冲刺 NOI2023 模拟试题

Day1

（请选手务必仔细阅读本页内容）

一．题目概况

中文题目名称	神奇的纸牌	凌乱平衡树	打扫笛卡尔
英文题目与子目录名	uno	treap	cartesian
可执行文件名	uno	treap	cartesian
输入文件名	uno.in	treap.in	cartesian.in
输出文件名	uno.out	treap.out	cartesian.out
每个测试点时限	1.0 秒	1.0 秒	1.0 秒
测试点数目	20	20	20
测试点是否等分	是	是	是
附加样例文件	有	有	有
结果比较方式	全文比较（过滤行末空格及文末回车）		
题目类型	传统型	传统型	传统型
运行内存上限	512MB	512MB	512MB

二．提交源程序文件名

对于 C++语言	uno.cpp	treap.cpp	cartesian.cpp

三．编译选项

对于 C++语言	-lm -std=c++14 -O2

四．注意事项：

1、文件名（程序名和输入输出文件名）必须使用英文小写。

2、C/C++中函数 main()的返回值类型必须是 int，程序正常结束时的返回值必须是 0。

3、全国统一评测时采用的机器配置为：Intel(R) Core(TM) i7-8700K CPU @ 3.70GHz，内存 32GB。上述时限以此配置为准。

4、只提供 Linux 格式附加样例文件。

5、特别提醒：评测在当前最新公布的 NOI Linux 下进行，各语言的编译器版本以其为准。

神奇的纸牌（uno）

题目描述

小 Y 和小 T 在打 UNO。

小 T 太菜了，哪怕小 Y 允许她复制她拥有的牌，她也总是输。小 Y 看不下去了，所以决定教她 UNO 的技巧。

UNO 的牌一共有两个属性：花色和点数。其中花色有四种，分别是红色 (R) 黄色 (Y) 蓝色 (B) 绿色 (G)，而点数一共有 n 种，分别从 1 到 n。每张属性相同的牌一共有 1 张，也就是说 UNO 牌一共有 $4n$ 张。一张牌的表示方法为数字加空格加字符，即 int char。

UNO 的规则是一开始先手任意打出一张牌，打出的这张牌就是目前局面上的牌，之后按照逆时针的方向逐个出牌。一个人可以出牌 x，当且仅当 x 的花色和局面上的牌相同，或者 x 的点数和局面上的牌相同。出完牌 x 后，局面上的牌会变成 x 。如果一个人不能出牌，那么 Ta 需要从牌堆摸一张牌。

玩 UNO 最重要的是要知道一组牌怎么通过合适的决策打得尽可能多。在小 Y 的教导下，小 T 很快就掌握了在允许复制的情况下的 UNO 的技巧。

小 T 现在想知道，在 $4n$ 张牌组成的 2^{4n} 个可能的手牌中，在允许复制牌的情况下（每张牌可以复制无限次），有多少组手牌在最优决策下可以一次出完。在这里，小 T 是先手，且小 T 的逆时针方向的下一个还是小 T（因为只有她一个人在打）。

因为数字太大了她认不出来，所以她会给你一个数 P，请对它取模。不保证 P 为质数。

输入格式

输入文件名为 uno.in。

第一行两个正整数 n, P，意义见题目描述。

输出格式

输出文件名为 uno.out。

一行一个整数表示答案。

输入输出样例 1

uno.in	uno.out
2 114514	206

见下发样例 example_uno1.in/out。

样例解释 1

直接枚举过于麻烦，故给出 $n = 2$ 的计算过程。

对于 $n = 2$，一个手牌不合法当且仅当两种数字都出现，且它们拥有的颜色集合没有交。故不合法的方案数为 $\sum_{i=1}^{3} \binom{4}{i} \times (2^i - 1) = 50$，所以合法方案数为 $2^{2\times 4} - 50 = 206$。

为了方便大家理解出牌方式，这里给出一种 1B,1G,2G,2R 的出牌方法：复制得到的 1G, 复制得到的 2G, 2R, 复制得到的 2G, 复制得到的 1G, 1B, 1G, 2G。当然，直接出：1B,1G,2G,2R 也是可行的，上面只是为了让大家明白怎么出牌怎么复制。

输入输出样例 2

见下发样例 example_uno2.in/out。数据性质同测试点 4。

输入输出样例 3

见下发样例 example_uno3.in/out。数据性质同测试点 7。

输入输出样例 4

见下发样例 example_uno4.in/out。数据性质同测试点 15。

数据规模与约定

对于 100% 的数据和所有样例，均保证 $1 \leq n \leq 10^{18}, 2 \leq P \leq 2 \times 10^9$。
对于不同的测试点，我们约定数据规模如下：

测试点编号	$n \leq$
1,2,3	5
4,5,6	10
7,8,9,10	2021
11,12,13,14	10^5
15,16,17	10^9
18,19,20	10^{18}

凌乱平衡树（treap）

题目描述

小 T 正在学习平衡树。

现在她手上有左右两个 Treap，其中左边的 Treap 上的权值严格小于右边的 Treap（即 fhq-treap 分裂后所得），她想把它们合并。（不难发现在合并的时候 Treap 上面的点的权值已经无用）

按照常规操作，在合并的时候，假设当前左 Treap 合并到的点是 x，右子树合并到的点是 y，子树大小分别是 sz_x, sz_y，那么 x 成为合并后这个点的概率是 $\frac{sz_x}{sz_x+sz_y}$。

但是小 T 忘记要随机化了，现在 x 成为合并后这个点当且仅当 $sz_x \geq sz_y$。

这样的话 Treap 的复杂度可能就不正确了。因为平衡树的复杂度和树的高度息息相关，所以小 T 定义一个 Treap 的复杂度定义为每个点的深度和，根节点的深度为 1。

她想知道用她的方法合并两个 Treap 之后，这个 Treap 的复杂度为多少。

但是因为她刚学平衡树，把 Treap 和 Splay 又搞混了，所以她还会对待合并的两个 Treap 进行一些 Rotate 操作（即 Zig-Zag）。

输入格式

输入文件名为 treap.in。

第一行两个正整数 n, m，表示左边的 Treap 节点个数和右边的 Treap 节点个数。

接下来 n 行，每行两个正整数 x_i, y_i，表示左边的 Treap 中编号为 i 的节点的左孩子标号和右孩子标号，如果它没有某个孩子，那么该值为 0。

接下来 m 行，每行两个正整数 x_i, y_i，表示右边的 Treap 中编号为 i 的节点的左孩子标号和右孩子标号，如果它没有某个孩子，那么该值为 0。

接下来给出一个正整数 q，表示小 T 会进行多少次 Rotate 操作。

接下来 q 行，每行两个正整数 op, x，若 $op = 1$，表示对左 Treap 的 x 号节点进行 Rotate，若 $op = 2$，表示对右 Treap 的 x 号节点进行 Rotate。保证 x 不为根（即 Rotate 操作合法）。

输出格式

输出文件名为 treap.out。

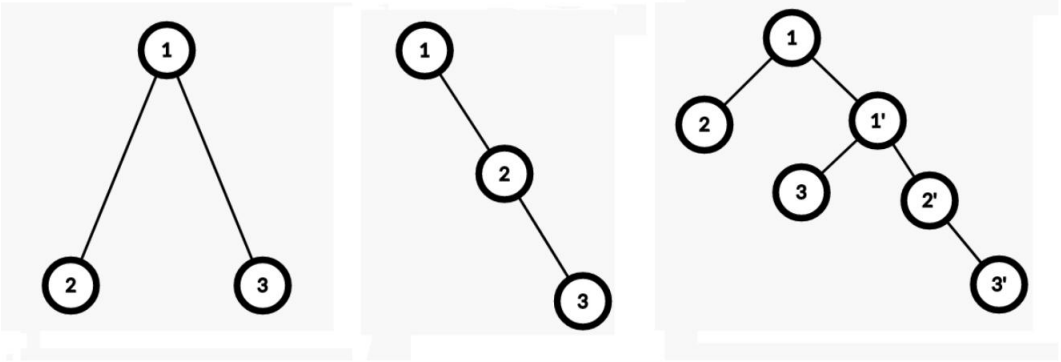
输出共 $q + 1$ 行，第 i 行表示对初始的两个 Treap 进行前 $i - 1$ 次 Rotate 操作后，合并得到的 Treap 的复杂度。

输入输出样例 1

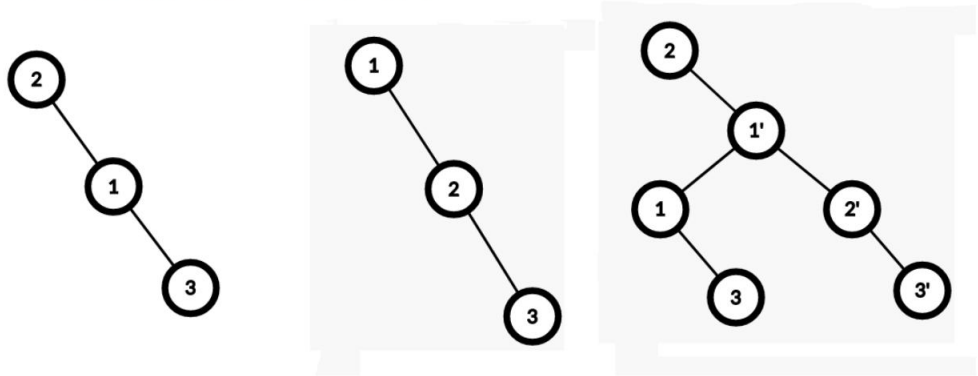
treap.in	treap.out
3 3 2 3 0 0 0 0 0 2 0 3 0 0 1 1 2	15 17

见下发样例 example_treap1.in/out。

样例解释 1



上面分别是第一个查询时左边的树，右边的树，合并后的树。为了方便，在合并后的树中左边的节点为 $1,2,3$，右边的节点为 $1',2',3'$。下同。



上面是第二个查询时的图。

输入输出样例 2

见下发样例 example_treap2.in/out。数据性质同测试点 1。

输入输出样例 3

见下发样例 example_treap3.in/out。数据性质同测试点 5。

输入输出样例 4

见下发样例 example_treap4.in/out。数据性质同测试点 13。

数据规模与约定

对于 100% 的数据和所有样例，均保证 $1 \le n, m, q \le 2 \times 10^5$，所有 Rotate 操作合法。
对于不同的测试点，我们约定数据规模如下：

测试点编号	$n \le$	$m \le$	$q \le$	特殊性质
1,2,3,4	2021	2021	2021	无
5,6,7,8	2×10^5	1	2×10^5	保证所有操作只在左边的树上
9,10,11,12		2×10^5		
13,14,15,16	5×10^4	5×10^4	5×10^4	无
17,18,19,20	2×10^5	2×10^5	2×10^5	

打扫笛卡尔（cartesian）

题目描述

小 Y 和小 T 在笛卡尔树上玩。

他们发现每次从根走到一个节点 x 都要经过 dep_x 个节点（包含根和 x，根的 dep 为 1），所以他们把一棵树的舒适度定义为 $\sum_{x\in T} dep_x$。

对于一棵笛卡尔树 T，自然要打扫干净在能在上面玩。为了偷懒，小 Y 在打扫笛卡尔树的时候并没有每个节点都打扫到，而是采取了一种随机的方式。方式如下：

1. 小 Y 一开始在根。
2. 如果当前节点没有打扫过，那么打扫它。
3. 假设现在小 Y 在的节点 x 还有 d 个孩子没有打扫过，那么他会等概率在节点 x 的双亲节点（这里假设根节点的双亲节点为【终止打扫节点】）和所有没去过的孩子节点组成的大小为 $d+1$ 的集合中随机选择一个节点（即概率是 $\frac{1}{d+1}$），并且走过去。
4. 如果当前节点是【终止打扫节点】，那么打扫终止。否则回到 2.。

最后打扫总会终止，而这时候如果把打扫过的节点拿出来，会得到一个新的树 T'，这棵树的根还是原来的根。

最后小 Y 和小 T 会在新树 T' 上玩。为了能更好的了解这棵树，他们想知道这棵新树 T' 的舒适度的期望。但是他们每过一会就会得到另外一棵笛卡尔树，所以请你求出对于所有长度为 n 的排列，把它建成**小根**笛卡尔树后，随机打扫得到的树的期望舒适度的和。

不过小 Y 和小 T 没有那么多时间来看这么长的数字，所以小 T 会告诉你一个她喜欢的数 P，请把答案对它取模。

输入格式

输入文件名为 cartesian.in。

第一行两个正整数 n, P，意义见题目描述。

输出格式

输出文件名为 cartesian.out。

一行一个整数表示答案。

输入输出样例 1

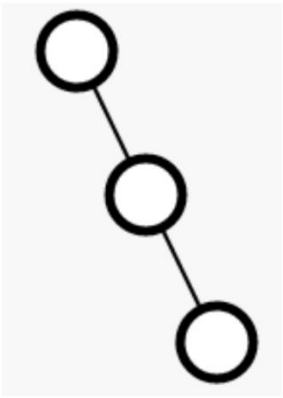
cartesian.in	cartesian.out
3 1000000007	17

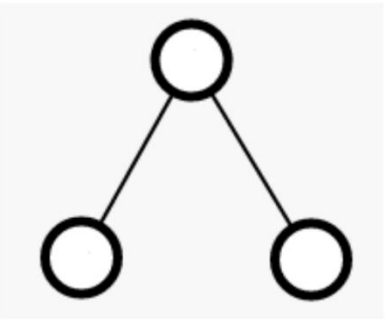
见下发样例 example_cartesian1.in/out。

样例解释 1

不难发现小 Y 的打扫并不关心孩子是左孩子还是右孩子（废话）。

所以一共有两种树：





排列 $1,2,3$、$1,3,2$、$2,3,1$、$3,2,1$ 形成的树是左图，排列 $2,1,3$、$3,1,2$ 形成的图是右图。

对于左图，随机打扫有以下三种方式：

根，结束，概率 $\frac{1}{2}$，舒适度 1。

根，根的孩子，根，结束，概率 $\frac{1}{2}\times\frac{1}{2}$，舒适度 $1+2=3$。

根，根的孩子，根的孩子的孩子，根的孩子，根，结束，概率 $\frac{1}{2}\times\frac{1}{2}$，舒适度 $1+2+3=6$。

故期望为 $\frac{1}{2}+\frac{1}{4}\times 3+\frac{1}{4}\times 6=\frac{11}{4}$。

对于右图，随机打扫有以下五种方式：

根，结束，概率 $\frac{1}{3}$，舒适度 1。

根，根的左孩子，根，结束，概率 $\frac{1}{3}\times\frac{1}{2}$，舒适度 $1+2=3$。

根，根的右孩子，根，结束，概率 $\frac{1}{3}\times\frac{1}{2}$，舒适度 $1+2=3$。

根，根的左孩子，根，根的右孩子，根，结束，概率 $\frac{1}{3}\times\frac{1}{2}$，舒适度 $1+2+2=5$。

根，根的右孩子，根，根的左孩子，根，结束，概率 $\frac{1}{3}\times\frac{1}{2}$，舒适度 $1+2+2=5$。

故期望为 $\frac{1}{3}\times 1+\frac{1}{6}\times 3\times 2+\frac{1}{6}\times 5\times 2=3$。

故总期望为 $\frac{11}{4}\times 4+3\times 2=17$。

输入输出样例 2

见下发样例 example_cartesian2.in/out。数据性质同测试点 3。

输入输出样例 3

见下发样例 example_cartesian3.in/out。数据性质同测试点 7。

输入输出样例 4

见下发样例 example_cartesian4.in/out。数据性质同测试点 15。

数据规模与约定

对于 100% 的数据和所有样例，均保证 $1\le n\le 10^7, 2\le P\le 2\times 10^9$。特别的，所有测试点编号 $1\sim 10$ 的测试点，和测试点编号在 $11\sim 20$ 之间且为奇数的测试点保证 P 为大于 n 的质数。

对于不同的测试点，我们约定数据规模如下：

测试点编号	$n \leq$
1,2	4
3,4	10
5,6	80
7,8,9,10	2021
11,12,13,14	3×10^4
15,16	10^5
17,18	10^6
19,20	10^7