1 到 n 中 1 出现的次数

【题目】

给定一个整数 n，返回从 1 到 n 的数字中 1 出现的个数。

例如：

n=5，1~n 为 1，2，3，4，5。那么 1 出现了 1 次，所以返回 1。

n=11，1~n 为 1，2，3，4，5，6，7，8，9，10，11。那么 1 出现的次数为 1（出现 1 次），10（出现 1 次），11（有两个 1，所以出现了 2 次），所以返回 4。

【难度】

校 ★★★☆

【解答】

方法一：容易理解但是复杂度较高的方法，即逐一考查 1~n 的每一个数里有多少个 1。具体请参看如下代码中的 solution1 方法。

```
public int solution1(int num) {
        if (num < 1) {
                return 0;
        }
        int count = 0;
        for (int i = 1; i != num + 1; i++) {
                count += get1Nums(i);
        }
        return count;
}

public int get1Nums(int num) {
        int res = 0;
        while (num != 0) {
                if (num % 10 == 1) {
                        res++;
                }
                num /= 10;
        }
        return res;
}
```

十进制的整数 N 有 $\log N$ 位(以 10 为底)，所以考察一个整数含有多少个 1 的代价是

$O(\log N)$，一共需要考察 N 个数，所以方法一的时间复杂度为 $O(N\log N)$(以 10 为底)。

方法二：不再依次考察每一个数，而是分析 1 出现的规律。

先看 n，如果只有 1 位的情况，因为 1~9 的数中，1 只出现 1 次，所以如果 n 只有 1 位时，返回 1。接下来以 n=114 为例来介绍方法二。先不看 1~14 之间出现了多少个 1，而是先求出 15~114 的数之间一共出现了多少个 1。15~114 之间，哪些数百位上能出现 1 呢？毫无疑问，100~114 这些数百位上才有 1，所以百位上的 1 出现的次数为 15 个，即 114%100+1。15~114 之间，哪些数十位上有 1 呢？110，111，112，113，114，15，16，17，18，19。这些数的十位上才有 1，一共 10 个。15~114 之间，哪些数个位上有 1 呢？101，111，21，31，41，51，61，71，81，91。这些数的个位上才有 1，一共 10 个。

所以，观察发现如下规律：

1. 十位上固定是 1 的话，个位从 0 变到 9 都是可以的。
2. 个位上固定是 1 的话，十位从 0 变到 9 都是可以的。
3. 无非就是最高位取值跟着变化，使构成的数落在 15~114 区间上即可。

所以，15~114 之间的数在十位和个位上的 1 的数量=10+10=20=1×2×10，即(最高位的数字)×(除去最高位后剩下的位数)×(某一位固定是 1 的情况下，剩下的 1 位数都可以从 0 到 9 自由变化，所以是 10 的 1 次方)。这样就求出了 15~114 之间 1 的个数，然后 1~14 的数字出现 1 的个数可以按照如上方式递归求解。

再举一例，n=21345。先不看 1~1345 之间出现了多少个 1，而是先求出 1346~21345 的数之间一共出现了多少个 1。1346~21345 之间，哪些数万位上能出现 1 呢？毫无疑问，10000~19999 这些数万位上都有 1，所以万位上的 1 出现的次数为 10000 个。与上一例不同的是，上一例 n 的最高位是 1，而这里大于 1。如果像上例那样最高位的数字等于 1，那么最高位上 1 的数量=除去最高位后剩下的数+1。而如果像本例那样最高位的数字大于 1，那么最高位上 1 的数量=10000=10^{k-1}（k 为 n 的位数，本例中 k 为 5）。1346~21345 之间，哪些数千位上有 1 呢？在 1346~11345 范围上，千位上固定是 1 的话，百位、十位和个位可自由从 0~9 变换，10^3 个，在 11346~21345 范围上，千位上固定是 1 的话，百位、十位、个位可自由从 0~9 变换，10^3 个，所以有 2×10^3 个千位上是 1。哪些数百位上有 1 呢？在 1346~11345 范围上，百位上固定是 1 的话，千位、十位、个位可自由从 0~9 变换，10^3 个，在 11346~21345 范围上，百位上固定是 1 的话，千位、十位、个位可自由从 0~9 变换，10^3 个，所以有 2×10^3 个百位上是 1。十位和个位也是一样的情况，所以千位、百位、十位、个位是 1 的总数量=$2\times4\times10^3$，即(最高位的数字)×(除去最高位后剩下的位数)×(某一位固定是 1 的情况下，剩下的 3 位数都可以从 0 到 9 自由变化，所以是 10^3)。这样就求出了 1346~21345 之间 1 的个数，

然后 1~1345 的数字上出现 1 的个数可以按照如上方式递归求解。

具体过程请参看如下代码中的 solution2 方法。

```
public int solution2(int num) {
        if (num < 1) {
                return 0;
        }
        int len = getLenOfNum(num);
        if (len == 1) {
                return 1;
        }
        int tmp1 = powerBaseOf10(len - 1);
        int first = num / tmp1;
        int firstOneNum = first == 1 ? num % tmp1 + 1 : tmp1;
        int otherOneNum = first * (len - 1) * (tmp1 / 10);
        return firstOneNum + otherOneNum + solution2(num % tmp1);
}

public int getLenOfNum(int num) {
        int len = 0;
        while (num != 0) {
                len++;
                num /= 10;
        }
        return len;
}

public int powerBaseOf10(int base) {
        return (int) Math.pow(10, base);
}
```

仅通过分析如上代码就可以知道，n 一共有多少位，递归函数最多就会被调用多少次，即 $\log N$ 次。在递归函数内部 getLenOfNum 方法和 powerBaseOf10 方法的复杂度分别为 $O(\log N)$和 $O(\log(\log N))$。求一个数的 A 次方的问题在系统内部实现的复杂度为 $O(\log A)$，A 为 N 的位数($A=\log N$)，所以 powerBaseOf10 方法的时间复杂度为 $O(\log(\log N))$。所以方法二的总时间复杂度为 $O(\log N\times\log N)$。

从 *N* 个数中等概率打印 *M* 个数

【题目】

给定一个长度为 N 且没有重复元素的数组 arr 和一个整数 n，实现函数等概率随机打印 arr 中的 M 个数。