

---

## 基础阶乘级算法演练

### background

是个被用烂了的 idea 出的题目, 当然并不是搬运题或者改编题.  
算是个半原创题吧, 所以不对题目解法和数据的正确性做保证

在过去的两周里, 我们学习了一些基础算法,  
其中有不少指数级/阶乘级搜索算法,  
这个题目将会帮助你熟悉它们.

spinach 的记忆力很差, 但是他非常肝, 出凑题目非常快.

---

### statement

hehelego/spinach 不喜欢位运算, 因为他在正式比赛中从来没有用上位运算.  
所以他出了一个和位运算有关的题目来鼓励学弟学妹

spinach 正在出题, 它接受了  $k$  场比赛的委托, 学弟学妹们的水平在比赛中快速得到提升,  
对于已经参加了  $i$  场比赛的 oier, 它在一场比赛中可以随时套板子不调试一次编码切掉  $i$  个题, 而恶毒的 spinach 不想让任何人 AK 离场.

spinach 的题目都是搬运原题, 太水了, 只能靠数量防止小朋友们 AK.

每场比赛是独立的, 这意味着 spinach 不用出搬运那么多题, 他可以把一个题目用在任意多场比赛中.

准备时间是  $n$  天, spinach 每天能出一个题剩下的时间用来学车, 在  $n$  天中出了编号分别为  $(a_0, a_1, a_2, a_3 \dots a_{n-1})$  的题目,

因为他的记忆力不足, 所以偶尔会出重题. 如果存在  $a_i = a_j$  那么在  $i, j$  两天它出里重题, 这样他的努力就"全部白费"了, 可用的题目数量会少于  $n$ ,

spinach 的思维是几乎处处不连续的, 我们有  $a_i = 5^i \bmod 3221225473$ .

现在出题工作已经结束里, 而 spinach 已经因为 996 工作进里 ICU 无力组装比赛里,  
他想请你, 一个即将在 NOIp 中 AK 的 oier 来帮助他组题, 它要你求出每场比赛组题方式数量的异或和. 他终于良心发现, 允许你不写高精度计算, 你只需要把每场比赛的组装方案数量模一个数  $p$ , 再把所有结果 xor 起来就行啦.

具体来说, 在第  $i$  场比赛中, 你要选一些不同的题目组成比赛, 而题目的顺序并不重要, 但是你要保证没有人 AK, 两个组题方案  $A_1, A_2$  有  $A_1 \neq A_2 \iff (\exists i)(a_i \in A_1 \wedge a_i \notin A_2)$

---

---

## I/O

### input

第一行三个整数, 分别为  $n, k, p$

### output

一行, 一个整数, 表示答案.

---

## example

### case1

- input

```
1 1 10 998244353
```

- output

```
1 1
```

- explanation

他太鸽了, 准备十场比赛只用一天来出题. 显然这样的情况下他不会出重复的题目, 于是他可用的题目有 1 个.

在 contest1, 所有人都没参加过比赛, 于是他只要出一个水题大家就 AK 不能了, 方案数是 1.

而之后的比赛中, 所有人至少可以切 1 个题, 至少需要出 2 个题, 然而他没那么多可用的题目, 于是比赛就没法组织了, 方案是 0.

于是答案是  $1 \wedge 0 \wedge 0 \dots 0 \wedge 0 = 1$

### case2

- input
-

---

```
1 3 2 2
```

- output

```
1 0
```

- explanation

他出了编号为  $a_0 = 1, a_1 = 5, a_2 = 25$  的题目, 真幸运, 没有出重题, 可用题目数是 3.

contest1 只要 1 个题就可以防 AK 了, 于是方案是 3, 而  $3 \equiv 1 \pmod{2}$ .

contest2 由于学弟和学妹水平提升, 需要 2 个题才行, 可以选  $\{a_0, a_1\}, \{a_0, a_2\}, \{a_1, a_2\}$  这 3 中方案, 有  $3 \equiv 1 \pmod{2}$ .

于是答案是  $1^1 = 0$

---

## restrictions

- compiler flags: -O2 (备注: luogu 使用的 gnu 套件版本高于 NOI linux 的 4.8.4, 两者的版本均高于 devcpp 自带的远古 mingw 和 cena 自带的编译器)
- TL=3.0s (备注: 现在 NOIp 的评测机性能和 Luogu 高性能模式是接近的, 但是蔽校一机房的远古教师机已经跟不上时代里... 请根据评测机实际性能对 TL 进行调整, 在 std 的 2 倍最大运行时间以上)
- ML=256MB

共 20 组数据, 不设置 subtask 模式评测, 而是传统的每个点分数相同, 得分互相独立.

- 对于前 50% 的数据, 有  $p$  为质数 (A 类别). 而后 50% 的数据中  $p$  为合数 (B 类别). 对于所有数据, 有  $p^2 < 2^{64}$  (备注, 你真的不用高精度)
- 对于 A, B: 前 50% 的数据, 有  $n \leq 10^5$ . 所有数据  $n \leq 10^{12}$
- 对于 A, B: 前 50% 的数据  $k \leq 200$ . 所有数据有  $k \leq 2000$
- 对于 A: 前 50% 的数据, 满足  $n < p$

说明: 精力有限, 数据没有划分出更多的 subtask, 深表歉意. 请各位自行训练在多个 subtask 的题目中的所谓按数据特性分治.