# 整数中1出现的次数（从1到n整数中1出现的次数）

## 题目

[牛客网](https://www.nowcoder.com/practice/bd7f978302044eee894445e244c7eee6?tpId=13&tqId=11184&rp=1&ru=%2Fta%2Fcoding-interviews&qru=%2Fta%2Fcoding-interviews%2Fquestion-ranking&tPage=2)

求出1~13的整数中 1 出现的次数,并算出 100~1300 的整数中1出现的次数？为此他特别数了一下 1~13 中包含1的数字有 1、10、11、12、13 因此共出现 6 次,但是对于后面问题他就没辙了。ACMer 希望你们帮帮他,并把问题更加普遍化,可以很快的求出任意非负整数区间中1出现的次数（从1 到 n 中1出现的次数）。

## 解题思路

1. 假定

$$n=21345$$

1. 将数字分为首位和非首位两个部分
2. 对于首位为 1 的情况，如果首位

$$>1$$

1. 那么

$$sum=sum+10^{len(n)−1}$$

1. ，如果首位

$$=1$$

1. 那么

$$sum=sum+1$$

1. 对于非首位 1，指定其中一位为 1，根据排列组合有

$$10^{len(n)−2}×(len(n)−1)$$

1. 个。那么非首位 1 总共有

$$2×10^{len(n)−2}×(len(n)−1)$$

public int NumberOf1Between1AndN\_Solution(int n) {
 int[] res = {0};
 NumberOf1Between1AndN(res, n);

 return res[0];
}

private void NumberOf1Between1AndN(int[] res, int n) {
 //假设 num=21345
 String num = String.valueOf(n);
 int firstNum = num.charAt(0) - '0';

 if (num.length() == 1) {
 if (firstNum > 0) res[0]++;
 return;
 }

 String nextNum = num.substring(1);
 int nextN = Integer.valueOf(nextNum);

 //数字 10000 ～ 19999 的第一位中的个数
 if (firstNum > 1) {
 res[0] += Math.pow(10, num.length() - 1);
 } else if (firstNum == 1) {
 res[0] += nextN + 1;
 }

 //1346 ～ 21345 除第一位之外的数的个数
 res[0] += firstNum \* (num.length() - 1) \* Math.pow(10, num.length() - 2);

 NumberOf1Between1AndN(res, nextN);
}