CSP 模拟赛

【数据删除】

2019 年 10 月 17 日

题目名称	控制	收容	保护
题目类型	传统型	传统型	传统型
目录	`contain`	`secure`	`protect`
可执行文件名	`contain`	`secure`	`protect`
输入文件名	`contain.in`	`secure.in`	`protect.in`
输出文件名	`contain.out`	`secure.out`	`protect.out`
每个测试点时限	1.0 秒	1.0 秒	1.0 秒
内存限制	512MB	512MB	512MB
测试点数目	10	10	10
测试点是否等分	是	是	是

提交源程序文件名

对于 C++ 语言	`contain.cpp`	`secure.cpp`	`protect.cpp`

编译选项

对于 C++ 语言	`-O2 -std=c++11`	`-O2 -std=c++11`	`-O2 -std=c++11`

注意事项

- 文件名（包括程序名和输入输出文件名）必须使用英文小写。
- 结果比较方式为忽略行末空格、文末回车后的全文比较。
- C++ 中函数 `main()` 的返回值类型必须是 `int`，值为 0。
- 评测时栈空间大小限制等同于内存限制。
- 题目名称与题目描述无关。

1 控制

1.1 题目描述

小 Z 惹上麻烦了。

事情的经过是这样的：猫老师与鸡老师约好在《大颗寿司三》中激情对射，为此猫老师准备了 n 种不同类型的装备，第 i 种装备有 a_i 件。在对战中，猫老师会携带每种装备恰好一件。

为了简化题意，我们认为每件装备都只有一个唯一的属性值：魔力值。根据套路，猫老师的战斗力是与他所携带的所有装备的魔力值之和正相关的，也即，携带的装备的魔力值之和越大，猫老师的战斗力越强，反之则越弱。

为了能够更加偷税地与鸡老师激情对射，猫老师把所有可行的携带装备的方案按照魔力值之和从大到小排序并取出前 m 种方案作为备选方案。他把这 m 种备选方案交给了小 Z 代为保管。

然而由于【数据删除】，小 Z 把这 m 种备选方案弄丢了。小 Z 企图跑路，然而还是很快就被猫老师抓了回来。猫老师对小 Z 说：“这是你的问题，你必须要解决。”当然猫老师也不会太为难小 Z，他只需要小 Z 告诉他这 m 种备选方案的魔力值之和各是多少。小 Z 很无奈，他现在想要寻求你的帮助。

1.2 输入格式

从文件 `contain.in` 中读入数据。

输入数据第一行包含两个正整数 n, m，意义见题目描述

接下来 n 行，每行第一个正整数 a_i，接下来 a_i 个正整数 $magic_{i,j}$ 表示第 j 件 i 类装备的魔力值。

1.3 输出格式

输出到文件 `contain.out` 中。

为避免输出过量，记魔力值之和第 i 大的方案的魔力值之和为 ans_i，你只需要输出 $\sum_{i=1}^{m} ans_i \times 23333^{m-i}$ 对 20190816170251 取模的结果即可。

1.4 样例 1 输入

```
3 10
4 19 26 8 17
3 14 19 5
3 14 19 36
```

1.5 样例 1 输出

```
7760814397409
```

1.6 样例 1 解释

$ans = \{81, 76, 74, 72, 69, 67, 67, 64, 63, 60\}$.

1.7 样例 2

见选手目录下的 `contain/ex_contain2.in` 和 `contain/ex_contain2.ans`。

1.8 数据范围

对于 30% 的数据，$n = 2$。
对于 50% 的数据，$2 \leq n \leq 10$。
对于 80% 的数据，$2 \leq n \leq 150$。
对于 100% 的数据，$2 \leq n \leq 10^5, 1 \leq m, \sum_{i=1}^{n} a_i \leq 3 \times 10^5, 1 \leq magic_{i,j} \leq 10^9$。

2 收容

2.1 题目描述

小 S 手中有一个很大很大的数 S，这个数究竟有多大呢？小 S 给出了一个长度为 n 的数组 $\{a_i\}$，并告诉你他手中的这个数 S 满足

$$S = \prod_{i=1}^{n} a_i!$$

其中，$x!$ 表示 x 的阶乘，即 $\prod_{i=1}^{x} i$。

小 S 对手中的这个数 S 很满意，他突发奇想，能不能换一种方式来表示手中的这个 S 呢？具体地，小 S 希望能够找到一个正整数 k 以及 m 个正整数二元组 (b_i, c_i)，满足

$$S = k \prod_{i=1}^{m} (b_i!)^{c_i}$$

其中 $\{b_i\}$ 满足单调递减，也即，$b_1 > b_2 > ... > b_{m-1} > b_m$。

小 S 发现这样的表示方式有很多很多种，他觉得只需要找出自己喜欢的任意一种就可以了，不料这时候 Itst 进来了，他对小 S 说：“我不要你觉得，我要我觉得。我觉得在你求出的表示方式中，序列 $\{b_1, c_1, b_2, c_2, ..., b_m, c_m\}$ 的字典序应该是所有表示方式中最大的，你不要说了，都听我的！”

小 S 很无奈，他只好按照 Itst 的要求，求出所有表示方式中，序列 $\{b_1, c_1, b_2, c_2, ..., b_m, c_m\}$ 的字典序最大的表示方式。现在他希望你能够提供帮助。

对于一个长度为 n 的序列 $\{a_i\}$ 和一个长度为 m 的序列 $\{b_i\}$，定义序列 $\{a_i\}$ 的字典序大于序列 $\{b_i\}$ 的字典序，当且仅当存在一个 $p \in [1, \min(n, m)]$，满足 $\forall i \in [1, p-1], a_i = b_i$，且 $a_p > b_p$，或不存在这样的 p 且 $n > m$。

2.2 输入格式

从文件 `secure.in` 中读入数据。

输入数据的第一行包含一个正整数 n。

第二行包含 n 个正整数 a_i。

2.3 输出格式

输出到文件 `secure.out` 中。

输出包含 $m+1$ 行，第一行一个正整数 m。

接下来 m 行，每行两个正整数 b_i, c_i。

2.4 样例 1 输入

```
5
6 7 8 9 10
```

2.5 样例 1 输出

```
2
10 3
4 2
```

2.6 样例 2 输入

```
4
114 514 1919 810
```

2.7 样例 2 输出

```
6
1930 1
640 1
520 1
172 1
73 1
21 1
```

2.8 样例 3

见选手目录下的 `secure/ex_secure3.in` 和 `secure/ex_secure3.ans`。

2.9 数据范围

对于 20% 的数据，$1 \leq n, a_i \leq 20$。
对于 50% 的数据，$1 \leq n, a_i \leq 5000$。
对于 70% 的数据，$1 \leq n, a_i \leq 7 \times 10^4$。
对于 100% 的数据，$1 \leq n, a_i \leq 3 \times 10^5$。

3 保护

3.1 题目描述

小 Y 不知道从哪里弄来了 n 个椰子。

小 Y 想要喝椰汁，但前提是需要先把椰子砸开（雾）。每个椰子都有一个硬度值 a_i，硬度值越大表示这个椰子越难以被砸开。

现在小 Y 需要砸开 m 个椰子，由于【数据删除】，所有椰子在被砸开后都是一样的，因此小 Y 只需要砸开硬度值最小的那 m 个椰子就可以以最少的砸击次数达成目标了。

然而就在小 Y 正准备开始砸椰子时，由于受到了来自球爷的神秘力量的影响，n 个椰子被随机打乱了。悲催的是，小 Y 不知道随机打乱后每个椰子的硬度值是多少了。现在小 Y 想要知道，如果他使用最优的砸椰子策略，在最坏情况下需要砸多少次才能砸开 m 个椰子。

3.2 输入格式

从文件 `protect.in` 中读入数据。

单个测试点包含多组测试数据，第一行一个正整数 T 表示测试数据组数。

对于每组数据，第一行包含两个正整数 n, m，第二行包含 n 个正整数 a_i。

3.3 输出格式

输出到文件 `protect.out` 中。

输出共 T 行，对于每组测试数据，输出一行一个正整数表示答案。

3.4 样例 1 输入

```
2
2 1
50 55
2 1
40 100
```

3.5 样例 1 输出

```
55
80
```

3.6 样例 1 解释

对于第一组数据，显然最优策略是对着一个椰子砸直到其裂开，因此在最坏情况下需要砸 55 次。

对于第二组数据，最优策略是对着一个椰子连砸 40 次，若没有裂开则改去砸另一个椰子，最坏情况下需要 80 次砸击。可以证明不存在比 80 次更优的最优策略。

3.7 样例 2

见选手目录下的 `protect/ex_protect2.in` 和 `protect/ex_protect2.ans`。

3.8 数据范围

对于 20% 的数据，$n, a_i \leq 5$。
对于 40% 的数据，$n, a_i \leq 100$。
对于 60% 的数据，$n, a_i \leq 300$。
对于另外 10% 的数据，$m = 1$。
对于另外 10% 的数据，$\forall i \in [1, n], a_i = i$。
对于 100% 的数据，$T \leq 5, 1 \leq m \leq n \leq 2000, 1 \leq a_i \leq 10^6$。