**1356. Sort Integers by The Number of 1 Bits: -**

Easy Accepted: 153.1K Submissions: 202.2K Acceptance Rate: 75.7%

You are given an integer array arr. Sort the integers in the array in ascending order by the number of 1's in their binary representation and in case of two or more integers have the same number of 1's you have to sort them in ascending order.

Return *the array after sorting it*.

**Example 1:**

**Input:** arr = [0,1,2,3,4,5,6,7,8]

**Output:** [0,1,2,4,8,3,5,6,7]

**Explantion:** [0] is the only integer with 0 bits.

[1,2,4,8] all have 1 bit.

[3,5,6] have 2 bits.

[7] has 3 bits.

The sorted array by bits is [0,1,2,4,8,3,5,6,7]

**Example 2:**

**Input:** arr = [1024,512,256,128,64,32,16,8,4,2,1]

**Output:** [1,2,4,8,16,32,64,128,256,512,1024]

**Explantion:** All integers have 1 bit in the binary representation, you should just sort them in ascending order.

**Constraints:**

* 1 <= arr.length <= 500
* 0 <= arr[i] <= 104

**Code: -**

bool mycmp(int &a, int &b){

    int x = a, c1 = 0;

    while(x > 0){

        x = x & (x-1);

        ++c1;

    }

    int y = b, c2 = 0;

    while(y > 0){

        y = y & (y-1);

        ++c2;

    }

    if(c1 == c2)

        return a < b;

    return c1 < c2;

}

class Solution {

public:

    vector<int> sortByBits(vector<int>& arr) {

        sort(arr.begin(), arr.end(), mycmp);

        return arr;

    }

};

**T.C: - O(N \* log N)**

**S.C: - O(1)**