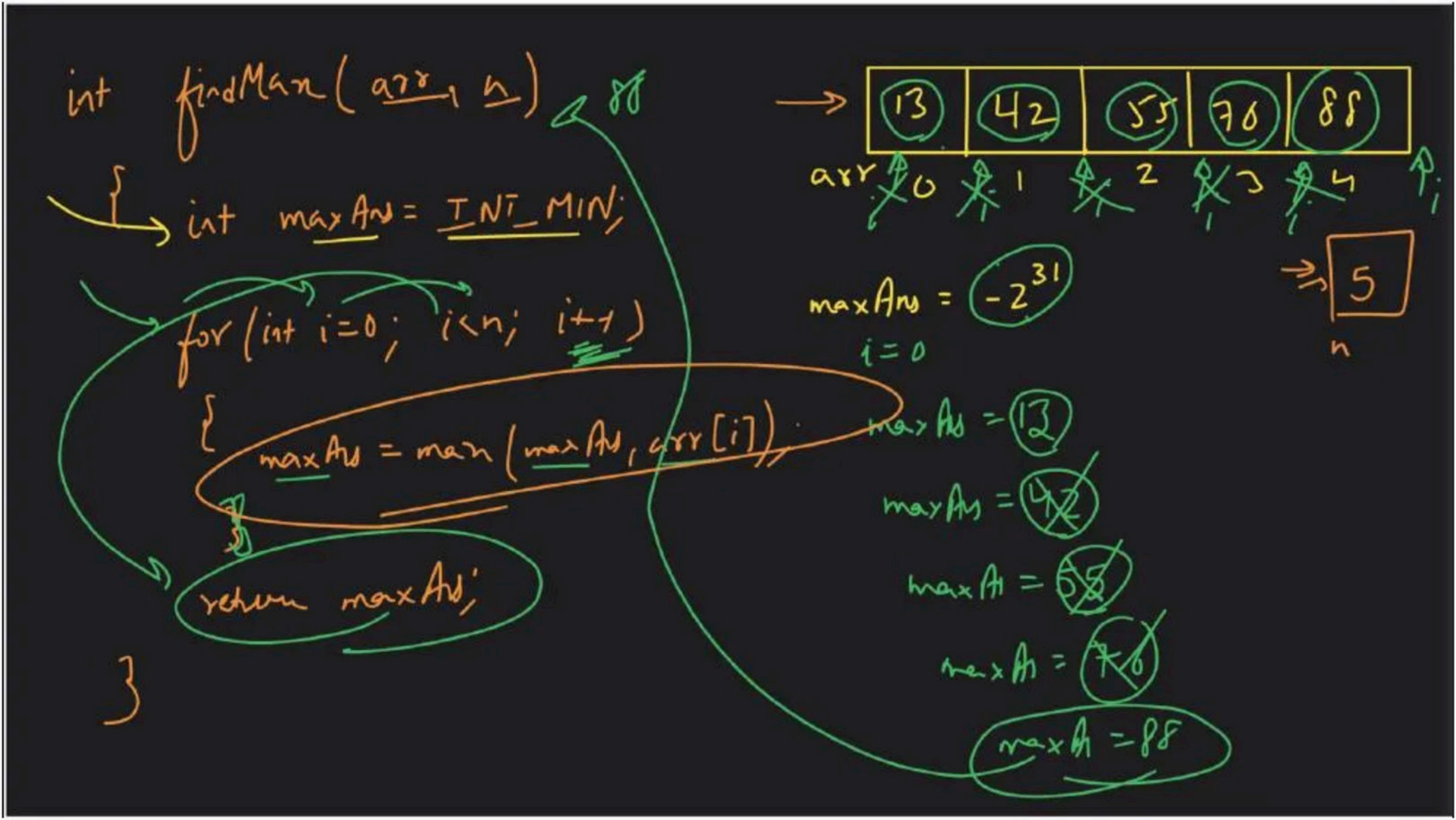
arr[i] < ans ⟶ ignore



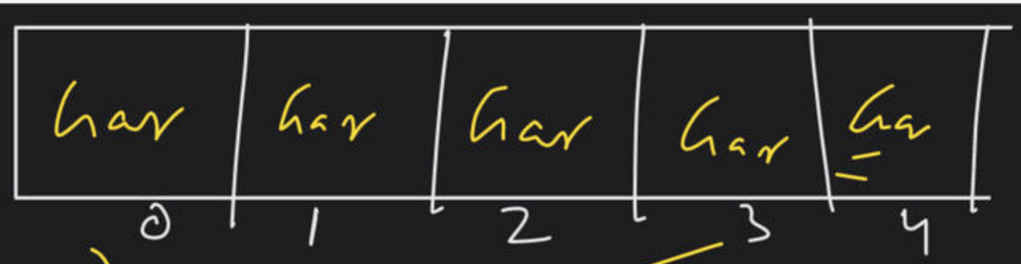
man

ans = INT_MIN

57 → finish


int findMax(arr, n)
{
int maxAns = INT_MIN;
for(int i=0; i<n; i++)
{
maxAns = max(maxAns, arr[i]);
}
return maxAns;
}
13
42
55
70
88
arr
0
1
2
3
4
i
n
5
maxAns = -2^31
i=0
maxAns = 13
maxAns = 42
maxAns = 55
maxAns = 70
maxAns = 88

int arr[5];



fill(arr, arr+5, 7)

arr → address

fill(arr[0], arr+5, 7)

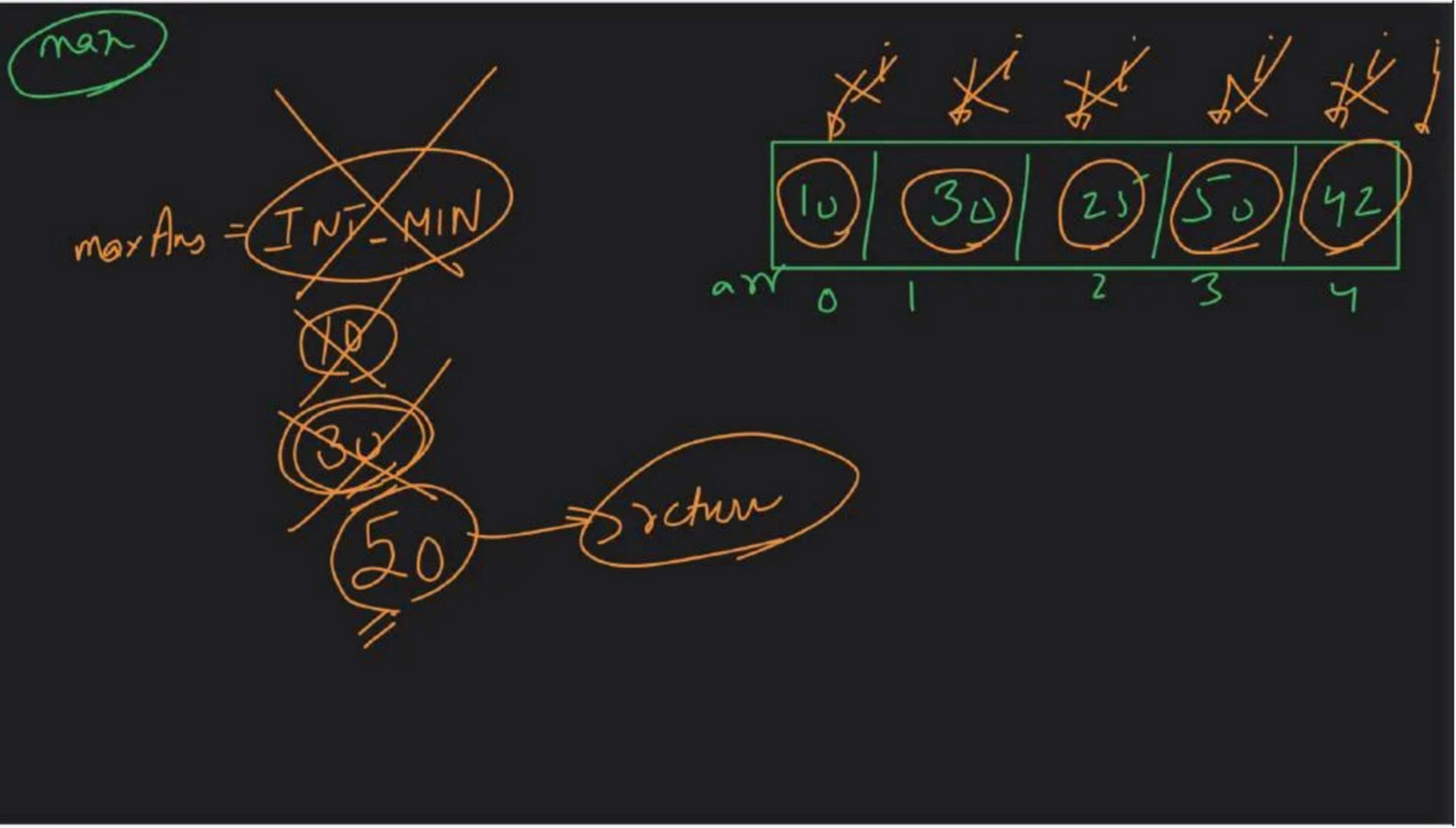
garbage

X

arr[0]

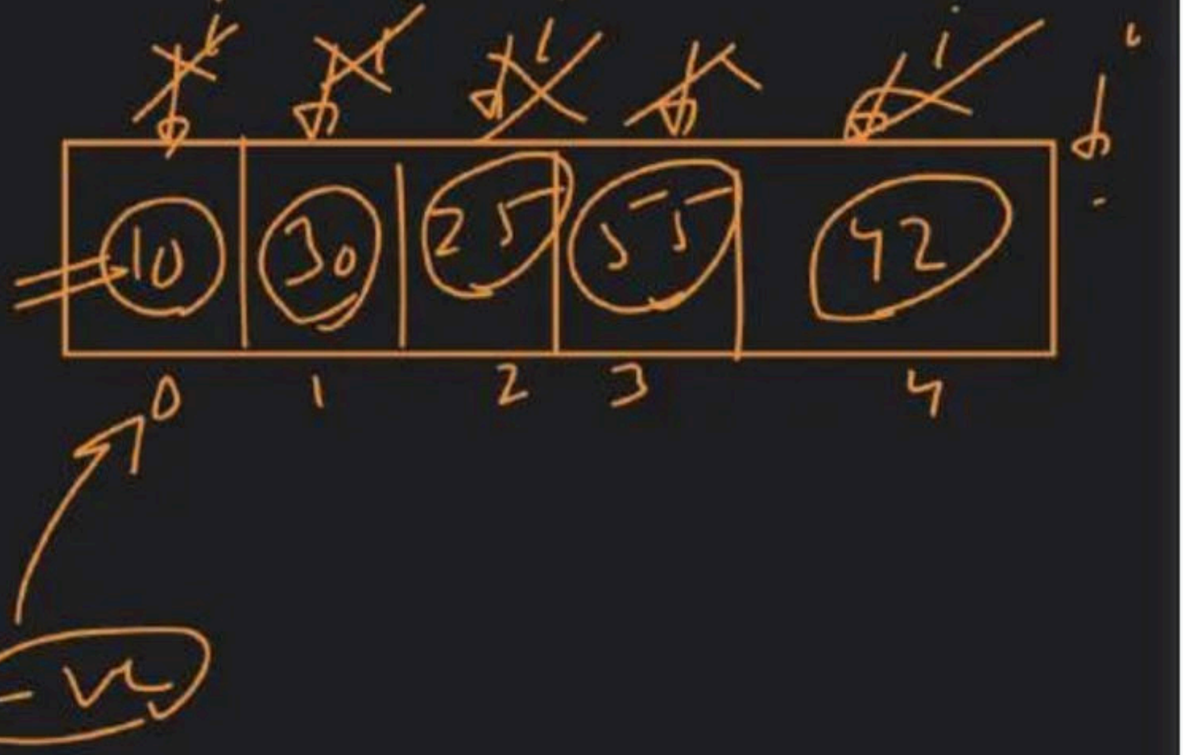
Value At [0]

```
int findMin(arr, n)
{
    int minAns = INT_MAX;
    for (i=0; i<n; i++)
    {
        minAns = min(minAns, arr[i])
    }
    return minAns;
}
```


max
maxAns = INT_MIN
10
30
50
return
10
30
25
50
42
arr
0
1
2
3
4

maxAns = arr[0]

maxAns = 10 30 55) ans



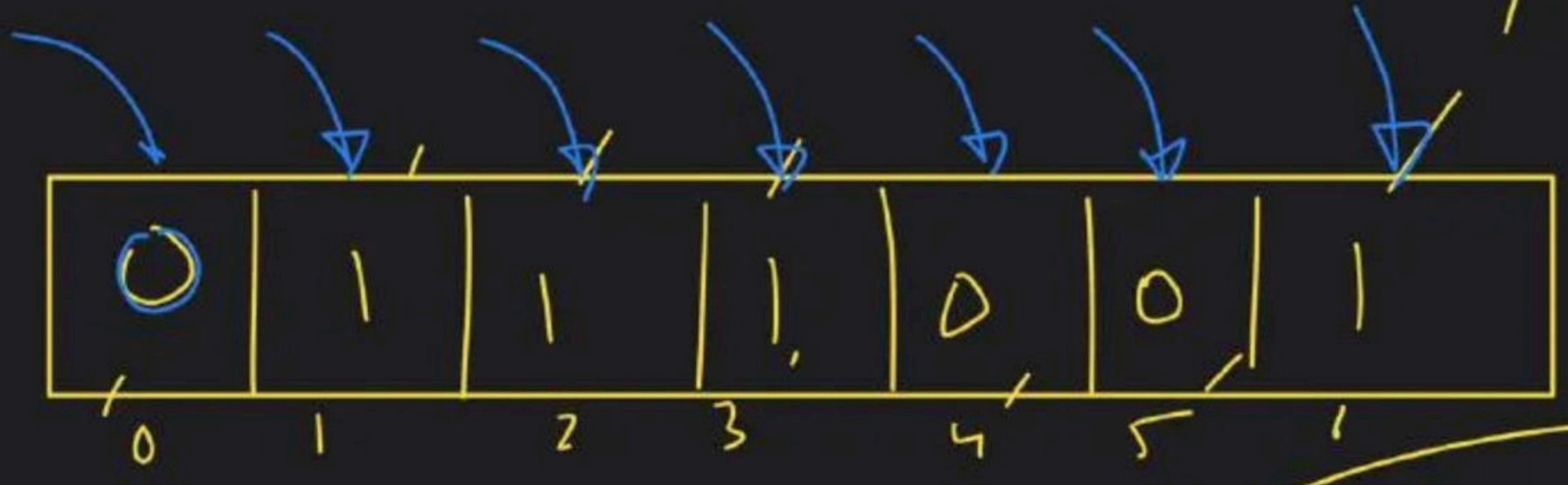
+ve

-ve


Baatein
Pattern

→ i/p → array

count → 0's / 1's

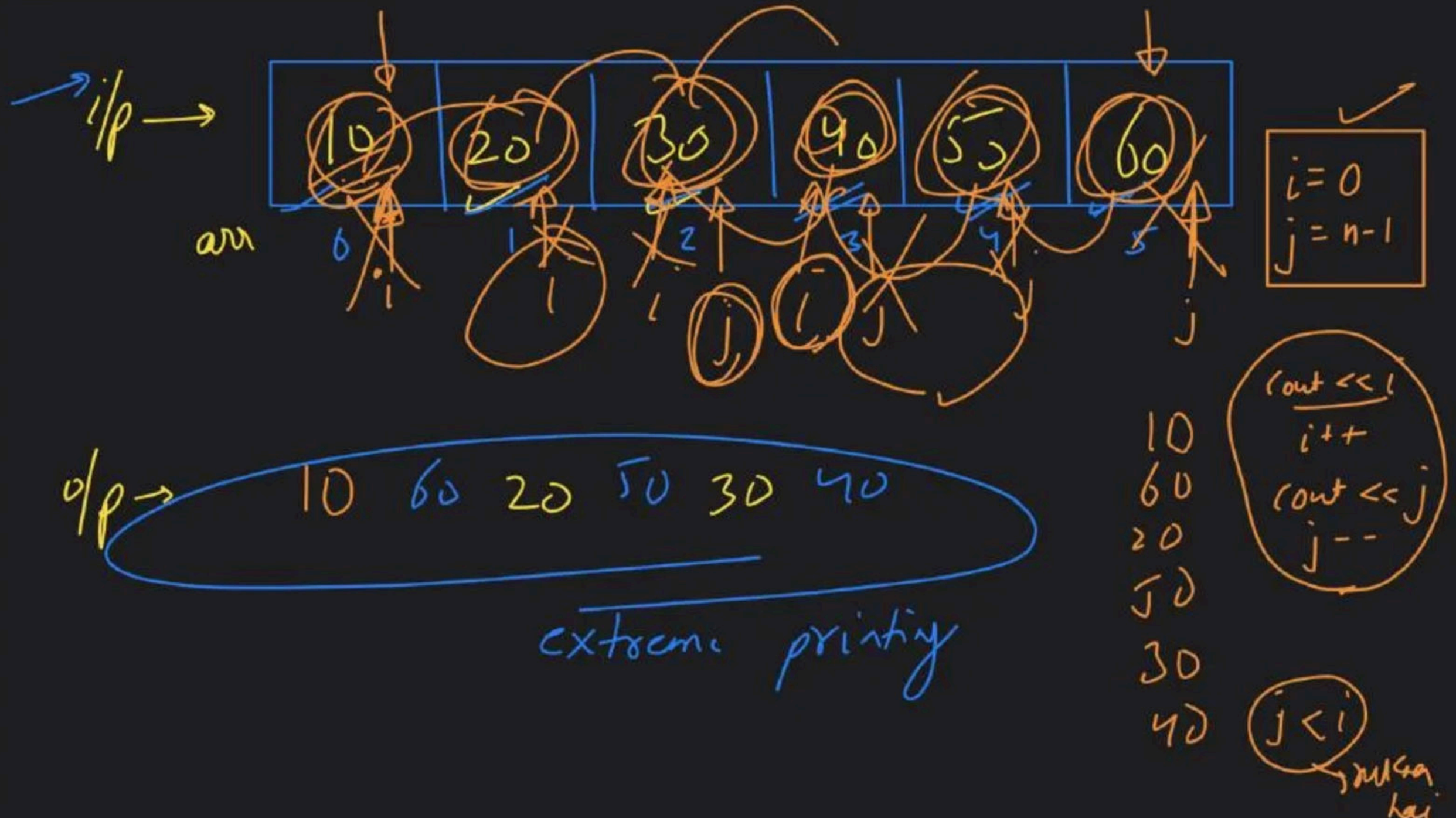


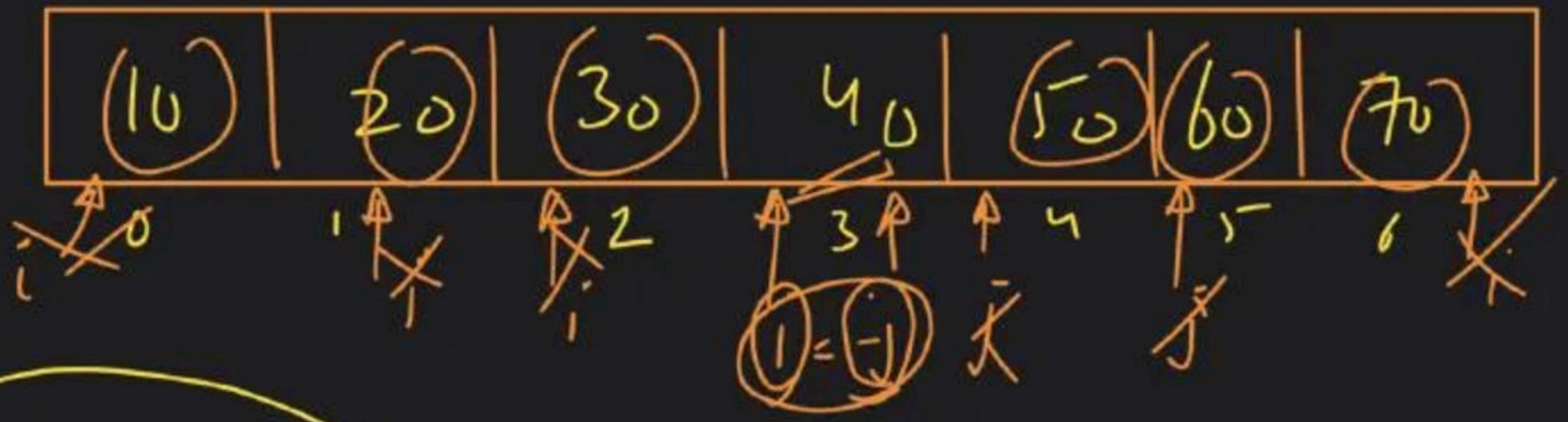
zeroCount = 0+1+1+1 → 3

oneCount = 0+1+1+1+1 → 4

totalZero → 3

totalOne → 4


i/p
10
20
30
40
50
60
arr
0
1
2
3
4
5
i
j
i = 0
j = n-1
o/p
10 60 20 50 30 40
extreme printing
cout << i
i++
cout << j
j--
10
60
20
50
30
40
j < i

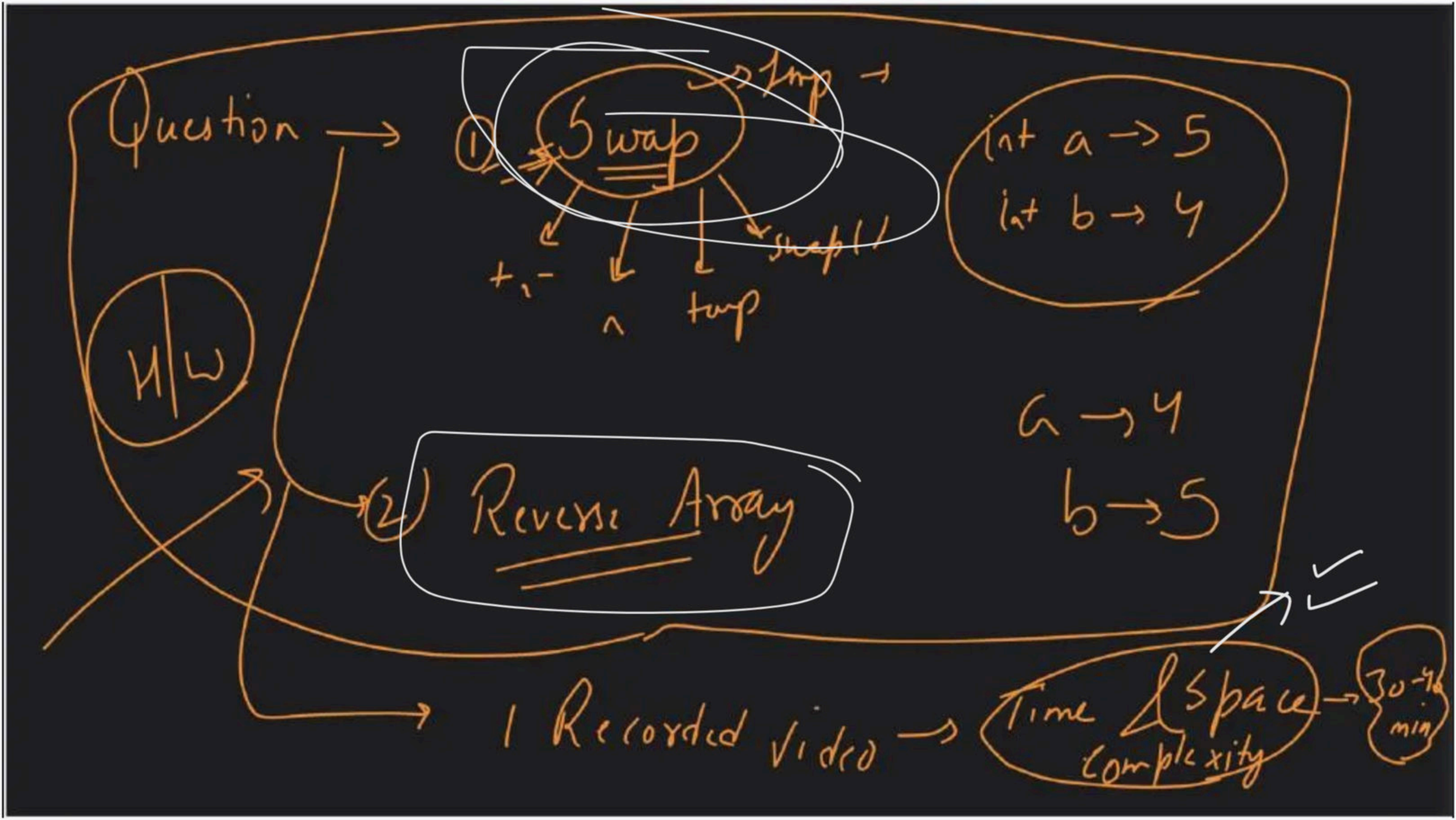


while (i > j)

{

}

10
70
20
60
30
50


Question
Swap
temp
int a → 5
int b → 4
H/W
Reverse Array
a → 4
b → 5
Recorded Video
Time & Space
Complexity
30-40 min

XOR swapping

initial → a = 5 , b = 6

① a = a ^ b ; a = 3 , b = 6

② b = b ^ a ; a = 3 , b = 5

③ a = a ^ b ; a = 6 , b = 5

a = 011

b = 101

110

b = 110

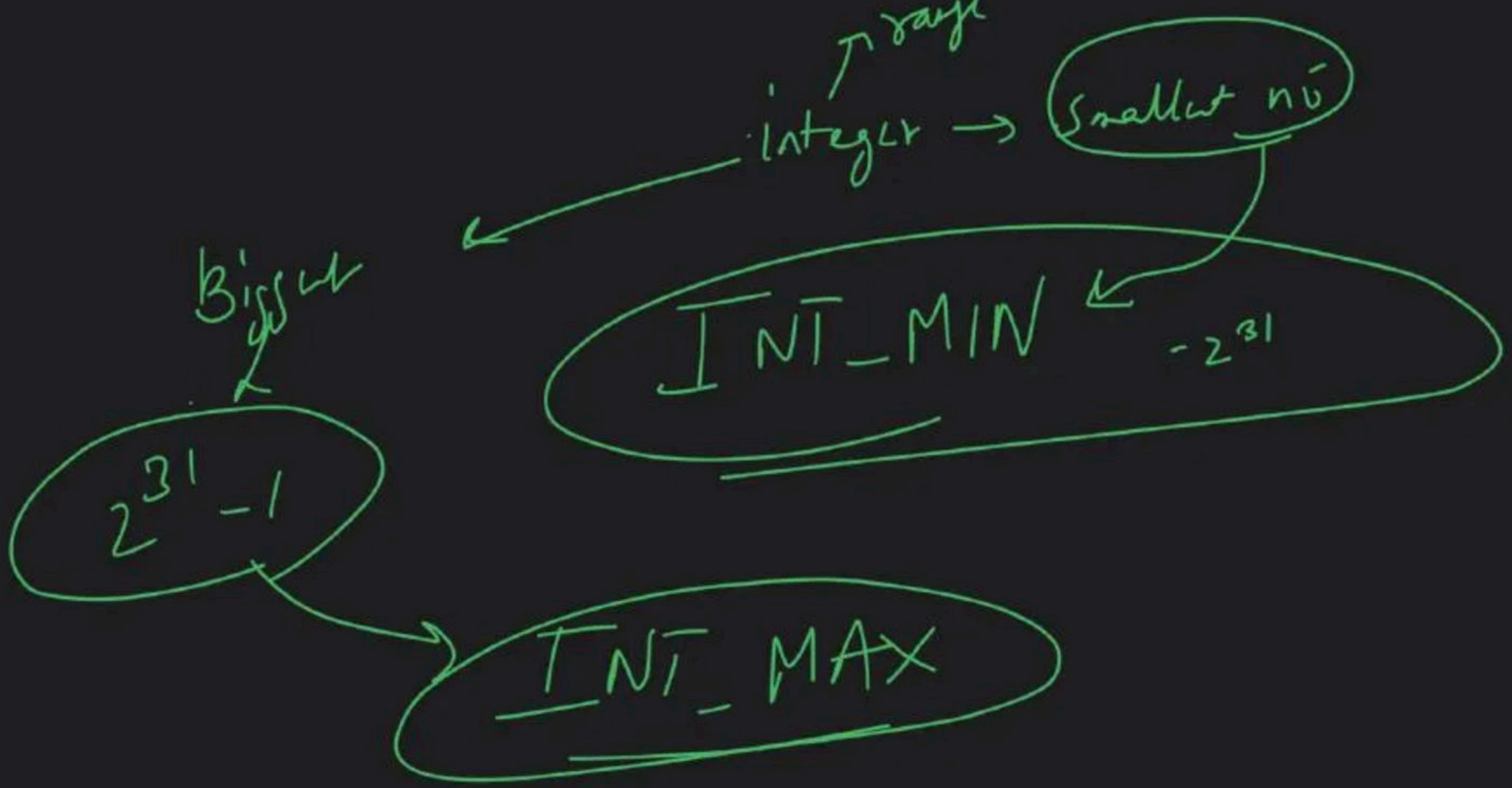
a = 011

101 → 5

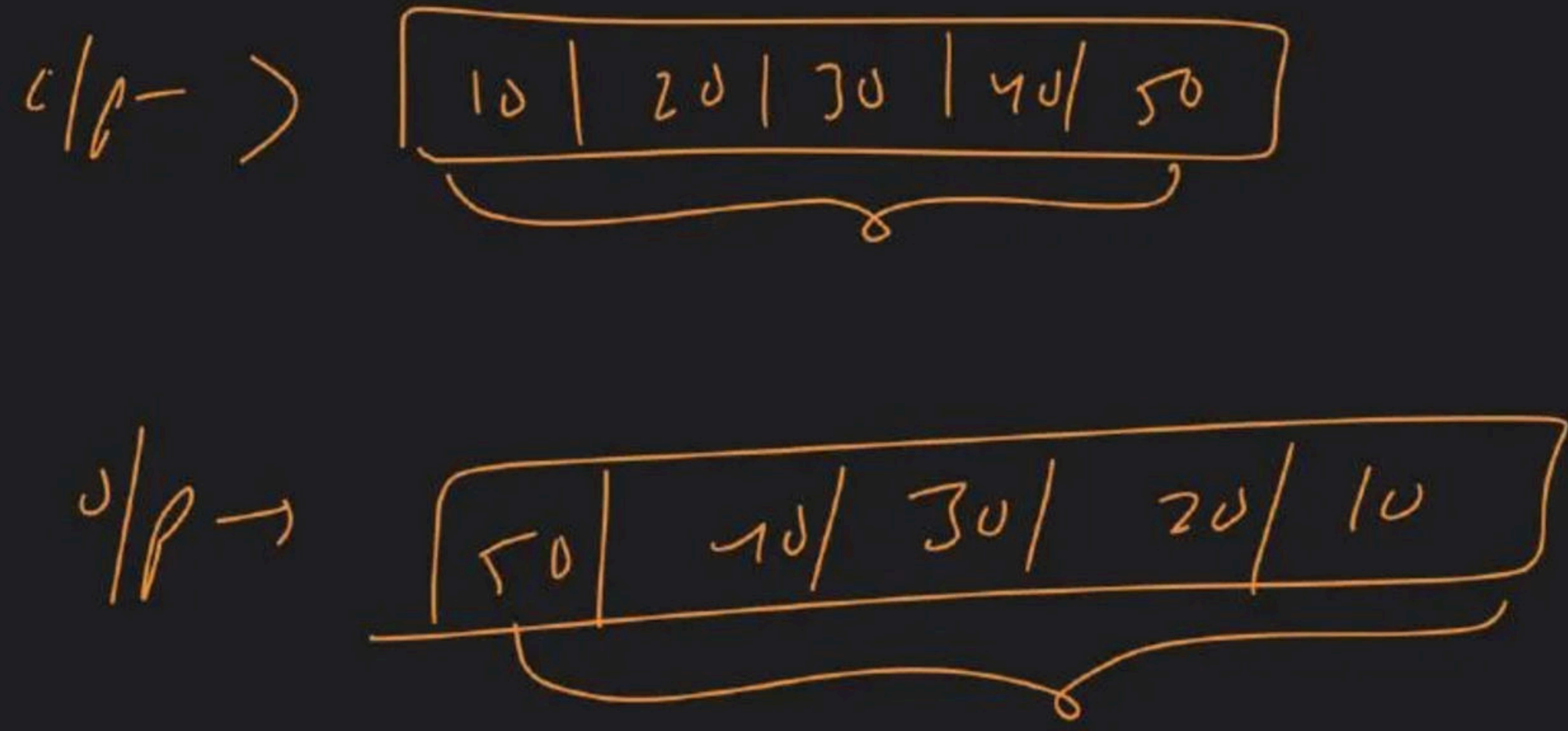
XOR swapping

1. a = 10, b = 11
2. a = 3, b = 17
3. a = 77, b = 13

Dry Run

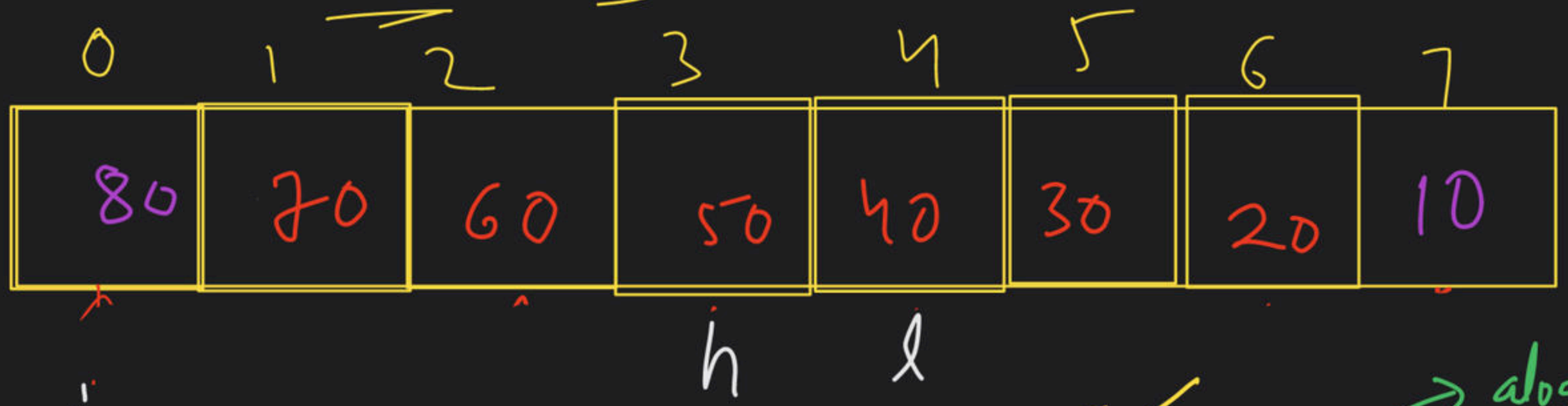

range
integer
smallest no
INT_MIN
-2^31
Biggest
2^31 - 1
INT_MAX

```
int findMax(arr, n)
{
    int maxAns = INT_MIN;
    for (i=0; i<n; i++)
    {
        if (arr[i] > maxAns)
        {
            maxAns = arr[i];
        }
    }
    return maxAns;
}
```


i/p ->
10 | 20 | 30 | 40 | 50
o/p ->
50 | 40 | 30 | 20 | 10

Reverse an array

N = 8



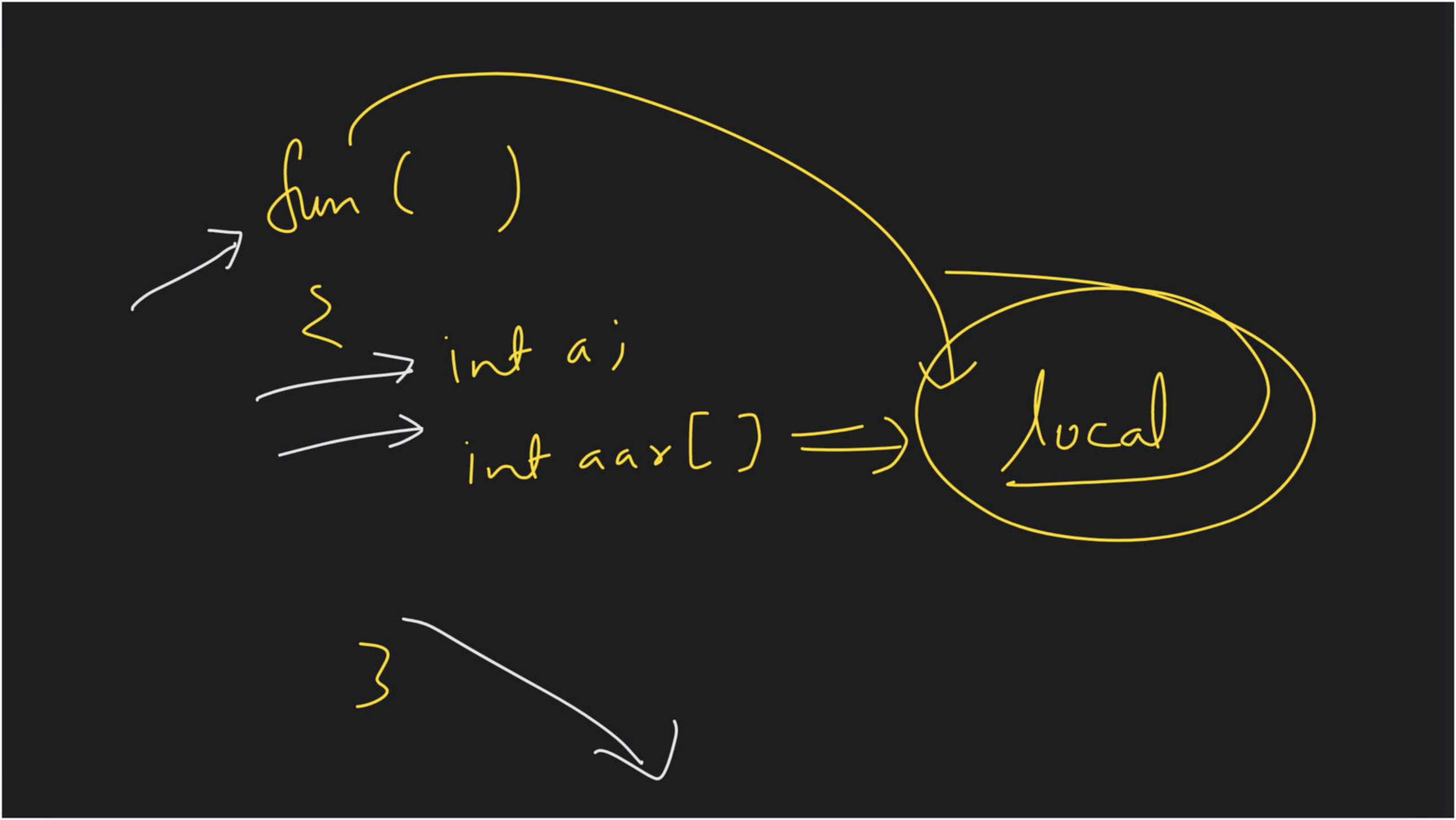
algorithm

Logic

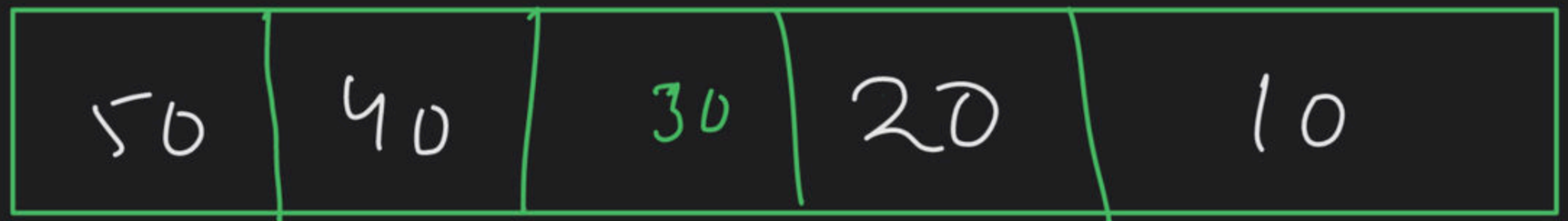
```
while( l < h )
{
    swap( arr[l], arr[h] )
    l++;
    h--;
}
```

initial value

$l = 0, h = n-1$


fun ( )
{
int a;
int aax[ ]
local
3

N = 5



l

h

→ while (l < h)

{

① swap

② l++ ;

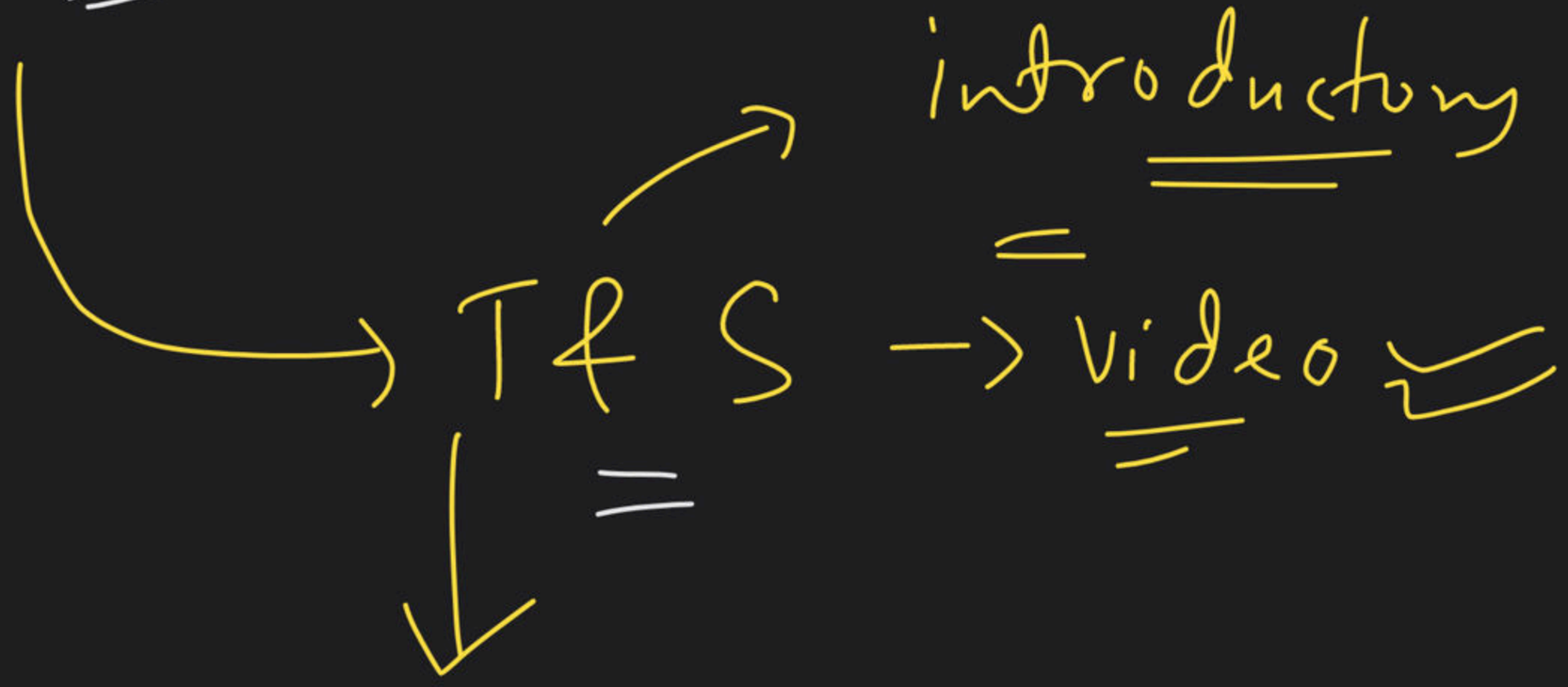
③ h-- ;

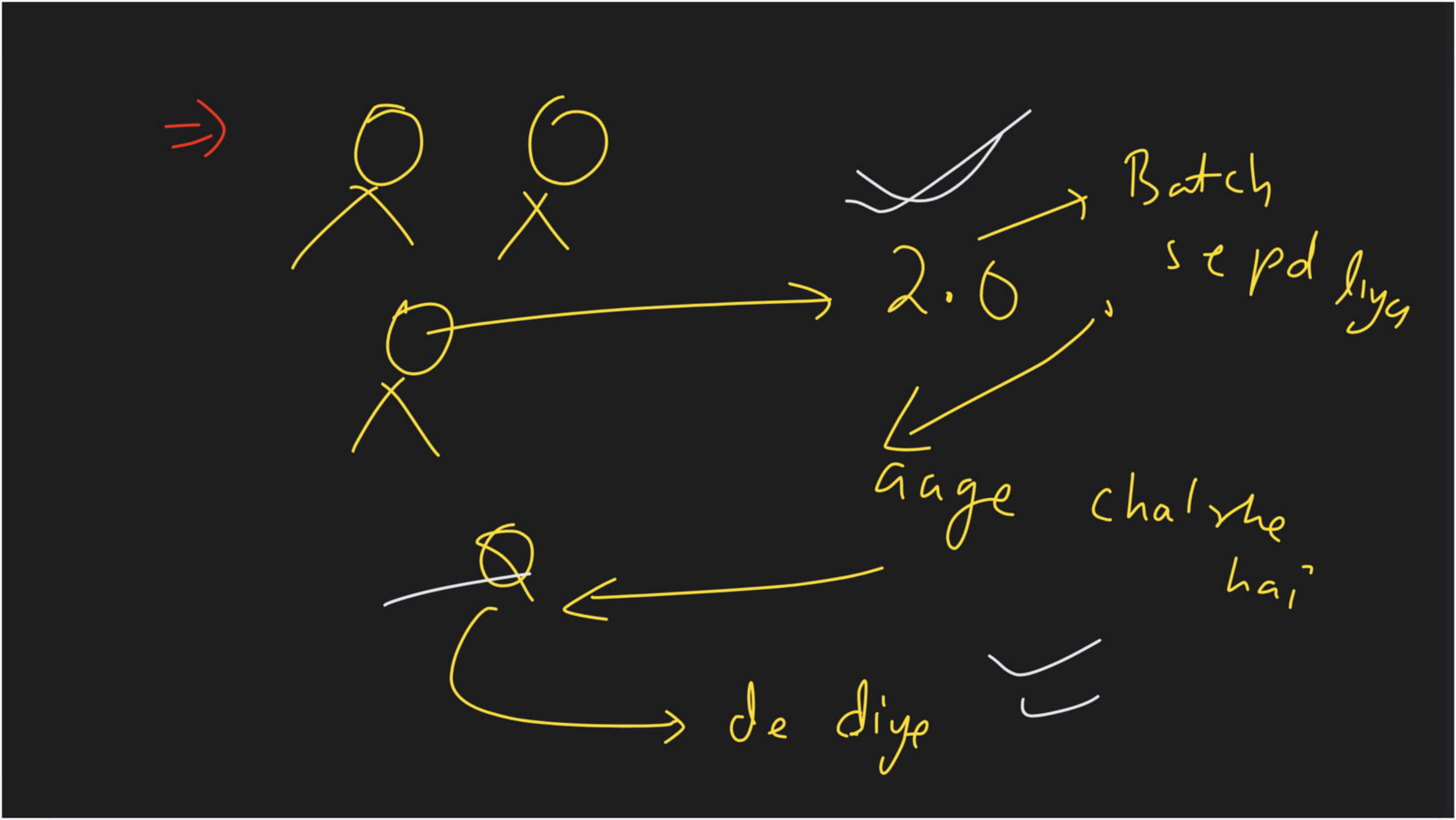
2 < 2

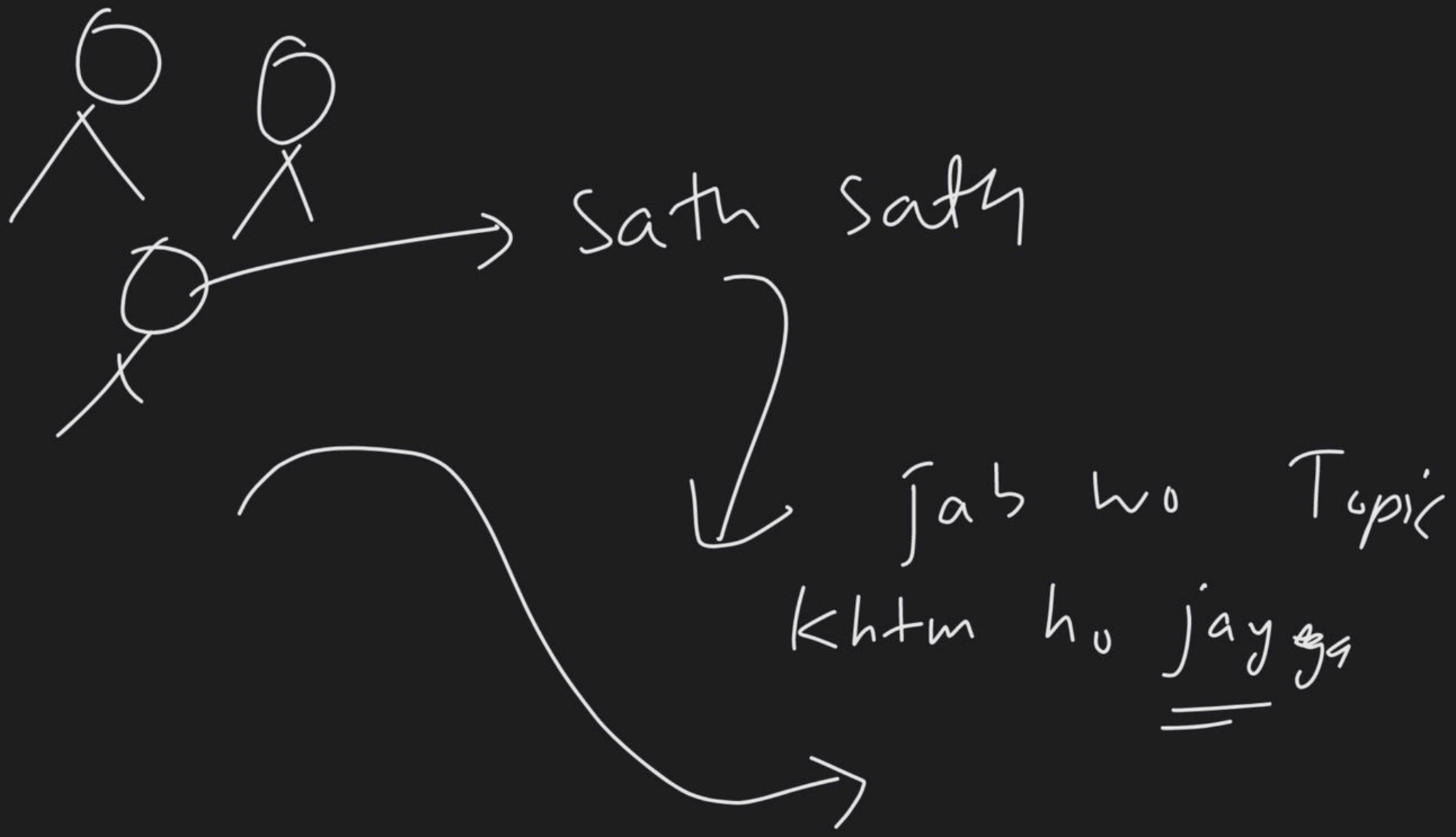
X

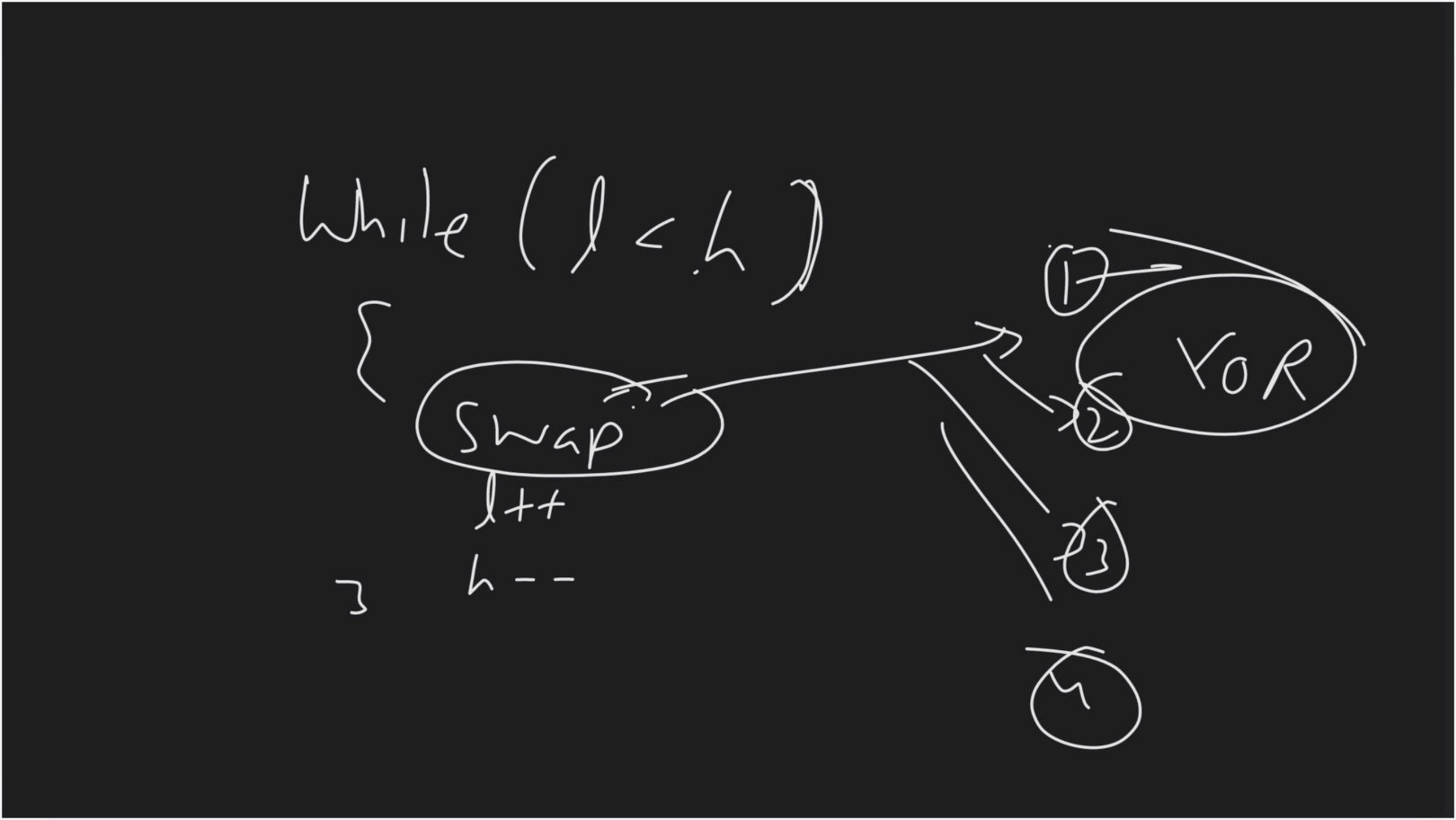
2 - pointer approach

Bonus




Batch
2.0
gaage chal rhe hai
de diye


Sath Sath
Jab ho Topic
khtm ho jayga


while (l < h)
{
swap
l++
h--
}
1
XOR
2
3
4