NOI2018 湖南省组队选拔赛

第一试试题

一．题目概况

题目名称	寻宝游戏	转盘	毒瘤
目录	hunt	circle	duliu
可执行文件名	hunt	circle	duliu
输入文件名	hunt.in	circle.in	duliu.in
输出文件名	hunt.out	circle.out	duliu.out
每个测试点时限	1 秒	2 秒	1 秒
测试点数目	10	10	20
每个测试点分值	10	10	5
结果比较方式	整数比较，多行单个数字比较	整数比较，多行单个数字比较	整数比较，单行单个数字比较
题目类型	传统	传统	传统
内存上限	512M	512M	512M

二．提交源程序需加后缀

对于 C++语言	hunt.cpp	circle.cpp	duliu.cpp
对于 C 语言	hunt.c	circle.c	duliu.c
对于 Pascal 语言	hunt.pas	circle.pas	duliu.pas

三．编译命令

对于 C++语言	g++ -o hunt hunt.cpp -lm	g++ -o circle circle.cpp -lm -O2	g++ -o duliu duliu.cpp -lm
对于 C 语言	gcc –o hunt hunt.c -lm	gcc -o circle circle.cpp -lm -O2	gcc -o duliu duliu.cpp -lm
对于 Pascal 语言	fpc hunt.pas	fpc circle.pas -O2	fpc duliu.pas

注意事项：

（1） 选手必须在自己的工作目录下操作，严禁在其他目录下工作。目录结构请遵从 NOI 规范，即需要在工作目录下再**为每个题目建相应子目录，子目录名为对应题目的英文名**。

（2） 选手最后提交的源程序（.pas 或.c 或.cpp）**必须在自己的工作目录里对应子目录下，**对于缺少文件者，不予测试，该题计零分。

（3） 子目录名、源程序文件名和输入输出文件名**必须使用英文小写**。

（4） 特别提醒：评测在 NOI Linux 下进行。

第 1 题：寻宝游戏(hunt)，运行时限 1s，内存上限 512M，100 分。

【问题描述】

某大学每年都会有一次 Mystery Hunt 的活动，玩家需要根据设置的线索解谜，找到宝藏的位置，前一年获胜的队伍可以获得这一年出题的机会。

作为新生的你，对这个活动非常感兴趣。你每天都要从西向东经过教学楼一条很长的走廊，这条走廊是如此的长，以至于它被人戏称为 infinite corridor。一次，你经过这条走廊的时候，注意到在走廊的墙壁上隐藏着n个**等长的**二进制的数字，长度均为m。你从西向东将这些数字记录了下来，形成一个含有n个数的二进制数组 $a_1,a_2,\ldots,a_n$。

很快，在最新的一期 Voo Doo 杂志上，你发现了q个长度也为m的二进制串 $r_1,r_2,\ldots,r_q$。

聪明的你很快发现了这些数字的含义。

保持数组 $a_1,a_2,\ldots,a_n$ 的元素顺序不变，你可以在它们之间插入∧（按位与运算）或者∨（按位或运算）两种二进制运算符。例如：11011∧00111=00011，11011∨00111=11111。

你需要插入恰好n个运算符，相邻两个数之间恰好一个，在**第一个数的左边**还有一个。**如果我们在第一个运算符的左边补入一个 0**，这就形成了一个运算式，我们可以计算它的值。与往常一样，运算顺序是**从左往右**。有趣的是，出题人已经告诉你这个值的可能的集合——Voo Doo 杂志里的那一些二进制数 $r_1,r_2,\ldots,r_q$，而解谜的方法，就是对 $r_1,r_2,\ldots,r_q$ 中的每一个值 r_i，分别计算出**有多少种方法填入这n个运算符**，使得这个运算式的值是 r_i。

然而，infinite corridor 真的很长，这意味着数据范围可能非常大。因此，答案也可能非常大，但是你发现由于谜题的特殊性，你只需要求答案模 1000000007(10^9+7,一个质数)的值。

【程序文件名】

源程序文件名为 hunt.cpp/c/pas。

【输入格式】

输入文件名为 hunt.in。

第一行三个数n,m,q，含义如题所述。

接下来n行，其中第i行有一个长度为m的二进制串，**左边是最高位**，表示a_i。

接下来q行，其中第 i 行有一个长度为m的二进制串，**左边是最高位**，表示 r_i。

【输出格式】

输出文件名为 hunt.out。

输出q行，每行一个数，其中第 i 行表示对应于 r_i 的答案。

【输入输出样例 1】

hunt.in	hunt.out
5 5 1 01110 11011 10000 01010 00100 00100	6

【样例解释 1】

有以下且仅有以下六个运算式的值是00100_2：**(下标 2 表示被标识的数是二进制数)**

$$0 \wedge 01110_2 \wedge 11011_2 \vee 10000_2 \wedge 01010_2 \vee 00100_2$$
$$0 \vee 01110_2 \vee 11011_2 \wedge 10000_2 \wedge 01010_2 \vee 00100_2$$
$$0 \wedge 01110_2 \vee 11011_2 \wedge 10000_2 \wedge 01010_2 \vee 00100_2$$
$$0 \vee 01110_2 \wedge 11011_2 \wedge 10000_2 \wedge 01010_2 \vee 00100_2$$
$$0 \wedge 01110_2 \wedge 11011_2 \wedge 10000_2 \wedge 01010_2 \vee 00100_2$$
$$0 \vee 01110_2 \vee 11011_2 \vee 10000_2 \vee 01010_2 \wedge 00100_2$$

【输入输出样例 2】

hunt.in	hunt.out
10 10 3 0100011011 0110100101 1100010100 0111000110 1100011110 0001110100 0001101110 0110100001 1110001010 0010011101 0110011111 1101001010 0010001001	69 0 5

【数据范围】

对于 10%的数据，$n \leq 20, m \leq 30, q = 1$

对于另外 20%的数据，$n \leq 1000, m \leq 16$

对于另外 40%的数据，$n \leq 500, m \leq 1000$

对于 100%的数据，$1 \leq n \leq 1000, 1 \leq m \leq 5000, 1 \leq q \leq 1000$

【提示】

输入文件可能很大，请注意读入效率。

【编译命令】

对于 c++语言： g++ -o hunt hunt.cpp –lm

对于 c 语言： gcc -o hunt hunt.c –lm

对于 pascal 语言： fpc hunt.pas

第 2 题：转盘(circle)，运行时限 2s，内存上限 512M，100 分。

【问题描述】

一次小 G 和小 H 原本准备去聚餐，但由于太麻烦了于是题面简化如下：

一个转盘上有摆成一圈的n个物品（编号 1 至 n），其中第i个物品会在T_i时刻出现。

在0时刻时，小 G 可以任选n个物品中的一个，我们将其编号记为s_0。并且如果i时刻选择了物品s_i，那么$i+1$时刻可以继续选择当前物品或者选择下一个物品。当s_i为n时，下一个物品为物品1，否则下一个物品为s_i+1。在每一时刻（包括0时刻）如果小 G 所选择的物品已经出现了，那么小 G 将会标记它。小 H 想知道,在物品选择的最优策略下，小 G 什么时候能标记所有物品？

但麻烦的是，物品的出现时间会不时修改。我们将其描述为m次修改，每次修改将改变其中一个物品的出现时间。每次修改之后，你也需要求出当前局面的答案。对于其中部分测试点，小 H 还追加了强制在线的要求。

【程序文件名】

源程序文件名为$circle.cpp/c/pas$。

【输入格式】

输入文件名为$circle.in$。

第一行三个非负整数n、m、p，代表一共有n个物品，m次修改。p只有0或1两种取值，强制在线时p为1，否则为0。本节后面将解释如何使用p。

接下来一行，有n个用空格隔开的**非负整数**，第i个数T_i代表物品i的出现时间。

接下来m行，每行两个非负整数x、y，代表一次修改及询问。修改方式如下：

（1）如果$p=0$，则表示物品x的出现时间T_x修改为y。

（2）如果$p=1$，则先将x和y分别异或$LastAns$得到x'和y'：即$x'=x\ xor\ LastAns,\ y'=y\ xor\ LastAns$。然后将物品$x'$的出现时间$T_{x'}$修改为$y'$。其中的$LastAns$是前一个询问的答案；特别的，第一次修改时的$LastAns$为初始局面的答案。其中的$xor$为按位异或运算，例如$1\ xor\ 2=3$，$4\ xor\ 5=1$，$6\ xor\ 11=13$。

保证输入合法。

【输出格式】

输出文件名为$circle.out$。

第一行一个整数代表初始局面的答案。

接下来$m+1$行每行一个整数分别代表每次修改后的答案。

【输入输出样例】

circle1.in	circle1.out
5 3 0 1 2 3 4 5 3 5 5 0 1 4	5 7 6 7

【样例解释】

第1次询问为：对于5个物品的出现时间分别是1、2、3、4、5时，最早什么时候标记完所有物品？其中一个最优策略为：在时刻0和1时都选择物品1，而在时刻2~5都选择下一物品。其中 0 时刻选择物品 1 尚未出现，无法标记。

第3次询问为：对于5个物品的出现时间分别是1、2、5、4、0时，最早什么时候标记完所有物品？。其中一个最优策略为：从时刻0至时刻6分别选择物品4、5、1、2、3、3、4。其中时刻0和时刻4时，所选物品尚未出现，其余时刻的所选物品都能被标记。

【数据范围】

测试点编号	n	m	T_i/T_x	p
1	≤ 10	≤ 10	≤ 10	$=0$
2	≤ 1000	$=0$	≤ 1000	
3	≤ 100000		≤ 100000	
4	≤ 5000	≤ 5000		
5	≤ 80000	≤ 80000		
6				$=1$
7	≤ 90000	≤ 90000		$=0$
8				$=1$
9	≤ 100000	≤ 100000		$=0$
10				$=1$

对于所有数据，保证$3 \leq n \leq 10^5$，$0 \leq m \leq 10^5$，$0 \leq T_i/T_x \leq 10^5$。

【编译命令】

对于 c++语言：　g++ -o circle circle.cpp -lm -O2

对于 c 语言：　gcc -o circle circle.c -lm -O2

对于 pascal 语言：　fpc circle.pas -O2

提示：本题将开启 O2 优化指令。

第 3 题：毒瘤(duliu)，运行时限 1s，内存上限 512M，100 分。

【问题描述】

从前有一名毒瘤。

毒瘤最近发现了量产毒瘤题的奥秘。考虑如下类型的数据结构题：给出一个数组，要求支持若干种奇奇怪怪的修改操作（例如给一个区间内的数同时加上c，或者将一个区间内的数同时开平方根），并且支持询问区间的和。毒瘤考虑了n个这样的修改操作，并将它们编号为$1 \sim n$。当毒瘤要出数据结构题的时候，他就将这些修改操作中选若干个出来，然后出成一道题。

当然了，这样出的题有可能不可做。通过精妙的数学推理，毒瘤揭露了这些修改操作之间的关系：有m对“互相排斥”的修改操作，第i对是第u_i个操作和第v_i个操作。当一道题中同时含有u_i和v_i这两个操作时，这道题就会变得不可做。另一方面，当一道题中不包含任何“互相排斥”的操作时，这个题就是可做的。此外，毒瘤还发现了一个规律：$m-n$是一个很小的数字（参见“数据范围”中的说明），且任意两个修改操作都是连通的。两个修改操作a, b是连通的，当且仅当存在若干操作$t_0, t_1, \ldots, t_l$，使得$t_0 = a$，$t_l = b$，且对任意$1 \le i \le l$，t_{i-1}和t_i都是“互相排斥”的修改操作。

一对“互相排斥”的修改操作称为互斥对。现在毒瘤想知道，给定值n和m个互斥对，他一共能出出多少道可做的不同的数据结构题。两个数据结构题是不同的，当且仅当其中某个操作出现在了其中一个题中，但是没有出现在另一个题中。

【程序文件名】

源程序文件名为 duliu.cpp/c/pas。

【输入格式】

输入文件名为duliu.in。

第一行为正整数n, m。

接下来m行，每行两个正整数u, v，代表一对“互相排斥”的修改操作。

【输出格式】

输出文件名为duliu.out。

输出一行一个整数，表示毒瘤可以出的可做的不同的数据结构题的个数。这个数可能很大，所以只输出模998244353后的值。

【输入输出样例 1】

duliu.in	duliu.out
3 2 1 2 2 3	5

【样例解释 1】

可做的数据结构题有：空集，{1}，{2}，{3}，{1,3}。注意：**空集是合法的数据结构题**。

【输入输出样例 2】

duliu.in	duliu.out
6 8 1 2 1 3 1 4 2 4 3 5 4 5 4 6 1 6	16

【输入输出样例 3】

duliu.in	duliu.out
12 18 12 6 3 11 8 6 2 9 10 4 1 8 6 2 11 5 10 6 12 2 9 3 7 6 2 7 3 2 7 3 5 6 2 11 12 1	248

【数据范围】

测试点编号	$n \leq$	$m \leq$	测试点编号	$n \leq$	$m \leq$
$1 \sim 4$	20	$n+10$	$5 \sim 6$	100000	$n-1$
$7 \sim 8$	100000	n	9	3000	$n+1$
$10 \sim 11$	100000	$n+1$	$12 \sim 14$	3000	$n+10$
$15 \sim 16$	100000	$n+7$	$17 \sim 20$	100000	$n+10$

对所有数据，$n \leq 10^5$，$n-1 \leq m \leq n+10$。

【编译命令】

对于 c++语言： g++ -o duliu duliu.cpp –lm

对于c语言： gcc -o duliu duliu.c –lm

对于pascal语言： fpc duliu.pas