How does the processor performs arthematic, relational and logical operations?

Let us say we want to add #2 numbers 12 and 23, which is 35, how to pass this data as an input to the processor asking to perform ADD operation?
The processor of the computer can only understand anything that is represented interms of (on/off) 1s and 0s only, so inorder to pass these 12 and 23 as an input to the processor we need to convert these numbers into 1s and 0s only which is in binary number format

decimal number = base(10) = 0 - 9 number
binary number = base(2) = 0 - 1

so we need to convert the decimal number into binary number, so that it can be passed as an input to the processor. For this we need to do LCM with 2

```
 20 / 5 = 4 = 0
 20 / 3 = 6 = 2
 divisor  dividend   quotient
  3)       20      (6
           18
           __
            2 (remainder)
```

when we divide a number with a divisor, the remainder will be always less than the divisor. based on this if we divide a number by 2, the remainder would be either 0/1, so that we can express the number in binary format

```
2|12
  ___
2|6 - 0
  ___
2|3 - 0
  ___
2|1 - 1
  ___
2|0 - 1

12     = 01100
 (10)        (2)
```

```
2|23
  _____
2|11 - 1
  _____
2| 5 - 1
  _____
2| 2 - 1
  _____
2| 1 - 0
  ______
2| 0 - 1

23    = 010111
 (10)       (2)
```

how to add these 2 binary numbers?
There are 4 math rules we need to follow in adding binary numbers

```
0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 10
```

based on this lets add above 2 binary numbers

```
12 = 0   1   1   0   0
23 = 1   0   1   1   1
_______________________
     10  0   0   1   1 = 100011
                             (2)
```

what is 111011   is equivalent in decimal system?
              (2)

to find that we need to convert binary number into decimal number let us understand how?

How to convert a binary number into decimal number?
mutiply each binary number with 2(power position) * number as below.

```
01100    = decimal ?
     (2)

 0      1        1       0       0
 *      *        *       *       *
2^4    2^3      2^2     2^1     2^0
____________________________________
 0      8 +      4 +     0    +  0 = 12
```

```
1      0       0      0       1        1
*      *       *      *       *        *
2^5   2^4     2^3    2^2     2^1      2^0
_____________________________________________
32  +  0    +  0   +  0    +  2    +  1   = 35
```