矿石

众所周知，九条可怜家里有矿。

你可以把可怜家的矿场抽象成一条数轴。可怜家有 n 种矿，第 i 种矿可以从 $[l_i, r_i]$ 中的任意位置开采得到。

这个暑假， 地理老师给了可怜一个列表：可怜的暑假作业就是收集齐这些矿石。为了保证可怜的安全，可怜的爸爸选定了 m 个相对安全的采矿点，第 i 个采矿点的坐标为 a_i。可怜只能选择其中一个采矿点开采她需要的矿石。

可怜是一个马虎的女孩子。暑假刚开始没多久，可怜就把老师的列表弄丢了。唯一的线索是，列表上的所有矿石都是可怜家有的：一共有 $2^n - 1$ 种可能的列表。

可怜现在想要知道，在所有的可能的任务列表中，有多少种是她能够在某一个安全的采矿点完全收集齐的。

输入格式

第一行两个整数 n, m。

接下来 n 行每行两个整数 l_i, r_i，接着 m 行每行一个整数 a_i。表示每一种矿出现的位置以及安全采矿点的坐标。

输出格式

输出一行一个整数，表示满足条件的列表数量。答案可能很大，你只需要输出对 998244353 取模后的结果。

样例输入

```
3 2
7 11
1 5
3 8
4
7
```

样例输出

```
5
```

数据范围

对于 20% 的数据，$n, m \leq 20$.

对于 40% 的数据，$n \leq 20$.

对于 60% 的数据，$n, m \leq 1000$.

对于 100% 的数据，$n \leq 10^5, 1 \leq l_i, r_i, a_i \leq 10^9$.

括号序列

可怜不喜欢括号序列，但是她发现总是有人喜欢出括号序列的题。

为了让全世界都能感受到她的痛苦，她想要写一个转换器：它能把普通的小写字符串转换成长度相同的合法的括号序列。

在可怜的构思中，这样的转换器需要满足如下两个条件：

1. 结果的括号序列必须要是合法的，即左右括号必须要是相匹配的。
2. 对于一堆相匹配的左右括号，他们所在的位置原来的小写字母必须相同。

举例来说，对于字符串 `aabaab`，`()(())` 就是一个合法的答案，而 `()()()` 不满足第二个条件，`((()))`不满足第一个条件。

可怜发现对于一个小写字符串，有时候有很多满足条件的括号序列，有些时候一个都没有。

于是可怜给出了一个小写字符串，她想让你帮她算一下，有多少不同的满足条件的括号序列。

输入格式

输入一行一个小写字符串 s。

输出格式

输出一行一个整数，表示满足条件的括号序列的数量，对 998244353 取模。

样例输入

```
aabaab
abcabcabc
aabbcc
```

样例输出

```
4
0
6
```

注意上面有三组数据，实际的测试数据中只会有一组。

数据范围

对于 30% 的数据，$|s| \leq 20$。

对于 60% 的数据，$|s| \leq 1000$。

对于 100% 的数据，$|s| \leq 10^6$。

传送带

众所周知，九条可怜家里有矿。

为了解决矿工们的伙食问题，可怜在家里建了一条长度为 n 的传送带。每天开饭时，传送带上什么都没有。传送带每一秒会向右移动一格，同时一格盘子会出现在位置 1 上。即第一秒开始时，传送带的最左端会有一个盘子，第 N 秒开始的时候，传送带的最左侧 N 个位置上会有盘子。

矿工们乐于制作食物，更乐于和别人分享自己的食物。矿工们会提出很多要求，每一个要求形如：我要把 p_i 单位的食物从 l_i 运送到 $r_i(l_i < r_i)$。

如果把食物放在 l_i 处，那么在 $r_i - l_i$ 秒之后，食物便会到达 r_i。但是事情通常没有那么简单。作为传送带的管理者，每一秒，你可以对传送带第 i 个位置做的操作有两种：

1. 如果第 i 个位置有空盘，那么你可以把一单位食物放到这个空盘中。
2. 如果第 i 个位置有装了食物的盘子，你可以把这个食物取出。（盘子变空）

你每一秒可以对任意多个位置同时进行操作，但是对每一个位置最多只能操作一次。举例来说，你可以在一秒内取出 $1, 3, 5$ 位置上的食物，在 $2, 4, 6$ 位置下放下食物。但是你不能在一秒内先取出 1 位置上的食物再在 1 位置上放下食物。

在运输过程中，你可以多次取出或放下同一个食物，例如有一单位食物是从 1 运送到 3 的，你可以在 2 先把它取出来，然后在合适的时机再把它放回到传送带上。但是你不能搞混食物的目的地，即如果有两个要求，分别是把一单位的食物从 1 运到 4，把一单位的食物从 2 运到 3，而你把 1 出发的食物运到了 3，2 出发的食物运到了 4，尽管 $3, 4$ 都收到了食物，但是这样的行为还是不被允许的。

现在你想要最优化运输的过程，即求解在最优操作策略下，开饭几秒后所有的要求都会被完成。

矿工总是会有新的要求提出，你需要实时地求解这个过程：按照顺序，矿工一共提出了 m 个要求，你需要对每一个 $i \in [1, m]$，求出只考虑前 i 个要求的答案。

输入格式

第一行输入两个整数 n, m，表示传送带长度以及矿工的要求数。

接下来 m 行每行三个整数 l_i, r_i, p_i，表示矿工想要把 p_i 单位的食物从 l_i 运到 r_i。

输出格式

输出 m 行每行一个整数，表示只考虑前 i 个要求的最少完成时间。

样例输入一

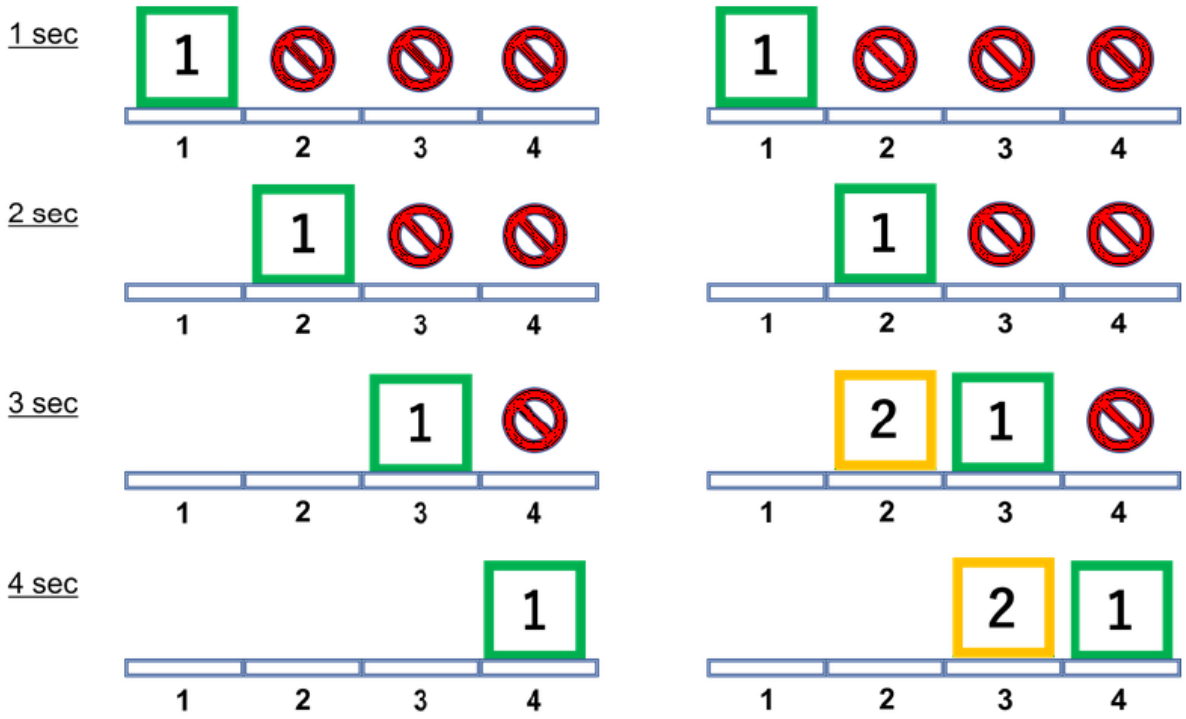
```
5 2
1 4 1
2 3 1
```

样例输出一

```
4
4
```

样例解释一

下图给出了两组询问的一种最优操作序列（左侧为第一组，右侧为第二组）：



样例输入二

```
10 4
3 5 2
5 7 5
8 9 2
1 7 5
```

样例输出二

```
6
11
11
16
```

数据范围

对于 20% 的数据，$n, m \leq 6, p_i = 1$.

对于 60% 的数据，$n, m \leq 2000$.

对于 100% 的数据，$n, m \leq 10^5, 1 \leq l_i < r_i \leq n, 1 \leq p_i \leq 10^9$.