2017 年全国青少年信息学奥林匹克

浙江省队选拔赛第一试

竞赛时间：3 月 23 日 8:00 – 13:00

题目名称	仙人掌	树状数组	多项式
目录	cactus	bit	poly
可执行文件名	cactus	bit	poly
输入文件名	cactus.in	bit.in	poly.in
输出文件名	cactus.out	bit.out	poly.out
每个测试点时限	1s	4s	3s
内存限制	512MB	512MB	512MB
测试点数目	10	10	10
每个测试点分值	10	10	10
是否有部分分	否	否	否
题目类型	传统型	传统型	传统型
是否有附加文件	是	是	是

提交源程序必须加后缀

对于 C++ 语言	cactus.cpp	bit.cpp	poly.cpp
对于 C 语言	cactus.c	bit.c	poly.c
对于 Pascal 语言	cactus.pas	bit.pas	poly.pas

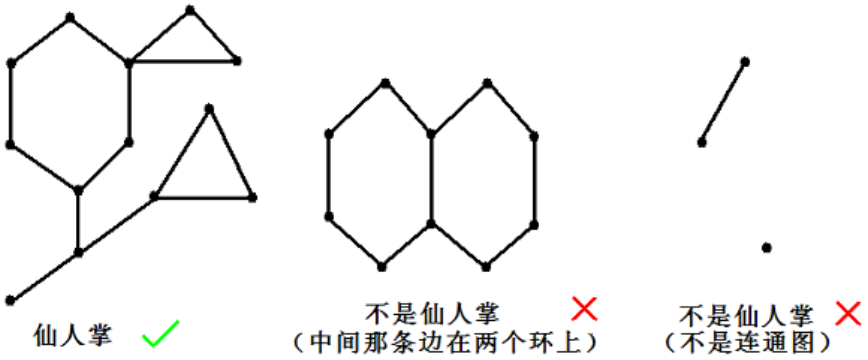
编译开关

对于 C++ 语言	-O2 -lm	-O2 -lm	-O2 -lm
对于 C 语言	-O2 -lm	-O2 -lm	-O2 -lm
对于 Pascal 语言	-O2	-O2	-O2

1 仙人掌

1.1 题目描述

如果一个**无自环无重边无向连通图**的任意一条边最多属于一个简单环，我们就称之为仙人掌。所谓简单环即不经过重复的结点的环。



现在九条可怜手上有一张**无自环无重边的无向连通图**，但是她觉得这张图中的边数太少了，所以她想要在图上连上一些新的边。同时为了方便的存储这张无向图，图中的边数又不能太多。经过权衡，她想要加边后得到的图为一棵仙人掌。

不难发现合法的加边方案有很多，可怜想要知道总共有多少不同的加边方案。

两个加边方案是不同的当且仅当一个方案中存在一条另一个方案中没有的边。

1.2 输入格式

多组数据，第一行输入一个整数 T 表示数据组数。

每组数据第一行输入两个整数 n, m，表示图中的点数与边数。

接下来 m 行，每行两个整数 $u, v(1 \leq u, v \leq n, u \neq v)$ 表示图中的一条边。保证输入的图联通且没有自环与重边。

1.3 输出格式

对于每组数据，输出一个整数表示方案数，当然方案数可能很大，请对 998244353 取模后输出。

1.4 样例输入

```
2
3 2
1 2
```

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

1.5 样例输出

```
2
8
```

1.6 样例解释

对于第一组样例合法加边的方案有 $\{\},\{(2,3)\}$，共 2 种。

1.7 数据范围与约定

测试点编号	$\sum n$	m	其他约定
1	≤ 5	≤ 10	无
2	≤ 2000	$\leq 2\times 10^5$	
3			
4	$\leq 10^5$	$=n-1$	图为一条链
5			
6			无
7			
8	$\leq 5\times 10^5$	$\leq 10^6$	
9			
10			

对于 100% 的数据，保证 $1\leq m\leq \frac{n(n-1)}{2},\sum m\leq 10^6$。

注意 T 可能会较大，请注意控制初始化的复杂度。

选手目录下的大数据中，四组数据依次满足第 $2,4,6,8$ 个点的条件。

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

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 ← (A_x + 1) mod 2
4:         x ← x − 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 ← 0
13:    while x ≤ n do
14:        ans ← (ans + A_x) mod 2
15:        x ← x + lowbit(x)
16:    end while
17:    return ans
18: end function
19:
20: function Query(l, r)
21:    ansl ← Find(l − 1)
22:    ansr ← Find(r)
23:    return (ansr − ansl + 2) mod 2
24: end function
```

其中 lowbit(x) 表示数字 x **最低**的非 0 二进制位，例如 lowbit(5) = 1, lowbit(12) = 4。进行第一类操作的时候就调用 Add(x)，第二类操作的时候答案就是 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，这样她就可以再次的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 Add(x)。
- 2 l r，表示询问执行 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 $x \times y^{-1} \mod 998244353$ 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 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			
8			无
9			
10			

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

3 多项式

3.1 题目描述

九条可怜最近研究了一下多项式在系数模 2 意义下的性质。她发现可以用多项式在模 2 意义下的乘法得到一个很长的字符串：

对于一个 n 次的系数为 0 或 1 的多项式 $f(x)$，我们在模 2 意义下计算 $g(x)=f(x)^m$，则 $g(x)$ 为一个 nm 次的多项式，它有 $nm+1$ 个系数，将这些系数从**高位到低位**写下来，就可以得到一个长度为 $nm+1$ 的 01 字符串。

例如对于多项式 $f(x)=x^3+x+1$，计算 $g(x)=f(x)^3=x^9+x^7+x^6+x^5+x^2+x+1$，这样我们得到了一个长度为 10 的字符串 1011100111。

现在可怜有一个次数为 n 的多项式 $f(x)$，整数 m,L,R 以及一个长度为 K 的 01 串 t。令 s 为 $f(x)^m$ 得到的字符串，$s[L,R]$ 为 s 的第 L 个字符到第 R 个字符，可怜想要知道 t 在 $s[L,R]$ 中出现了多少次。

3.2 输入格式

第一行输入一个整数 T 表示数据组数。

每组数据第一行输入五个整数 n,m,K,L,R。

第二行输入一个长度为 $n+1$ 的 01 串表示多项式 $f(x)$ 的系数，其中第 i 位表示 $f(x)$ 的第 $n-i+1$ 次系数。

第三行输入一个长度为 K 的字符串表示字符串 t。

3.3 输出格式

对于每组数据输出一个整数表示答案。

3.4 样例输入

```
1
3 3 2 1 10
1011
01
```

3.5 样例输出

```
2
```

3.6 数据范围与约定

测试点编号	n	m	K	其他约定
1	≤ 18	≤ 500	≤ 18	无
2		$\leq 2 \times 10^5$		
3				
4		$\leq 10^{16}$		$R - L \leq 10^4$
5				
6				$L = 1, R = nm + 1$
7				
8				
9				无
10				

对于 100% 的数据，保证 $1 \leq T \leq 5$，$1 \leq L \leq R \leq nm + 1$。