# 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* + 2*d*, *a* + 3*d*, *a* + 4*d*, ..., 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, ...,

contains infinitely many prime numbers

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

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* <= 9307, *d* <= 346, and *n* <= 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 106 (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。

AC: <http://poj.org/showsource?solution_id=17048825>

## 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:

D

/ \

/ \

B E

/ \ \

/ \ \

A C G

/

/

F

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>

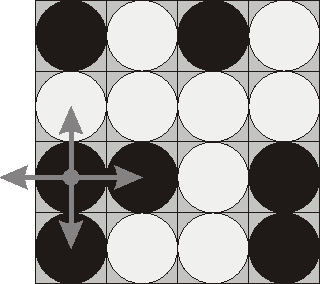
# 枚举

## 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   
wwww   
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 4х4. You can change the state of a handle in any location **[i, j]** (1 ≤ i, j ≤ 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>=1 and an integer p>= 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 nth. 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<=n<= 200, 1<=p<10101 and there exists an integer k, 1<=k<=109 such that kn = 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

3

1234

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 ) % b就不会爆long long了。

。。。。。。。。。

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，每次更新操作和问询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：

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

。。。

：<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自动机怎么写了吧...这题耍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], …, 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(…)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 <= 5e5 ) 个 ”| & ^” 的对0~1023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->1, 0->0; ///(no operation)
2. 1->1, 0->1 /// (| 1)
3. 1->0, 0->0 /// (& 1)
4. 1->0, 0->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：
   1. 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。
   2. 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nxt[root][i]不为0)则该节点的fail指向root并入队，否则(nxt[root][i] == 0时)使nxt[root][i] = root然后while(que.size())循环，遍历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(fail[nxt[now][i]] = nxt[fail[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(nxt[now][i] = nxt[fail[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：  
   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dp[maxn][maxn]为INF，然后将dp[0][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dp[i][j]存在(即dp[i][j] < INF)对于每个节点再遍历其每个子节点k，下一状态nextStatus=nxt[j][k]，若finish[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  
    dp[i+1][nextStatus] = min(dp[i+1][nextStatus], dp[i][j] + (k != 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(dp[i+1][nextStatus], dp[i][j] + (k != idx(s[i])))，即从dp[i][j]转移到dp[i+1][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：dp[i] = (dp[i-1] + jmp[i] + 1) % mod;

答案即为dp[n].其中jmp[i] 为第i位经过几次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 --------------------------------------------------

：<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 <= 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 <= 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--------------------------------------------------

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

多组输入，每组长度len<=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---------------------------------------------------

：<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-------------------------------------------------

：<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母，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://blurrgy.wordpress.com/2018/05/29/503/>

## HDU 5044--------------------------------------------------

🔗：<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[l] += 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 ----------------------------

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

共有n种ice cream，给一个有k面的骰子(1 n,k 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 次骰子将会产生 种等概率的情况，若，则可以实现每种ice cream 等概率选取；若，则无论为多大都不能实现。

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∈k的质因数种类，则的质因数个数每个为的质因数个数的倍。求出使得的对应质因数个数全部的质因数个数的x。

也可以直接循环每次计算，若则break 输出x，循环次数最多为（因为x每个质因数的个数最多也为），控制循环次数即可。

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

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

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

序列满足条件，给出中的某一项的值 ，求出包含该项的序列的和，要求在保证最小的前提下，使也最小。

👊：记的序列为，则对每个的前两项有

其中，于是有

解得

**观察序列**显然有：

对前两项增加个将造成

对前两项增加个将造成

于是个和个对的影响为

即

带入有

即

其中即

🍞：于是考虑先把打表到，对给出的找出第一个小于等于它的数后，从开始从大到小枚举，对每个从1从小到大枚举 ，由式可解得

若解出的则枚举下一个，若解出的则枚举下一个，找到的第一个满足式的即为答案。

☂：**我怎么写的这么麻烦**

*有人说直接把给出的作为，将 作为，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 -----------------------

🔗：<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。

给出规则-若电梯在第i层停靠：

对于每个，产生的愤怒值为；

对于每个，产生的愤怒值为.

电梯只能向上运行，求电梯运行所能产生的最小愤怒值。

🐕：dp.

认为表示电梯在第层停靠时，在第层所能产生的最小愤怒值。有

//两种sum都要前缀和

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

## BAPC 2014 FINAL E – Excellent Engineers-----------------------

🔗：<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rankA=1的人的A能力强于rankA=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，以为下标，以为值建立权值线段树。遍历到第i个人时在插入，维护区间最小值，若区间的最小值小于，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-----------------

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

* 问题1：

有位乘客上飞机，飞机上有个座位，编号为的乘客的座位号也为。这时编号为的乘客弄丢了机票。现在位乘客按照乘客编号的顺序上飞机，乘客由于丢失机票，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。  
 求第位乘客做到座位号为的座位上的情况的概率。

* 问题2：

有位乘客上飞机，飞机上有个座位，编号为的乘客的座位号也为。这时编号为的乘客弄丢了机票。现在位乘客随机顺序上飞机，乘客由于丢失机票，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。  
 求第位乘客做到座位号为的座位上的情况的概率。

🌰：问题1：

记个乘客时的概率为，显然有

对于个人的情况：首先，第一个人坐在每个座位上的概率是相等的，均为.

* 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；
* 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

；

* 若第一个人坐在号座位，那么座位号在区间内的每个座位上的乘客都坐对了位置。**于是问题转化为个人时的初始情况，这时号乘客相当于位乘客时的号乘客。**概率为

.

则有总概率：

问题2：

记个乘客时的概率为，显然有

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
* 若1号乘客第k个登机，问题此时转化为个人时的问题1. 概率为

于是总概率为：

即

## GCPC 2018 D – Down the Pyramids -----------------------------

🔗：<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 -------------------------------

🔗：<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 ---------------------------------

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

🍵：输入n，要求找出一个三元组满足,,,且最大。（其中表示为的因子）

❀ ：

且

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

## HDU-6299 Balanced Sequence---------------------------------

🔗：<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，得到的一定是个‘(’加上个‘)’ 的序列。于是对每个括号序列可以得到一个，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，即要使

* 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

按照的符号分类：

* ，按r从小到大排序
* ，无所谓
* ，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-----------------------------------

🔗：<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

乘法爆int 🤦‍♂️

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

## HDU-6301 Distinct Values-------------------------------------

🔗：<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 -------------------------------------------------

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

🎰：序列递推式为

输入n求 , .

🎦：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: 除了第一项以外，后面的序列相当于多个等差数列, , , , *…*合并之后sort所得的序列。

则只要能由n算出，然后即可等差数列求和算出所求的.

记的等差数列为. 若已知，则在之前（包括，并假设n为序列中与相等的数字的最后一个的下标），中的元素出现了项数为。

则若已知，可得

则可以利用上式二分，根据n求出. 复杂度

求出后，对，首项和公差，末项为，项数为.分别求和即可。注意要加上.

📦：求和公式中的“➗2️⃣”要用代替。

每次乘法和每次加法都给我加上。

🐴的

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

## HDU-6305 RMQ Similar Sequence -----------------------------

🔗：<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，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，，且满足：

* 如果仅看则笛卡尔树是一棵搜索树
* 如果仅看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和每个元素的值。直观感受就是把序列中的下标为轴，元素值为轴弄个条形统计图，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

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，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

最终的概率为

于是价值期望为

mod=(1e9+7)，就直接ll ans初始化成，在计算节点size时每次乘进去inv(now->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](http://acm.hdu.edu.cn/viewcode.php?rid=25667979)

自己对拍还WA了

算了不改了

## HDU-6315 Naïve Operations----------------------------------

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

🔋：b为1到n的一个排列()，a为初始全为0的数组。两种操作：

* 对a的l到r区间+1
* 求和

🎨：每次add操作对区间+1，则query可以理解为a每当有某个增加到等于时，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，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 ----------------------------------

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

🍪：一个长为的模板串，为模板串第位，给出上所有可能出现的字符，和一个文本串，找出中所有匹配了的子串并输出。

🔌：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]表示长度为i的前缀是s[0…x]的后缀。于是转移方程为：

即

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>

## [Contest]Codeforces #509 div.2--------------------------------

**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------------------------------------------

🔗：<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。

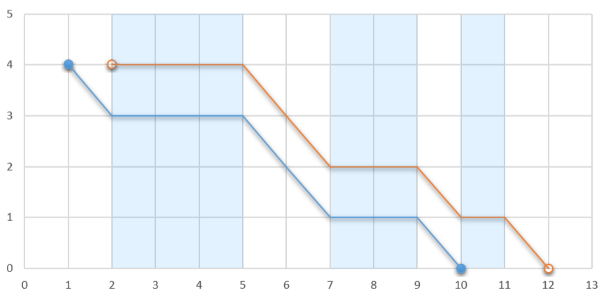
* **一个样例**

**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。

并且有：滑翔伞飞行轨高度为一个单调递减序列，则考虑二分。

于是枚举所有区间起点，二分出在这一点跳下飞机后的落地坐标pos（lower\_bound）。

对于一个起始点start和一点pos>start，有：在pos处的滑翔伞高度为

其中为l到r区间内的上升气流覆盖长度，二分+前缀和。

🤖：现场时写了尺取，尺取出“非上气流的区间和==h”的部分，最后contest结束都没改出来样例，终测完后交了一发WA了才发现这办法的细节多了点没写全。

然后看了tutorial

还是二分答案简单粗暴

题解倒是说了尺取法其实更快，这份代码跑了1000+ms

不管了睡觉奥

📄：<https://codeforces.com/contest/1041/submission/42956428>

//btw，又申请了个小号Benet，以后随便打CF了

///明天下午四点再打一场，难得不是深夜场的CF

////就拿Benet打

## [Contest]Codeforces 510 div.2--------------------------------

**1504+38\_\_\_rank827\_\_\_3**

又是想出4题写出3题，瓶颈吧，还是得多做题

A模拟

B感觉是某种dp？手动写了一堆min(min(min(min(min()))))就过了

C也挺水，就是代码中有两个部分要加同一段求一个“多余的负数的下标”，然后第一部分的细细写对了，第二部分着急就糊上去一片结果WA10，下来从头到尾读了一遍才发现原来第二部分和第一部分竟然写的不一样，写错一个变量名，改掉交了又WA10，再看刚才改的地方原来还少加了个绝对值。

然后把第一部分复制过来交上去A了

D看下面题解吧反正现场想法对了结果40min没写出来刚A掉

## Codeforces 1042D【离散化+树状数组】----------------------

🔗：<https://codeforces.com/problemset/problem/1042/D>

🖊：给出长度为的序列，其中，和一个整数求所有不同的的对数，其中 满足 .

🖥：如果序列中全为非负数就成了尺取的裸题啦，然而是可以小于0的，那么另辟蹊径。

做出前缀和，则区间 的区间和为pref[r] – pref[l]，即比较该两数之差与t的大小，即比较pref[r] – t 与 pref[l] 的大小。

那么遍历pref，对于所有的x使得pref[x] > pref[i] - t，有pref[i] - pref[x] < t，这些即为所求的区间。

问题转化为计数pref中所有“有序对”，有序对(l,r)定义为使得pref[r] – t > pref[l]的整数对。

于是类似树状数组求逆序对，开一个权值树状数组，从左向右遍历，每次将当前值x插入到树状数组中(bit[x]++)，每次求和sum(pref[r] - t + 1, tot)，其中pref[r] – t + 1是第一个满足条件的位置，tot为树状数组长度。

这题又有pref贼大，就加个离散化行了

🏺：代码能力, man.

题解竟然写了快半小时

对了这题我是撕出一棵线段树上去的反正好写而且不卡常数

刚看了题解就是以上(nice)

📄：<https://codeforces.com/contest/1042/submission/42999507>

## Codeforces 1041E-------------------------------------------

🔗：<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-------------------------------

🔗：<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。

，

📏：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到x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------------------------------------

🔗：<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：

DISLIKE 如果不能通过一种替换’?’的方式使得满足上述限制

LIKE 如果不能通过一种替换’?’的方式使得不满足上述限制

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] >= x &&（且） dp[1][i] >= 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—

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

🌏：对于给定的n，将[1,n]的 种排列，根据字典序从小到大前后连接，会得到一个长度为 的序列，求出其中的长度为 且序列和为 的序列个数，答案对 取模。

☀：对于一个长度为 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 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 个序列各自看作整体，将第 个排列记为 ，那么对于一个 ，考虑序列 对答案的贡献：显然有

那么对于每个 ，序列 对答案有贡献 ，这样的序列个数为 。故刚刚考虑的所有序列对答案的贡献为

再将刚才考虑过的所有形如 的序列看作整体**（因为根据上面的讨论，每一个 序列内部对答案的贡献已经全部被考虑到了，故可以将整个序列看作整体）**，将第 个排列记为 ，类似的对于一个 ，考虑序列 对答案的贡献：

这个序列的长度是 ，即其包含了 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 种情况）。而前面的 位都是相同的，即每两个排列之间对答案的贡献为 。而对于每一个 ，其内部的贡献已经被计算过了一次，所以每个序列 对答案有额外贡献 ，这样的序列个数为 ，于是上面所有序列对答案的贡献和为

…

上面就是第一次每 个排列分为一组，计算组内贡献；第二次每 个序列分为一组，再计算组内贡献，以此类推，每 个序列分为一组时，每一组内的对答案有额外贡献 ，而总序列个数为 ，故有

则最终答案为

🌙：总结一下，就是基础值 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>