C2-C题题解

> 作者：余绍函

题目描述

对于一个数组$A=[a_1,a_2,\cdots,a_n]$，定义

$$f(A)=\sum_{i=1}^{n} order(a_i)*a_i$$

其中

$$order(a_i)=1+\sum_{a_j<a_i,1\le j\le n} 1$$

现给出一个随机数生成器如下：

```
#include <stdio.h>
#define N 5000005

int nextRand() {
    static unsigned int rnd_num = 0x80000001;
    static int mod = 1e5 + 3;

    rnd_num ^= rnd_num >> 10;
    rnd_num ^= rnd_num << 9;
    rnd_num ^= rnd_num >> 25;
    return rnd_num % mod;
}

int a[N];

int main() {
    int tt;
    scanf("%d", &tt);
    while (tt--) {
        int n;
        scanf("%d", &n);
        for (int i = 1; i <= n; i++) {
            a[i] = nextRand();
```

```
        }
    }
    return 0;
}
```

对于每组测试数据，你需要使用上述随机数生成器生成 n 个数 $a_1, a_2, \cdots, a_n$ 记 $A = [a_1, a_2, \cdots, a_n]$，你需要求出 $f(A)$ 的值。
注意：对于一个测试点有多组测试数据，在每组测试数据之间不需要重置随机数生成器，在后一组测试中，只需要继续调用随机数生成器直接得到随机数即可。

输入

第一行一个正整数 T (1≤ T≤ 10) 表示数据组数。

接下来T组数据：

一行一个正整数 $n(1 \leq n \leq 5 * 10^6)$ 表示数组A的长度。

输出

对于每组数据，输出一行一个正整数表示$f(A)$的值。

输入样例

```
3
11
101
1009
```

输出样例

```
3832359
353053098
34254309646
```

提示

一个数 a_i 的 $order$ 定义为 **数组内全部元素** 比它小的数的个数加一，例如对于 $A = [1, 1, 2, 3, 3, 3, 4]$ 有：$order(1) = 1,\ order(2) = 3, order(3) = 4, order(4) = 7$

因此

$$f(A) = 1 \cdot 1 + 1 \cdot 1 + 3 \cdot 2 + 4 \cdot 3 + 4 \cdot 3 + 4 \cdot 3 + 7 \cdot 4 = 72$$

另外，提醒大家多多注意代码中 mod 的范围。

解题思路

因为这道题需要求 $order(a_i)$，所以这道题需要给生成的随机数进行排序。由于时间限制比较严格，同时数据组数多、数据量大，如果使用 `qsort` 或者其余时间复杂度为 $O(nlogn)$ 的排序函数很容易TLE。不难发现，生成随机数的范围是 $[0, 100002]$ ，范围比较小。因此我们可以采用的计数排序法。

计数排序

计数排序假设n个输人元素中的每一个都是在0到区间内的一个整数，其中为某个整数。当 $k = O(n)$ 时，排序的运行时间为 $\Theta(n)$ 。

计数排序的基本思想是:对每一个输入元素，确定小于的元素个数。利用这一信息，就可以直接把放到它在输出数组中的位置上了。例如，如果有 17 个元素小于 x，则 x 就应该在第18个输出位置上。当有几个元素相同时，这一方案要略做修改。因为不能把它们放在同一个输出位置上。

伪代码:

```
COUNTING-SORT(A, B, k)
 1  let C[0..k] be a new array
 2  for i = 0 to k
 3      C[i] = 0
 4  for j = 1 to A.length
 5      C[A[j]]=C[A[j]]+1
 6  //C[i] now contains the number of elements equal to i.
 7  for i = 1 to k
 8      C[i]=C[i]+C[i-1]
 9  //C[i] now contains the number of elements less than or equal to i.
10  for j = A.length downto 1
11      B[C[A[j]]]=A[j]
12      C[A[j]]=C[A[j]]-1
```

代码实现

```
#include <stdio.h>
#define N 5000005

int nextRand() {
    static unsigned int rnd_num = 0x80000001;
    static int mod = 1e5 + 3;
    rnd_num ^= rnd_num >> 10;
    rnd_num ^= rnd_num << 9;
    rnd_num ^= rnd_num >> 25;
    return rnd_num % mod;
}

int array[N], barrel[100010], last_count;
long long f;

int main() {
    int tt;
    scanf("%d", &tt);
    while (tt--) {
        int n;
        scanf("%d", &n);
        for(int i = 0; i <= 100003; i++) // 初始化用于计数的数组
            barrel[i] = 0;
        for (int i = 1; i <= n; i++) { // 初始化随机数并计数
            array[i] = nextRand();
            barrel[array[i]]++;
        }
        f = last_count = 0;
        for(int i = 0; i <= 100003; i++) // 在进行计数排序的统计过程中直接计算
f(A)
        {
            if(barrel[i] > 0)
            {
                f += (long long)barrel[i] * (long long)(last_count + 1) *
(long long)i;
                last_count += barrel[i];
            }
        }
        printf("%lld\n", f);
    }
```

```
    return 0;
}
```

思考与拓展

计数排序的使用条件

- 给整数排序
- 需要排序的数据范围较小
- 拥有充足的空间（空间复杂度与需要排序的数据范围相关）

带有负数时计数排序的使用

假设需要排序的数组是$A=[a_1,a_2,\dots,a_n]$，a_i的范围是 $[l,r](l<0<r)$，我们可以把计数数组 $count$ 的大小设定为 $r-l+1$，在计数时执行`count[A[index] - l]++`，在遍历计数数组取`count[i]+l`，其中 $i\in[0,r-l]$ 。

一些可以使用计数排序的题目

leetcode 83
leetcode 147
leetcode 148