

2 树状数组

2.1 题目描述

漆黑的晚上，九条可怜躺在床上辗转反侧。难以入眠的她想起了若干年前她的一次悲惨的OI比赛经历。那是一道基础的树状数组题。

给出一个长度为 n 的数组 A ，初始值都为 0，接下来进行 m 次操作，操作有两种：

- 1 x ，表示将 A_x 变成 $(A_x + 1) \bmod 2$ 。
- 2 $l\ r$ ，表示询问 $(\sum_{i=l}^r A_i) \bmod 2$ 。

尽管那个时候的可怜非常的 simple，但是她还是发现这题可以用树状数组做。当时非常 young 的她写了如下的算法：

```
1: function ADD( $x$ )
2:   while  $x > 0$  do
3:      $A_x \leftarrow (A_x + 1) \bmod 2$ 
4:      $x \leftarrow x - \text{lowbit}(x)$ 
5:   end while
6: end function
7:
8: function FIND( $x$ )
9:   if  $x == 0$  then
10:    return 0
11:   end if
12:    $ans \leftarrow 0$ 
13:   while  $x \leq n$  do
14:      $ans \leftarrow (ans + A_x) \bmod 2$ 
15:      $x \leftarrow x + \text{lowbit}(x)$ 
16:   end while
17:   return  $ans$ 
18: end function
19:
20: function QUERY( $l, r$ )
21:    $ansl \leftarrow \text{FIND}(l - 1)$ 
22:    $ansr \leftarrow \text{FIND}(r)$ 
23:   return  $(ansr - ansl + 2) \bmod 2$ 
24: end function
```

其中 $\text{lowbit}(x)$ 表示数字 x 最低的非 0 二进制位，例如 $\text{lowbit}(5) = 1, \text{lowbit}(12) = 4$ 。进行第一类操作的时候就调用 $\text{Add}(x)$ ，第二类操作的时候答案就是 $\text{Query}(l, r)$ 。

如果你对树状数组比较熟悉，不难发现可怜把树状数组写错了：**Add 和 Find 中 x 变化的方向反了**。因此这个程序在最终测试时华丽的爆 0 了。

然而奇怪的是，在当时，这个程序通过了出题人给出的大样例——这也是可怜没有进行对拍的原因。

现在，可怜想要算一下，这个程序回答对每一个询问的概率是多少，这样她就可以再次地感受到自己是一个多么非的人了。然而时间已经过去了很多年，即使是可怜也没有办法完全回忆起当时的大样例。幸运的是，她回忆起了大部分内容，唯一遗忘的是每一次第一类操作的 x 的值，因此她假定这次操作的 x 是在 $[l_i, r_i]$ 范围内 **等概率随机** 的。

具体来说，可怜给出了一个长度为 n 的数组 A ，初始为 0，接下来进行了 m 次操作：

- $1\ l\ r$ ，表示在区间 $[l, r]$ 中等概率选取一个 x 并执行 $\text{Add}(x)$ 。
- $2\ l\ r$ ，表示询问执行 $\text{Query}(l, r)$ 得到的结果是正确的概率是多少。

2.2 输入格式

第一行输入两个整数 n, m 。

接下来 m 行每行描述一个操作，格式如题目中所示。

2.3 输出格式

对于每组询问，输出一个整数表示答案。如果答案化为最简分数后形如 $\frac{x}{y}$ ，那么你只需要输出 $x \times y^{-1} \bmod 998244353$ 后的值。（即输出答案模 998244353）。

2.4 样例输入

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

2.5 样例输出

```
1
0
665496236
```

2.6 样例解释

在进行完 $\text{Add}(3)$ 之后， A 数组变成了 $[0, 1, 1, 0, 0]$ 。所以前两次询问可怜的程序答案都是 1，因此第一次询问可怜一定正确，第二次询问可怜一定错误。

2.7 数据范围与约定

测试点编号	n	m	其他约定
1	≤ 5	≤ 10	无
2	≤ 50	≤ 50	
3			
4	≤ 3000	≤ 3000	
5			
6	$\leq 10^5$	$\leq 10^5$	所有询问都在修改后
7			无
8			
9			
10			

对于 100% 的数据，保证 $1 \leq l \leq r \leq n$ 。