In [1]:
```
25
```

Out[1]: 25

In [3]:
```
bin(25)
```

Out[3]: '0b11001'

In [5]:
```
int(0b11001)
```

Out[5]: 25

In [7]:
```
bin(35)
```

Out[7]: '0b100011'

In [9]:
```
print(bin(10))
print(bin(15))
```

```
0b1010
0b1111
```

Bitwise Operator

In [14]:
```
~12 #Complement means reverse of binary
```

Out[14]: -13

In [16]:
```
~49
```

Out[16]: -50

In [18]:
```
~-1
```

Out[18]: 0

In [20]:
```
~-12
```

Out[20]: 11

AND # 1 & 1 is 1 and rest of them are 0

In [23]:
```
12 and 13
```

Out[23]: 13

In [25]:
```
bin(12)
```

Out[25]: '0b1100'

In [27]:
```
bin(13)
```

Out[27]: '0b1101'

In [29]:
```
12 | 13
```

Out[29]: 13

In [31]:
```
1 | 1
```

Out[31]: 1

In [33]:
```
40 | 60
```

Out[33]: 60

OR mean 0 & 0 are 0 and rest of them are 1

In [35]:
```
13 or 15
```

Out[35]: 13

In [38]:
```
98 or 28
```

Out[38]: 98

In [42]:
```
print(bin(98))
print(bin(28))
```

```
0b1100010
0b11100
```

XOR means 0 & 0 is 0 and 1 & 1 is 0 rest of them are 0

In [45]:
```
12 ^ 13
```

Out[45]: 1

In [47]:
```
25 ^ 30
```

Out[47]: 7

In [49]:
```
bin(7)
```

Out[49]: '0b111'

Left Shift << (we will gain 0 means add 0)

In [52]:
```
35<<1
```

Out[52]: 70

In [54]:
```
bin(35)
```

Out[54]: '0b100011'

In [56]:
```
10<<1
```

Out[56]: 20

In [58]:
```
20<<2
```

Out[58]: 80

Right Shift >> we will lose the bits means sub of 0

In [66]:
```
bin((10))
```

Out[66]: '0b1010'

In [60]:
```
10 >> 1
```

Out[60]: 5

In [62]:
```
10 >> 2
```

Out[62]: 2

In [64]:
```
20 >> 5
```

Out[64]: 0

In [ ]:
```

```