1～n 整数中 1 出现的次数

输入一个整数 n，求 1～n 这 n 个整数的十进制表示中 1 出现的次数。

例如，输入 12，1～12 这些整数中包含 1 的数字有 1、10、11 和 12，1 一共出现了 5 次。

示例 1：

输入：n = 12

输出：5

示例 2：

输入：n = 13

输出：6

优化的解法是求出 1~n 数字中每位出现 1 的次数相加。

我们举个例子先：12345

首先个位出现 1 的情况有 1234+1 种，因为只要个位为 1 即可，高位可以随便取，为什么 +1 呢，是因为可以取 0。

十位出现 1 的情况有 1239 + 1 种

同理百位出现 1 的情况有 1299 + 1 种，千位 1999 + 1 种，万位 2345 + 1 种。

那么 1 的总的出现次数为 1235 + 1240 + 1300 + 2000 + 2346 = 8121

我们举得例子中每位都大于 0 ，如果等于 0 的话

12045

第三位为 0，那么第三位出现 1，其他位的取值范围位 [0， 1199]，那么总次数为 1200

综上，我们可以得出一个结论

先定义几个变量，cur 表示当前位，idx 表示当前的位数(从 0 开始算)，left 表示高位的值，right 表示低位的值，

1 2 3 4 5

cur

如果当前位为 1，当前位出现 1 的次数为 left*10^idx + right + 1

如果当前位为 0，当前位出现 1 的次数为 left*(10^idx)

其余情况，当前位出现 1 的次数为 (left+1)*(10^idx)

其实就是相当于除去当前位其他位一共可以组成多大的数字。应该不难理解

最简单的暴力求解会超出时间限制

```
/*class Solution {
public:
    int coutone(int n)
    {
        int sum=0;
        while(n>0)
        {
            if(n%10==1)
                sum++;
             n=n/10;
        }
        return sum;
    }
    int countDigitOne(int n) {
        int res=0;
        for(int i=1;i<=n;i++)
        {
            res+=coutone(i);
        }
        return res;
    }
};*/
class Solution {
public:
    int countDigitOne(int n) {
        int ans = 0;
        int left, right = 0, idx = 0, cur = 0;
        while(n){
            cur = n % 10;
            left = n/10;
            if(cur == 0){
                ans += left*pow(10, idx);
            }else if(cur == 1){
                ans += left*pow(10, idx) + right + 1;
            }else{
                ans += (left + 1)*pow(10, idx);
```

```
            }
            right += cur*pow(10, idx);
            n /= 10;
            idx ++;
        }
        return ans;
    }
};
```