1356. Sort Integers by The Number of 1 Bits

```
res = {}

#         for ele in arr:
#             count = self.kernighansAlgo(ele)
#             if count in res:
#                 res[count].append(ele)
#             else:
#                 res[count] = [ele]

#         ans = sorted(res.items(),key=lambda x:x[0])
#         ans2 = []
#         for i,j in ans:
#             ans2 = ans2+sorted(j)
#         return ans2



#     def kernighansAlgo(Self,num):
#         count = 0

#         while num!=0:
#             ans = num & (-num)
#             num = num - ans
#             count = count +1
#         return count
```

Given an integer array `arr`. You have to 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 sorted array*.

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.

Example 3:

Input: arr = [10000,10000]
Output: [10000,10000]

Example 4:

Input: arr = [2,3,5,7,11,13,17,19]
Output: [2,3,5,17,7,11,13,19]

Example 5:

Input: arr = [10,100,1000,10000]
Output: [10,100,10000,1000]

Constraints:

- `1 <= arr.length <= 500`