Related Question: Maximum number that Sum of prices is less than or equal to k

Count number of bits till a number starting from 0.

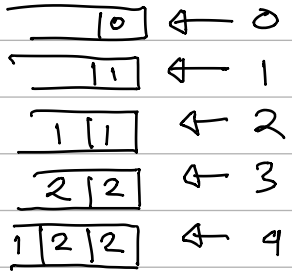
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
count	2	2	3	

for any number K return bit counts.

K=5 ans: 2, 2, 3.

>> intuition :

0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	2	2	
	4/2	4/2	



0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11

getCount(11) → [3+1 | 4 | 4 | 4]
+
getCount(11-8)

```
vector<int> getCount( int num )
{   vector<int> temp (64, 0);
    if( num == 0) return temp;

    int digit = log2(num)
    LL nearestPower = (1LL << digit)

    vector <LL> others = getCount(
                          num - nearestPower)
    bitCount [digit] = (num - nearestP+1)

    for(int i=0 ; i< digits ; i++)
        bitCount [i] = others [i]
                     +
                     nearestP /2 ;

    return bitCount;
}
```

And of all Numbers in a range left to Right

Left 10 : 0 0 1 0 1 0
11 : 0 0 1 0 1 1
12 : 0 0 1 1 0 0
Right 13 : 0 0 1 1 0 1

x: 0 0 1 0 1 → 0 0 1 0 → 0 0 1
y: 0 0 1 1 0 → 0 0 1 1 → 0 0 1
x != y, x != y, x == y

* until x == y there's a 0 hidden in between x to y.

x: 0 1 0 1
y: 0 1 1 1

→

x: 0 1 0 1
0 1 1 0
y 0 1 1 1

hence and of Last unmatched bits are 0.

AND(10..13) = 001000

Approach 2

Redundant to compute 8 & 10

10 1 0 1 0
11 1 0 1 1 12 & 11 = 8 Ans
12 1 1 0 0
13 1 1 0 1 → 13 & 12 = 12

no need to and all number from Right side to left side.
only need to do n & (n-1)
until n <= left

Gray Code

between 2 consecutive number 1 bit difference must be present.

Recursive Soln:

2 Bit: 00 01 11 10

3 Bit: 000 001 011 010
110 111 101 100

000
001
011
010
110
111
101
100

order of 2 bit
reverse of 2 bit