Problem 338: Counting Bits

Problem Information

Difficulty: Easy
Acceptance Rate: 80.17%
Paid Only: No
Tags: Dynamic Programming, Bit Manipulation

Problem Description

Given an integer `n`, return _an array_`ans` _of length_`n + 1` _such that for each_`i` __(`0 <= i <= n`)_,_`ans[i]`_is the**number of**_`1` _**' s** in the binary representation of _`i`.

Example 1:

Input: n = 2 **Output:** [0,1,1] **Explanation:** 0 --> 0 1 --> 1 2 --> 10

Example 2:

Input: n = 5 **Output:** [0,1,1,2,1,2] **Explanation:** 0 --> 0 1 --> 1 2 --> 10 3 --> 11 4 --> 100 5 --> 101

Constraints:

* `0 <= n <= 105`

Follow up:

* It is very easy to come up with a solution with a runtime of `O(n log n)`. Can you do it in linear time `O(n)` and possibly in a single pass? * Can you do it without using any built-in function (i.e., like `__builtin_popcount` in C++)?

Code Snippets

C++:

```
class Solution {
public:
vector<int> countBits(int n) {

}
};
```

Java:

```
class Solution {
public int[] countBits(int n) {

}
}
```

Python3:

```
class Solution:
def countBits(self, n: int) -> List[int]:
```