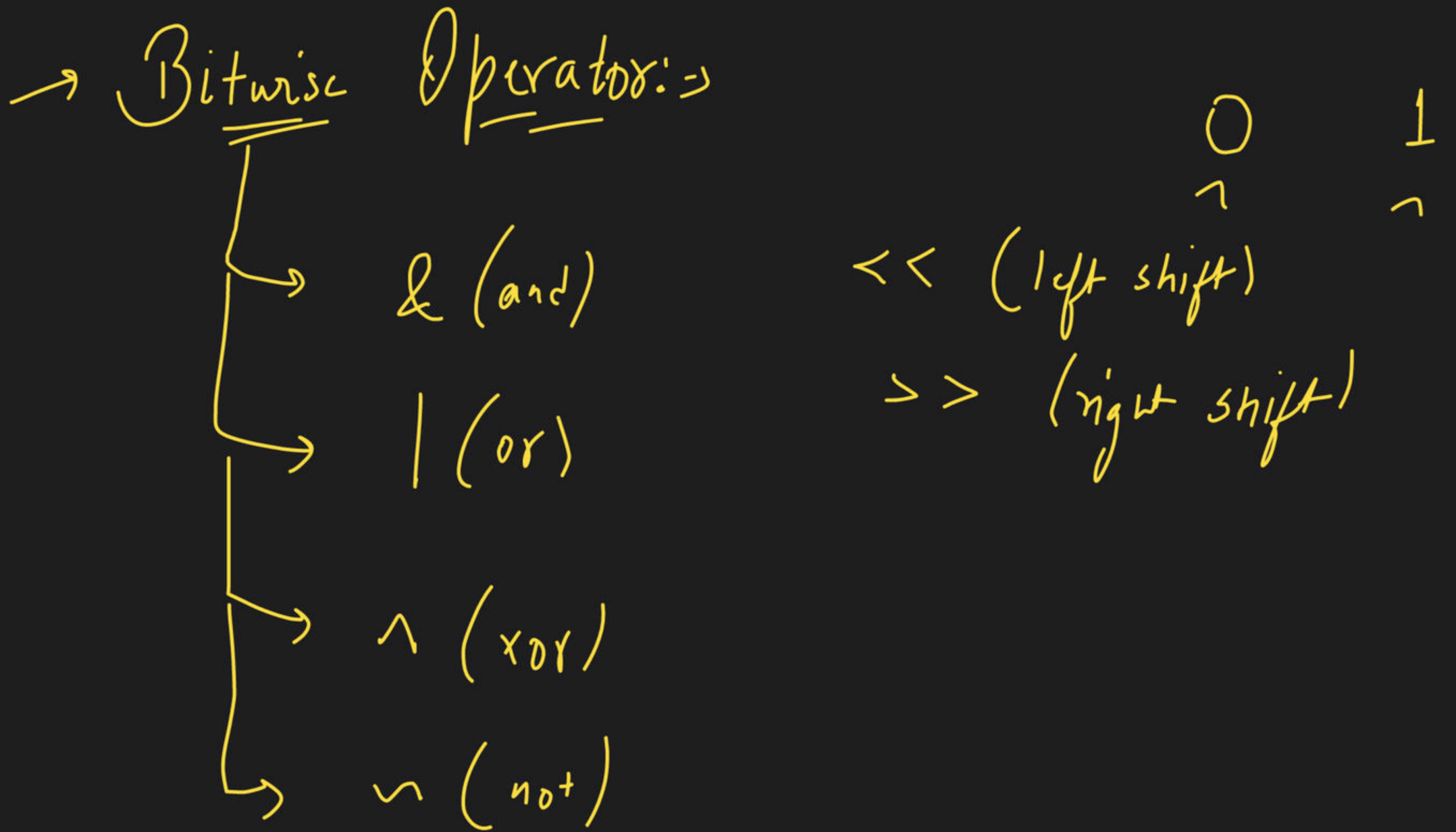
Bitwise Operators && Functions

Special class

Love Babbar • May 13, 2024


Bitwise Operator:-
& (and)
| (or)
^ (xor)
~ (not)
0
1
<< (left shift)
>> (right shift)

① &

a	b	o/p
0	0	→ 0
0	1	→ 0
1	0	→ 0
1	1	→ 1

5 & 3 → 1

0000 - - - 0001

0000 - - - 0001

1001 - - - 1001

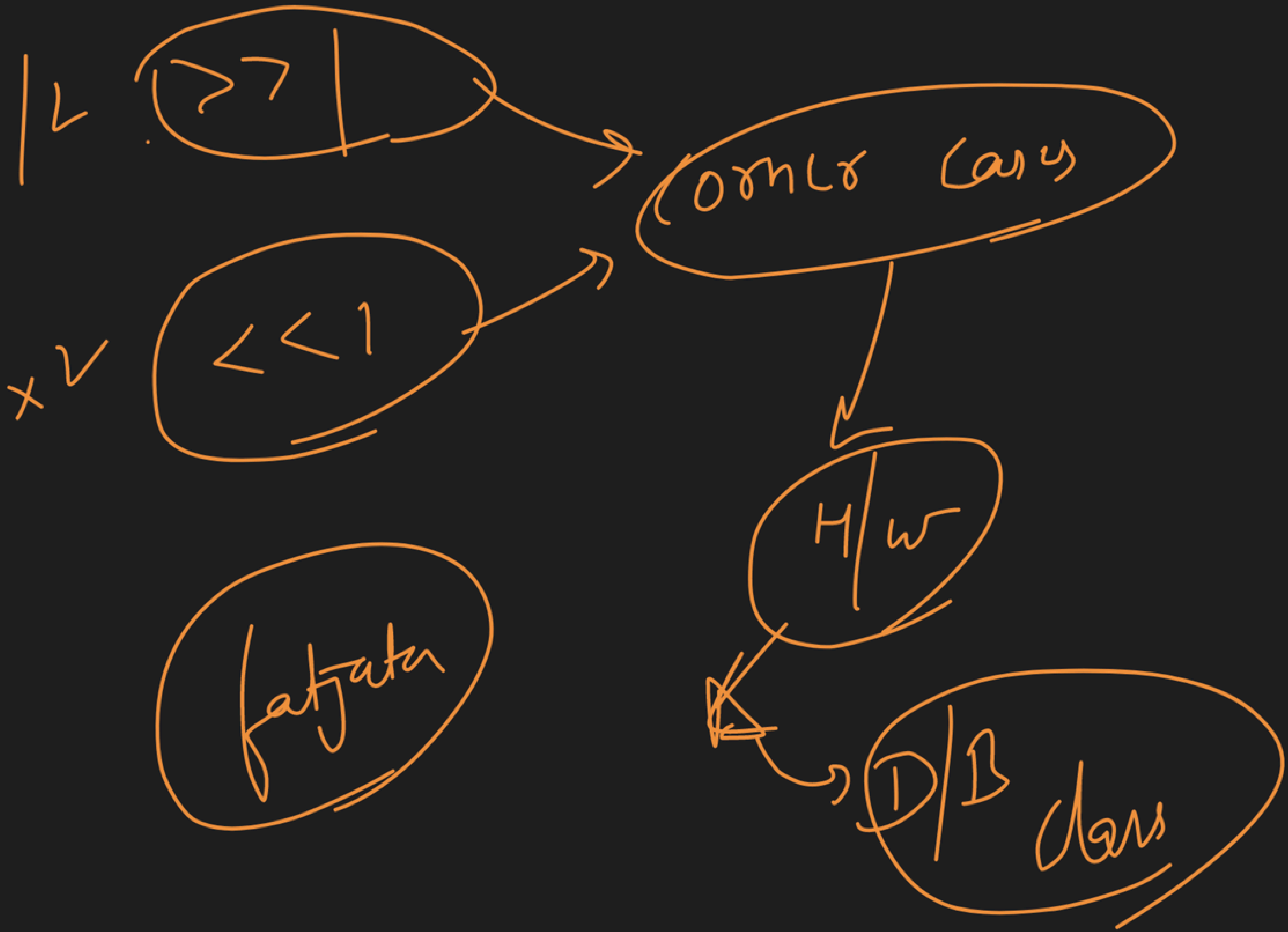
5 & 5 → 5

1

000 - - - 101

000 - - - 101

8 bits . 101 → 5

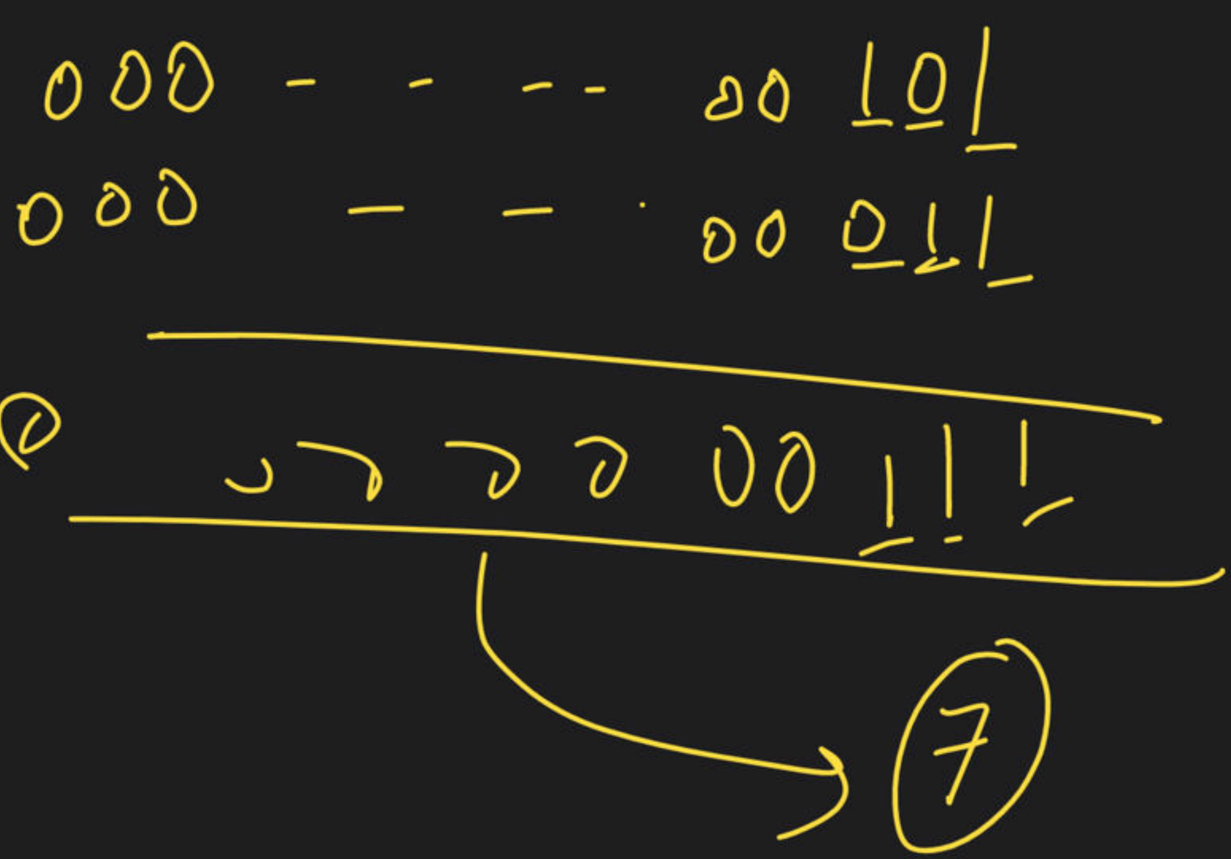


>> 1
<< 1
Corner cases
H/W
D/B
class

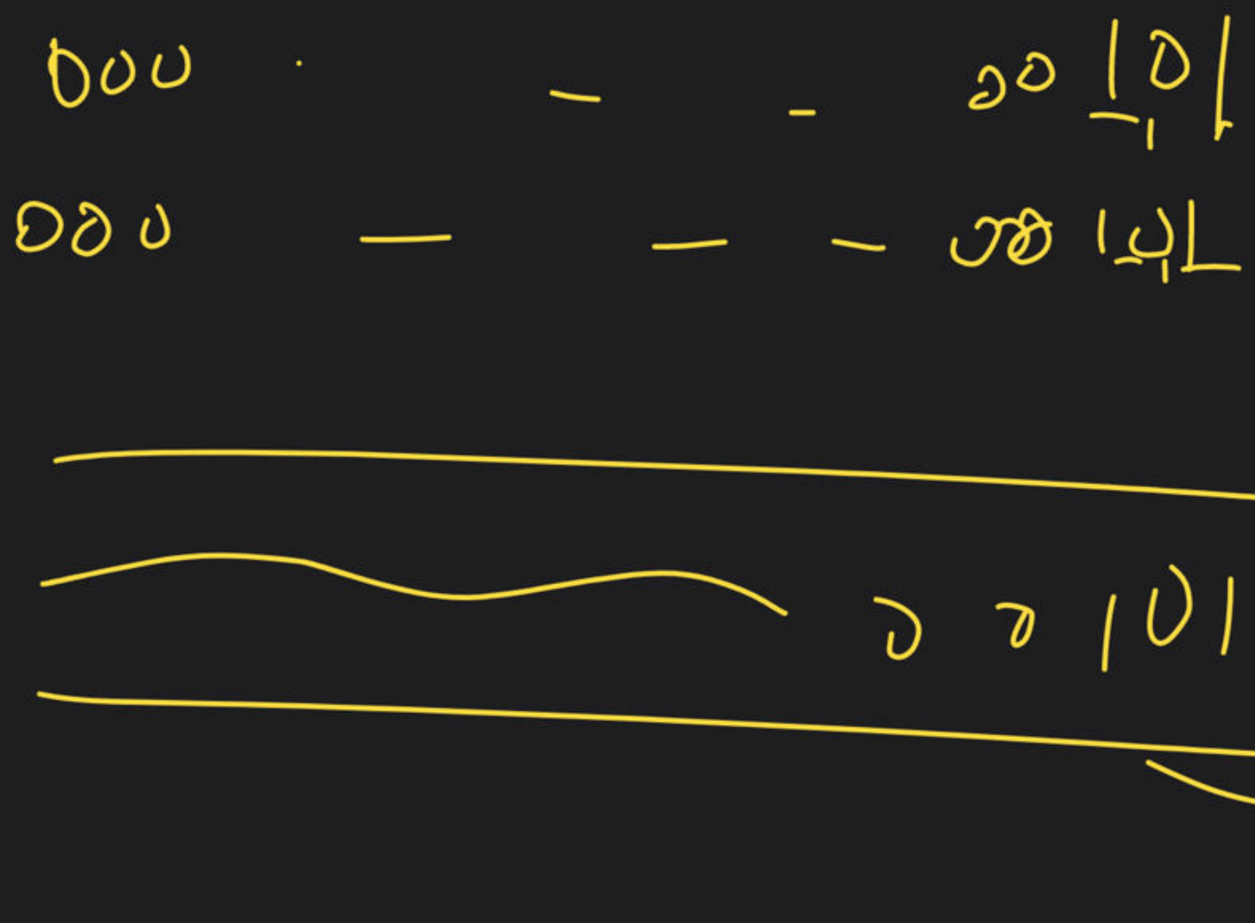
(2) | (OR)

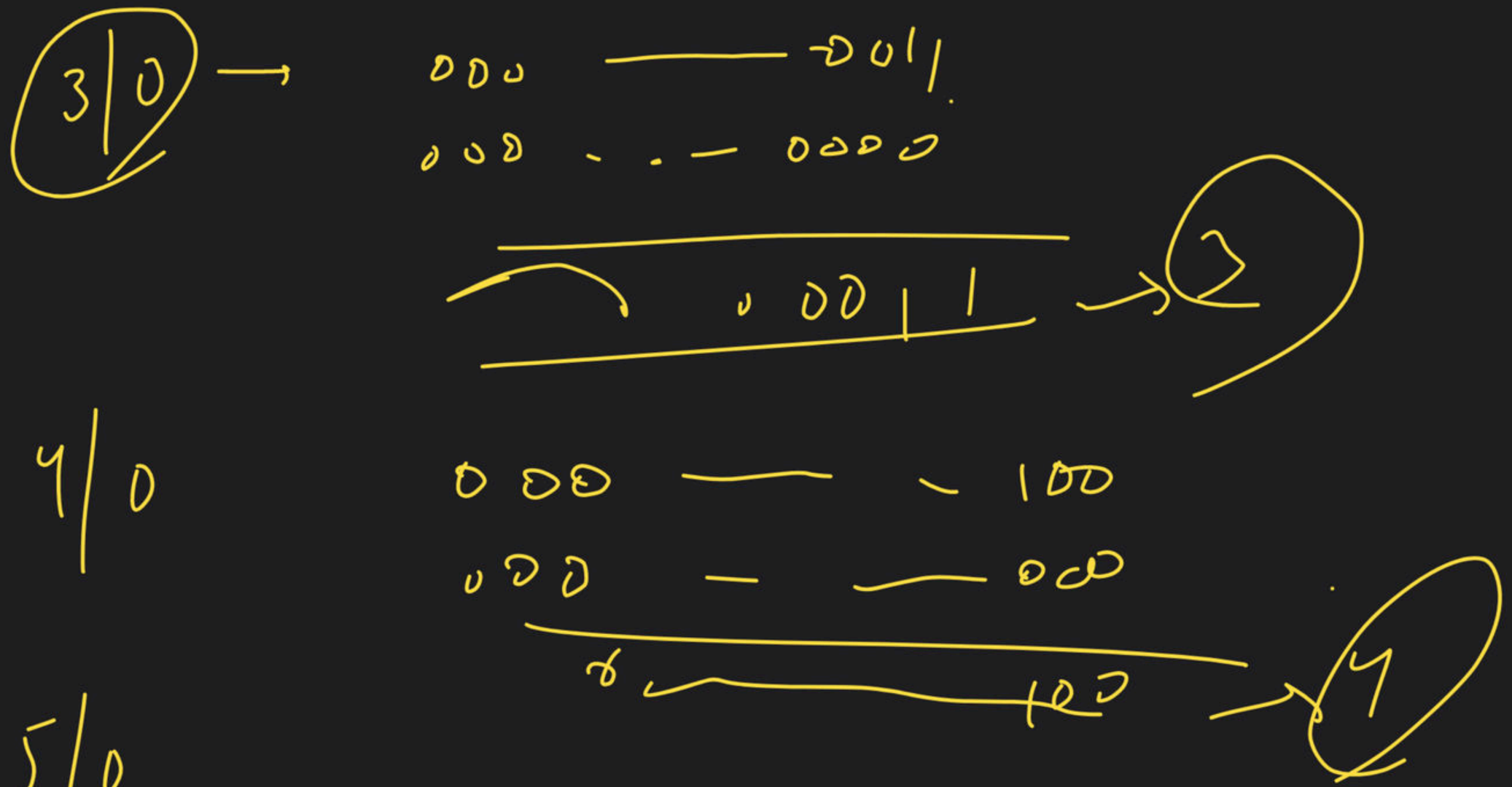
a	b	o/p
0	0	0
0	1	1
1	0	1
1	1	1

5 | 3



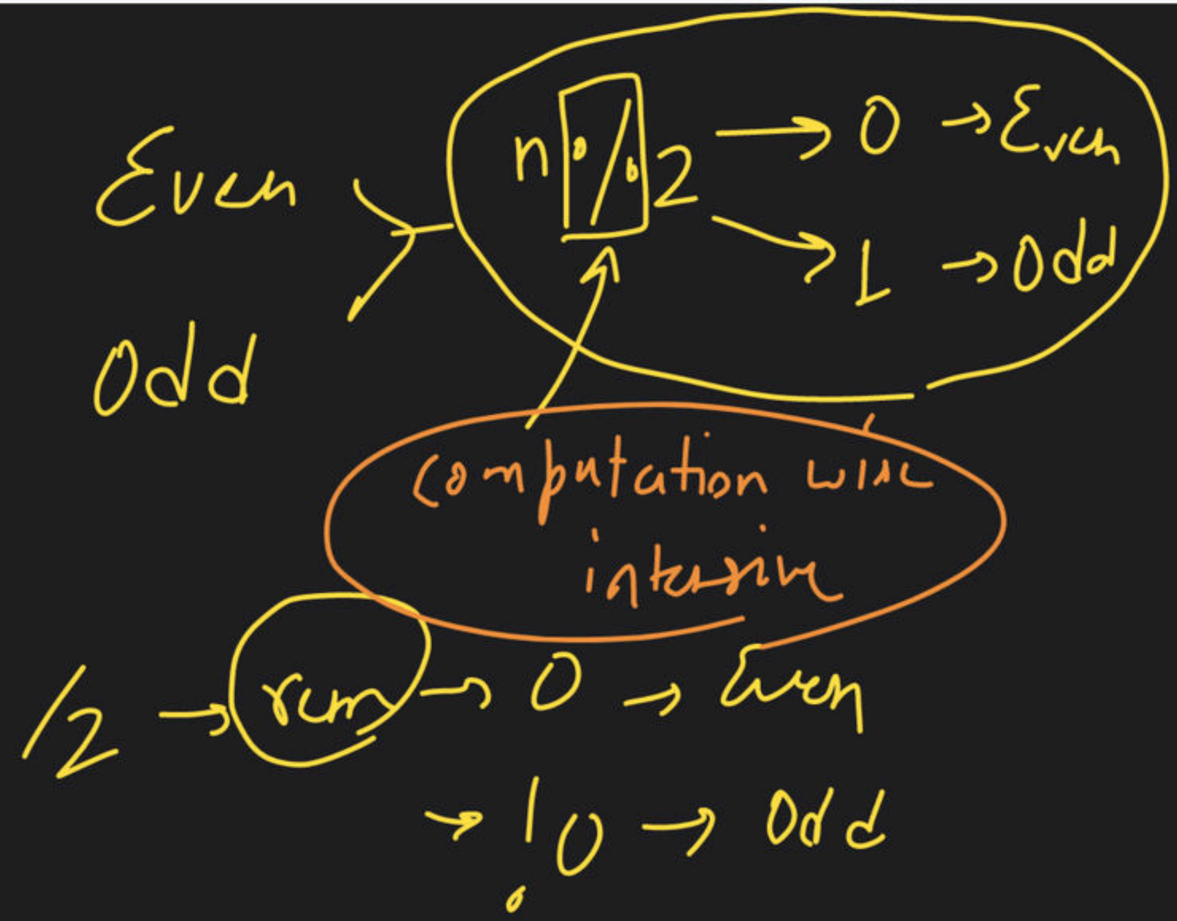
5 | 5

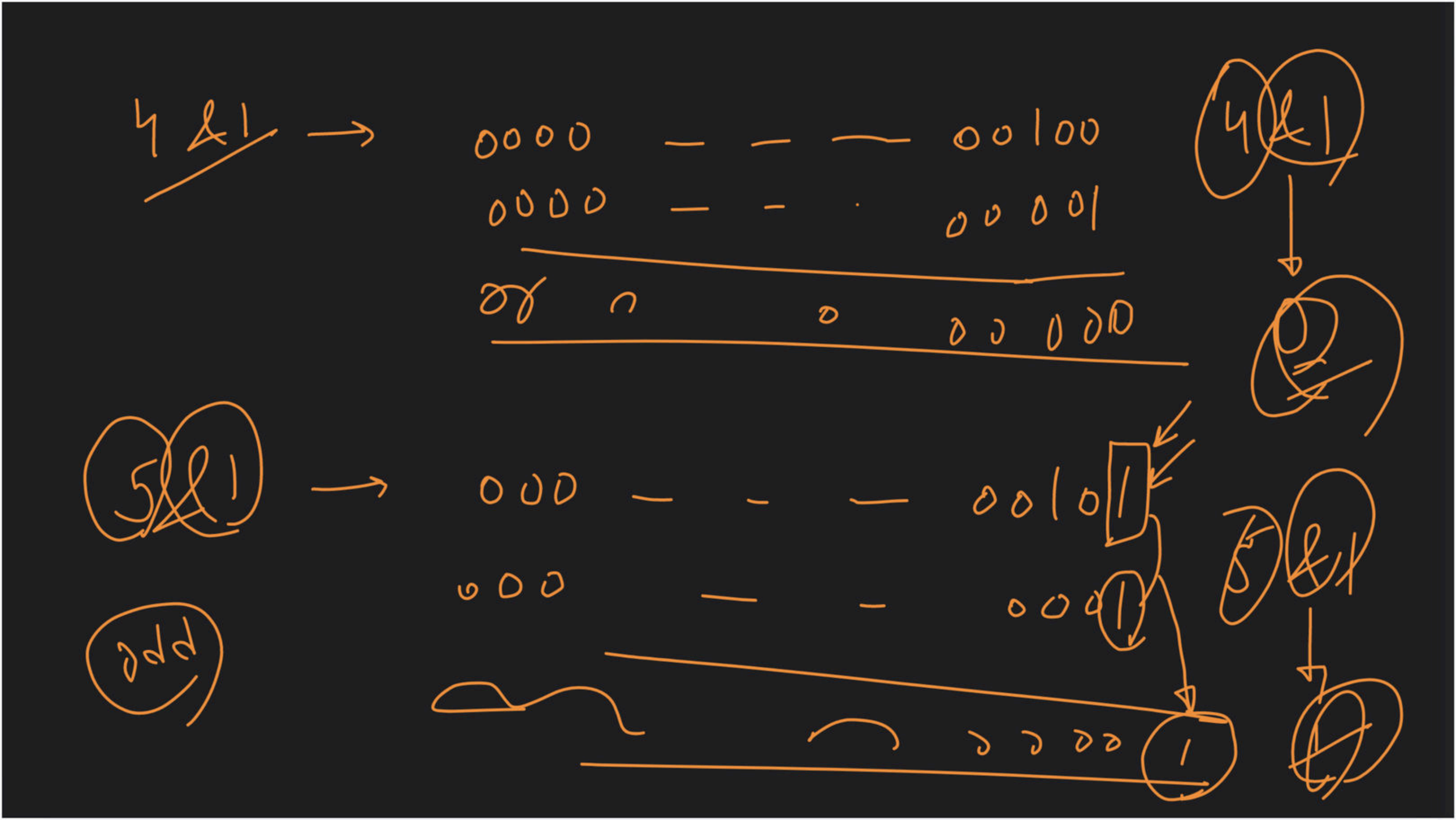


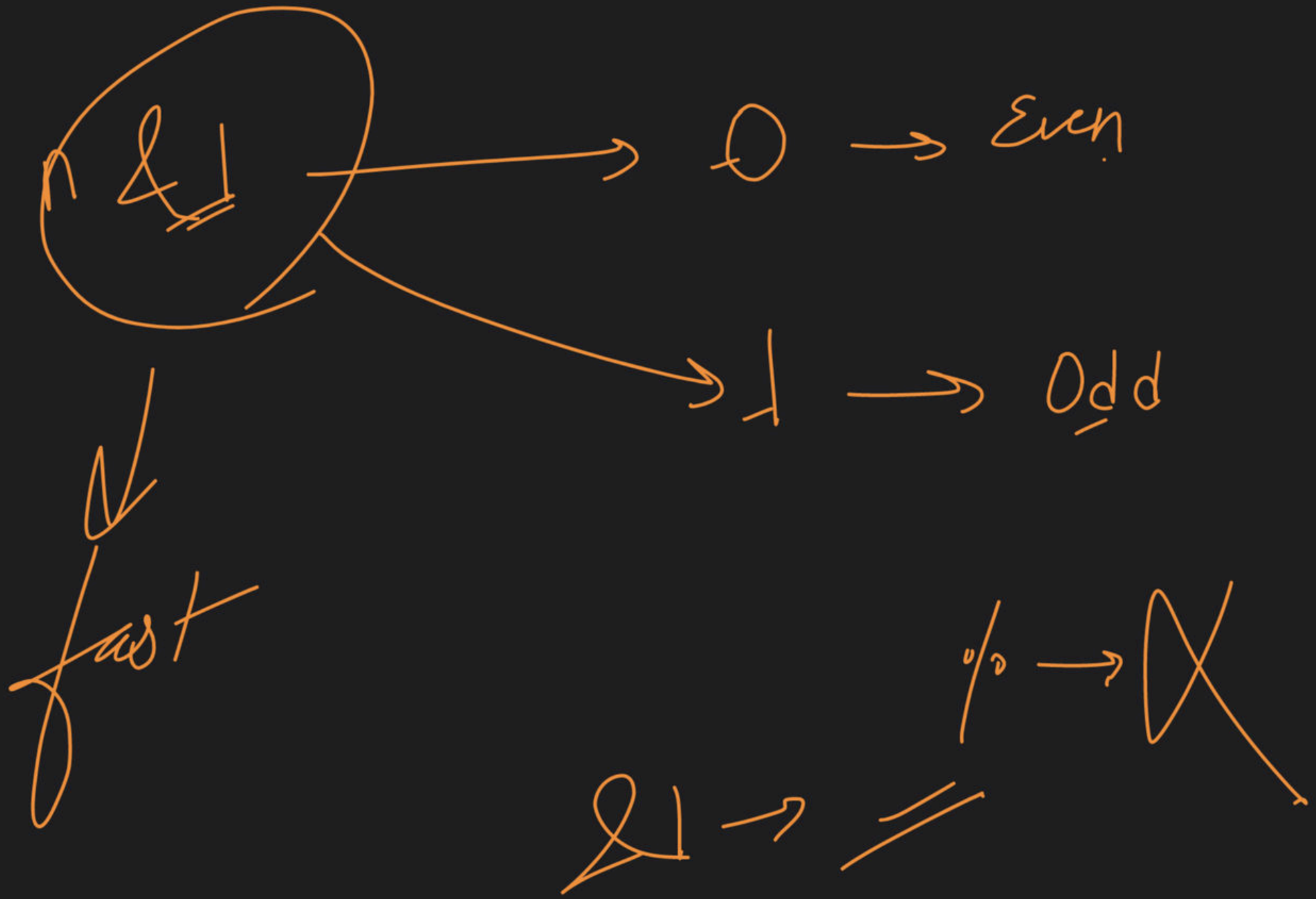

3/0
4/0
5/0

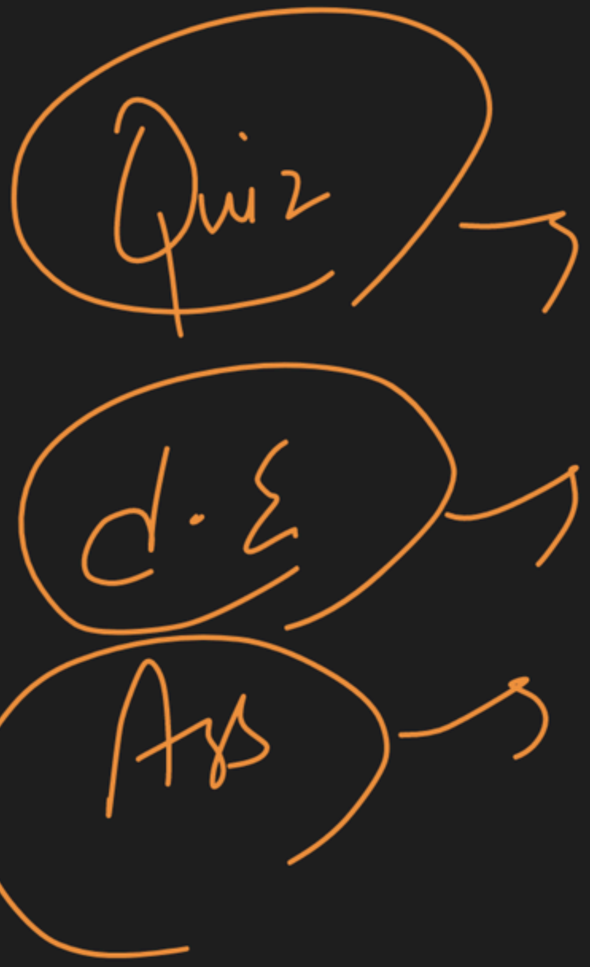
& (Bitwise AND)

n & 1 →




4 & 1
5 & 1
add

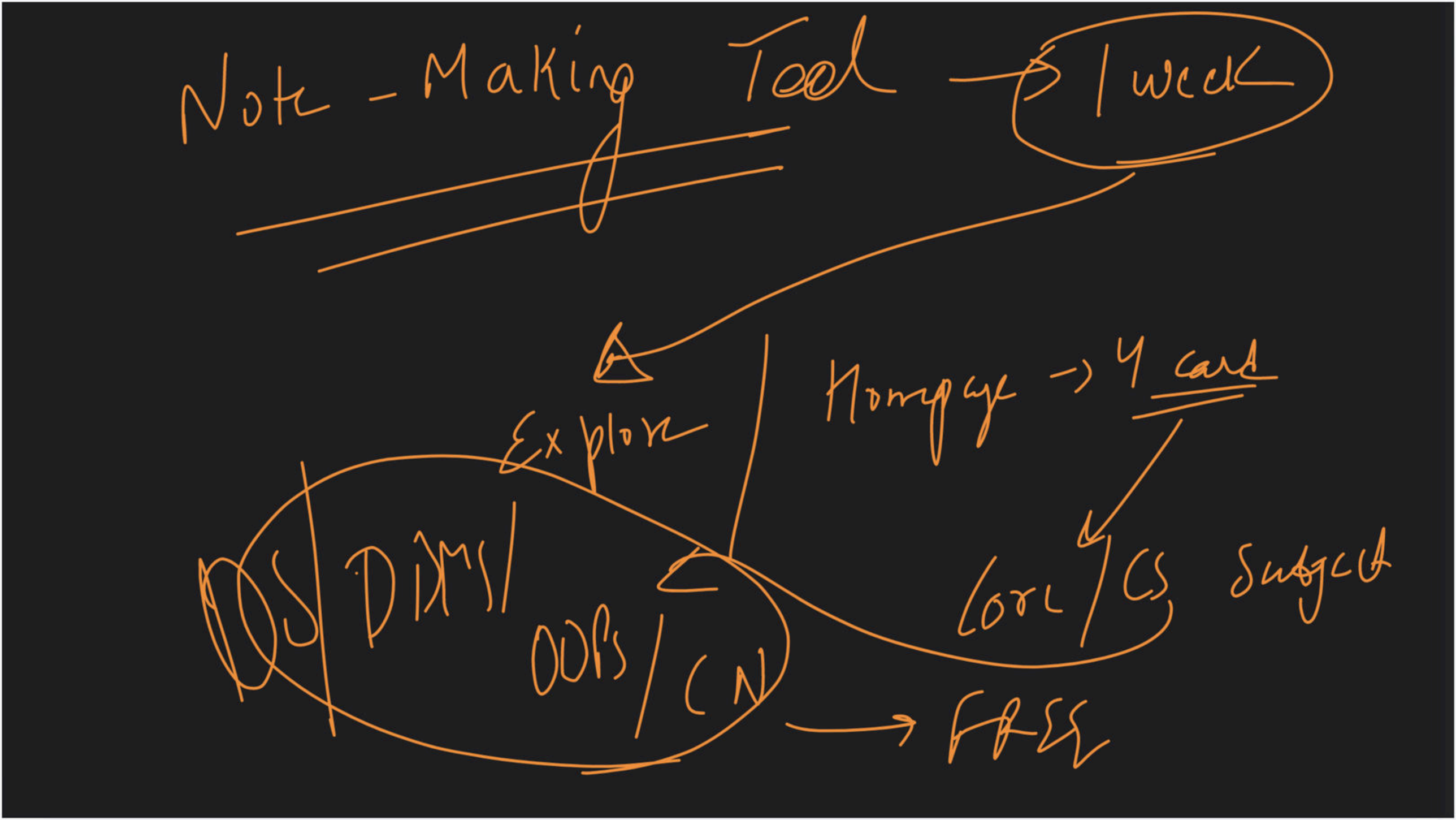

n & 1
0
Even
1
Odd
fast
%
==

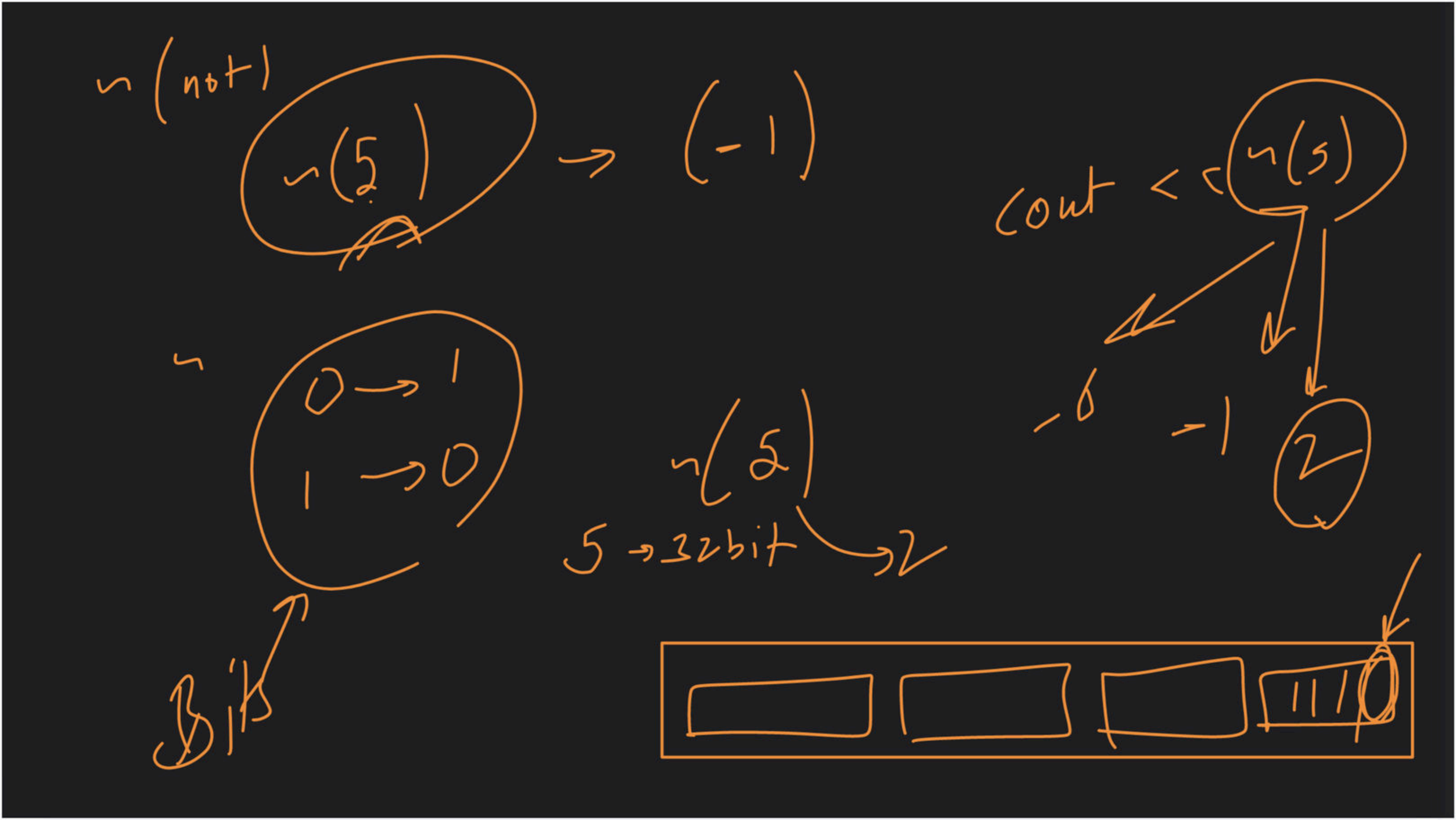

Quiz
d.E
Ass

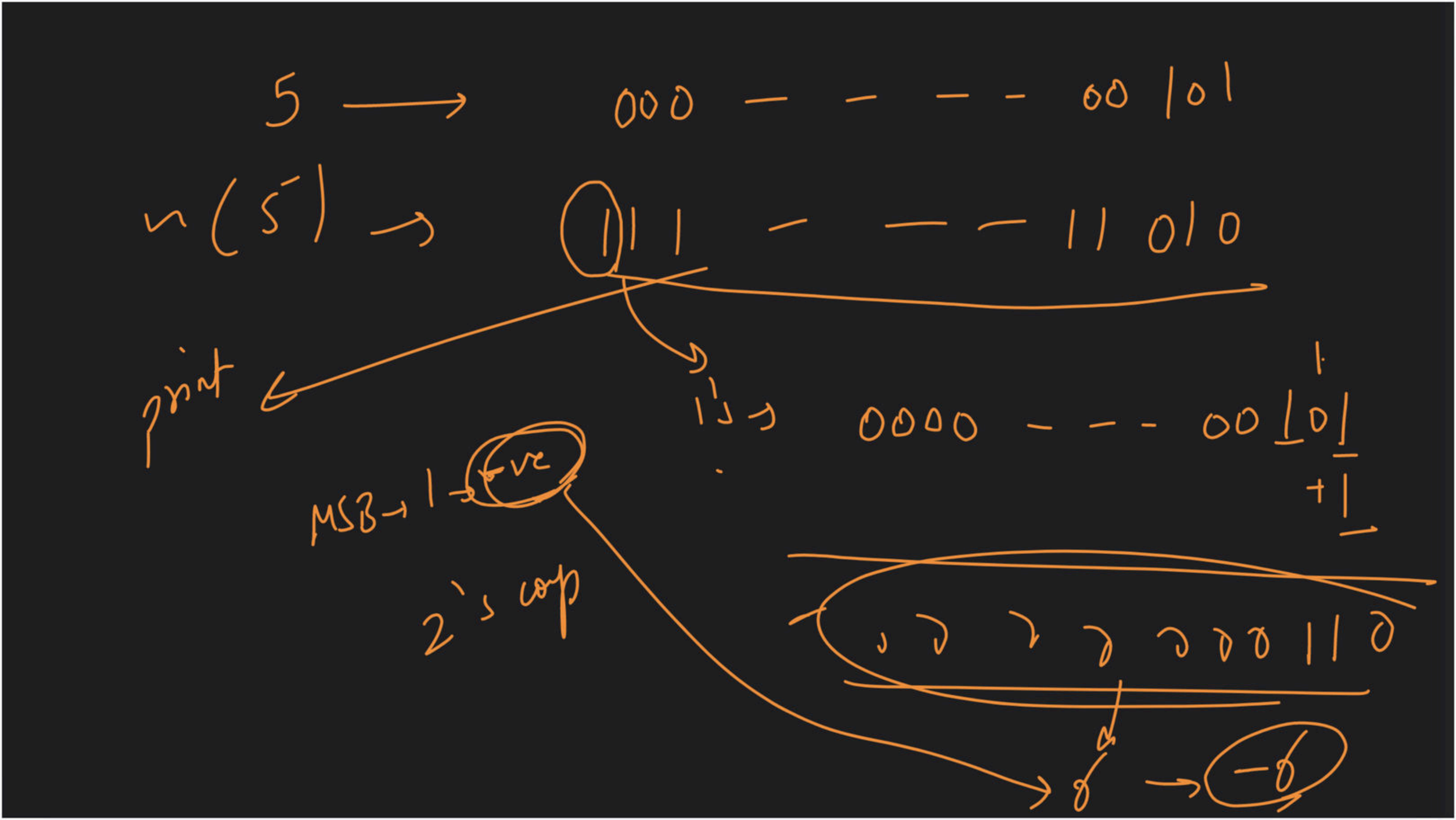
1 Even or Odd

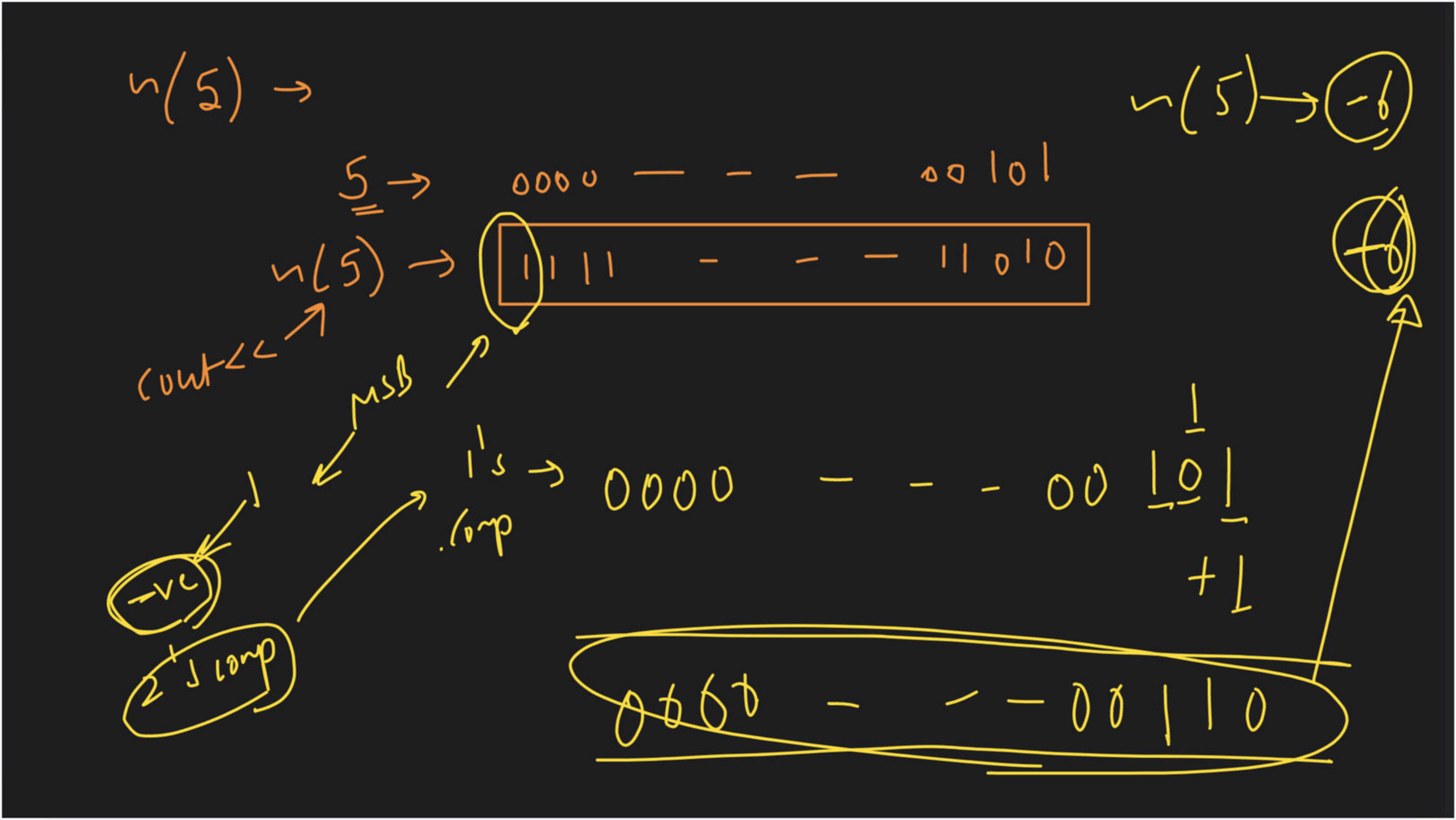
$\&1 \rightarrow 0 \rightarrow$ Even

$\&1 \rightarrow !0 \rightarrow$ odd


Note - Making Tool
$5 / week
Explore
Homepage → 4 cards
DS
DBMS
OOPs
CN
Core CS
Subject
FREE


~(not)
~(5)
-1
~
0 → 1
1 → 0
Bit
~(5)
5 → 32bit
2
cout << ~(5)
-6
-1
2


5 → 000 - - - - 00101
n(5) → 1111 - - - - 11010
print
1's
0000 - - - 00101
+1
MSB → 1 → -ve
2's comp
-5


n(5) →
5 → 0000 — — — 00101
n(5) → 11111 — — — 11010
count<<
msb
1's comp → 0000 — — — 00101
+1
-ve
2's comp
0000 — — — 00110
n(5) → -6
0

^ (xor)

same → 0

diff → 1

find unique number

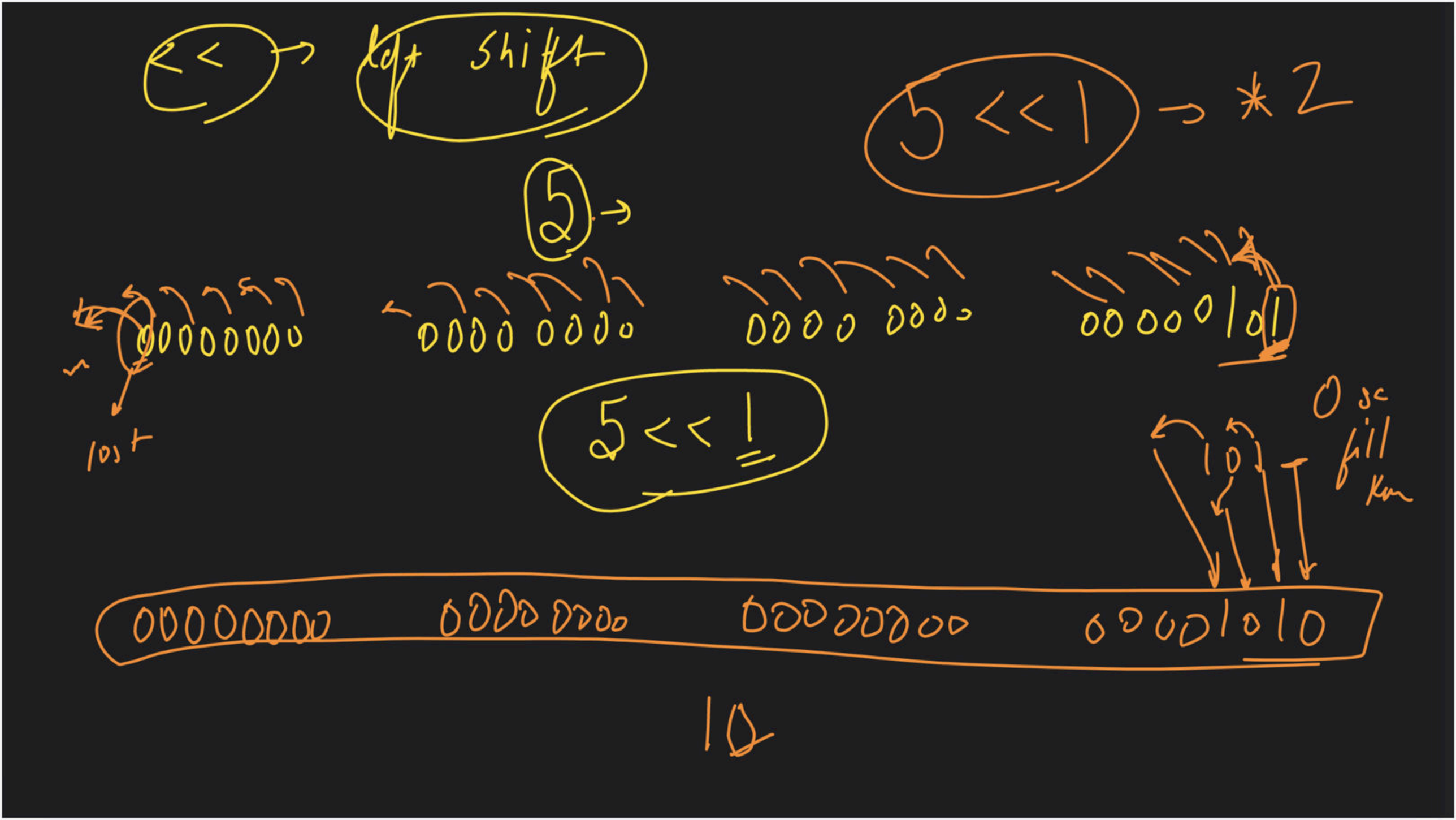
A

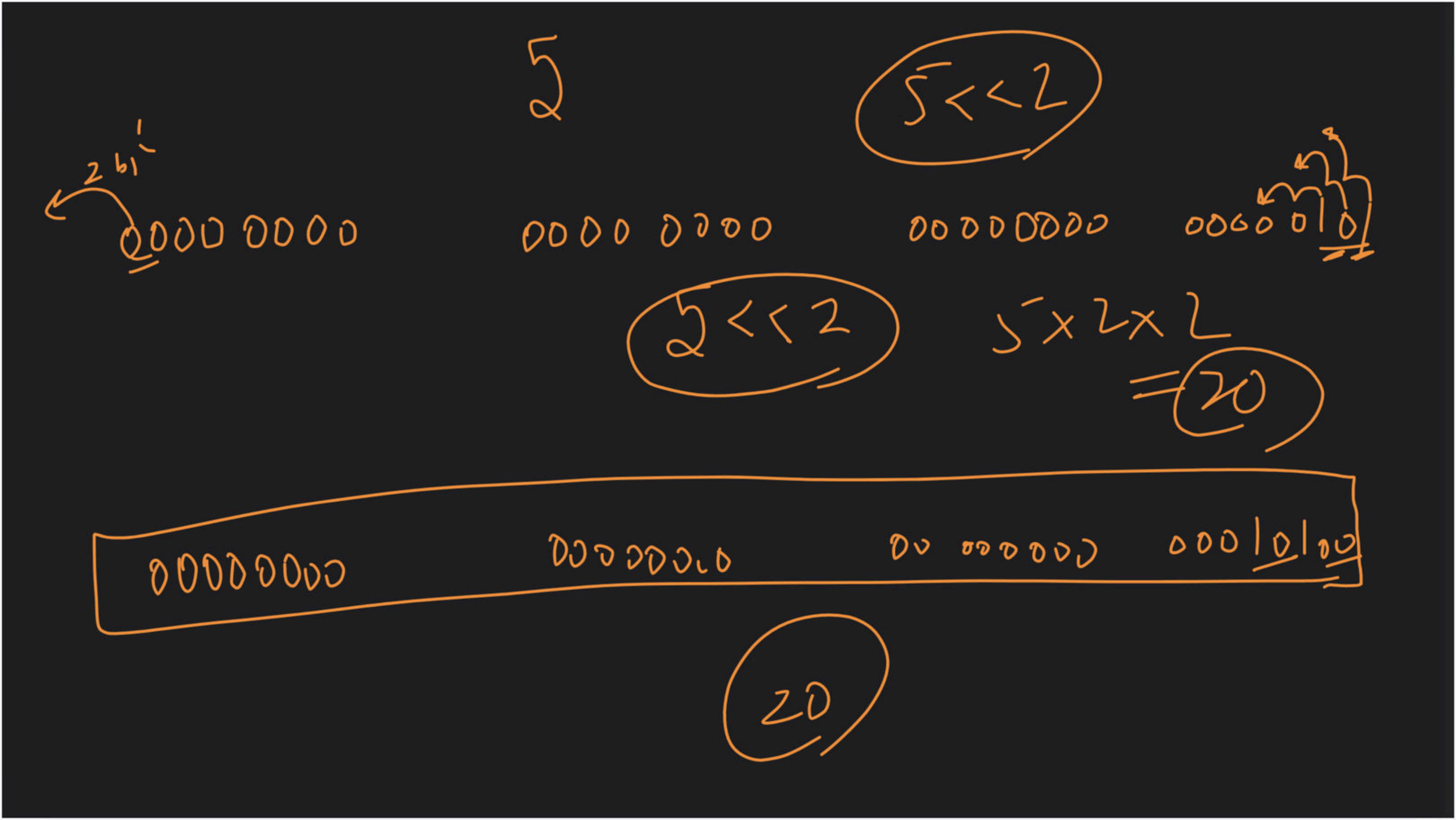
a	b	a/b
0	0	0
0	1	1
1	0	1
1	1	0

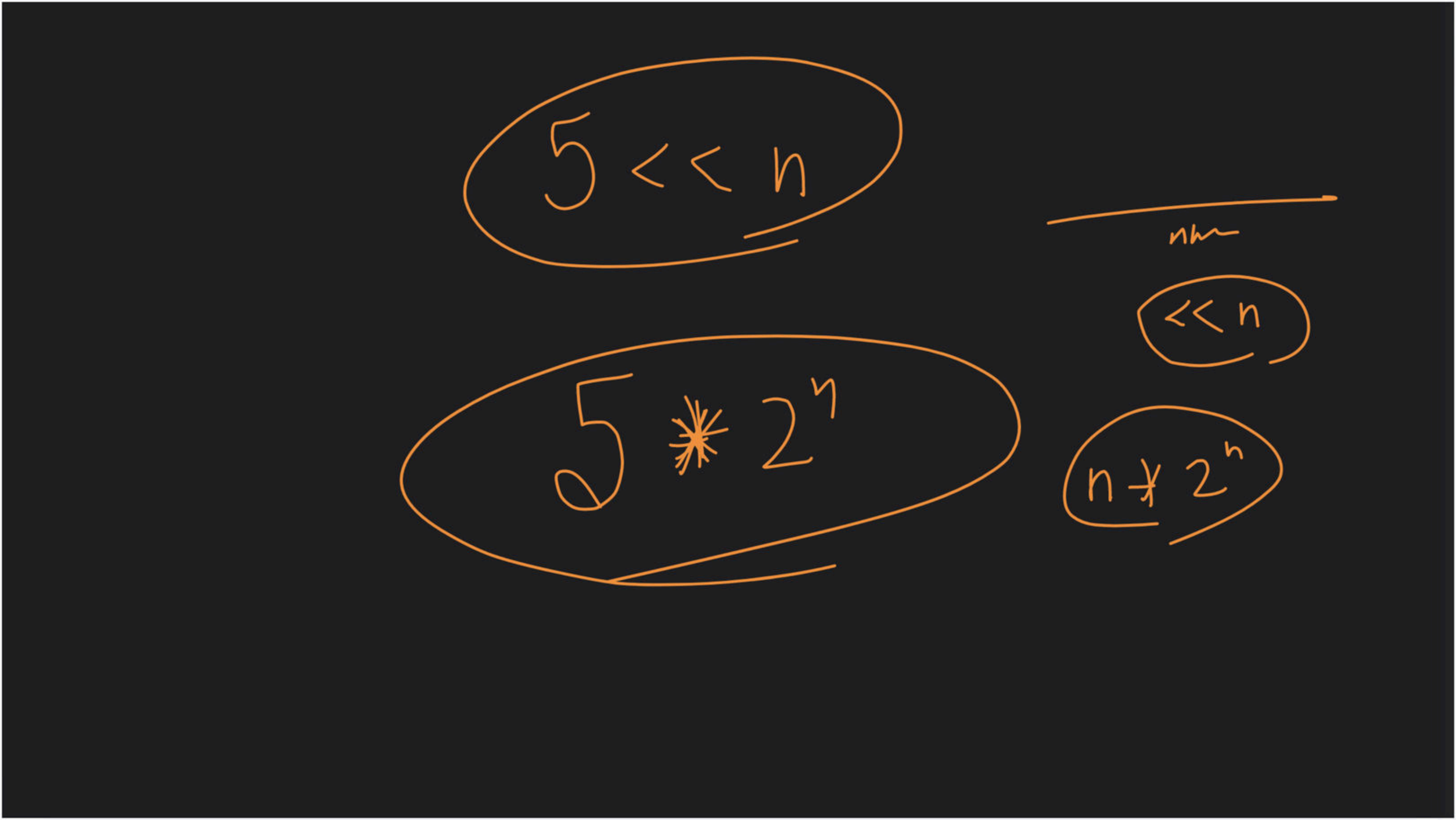
3, 8, 7, 4, 4, 12, 3, 12, 8.

3 ^ 8 ^ 7 ^ 4 ^ 4 ^ 12 ^ 3 ^ 12 ^ 8

7 ans


<<
left shift
5 <<1 → *2
5 →
00000000
00000000
00000101
lost
5 << 1
0 is fill kar
10
00000000 00000000 00000000 00001010
10


5
5<<2
2 bit
5*2*2
=20
2<<2
20


5 << n
5 * 2^n
n
<< n
n ≠ 2^n

number = A

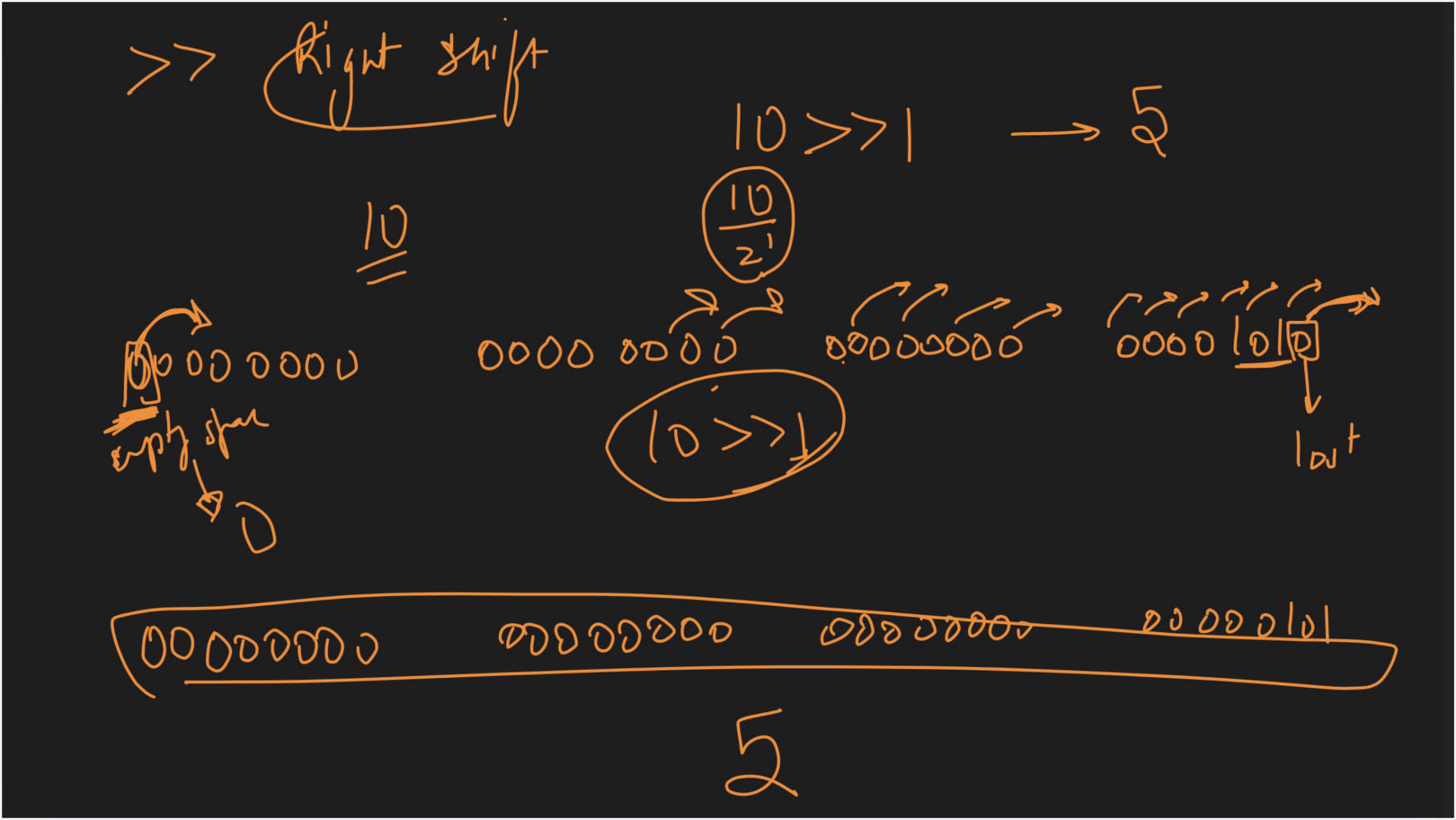
shift by n bits

$A * 2^n$

number = 5

shift 3 bits

5×2^3


>> Right shift
10 >>1 → 5
10
10/2^1
empty space
0
10 >>1
lost
2

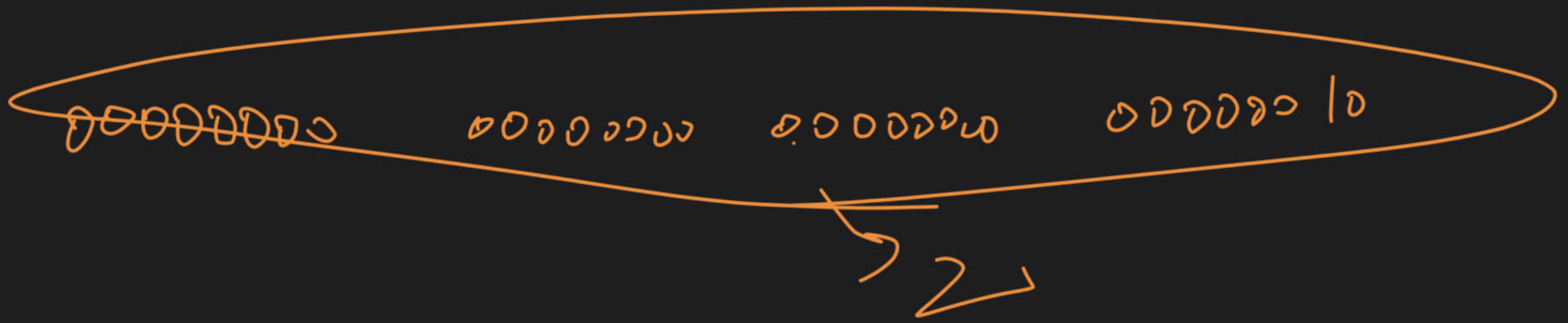
10 >> 2

10 >> 2 → 2

10 →

$\frac{10}{2^2} = \frac{10}{4} = 2$

>> 2



number → A

right shift by n bits

$\frac{A}{2^n}$

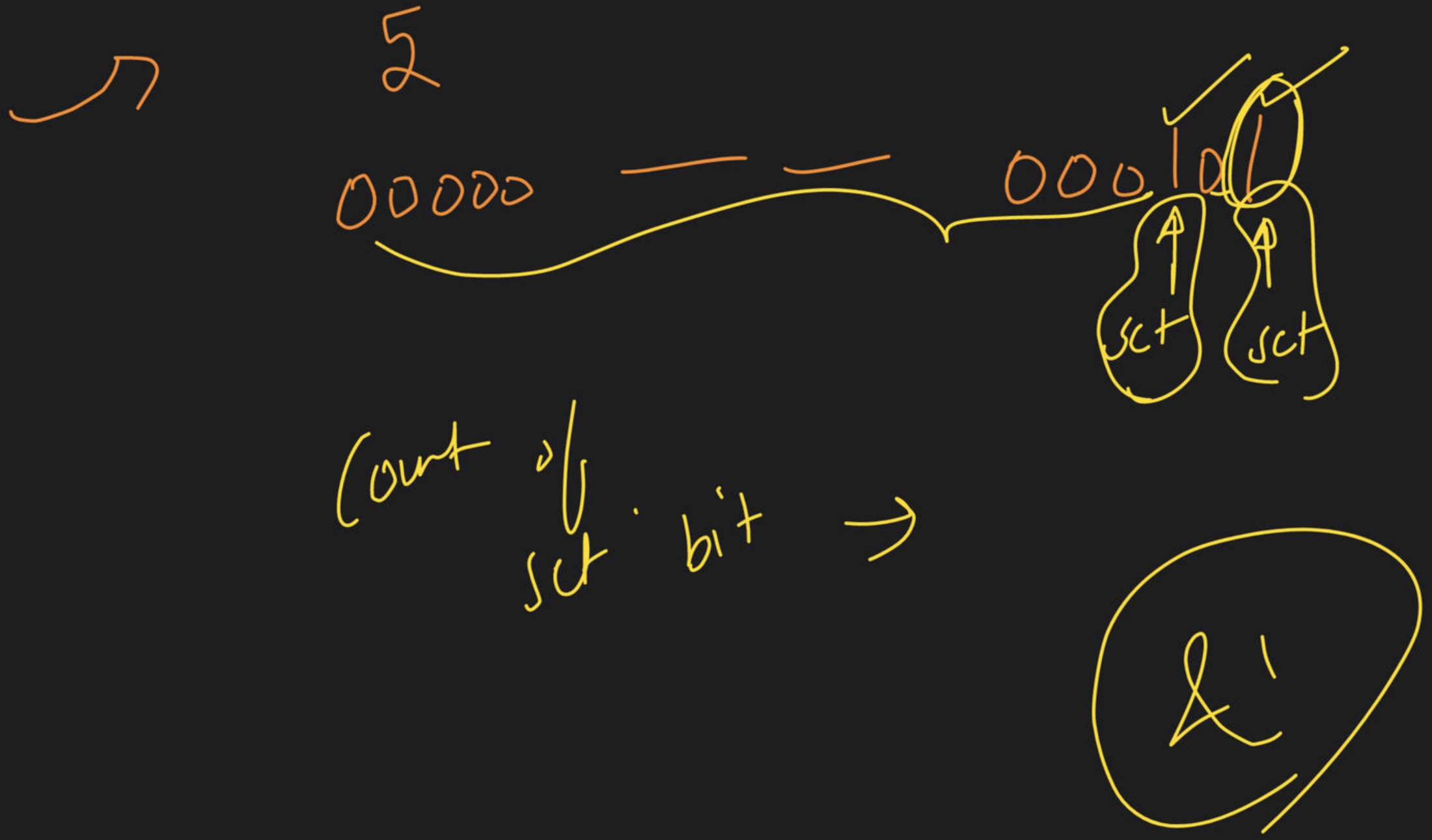
$5 >> 2$

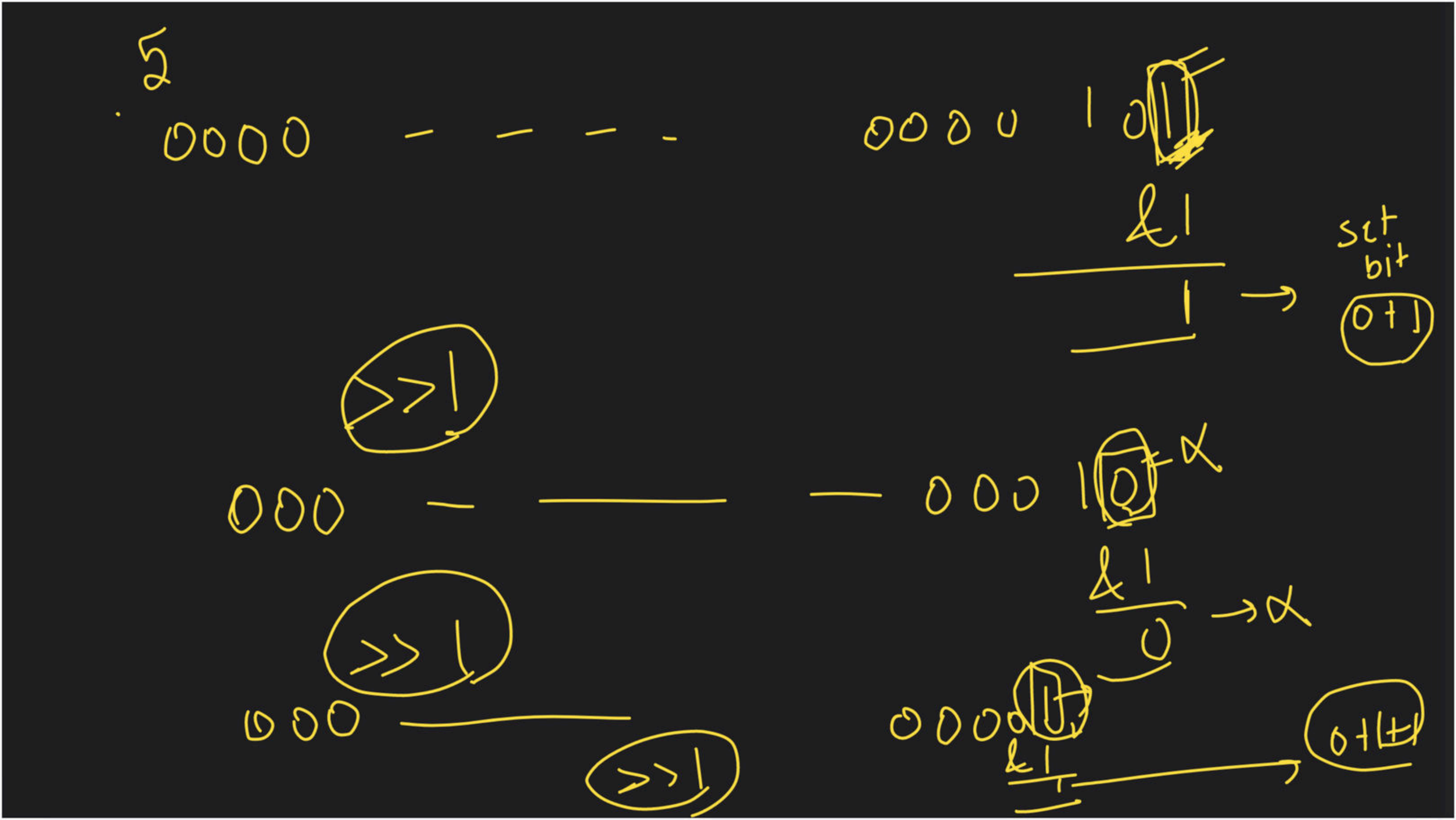
$\frac{5}{2^2} = \frac{5}{4} = 1$

number = 20

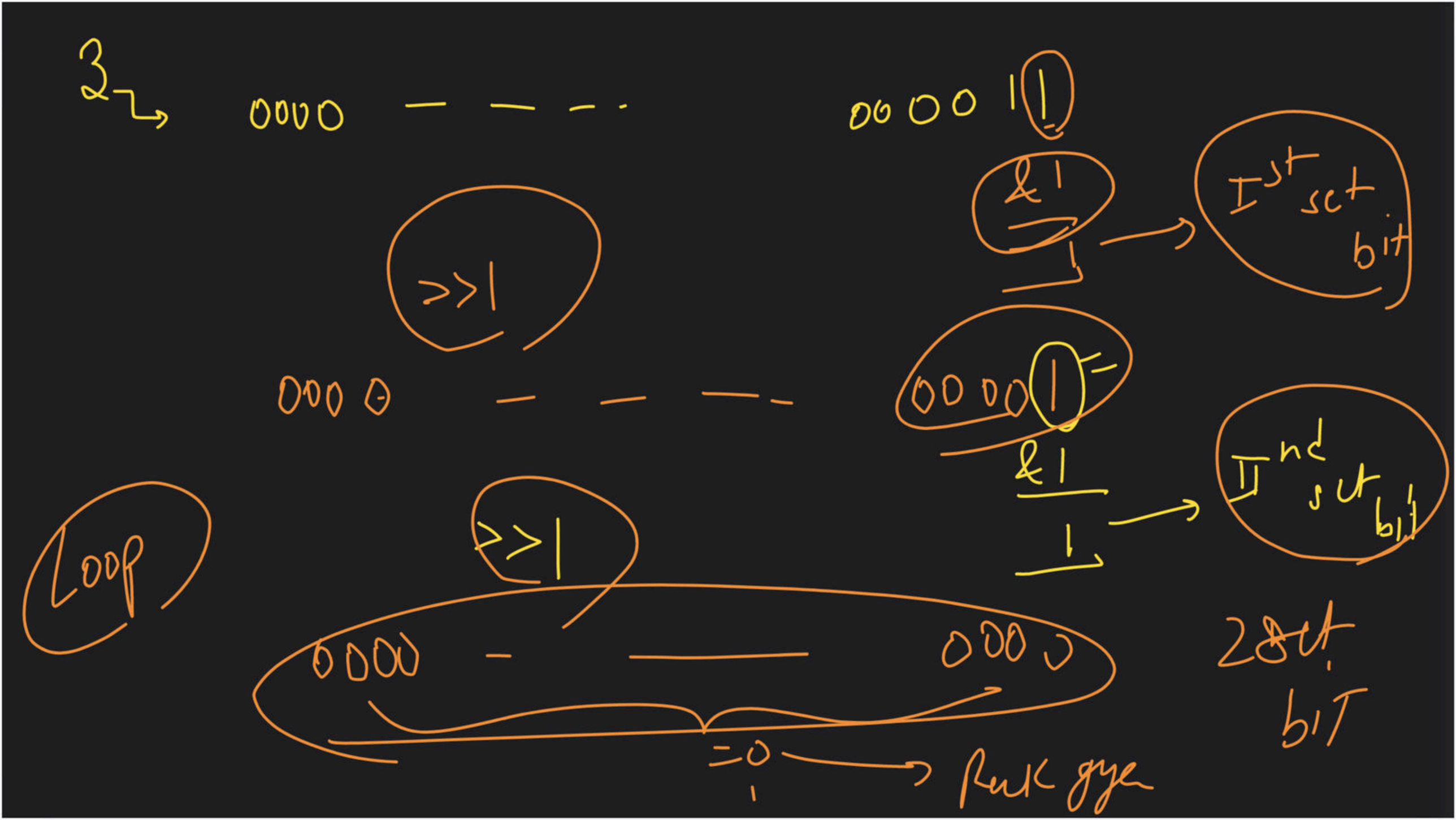
R.S. by → 3 bits

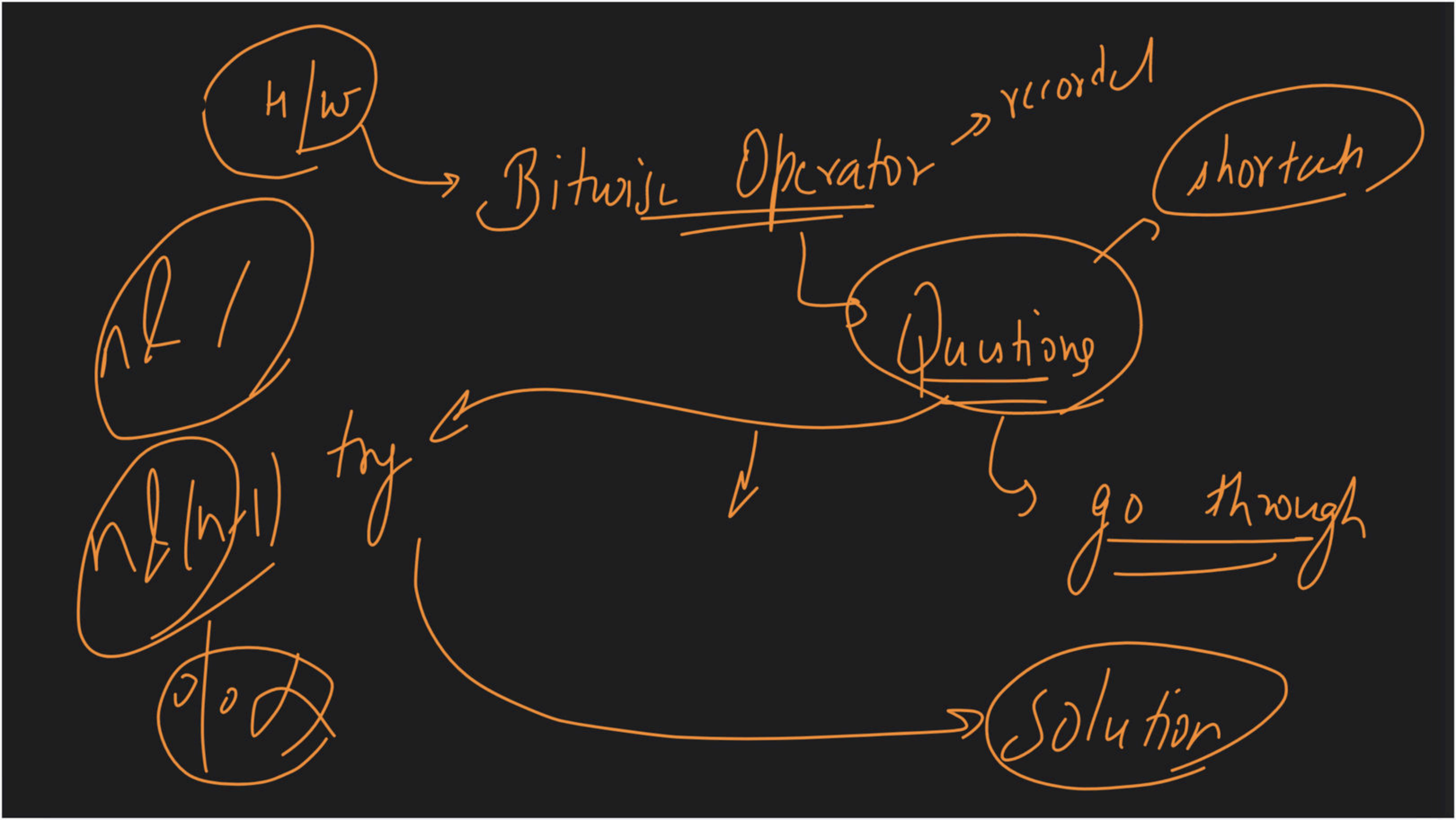
$\frac{20}{2^3} = \frac{20}{8} = 2$

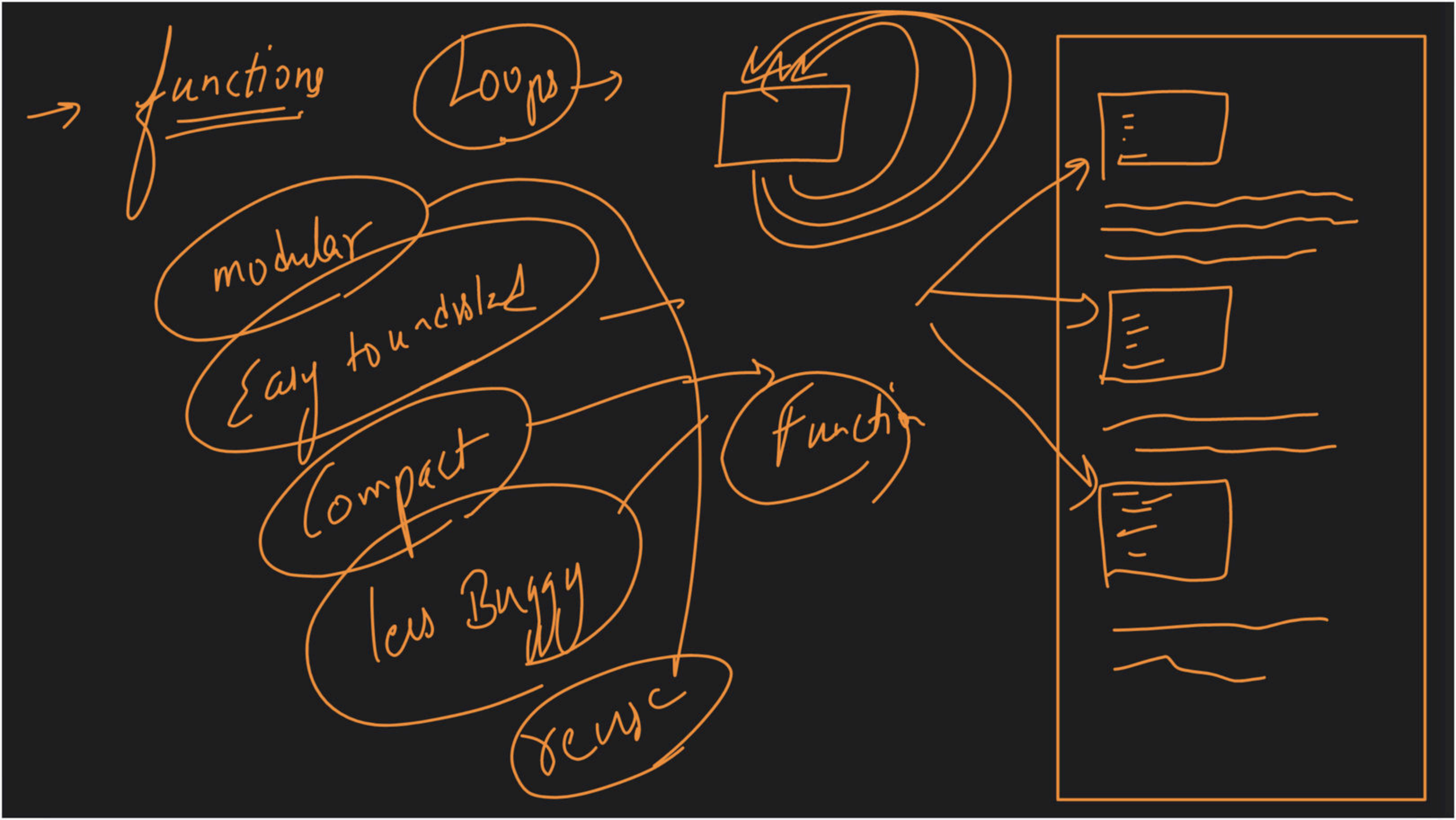

5
00000
000
sct
sct
count of
sct. bit
&1

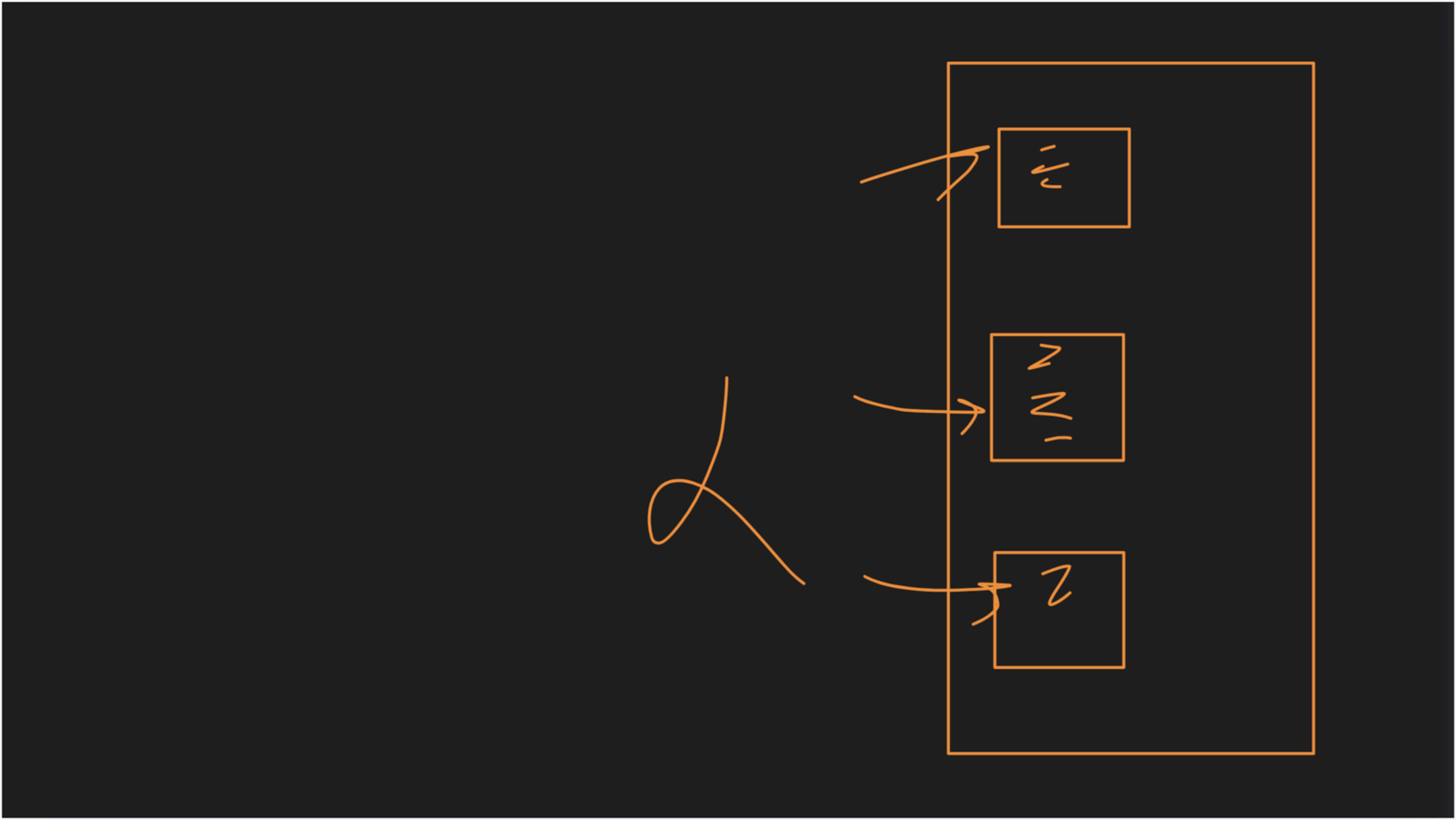

5
0000 - - - -
0000 1 0
&1
1
set bit
0 / 1
>>1
000 - - - 000 1 0
&1
0
>>1
000
000 0
&1
1
>>1
0 / 1

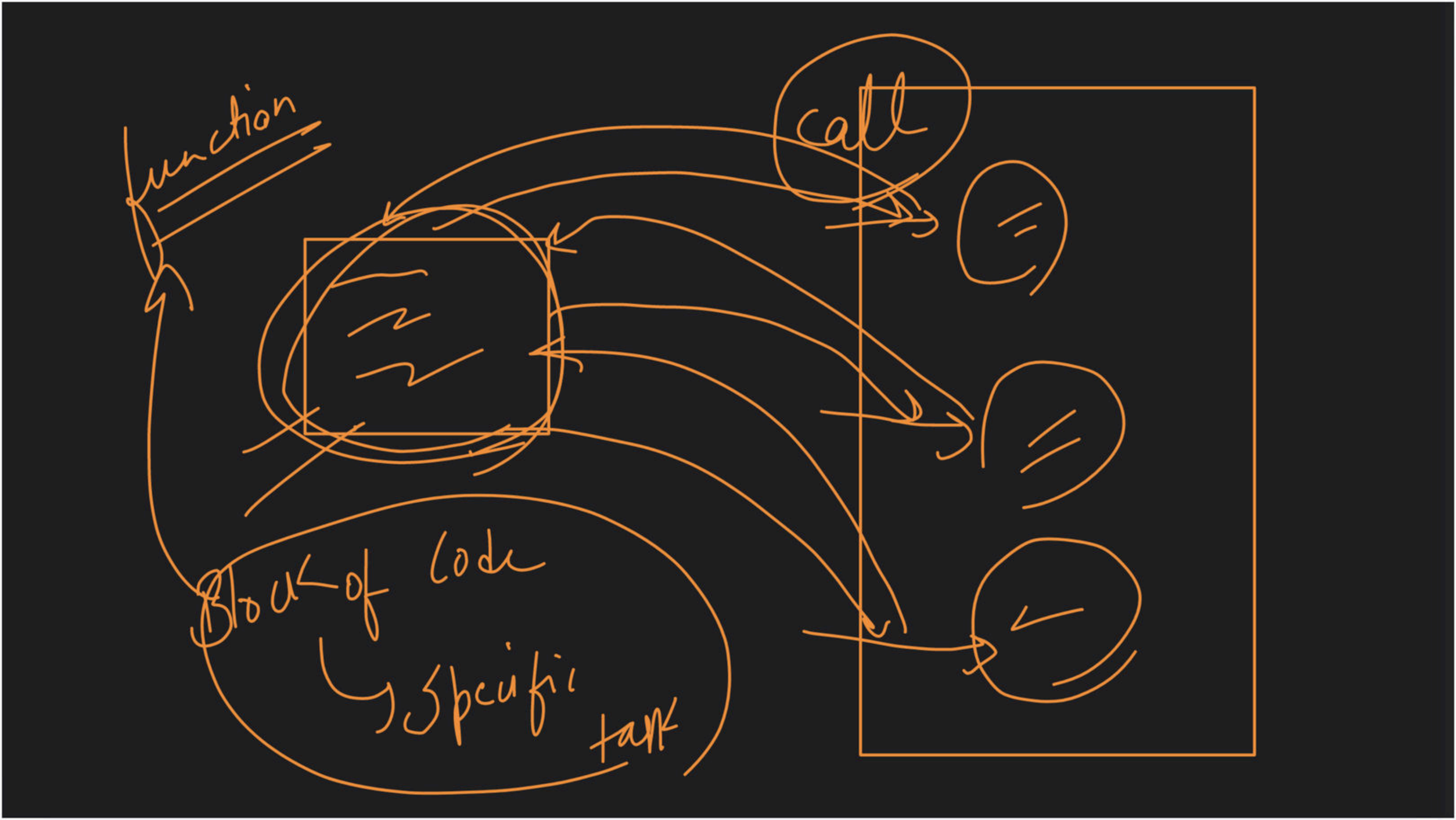

=0

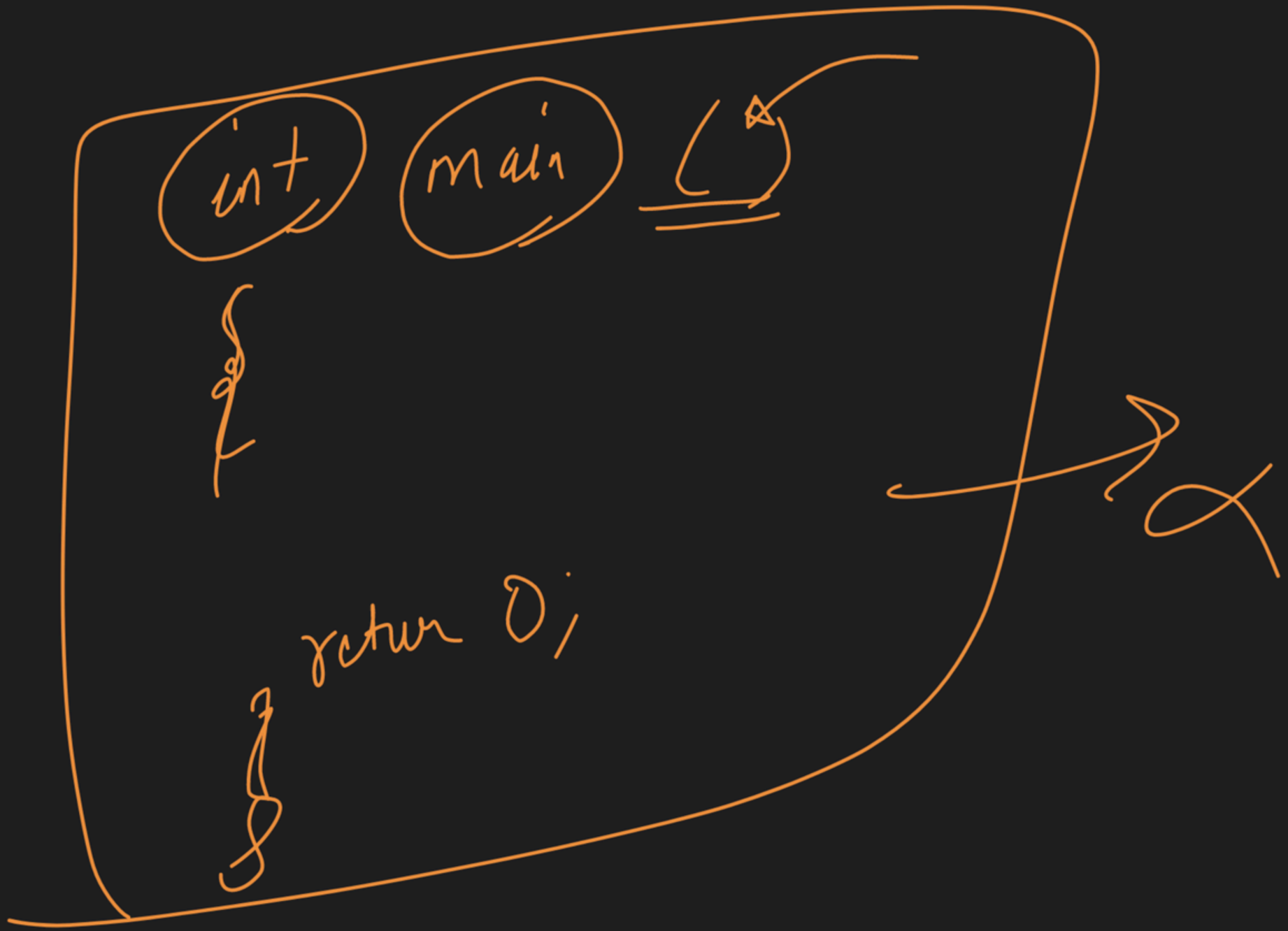

3
0000
0000 1
&1
1st set bit
>>1
0000
0000 1
&1
1
2nd set bit
Loop
>>1
0000
000 0
=0
1
Break gya
2set bit


H/W
Bitwise Operator
recorded
shortcut
Questions
try
go through
Solution

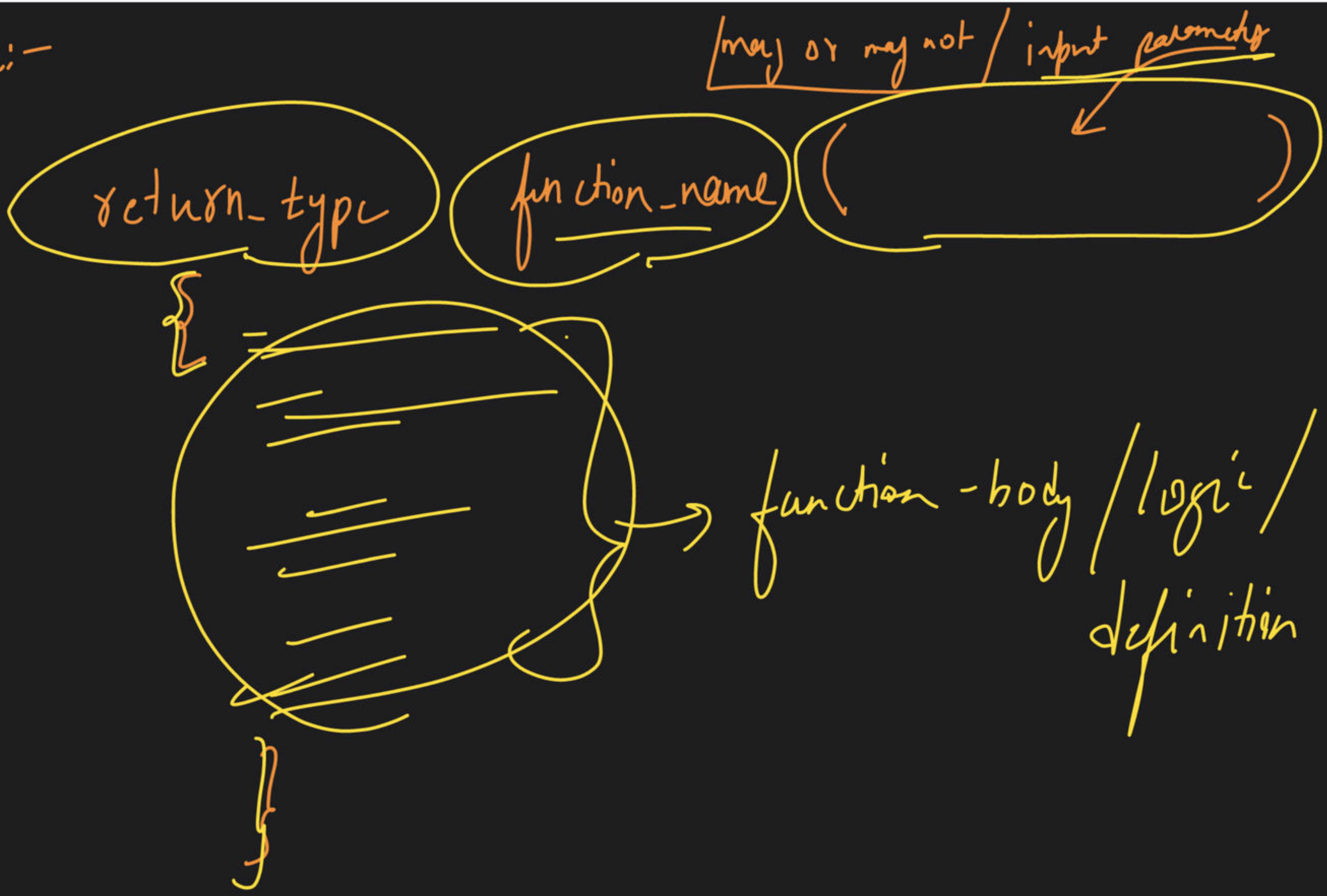

functions
Loops
modular
Easy to understand
Compact
less Buggy
reuse

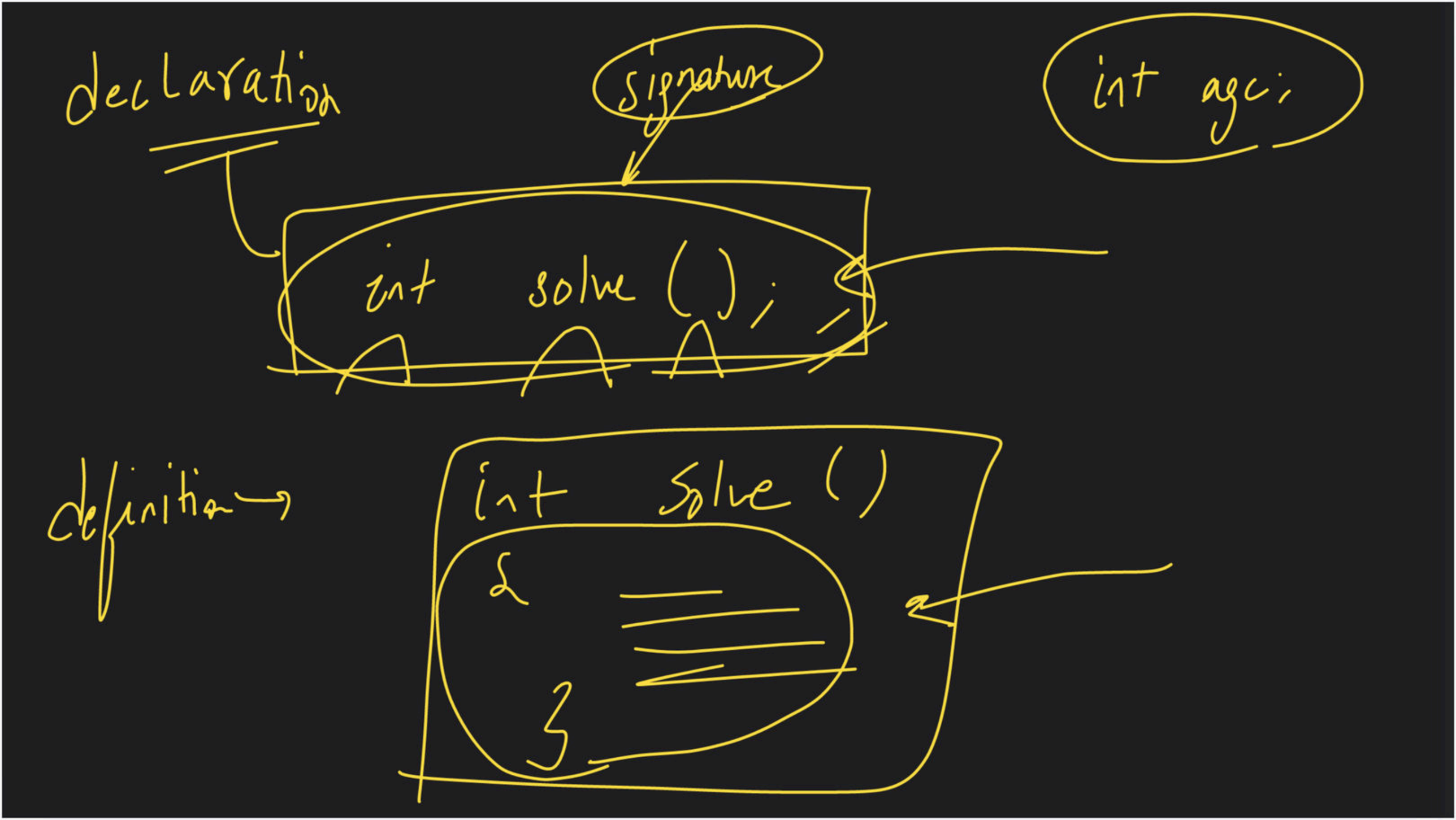


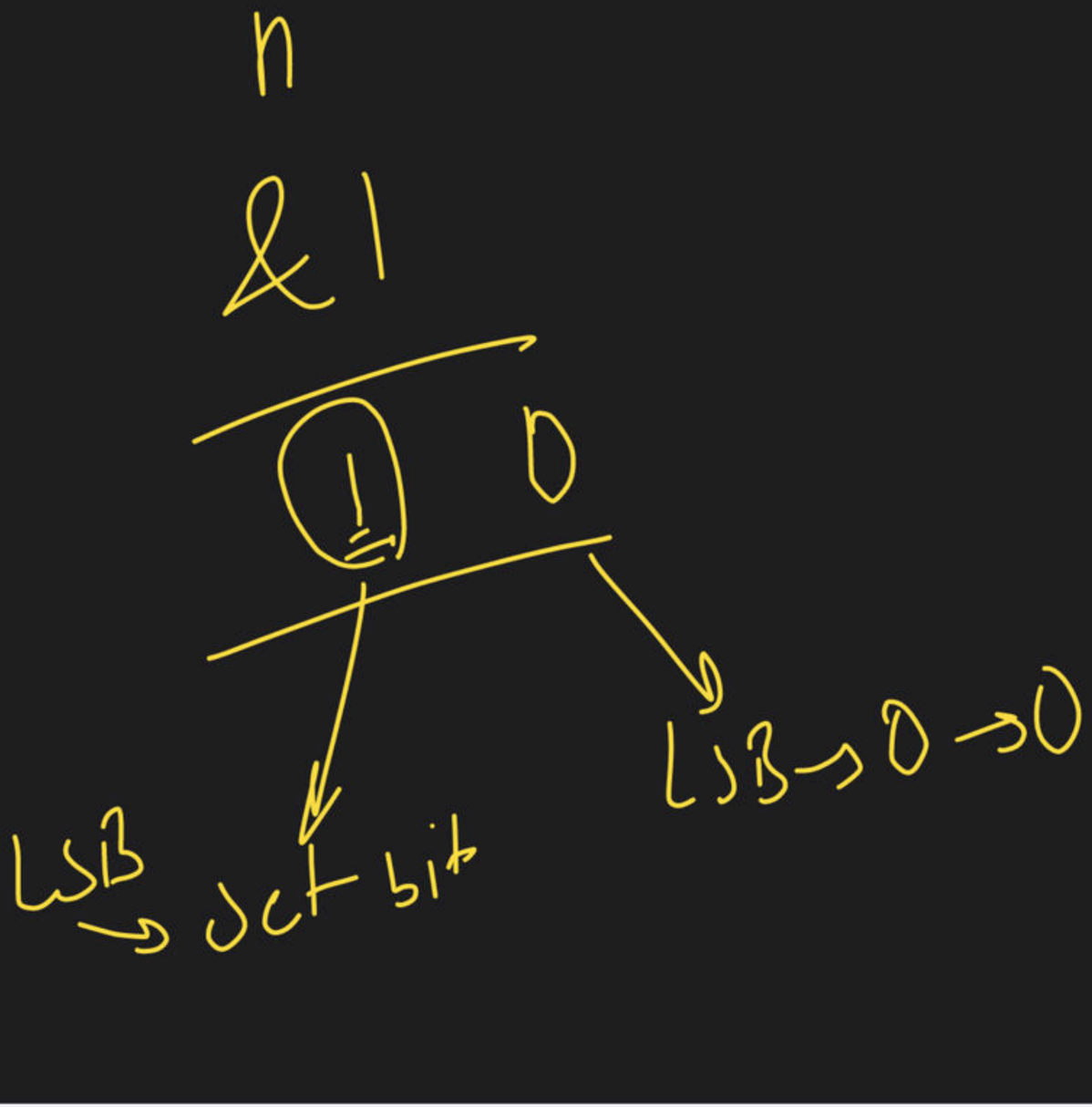

function
call
Block of code
specific task

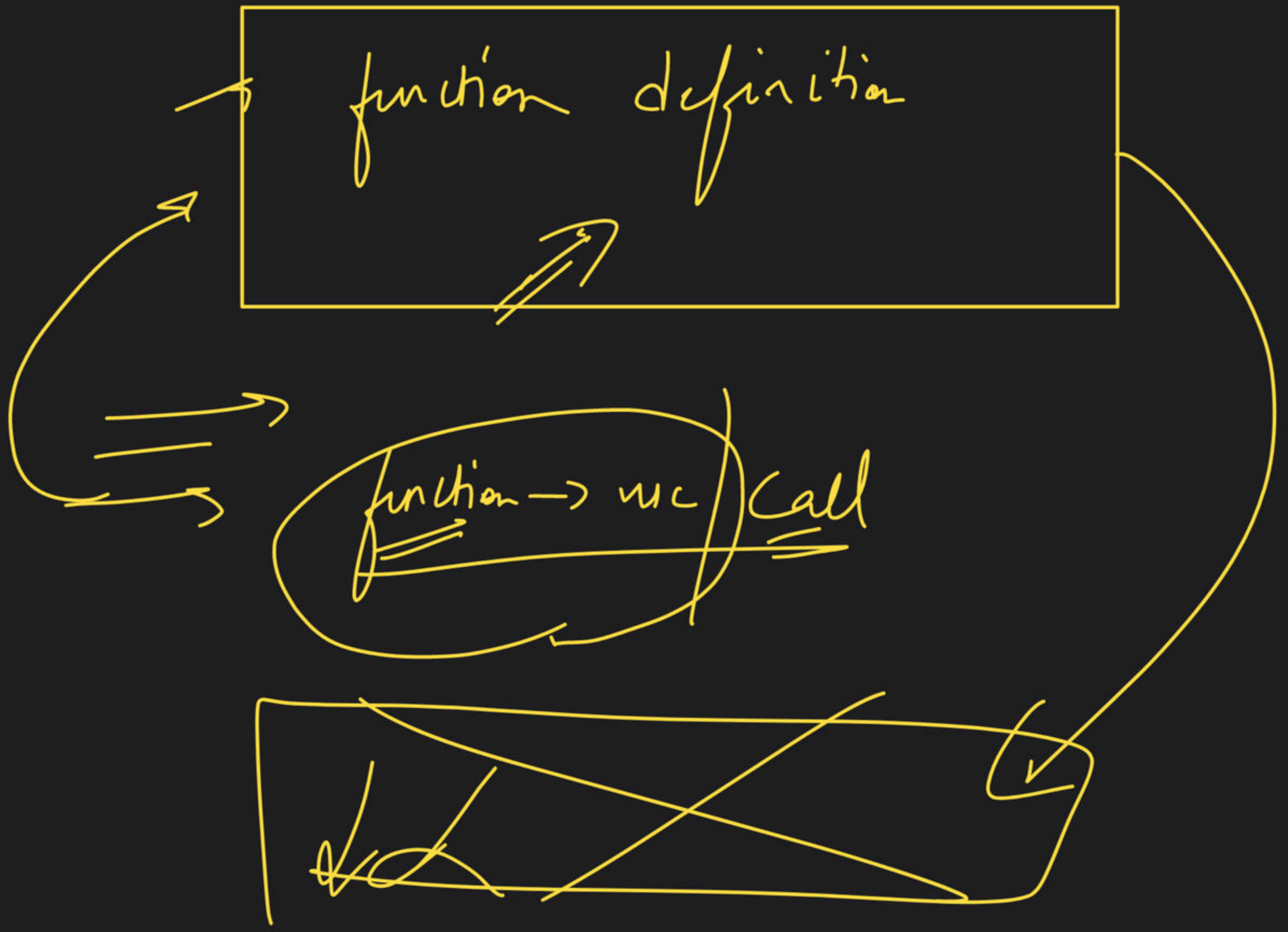

int
main
return 0;

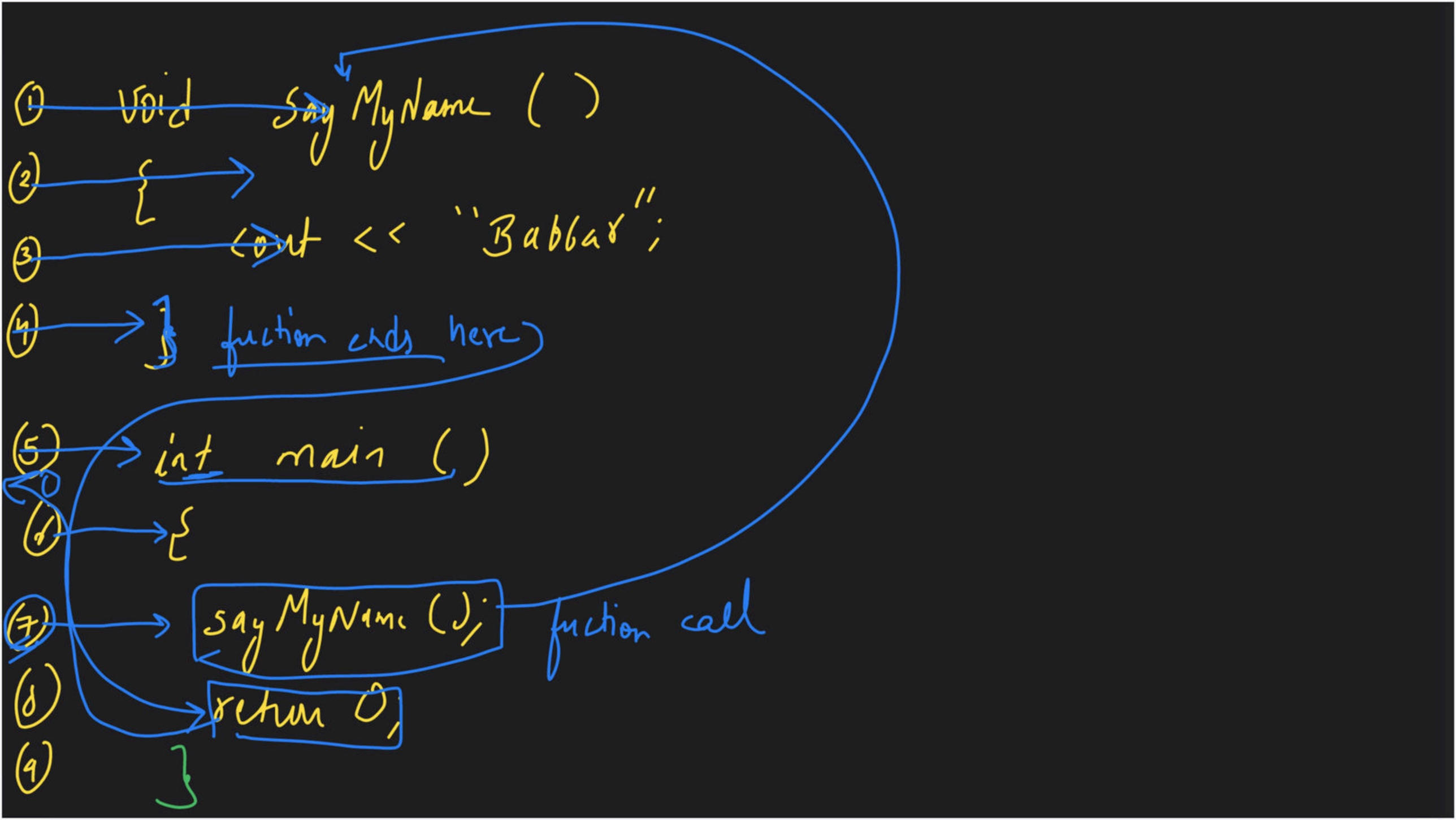
Syntax:-

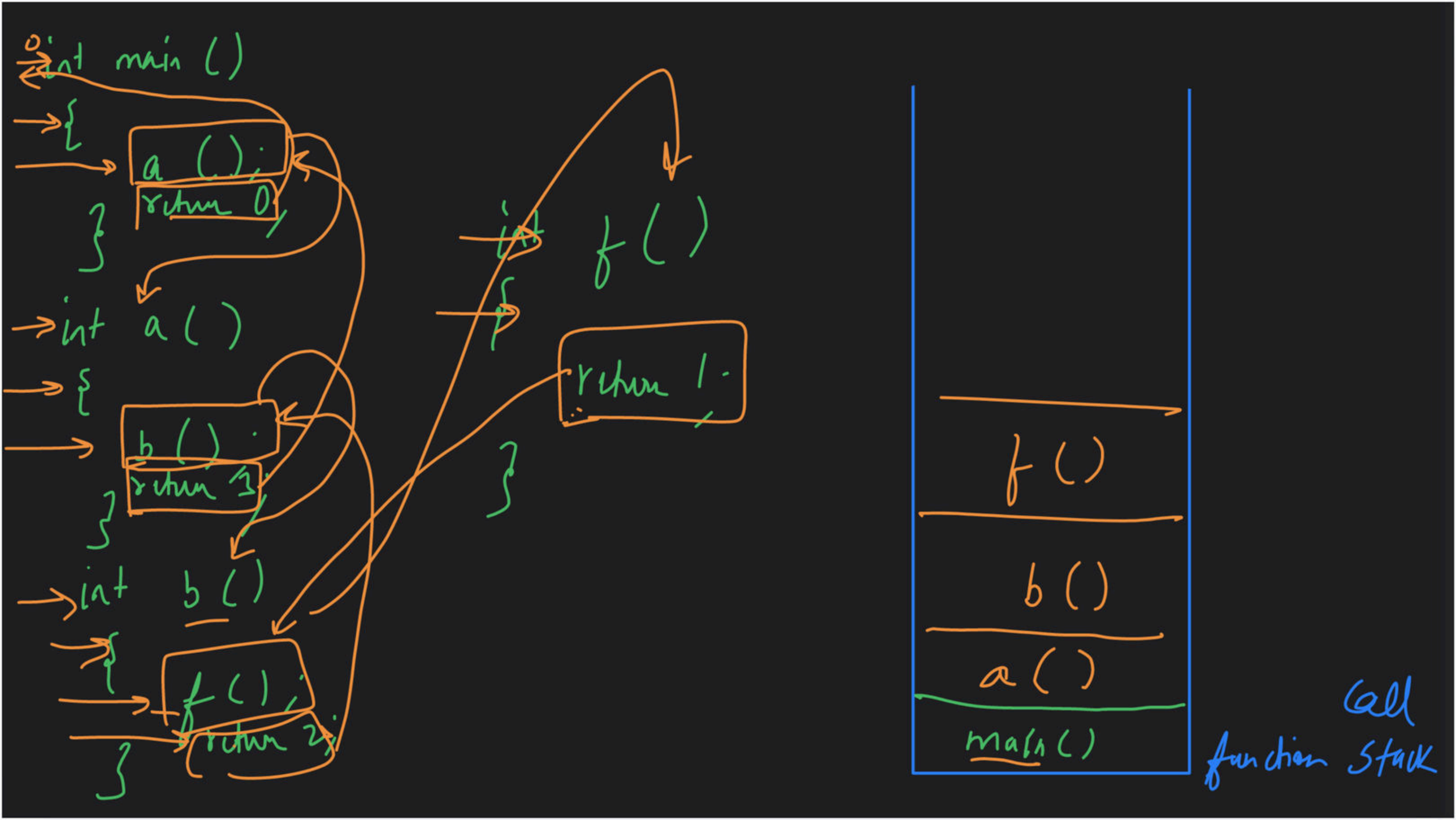

(may or may not) / input parameters
return_type
function_name
(
{
function-body / logic / definition
}

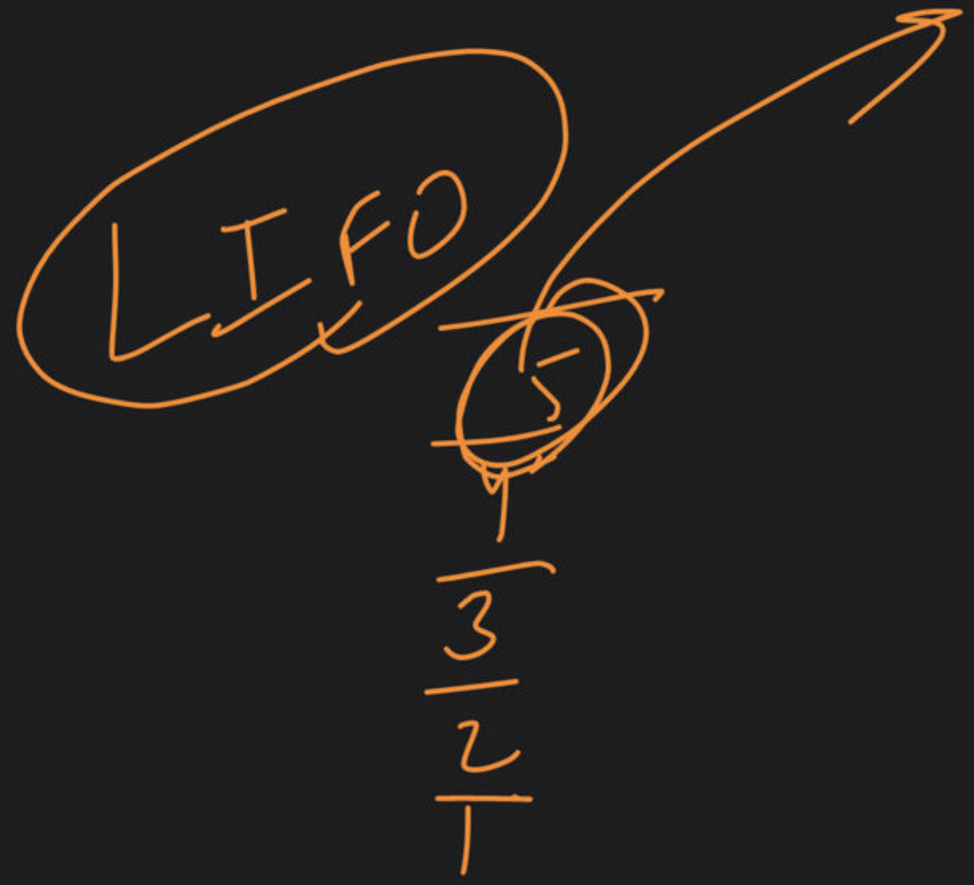

declaration
signature
int age;
int solve ();
definition
int solve ()
{
}

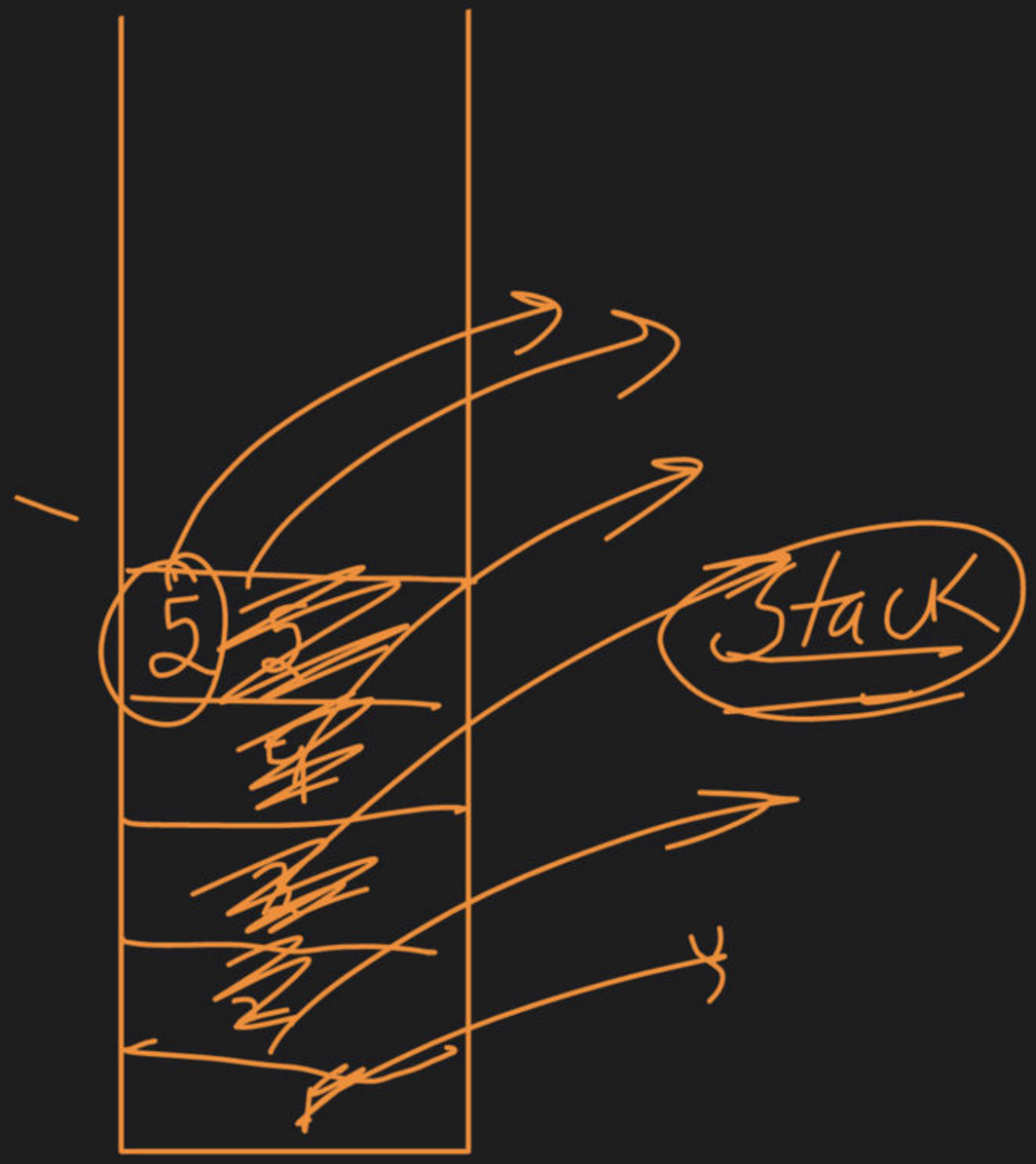

n
& 1
1
0
LSB → set bit
LSB → 0 → 0


function definition
function -> recursive call

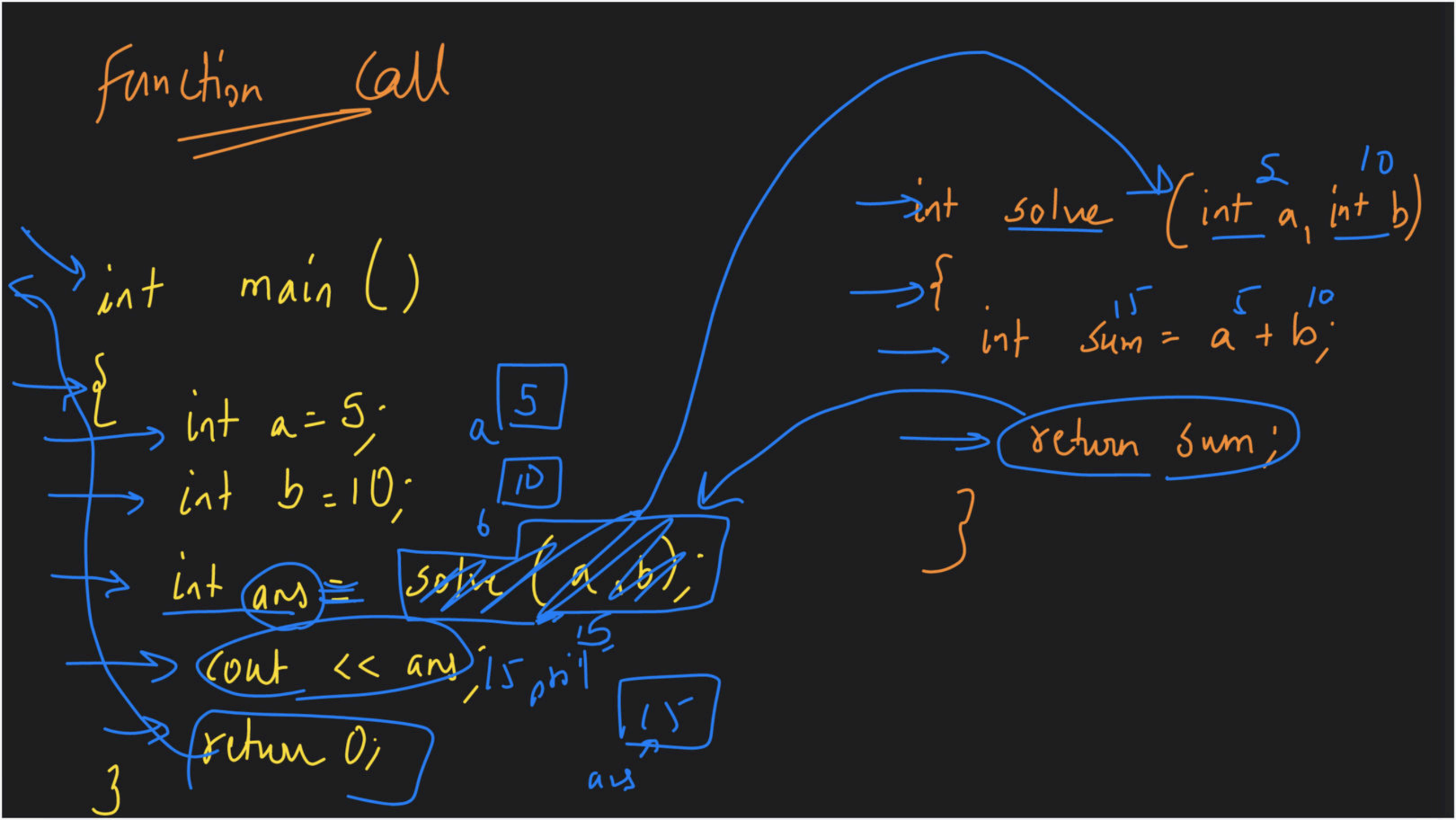

1 void sayMyName ( )
2 {
3 cout << "Babbar";
4 } function ends here
5 int main ( )
6 {
7 sayMyName ( ); function call
8 return 0;
9 }


int main ()
{
a ();
return 0;
}
int a ()
{
b ();
return 3;
}
int b ()
{
f ();
return 2;
}
int f ()
{
return 1;
}
f ()
b ()
a ()
main ()
Call function Stack


LIFO
5
3
2


5
Stack

Function Call



Call stack

```
int main()
{
    cout << 1;
    solve();
    cout << 2;
    return 0;
}
```

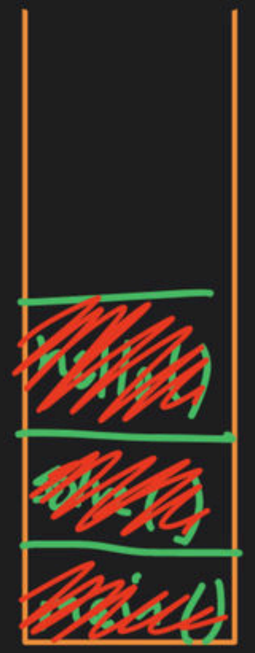
```
void solve()
{
    cout << 3;
    hello();
    cout << 4;
}
```

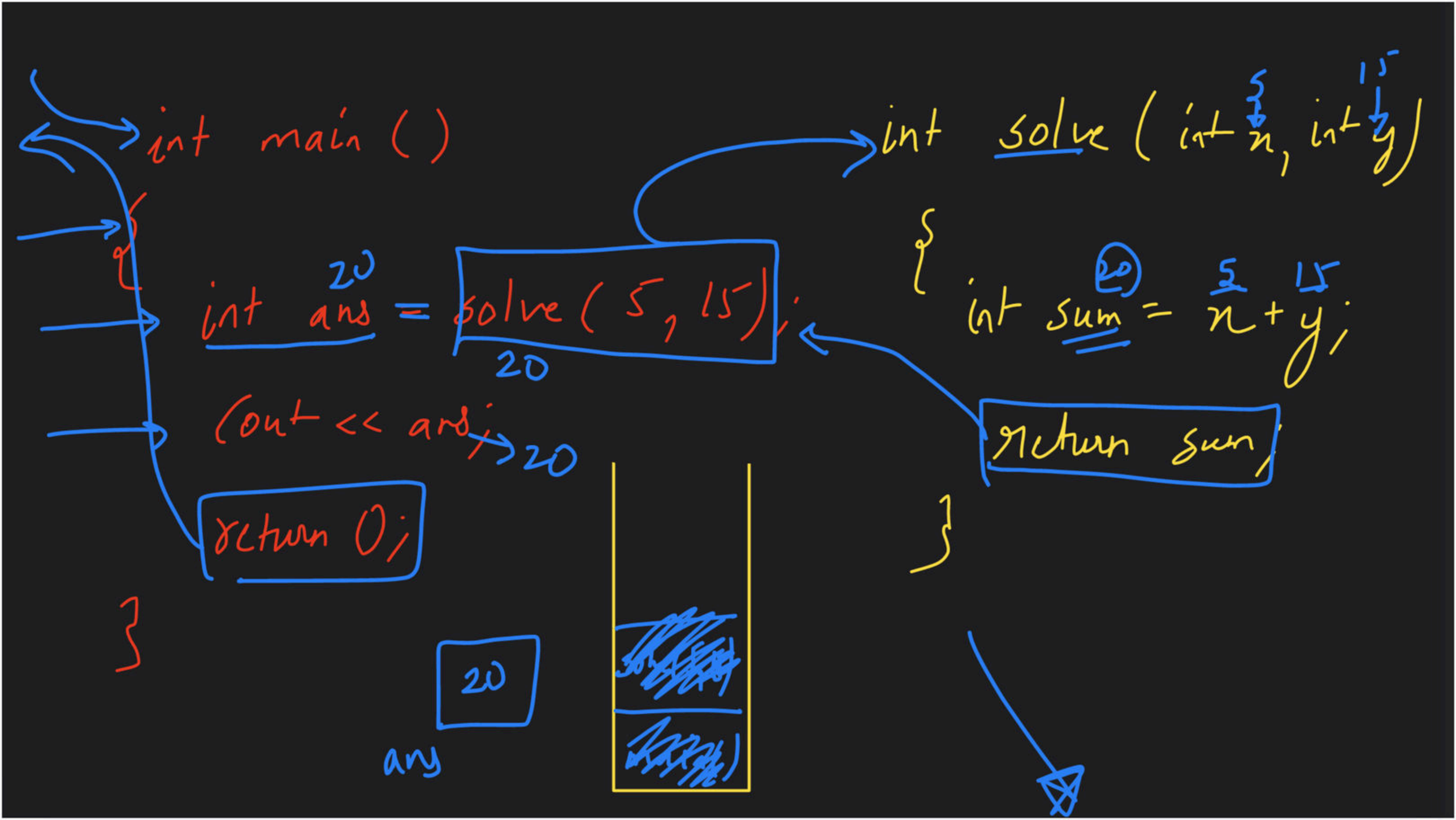
```
void hello()
{
    cout << 9;
}
```

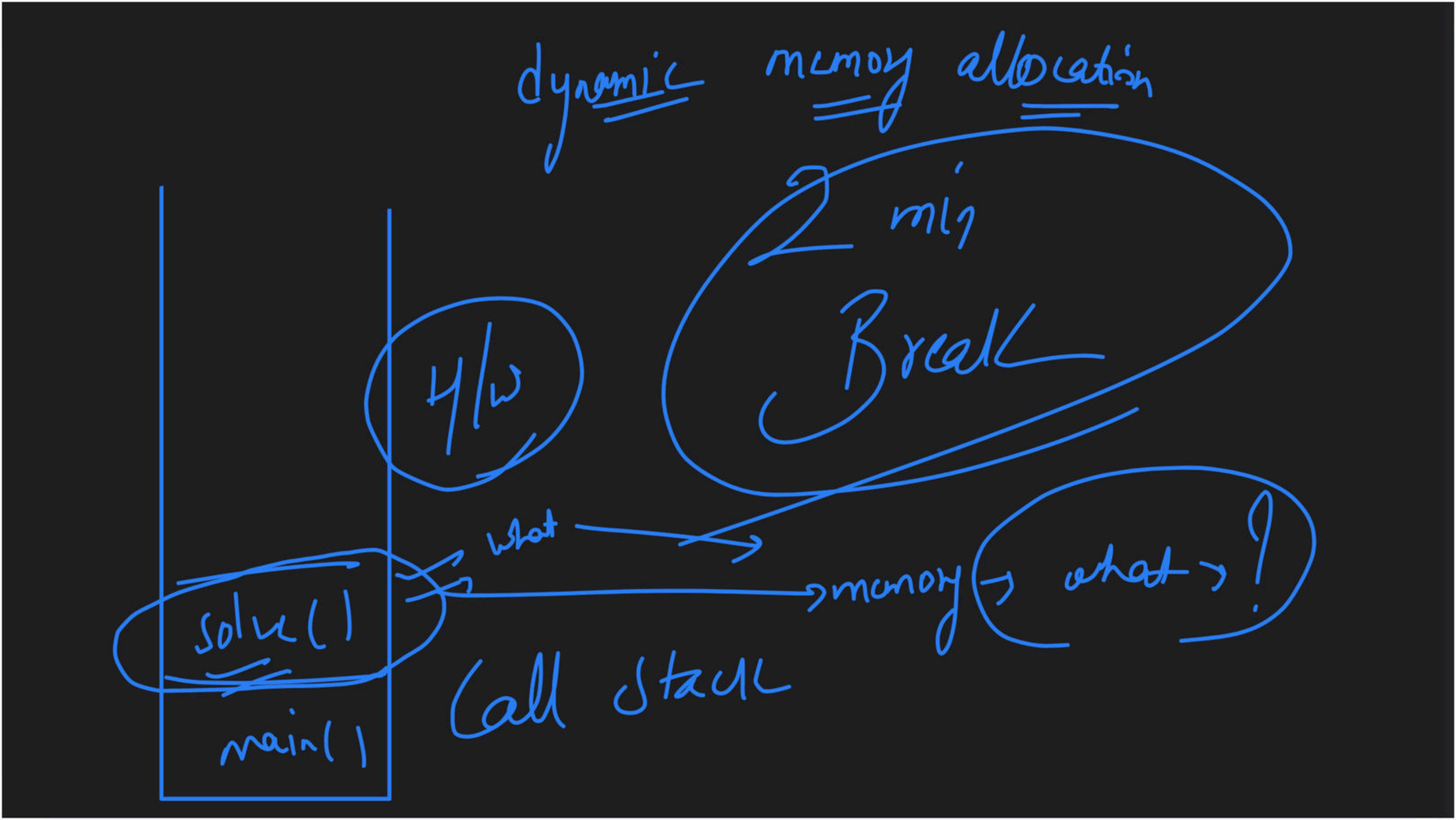
function khtm

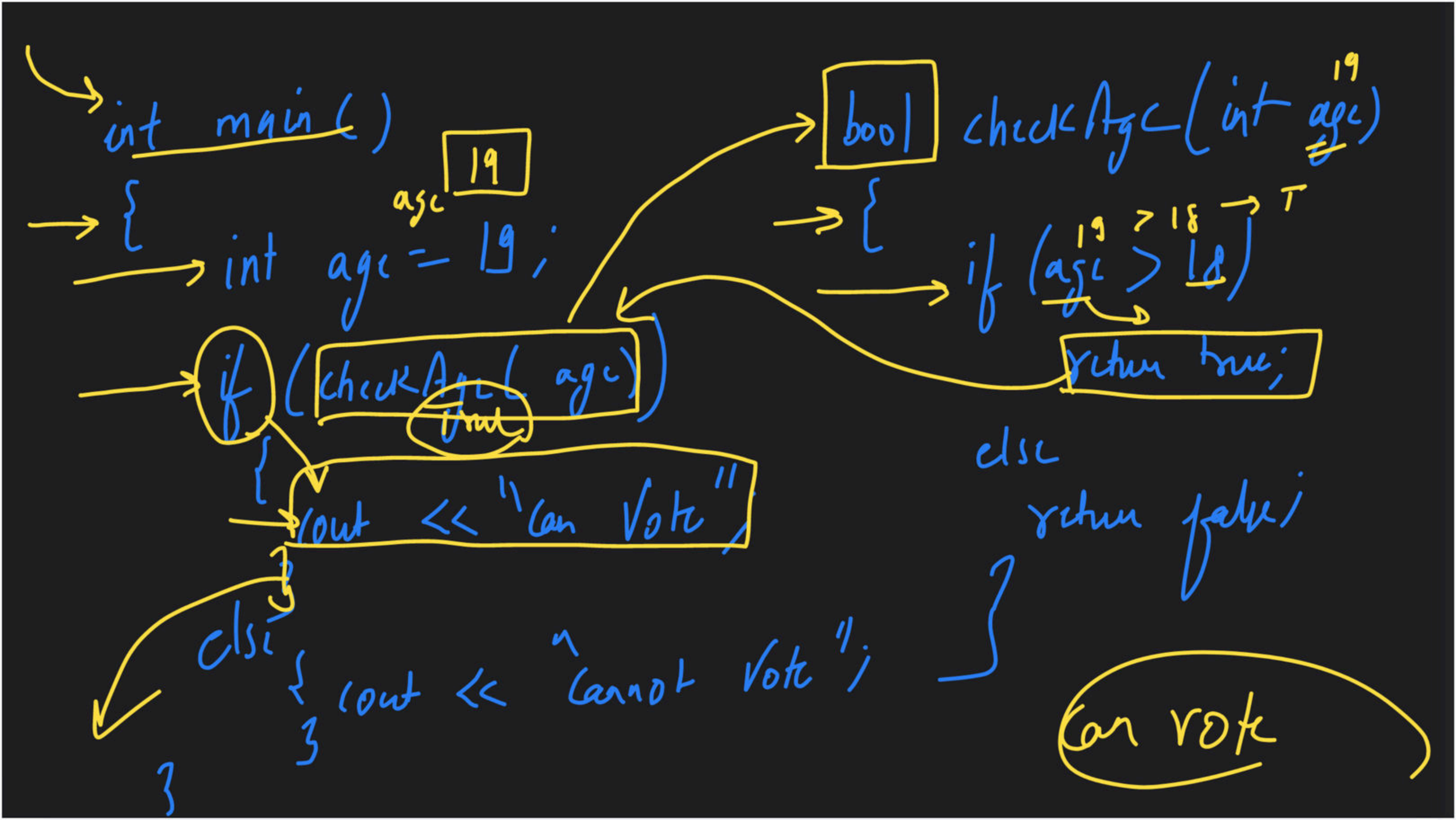
function khtm

print → 1 3 9 4 2




int main ()
{
int ans = solve (5 , 15);
20
cout << ans; → 20
return 0;
}
int solve (int x, int y)
{
int sum = x + y;
20
return sum;
}
20
ans


dynamic memory allocation
malloc
Break
H/W
what
memory
what ?
solve()
main()
Call stack


int main()
{
int age = 19;
age
19
if (checkAge(age))
true
{
cout << "Can Vote";
}
else
{
cout << "Cannot Vote";
}
bool checkAge(int age)
19
{
if (age > 18)
19 > 18
T
return true;
else
return false;
}
Can vote

int main

Sum of A.P

2, 4, 6, 8, 10

n → no of terms

a → first term

l → last term

$$\text{Sum} = \frac{n}{2}(a+l)$$

$n = 5$

$a = 2$

$l = 10$

$$= \frac{5 \times 12}{2} = \frac{60}{2} = 30$$

$$= \frac{5}{2}(2+10)$$

Prime no check

i/p → n → 15

1 ∝

n = 15 ⟶ 15∝

↳ rem → 0 → not prime

↳ !0 → prime

↳ True → prime

↳ false → not prime

-2

-3

-4

5 % 2 → 1

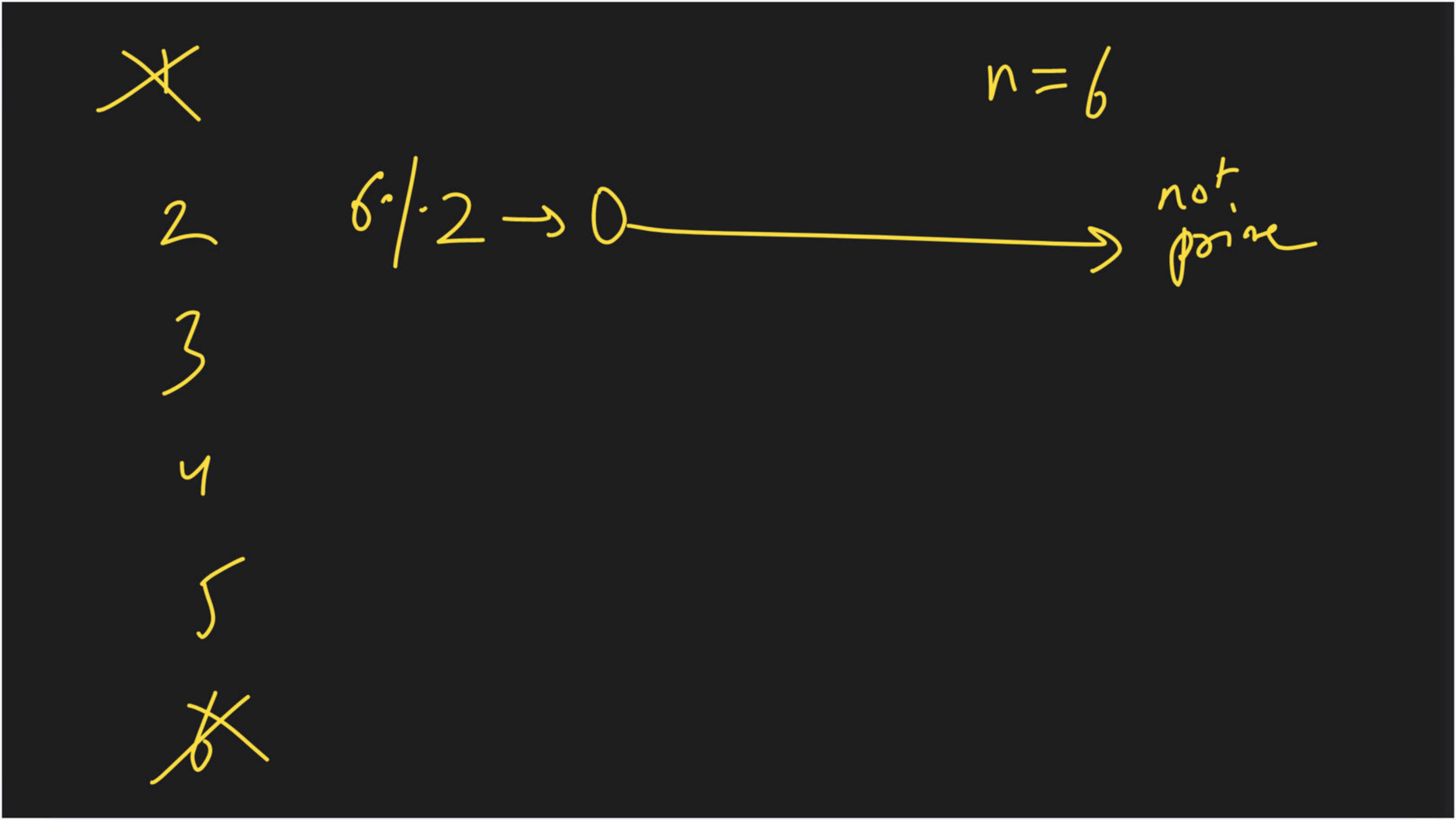
5 % 3 → 2

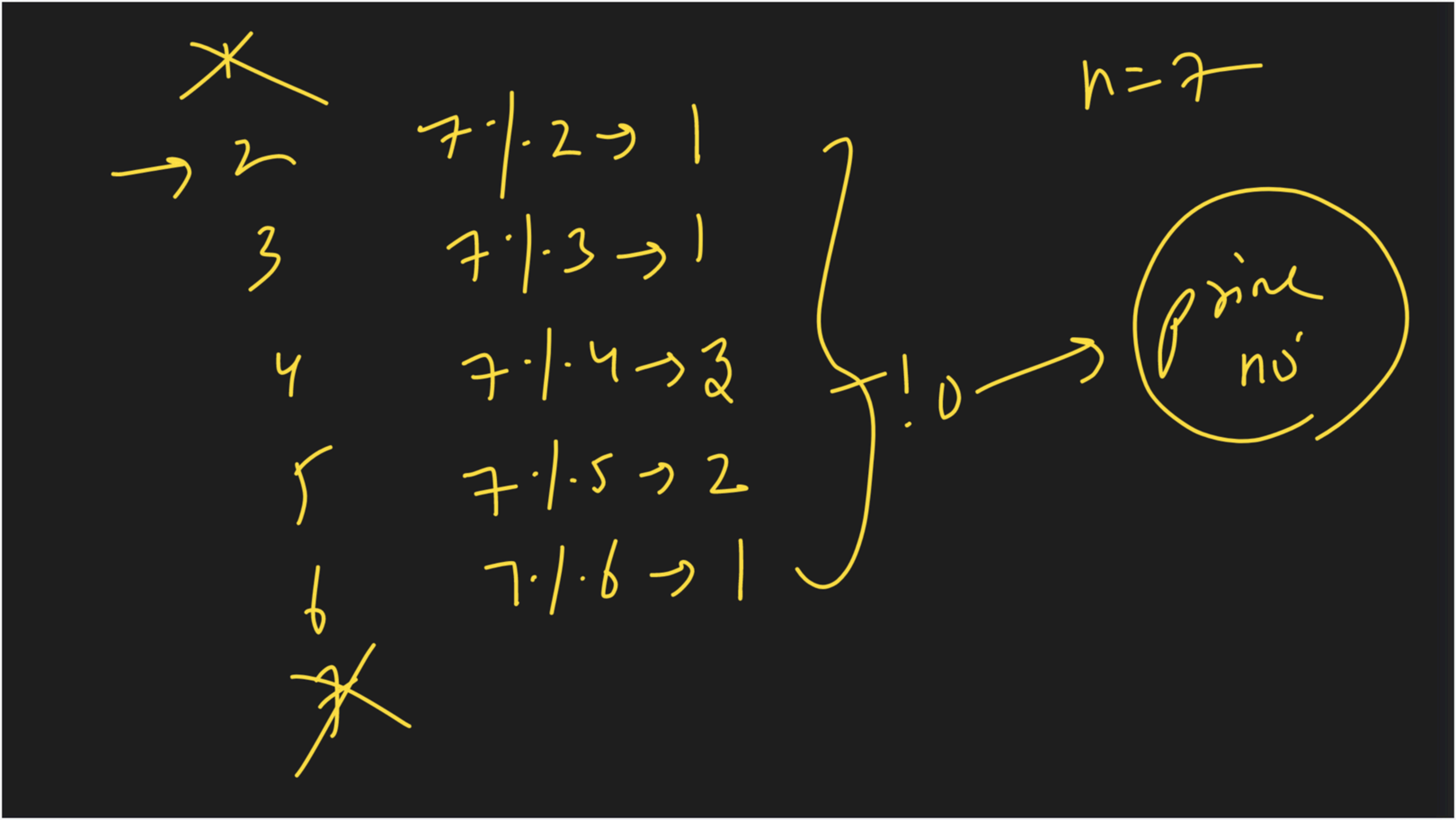
5 % 4 → 1

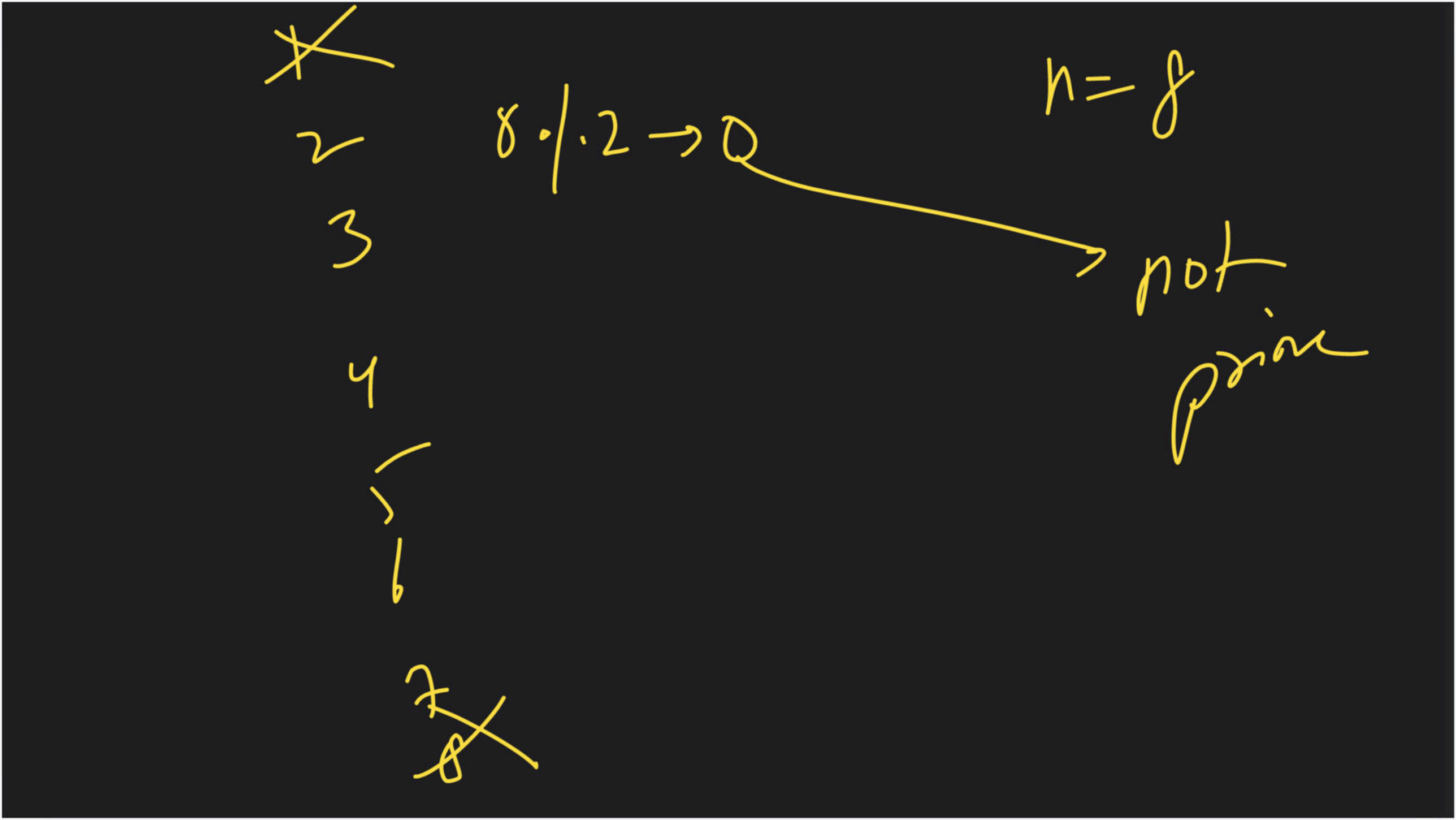
→ !0

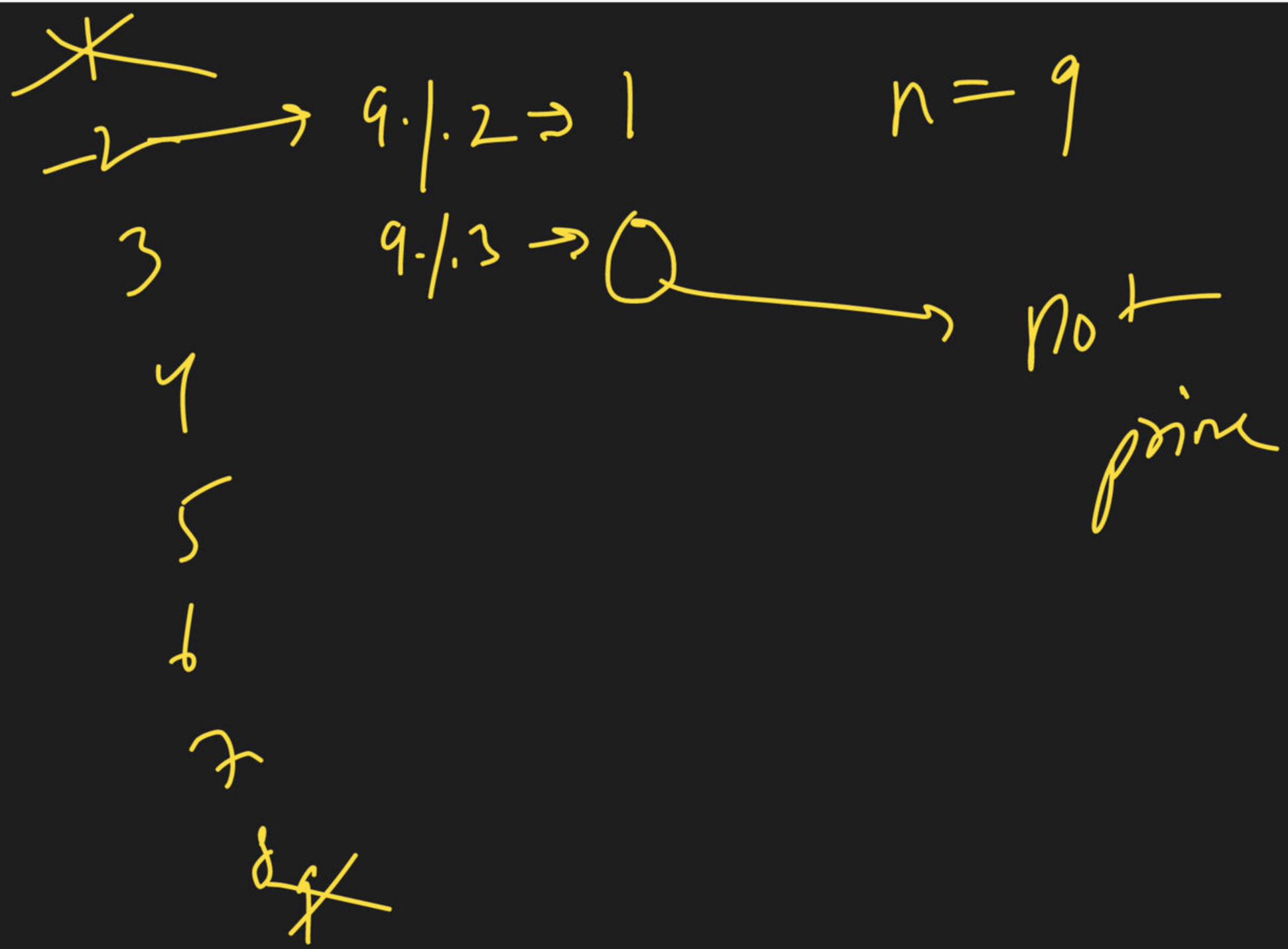
n = 5

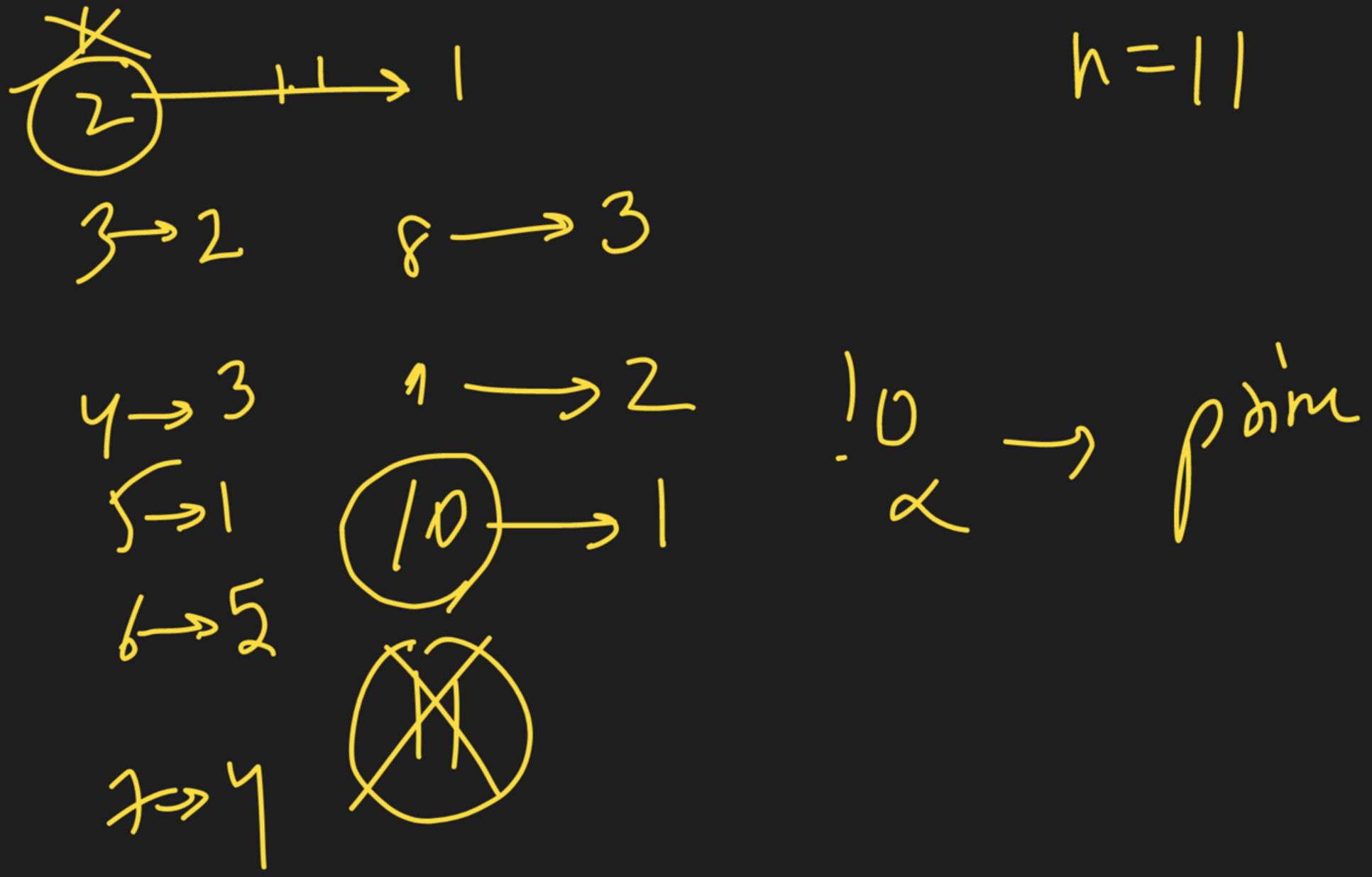
→ prime


n = 6
2
6 % 2 → 0
not prime
3
4
5
6


n=7
2
3
4
5
6
7%2 → 1
7%3 → 1
7%4 → 3
7%5 → 2
7%6 → 1
!0
prime no


8 % 2 → 0
n = 8
not prime
2
3
4
5
6
7
8

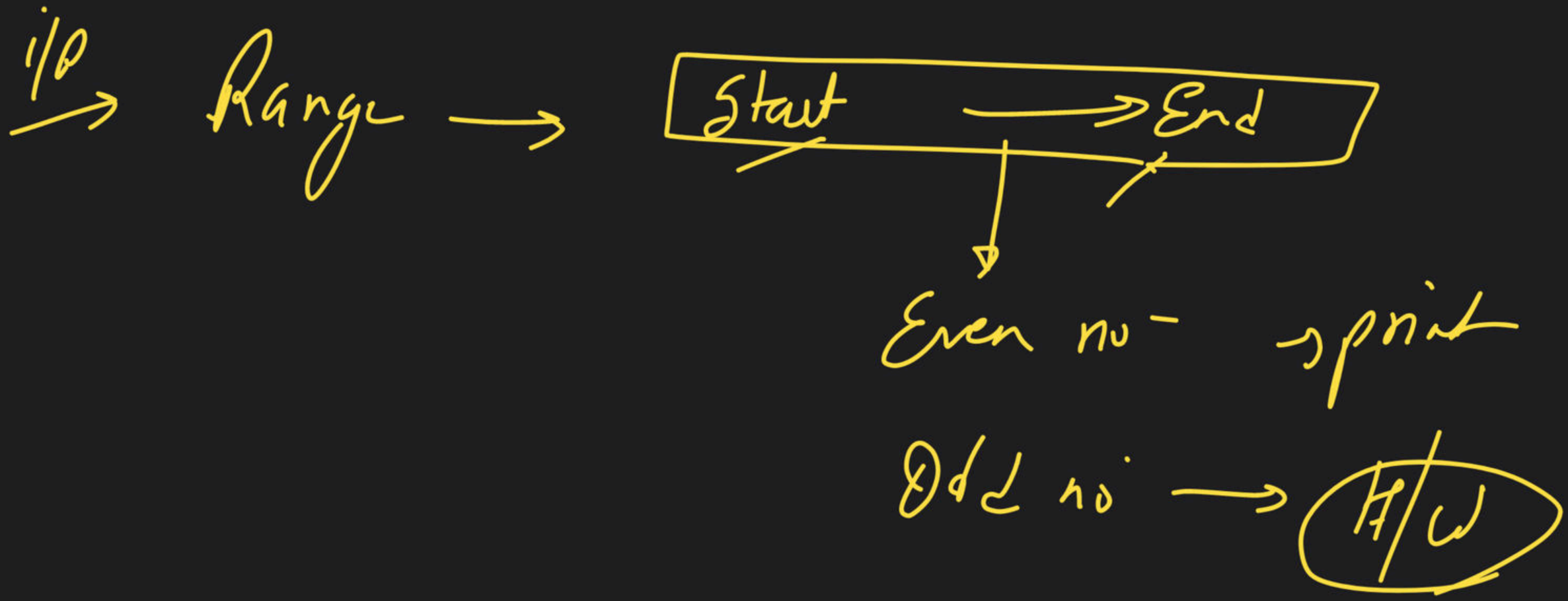

2
9 % 2 ⇒ 1
n = 9
3
9 % 3 → 0
not prime
4
5
6
7
8

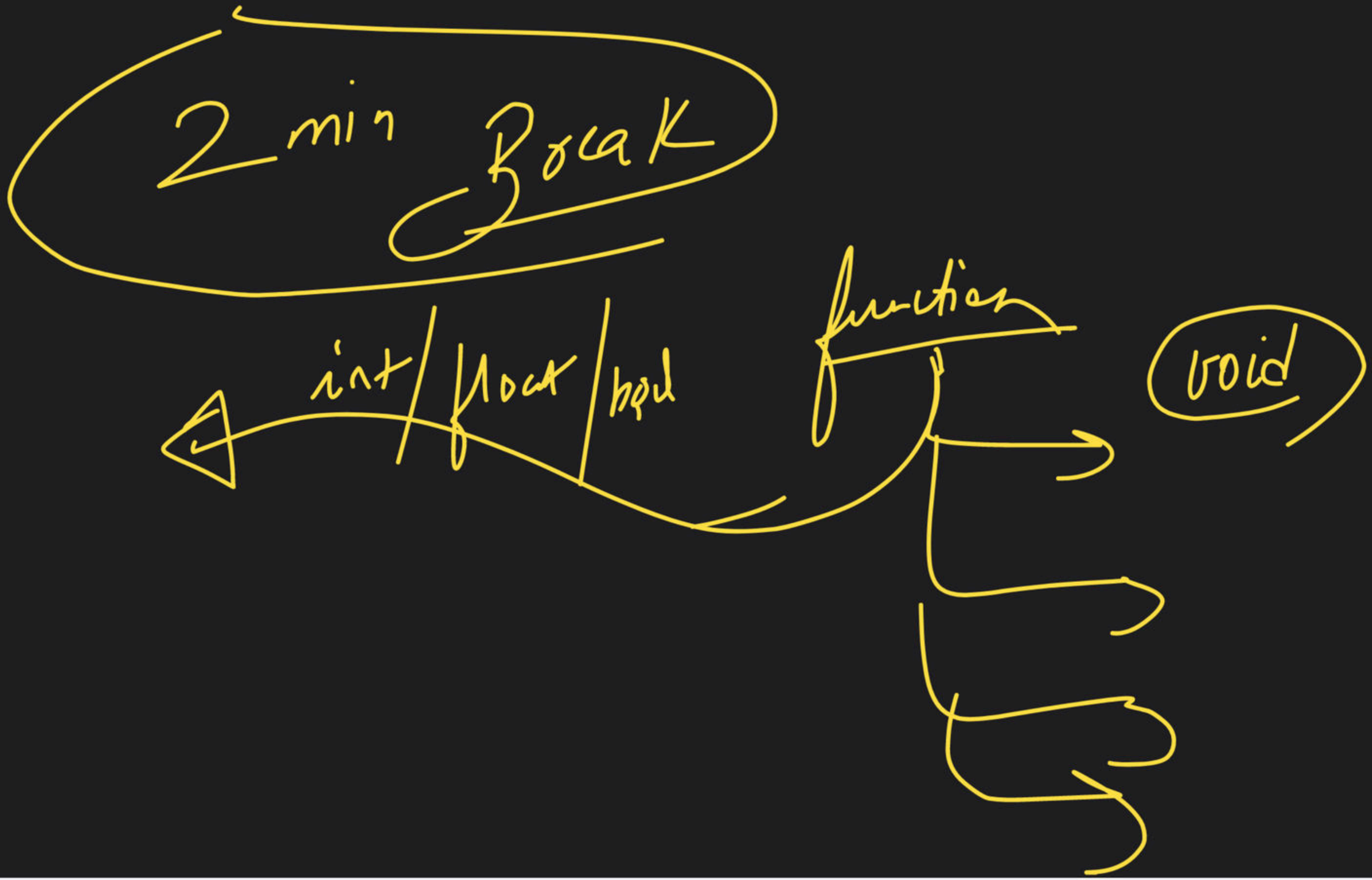

2
1
1
3 → 2
8 → 3
4 → 3
1 → 2
5 → 1
10
1
6 → 5
7 → 4
n = 11
10
prime

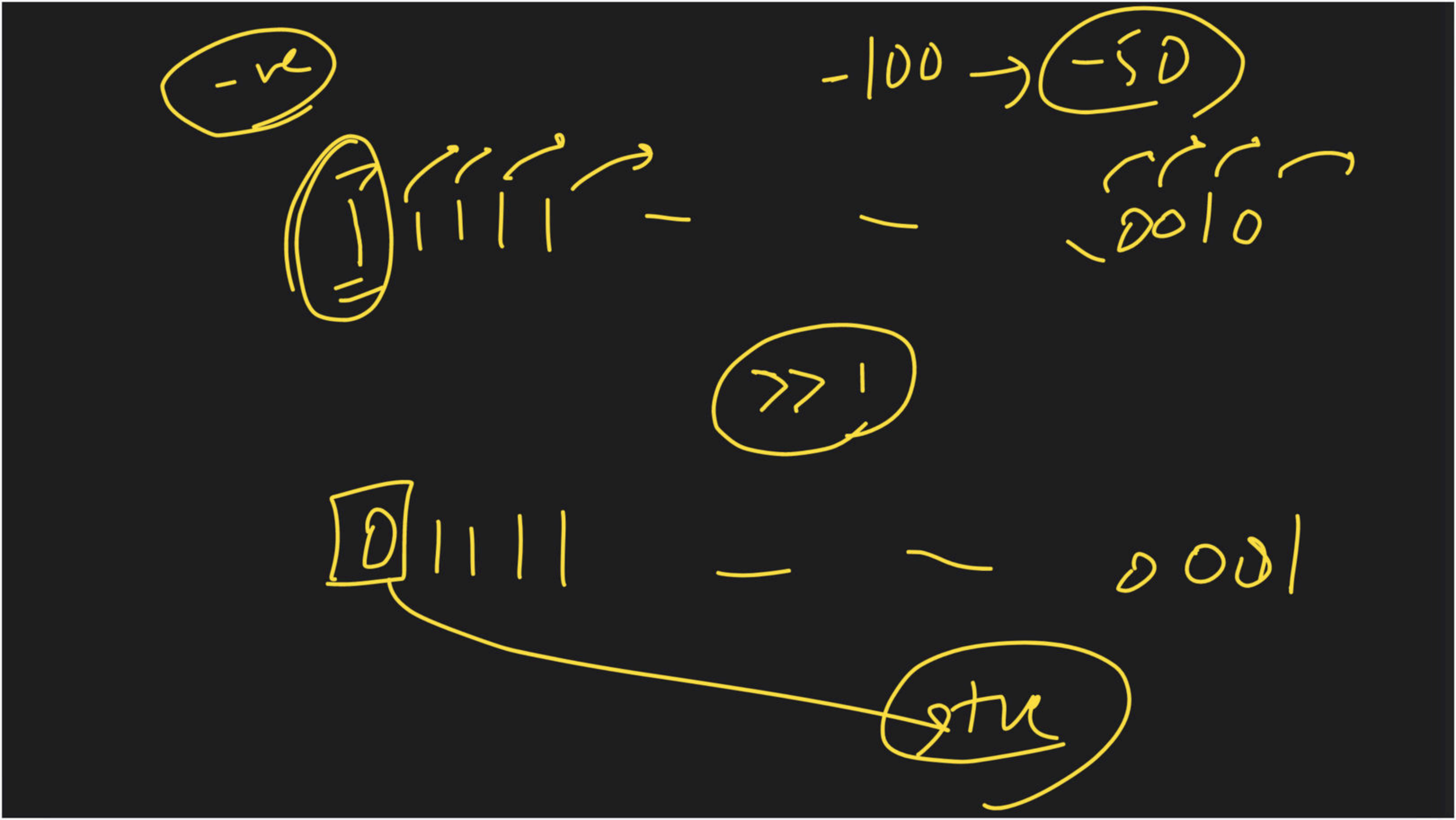
n = 13

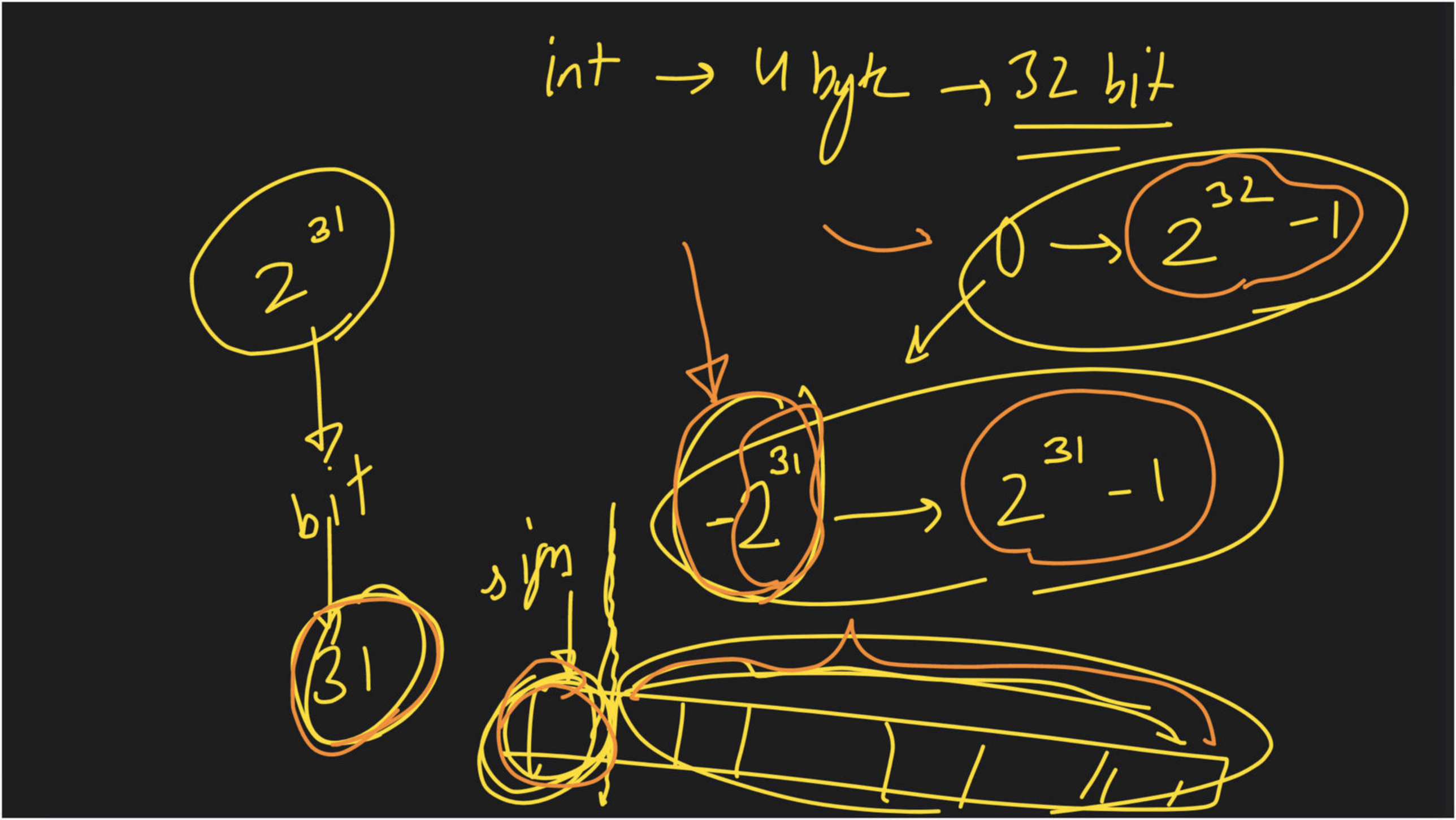
% → remain

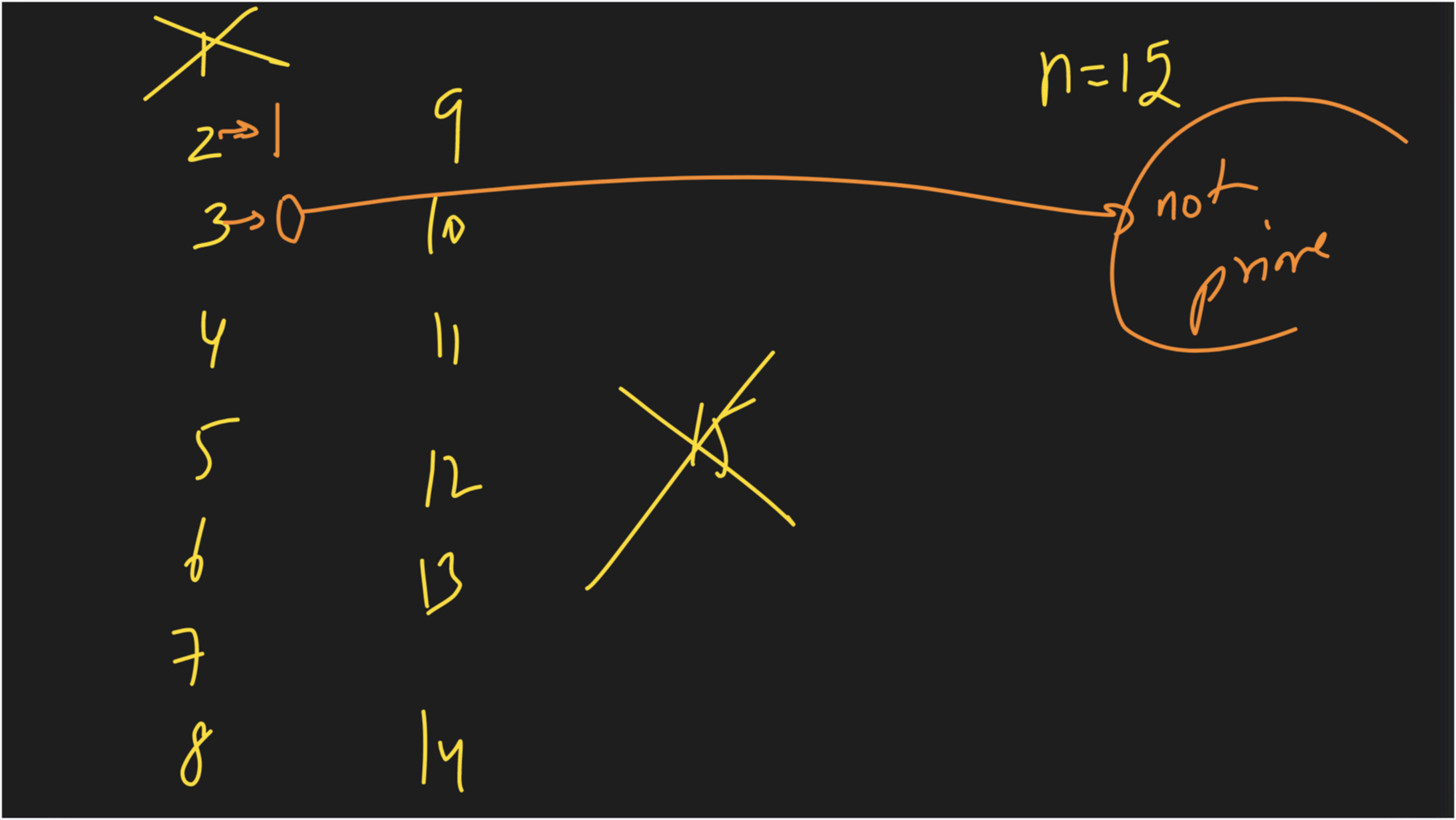
for (i = 2 → <= (n-1))

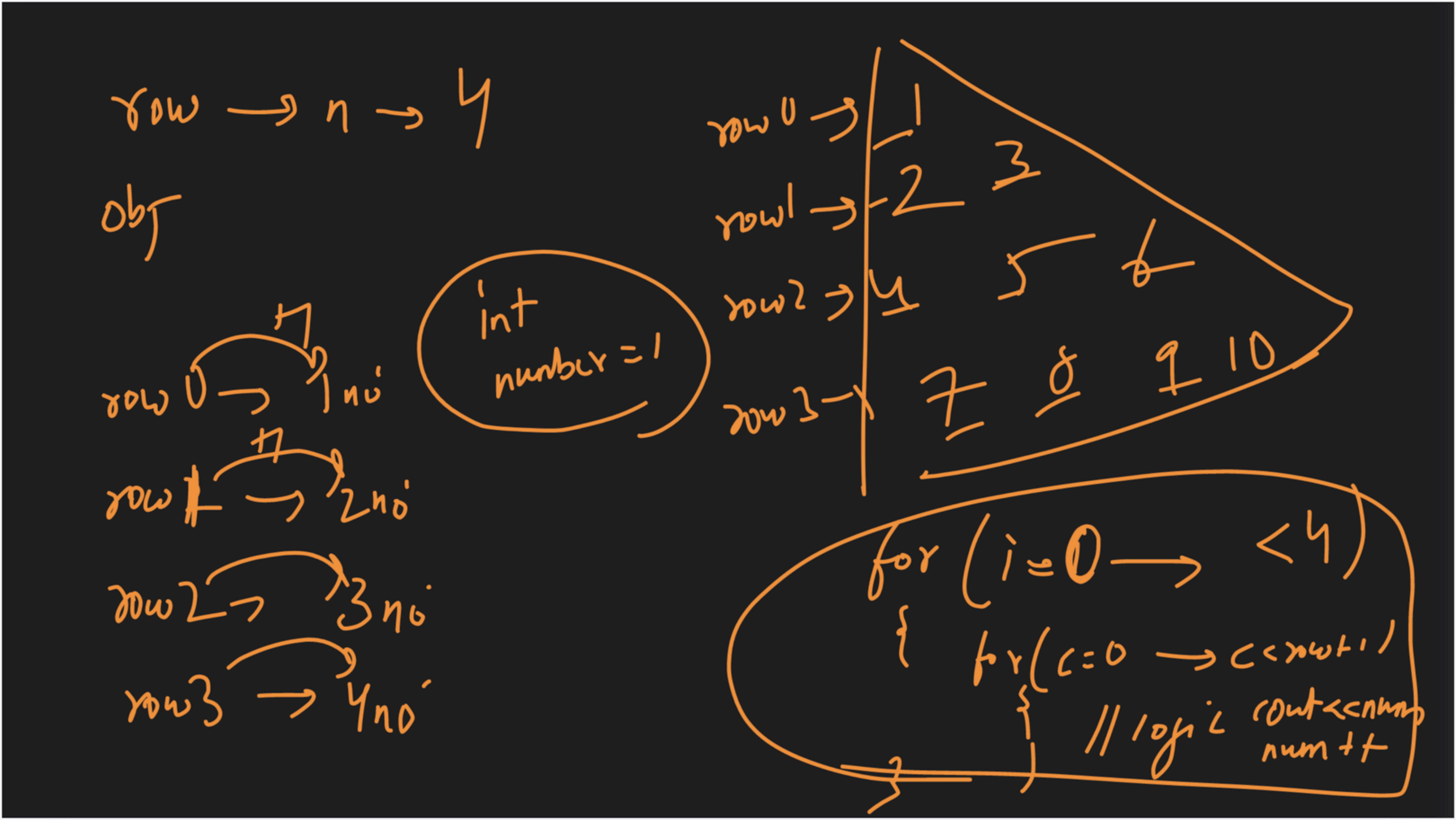

i/p
Range
Start
End
Even no - print
Odd no
H/W


2 min Break
int / float / bool
function
void

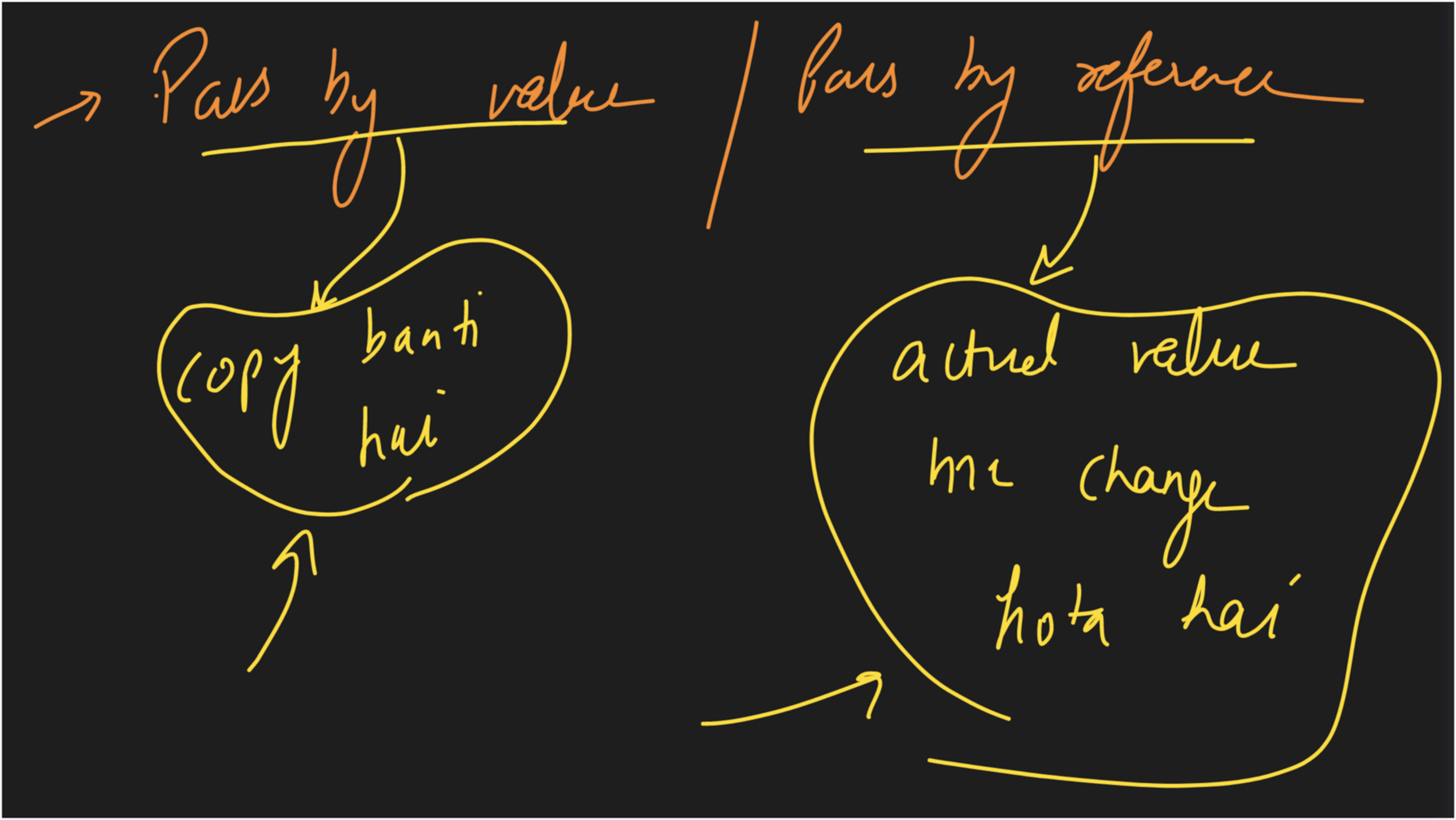

-ve
-100 → -50
>> 1
stu

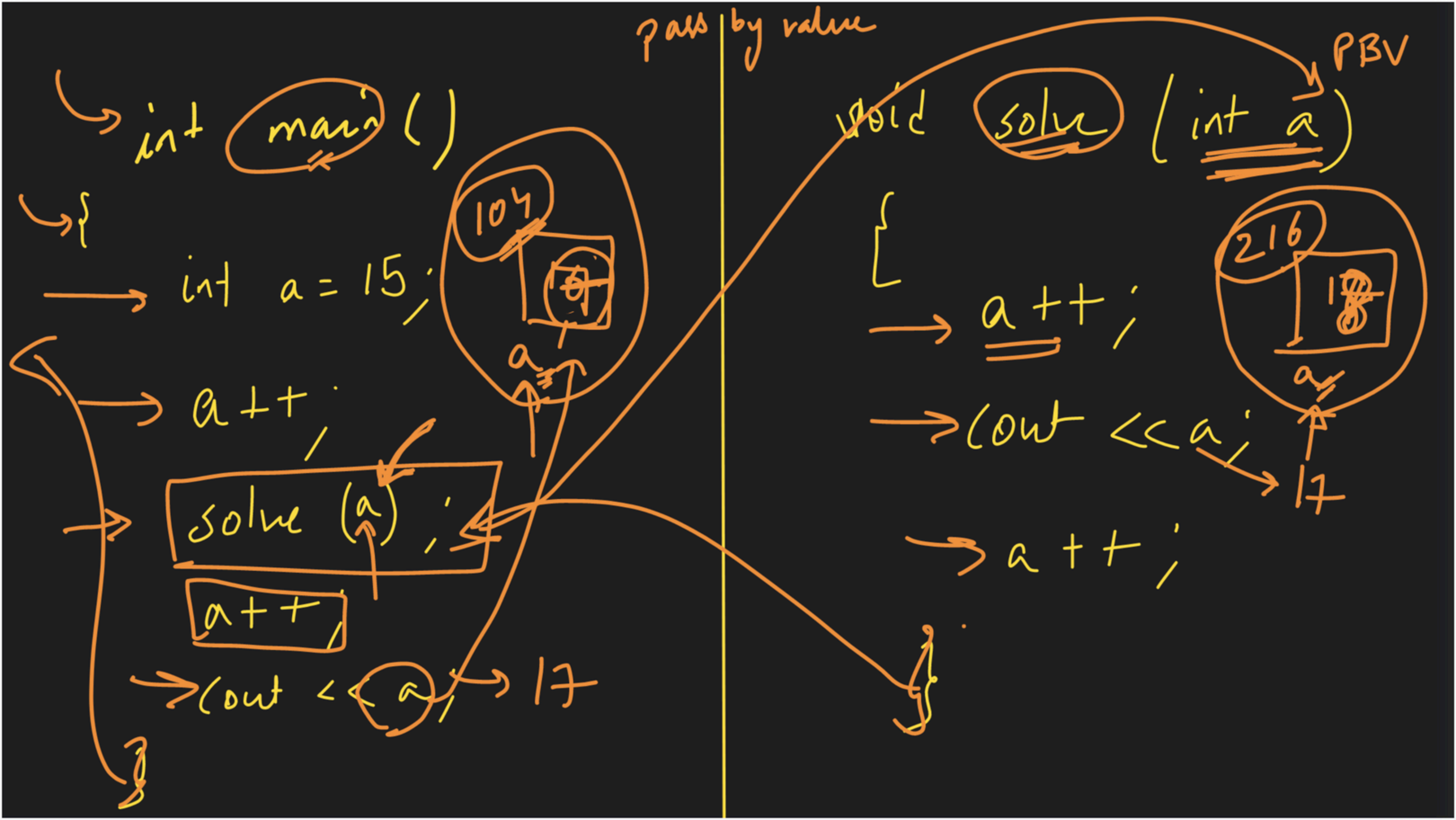

int → 4 byte → 32 bit
2^{31}
bit
31
sign
0 → $2^{32}-1$
-2^{31}
$2^{31}-1$

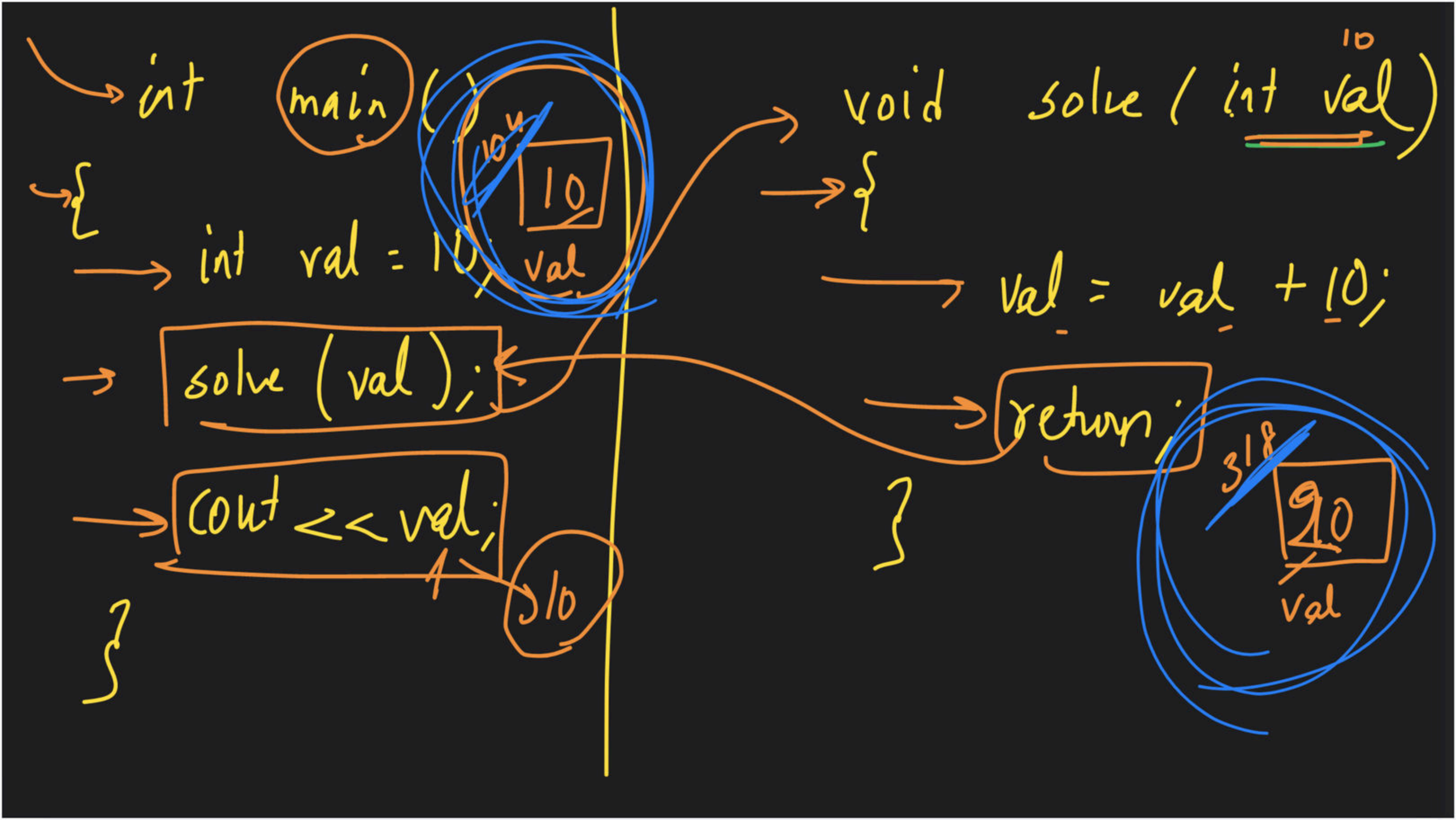

n=15
2→1
3→0
4
5
6
7
8
9
10
11
12
13
14
not prime


row → n → 4
obj
row 0 → 1 no
row 1 → 2 no
row 2 → 3 no
row 3 → 4 no
int number = 1
row 0 → 1
row 1 → 2 3
row 2 → 4 5 6
row 3 → 7 8 9 10
for (i=0 → <4)
{
for(c=0 → c<row+1)
{
// logic cout<<num
num++
}
}

```
int num =1
for ( r -> 0 ----> <n )
{
    for ( c=0 ----> <row+1 )
    {
        cout <<num;
        num++;
    }
    cout <<endl;
}
```

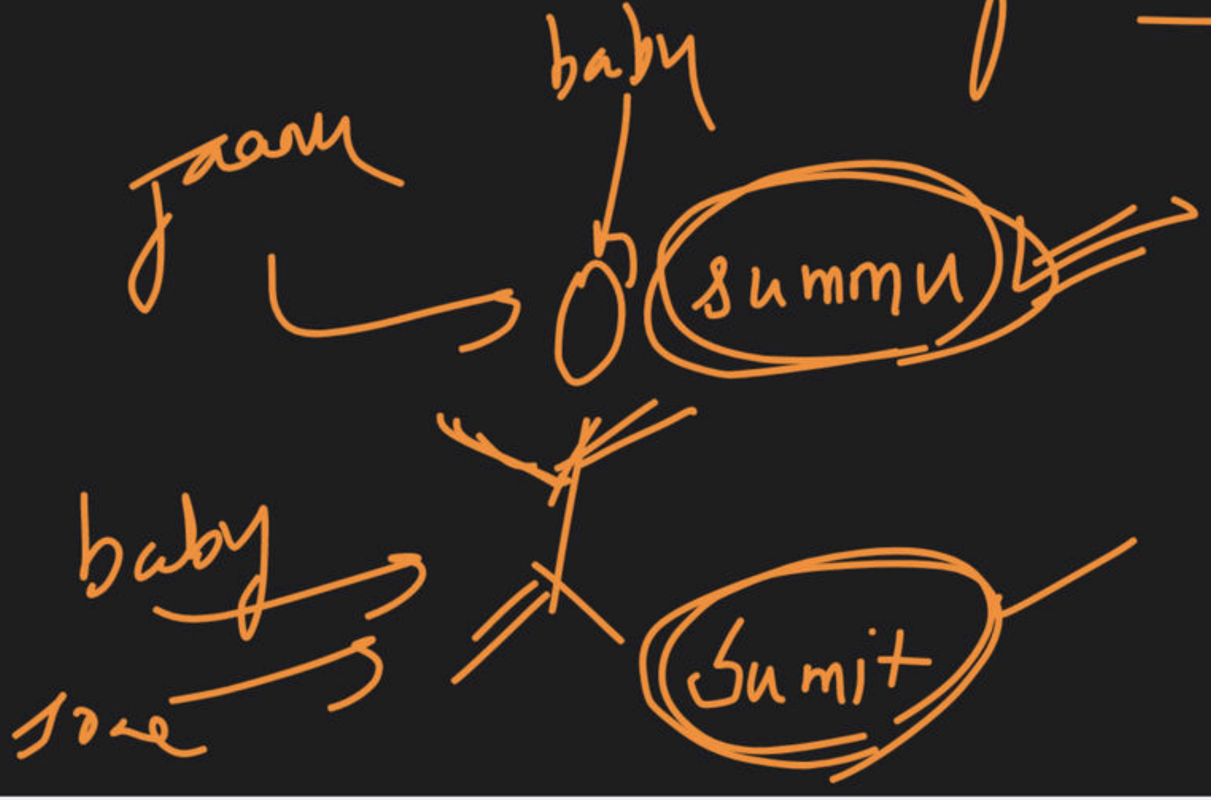

→ Pass by value
copy banti hai
Pass by reference
actual value me change hota hai


pass by value
int main()
{
int a = 15;
a++;
solve (a);
a++;
cout << a;
104
a
17
PBV
void solve (int a)
{
a++;
cout << a;
216
a
17
a++;


int main ()
{
int val = 10;
solve (val);
cout << val;
}
10
10
val
void solve (int val)
10
{
val = val + 10;
return;
}
318
20
val

Pass by reference

Reference Value



Pass by reference

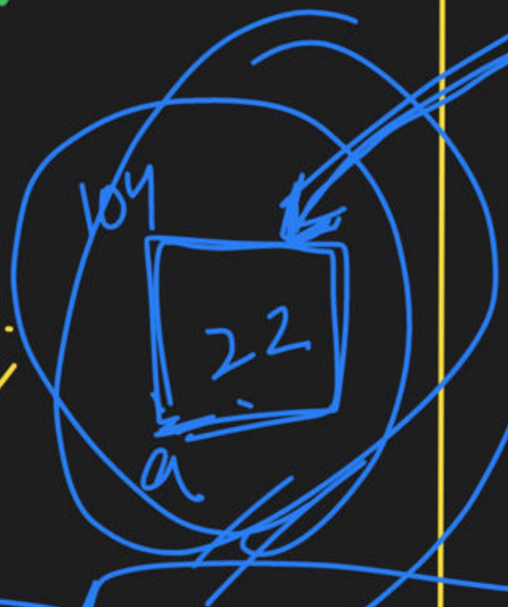
(12, 22, 21)

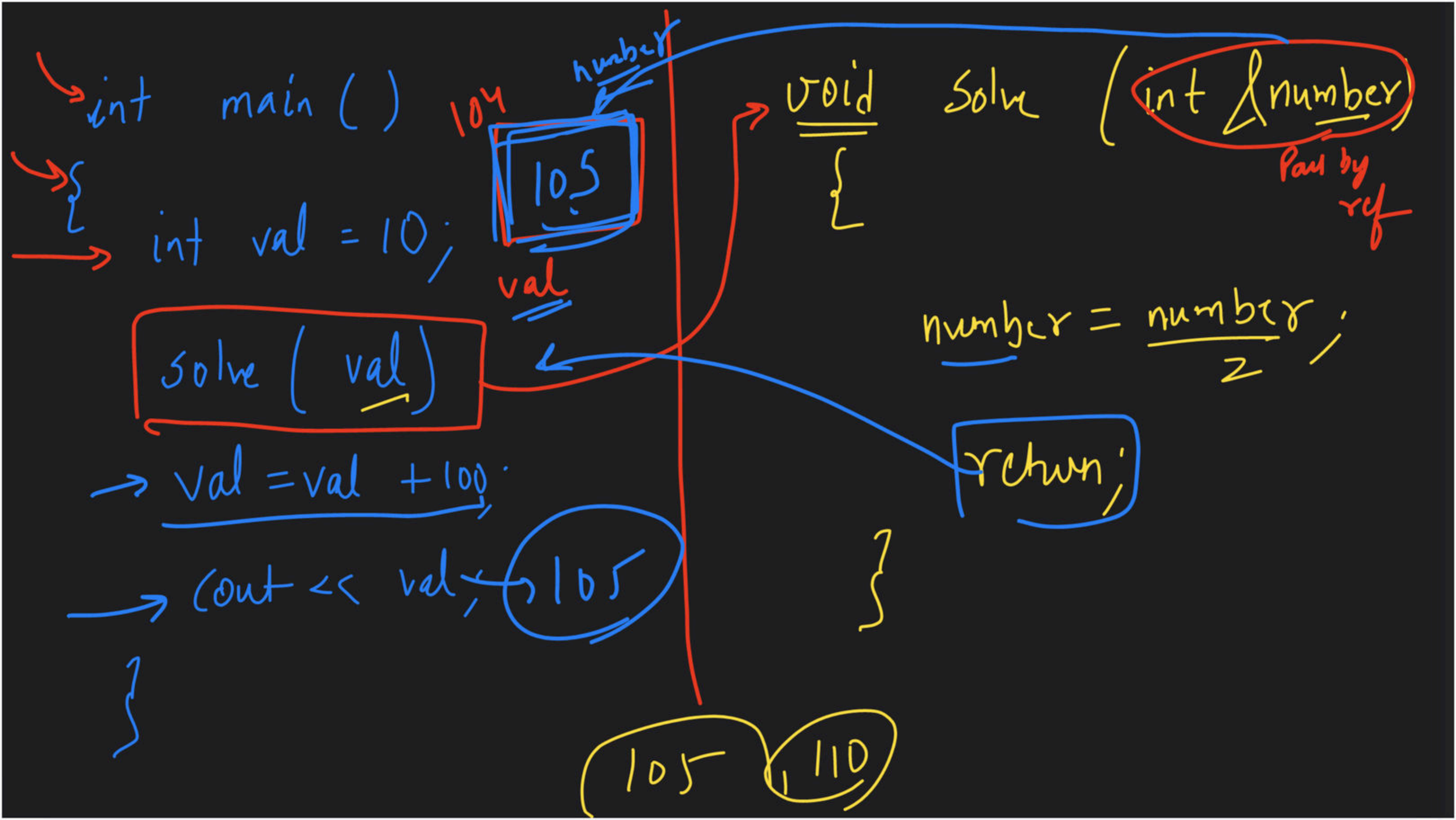
```
int main()
{
    int a = 10;
    a++;
    solve(a);
    a++;
    cout << a;
}
```

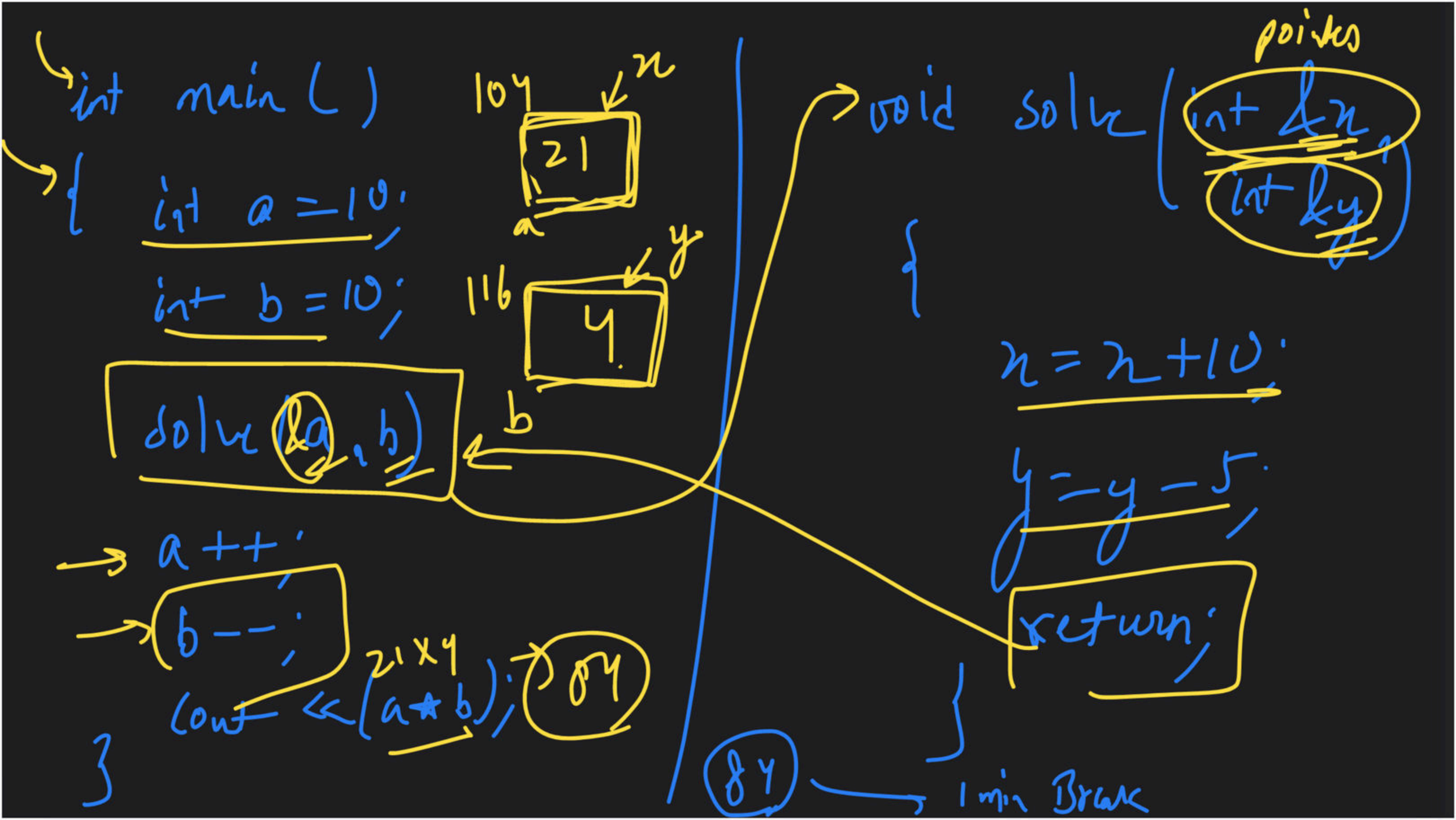
→ 12

```
void solve(int &a)
{
    a = a + 10;
    return;
}
```

Pass by reference




int main ( )
{
int val = 10;
solve (val)
val = val + 100;
cout << val;
}
104
105
val
number
void solve (int &number)
{
number = number / 2;
return;
}
105
110
Pass by ref


int main()
{
int a=10;
int b=10;
solve(a,b)
a++;
b--;
cout<<(a*b);
}
104
n
21
116
y
4
b
void solve(int &n, int &y)
{
n=n+10;
y=y-5;
return;
}
pointer
84
8 y
1 min Break

pass by reference

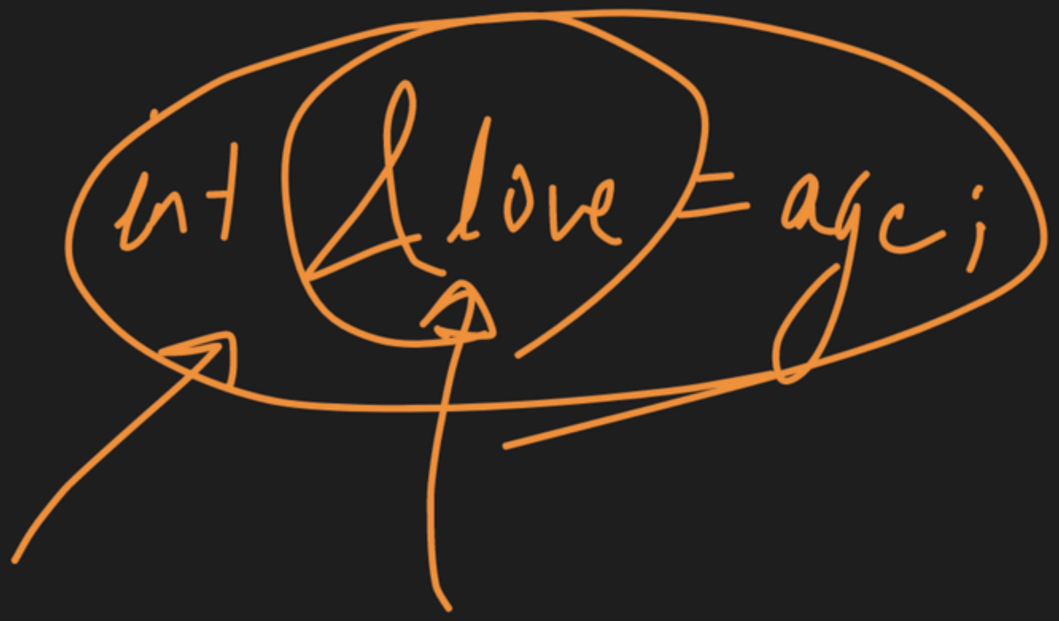
arg → var

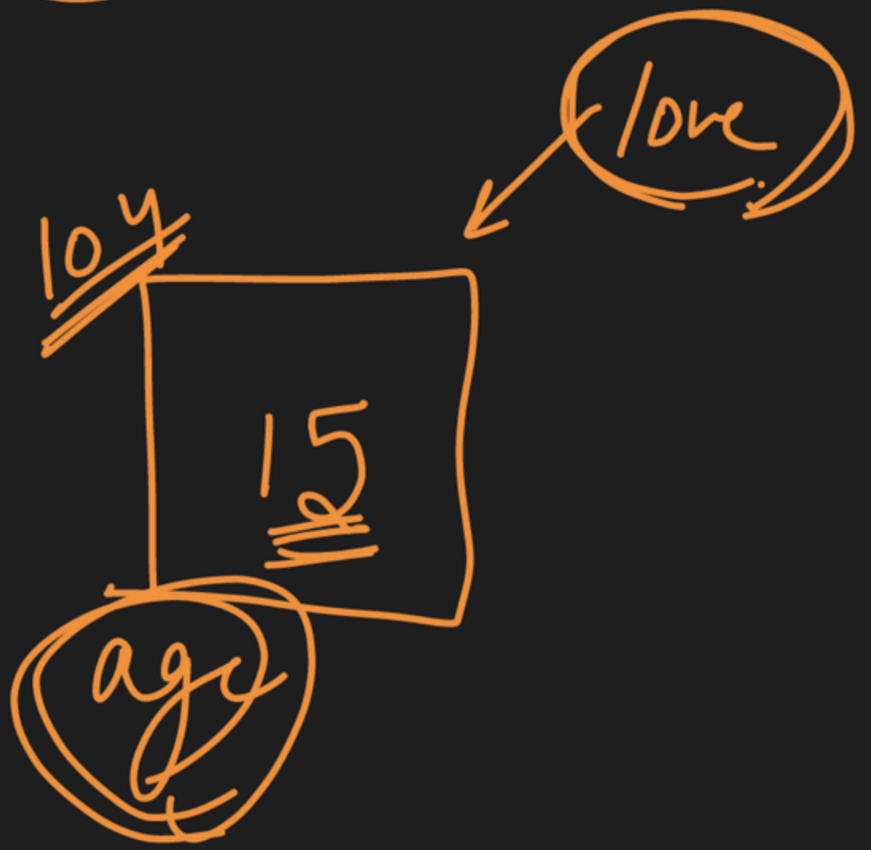
pass by value

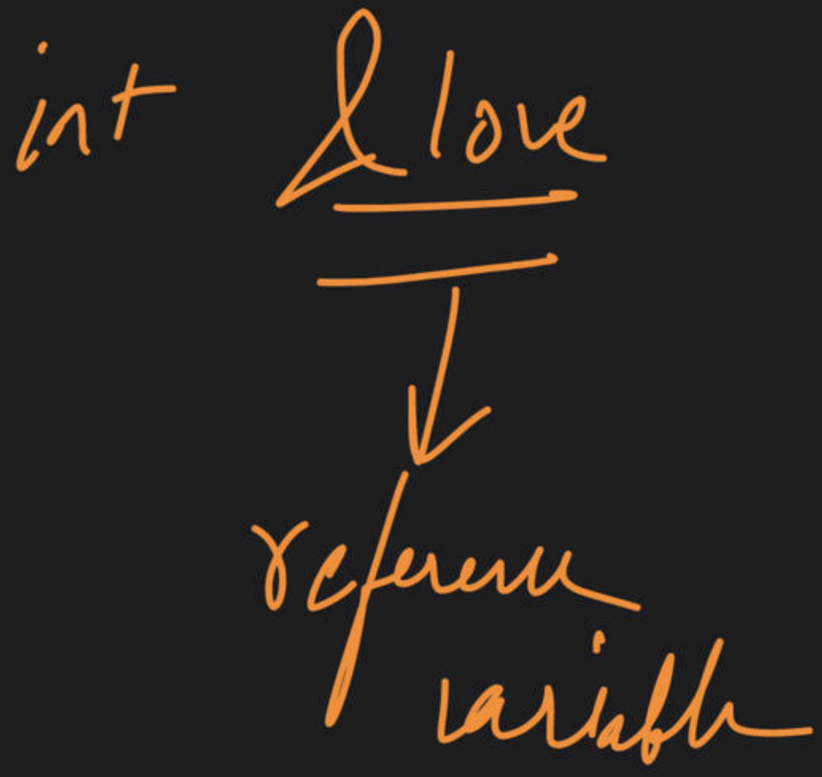
normally use

arg ka track
nahi hota

int age = 15







int val = 15 →

int demo = val

val = val + 10

cout << val → 21

val - -;

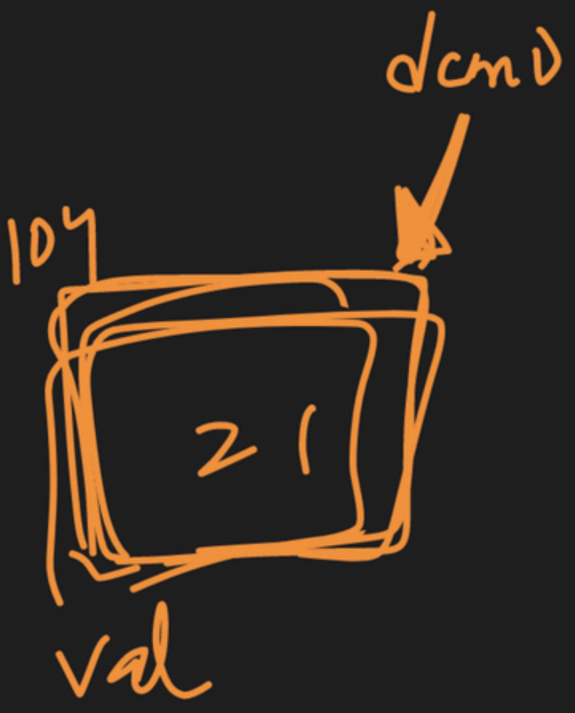
cout << demo → 14

demo - -

~~val~~ val - -

demo - -;

cout << demo → 21



int a = 10;

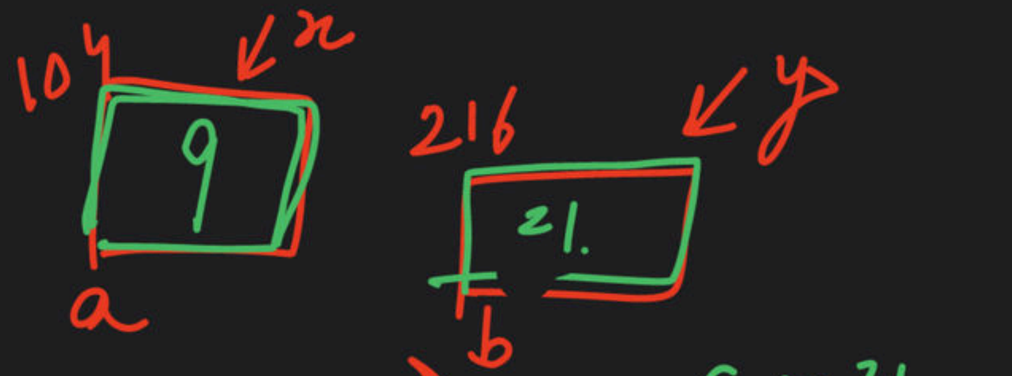
int b = 20;

int &x = a;

int &y = b;

x--;

b++;



cout << (a * b);

189