PYTHON PROGRAMMING
GATE DA/DSA

Agenda:

→ Operators (Assignment, Bitwise, Membership, Identity)

→ Operator Precedence ←

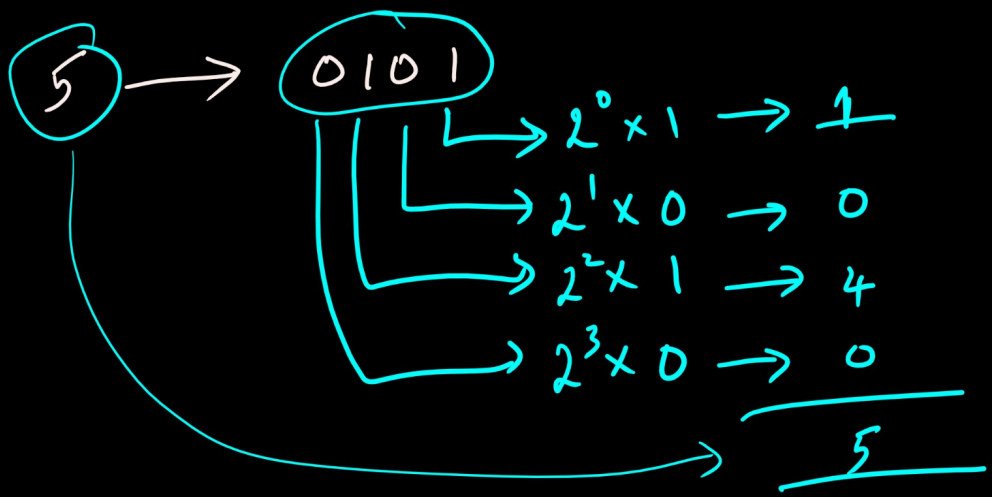
→ Unicode Representation ←

Operators
└ 7 operators

Operators: Assignment ✓

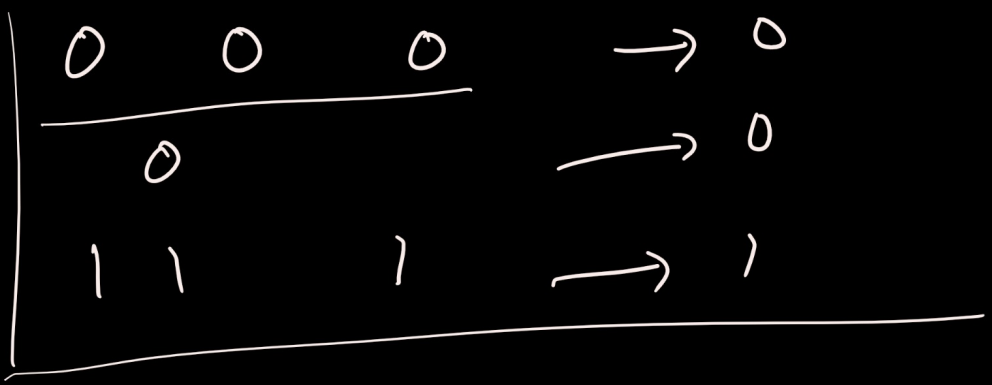
Bitwise

Bitwise Operator:



& - Bitwise AND

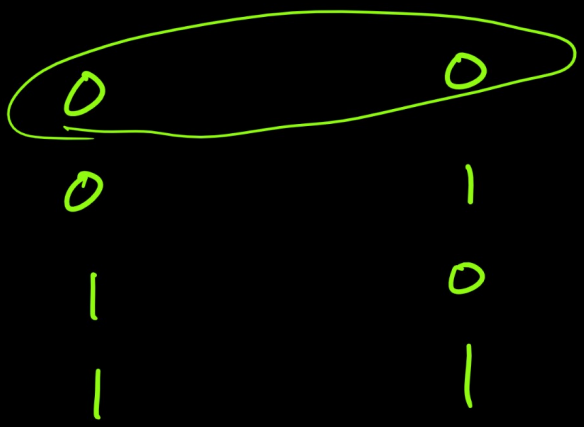
X	Y	Result (4) - Bitwise And.
0	0	0 & 0 = 0
0	1	0 & 1 = 0
1	0	1 & 0 = 0
1	1	1 & 1 = 1



| → Bitwise (OR)

X Y

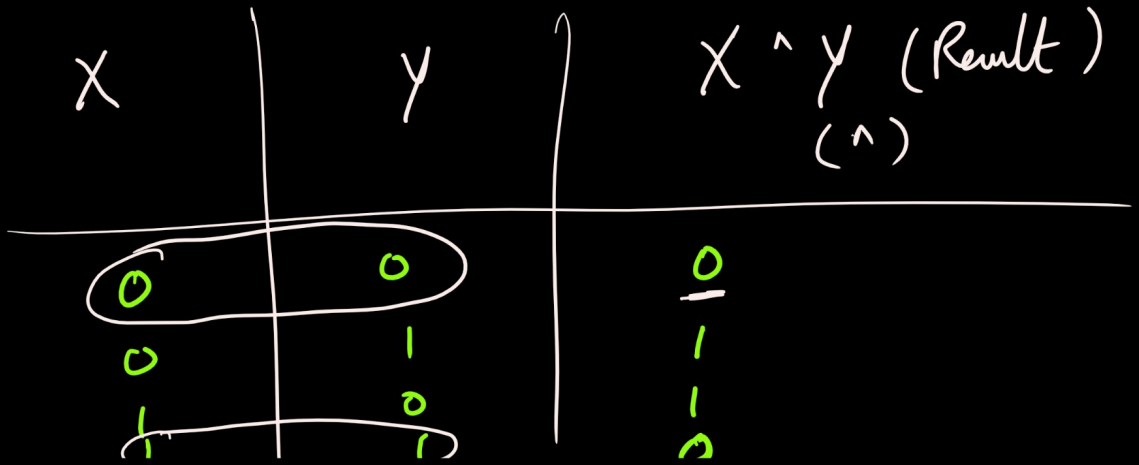
Result (Bitwise OR) (|)

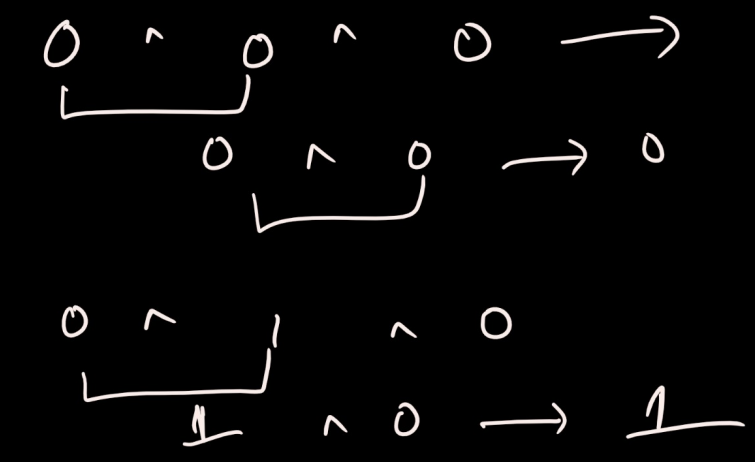


0|0 = 0
0|1 = 1
1|0 = 1
1|1 = 1

0 0 0 → 0

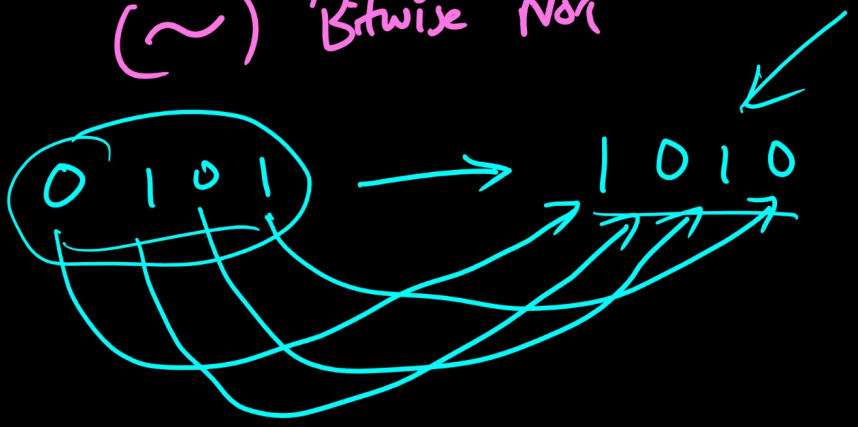
^ - (Bitwise XOR)





~13

(~) Bitwise NOT



14 → 1110

0×2^0

1×2^1

1×2^2

1×2^3

14

21 → 10101

0 1 1 1 0

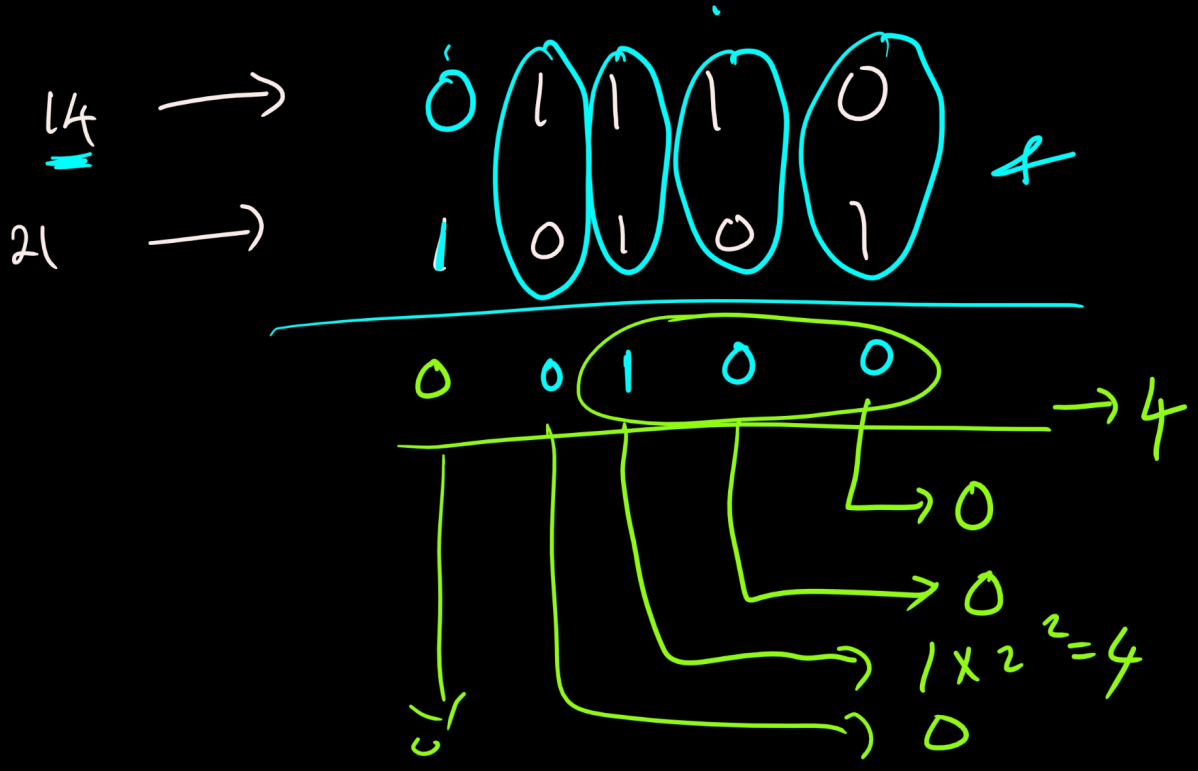
14 → 1 1 1 0

21 → 1 0 1 0 1

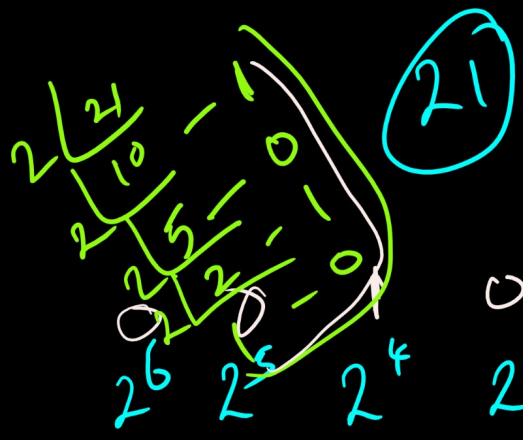
4 bit

0 0 0 0 | 1 1 1 0

0 0 0 1 | 0 1 0 1



0 →	0 0	7 → 1 1 1
1 →	0 1	8 → 1 0 0 0
2 →	1 0	
3 →	1 1	
4 →	1 0 0	
5 →	1 0 1	
6 →	1 1 0	



$\rightarrow 16+4+1$

2

23

1 0 1 0 1

$16+4+2+1$

$1+2+4+16+32+$

14 → 0 1 1 1 0 0 1 1

21 → 1 0 1 0 1

---

1 1 1 1 1

↳ $2^0 \times 1 = 1$

$2^1 \times 1 = 2$

$2^2 \times 1 = 4$

$2^3 \times 1 = 8$

$2^4 \times 1 = 16$

31

5 bits → 11111

$2^5 - 1 = 31$

7 → 1 1 1 1 1 1 1

$2^7 - 1 = 127$

14 → 0 1 1 1 0

21 → 1 0 1 0 1 (^)

1 1 0 1 1

16 8 0 2 1

27

& , | , ^

~ → 13

13 →

~13 →

8 4 2 1

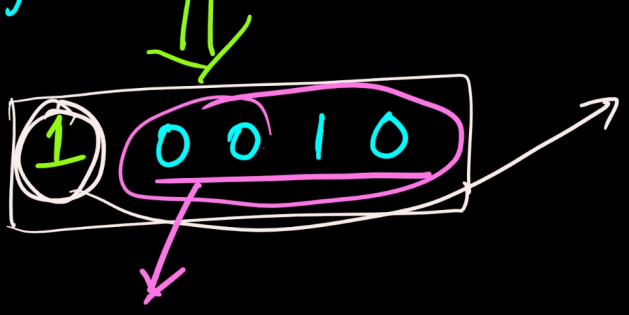
13 → 1101

~13 → 0010 ← (2)

Sign bit → —

13 → 0 1101

sign bit



if sign bit is 1, it indicates its a negative number.

14 → 0 1110

5 → 101

2 → 10

⊖14 → 1110

0 0 1 1 → 1's Complement → 1 1 0 0

0 1 → 2's Complement → 1 1

0 → 0

1 0

1110

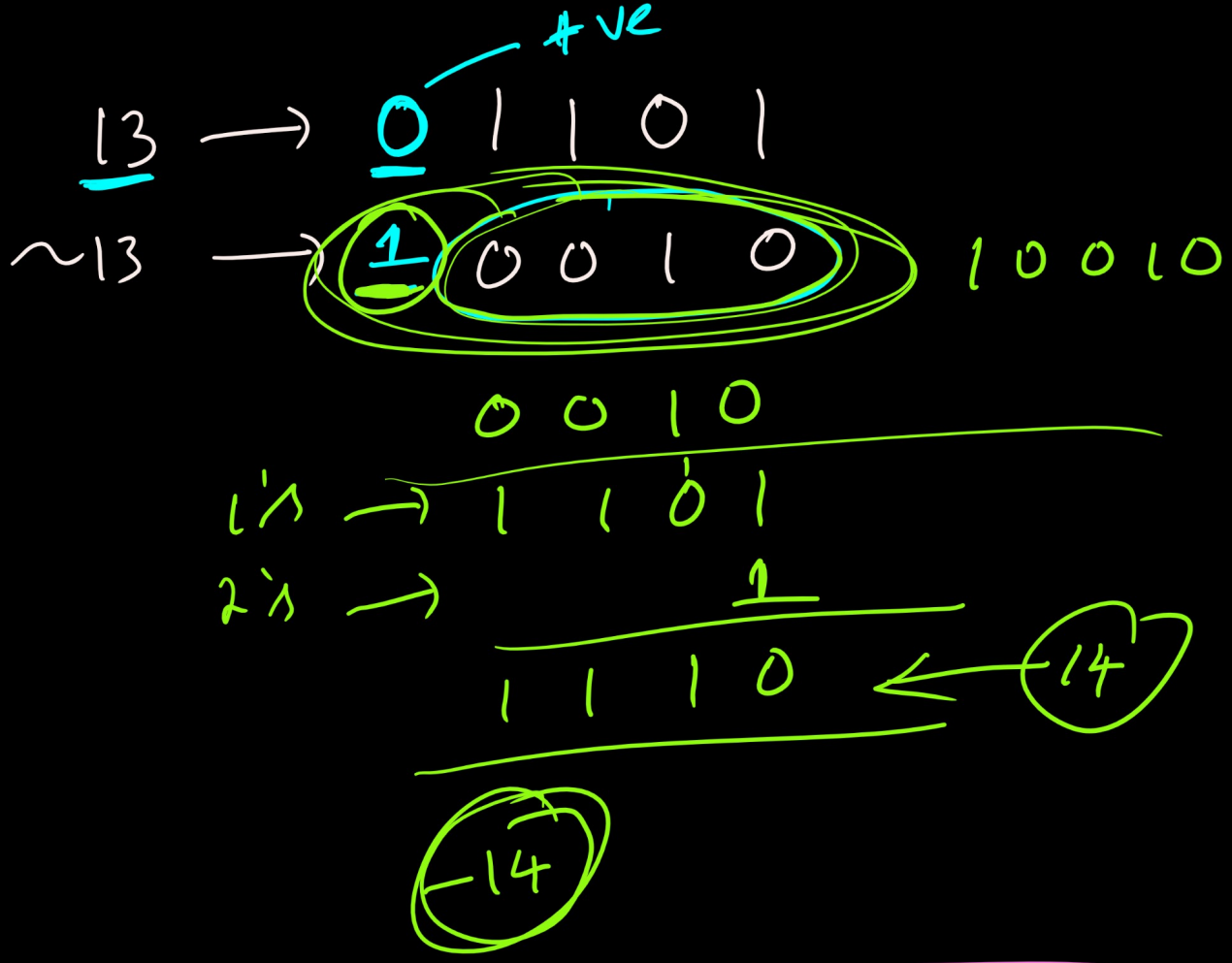
0001

1

0010

10010 → -14 Computer

Negative numbers stored in 2's Complement format.



$$\sim n \Rightarrow -(n+1)$$

$\sim 13 \Rightarrow -(13+1)$

$= -14$

$\sim 105 \Rightarrow -(105+1)$

$= -106$

$$\sim -14 = -(-14+1)$$

$$= -(-13)$$

$$= 13$$

~ -14

	0	1	1	1	0
1's:	1	0	0	0	1
2's:					1

negate this → $-14 \rightarrow$ | 1 0 0 1 0 |

0	1	1	0	1

+ve 13

$\sim -14 \rightarrow 13$

16	8	4	2	1
1	0	0	1	0

$-14 \rightarrow$ sign bit

$-16 + 1 = -14$

$\sim(n) \rightarrow -(n+1)$

+ve

-ve

$101 \rightarrow -(101+1) = -102$

$-101 \rightarrow -(-101+1) = 100$

$14 \rightarrow \underline{0}\ 1110$

sign bit

0001

1

0010

$-14 \quad \underline{1}\ 0010$

sign bit

$10 \quad \underline{0}\ 1010$

0101

1

0110

$-10 \quad \underline{1}\ 0110$

(& | ^ ~)

Left Shift & Right Shift

Left Shift :- 10 << 2

Right Shift :- 10 >> 2

10

2 → shift 2 bits to left

10 → 0 0 0 0 1 0 1 0

0 0 0 0 1 0 1 0

32 16 8 4 2 1

0 0 1 0 1 0 0 0

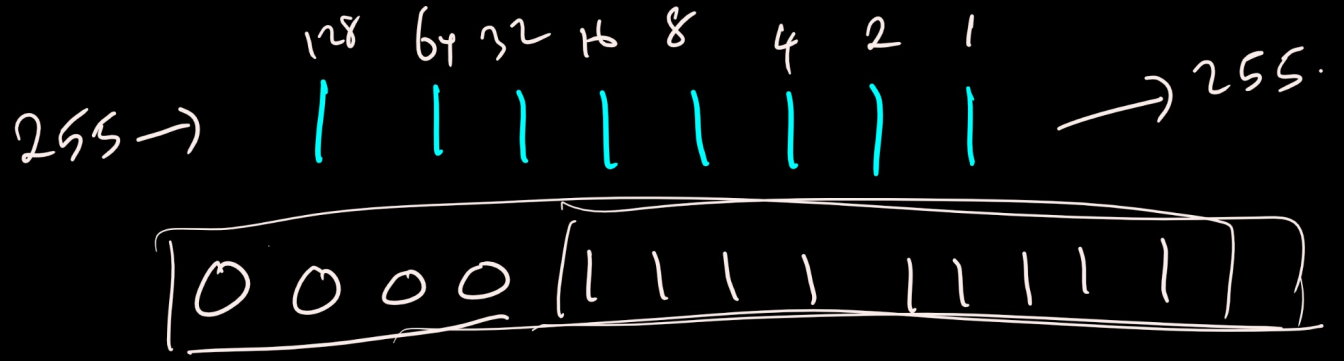
32 0 8 0 0 0

32 + 8 = 40

14 → 0 0 0 0 1 1 1 0

0 0

0 0 1 1 1 0 0 0



0 0 1 1 1 1 1 1 1 1 0 0 → 1020

512 256 128 64 32 16 8 4 2 1

$n >> 1$

$\hookrightarrow n \times 2$

$n >>> 2$

$\hookrightarrow n \times 4$

$n << k$

$n * 2^k$

Left Shift (<<)

$n << K$

Result: $n * 2^K$

Right Shift (>>)

$n >> K$

$n // 2^K$

$n =$ 20

16 8 4 2 1

20 → 0 0 0 1 0 1 0 0

0 0 0 1 0 1 0 0

128 64 32 16 8 4 2 1

Left shift by 1: 0 0 1 0 1 0 0 0 → 40

Left shift by 2: 0 1 0 1 0 0 0 0 → 80

160

20 → 0 0 0 1 0 1 0 0

20

128 64 32 16 8 4 2 1

Right shift by 1: 0 0 0 0 1 0 1 0 → 10

8 4 2 1

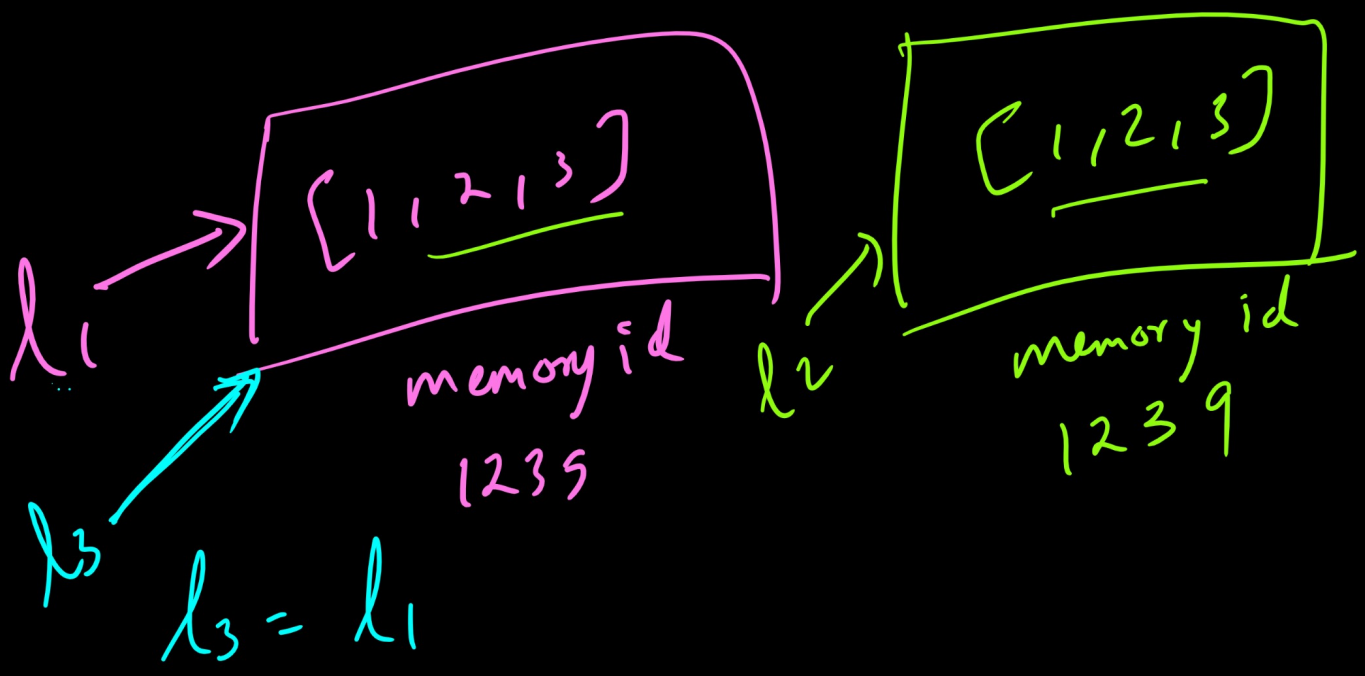
Right shift by 2: 0 0 0 0 0 1 0 1 → 5

101

10

1

[1, 2, 3]



Operator Precedence :

2. Arithmetic Operator

$5 + 12 * 7 / (2 - 10)$

$17 * 7$

$117 / 2$ $\rightarrow 58.5 - 10 \rightarrow 48.5$

$7 / -8$

$-1.2 - 0.8 \ldots 5$

Order of Priority :

1. Parenthesis / Brackets $\rightarrow$ ( ) , [ ] , { }

2. Exponents / Order $\rightarrow$ ** Right to Left

3. Division, Multiplication $\rightarrow$ *, /, //, %. Left to Right

4. Addition, Subtraction → (+, −) Left to Right

$5 + 9 - 2$

$5 + (12 * 7 / 2) - 10$

1. *, /

i) 84

ii) $84/2 \rightarrow 42.0$

$5 + 42.0 - 10$

iii) $47.0 - 10 = 37.0$

2**3**2 → $8^{\wedge} = 64$ ✗

2**(3**2) → $2^9 = 512$

Right to left

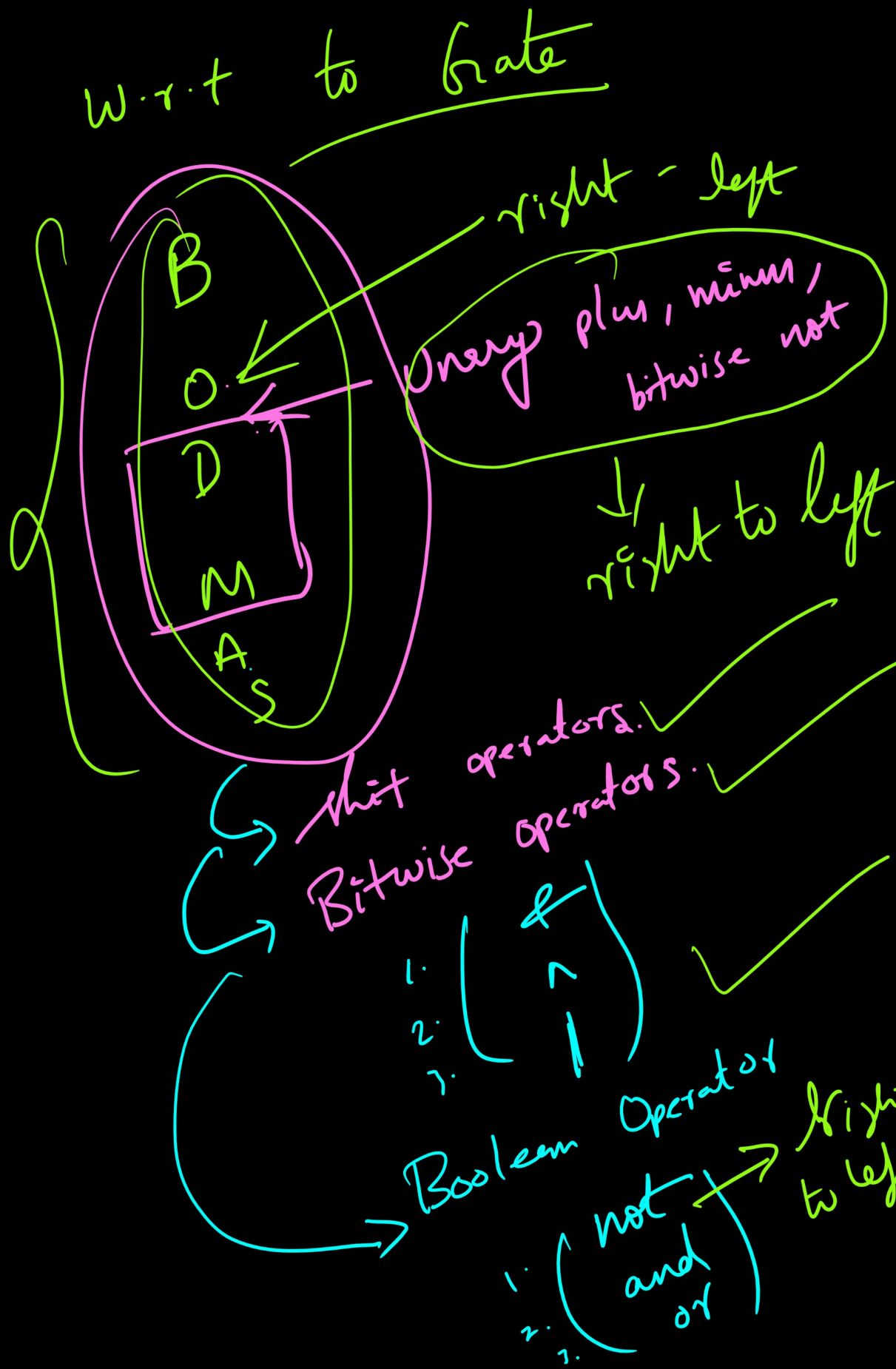
(3**2) → $3^2 = 9$

Operator Precedence Table :-

Precedence	Operator	Associativity
1	Parenthesis	Left to Right
2	Exponentiation (**)	Right to Left
3	Positive negative +x, -x, (~x) → bitwise not	Right to Left
4	*, /, //, % mul, div, floor div, modulus	Left to Right
5	+, - Add, sub	Left to Right
6	<<, >> shift	Left to Right

7.	AND `&` — bitwise AND	Left to Right
8	XOR `^` — bitwise XOR	Left to Right
9	OR `|` — bitwise 'OR'	Left to Right
10	in, not in, is, is not, <, <=, >=, !=, ==	Left to Right
11	not X — Boolean not	Right to Left
12	and — Boolean AND.	Left to Right

13	or Boolean or.	Left to Right



Unicode & Tomorrow

Colab Class - 5:
https://colab.research.google.com/drive/1Ok6U8iQBNJG6Y4Xm1-IkFotd4Lcsb2B5?usp=sharing