10.18 模拟赛

StudyingFather

2020 年 10 月 17 日

题目名称	馒头	求和	装置
题目类型	传统	传统	传统
附件目录	`bread`	`sum`	`device`
时间限制	1.0 秒	2.0 秒	4.0 秒
内存限制	256 MiB	512 MiB	512 MiB
子任务数目	3	10	8
子任务是否等分	否	是	否

注意事项：

1. 所有题目均采用标准输入输出。
2. 题目栈空间限制和内存限制一致。
3. 评测在 NovaOJ 上进行，比赛采用 OI 赛制，即每道题取最后一次提交计分。
4. 对于采用子任务捆绑测试的题目，你在该题上的得分等于各子任务的得分之和，而各子任务的得分，等于该子任务下每个测试点的最低得分。
5. 每道题目的样例文件和其他资源文件已随本 PDF 一并下发。请做好备份，以防误删。
6. 请各位合理安排比赛时间，充分利用每道题目给出的部分分，达到让自己的总分最大化的目标。
7. 赛后可以在组题人的洛谷博客[1]上找到题解。

[1] `https://studyingfather.blog.luogu.org/simulation-contests-log`

1 馒头

1.1 题目描述

有 M 种互不相同的馒头各一个，第 i 个馒头卖 P_i 元。

有 N 个包装盒，第 j 个包装盒最多能装 C_j 个馒头，买第 j 个包装盒的花费为 E_j 元。要求只能将一些馒头放进包装盒中打包出售，不能零售，当然也可以不出售某些馒头（卖剩的馒头被出题人吃了，出题人还吃得津津有味）。售出一盒馒头得到的利润为盒内所有馒头的价格减去包装盒的价格。

现在小 X 准备买下（这 N 个包装盒）其中的一些包装盒（也可以不买，还可以全买），将馒头打包出售，求小 X 最大可能利润。

1.2 输入格式

第一行两个正整数 M, N，意义如题目描述；

接下来 M 行，每行一个正整数 P_i，表示第 i 个馒头的价格；

接下来 N 行，每行两个正整数 C_j, E_j，表示第 j 个包装盒最多能装 C_j 个馒头，花费 E_j 元。

1.3 输出格式

一行一个整数，表示最大可能利润。

1.4 样例

1.4.1 样例输入 1

```
4 3
180
160
170
190
2 100
3 120
4 250
```

1.4.2 样例输出 1

```
480
```

1.4.3 样例解释 1

在本例中，我们选择第一、第二个包装盒，第一个包装盒装第 $1, 2$ 个馒头，第二个包装盒装第 $3, 4$ 个馒头。第一盒馒头的利润是 $180 + 160 - 100 = 240$ 元，第二盒馒头的利润是 $170 + 190 - 120 = 240$ 元，因此总利润为 $240 + 240 = 480$ 元。

1.4.4 样例输入 2

```
2 2
1000
2000
1 6666
1 7777
```

1.4.5 样例输出 2

```
0
```

1.4.6 样例解释 2

在本例中，为了最大化利益，最好不买包装盒（也就是不卖馒头）。

1.4.7 样例输入 3

```
10 4
200
250
300
300
350
400
500
300
250
200
3 1400
2 500
2 600
1 900
```

1.4.8 样例输出 3

```
450
```

1.4.9 样例 4

见下发文件夹下的 *bread/sample-04.in* 和 *bread/sample-04.out*。

1.5 子任务

对于全部数据，$1 \leq M \leq 10^4, 1 \leq N \leq 500, 1 \leq P_i, C_j, E_j \leq 10^4$。
详细子任务及数据满足条件如下表：

子任务编号	约束	分值
1	$N \leq 10$	25
2	$C_j \leq 10$	35
3	无附加限制	40

2 求和

2.1 题目描述

小 Y 对树上的求和非常感兴趣。他生成了一棵有根树，并且希望多次询问这棵树上一段路径上所有节点深度的 k 次方和，而且每次的 k 可能是不同的。此处节点深度的定义是这个节点到根的路径上的边数。

他把这个问题交给了你，你能帮他解决吗？

2.2 输入格式

第一行包含一个正整数 n，表示树的节点数。

之后 $n-1$ 行每行两个空格隔开的正整数 i,j，表示树上的一条连接点 i 和点 j 的边。

之后一行一个正整数 m，表示询问的数量。

之后每行三个空格隔开的正整数 i,j,k，表示询问从点 i 到点 j 的路径上所有节点深度的 k 次方和。由于这个结果可能非常大，输出其对 998 244 353 取模的结果。

树的节点从 1 开始标号，其中 1 号节点为树的根。

2.3 输出格式

对于每组数据输出一行一个正整数表示取模后的结果。

2.4 样例

2.4.1 样例输入

```
5
1 2
1 3
2 4
2 5
2
1 4 5
5 4 45
```

2.4.2 样例输出

```
33
503245989
```

2.4.3 样例解释

以下用 $d(i)$ 表示第 i 个节点的深度。

对于样例中的树，有 $d(1)=0, d(2)=1, d(3)=1, d(4)=2, d(5)=2$。

因此第一个询问答案为 $(2^5+1^5+0^5) \bmod 998\,244\,353 = 33$。

而第二个询问答案为 $(2^{45}+1^{45}+2^{45}) \bmod 998\,244\,353 = 503\,245\,989$。

2.5 子任务

- 对于 30% 的数据，$1 \leq n, m \leq 100$；
- 对于 60% 的数据，$1 \leq n, m \leq 1000$；
- 对于 100% 的数据，$1 \leq n, m \leq 3 \times 10^5, 1 \leq k \leq 50$。

3 装置

3.1 题目描述

小 Z 发现古代文明留下了一种奇怪的装置。该装置包含两个屏幕，分别显示两个整数 x 和 y。

经过研究，小 Z 对该装置得出了一个结论：该装置是一个特殊的时钟，它从过去的某个时间点开始测量经过的时刻数 t，但该装置的创造者却将 t 用奇怪的方式显示出来。若从该装置开始测量到现在所经过的时刻数为 t，装置会显示两个整数：$x = ((t + \lfloor \frac{t}{B} \rfloor) \bmod A)$，与 $y = (t \bmod B)$。这里 $\lfloor x \rfloor$ 是下取整函数，表示小于或等于 x 的最大整数。

小 Z 通过进一步研究还发现，该装置的屏幕无法一直工作。实际上，该装置的屏幕只在 n 个连续的时间区间段中能正常工作。第 i 个时间段从时刻 l_i 到时刻 r_i。现在科学家想要知道有多少个不同的数对 (x, y) 能够在该装置工作时被显示出来。

两个数对 (x_1, y_1) 和 (x_2, y_2) 不同当且仅当 $x_1 \neq x_2$ 或 $y_1 \neq y_2$。

3.2 输入格式

第一行包含三个整数 n, A 与 B。

接下来 n 行每行两个整数 l_i, r_i，表示装置可以工作的第 i 个时间区间。

3.3 输出格式

输出一行一个整数表示问题的答案。

3.4 样例

3.4.1 样例输入 1

```
3 3 3
4 4
7 9
17 18
```

3.4.2 样例输出 1

```
4
```

3.4.3 样例解释 1

对于第一个样例，装置屏幕将显示如下这些数对。

t	(x, y)
4	$(2, 1)$
7	$(0, 1)$
8	$(1, 2)$
9	$(0, 0)$
17	$(1, 2)$
18	$(0, 0)$

共有四个不同的数对：$(0,0),(0,1),(1,2),(2,1)$。

3.4.4 样例输入 2

```
3 5 10
1 20
50 68
89 98
```

3.4.5 样例输出 2

```
31
```

3.4.6 样例输入 3

```
2 16 13
2 5
18 18
```

3.4.7 样例输出 3

```
5
```

3.4.8 样例 4

见下发文件夹下的 *device/sample-04.in* 和 *device/sample-04.out*。
该样例满足子任务 6 的限制。

3.4.9 样例 5

见下发文件夹下的 *device/sample-05.in* 和 *device/sample-05.out*。
该样例满足子任务 8 的限制。

3.5 提示

在 C++ 中，`long long` 为 64 位带符号整数，可以存储 $-2^{63} \sim 2^{63}-1$ 范围内的整数。

在本题中,你可能需要位宽更大的整数类型。C++ 中提供了 128 位的带符号整数 `__int128`，您可以使用该类型来存储更大的整数。

需要注意的是，`__int128` 只能在 64 位的编译器上进行编译。NovaOJ 采用 64 位编译器编译你的程序，因此您在提交时可以放心使用 `__int128` 类型。

另外，`__int128` 的输入输出并不能直接使用 `scanf`/`printf` 或 `cin`/`cout`。如果你需要输入或输出 `__int128` 类型的整数，我们在题目附件中提供了一份可以实现 `__int128` 输入输出的代码（见下发文件夹下的 *device/io.cpp*），你可以根据需要使用。

3.6 子任务

对于全部数据，$1 \le n \le 10^6, 1 \le A, B \le 10^{18}, 0 \le l_i \le r_i \le 10^{18}, r_i < l_{i+1}$。
令 $S = \sum_{i=1}^{n}(r_i - l_i + 1)$ 与 $L = \max_{i=1}^{n}(r_i - l_i + 1)$。
详细子任务附加限制与分值如下表。

子任务编号	约束	分值
1	$S \le 10^6$	10
2	$n = 1$	5
3	$A \cdot B \le 10^6$	5
4	$B = 1$	5
5	$B \le 3$	5
6	$B \le 10^6$	20
7	$L \le B$	20
8	无附加约束	30