Question: https://leetcode.com/problems/counting-bits/

Okay so here is the problem where need to return an array ar

in which at pos

0: it signifies number of 1 in binary of that pos,i.e binary of 0 is 0, therefore ar[0]=0

1: it signifies number of 1 in binary of that pos,i.e binary of 1 is 1, therefore ar[1]=1

2: it signifies number of 1 in binary of that pos,i.e binary of 2 is 10, therefore ar[2]=1

3: it signifies number of 1 in binary of that pos,i.e binary of 3 is 11, therefore ar[3]=2

and so on....

therefore we can have an approach to find the binary of each number and then again traverse through the binary number to check number of 1's in it

Therefore one loop to traverse i 0 to n, and another to traverse the binary of i

Complexity: O(n log n)

Now if we carefully look at the binary numbers we can make some observations

0:0

1:1

2:10

3:11

4:100

5:101

6:110

7:111

8:1000

there we can see that we get binary number of n by storing (binary of n/2 \*10)+(binaryof n%2)

Thus we can also use this to find number of ones i.e; number of 1's in n=(number of 1's in n/2)+(number of 1's in n%2)

where the base case will be that number of 1 in 0 is 0 and 1 is 1

Using this approach we can solve it in O(n)

Code:  
class Solution {

public int[] countBits(int n) {

int i=2,ar[]=new int[n+1];

ar[0]=0;

if(n==0){

return ar;

}

ar[1]=1;

while(i<=n){

ar[i]=ar[i/2]+ar[i%2];

i++;

}

return ar;

}

}

Github Link :<https://lnkd.in/ecwtJeaz>