

水题

3006. Dirichlet's Theorem on Arithmetic Progressions

Dirichlet's Theorem on Arithmetic Progressions

Time Limit: 1000MS

Memory Limit: 65536K

Total Submissions: 18866 Accepted: 9468

Description

If a and d are relatively prime positive integers, the arithmetic sequence beginning with a and increasing by d , i.e., $a, a + d, a + 2d, a + 3d, a + 4d, \dots$, contains infinitely many prime numbers. This fact is known as Dirichlet's Theorem on Arithmetic Progressions, which had been conjectured by Johann Carl Friedrich Gauss (1777 - 1855) and was proved by Johann Peter Gustav Lejeune Dirichlet (1805 - 1859) in 1837.

For example, the arithmetic sequence beginning with 2 and increasing by 3, i.e.,

2, 5, 8, 11, 14, 17, 20, 23, 26, 29, 32, 35, 38, 41, 44, 47, 50, 53, 56, 59, 62, 65, 68, 71, 74, 77, 80, 83, 86, 89, 92, 95, 98, \dots ,

contains infinitely many prime numbers

2, 5, 11, 17, 23, 29, 41, 47, 53, 59, 71, 83, 89, \dots

Your mission, should you decide to accept it, is to write a program to find the n th prime number in this arithmetic sequence for given positive integers a , d , and n .

Input

The input is a sequence of datasets. A dataset is a line containing three positive integers a , d , and n separated by a space. a and d are relatively prime. You may assume $a \leq 9307$, $d \leq 346$, and $n \leq 210$.

The end of the input is indicated by a line containing three zeros separated by a space. It is not a dataset.

Output

The output should be composed of as many lines as the number of the input datasets. Each line should contain a single integer and should never contain extra characters.

The output integer corresponding to a dataset a , d , n should be the n th prime

number among those contained in the arithmetic sequence beginning with a and increasing by d .

FYI, it is known that the result is always less than 10^6 (one million) under this input condition.

Sample Input

```
367 186 151
179 10 203
271 37 39
103 230 1
27 104 185
253 50 85
1 1 1
9075 337 210
307 24 79
331 221 177
259 170 40
269 58 102
0 0 0
```

Sample Output

```
92809
6709
12037
103
93523
14503
2
899429
5107
412717
22699
25673
```

本是水题, 结果第一次 TLE 了。其实就是素数的判断要尽量节省时间。这次的 `check_prime()` 函数会计入 Sources 文件夹中。

这里的 for 循环小括号内的 `i*i<=n` 就可以解决以前一直蜜汁出错的 `cmath` 头文件和 `sqrt` 函数的报错。(好像只在 POJ 见过) 可能用一个 `int` 来保存 `sqrt(n)` 会更加节约时间。不过我还没搞懂 POJ 为什么会在我写了 `cmath` 头文件的情况下爆出 `Compile Error...`

主要是要记下这个检查素数的 `check` 函数的写法。

2255. Tree Recovery

Tree Recovery

Time Limit: 1000MS

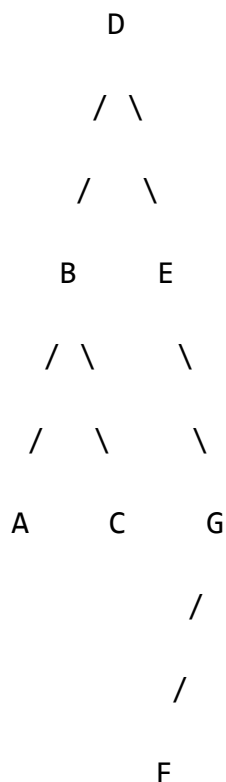
Memory Limit: 65536K

Total Submissions: 14977 Accepted: 9240

Description

Little Valentine liked playing with binary trees very much. Her favorite game was constructing randomly looking binary trees with capital letters in the nodes.

This is an example of one of her creations:



To record her trees for future generations, she wrote down two strings for each tree: a preorder traversal (root, left subtree, right subtree) and an inorder traversal (left subtree, root, right subtree). For the tree drawn above the preorder traversal is DBACEGF and the inorder traversal is ABCDEFG.

She thought that such a pair of strings would give enough information to reconstruct the tree later (but she never tried it).

Now, years later, looking again at the strings, she realized that

reconstructing the trees was indeed possible, but only because she never had used the same letter twice in the same tree. However, doing the reconstruction by hand, soon turned out to be tedious. So now she asks you to write a program that does the job for her!

Input

The input will contain one or more test cases. Each test case consists of one line containing two strings preord and inord, representing the preorder traversal and inorder traversal of a binary tree. Both strings consist of unique capital letters. (Thus they are not longer than 26 characters.) Input is terminated by end of file.

Output

For each test case, recover Valentine's binary tree and print one line containing the tree's postorder traversal (left subtree, right subtree, root).

Sample Input

```
DBACEGF ABCDEFG
BCAD CBAD
```

Sample Output

```
ACBFGED
CDAB
```

用递归来恢复。
还是看的博客

AC: http://poj.org/showsource?solution_id=17051023

枚举

1753. Flip Game

Description

Flip game is played on a rectangular 4x4 field with two-sided pieces placed on each of its 16 squares. One side of each piece is white and the other one is black and each piece is lying either it's black or white side up. Each round you flip 3 to 5 pieces, thus changing the color of their upper side from black to white and vice versa. The pieces to be flipped are chosen every round according to the following rules:

1. Choose any one of the 16 pieces.
2. Flip the chosen piece and also all adjacent pieces to the left, to the right, to the top, and to the bottom of the chosen piece (if there are any).

Consider the following position as an example:

bwbw

www

bbwb

bwwb

Here "b" denotes pieces lying their black side up and "w" denotes pieces lying their white side up. If we choose to flip the 1st piece from the 3rd row (this choice is shown at the picture), then the field will become:

bwbw

bwww

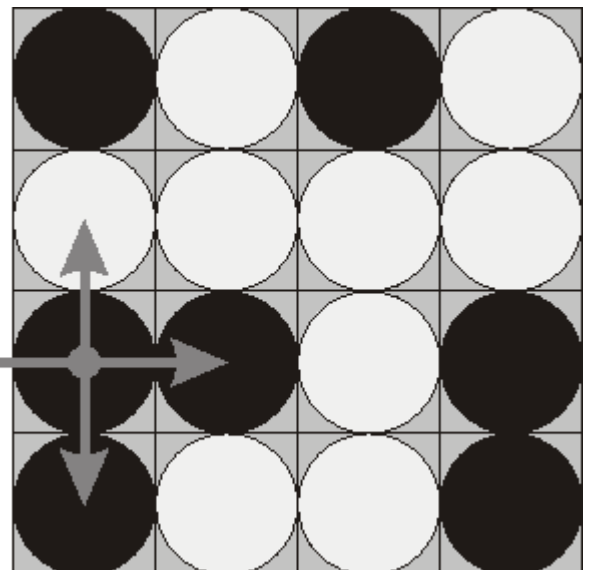
wwwb

wwwb

The goal of the game is to flip either all pieces white side up or all pieces black side up. You are to write a program that will search for the minimum number of rounds needed to achieve this goal.

Input

The input consists of 4 lines with 4 characters "w" or "b" each that denote game field position.



Output

Write to the output file a single integer number - the minimum number of rounds needed to achieve the goal of the game from the given position. If the goal is initially achieved, then write 0. If it's impossible to achieve the goal, then write the word "Impossible" (without quotes).

Sample Input

```
bwwb  
bbwb  
bwwb  
bwww
```

Sample Output

```
4
```

就是对于给定数组生成所有子集。把棋盘上的 16 个坐标记录在 `cord` 数组里，生成 16 个坐标的所有组合，对每种组合翻转该坐标上的棋子，如果翻转后棋盘满足全是一色则直接返回 `true` 表示当前的步数满足题意，最后输出步数最少的那个。如果对于所有子集都不满足，那么输出 `Impossible`。第一发 WA 竟然是因为没输出 `Impossible`.....

AC: http://poj.org/showsource?solution_id=16985496

WA: http://poj.org/showsource?solution_id=16985472

2965. The Pilots Brothers' Refrigerator

Description

The game "The Pilots Brothers: following the stripy elephant" has a quest where a player needs to open a refrigerator.

There are 16 handles on the refrigerator door. Every handle can be in one of two states: open or closed. The refrigerator is open only when all handles are open. The handles are represented as a matrix 4x4. You can change the state of a handle in any location `[i, j]` ($1 \leq i, j \leq 4$). However, this also changes states of all handles in row `i` and all handles in column `j`.

The task is to determine the minimum number of handle switching necessary to

open the refrigerator.

Input

The input contains four lines. Each of the four lines contains four characters describing the initial state of appropriate handles. A symbol “+” means that the handle is in closed state, whereas the symbol “-” means “open”. At least one of the handles is initially closed.

Output

The first line of the input contains N - the minimum number of switching. The rest N lines describe switching sequence. Each of the lines contains a row number and a column number of the matrix separated by one or more spaces. If there are several solutions, you may give any one of them.

Sample Input

```
--+-  
----  
----  
--+-
```

Sample Output

```
6  
1 1  
1 3  
1 4  
4 1  
4 3  
4 4
```

和 1753 一样都是枚举 4*4 矩阵内的所有坐标 (忽然想起线代全不会), 不过不同的是这里不会出现 Impossible 的情况。结果第一发 WA 不知道哪里出了错, 最后发现是枚举子集时本身没有枚举到 (即 16 个坐标全部做一次 change 操作)。Change 操作时要改变当前坐标的行和列, 用循环把行列上的坐标都操作后, 坐标 (i, j) 改变过两次, 最后应该再改变一次。

发现 1753 我就没有枚举数组本身...好像不太对? 以后加上吧, 反正只增加一次计算的流程。

具体为什么可以通过枚举 16 个坐标的组合来实现...我好像还没搞太懂...

睡吧

WA: http://poj.org/showsource?solution_id=16989341

AC: http://poj.org/showsource?solution_id=16989360

贪心

1328. Radar Installation

Description

Accounting for Computer Machinists (ACM) has suffered from the Y2K bug and lost some vital data for preparing annual report for MS Inc.

All what they remember is that MS Inc. posted a surplus or a deficit each month of 1999 and each month when MS Inc. posted surplus, the amount of surplus was s and each month when MS Inc. posted deficit, the deficit was d . They do not remember which or how many months posted surplus or deficit. MS Inc., unlike other companies, posts their earnings for each consecutive 5 months during a year. ACM knows that each of these 8 postings reported a deficit but they do not know how much. The chief accountant is almost sure that MS Inc. was about to post surplus for the entire year of 1999. Almost but not quite.

Write a program, which decides whether MS Inc. suffered a deficit during 1999, or if a surplus for 1999 was possible, what is the maximum amount of surplus that they can post.

Input

Input is a sequence of lines, each containing two positive integers s and d .

Output

For each line of input, output one line containing either a single integer giving the amount of surplus for the entire year, or output Deficit if it is impossible.

Sample Input

```
59 237
375 743
200000 849694
2500000 8000000
```

Sample Output

```
116
28
300612
Deficit
```


已经写成了自己生成的随机测试数据能和网上下载的答案得到同样的结果了，然而特么就是不能 AC 我很懊恼。我可能缺少考虑了什么极端的数据。最后用网上下载的答案交上去 A 掉不想再看了（扶额）。

最后一版 WA: http://poj.org/showsource?solution_id=17005687

AC: http://poj.org/showsource?solution_id=17005687

2109. Power of Cryptography

Description

Current work in cryptography involves (among other things) large prime numbers and computing powers of numbers among these primes. Work in this area has resulted in the practical use of results from number theory and other branches of mathematics once considered to be only of theoretical interest.

This problem involves the efficient computation of integer roots of numbers. Given an integer $n \geq 1$ and an integer $p \geq 1$ you have to write a program that determines the n th positive root of p . In this problem, given such integers n and p , p will always be of the form k to the n^{th} power, for an integer k (this integer is what your program must find).

Input

The input consists of a sequence of integer pairs n and p with each integer on a line by itself. For all such pairs $1 \leq n \leq 200$, $1 \leq p < 10^{101}$ and there exists an integer k , $1 \leq k \leq 10^9$ such that $k^n = p$.

Output

For each integer pair n and p the value k should be printed, i.e., the number k such that $k^n = p$.


Sample Input

```
2 16
3 27
7 4357186184021382204544
```

Sample Output

```
4
```

其实和贪心没关系(?) 反正最后踏马十几行就 AC 了, 我是不是还不知道贪心算法确切的定义...无所谓了能 A 题就是好的。

这就是对于对数函数的性质的考察 。

AC: http://poj.org/showsource?solution_id=17009762

2586. Y2K Accounting Bug

Description

Accounting for Computer Machinists (ACM) has suffered from the Y2K bug and lost some vital data for preparing annual report for MS Inc.

All what they remember is that MS Inc. posted a surplus or a deficit each month of 1999 and each month when MS Inc. posted surplus, the amount of surplus was s and each month when MS Inc. posted deficit, the deficit was d . They do not remember which or how many months posted surplus or deficit. MS Inc., unlike other companies, posts their earnings for each consecutive 5 months during a year. ACM knows that each of these 8 postings reported a deficit but they do not know how much. The chief accountant is almost sure that MS Inc. was about to post surplus for the entire year of 1999. Almost but not quite.

Write a program, which decides whether MS Inc. suffered a deficit during 1999, or if a surplus for 1999 was possible, what is the maximum amount of surplus that they can post.

Input

Input is a sequence of lines, each containing two positive integers s and d .

Output

For each line of input, output one line containing either a single integer giving the amount of surplus for the entire year, or output Deficit if it is impossible.

Sample Input

59 237

375 743
200000 849694
2500000 8000000

Sample Output

116
28
300612
Deficit

本题重点在于理解题意。据称此题可称 poj 最大纸老虎...题意大致为：

微软发布财报时只会发布连续的五个月累计盈亏（如 1~5, 2~6, ..., 8~12）。现在已知 1999 年时微软整年所发布的八个财报全为亏损，且对于每个月，若公司盈利，则盈利值必为 s ，若公司亏损，则亏损值必为 d ，则下面输入 s 和 d ，输出公司 1999 年全年的最大盈利值。若公司亏损则输出 Deficit。

然后嘞，思路已经正确最后却还是蜜汁 WA...然后迅速重写了一个更短的版本就蜜汁 AC 了 (wtf) 。

WA: http://poj.org/showsource?solution_id=17010223

AC: http://poj.org/showsource?solution_id=17010283

扩展欧几里得

UVaLive 6428

:

:

https://icpcarchive.ecs.baylor.edu/index.php?option=com_onlinejudge&Itemid=8&page=show_problem&problem=4439

Exgcd 函数算出的 x 和 y 是使得 $a * x + b * y = \gcd(a, b)$ 的 x 和 y ，要求 $a * x' + b * y' = s$ (若 s 不能被 $\gcd(a, b)$ 整除则不存在)，则 $x' = x * (s / \gcd)$ ， $y' = y * (s / \gcd)$ ；之后要求最小的正整数 x ，可通过 $x = (x \% b + b) \% b$ 得到。

这道题据说会在求 $x' = x * (s / \gcd)$ 就爆 long long，由于有 $(a*b)\%c = ((a\%c)*(b\%c))\%c$ ，则直接求最小正整数 $x = ((x \% b) * (s / \gcd) \% b) \% b$ 就不会爆 long long 了。

oooooooo

好多扩展欧几里得都不会
再多做些吧

: <http://paste.ubuntu.com/25252958/>

Segment Tree

POJ 2777

: <http://poj.org/problem?id=2777>

注意题目中的说明“A may be larger than B”，在主函数中调用 Update 函数和 Query 函数时注意左边界应为 $\min(a, b)$ ，右边界应为 $\max(a, b)$ 。

题目特意限定颜色不超过 30 种，而最后要求输出 A 到 B 区间内共有多少种颜色。一开始我用一个 bool 数组来记录本区间内颜色是否出现过，每次更新操作和询问操作都花费了一个循环来实现，果然 T 了。。。

应该用一个 32 位整形变量代替我之前的 bool 数组，这样可以在 pushUp 时直接这样写

```
Tree[index].rec = 0;
```

```
Tree[index].rec |= Tree[index<<1].rec | Tree[index<<1|1].rec;
```

位运算又快，剩下了很多时间。

就差这一个二进制优化。然后只用了 360ms 就 A 掉了。。这优化好明显原先限时 1000ms

: <http://paste.ubuntu.com/25324405/>

KMP

POJ 2752

: <http://poj.org/problem?id=2752>

这道题用于加深对 Next 数组的含义的理解。做这道题之前我还没太懂 Next 里面的数字的意思。。做这道题时一番思考后发现了 Next 的一个小性质。具体看代码算了。。这题代码很短。。

主要是.....我 T 了好几发，最后竟是因为我用了 STL 的 set 来储存用于输出的 ans...

最后改成了 `int ans[maxn]` 就 A 了。下面给出改 STL 前后代码。

: TLE: <http://paste.ubuntu.com/25445880/>

AC: <http://paste.ubuntu.com/25445875/>

POJ 2406

: <http://poj.org/problem?id=2406>

又一个理解 Next 数组含义的题。到网上找到一个看似颇有道理的公式，但还不太懂自己之前的代码哪里错了... 我好菜啊

公式: $\text{len} / (\text{len} - \text{Next}[\text{len}])$

如果上面分母可以整除分子，那么答案就是这个分数的值；如果分母不能整除分子，答案就是 1（即这个字符串只能由其本身的 1 次方得到）

: 1: <http://paste.ubuntu.com/25455919/>（自己写的）

2: <http://paste.ubuntu.com/25455933/>（神奇公式）

POJ 1961

: <http://poj.org/problem?id=1961>

比上面的 2406 麻烦了一点，其实基本是同一道题，公式也套用的 2406 用的那个公式。

: <http://paste.ubuntu.com/25455973/>

HDU 1711

: <http://acm.hdu.edu.cn/showproblem.php?pid=1711>

模板题

Log 进来只是为了记一下 Next 数组的优化

虽然在这道题好像优化前后时间没变多少

ooo

: <http://paste.ubuntu.com/25508019/>

HDU 1003

: <http://acm.hdu.edu.cn/status.php>

dp 水题

$dp[i] = \max(dp[i-1] + num[i], num[i]);$

最后还要记一下当前的 dp 值是哪个区间加起来得到的。。

: <http://paste.ubuntu.com/25528462/>

HDU 1005

: <http://acm.hdu.edu.cn/showproblem.php?pid=1005>

矩阵快速幂

交了两发才 A

最近水题做的不少啊

水题伤身

如果不是因为晚上敲键盘不尽兴还怕吵到室友睡觉, I would very like to write this some more times...

今天先这样吧

以及 Google 要停止 Google Drive 是什么鬼决策

马的我才刚开始用啊

: <http://paste.ubuntu.com/25528730/>

HDU 5269

: <http://acm.hdu.edu.cn/showproblem.php?pid=5269>

把二进制从低位到高位插入字典树.., 每当出现和前面插入的数字有不同时就加上 $2^n * m$, n 表示当前插入到第几位, m 表示插入当前的整数之前有多少个数在当前位出现不同。

节点里面应该用 `cnt[2]` 来存所有经过这个节点的当前位为 `index` 的数字个数。(`index = 0, 1`)
多组输入所以要在每次输入之前 `delete`。
之前之所以错是因为在要使用“当前位为 `index` 的数字个数”时用的是当前位的父亲节点的 `cnt`。
于是 `de` 了好久 bug

//再也不想看这个题了

: <http://paste.ubuntu.com/25620640/>

HDU 2846

: <http://acm.hdu.edu.cn/showproblem.php?pid=2846>

输入 10000 个模板串(长度不超过 20)，然后 100000 次询问，每次查找输入的文本串是哪些模板串的子串。由于一个字符串的子串可以看作该字符串某个后缀字符串的前缀，则只要把每个模板串及他的所有后缀都插入到字典树中即可。为了防止出现在插入字符串“add”时会在第一层插入两次 d 的情况，每个节点保存最近一次插入到此节点的字符串的标记，若下次插入时，插入标记和节点里存的字符串的标记相同则节点的 `cnt` 不增加；否则 `++cnt`；

: <http://paste.ubuntu.com/25638053/>

HDU 2222

: <http://acm.hdu.edu.cn/showproblem.php?pid=2222>

AC 自动机模板题。。我写了快 10 遍了还是不能一遍编译过...算了直接存下这个模板先...

然后去做几道题

马的最近总让自己或别人失望

可能我有些自卑吧/扶额

: <http://paste.ubuntu.com/25639423/>

HDU 2896

: <http://acm.hdu.edu.cn/showproblem.php?pid=2896>

AC 自动机模板题..吧.....

前几次 T 了感觉很莫名其妙，debug 到最后都写了第二个版本了才发现那几次 T 是因为吧 `Get_fail` 函数放进了 `Automation` 函数里面，于是每次查询都做一遍无意义的 `Get_fail`...不过第一个版本后来又莫名 MLE，到现在还不知道为什么...哎宿舍里深夜撸码体验好差，室友在睡觉于是敲键盘不敢出声，而且这 TM 蚊子叮了我快一个小时了.....现在是 1:07AM，我写了一遍 HDU 3065 结果 WA 了...睡吧

下面是 2896 的代码。

对了，听说国庆期间 HDU 要被关掉...? WTF

: <http://paste.ubuntu.com/25640292/>

HDU 3065

: <http://acm.hdu.edu.cn/showproblem.php?pid=3065>

基本知道 AC 自动机怎么写了把...这题要流氓啊他不说是多组输入，我说怎么刚交上去就直接给我 WA...到网上查了一下才知道是多组。改了多组立刻过了，马的。

: <http://paste.ubuntu.com/25643153/>

CodeForces 864C

: <http://codeforces.com/problemset/problem/864/C>

纯模拟就行了，一开始写了很多无意义的(面向答案的)特判，结果搞得思路越来越乱。其实在模拟过程中判断就完全可以解决问题。

对了第一次 WA 在了 test8，是因为数据范围中 a 是 $1e6$ ，k 是 $1e4$ ，而我要用 $a*k$ 于是就爆了。注意就算是“`long long p = a * k`”这种写法也会溢出，必须把 a 和 k 也改为 `long long` 型。

: <http://paste.ubuntu.com/25652710/>

少做水题.....

CodeForces Gym 100801-J

: <http://codeforces.com/gym/100801/problem/J>

线段树+dp，二分答案

写第一发时，先是在输入部分维护单调栈时搞错，会越界而且维护错了；然后发现在 dp 部分出现了问题， $dp[i]$ 表示走到第 i 站所需要的最短时间。于是有 $dp[i] = \min(dp[i-r], dp[i-r+1], dp[i-r+2], \dots, dp[i-1]) + re-enter[i]$ (这里 $reenter[i]$ 也就是 $cost[i]$ ，表示在第 i 站下车后到重新上车中间所花费的时间，题目里有解释这个部分)。结果我一开始时求得是从 $dp[i-r]$ 到 $dp[i]$ 的最小值，显然是错误的，因为 $dp[i]$ 被初始化成了 0，所以 $\min(\dots)$ 这部分在一定区域内总是 0，而且我一开始还把当 $i-r < 1$ 的情况给 `continue` 掉了，于是 dp 出了错。一开始写了个大暴力，唐汉林帮我把题目交到湖南大学 OJ 上 (这个 OJ 的评测机不知道是什么年代的机器，慢到爆炸，基本数据一大就超时，后来我交到 CF 上第一版只用了 317ms，第二版快了几十 ms，在湖南大学 OJ，大些的数据就直接 2000+ms 被 T 掉)，发现并没有数据 WA，不是 T 就是 AC，于是我把最外层暴力改成了二份答案，写完交上去结果 WA 了。最后终于发现我在线段树单点更新时爆了 `int`，调用函数时传进的参数是 `long long`，但是函数定义中的参数类型是 `int` = =。

就很烦

还好了...后来带着清晰的思路写了第二发，比第一版的代码短了几十行而且在 CF 上比第一发快了几十 ms。(在湖南大学 OJ 仍然 T 在 test 12。

: <http://paste.ubuntu.com/25686873/>

CodeForces 879C

: <http://codeforces.com/problemset/problem/879/C>

Bitmasks

给出 n ($n \leq 5e5$) 个 “ $|$ & \wedge ” 的对 $0 \sim 1023$ 中所有常数的位运算操作，要求将这些操作简化到 5 步以内，运算结果不变。

对于每一位，操作前后关系共有四种情况：

1. $1 \rightarrow 1, 0 \rightarrow 0$; `///(no operation)`
2. $1 \rightarrow 1, 0 \rightarrow 1$ `/// (| 1)`
3. $1 \rightarrow 0, 0 \rightarrow 0$ `/// (& 1)`
4. $1 \rightarrow 0, 0 \rightarrow 1$ `/// (^ 1)`

于是设定两个值 $v1 = 1023$ ， $v2 = 0$ ，用他们分别去做以上的 $5e5$ 次操作，通过比较两者每一位上的关系得到答案。用三个数 a, o, x 分别记录简化后的用于 &、|、^ 的数字。

: <http://paste.ubuntu.com/25837408/>

HDU 2457/POJ 3691

: <http://acm.hdu.edu.cn/showproblem.php?pid=2457>

<http://poj.org/problem?id=3691>

给出 n 个模板串和另一个字符串，要求修改所给字符串使串中不包含任何模板串，求最少修改次数。

● dp + 静态 AC 自动机

将问题转化为了：生成一个字符串，使得所生成的字符串中不包含任何模板串，求生成字符串与所给字符串的最小差异。

1. 建立静态 AC 自动机：

a) root 节点 id 为 1，动态 AC 自动机中的 NULL 即为 0，所有指针初始指向 0。

b) 建立 fail 指针时，先把 root 的 fail 指向自己，然后遍历 root 的所有子节点，如果存在子节点 (即 $\text{nxt}[\text{root}][i] \neq 0$) 则该节点的 fail 指向 root 并入队，否则 ($\text{nxt}[\text{root}][i] == 0$ 时) 使 $\text{nxt}[\text{root}][i] = \text{root}$ 然后 while($\text{que.size}()$) 循环，遍历 $\text{que.front}()$ 的所有子节点，若子节点存在则使该子节点的 fail 指向当前节点 fail 的对应子节点 ($\text{fail}[\text{nxt}[\text{now}][i]] = \text{nxt}[\text{fail}[\text{now}][i]]$)，否则 (子节点不存在) 使该子节点指向当前节点 fail 的对应子节点 ($\text{nxt}[\text{now}][i] = \text{nxt}[\text{fail}[\text{now}][i]]$)，因为若 now 的 fail 有对应子节点，则指向该子节点，否则由于 now 在入队前执行过相同的操作，于是显然有该子节点现在指向 root，即使得了当前节点子节点指向 root)，而且每次取 $\text{que.front}()$ 时，使当前节点的 finish 标记与当前节点 fail 的 finish 标记取或 (||)，即若当前节点所在串中包含了某个不是自身的模板串，当前节点也要标记为某一模板串的 finish。

2. 动态规划：

$\text{dp}[i][j]$ 表示长度为 i ，在自动机的节点 j 处结束的串，所需要的最小修改次数。

首先初始化 $\text{dp}[\text{maxn}][\text{maxn}]$ 为 INF，然后将 $\text{dp}[0][\text{root}]$ 赋值为 0 (即长度为 0，在自动机 root 节点处结束的生成串，最小修改次数为 0)。然后对于给出字符串的每一位 i ，遍历自动机上的所有节点 j ，如果

$\text{dp}[i][j]$ 存在 (即 $\text{dp}[i][j] < \text{INF}$) 对于每个节点再遍历其每个子节点 k ，下一状态

$\text{nextStatus} = \text{nxt}[j][k]$ ，若 $\text{finish}[\text{nextStatus}]$ 为 true，则无法转移，跳过此次循环，否则进行状态转移，状态转移方程为

$$\text{dp}[i+1][\text{nextStatus}] = \min(\text{dp}[i+1][\text{nextStatus}], \text{dp}[i][j] + (k \neq s[i]));$$

即在 nextStatus 节点处结束的长度为 $i+1$ 的串，其最少修改次数为 $\min(\text{dp}[i+1][\text{nextStatus}], \text{dp}[i][j] + (k \neq \text{idx}(s[i])))$ ，即从 $\text{dp}[i][j]$ 转移到 $\text{dp}[i+1][\text{nextStatus}]$ 时，若转移时生成的新一位字符与原串中的对应位的字符相同，则转移时不用修改，否则 (生成的新一位与原串中对应位不相同的话则) 修改次数加一。

: <http://paste.ubuntu.com/25876613/>

附静态 AC 自动机模板 (以后还会补充): <http://paste.ubuntu.com/25876618/>

HDU 3336

: <http://acm.hdu.edu.cn/showproblem.php?pid=3336>

方程: $\text{dp}[i] = (\text{dp}[i-1] + \text{jmp}[i] + 1) \% \text{mod};$

答案即为 $\text{dp}[n]$. 其中 $\text{jmp}[i]$ 为第 i 位经过几次 $\text{nxt}[]$ 的回溯可以回到字符串起点，在 getnxt 时处理出。

: <http://paste.ubuntu.com/26151437/>

ZOJ 3993 -----

19:34
星期三
2018/5/2

: <http://acm.zju.edu.cn/onlinejudge/showProblem.do?problemCode=3993>

缩圈，初始时有一个半径为 R 的大圈圆心在 $(0, 0)$ ，下一次会缩到一个半径为 r ($r \leq R$) 的小圈里，满足小圆 r 一定完全包含在 R 内 (可内切) 以及给定 n 个点的坐标 (x, y) ，求缩圈后仍在圈内的概率最大的所有的点在输入时的 index

1. $2r \leq R$ 时

在半径为 $R - 2 * r$ 的圆内的所有点有相等且最高概率，在圆外随其距 $(0, 0)$ 的距离增加，相应点仍在圈内的概率单调减小。

2. $2r > R$ 时

在半径为 $2 * r - R$ 的圆内的所有点有相等且最高概率，在圆外随其距 $(0, 0)$ 的距离增加，相应点仍在圈内的概率单调减小。

Contest 的时候一直 WA 了 8 发后来发现是把平方和给输出了 (要输出的是该点是第几个输入的)/扶额
下次要把变量名的长度也分开不然 debug 都找不出

//为什么我做的题越来越水啊靠 这题放去年这会我也能做出来了

: <https://vjudge.net/solution/13759532>

HDU 3518-----

23:39
星期二
2018/5/8

: <http://acm.hdu.edu.cn/showproblem.php?pid=3518>

多组输入，每组长度 $len \leq 1000$ 的字符串，求其中出现次数为 2 次或以上的串的个数。(串不互相重叠)

后缀数组 & Height 数组

发现之前一直把 sa[] 和 rk[] 的名字写反了

其实还能优化掉一个 log

我写不粗来了然后同一份代码写了 4 次

: <https://vjudge.net/solution/13844164>

UOJ 0002-----

20:05
星期一
2018/5/28

: <http://uoj.ac/problem/2>

因为对每一位上初始只有两种情况 0 or 1, 于是预处理出每一位上初始是 0 或 1 时经过所有防御门后的结果，然后从最高位到最低位遍历，在所有小于 m 的初始值中优先取答案更大的，若取 0 和 1 后答案相同，则取 0，这样后面的所有位置都可以自由选择 0 或 1 了。

注意答案可以比 m 大。

.....垃圾题

: <http://uoj.ac/submission/253195>

BZOJ 1050-----

15:50
星期二
2018/5/29

: <https://www.lydsy.com/JudgeOnline/problem.php?id=1050>

好像把 Kruskal 给忘了 emmm

这里就是给所有边排序后从小到大每条边加进生成树，结束后第一条加进树的边是分子，最后一条加进树的边是分子，然后从小到大枚举分母的取值取这个过程中最小的分数，如果 s 和 t 不连通即为 IMPOSSIBLE。

(原来 kruskal 加双向边只需要加一次啊如果加两次边就会 T)

UOJ3 不会做网上说要 lct(?) 去查 lct 又出现了 splay(?) 他们说这题也能 lct 做我要学

洗衣服去

: <https://blurrngy.wordpress.com/2018/05/29/503/>

HDU 5044-----

20:19
星期三
2018/6/6

🔗: <http://acm.hdu.edu.cn/showproblem.php?pid=5044>

这题一开始 T 了我还以为是树链剖分写坏了，优化不出来，去查题解发现是线段树 T 了

.....
既然是每次区间加减，最后一次全部输出
所以
只要每次区间[1] += val, [r+1] -= val
最后全输出就好了
Oh shii'

/*****/

先不管边权点权把树链剖出来，然后维护两个数组 `nodev[]` 和 `edgev[]`，分别记录区间修改的延迟标记，把边权下降到儿子节点上(即每个节点上的边权代表该节点到该节点的父亲的边的权)，点权就正常作为点权。更新后根据刚才的延迟标记把答案数组处理出来最后输出就好。

以及用" \n"[i == n]会 PE 惊了
后面还忘关文件读入一直 WA
惊了
太
菜
了
是个裸题吧
网上有人加了读入挂才过。。

📄: <https://vjudge.net/solution/14201585>

BAPC2014 A – Choosing Ice Cream -----

10:46
星期二
2018/7/10

🔗: <https://nanti.jisuanke.com/t/28201>

共有 n 种 ice cream，给一个有 k 面的骰子($1 \leq n, k \leq 1e9$)，要通过 x 次掷骰子来实现等概率的决定选择哪一种 ice cream to 吃。

样例 $n = 4, k = 2$ 解释:

记四种 ice cream 分别为 a, b, c, d ，掷两次骰子，00 则选择 a ，01 则选择 b ，10 则选择 c ，11 则选择 d 。

显然掷 x 次骰子将会产生 k^x 种等概率的情况，若 $k^x \% n = 0$ ，则可以实现每种 ice cream 等概率选取；若 $k^x \% n \neq 0$ ，则无论 x 为多大都不能实现。

可以分解 n 和 k 的质因数并记录每个质因数的个数，若 n 的质因数种类 $\in k$ 的质因数种类，则 k^x 的质因数个数每个为 k 的质因数个数的 x 倍。求出使得 k 的对应质因数个数全部 $\geq n$ 的质因数个数的 x 。

也可以直接循环每次计算 $k^x \% n$ ，若 $k^x \% n = 0$ 则 break 输出 x ，循环次数最多为 $\log_2 x \leq 29$ (因为 x 每个质因数的个数最多也为 $\log_2 x$)，控制循环次数即可。

📄: <https://paste.ubuntu.com/p/Mw9X6cPHDh/>

BAPC 2014 FINAL I – Interesting Integers-----

11:10
星期三
2018/7/11

🔗: <https://nanti.jisuanke.com/t/28319>

序列满足条件 $G[i] = G[i - 1] + G[i - 2]$, ($G[1] = a, G[2] = b, a \leq b, a, b > 0$)，给出 G 中的某一项的值 $n(n < 1e9)$ ，求出包含该项的序列 G 的 a 和 b ，要求在保证 b 最小的前提下，使 a 也最小。

🔗: 记 $(G[1], G[2]) = (1, 1)$ 的序列为 $base$ ，则对每个 G 的前两项 $G[1], G[2]$ 有

$$(G[1], G[2]) = (1, 1) + x * (0, 1) + y * (1, 1),$$

其中 $G[1] = a, G[2] = b$, 于是有

$$\begin{cases} 1 + y = a, \\ 1 + x + y = b \end{cases}$$

$$\text{解得} \quad \begin{cases} x = b - a, \\ y = a - 1 \end{cases}$$

观察序列显然有:

对 $base$ 前两项增加 x 个 $(0, 1)$ 将造成

$$G[i] - base[i] = x * base[i - 1],$$

对 $base$ 前两项增加 y 个 $(1, 1)$ 将造成

$$G[i] - base[i] = y * base[i],$$

于是 x 个 $(0, 1)$ 和 y 个 $(1, 1)$ 对 G 的影响为

$$\begin{aligned} G[i] - base[i] &= x * base[i - 1] + y * base[i], \\ \text{即} \quad G[i] &= x * base[i - 1] + (y + 1) * base[i], \end{aligned}$$

带入 x, y 有

$$\begin{aligned} G[i] &= (b - a) * base[i - 1] + a * base[i] \\ \text{即} \quad G[i] &= b * base[i - 1] + a * base[i - 2] \end{aligned}$$

其中 $G[i] = n$. 即

$$n = b * base[i] + a * base[i - 1] \dots\dots\dots (*)$$

☞: 于是考虑先把 $base$ 打表到 $1e9$, 对给出的 n 找出第一个小于等于它的数 $base[pos]$ 后, 从 pos 开始从大到小枚举 $base[i]$, 对每个 $base[i]$ 从 1 从小到大枚举 b , 由 $(*)$ 式可解得

$$a = \frac{n - b * base[i]}{base[i - 1]}$$

若解出的 $a > b$ 则枚举下一个 b , 若解出的 $a \leq 0$ 则枚举下一个 $base[i]$, 找到的第一个满足 $(*)$ 式的 (a, b) 即为答案。

☂: 我怎么写的这么麻烦

有人说直接把给出的 n 作为 $G[i]$, 将 $n * 0.618$ 作为 $G[i - 1]$, 然后逐次互相减到最后就是答案了这什么骚操作

📄: <https://drive.google.com/open?id=1HAQbyhHT070jhcbnqzwHUdW4uuoJqESP>

BAPC 2014 Preliminary D – Lift Problems -----

10:30
星期一
2018/7/16

🔗: <https://nanti.jisuanke.com/t/28204>

所有人从 0 层登上一部电梯, 电梯统计每层楼有多少个人要去, 要去第 i 层的人数为 $s[i]$ 。

给出规则-若电梯在第 i 层停靠:

① 对于每个 $j < i$, 产生的愤怒值为 $s[j] * (i - j)$;

② 对于每个 $j > i$, 产生的愤怒值为 $s[j]$ 。

电梯只能向上运行, 求电梯运行所能产生的最小愤怒值。

🐶: dp.

认为 $dp[i]$ 表示电梯在第 i 层停靠时, 在第 i 层所能产生的最小愤怒值。有

$$dp[i] = \min_{j=0}^{i-1} \left\{ dp[j] + \sum_{k=j+1}^{i-1} (s[k] * (i - k)) + \sum_{k=i+1}^n s[k] \right\}.$$

// 两种 sum 都要前缀和

📄: <http://codeforces.com/contest/1/submission/40393331>

🔗: <https://nanti.jisuanke.com/t/28315>

给出 n 个人，每个人有三种能力 A, B, C ，每种能力的评级(rank)数字越小，认为该能力越强（如 $\text{rank}A=1$ 的人的 A 能力强于 $\text{rank}A=2$ 的人）。如果第 i 个人的 A, B, C 三种能力都比第 j 个人弱，则第 i 个人被淘汰。给出人数 n 和每个人的三种能力等级，求最终有多少人没有被淘汰。

考虑求被淘汰的人数，用总数 n 减去被淘汰的人数即为答案。

📦: 对 n 个人，先根据 A 排序，然后按 A 的 rank 从小到大遍历，以 $\text{rank}[i].B$ 为下标，以 $\text{rank}[i].C$ 为值建立权值线段树。遍历到第 i 个人时在 $a[\text{rank}[i].B]$ 插入 $\text{rank}[i].C$ ，维护区间最小值，若区间 $[1, \text{rank}[i].B - 1]$ 的最小值小于 $\text{rank}[i].C$ ，则当前遍历到的人（第 i 个人）被淘汰。计数即可。

📄: <http://codeforces.com/gym/101512/submission/40396094>

2018 ICPC-CCPC(?) (Ningxia) D – Take Your Seat-----

20:16
星期四
2018/7/19

🔗: <https://nanti.jisuanke.com/t/28404>

• 问题 1:

有 n 位乘客上飞机，飞机上有 n 个座位，编号为 i 的乘客的座位号也为 i 。这时编号为 1 的乘客弄丢了机票。现在 n 位乘客按照乘客编号 $1 \sim n$ 的顺序上飞机，乘客 1 由于丢失机票，会随机坐在飞机上任意位置，其他乘客按照自己的机票找到自己的对应位置。如果有一位乘客发现自己的座位被占，这位乘客也会随机选择飞机上剩余的位置。

求第 n 位乘客做到座位号为 n 的座位上的情况的概率。

• 问题 2:

有 m 位乘客上飞机，飞机上有 m 个座位，编号为 i 的乘客的座位号也为 i 。这时编号为 1 的乘客弄丢了机票。现在 m 位乘客随机顺序上飞机，乘客 1 由于丢失机票，会随机坐在飞机上任意位置，其他乘客按照自己的机票找到自己的对应位置并坐下。如果有一位乘客发现自己的座位被占，这位乘客也会随机选择飞机上剩余的位置。

求第 m 位乘客做到座位号为 m 的座位上的情况的概率。

🔗: 问题 1:

记 n 个乘客时的概率为 p_n ，显然有

$$p_1 = 1, p_2 = \frac{1}{2}$$

对于 n 个人的情况：首先，第一个人坐在每个座位上的概率是相等的，均为 $\frac{1}{n}$ 。

- 若第一个人坐在 1 号座位上，必定每个人都坐在自己对应的座位上，第 n 个人也必定坐在第 n 个座位上，概率为 $\frac{1}{n} \times 1 = \frac{1}{n}$;

- 若第一个人坐在 n 号座位上，则第 n 个人必定不在第 n 个座位上，概率为

$$\frac{1}{n} \times 0 = 0;$$

- 若第一个人坐在 $k, k \in [2, n-1]$ 号座位，那么座位号在 $[2, k-1]$ 区间内的每个座位上的乘客都坐对了位置。于是问题转化为 $n-k+1$ 个人时的初始情况，这时 k 号乘客相当于 $n-k+1$ 位乘客时的 1 号乘客。概率为

$$\frac{1}{n} \times p_{n-k+1} \cdot$$

则有总概率：

$$p_n = \frac{1}{n} \sum_{k=1}^{n-1} p_k = \frac{1 + \frac{1}{2}(n-2) + 0}{n} = \frac{1}{2}$$

问题 2：

记 m 个乘客时的概率为 q_m ，显然有

$$q_1 = 1$$

与问题 1 的区别仅在于所有乘客**随机顺序**登上飞机。即相当于**随机 1 号乘客登机时间**。

- 若 1 号乘客第 m 个登机，则第 m 位乘客必定坐对了位置，概率为

$$\begin{aligned} \frac{1}{n} \times 1 &= \frac{1}{n} \\ &= \frac{1}{n} p_1 \end{aligned}$$

- 若 1 号乘客第 k 个登机，问题此时转化为 $m - k + 1$ 个人时的**问题 1**。概率为

$$\frac{1}{n} \times p_{m-k+1}$$

于是总概率为：

$$q_m = \frac{1}{m} \sum_{k=1}^m p_m = \frac{1 + \frac{1}{2}(m-1)}{m} = \frac{m+1}{2m}$$

即

$$\begin{cases} p_n = \frac{1}{2} \\ q_m = \frac{m+1}{2m} \end{cases}$$

GCPC 2018 D – Down the Pyramids

14:18
星期六
2018/7/21

🔗: <https://nanti.jisuanke.com/t/28855>

📖: 水题

🔍: 一直卡最后绝望到把全部 `int` 改成了 `long long` 就过了???

然后用评测机 `debug` 发现，并不会爆 `int`

?

但是改成 `int` 还是给我 WA

📄: <http://codeforces.com/contest/1/submission/40597245>

GCPC 2018 C – Coolest Ski Route

16:05
星期六
2018/7/21

🔗: <https://nanti.jisuanke.com/t/28854>

🔍: 对每个入度为 0 的节点 `dijkstra`，同一个样例中不要每个入度为 0 的节点都初始化

📄: <http://codeforces.com/contest/1/submission/40600271>

HDU-6298 Maximum Multiple

15:34
星期六
2018/7/28

🔗: <http://acm.hdu.edu.cn/showproblem.php?pid=6298>

🔍: 输入 n ，要求找出一个三元组 x, y, z 满足 $x|n, y|n, z|n$ ，且 xyz 最大。（其中 $a|b$ 表示 a 为 b 的因子）

$$\star: 1 = \frac{1}{3} + \frac{1}{3} + \frac{1}{3} = \frac{1}{2} + \frac{1}{4} + \frac{1}{4} = \frac{1}{2} + \frac{1}{3} + \frac{1}{6}$$

且

$$\frac{1}{3} \times \frac{1}{3} \times \frac{1}{3} = \frac{1}{9} > \frac{1}{2} \times \frac{1}{4} \times \frac{1}{4} = \frac{1}{16} > \frac{1}{2} \times \frac{1}{3} \times \frac{1}{6} = \frac{1}{36}$$

📄: <http://acm.hdu.edu.cn/viewcode.php?rid=25528010>

HDU-6299 Balanced Sequence-----

10:15
星期一
2018/7/30

🔗: <http://acm.hdu.edu.cn/showproblem.php?pid=6299>

📖: “平衡序列（只含左右括号字符）”定义：

- 空串是平衡序列
- 如果串 A 和串 B 都是平衡序列，则串 AB 是平衡序列
- 如果串 A 是平衡序列，则串“(A)”是平衡序列

给出 n 个括号序列，对它们重排并前后相接，求整个序列可获得的最长平衡序列（不要求连续）

🔗: 每个括号序列中，已经匹配了的括号对最终结果的贡献显然是确定的。首先对每个序列预处理，把其中已经匹配了的括号去除，得到的一定是 a 个 ‘(’ 加上 b 个 ‘)’ 的序列。于是对每个括号序列可以得到一个 $pair(r, l)$ ，表示前面是 r 个右括号，后面是 l 个左括号的序列。

贪心：要尽可能多地匹配其中的左右括号，即使

- 排在前面的括号序列 r 小 l 大
- 排在后面的括号序列 r 大 l 小

按照 $l - r$ 的符号分类：

- $l - r > 0$ ，按 r 从小到大排序
- $l - r = 0$ ，无所谓
- $l - r < 0$ ，按 l 从大到小排序

排序后再线性从前到后扫一遍得到答案。

📖: 一开始是按 r 从小到大，若 r 相等时按 l 从大到小排序，可以被 hack：

```
3
)))(((
)(
))()
```

📄: <http://acm.hdu.edu.cn/viewcode.php?rid=25556330>

HDU-6300 Triangle Partition-----

11:20
星期一
2018/7/30

🔗: <http://acm.hdu.edu.cn/showproblem.php?pid=6300>

📖: 极角排序后顺序输出即可

为了防止精度问题排序比较用

$$x_2 y_1 < x_1 y_2$$

乘法爆 int 🐼

📄: <http://acm.hdu.edu.cn/viewcode.php?rid=25559132>

HDU-6301 Distinct Values-----

13:34
星期一
2018/7/30

🔗: <http://acm.hdu.edu.cn/showproblem.php?pid=6301>

📖: 如果区间 A 包含区间 B 则区间 B 可以忽略。
完全不卡时间的
(吧)

📄: <http://acm.hdu.edu.cn/viewcode.php?rid=25560476>

HDU-6304 -----

11:33
星期二
2018/7/31

🔗: <http://acm.hdu.edu.cn/showproblem.php?pid=6304>

📖: 序列递推式为

$$a_n = \begin{cases} 1 & , n = 1, 2 \\ a_{n-a_{n-1}} a_{n-1-a_{n-2}} & , n \geq 3 \end{cases}$$

输入 n 求 $\sum_{i=1}^n a_i$, $n \in [1, 1e18]$.

📖: 先按递推式打表得到

1, 1, 2, 2, 3, 4, 4, 4, 5, 6, 6, 7, 8, 8, 8, 8, 9, 10, 10, 11, 12, 12, 12, 13, 14, 14, 15, 16, 16, 16, 16, 16, 17, 18, 18

找规律吧然后就

Through observation we can easily(?) come to this conclusion: 除了第一项 $a[1] = 1$ 以外, 后面的序列相当于多个等差数列 $1, 2, 3, 4, 5, \dots$, $2, 4, 6, 8, 10, \dots$, $4, 8, 12, 16, 20, \dots$, $8, 16, 24, 32, 40, \dots$, ... 合并之后 sort 所得的序列。

则只要能由 n 算出 a_n , 然后即可等差数列求和算出所求的 sum.

记 $a_1 = d = k$ 的等差数列为 seq_k . 若已知 a_n , 则在 a_n 之前 (包括 a_n , 并假设 n 为序列中与 a_n 相等的数字的最后一个的下标), seq_k 中的元素出现了项数为 a_n/k .

则若已知 a_n , 可得

$$n = 1 + \sum_{k=1}^{k \leq n} \frac{a_n}{k} \quad (\text{因为第一项 } a_1 = 1)$$

则可以利用上式二分, 根据 n 求出 a_n . 复杂度 $O(\log^2 n)$

求出 a_n 后, 对 seq_i , 首项和公差 $a_1 = d = 2^{i-1}$, 末项为 $a_n - a_n \% d$, 项数为 $\frac{a_n}{d}$. 分别求和即可。注意要加上

$a_1 = 1$.

📖: 求和公式中的“ \div ”要用 $\times inv(2)$ 代替。

每次乘法和每次加法都给我加上 $\%mod$ 。

📖 的

📄: <http://acm.hdu.edu.cn/viewcode.php?rid=25581483>

HDU-6305 RMQ Similar Sequence -----

13:18
星期五
2018/8/3

🔗: <http://acm.hdu.edu.cn/viewcode.php?rid=25663035>

📖: 称两个等长序列互为“RMQ Similar Sequence”, 当且仅当两者的笛卡尔树同构。

即离散化后的序列完全相同。(如果序列中有重复元素, 对每组重复元素, 原序列中下标越小的元素离散化后的值越小, eg: $[5, 2, 2, 4]$ 离散化后为 $[4, 1, 2, 3]$).

给出序列 A, 现在生成序列 B 的方法为: B 中每一个元素都在 $[0, 1]$ 中随机等概率选择 (实数)。当 B 与 A 互为“RMQ Similar Sequence”时, 序列 B 的价值定义为 $\sum b_i$, 否则 B 的价值为 0.

求序列 B 的价值期望。(mod $(1e9+7)$)

📖：全世界题解都在写笛卡尔树同构这几个字，可能如果每次查询序列最值会 T 吧

如果维护的两个值中某一个已经有序，笛卡尔树可以线性建立。

笛卡尔树就是维护两个值 $val1$, $val2$ ，且满足：

- 如果仅看 $val1$ 则笛卡尔树是一棵搜索树
- 如果仅看 $val2$ 则笛卡尔树是一个堆。

的一棵树。

对这道题，笛卡尔树维护的是序列中每个元素的下标($val1$)和每个元素的值($val2$)。直观感受就是把序列中的下标为 x 轴，元素值为 y 轴弄个条形统计图，然后条形最高的就是笛卡尔树的 $root$ ，然后每个节点的左孩子就是其左边序列中最高的条形，右孩子同理。拎起来就是一棵笛卡尔树了。

//WTF，“拎”这个字原来是前鼻音吗

显然序列每个元素的下标是有序的。

然后按元素值建大顶堆（元素值相同的两个元素，下标小的在上面）。

用一个 $root$ 表示笛卡尔树的超级根， $root$ 维护的下标为 0，元素值为 inf ，就保证了 $root$ 总为笛卡尔树的根。

每新来一个节点（设为 now ），由于要维护搜索树的性质，假设把这个节点作为当前树最右边节点的右孩子上，然后上溯。

用一个栈维护笛卡尔树最右边一条链，从栈顶向栈底遍历，直到找到某个节点（设为 $node$ ）的元素值大于等于 now ，就把原先 $node$ 的右孩子（设为 $node.r$ ）作为 now 的左孩子，然后把 now 作为 $node$ 的新的右孩子。

这样笛卡尔树建完了就

/*****

以上是笛卡尔树

下面是题意的转换

/*****

由于 B 的价值的定义，如果 B 与 A 不是“RMQ Similar Sequence”， B 的价值为 0，则 B 与 A 不是“RMQ Similar Sequence”的情况对最终价值期望的贡献为 0。则下面计算 B 的价值期望时不考虑 B 与 A 不是“RMQ Similar Sequence”的情况。

B 中所有元素是在 $[0,1]$ 区间中随机选取，则 B 中出现重复元素的概率为 0，即 B 中所有元素的值唯一；且 B 的所有元素的值之和的期望为

$$n * \frac{1}{2} = \frac{n}{2}$$

A 与 B 互为“RMQ Similar Sequence”的要求已经转化为了“ A 与 B 的笛卡尔树同构”，则可以知道笛卡尔树的每一个节点的值是该节点管辖范围内最大的。根节点是 n 个数中最大的，概率为 $\frac{1}{n}$ ，其他节点分别是该节点的管辖

范围(即该节点的 $size$)内最大的，概率为 $\frac{1}{size_i}$

最终的概率为

$$\prod \frac{1}{size_i} \quad (size_i \text{ 为第 } i \text{ 个节点的子节点个数} + 1)$$

于是价值期望为

$$\frac{n}{2} \times \prod \frac{1}{size_i}$$

$mod=(1e9+7)$ ，就直接 `ll ans` 初始化成 $n \times inv(2) \% mod$ ，在计算节点 $size$ 时每次乘进去 $inv(now \rightarrow size)$ 。

💡：前天就写对啦

写完发现多组数据建的树，每次初始化时并没有释放空间

然后就加上了个 `del` 函数，函数最后一行是 `delete now;`

当然 `wa` 了，这样写，每组的 $root$ 就都被我给删掉了啊

而且如果不释放空间也能 `A` 了

搞什么



📄: <http://acm.hdu.edu.cn/viewcode.php?rid=25663224>

Btw, 果然线段树 T 了
自己对拍还 WA 了
算了不改了

HDU-6315 Naïve Operations-----

10:03
星期三
2018/8/8

🔗: <http://acm.hdu.edu.cn/showproblem.php?pid=6315>

📄: b 为 1 到 n 的一个排列 ($n \leq 100,000$), a 为初始全为 0 的数组。两种操作:

- add l r 对 a 的 l 到 r 区间+1
- query l r 求和 $\sum_{i=l}^r \lfloor \frac{a_i}{b_i} \rfloor$ (向下取整)

🧠: 每次 add 操作对区间+1, 则 query 可以理解为 a 每当有某个 a_i 增加到等于 b_i 时, 该位对整体的答案贡献+1
对 b 建立一棵线段树, 维护区间最小值。

add 操作就对这棵线段树区间-1(lazy), 若这次 update 完更新区间中出现了最小值为 0 的区间, 那么一直 pushDown 到值为 0 的节点 (可以是一个或多个, 我用了 queue 完成更新), 把这个节点的值修改为 b_i , 这个节点的 ans ++, 并直接从这个节点一直 pushUp 到 root, 更新途中的 val 和 ans.

query 就区间求和 ans 即可

这题过的人好多

📄: <http://acm.hdu.edu.cn/viewcode.php?rid=25761495>

HDU-5972 Regular Number -----

14:25
星期五
2018/9/14

🔗: <http://acm.hdu.edu.cn/showproblem.php?pid=5972>

🧠: 一个长为 N 的模板串, $T[i]$ 为模板串第 i 位, 给出 $T[i]$ 上所有可能出现的字符, 和一个文本串 s, 找出 s 中所有匹配了 T 的子串并输出。

🔗: 又一个线性字符串匹配的算法名曰 Shift-And.

相当于 dp 吧

用 $rec[type_N][LEN]$ 记录每个字符出现的位置。 $rec[i][j]$ 为 1, 表示编号为 i 的字符在文本串 s 的第 j 位出现过;

然后对文本串遍历, 遍历到位置 x 时, $d[i]$ 表示 T 长度为 i 的前缀是 $s[0...x]$ 的后缀。于是转移方程为:

$$d[i] = d[i-1] \& \& (s[i] == T[i])$$

$$\text{即 } d[i] = d[i-1] \& \& rec[s[i]]$$

由于以上 d、rec 中每一位只有是/否两种情况, 则用 bitset 优化, 转移方程可写成

```
d = (d << 1 | 1) & rec[s[i]];
/*bitset 不能与(int)1 位运算, 则*/
d <= 1;
d.set(0);
d &= rec[s[i];
```

这个题用 scanf 和 printf 会 T 因为要输出的模板串太多得读入挂
一开始加了读入挂还 T, 刚发现是 s 开小了

📄: <http://acm.hdu.edu.cn/viewcode.php?rid=26272874>

1500+4__rank1763__3

就是代码写少了吧 ABCD 都是水题结果 CD 写的时候卡了半天最后 D 没写出来
终测完了再说

终测 ABC 过了

交了一发 D 然后 WA5 了

行吧

忽然好累(wtf)

Codeforces 1041D-----

0:15
星期一
2018/9/17🔗: <https://codeforces.com/contest/1041/problem/D>

📖: 飞机在高度 h 处平行于 x 轴正方向飞行，现飞机上有人乘滑翔伞，可以在任意整数坐标上跳下飞机，跳下飞机后，滑翔伞将继续与飞机同方向飞行，每秒向 x 轴正方向飞行一个单位，并且每秒下降一个单位。

又知在 n 个区间 $[l_i, r_i]$ 上有上升气流，which means 当滑翔伞在这些区间内飞行时，滑翔伞将不下降，仅向 x 轴正方向移动（仍为每秒一个单位）。

给出滑翔伞初始高度 h ，以及所有区间端点，求滑翔伞轨迹覆盖 x 轴的最长长度。

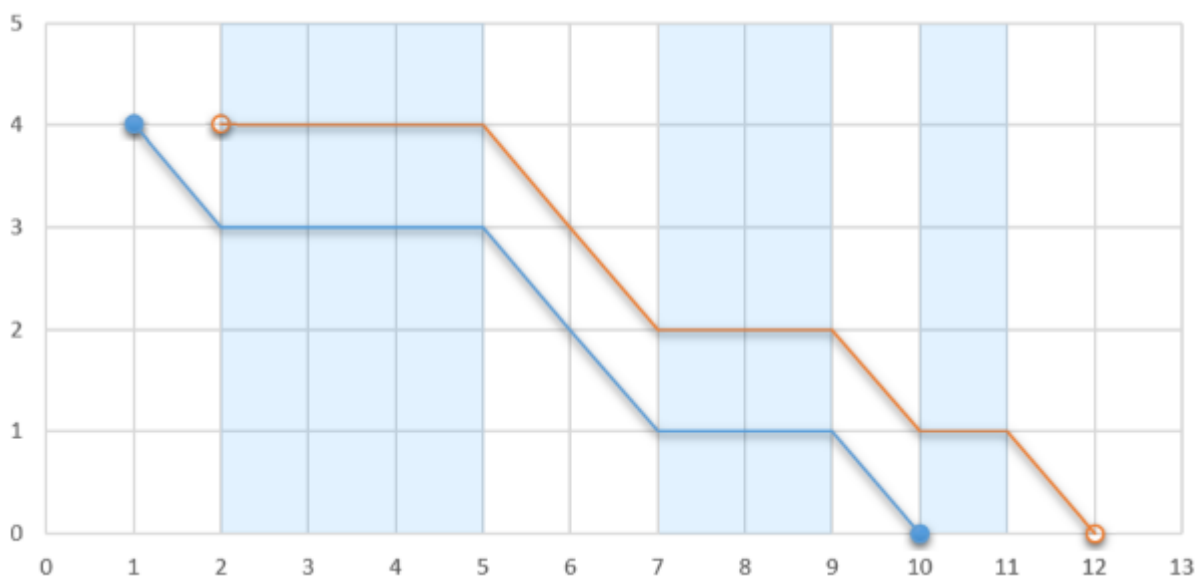
● 一个样例

input

```
3 4
2 5
7 9
10 11
```

output

10



红线为答案。

📖: 显然有：飞机处于某个上升气流的起始位置时跳下飞机，才可能有最优解。

Codeforces 1041E-----

22:18
星期三
2018/9/19

🔗: <https://codeforces.com/problemset/problem/1041/E>

🐿: 有一棵树，节点数为 n ，每个节点有标号 $1-n$

其每一条边将树的两部分连接起来，定义这条边的性质为一个二元组 (a,b) ，其中 a,b 分别为树第一部分中的节点标号最大值和树第二部分中的节点标号的最大值。

现给出树的节点数 n 和树上每条边的性质 (a,b) ，构造出一棵树使得构造出的树的每条边的性质 (a,b) 与输入的相同(顺序不限)。如果无法构造出一棵树，输出 NO。

🐿: 自己模拟了几遍样例就发现每个二元组中必定有且仅有一个元素是 n
并且如果有这样一棵树，那么这颗树一定可以被构造成一条链。

然后就很水了

每个二元组都比有一个元素为 n ，那么把输入都处理成 (n,x) 的形式然后按 x 由小到大排序。对排序后的 x 序列从左向右遍历，记录每个元素是否出现过，如果某一元素之前出现过，则将其修改为最小的未出现过的元素。

处理完的这条链就是答案了

🐿: 搞什么这题怎么比 D 题切得还快可能因为 D 题是现场做的紧张么

📄: <https://codeforces.com/contest/1041/submission/43082679>

Codeforces 235C Cyclical Quest-----

23:24
星期六
2018/10/20

🔗: <https://codeforces.com/contest/235/problem/C>

🐿: 对字符串 s 做 n 次询问，每次输入一个串 x ，求出 s 的所有子串中，与 x 循环同构的子串的个数。

$1 \leq |s|, |x| \leq 1,000,000, 1 \leq n \leq 100,000$

🐿: 对 s 建立后缀自动机，每次询问 x 时，把 x 最后一个字符删除后拼接上构成一个新串 t ，则 t 中所有长度为 $|x|$ 的子串即为 x 的所有循环同构的串。问题转化为询问串 t 中的长度为 $|x|$ 的串在串 s 中的出现次数。

🐿: 在自动机上 match 时维护当前匹配的最长长度 l ，如果匹配到某一节点， l

Codeforces Gym101775H-----

13:29
星期一
2018/10/22

🔗: <https://codeforces.com/gym/101775/problem/H>

🐿: 给出一个仅由小写字母和问号 '?' 组成的字符串 s ，将串中的 '?' 替换为任意小写字母后，限制不允许出现连续 x 个元音字母，同时不允许出现连续 y 个辅音字母。问是否有一种替换 '?' 的方式可以满足上述限制；以及是否有一种替换 '?' 的方式可以不满足上述限制。输出三种情况：

- ① DISLIKE 如果不能通过一种替换 '?' 的方式使得满足上述限制
- ② LIKE 如果不能通过一种替换 '?' 的方式使得不满足上述限制
- ③ SURPRISE 如果有至少一种方式可以满足上述限制，又有至少一种方式可以不满足上述限制

🐿: dp.

怎么说，一开始一眼看到这题是每一位上可以匹配多个字符，而且模板串是给定长度的，头脑发热就去写了 Shift-And，然后现场没调出来样例。赛后去写完了交上去 WA，然后发现可以 $aa?bb$ 3 3 就 hack 掉。于是多写了两次 shift-and 加了个特判如果元音和辅音匹配部分有重叠，那么整个匹配部分之间如果只有一个 '?' 则 DISLIKE，否则继续按照之前的办法做。交上去又 WA，然后发现可以 $aa?b?aa$ 3 3 又给 hack 掉。这不是没完了么，只要一直这么添加 $aa?b?a?b?a?b?aa$ 3 3 都是要被特判掉的情况。

并且元音和辅音完全可以被标记成 0 和 1，这样就不需要 shift-and 了。

所以这题和字符串没有一点关系（除了用了个 `strlen()` 函数？）。

Dp:

$dp[0][i]$ 表示在位置 i 处结尾的元音串的最长长度;

$dp[1][i]$ 表示在位置 i 处结尾的辅音串的最长长度。

字符串下标从 1 开始。

转移:

初始化 `memset(dp, INF, sizeof(dp));`

边界 `dp[0][0] = dp[1][0] = 0;`

如果到某位置出现了 $dp[0][i] \geq x$ 且 $dp[1][i] \geq y$ 那么该串不可能通过某种替换 '?' 的方式使得满足限制条件, 直接输出 DISLIKE, 否则贪心把所有 '?' 替换为 'a' 或 'b', 如果出现不满足限制条件的情况, 则输出 SURPRISE 否则输出 LIKE

📄: <https://codeforces.com/gym/101775/submission/44675783>

Codeforces 1091D-New Year and the Permutation Concatenation—

23:01
星期四
2019/1/3

👁: <https://codeforces.com/contest/1091/problem/D>

🌀: 对于给定的 n , 将 $[1, n]$ 的 $n!$ 种排列, 根据字典序从小到大前后连接, 会得到一个长度为 $n \times n!$ 的序列, 求出其中的长度为 n 且序列和为 $\frac{n \times (n+1)}{2}$ 的序列个数, 答案对 998244353 取模。

🌀: 对于一个长度为 n 的序列, 有两种情况:

1. 是一个 $[1, n]$ 的排列
2. 序列跨越了两个排列的空隙

故首先有 $ans_1 = n!$ 个序列满足条件, 是 $[1, n]$ 的所有排列。

下面考虑跨越空隙的情况。

将所有上面考虑过的 $n!$ 个序列各自看作整体, 将第 $i (1 \leq i \leq n!)$ 个排列记为 $x[i]$, 那么对于一个 $i = 2k + 1$, 考虑序列 $x[i]x[i+1]$ 对答案的贡献: 显然有

$$x[i][n-1] = x[i+1][n]$$

$$x[i][n] = x[i+1][n-1]$$

$$x[i][j] = x[i+1][j], \text{ 对于所有的 } 1 \leq j < (n-1)$$

那么对于每个 $i = 2k + 1$, 序列 $x[i]x[i+1]$ 对答案有贡献 $n-2$, 这样的序列个数为 $\frac{n!}{2!}$ 。故刚刚考虑的所有序列对答案的贡献为

$$ans_2 = ans_1 + (n-2) \times \frac{n!}{2}$$

再将刚才考虑过的所有形如 $x[i]x[i+1]$ 的序列看作整体 (因为根据上面的讨论, 每一个 $x[i]x[i+1]$ 序列内部对答案的贡献已经被全部考虑到了, 故可以将整个序列看作整体), 将第 $i (1 \leq i \leq \frac{n!}{2})$ 个排列记为 $y[i]$, 类似的对于一个 $i = 3k + 1$, 考虑序列 $y[i]y[i+1]y[i+2]$ 对答案的贡献:

这个序列的长度是 $3! \times n$, 即其包含了 $3! = 6$ 个 $[1, n]$ 的排列。如果只看这 6 个排列的最后三位, 他们一定是这三个数字的全排列 (共 $3! = 6$ 种情况)。而前面的 $n-3$ 位都是相同的, 即每两个排列之间对答案的贡献为 $n-3$ 。而对于每一个 y , 其内部的贡献已经被计算过了一次, 所以每个序列 $y[i]y[i+1]y[i+2]$ 对答案有额外贡献 $(n-3) \times 2$, 这样的序列个数为 $\frac{n!}{3!}$, 于是上面所有序列对答案的贡献和为

$$ans_3 = ans_2 + (n-3) \times 2 \times \frac{n!}{3!}$$

...

上面就是第一次每 $2!$ 个排列分为一组，计算组内贡献；第二次每 $3!$ 个序列分为一组，再计算组内贡献，以此类推，每 $k!$ 个序列分为一组时，每一组内的对答案有额外贡献 $(n - k) \times (k - 1)$ ，而总序列个数为 $\frac{n!}{k!}$ ，故有

$$ans_k = ans_{k-1} + (n - k) \times (k - 1) \times \frac{n!}{k!}$$

则最终答案为

$$ans_n = n! + \sum_{k=2}^{n-1} \left\{ (n - k) \times (k - 1) \times \frac{n!}{k!} \right\}$$

🐼：总结一下，就是基础值 $n!$ 加上包含排列之间的空隙的满足条件的序列个数，而后者（不是基础值那部分）要将空隙分类，每种空隙各自有一个贡献，求和即可。

📄： <https://codeforces.com/contest/1091/submission/47765091>