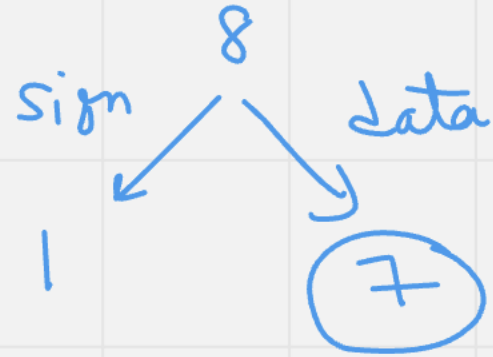
Circular Property

1 byte = 8 bit

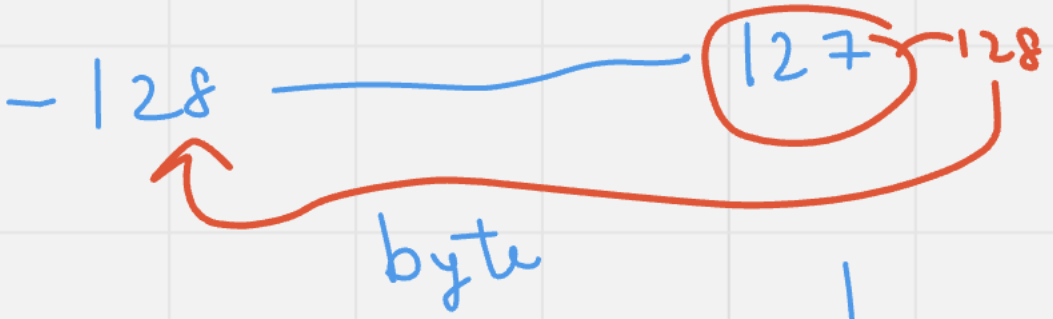


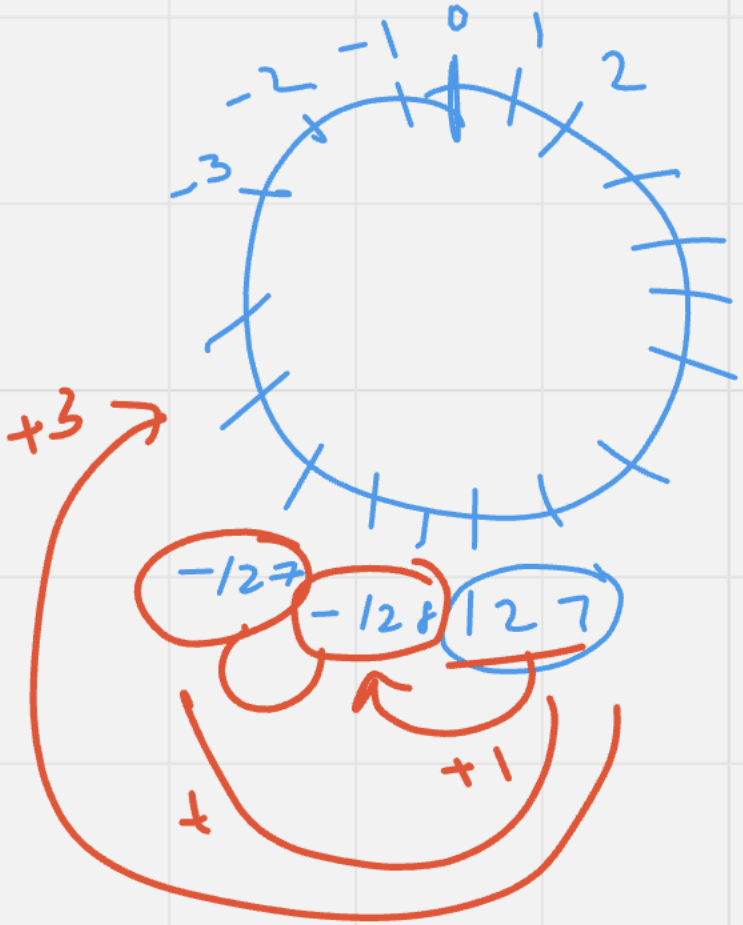
-2^7

min

$2^7 - 1$

max





127

129

byte b = 127

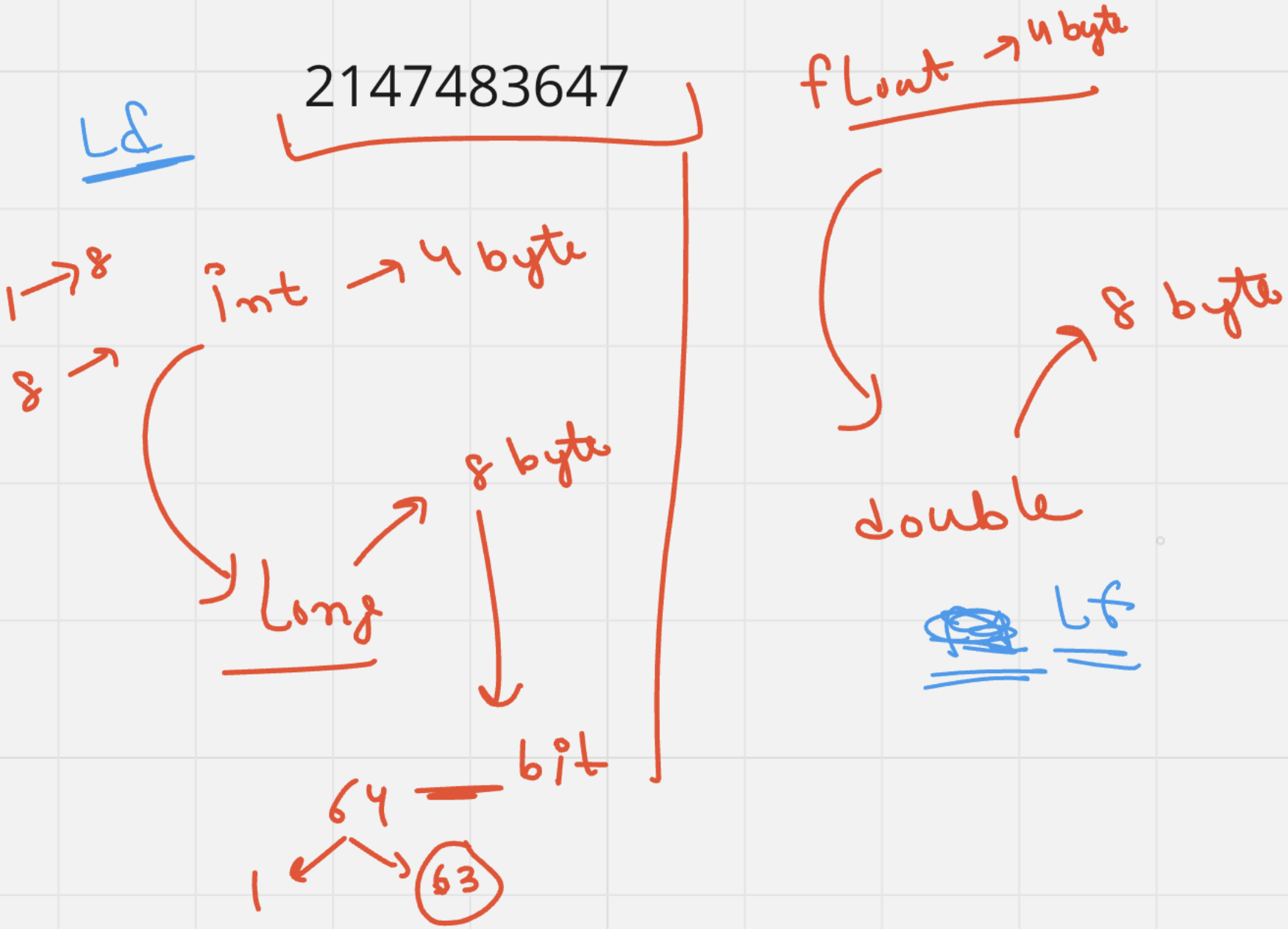
b = 128

Print(b) ≡ 127

-128

130

-128 ~


2147483647
Ld
1 → 8
8 →
int → 4 byte
Long
8 byte
64 bit
1
63
float → 4 byte
8 byte
double
Lf

```
char c = 'a';
    //printf("%c", c);

    printf("%d", c);
```

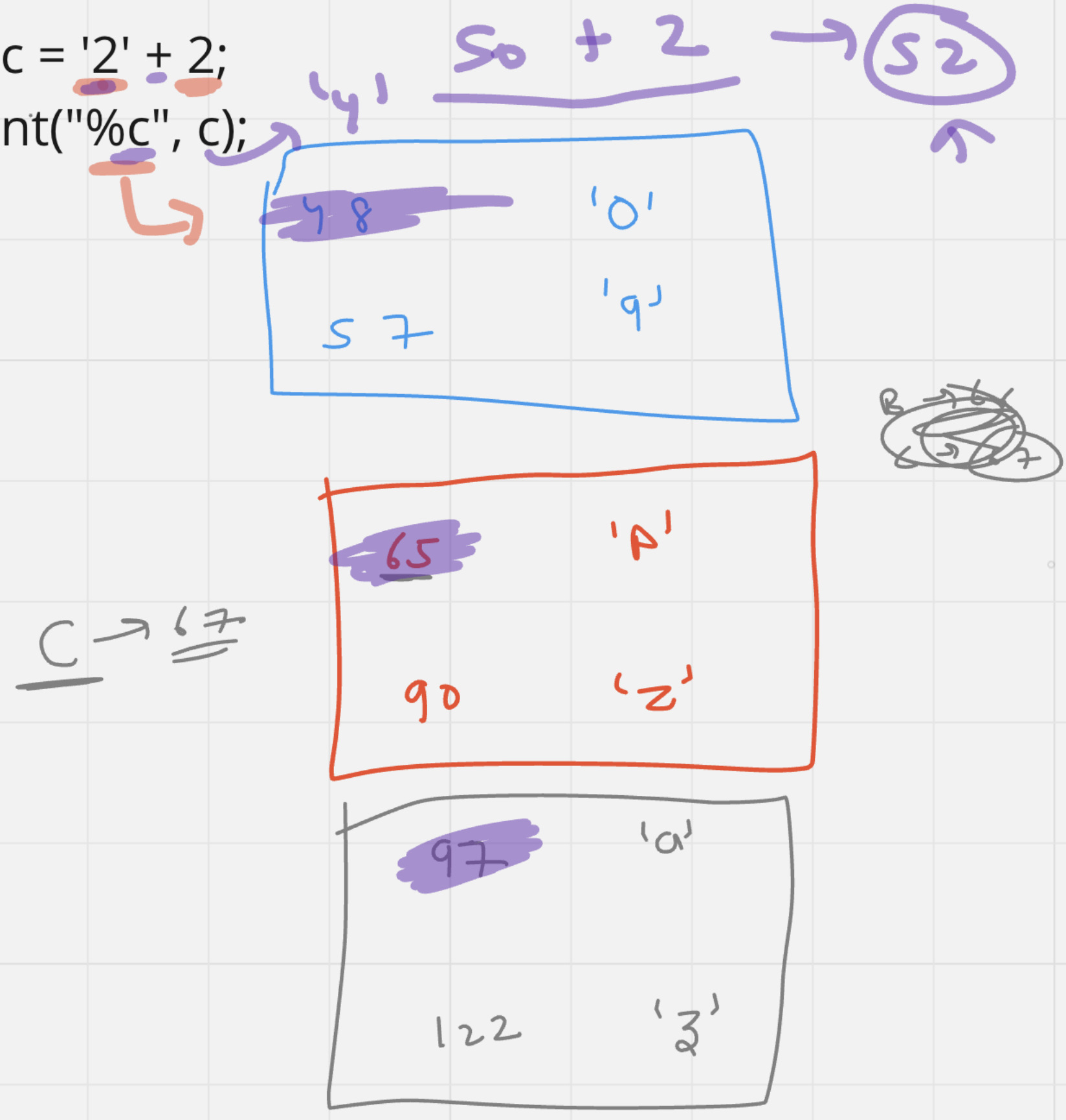
↳ 97

Ascii table

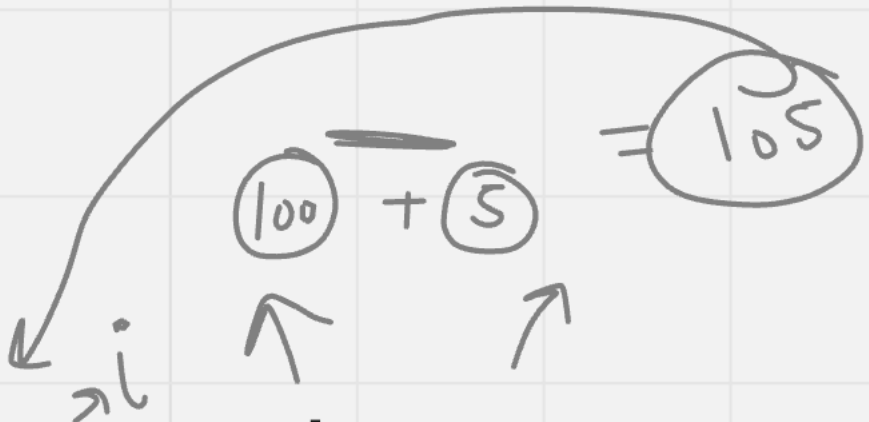
ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[END OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

```
char c = '2' + 2;
print("%c", c);
```

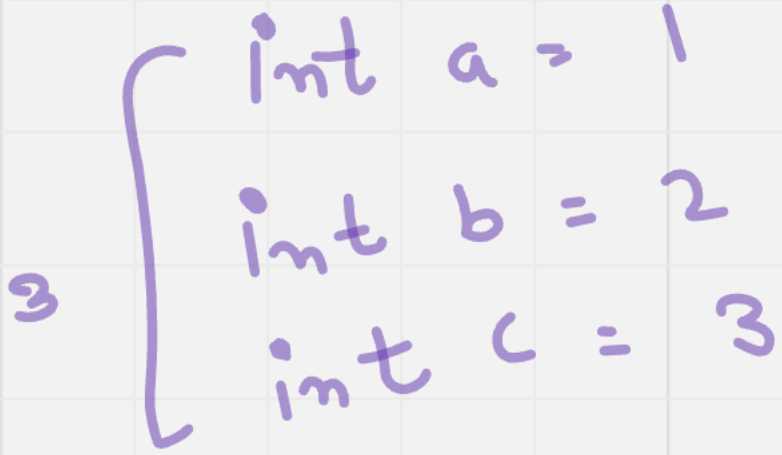


num



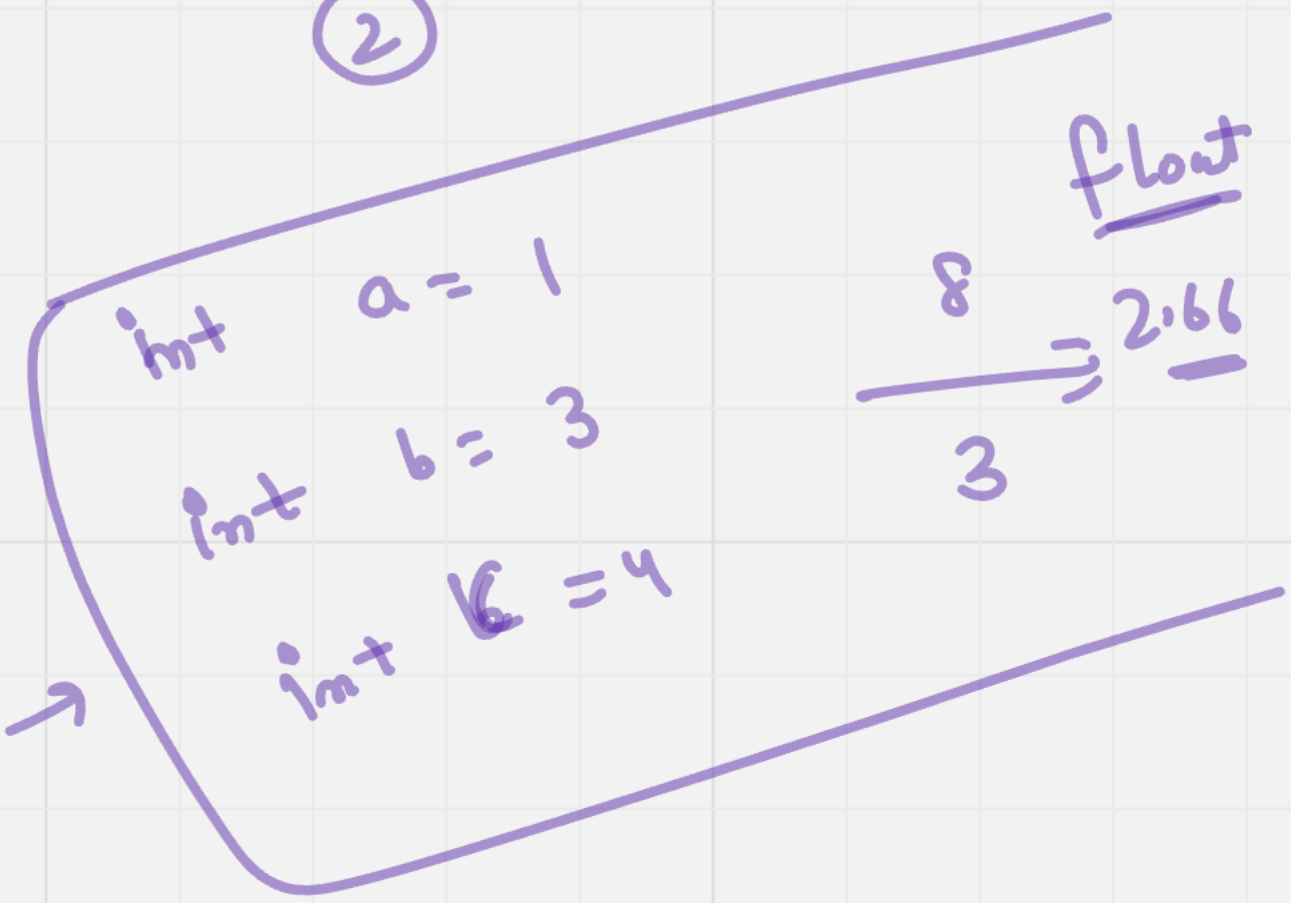

```
char c = 'd' + 5;
    //printf("%c", c);
    //printf("%d", c);
    printf("%c", c);
```

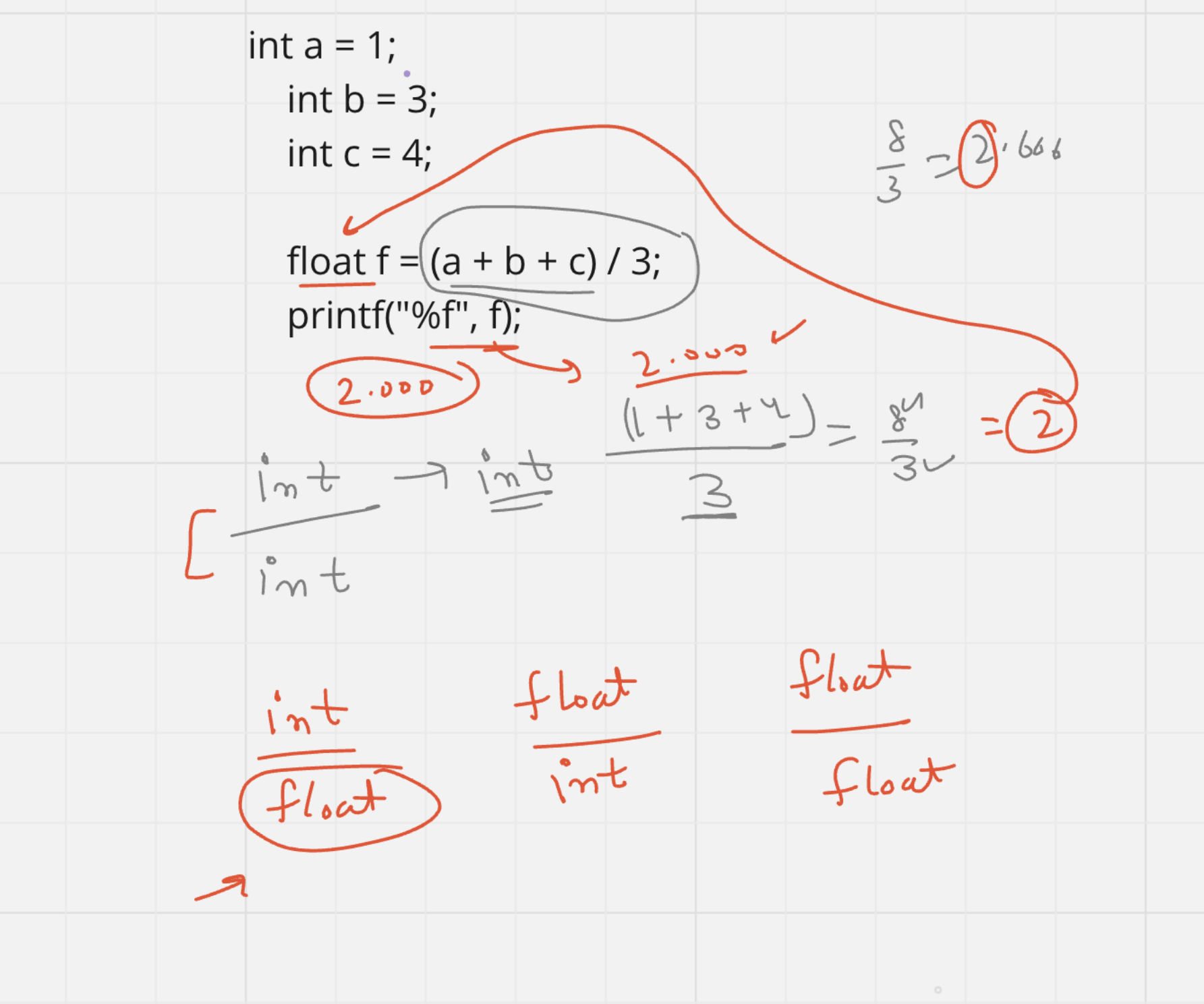
i



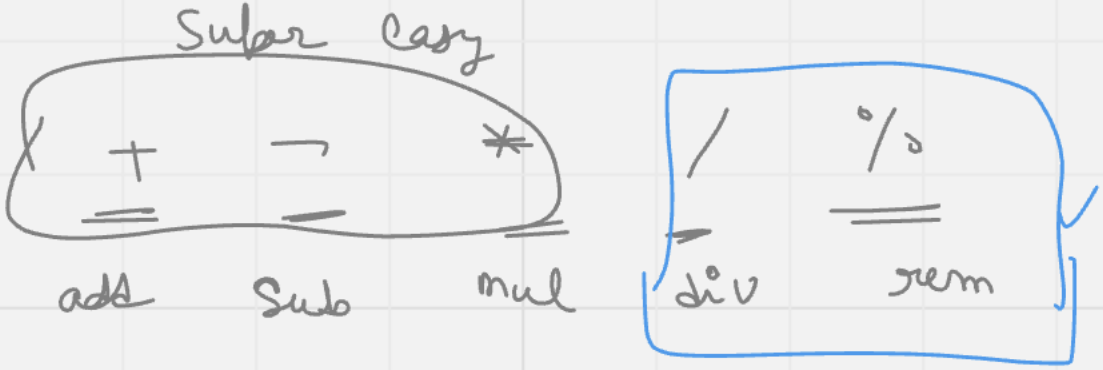
$$Avg \equiv \frac{(a+b+c)}{3} \rightarrow \frac{6}{3} \rightarrow 2$$

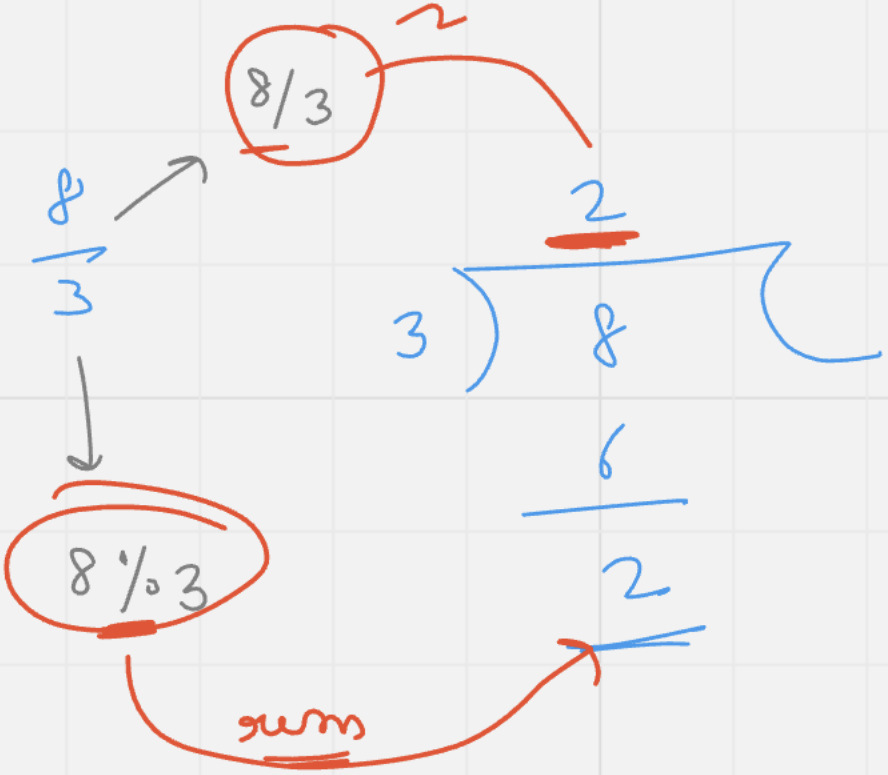
(2)




int a = 1;
int b = 3;
int c = 4;
float f = (a + b + c) / 3;
printf("%f", f);
2.000
2.000
(1 + 3 + 4) / 3 = 8/3 = 2
8/3 = 2.666
int / int → int
int
float
float / int
float
float / float

① Airthmatic operators





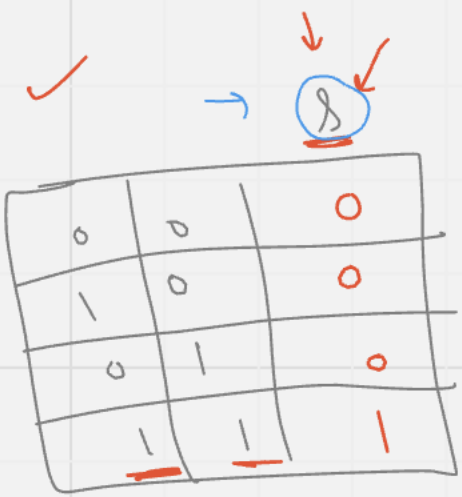
Bits manuf.

Bitwise operator

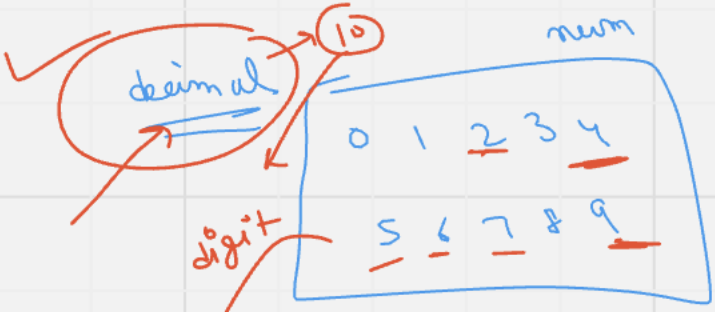
and (&) or (|) not (~) xor (^)

left shift <<

Right shift >>



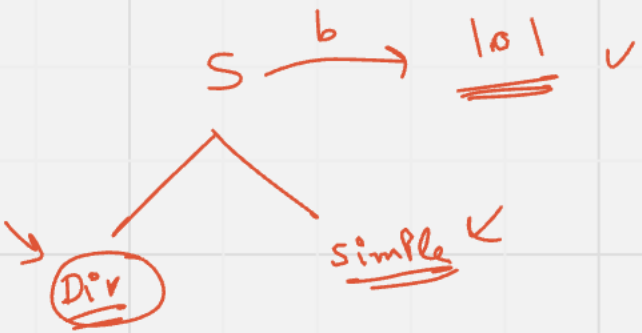
any one bit is zero then output is also 0

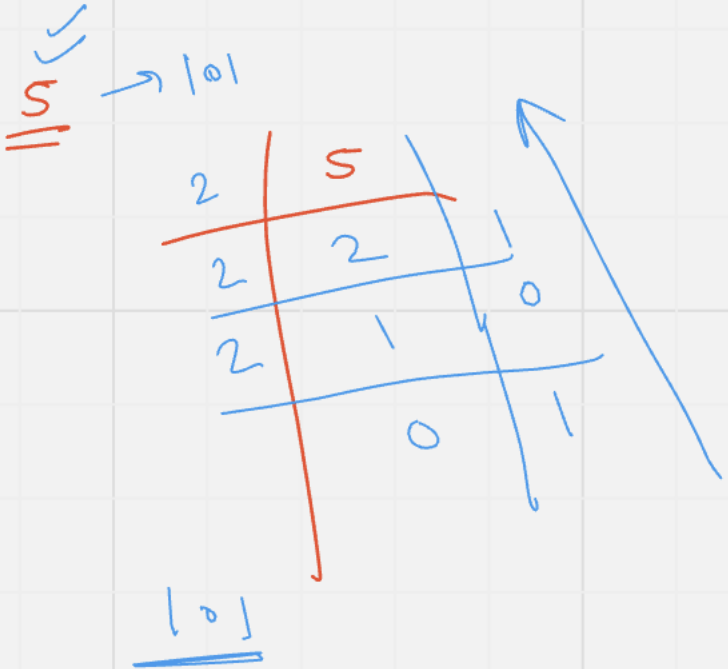


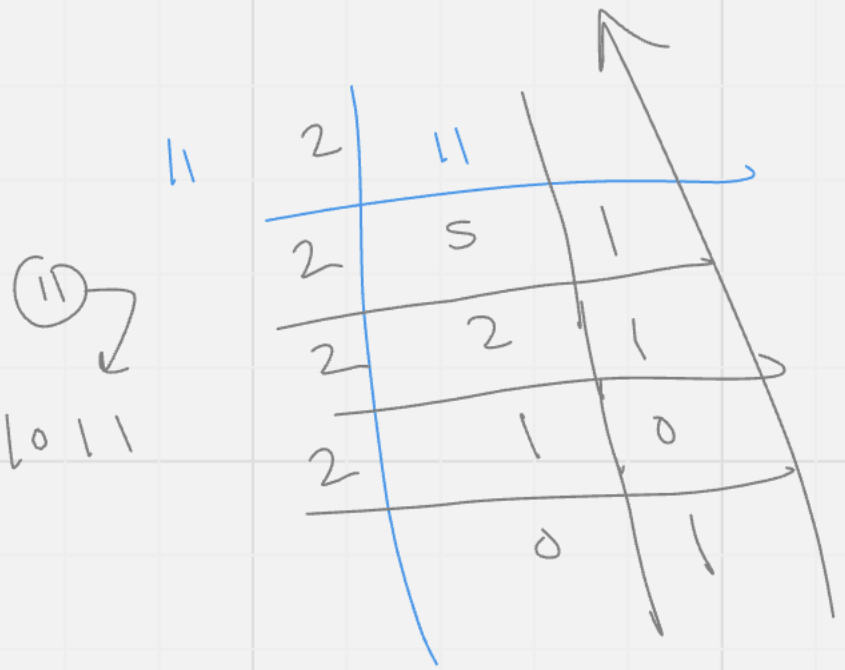
49 56

num 27

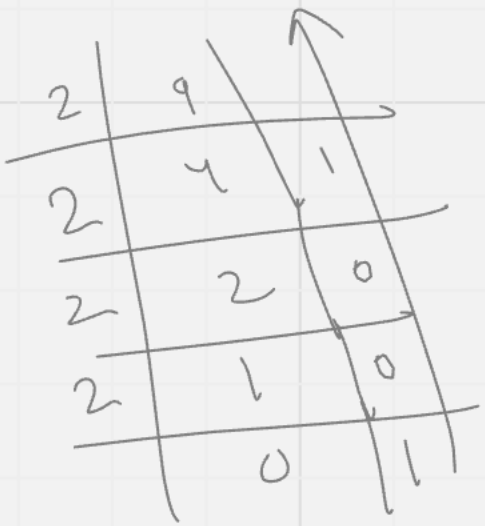
binary { 0 1 }


S
b
101
Dir
simple


S
101
2
S
2
2
1
0
2
1
0
1
101


11
2 11
2 5 1
2 2 1
2 1 0
2
0 1
11
1011


binary
9
1001


2 9
2 4 1
2 2 0
2 1 0
2
0 1

2nd simple

2^6 2^5 2^4 2^3 2^2 2^1 2^0

64 32 16 8 4 2 1

1 1 0

13 → 1101

32 16 8 4 2 1

1 1 0 1

17 → 10001 ✓

64 32 16 8 4 2 1

1 0 0 0 1

10 → 1010 ✓

10 → 1010 ✓

8	4	2	1
1	0	1	0

a → 10
b → 5

1010
0101
(8) 0000

ab

8	4	2	1
0	0	0	0

→ (0)

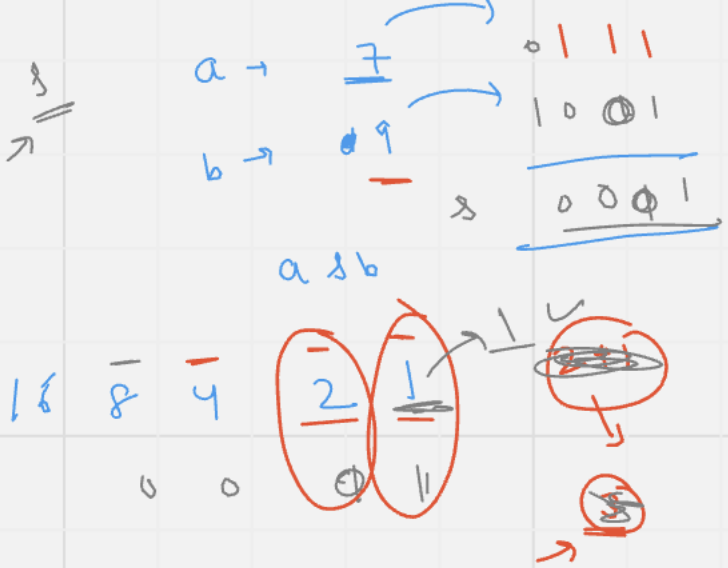
a → 6
b → 11

0110
1011
(5) 0010

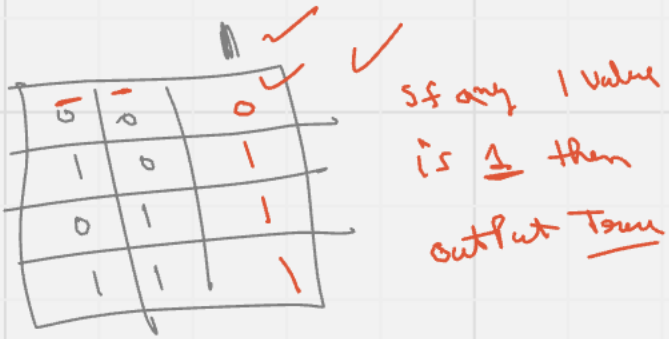
16	8	4	2	1
	1	0	1	1

16	8	4	2	1
			1	0

→ 2



Or (|)

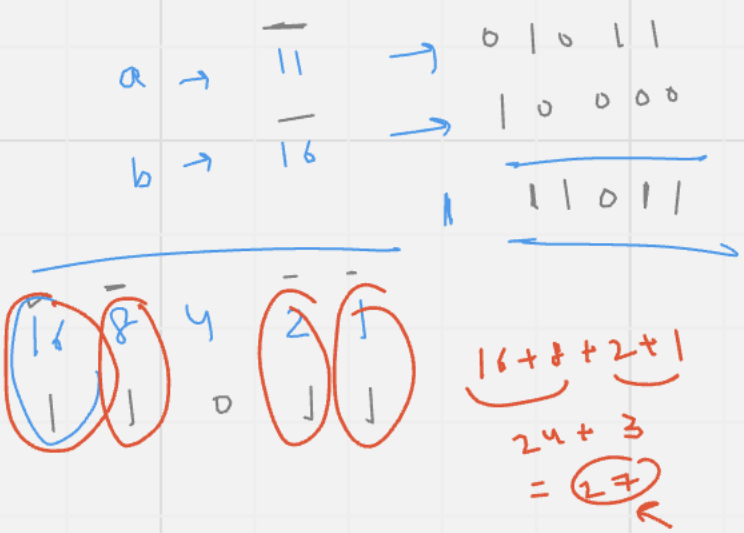


a = 6 → 1 1 0

b = 5 → 1 0 1

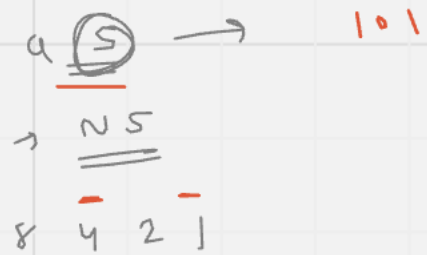
1 1 1

4+2+1 = 7

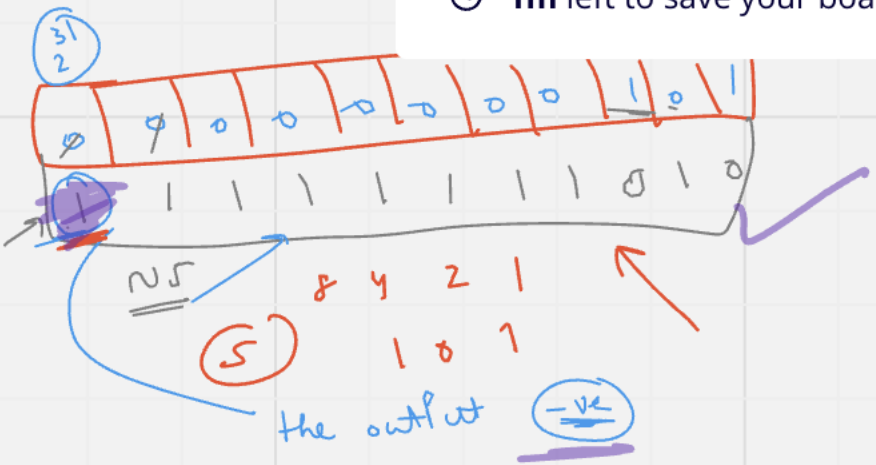


not (~)

	~
0	1
1	0

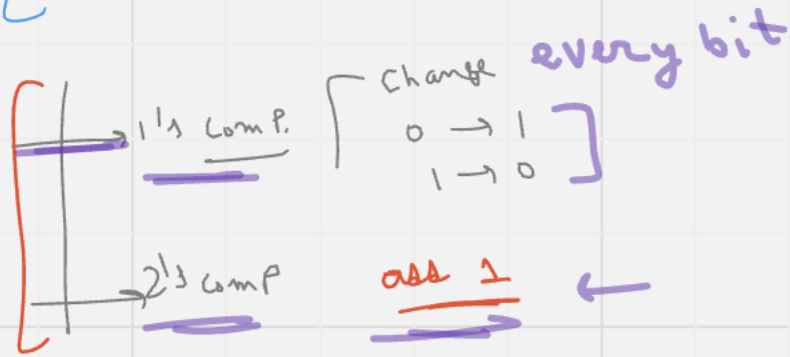


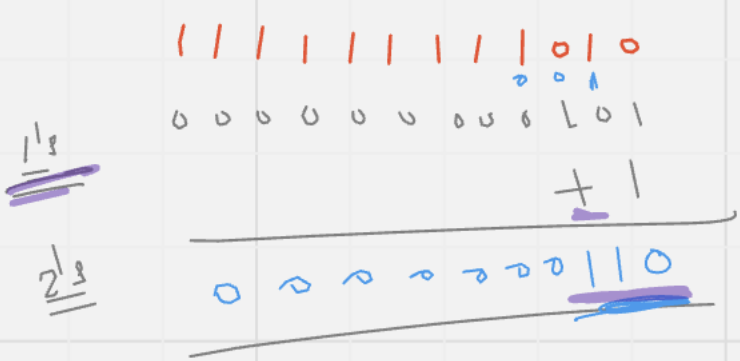
int → bit = 32



modulus of the num

2's compliment





4 + 2 = 6

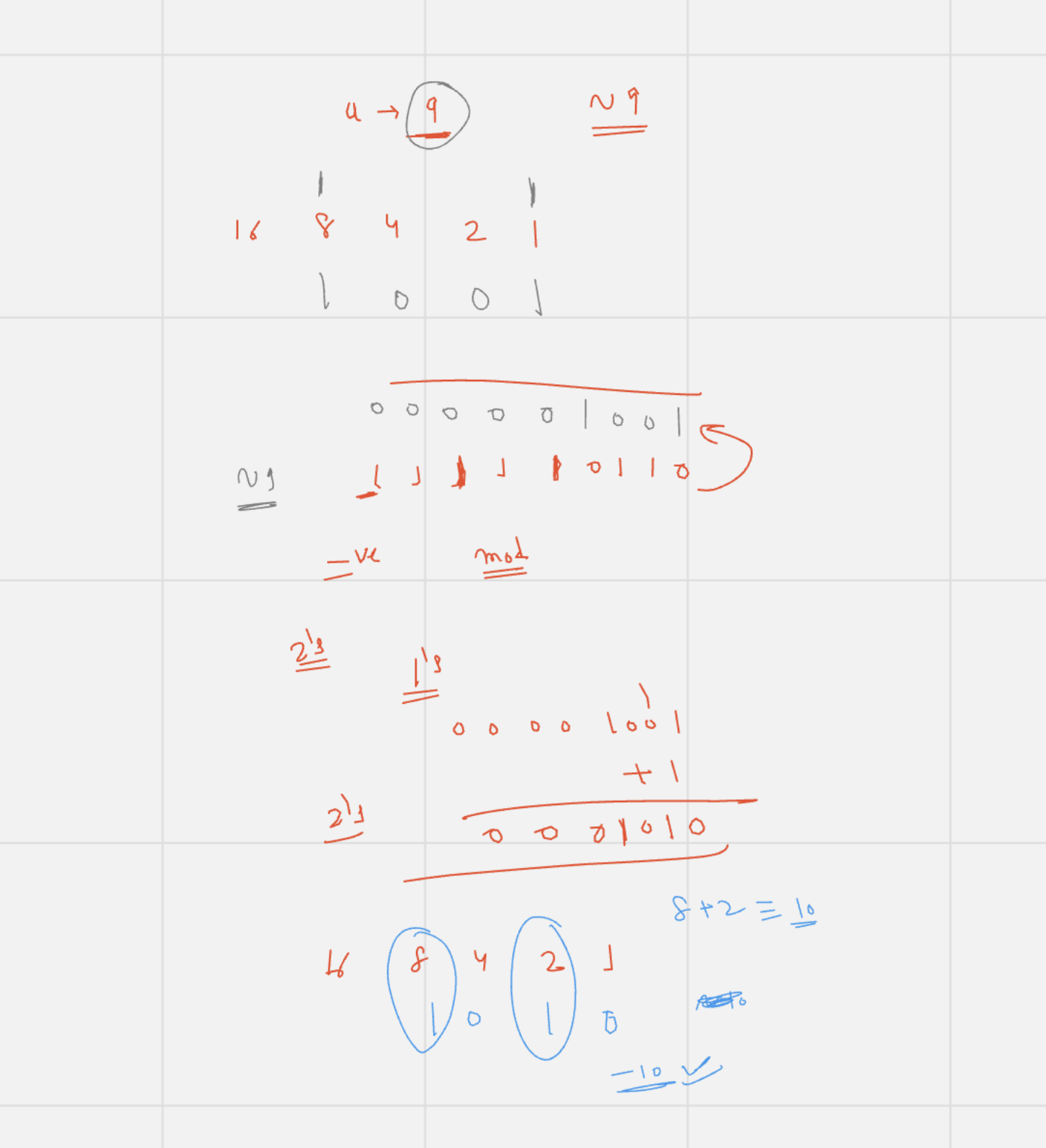
16 8 4 2 1

1 1 0

~5 → -6 ✓

a → 9

~9


4 → 9
~9
16 8 4 2 1
1 0 0 1
0 0 0 0 0 0 0 1
~9
-ve
mod
2's
1's
0 0 0 0 1 0 0 1
+ 1
2's
0 0 0 1 0 1 0
8+2 = 10
4 8 4 2 1
1 0 1 0
-10