BTD
DTB

① Decimal to Binary → 0101

10 , 11 , 2 , 3

3 → 0 1 1 ✓
2 → 0 1 0

1st Approach.

2nd coding

4

2 | 4
2 | 2 – 0 ↓
2 | 1 – 0
0 – 1

001
⇒ 100 – Binary no. of 4

1) Divide number by 2
2) Write remainder
3) Repeat until quotient becomes 0
4) Then Reverse the remainders (Bottom to top)

2 | 10
2 | 5 – 0
2 | 2 – 1
2 | 1 – 0 ↑
0 – 1

0101
1010

1010 → Binary
10

12 → binary?

1st Approach

0011 → 1100

1100

Divide by 2	Quotient	Remainder
12/2	6	0 ↓
6/2	3	0
3/2	1	1
1/2	0	1 ↑

2nd Approach → Odd & Even

& 0

last => 1 (odd)

last => 0 (even)

3 => 0011

2 => 0010

7 => 0111

8 => 1000

1 } Ev

0

2^-

0 0 1 1

2^3 2^2 2^1 2^0

8 4 2 ¥ = odd / Even

1) Check last bit (num&1)

number1 = 7

1 1 1

0 & 1

0 0 1 → last

number = 2

0 1 0

& 1

0 0 0 => 0

&

0 0 → 0

0 1 → 0

1 0 → 0

1 1 → 1

(number & 1) => last bit

number = 10

1 0 1 0

& 1

0 0 0 0 => 0 → Even

2) store last bit.

variable = last bit

3) Divide number by 2 (num >> 1)

4) num != 0

while (num != 0) {

lastbit = num&1;

num = (num >>1)

}

1 0 1 0 >> 1 => 0 1 0 1 => 5

stored 0

>> 1

10 = 5

/2

num4 ✓ 4!=0 ✓

lastbit = 0

num = num>>1 (2)

4/2

100 =>

1

000 → 0

2 != 0 ✓

last bit = 0

num => 1

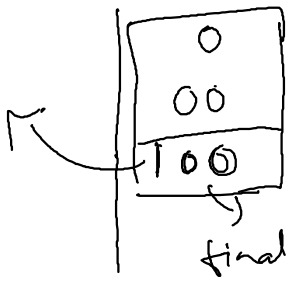
010

1

0

$\frac{2}{2} = 1$

4



001

&1

1

$\frac{1}{2} \Rightarrow 0$

1 != 0 ✓

lastbit = 1

num = 0

0 != 0 ✗

} loop ke bahaar.

$10_{De} \rightarrow B - 2$

answer = 0

answer = 10^0 + answer

1 + 0 => 1✗

=> (lastbit * 10^0) + answer

=> (0 * 1) + 0

=> 0 //

answer = (10^1 * 0 + 0)

= 0

answer = (10^2 * 1 + 0)

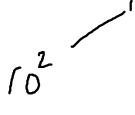
=> 100 + 0

=> 100

0 -> B

10 -> Base

n = 5



3> (1 * 10^2) + 1

=> 100 + 1

=> 101 //

1> (1 * 10^0) + 0

(1 * 1) + 0

=> 1

2> (0 * 10^1) + 1

0 + 1

=> 1

101

01

101

$10^0, 10^1$

pow(10,1) $\Rightarrow 10^1$

```
int main(){
   int num=7,answer=0,i=0;
   while(num!=0){
      int lastBit=num&1;
      answer=(lastBit*pow(10,i))+answer;
      num=num>>1;
      i++;
   }
   cout<<answer;
}
```

Initial values

num = 7, answer = 0, i = 0

1) 7 != 0 ✓

lastbit = 1

answer = (1 * 10^0) + 0

=> (1 * 1) + 0

answer => 1

num = 3

i = 1

111 / 1 / 1

7 / 2

2) 3 != 0 ✓

lastbit = 1

answer = (1 * 10^1) + 1

=> (1 * 10) + 1

=> 11

num = 1

i = 2

011 / 1 / 1

3 / 2

3) 1 != 0 ✓

lastbit = 1

answer = (1 * 10^2) + 11

=> 100 + 11

=> 111

num = 0

i = 3

001 / 1 / 1

4) 0 != 0 ✗

cout<<answer

111

1/2

1) last

2) last bit store = a

3) num /2 >>

4) num != 0

Homework, n = -7 // infite loop.

Hint!

111

1's 000

2's +1

001

1's 110

+1

111

3bits

num

32 bits //

int

4 bytes

32bits

Hint 2.

= = = => (bit)

8 / 32

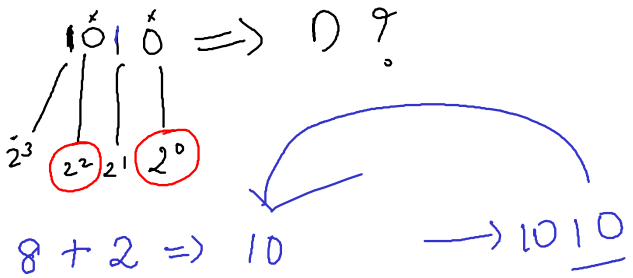
2) Binary → Decimal

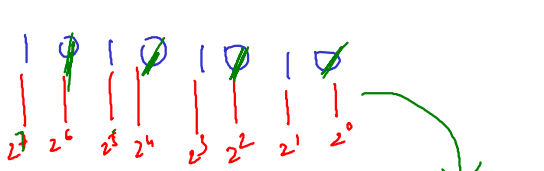
↳ 7, 8, 9, 10 → 10

1010

↳ 2

10 10 0





128 + 32 + 8 + 2 => 170

=> BO (naughty)

8 + 2 + 1 = 11 → output

Bi → 2

=> binary = 1010 (1)

1) lastbit
2) answer → formula
3) binary => reduce
4) num > 0

D → B

7 → last bit ne 1 ∝

```
{ lastbit = num % 10;
  num = num / 10 }
```

m% → remainder → lastbit

/ → quotient → num

10 | 11 (1-9) → 10 → (1) → 1

10 | 33 (3) → 30 → 3

10 | 44 (4 4) → 40 (8) ↓ → 48 → 40 → 8 → 0

=> 100

1) 100 > 0 ✓

lastbit => 100 % 10 => 0

num = 10

2) 10 > 0 ✓

lastb = 0

num = 1

3) 1 > 0 ✓

last = 1

num = 0

4) => 0 > 0 ∝

10 | 100 → 100 → 0

10 % 10 => 0

10 | 1 (0) → 0 → 1

14 | 10 → 10 → 0

0 → 2^0

00 → 2^1

100 → 2^2

ans + (last * 2^0)

output = 0

```
int main(){
   int binary=1010,i=0,answer=0;
   while(binary>0){
      int lastbit=binary%10;
      answer=(lastbit*pow(2,i))+answer;
      binary=binary/10;
      i++;
   }
    cout<<answer;
}
```

initial

binary = 1010 , i = 0 , answer = 0

1) 1010 > 0 ✓

lastbit = 0

answer = (0 * 2^0) + 0 = 0

binary => 101

i = 1

2) 101 > 0 ✓

lastbit = 1

answer = (1 * 2^1) + 0

=> 2

binary = 10

i = 2

3) 10 > 0 ✓

lastbit = 0

answer = (0 + 2^2) + 2

=> 2

binary = 1

i = 3

4) 1 > 0 ✓

lastbit = 1

answer = (1 * 2^3) + 2

=> 8 + 2

=> 10 //

binary = 0

i = 4

5) 0 > 0 ✗

cout << answer

=> 10

Homework: Convert Binary String to Decimal

Input : "101.1"

Output : 1.1

{ ≡ Soln comment