Number System

Method to represent numeric values or quantities using different digits.

Symbols → differently arranged ⇒ quantity

eg 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

eg ⇒ 17, 100, 91 etc.

⊙ **Decimal Number System →**

→ It has base 10. → (0 - 9 digits)

↳ Base is the number of symbols (digits) a number system uses.

→ It uses digits from 0 to 9.

⊙ **Binary Number System →**

→ It has base 2. → (0 and 1 digits)

→ Only two symbols (or digits) are used i.e 0 and 1.

Anything inside memory is stored in binary and CPU also does calculation in binary.

eg → 5 (In decimal) ⟶ 101 (In binary)

⊙ **Decimal to Binary Conversion -**

1. **Division Method -**

1. Divide Number by 2.
2. Store remainder. (That will be a bit in binary).
3. Repeat above steps with the quotient until quotient is less than 2.
4. Reverse the bits.

eg → N = 10

Division	Rem
10/2 = 5	0
5/2 = 2	1
2/2 = 1	0
1/2 = 01	1

we got

↳ 0101

now reverse the bits

⇒ 1010, which is 10 in binary no. system.

~~int dem~~

```
int decimalToBinary(int n){
    int binaryNo = 0; i = 0;
    while(n>0){
        int bit = n%2;
        n /= 2;
        binaryNo += bit * pow(10, i++);
        n /= 2;
    }
    return binaryNo;
}
int main(){
    int n;
    cin>>n;
    int binary = decimalToBinary(n);
    cout<<binary<<endl;
    return 0;
}
```

10 10
↳ $0\times10^0+0=0$
↳ $1\times10^1+0=10$
↳ $0\times10^2+10=10$
↳ $1\times10^3+10=1010$

- dry run →

n = 10, i = 0

binaryNo: 0 → 0 → 1010

i: 0 → 1 → 2 → 3

n > 0 → 10 > 0 → T

bit = 10 % 2 = 0

binaryNo = 0 + bit × pow(10, 0) = 0 + 0 × 10^0 = 0

(i is incremented by 1)

n = 5

5 > 0 → T

bit = 5 % 2 = 1

binaryNo = 0 + 1 × 10^1 = 10

i = 1 + 1 = 2

n = 2

2 > 0 → T, bit = 2 % 2 = 0, binaryNo = 10 + 1 × 10^2 = 10, i = 2 + 1 = 3

n = 2/2 = 1

1 > 0 → T, bit = 1 % 2 = 1, binaryNo = 10 + 1 × 10^3 = 10 + 1000 = 1010, i = 3 + 1 = 4

n = 1/2 = 0

0 > 0 → False (out of loop) ⇒ return binaryNo. 1010

2. Bitwise Method -

~~Divide number by 2.~~

1. Obtain bit with Bitwise AND operation i.e (N & 1)
2. Right shift n by 1. (N = N >> 1)
3. Repeat above steps till N > 0.
4. Reverse the bits. ~~so~~

eg → n = 10

0, 1, 0, 1

10 & 1 = 0 , 10 >> 1 = 5 = 101

101 & 1 = 1 , 5 >> 1 = 2 = 10

10 & 1 = 0 , 10 >> 1 = 1

1 & 1 = 1

10 ⇒ 1010 (in binary)

code -

```
int bitwiseMethod (int n){
    int binaryNO = 0;
    int i = 0;
    while(n > 0){
        int bit = (n & 1);
        binaryNO += bit * pow(10, i);
        n = n >> 1;
    }
    return binaryNO;
}
```

⊙ Binary to Decimal Conversion →

1. Multiply each digit with its place value.
2. Add up all place values.
3. Sum is the decimal number.

place value = digit × (base)^i

1 2 3

1×10^2 + 2×10^1 + 3×10^0

100 + 20 + 3 = 123

1	0	1	0
1×2^3	0×2^2	1×2^1	0×2^0

$8 + 0 + 2 + 0 = \underline{10}$

```
      11
10 | 111 |
     10
    ----
     11
     10
    ----
      1
```

$111 \% 10 = 1$

code →

```
int binaryToDecimal (int n){
    int decimal = 0;
    int i = 0;
    while (n>0){
        int bit = n % 10;
        deimal = decimal + bit * pow(2, i++);
        n /= 10;
    }
    return decimal;
}
```

dry run -

Binary -> $(1010)_2$

Bit = $1010 \% 10 = 0$

decimal = $0 + 0\times2^0 = 0$

$n = 1010 / 10 = 101$

Bit = $101 \% 10 = 1$

decimal = $0 + 1\times2^1 = 2$

$n = n/10 = 10$

Bit = $10 \% 10 = 0$

decimal = $2 + 0\times2^2 = 2$

$n = n/10 = 1$

Bit = $1 \% 10 = 1$

decimal = $2 + 1\times2^3 = 2+8 = 10$ ✓

$n = 1/10 = 0$