[CSP-J 2022] 乘方

题目描述

小文同学刚刚接触了信息学竞赛，有一天她遇到了这样一个题：给定正整数 a 和 b，求 a^b 的值是多少。

a^b 即 b 个 a 相乘的值，例如 2^3 即为 3 个 2 相乘，结果为 $2 \times 2 \times 2 = 8$。

“简单！”小文心想，同时很快就写出了一份程序，可是测试时却出现了错误。

小文很快意识到，她的程序里的变量都是 `int` 类型的。在大多数机器上，`int` 类型能表示的最大数为 $2^{31} - 1$，因此只要计算结果超过这个数，她的程序就会出现错误。

由于小文刚刚学会编程，她担心使用 `int` 计算会出现问题。因此她希望你在 a^b 的值超过 10^9 时，输出一个 `-1` 进行警示，否则就输出正确的 a^b 的值。

然而小文还是不知道怎么实现这份程序，因此她想请你帮忙。

输入格式

输入共一行，两个正整数 a, b。

输出格式

输出共一行，如果 a^b 的值不超过 10^9，则输出 a^b 的值，否则输出 `-1`。

样例 #1

样例输入 #1

```
10 9
```

样例输出 #1

```
1000000000
```

样例 #2

样例输入 #2

```
23333 66666
```

样例输出 #2

```
-1
```

提示

对于 10% 的数据，保证 $b = 1$。
对于 30% 的数据，保证 $b \leq 2$。
对于 60% 的数据，保证 $b \leq 30$，$a^b \leq 10^{18}$。
对于 100% 的数据，保证 $1 \leq a, b \leq 10^9$。

upd 2022.11.14：新增加一组 Hack 数据。

[CSP-J 2022] 解密

题目描述

给定一个正整数 k，有 k 次询问，每次给定三个正整数 n_i, e_i, d_i，求两个正整数 p_i, q_i，使 $n_i = p_i \times q_i$、$e_i \times d_i = (p_i - 1)(q_i - 1) + 1$。

输入格式

第一行一个正整数 k，表示有 k 次询问。

接下来 k 行，第 i 行三个正整数 n_i, d_i, e_i。

输出格式

输出 k 行，每行两个正整数 p_i, q_i 表示答案。

为使输出统一，你应当保证 $p_i \leq q_i$。

如果无解，请输出 `NO`。

样例 #1

样例输入 #1

```
10
770 77 5
633 1 211
545 1 499
683 3 227
858 3 257
723 37 13
572 26 11
867 17 17
829 3 263
528 4 109
```

样例输出 #1

```
2 385
NO
NO
NO
11 78
3 241
2 286
NO
NO
6 88
```

提示

【样例 #2】

见附件中的 `decode/decode2.in` 与 `decode/decode2.ans`。

【样例 #3】

见附件中的 `decode/decode3.in` 与 `decode/decode3.ans`。

【样例 #4】

见附件中的 `decode/decode4.in` 与 `decode/decode4.ans`。

【数据范围】

以下记 $m = n - e \times d + 2$。

保证对于 100% 的数据，$1 \leq k \leq 10^5$，对于任意的 $1 \leq i \leq k$，$1 \leq n_i \leq 10^{18}$，$1 \leq e_i \times d_i \leq 10^{18}$
，$1 \leq m \leq 10^9$。

测试点编号	$k \leq$	$n \leq$	$m \leq$	特殊性质
1	10^3	10^3	10^3	保证有解
2	10^3	10^3	10^3	无
3	10^3	10^9	6×10^4	保证有解
4	10^3	10^9	6×10^4	无
5	10^3	10^9	10^9	保证有解
6	10^3	10^9	10^9	无

测试点编号	$k \le$	$n \le$	$m \le$	特殊性质
7	10^5	10^{18}	10^9	保证若有解则 $p = q$
8	10^5	10^{18}	10^9	保证有解
9	10^5	10^{18}	10^9	无
10	10^5	10^{18}	10^9	无

[CSP-J 2022] 逻辑表达式

题目描述

逻辑表达式是计算机科学中的重要概念和工具，包含逻辑值、逻辑运算、逻辑运算优先级等内容。

在一个逻辑表达式中，元素的值只有两种可能：0（表示假）和 1（表示真）。元素之间有多种可能的逻辑运算，本题中只需考虑如下两种：“与”（符号为 `&`）和“或”（符号为 `|`）。其运算规则如下：

$0 \mathbin{\&} 0 = 0 \mathbin{\&} 1 = 1 \mathbin{\&} 0 = 0$，$1 \mathbin{\&} 1 = 1$；
$0 \mid 0 = 0$，$0 \mid 1 = 1 \mid 0 = 1 \mid 1 = 1$。

在一个逻辑表达式中还可能有括号。规定在运算时，括号内的部分先运算；两种运算并列时，`&` 运算优先于 `|` 运算；同种运算并列时，从左向右运算。

比如，表达式 `0|1&0` 的运算顺序等同于 `0|(1&0)`；表达式 `0&1&0|1` 的运算顺序等同于 `((0&1)&0)|1`。

此外，在 C++ 等语言的有些编译器中，对逻辑表达式的计算会采用一种“短路”的策略：在形如 `a&b` 的逻辑表达式中，会先计算 `a` 部分的值，如果 $a = 0$，那么整个逻辑表达式的值就一定为 0，故无需再计算 `b` 部分的值；同理，在形如 `a|b` 的逻辑表达式中，会先计算 `a` 部分的值，如果 $a = 1$，那么整个逻辑表达式的值就一定为 1，无需再计算 `b` 部分的值。

现在给你一个逻辑表达式，你需要计算出它的值，并且统计出在计算过程中，两种类型的“短路”各出现了多少次。需要注意的是，如果某处“短路”包含在更外层被“短路”的部分内则不被统计，如表达式 `1|(0&1)` 中，尽管 `0&1` 是一处“短路”，但由于外层的 `1|(0&1)` 本身就是一处“短路”，无需再计算 `0&1` 部分的值，因此不应当把这里的 `0&1` 计入一处“短路”。

输入格式

输入共一行，一个非空字符串 s 表示待计算的逻辑表达式。

输出格式

输出共两行，第一行输出一个字符 `0` 或 `1`，表示这个逻辑表达式的值；第二行输出两个非负整数，分别表示计算上述逻辑表达式的过程中，形如 `a&b` 和 `a|b` 的“短路”各出现了多少次。

样例 #1

样例输入 #1

```
0&(1|0)|(1|1|1&0)
```

样例输出 #1

```
1
1 2
```

样例 #2

样例输入 #2

```
(0|1&0|1|1|(1|1))&(0&1&(1|0)|0|1|0)&0
```

样例输出 #2

```
0
2 3
```

提示

【样例解释 #1】

该逻辑表达式的计算过程如下，每一行的注释表示上一行计算的过程：

```
0&(1|0)|(1|1|1&0)
=(0&(1|0))|((1|1)|(1&0)) //用括号标明计算顺序
=0|((1|1)|(1&0))    //先计算最左侧的 &，是一次形如 a&b 的“短路”
=0|(1|(1&0))        //再计算中间的 |，是一次形如 a|b 的“短路”
=0|1                //再计算中间的 |，是一次形如 a|b 的“短路”
=1
```

【样例 #3】

见附件中的 `expr/expr3.in` 与 `expr/expr3.ans`。

【样例 #4】

见附件中的 `expr/expr4.in` 与 `expr/expr4.ans`。

【数据范围】

设 $|s|$ 为字符串 s 的长度。

对于所有数据，$1 \leq |s| \leq 10^6$。保证 s 中仅含有字符 `0`、`1`、`&`、`|`、`(`、`)` 且是一个符合规范的逻辑表达式。保证输入字符串的开头、中间和结尾均无额外的空格。保证 s 中没有重复的括号嵌套（即没有形如 `((a))` 形式的子串，其中 `a` 是符合规范的逻辑表达式）。

测试点编号	$\|s\| \leq$	特殊条件
$1 \sim 2$	3	无
$3 \sim 4$	5	无
5	2000	1
6	2000	2
7	2000	3
$8 \sim 10$	2000	无
$11 \sim 12$	10^6	1
$13 \sim 14$	10^6	2
$15 \sim 17$	10^6	3
$18 \sim 20$	10^6	无

其中：
特殊性质 1 为：保证 s 中没有字符 `&`。
特殊性质 2 为：保证 s 中没有字符 `|`。
特殊性质 3 为：保证 s 中没有字符 `(` 和 `)`。

【提示】

以下给出一个"符合规范的逻辑表达式"的形式化定义：

- 字符串 `0` 和 `1` 是符合规范的；
- 如果字符串 `s` 是符合规范的，且 `s` 不是形如 `(t)` 的字符串（其中 `t` 是符合规范的），那么字符串 `(s)` 也是符合规范的；
- 如果字符串 `a` 和 `b` 均是符合规范的，那么字符串 `a&b`、`a|b` 均是符合规范的；
- 所有符合规范的逻辑表达式均可由以上方法生成。

[CSP-J 2022] 上升点列

题目描述

在一个二维平面内，给定 n 个整数点 (x_i, y_i)，此外你还可以自由添加 k 个整数点。

你在自由添加 k 个点后，还需要从 $n + k$ 个点中选出若干个整数点并组成一个序列，使得序列中任意相邻两点间的欧几里得距离恰好为 1 而且横坐标、纵坐标值均单调不减，即 $x_{i+1} - x_i = 1, y_{i+1} = y_i$ 或 $y_{i+1} - y_i = 1, x_{i+1} = x_i$。请给出满足条件的序列的最大长度。

输入格式

第一行两个正整数 n, k 分别表示给定的整点个数、可自由添加的整点个数。

接下来 n 行，第 i 行两个正整数 x_i, y_i 表示给定的第 i 个点的横纵坐标。

输出格式

输出一个整数表示满足要求的序列的最大长度。

样例 #1

样例输入 #1

```
8 2
3 1
3 2
3 3
3 6
1 2
2 2
5 5
5 3
```

样例输出 #1

```
8
```

样例 #2

样例输入 #2

```
4 100
10 10
15 25
20 20
30 30
```

样例输出 #2

```
103
```

提示

【样例 #3】

见附件中的 `point/point3.in` 与 `point/point3.ans`。

第三个样例满足 $k = 0$。

【样例 #4】

见附件中的 `point/point4.in` 与 `point/point4.ans`。

【数据范围】

保证对于所有数据满足：$1 \le n \le 500$，$0 \le k \le 100$。对于所有给定的整点，其横纵坐标 $1 \le x_i, y_i \le 10^9$，且保证所有给定的点互不重合。对于自由添加的整点，其横纵坐标不受限制。

测试点编号	$n \le$	$k \le$	$x_i, y_i \le$
$1 \sim 2$	10	0	10
$3 \sim 4$	10	100	100
$5 \sim 7$	500	0	100
$8 \sim 10$	500	0	10^9
$11 \sim 15$	500	100	100
$16 \sim 20$	500	100	10^9