n → 10 → false

n → $4 = 2^2 =$ True

$\underline{\underline{O(1)}}$

$$\boxed{2^0 \quad 2^1 \quad 2^2 \quad - - - - \quad 2^n}$$

$\log_2 n$

Decimal →

$\underline{0}\ \ \underline{0}\ \ \underline{\cancel{0}1}\ \ \underline{1}\ \ \underline{0}\ \ \underline{0}$ = 2^2

$2^4\ \ 2^3\ \ 2^2\ \ 2^1\ \ 2^0$ = 2^0

100100

100000

001000

0000000

n → bit (one 1 bit is on)

4

$2^2 + 2^3$

$4 + 8 = 12$

rsb = $\boxed{n \,\&\, -n}$

$n - (n \,\&\, -n)$

010100

rhs → 100

010100

rsb & 100

000100

001000

& 10000

0010000

$n \rightarrow$

$n-1$

$$\boxed{n \,\&\, (n-1)}$$

$$\begin{array}{r} -1 \\ 0 \quad -1 \\ 0\,1\,0\,0\,1\,0\,0 \\ -1 \\ \hline 0\,1\,0\,0\,0\,1\,1 \end{array}
\qquad\qquad
\begin{array}{r} \downarrow \\ 0\,1\,0\,0\,1\,0\,0 \\ \&\; 0\,1\,0\,0\,0\,1\,1 \\ \hline 0\,1\,0\,0\,0\,0\,0 \end{array}$$

set rsb to 0

off rsb

$$4^n = (2^2)^n = 2^{2n}$$

$4^0 = 1$ 2^0

$4^1 = 4$ 2^2

$4^2 = 16$ 2^4

$2^4 \quad 2^3 \quad 2^2 \quad 2^1 \quad 2^0$

0 0 0 0 0 0 0 1

0 0 0 0 0 0 1 0

$\boxed{1 0 1 0 1 0 1 0 1 0 1}$

single bit

↓

$n \to 100100$

$\& \; 0 1 0 1 0 0$

0 0 0 0 0 0

2^6 2^5 $\underline{2^4}$ $\underline{2^3}$ $\underline{2^2}$ $\underline{2^1}$ $\underline{2^0}$

0 0 0 0 0 0 1 = 2^0

0 0 0 0 1 0 0 = 2^2

0 0 1 0 0 0 0 = 2^4

2^4 ↓

0 0 0 1 0 0 0 0

& 0 1 0 1 0 1 0 1 ← Mask

0 0 0 1 0 0 0 0

> 0

0 0 1 0 0 0 = 8

& 0 1 0 1 0 1 0 1 0 1 0 1

0 0 0 0 0 0 0 = 0

✓ 0 ✓ 0 ✓ 0 ●

$\text{int val} = (5)_{10}$

binary

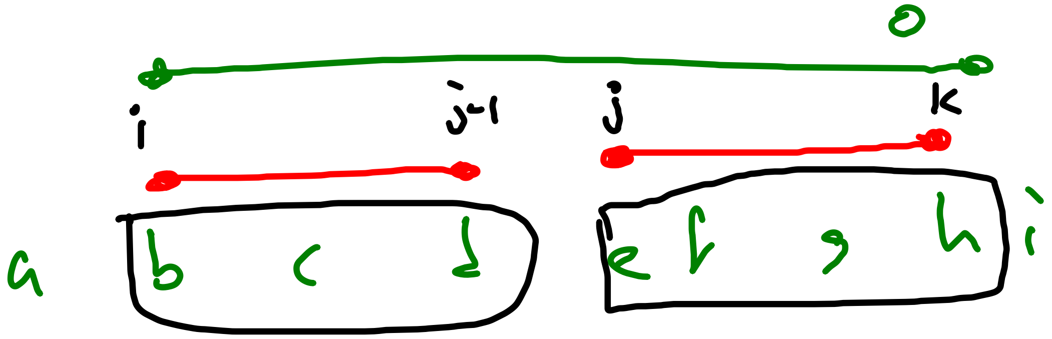
$\text{int val} = 0b\,\underbrace{010101010101010101}_{32}$

$\text{int val} = 0o\ \underbrace{0\,1\,2\,3\,4\,5\,6\,7\,0\,1\,2\,3\,\text{----}}$

000 001 010 011 111

0...7

- - -



XOR

$ar[i] \dots ar[j-1]$

$ar[j] \dots ar[k]$

$\boxed{0}$

$[b \wedge c \wedge d]$ $e \wedge f \wedge g \wedge h$

$\boxed{u = u}$

1 2 3

$= 1 - 2 \wedge 3$ $\boxed{2}$

$1 - 1$

$= 1 \wedge 2 - 3$

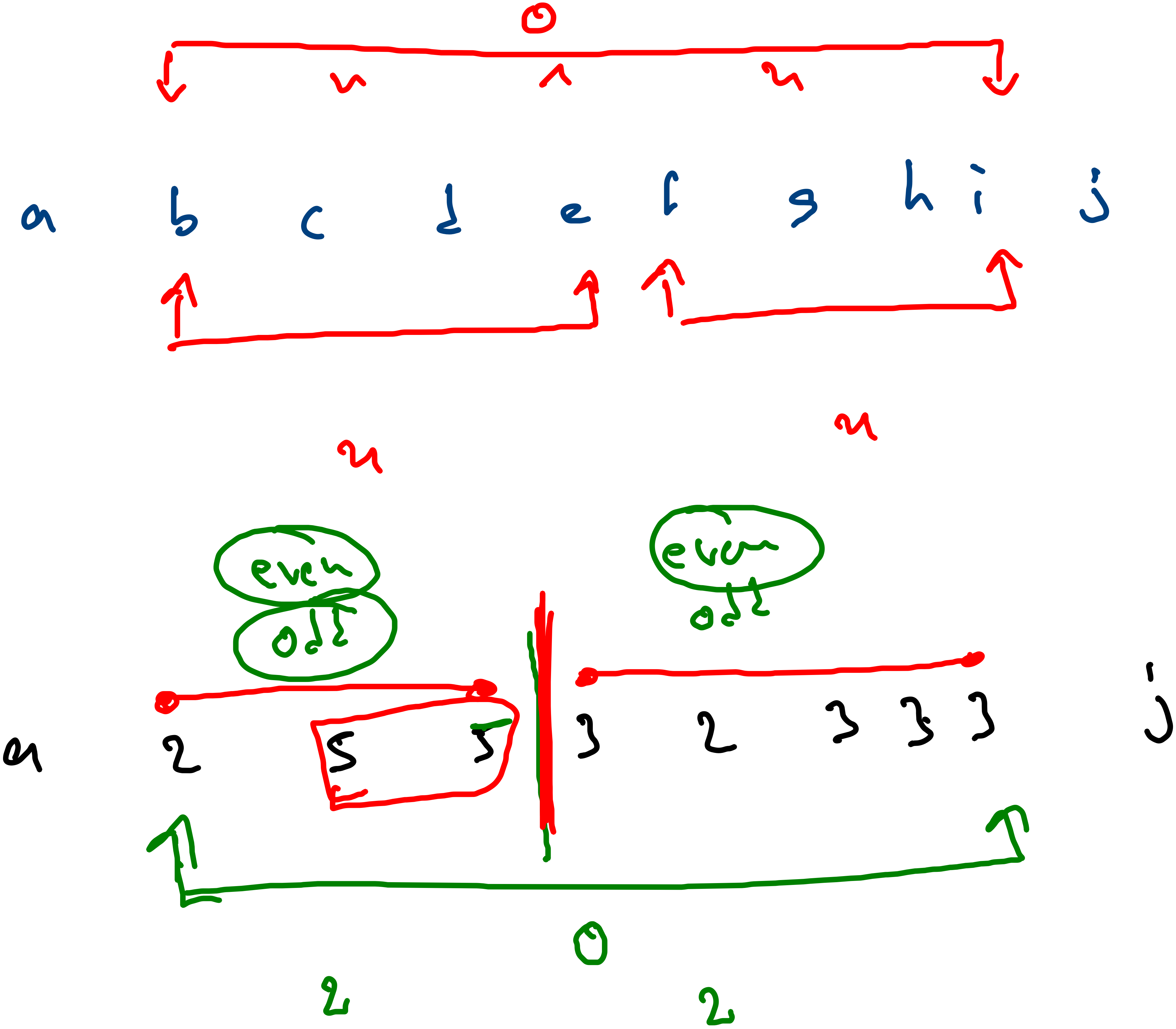
$3 = 3$

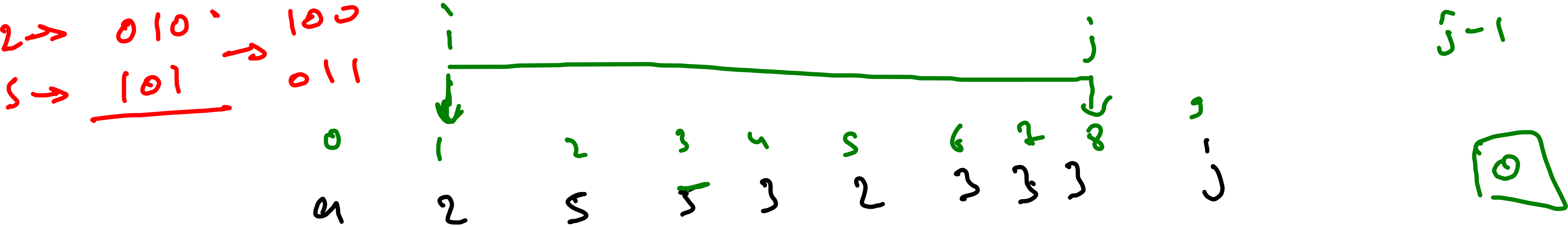
$$\begin{array}{r} 001 \\ 010 \\ \hline 011 \end{array} \qquad \begin{array}{r} 010 \\ 011 \\ \hline 001 \end{array}$$

1 → 001

2 → 010

3 → 011


o
n
n
n
a b c d e f g h i j
n
n
even
odd
even
odd
a 2 5 5 3 2 3 3 3 j
0
2
2



2 — ~~5~~ ~~5~~ ~~5~~ 2 ~~3~~ ~~3~~ ~~3~~

2 5 — 5 ~~5~~ 2 ~~3~~ ~~3~~ ~~3~~

2~~5~~ ~~5~~ — ~~5~~ 2 ~~3~~ ~~3~~ ~~3~~

2~~5~~ ~~5~~ 3 — 2 3 ~~3~~ ~~3~~

2^3 2^3

~~2~~~~5~~ ~~5~~ 3 ~~2~~ — 3 ~~3~~ ~~3~~

~~2~~~~5~~ ~~5~~ ~~3~~ ~~2~~ ~~3~~ — ~~3~~ ~~3~~

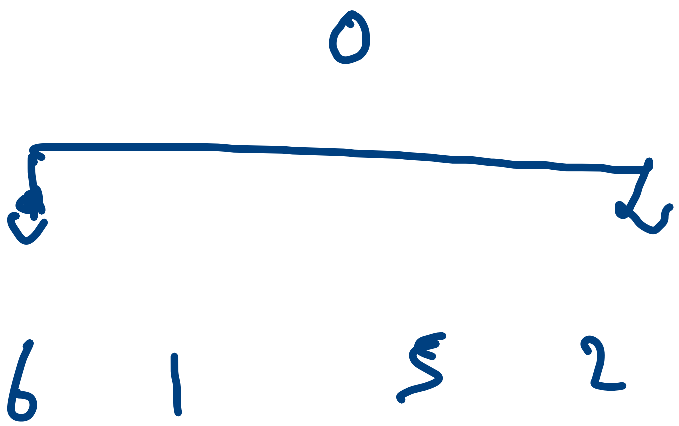
~~2~~~~5~~ ~~5~~ 3 ~~2~~ ~~3~~ ~~3~~ — 3

6 – 110

→ 001

101

010



110

1	001
5	101
2	010

110

110

2

000

$6 - \overset{6}{(1 \wedge 5 \wedge 2)}$

$6 \wedge 1 = 5 \wedge 2$

$6 \wedge 1 \wedge 5 = 2$

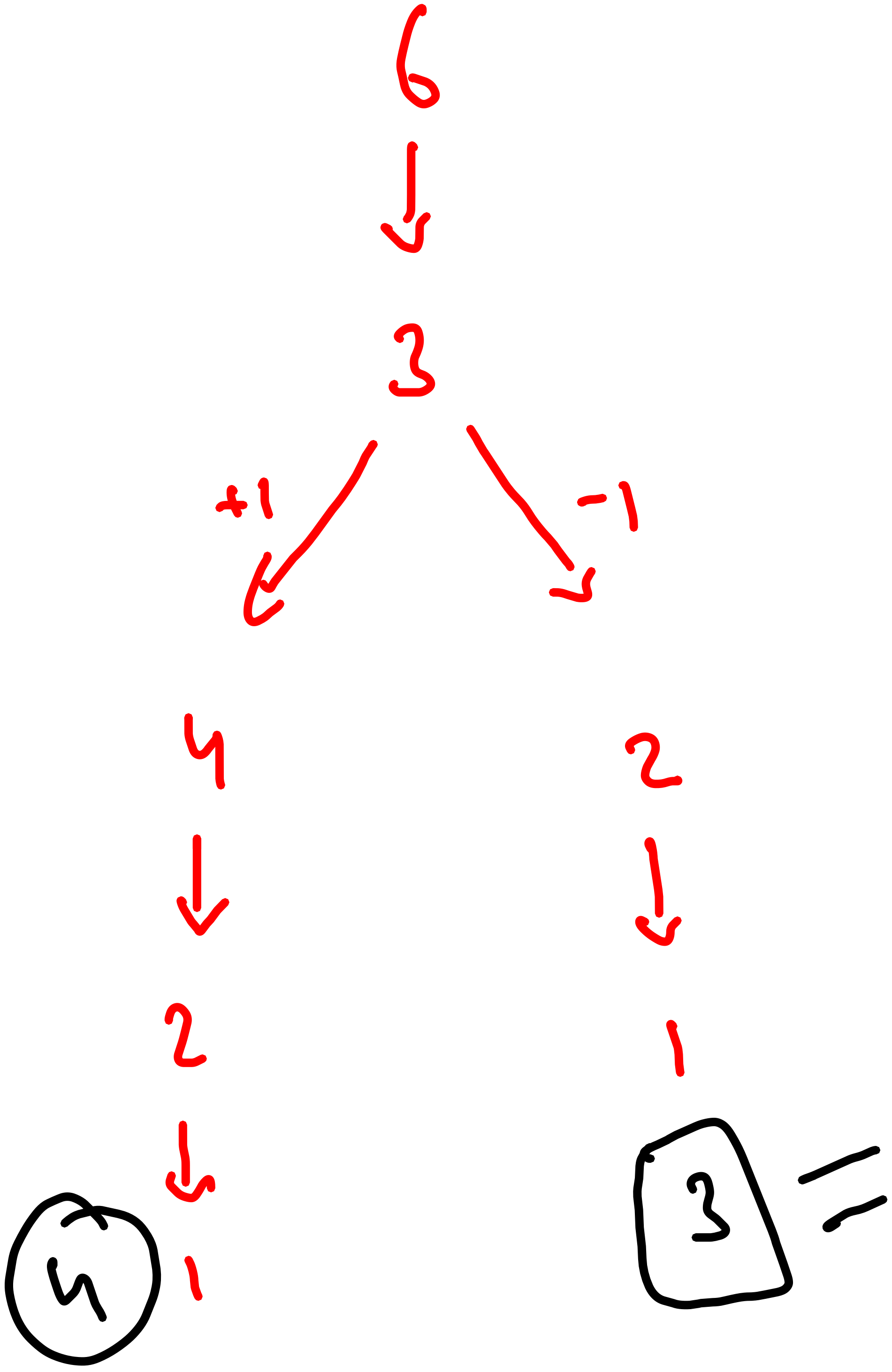
n → ?

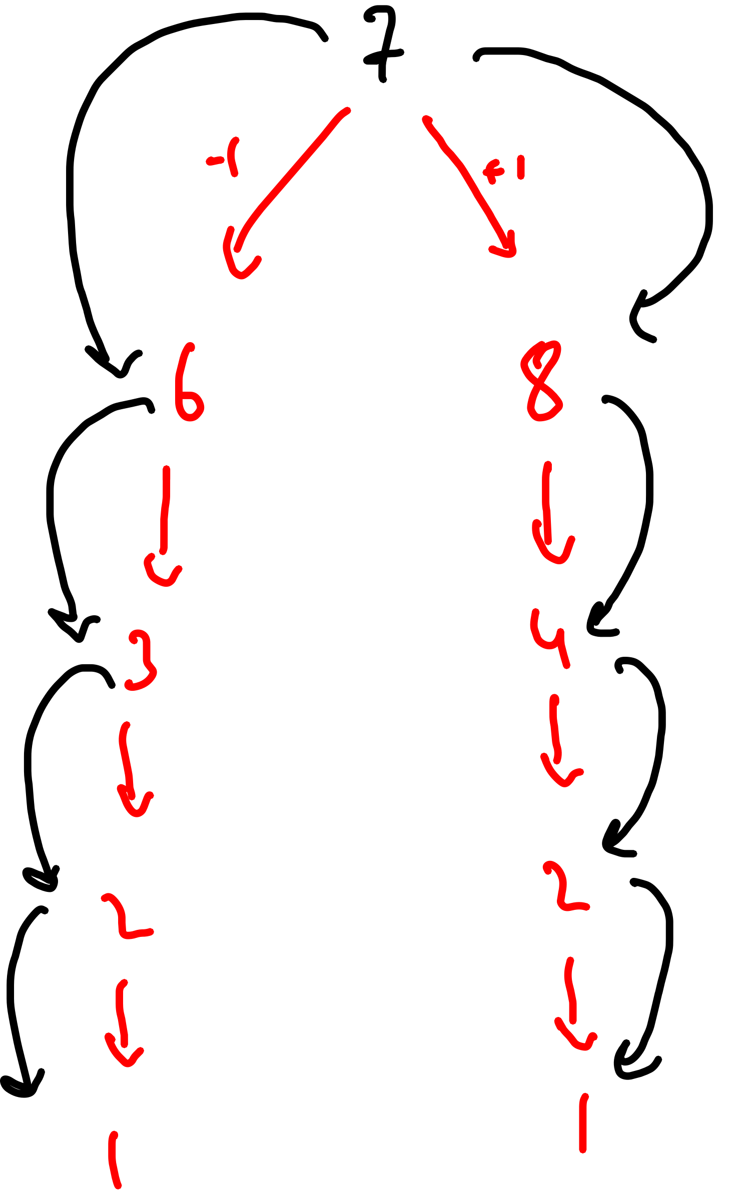
6 → 1 3

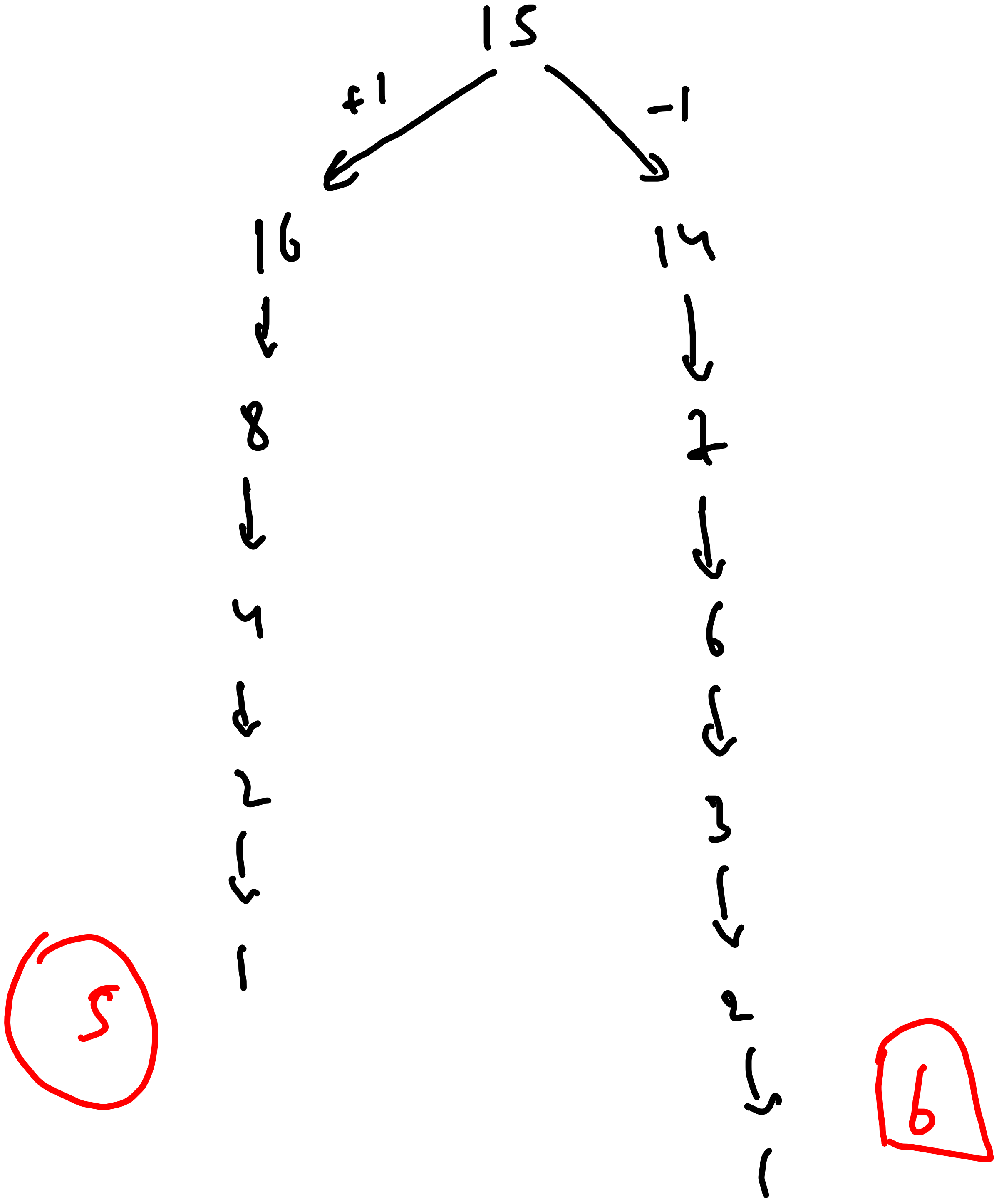
even n/2

odd n−1

n+1




7
-1
+1
6
8
3
4
2
2
1
1


15
+1
-1
16
14
8
7
4
6
2
3
1
2
1
5
6

- divide 2
- +1
- −1

$$
\begin{array}{ccc}
100000 & \rightarrow & 2^5 \\
\downarrow & & \downarrow \\
10000 & & 2^4 \\
\downarrow & & \downarrow \\
1000 & & 2^3 \\
100 & & \downarrow \\
 & & 2^2 \\
10 & & \downarrow \\
1 & & 2^1
\end{array}
$$

010100

101000

$2^5 \quad 2^3$

$$\boxed{n/2 = n>>1}$$

$$n = \left(2^5 + 2^3\right)$$

$$n/2 \quad \frac{2^5 + 2^3}{2} = 2^4 + 2^2$$

- - - - - 0 → even
- - - - - 1 → odd

H.W

000001

10010011
 - 1

10010010

10100100

1010010 0

101001

100110

10011

10010

1001

1000 → 100 10 1

0→0

1001001

+1

1001010

$\boxed{-1}$

1→0

1001001

-1

1001000

$\boxed{1→1 \quad 2→0}$

1→0

+1 +1

100 1011

+1

100 1100

1→0

100 1011

$\boxed{-1}$

1001010

even

/

odd

1→1 3→0

1 +1 +1 +1

10 1111

+1

110000

101111

-1

101110

2→0

+1 +1 +1

100 $\boxed{111}$

$\boxed{+1}$

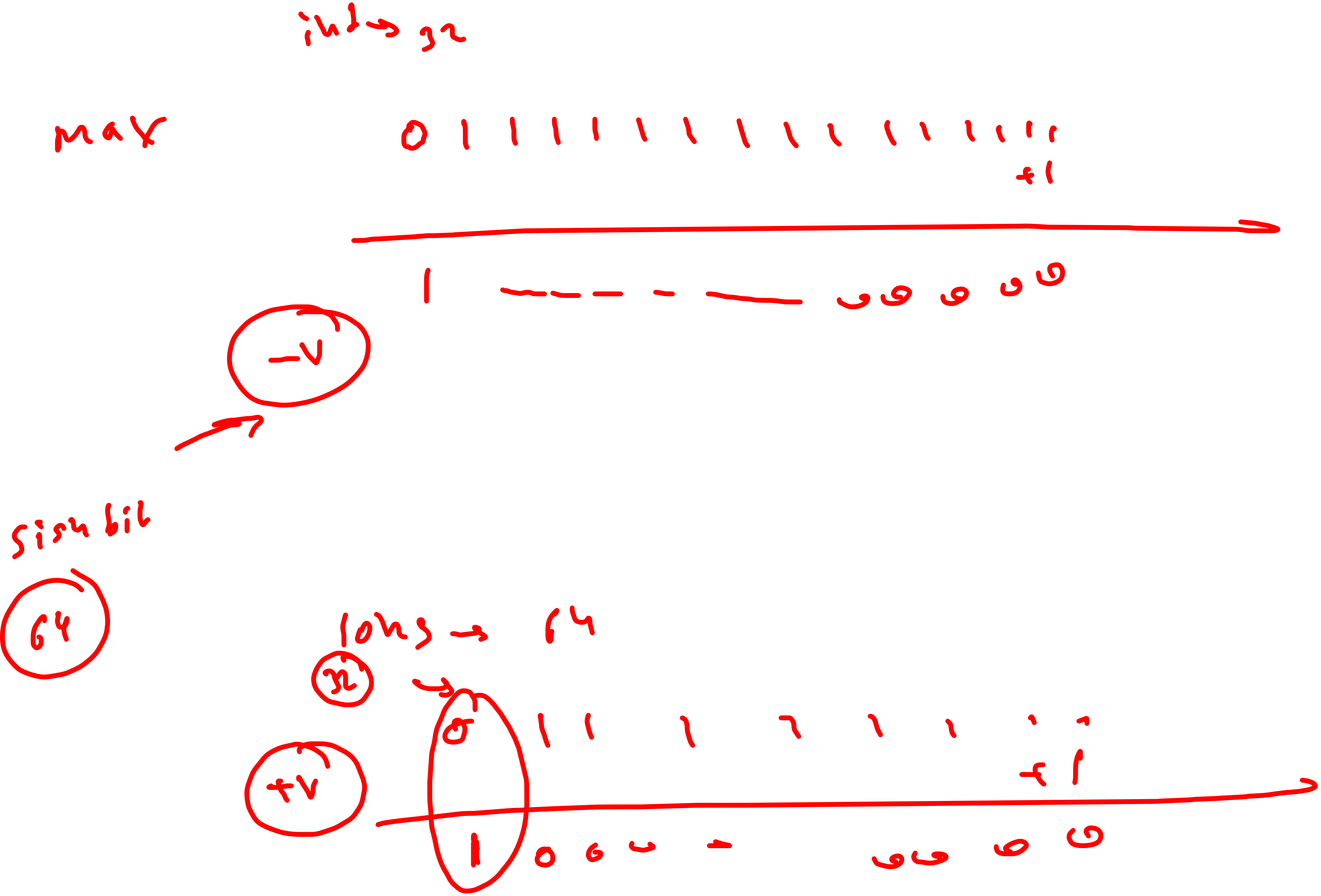
101000

100111

-1

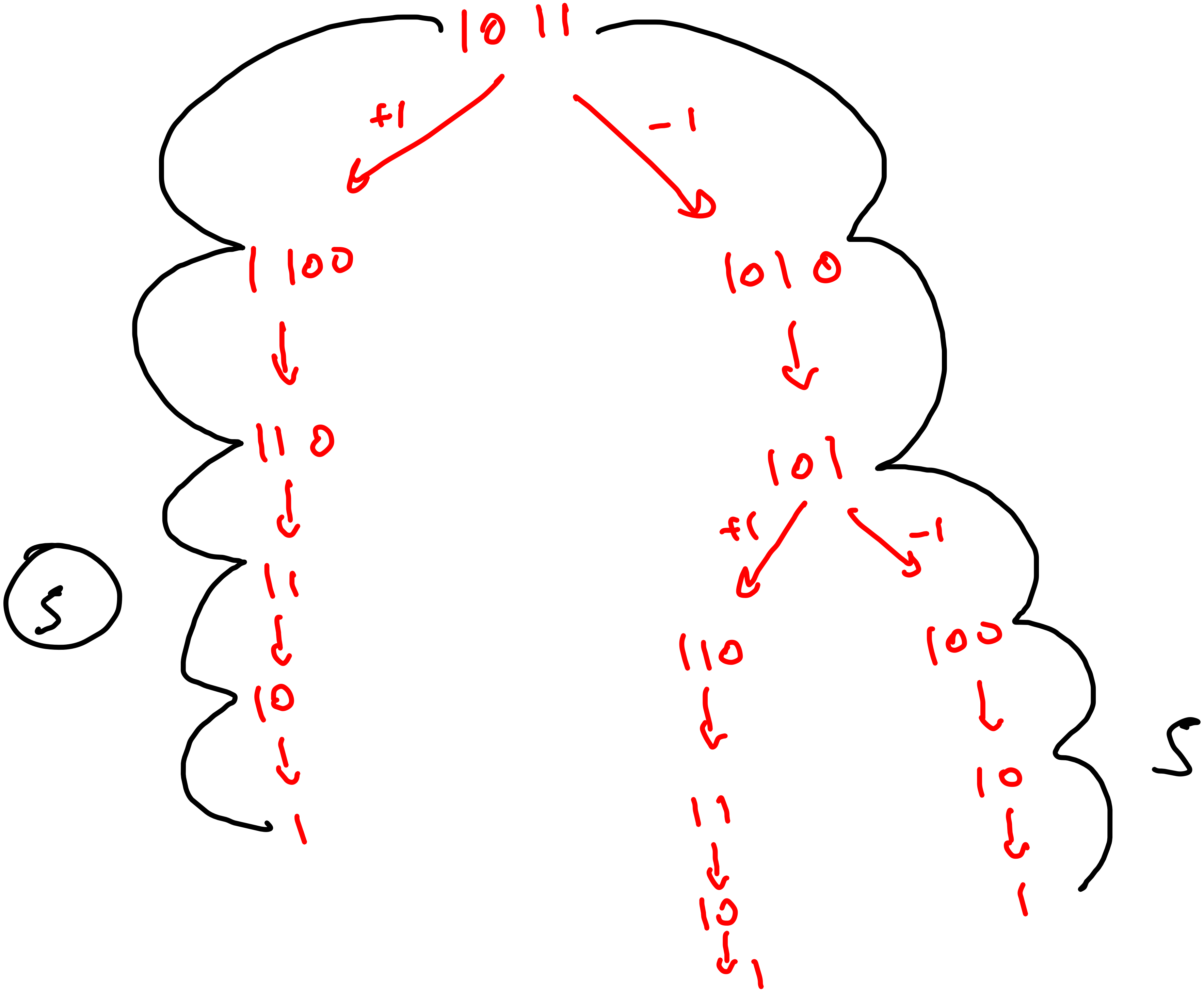
100110

1→0


int → 32
max
0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
+1
1 - - - - - - - - 0 0 0 0 0 0 0 0
-v
sign bit
64
long → 64
32
0 1 1 1 1 1 1 1 1
+1
1 0 0 0 - 0 0 0 0 0 0
+v

011 → −1

+1011 → +1
→ −1



$2^n = 0 \ldots 2^n - 1$ pep

$3 = 2^3$

n

2^n

0

7

$2^n = 8$

000	pep	pep
001	pe-	pe1
010	p-p	p1p
011	p--	p2
100	-ep	1ep
101	-e-	1e1
110	--p	2p
111	---	3

pep
pe1
p1p
p2
1ep
1e1
2p
3

$0 \ldots < 2^n$

i 0, 1, 2

0 1 2
p 1 p

$b = n-1-i$

i
0 → n-1
1 → n-2
2 → n-3

i
0 → n-1 -0
1 → n-1 -1
2 → n-1 -2

i
$b = n-1-i$

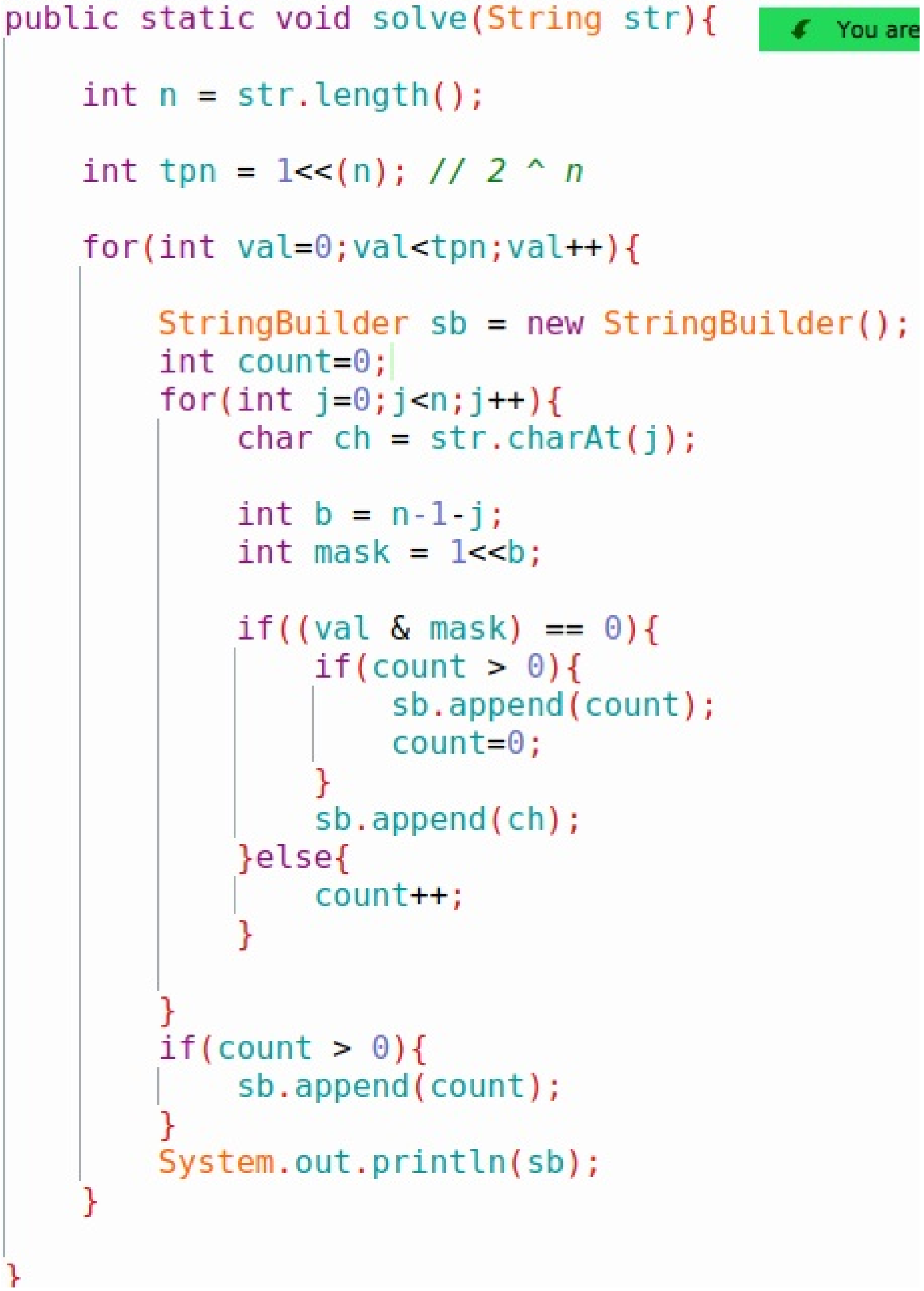
0
1 << b

if((val & mask) == 0)
= 0

else {
= 1
}

3 2 1 0

000	p e p	
001	p e 1	
010	p 1 p	
011	p 2	count = ~~1~~ 2
100	1 e p	count = ~~1~~ 0
101	1 e 1	~~0~~ ~~1~~ ~~0~~ 1
110	2 p	~~0~~ ~~1~~ ~~2~~ 0
7 111	3	~~0~~ ~~1~~ ~~2~~ 3

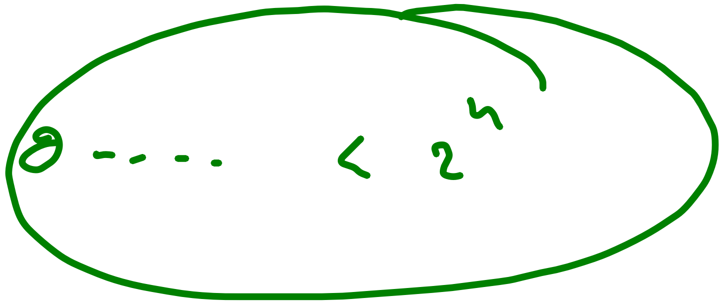


```
public static void solve(String str){

    int n = str.length();

    int tpn = 1<<(n); // 2 ^ n

    for(int val=0;val<tpn;val++){

        StringBuilder sb = new StringBuilder();
        int count=0;
        for(int j=0;j<n;j++){
            char ch = str.charAt(j);

            int b = n-1-j;
            int mask = 1<<b;

            if((val & mask) == 0){
                if(count > 0){
                    sb.append(count);
                    count=0;
                }
                sb.append(ch);
            }else{
                count++;
            }

        }
        if(count > 0){
            sb.append(count);
        }
        System.out.println(sb);
    }

}
```

count=~~0~~ ~~1~~ ~~0~~ ~~1~~ 2

0 1 2 3
pepe

3 2 1 0
val → 1011 → 1e2



UTF-8

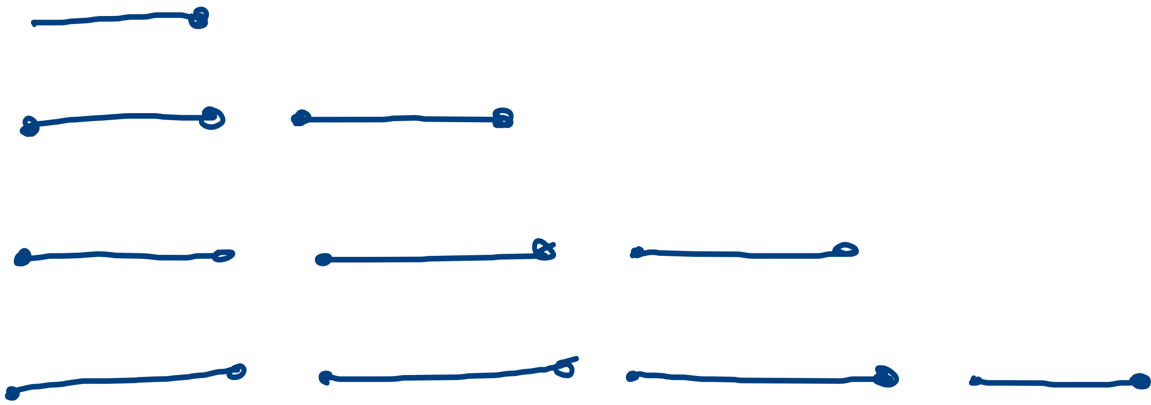
1 byte = 8 bit

0 - - - - - - -

110 - - - - - 10 - - - - - -

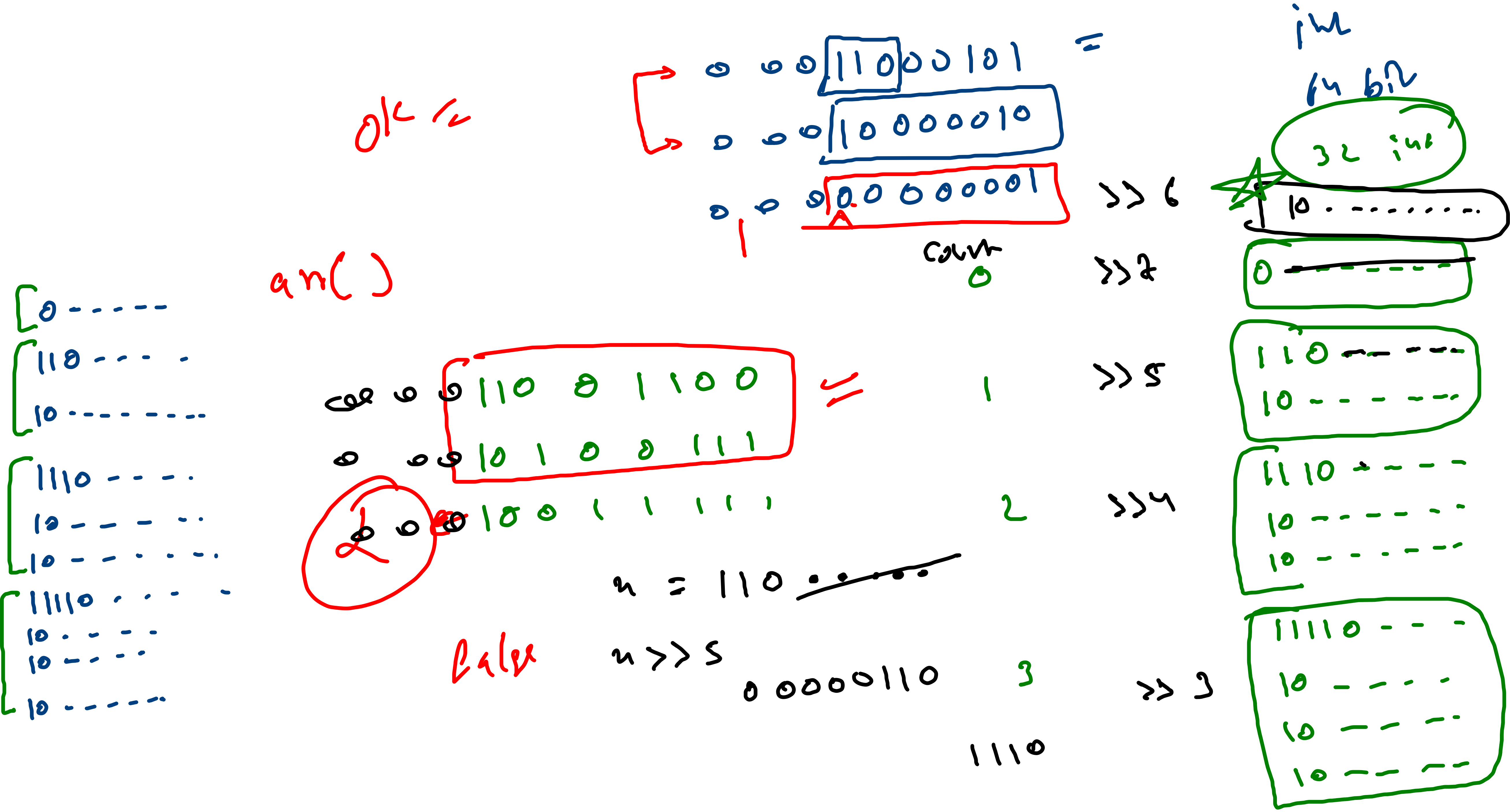
1110 - - - - - 10 - - - - - - 10 - - - - - -

11110 - - - 10 - - - - - - 10 - - - - - - 10 - - - - - -

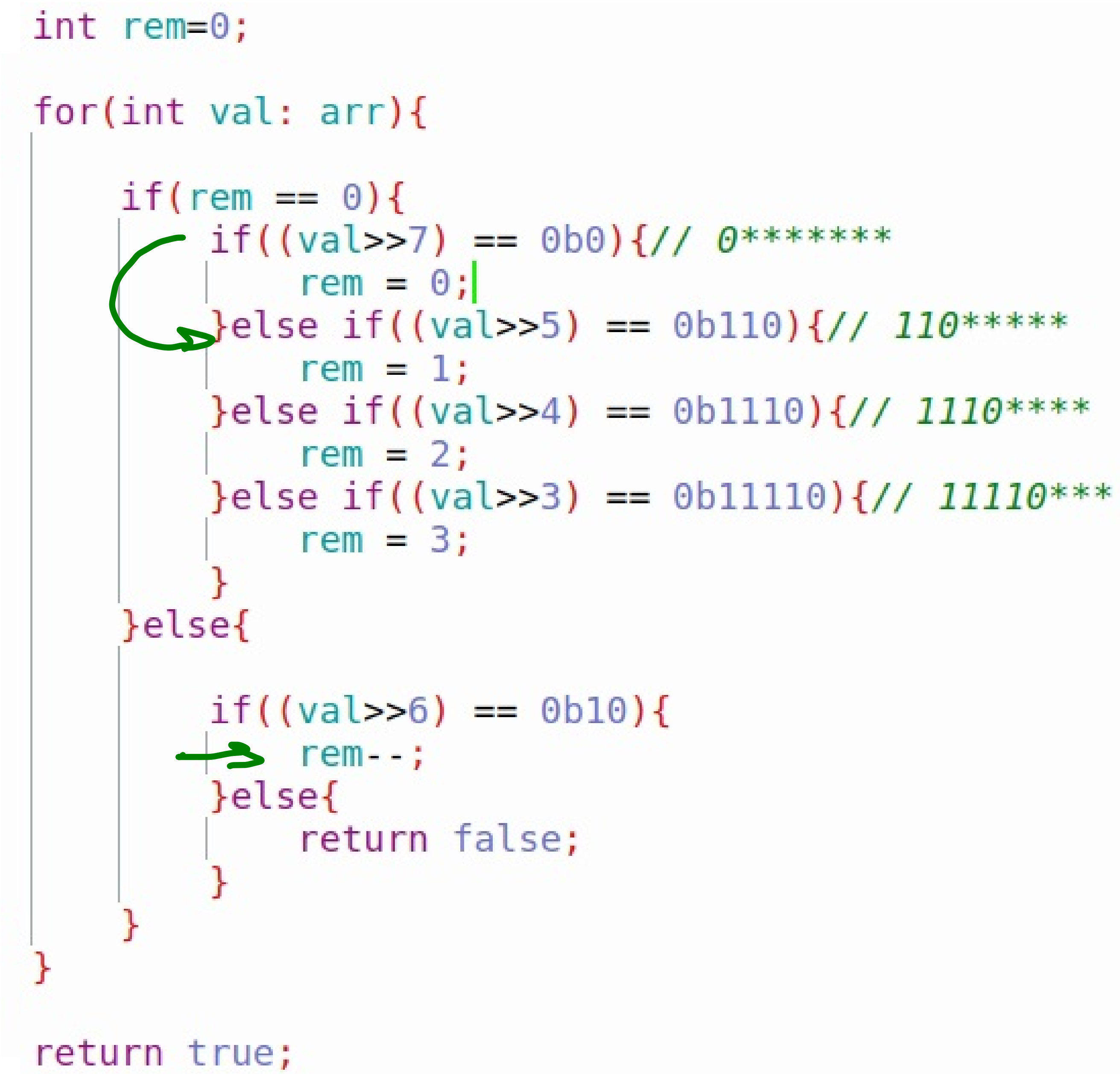


arr(int

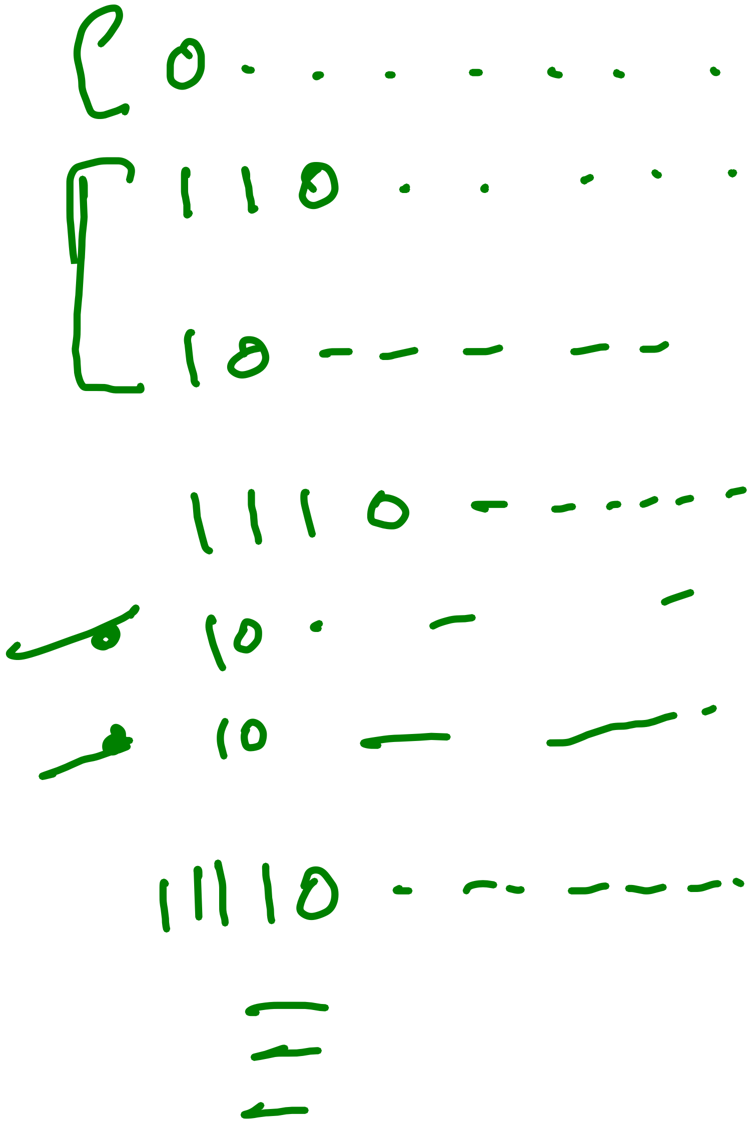
)


int
64 bit
32 int
ok ✓
0 00 11000101 =
0 00 10000010
0 0 0 00000001
>> 6
count
0
>> 7
an( )
0 - - - - -
110 - - - -
10 - - - - - -
1110 - - - -
10 - - - -
10 - - - -
11110 . . .
10 - - -
10 - - -
10 - - - -
000 110 0 1100 ✓
0 00 10 1 0 0 111
2 0 0 100 1 1 1 1 1
1
>> 5
2
>> 4
x = 110 . . . . .
Calse
x >> 5
0 0000110
3
>> 3
1110
10 . . . .
0 - - - - - -
110 - - - -
10 - - - -
1110 - - -
10 - - - -
10 - - - -
11110 - -
10 - - -
10 - - -
10 - - -

rem: ~~0~~ ~~1~~ ~~0~~ ~~2~~ ~~1~~ ~~0~~ 3



```
int rem=0;

for(int val: arr){

    if(rem == 0){
        if((val>>7) == 0b0){// 0*******
            rem = 0;
        }else if((val>>5) == 0b110){// 110*****
            rem = 1;
        }else if((val>>4) == 0b1110){// 1110****
            rem = 2;
        }else if((val>>3) == 0b11110){// 11110***
            rem = 3;
        }
    }else{

        if((val>>6) == 0b10){
            rem--;
        }else{
            return false;
        }
    }
}

return true;
```


0 . . . .
110 . . . . .
10 - - - - - -
1110 - - - - -
10 . - -
10 - -
11110 . - - - -