Bitwise Operator

⇒ &, |, ^, ~, <<, >>

& → AND ⇒

A	B	R
T	T	T
T	F	F
F	T	F
F	F	F

→ 1, 0 → T, F

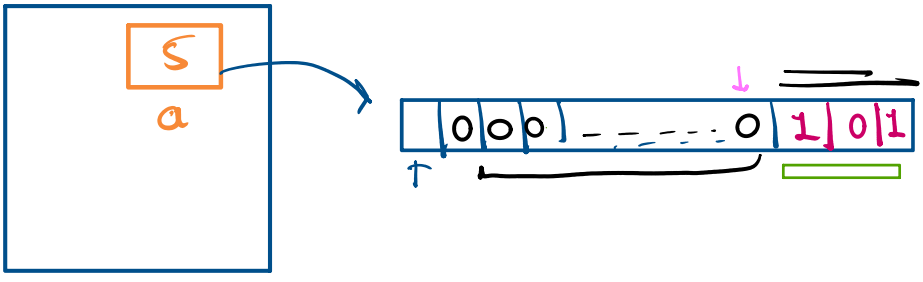
10 → 1010 → TFTF

→ 5, 10

& 0101 → & FTFT
1010 → TFTF

int a=5

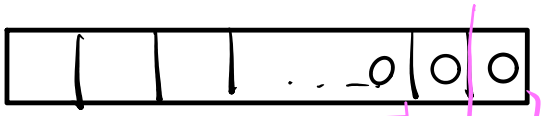
int - 4 byte → 1 byte = 8 bits



T & T = T ⇒ FTFT / TFTF

0 ← 0000 ← FFFF

0 ← 0000 ← FFFF



⇒ 0

Bit mani-

1 = 01 →
2 ⇒ 10 →
3 ⇒ 11
4 ⇒ 100
5 ⇒ 101

decimal → binary

lost digit

1 → odd
0 → even

t & T = t

0 1

1 ⇒

| = OR

A	B	R
T	T	T
T	F	T
F	T	T
F	F	F

⇒ F

5 | 10 = 5 = 101

10 = 1010

0 1 0 1
1 0 1 0

1 1 1 1 ⇒ 15

1 1 1 1 ⇒ 15

∧ =

A	B	R
T	T	F
T	F	T
F	T	T
F	F	F

⇒

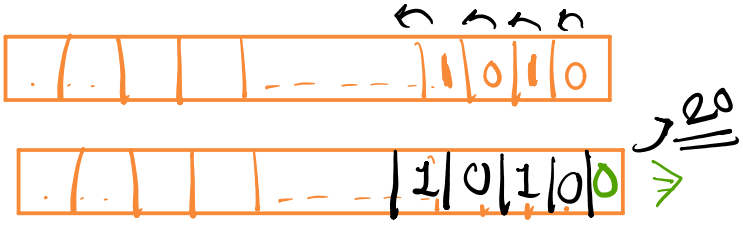
~ =

T	F
F	T

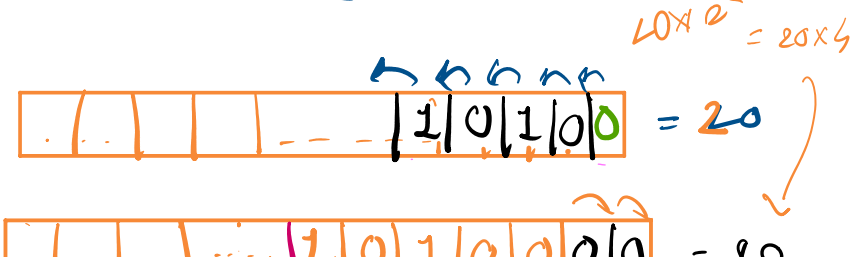
<< ⇒ Right shift

num = 10 ⇒ 1010

x = num << 1



num << 2



num = num << 2

↳ binary

↳ end → 00

num = num << 1

= num * 2^1 num * 2^3

= num * 2^2

num = num << x

num * 2^x

>> = left shift

10 = 1010 >> 1

101

10

1

0

$10/2^x = 10/2^{-3}$

$10/2^2 = 2$

$10/2^3 = 1$

$10/2^4 = 0$

$10/4 =$

$10/8 = 10/16$

$10/2^5 =$

⇒ num = 10 Binary ⇒ ?

↳ Decimal = 0, ---- 9 = $(x)_{10}$

↳ Octal = 0 ---7 = $(x)_8$

↳ Binary = 0, 1 = $(x)_2$

↳ Hexadecimal = 0 --- 9 ... A-F

$(x)_{16}$

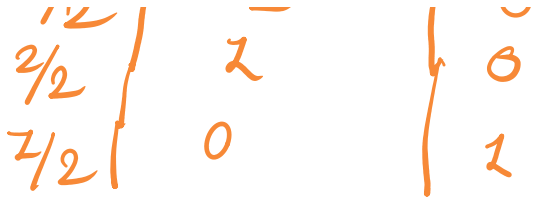
$x = 294_{(10)} \rightarrow (2)$

↳ 294/2 → 0 →

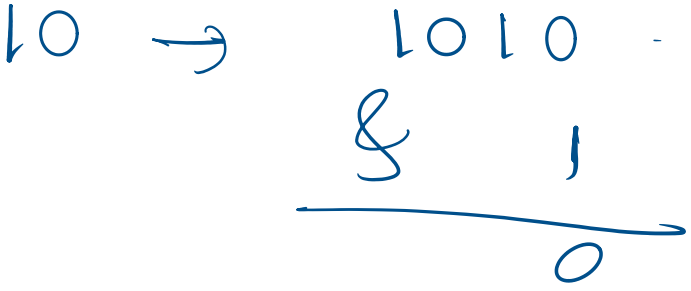
num	Quotient	rem
	295	
294/2	147	0
147/2	73	1
73/2	36	1
36/2	18	0
18/2	9	0
9/2	4	1
4/2	2	0
2/2	1	0

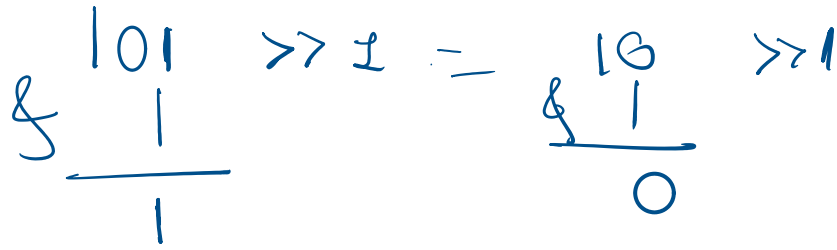
MSB ↓ LSB ↓

$(100100110)_2$



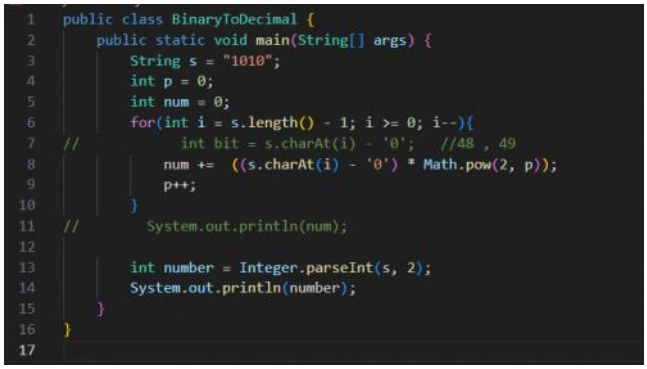
1. We Divide N until it not becomes 0
   a. While dividing we store it's remainder and once N become 0 we simply reverse the remainder (Array / String) and return or print to get valid binary conversion





Homework

↳ try using >> operator



```java
public class BinaryToDecimal {
    public static void main(String[] args) {
        String s = "1010";
        int p = 0;
        int num = 0;
        for(int i = s.length() - 1; i >= 0; i--){
//            int bit = s.charAt(i) - '0';   //48 , 49
            num +=  ((s.charAt(i) - '0') * Math.pow(2, p));
            p++;
        }
//        System.out.println(num);

        int number = Integer.parseInt(s, 2);
        System.out.println(number);
    }
}

```