困难题：不含连续1的非负整数

题目描述:

给定一个正整数 n，找出小于或等于 n 的非负整数中，其二进制表示不包含 **连续的1** 的个数。

示例 1:

```
输入: 5
输出: 5
解释:
下面是带有相应二进制表示的非负整数<= 5:
0 : 0
1 : 1
2 : 10
3 : 11
4 : 100
5 : 101
其中，只有整数3违反规则（有两个连续的1），其他5个满足规则。
```

题目分析:

- 从我的角度来讲：这道题得用动态规划；动态规划的角度大概可以相当，一个变量就是二进制的每一个位，一个变量就是二进制的每一个位上的取值**0**或者**1**；
- 但是这道题有所限制：一是不能保证在任何条件下，每一位都能取到**0**和**1**，是由条件限制的；而这样的条件限制体现在我的编码上和思考上就有点困难；
- 新思路：对于一个角度变量只有两种取值的情况下，可以借助二叉树来分析一下；

我的代码:

```cpp
class Solution {
public:
    int findIntegers(int n) {
        int num=ceil(log(n+1)/log(2));
```

```
        vector<vector<int>>dp(num,vector<int>(2,0));
        dp[0][0]=dp[0][1]=1;
        for(int i=1;i<num;i++){
            dp[i][0]=dp[i-1][0]+dp[i-1][1];
            dp[i][1]=dp[i-1][0];
        }
        int res=0;
        for(int i=num-1;i>=0;i--){
            if(n>>i&1==1)res+=dp[i][0];
        }
        return res+1;
    }
};
```

二叉树+动态规划+位运算（位运算DP）

dp[t]=dp[t-1]+dp[t-2]的动态转移分析：

https://leetcode-cn.com/problems/non-negative-integers-without-consecutive-ones/solution/bu-han-lian-xu-1de-fei-fu-zheng-shu-by-l-9l86/

```
class Solution {
public:
    int findIntegers(int n) {
        //动态规划
        //dp[t]表示高度为t-1，根节点为0的完全二叉树中符合条件的数量
        vector<int>dp(31);
        dp[0]=dp[1]=1;
        for(int i=2;i<31;i++)dp[i]=dp[i-1]+dp[i-2];
        //逐一数位分析
        int res=0;
        int pre=0;//防止出现连续的两个1
        for(int i=29;i>=0;i--){
            int val=1<<i;
            if((n&val)!=0){
                //说明i的二进制位的第i位为1
                res+=dp[i+1];
                if(pre==1)break;//当出现连续两个一，终止分析
                pre=1;
            }else pre=0;

            if(i==0)res+=1;
        }
        return res;
    }
};
```