# 2015年集训队作业试题泛做

长春吉大附中实验学校 吴一凡

December 3, 2015

## 1 August Challange 2015

## 2 July Challange 2015

# 3 June Challange 2015

# 4 May Challange 2015

## 4.1 CBAL

### 4.1.1 题目大意

定义一个字符串是平衡的当且仅当这个字符串中的所有字符能被完全分成两个相同的集合。对于一个字符串$s$，$s$的长度用$|s|$表示，$s[l,r]$表示$s$的第$l$个字符到第$r$个字符形成的字符串（$1 \le l \le r \le |s|$）。
现在给定一个长度为$n$的字符串，另有$q$组询问，每次给定一个区间$[l,r]$，再给定一个非负整数$type$，求：

$$\sum_{l \le l' \le r' \le r} |s[l',r']|^{type}[IsBalanceString(s[l',r'])]$$

要求强制在线，数据范围$n,q \le 10^5, 0 \le type \le 2$。

### 4.1.2 算法讨论

一个串是平衡的当且仅当串中的每种字符出现的次数都是偶数，一个串中每种字符的出现次数都能用两个前缀相减，因此如果一个串是平衡的，那么两个前缀中每种字符出现次数的奇偶性相同。
我们用一个$2^{26}$以内的整数存下每个前缀每种字符的奇偶性，并去重重新标号为区间$[1,n+1]$中的数，令$label_i$表示长度为$i$的前缀的标号。
那么对于询问$[l,r]$，其实也就是询问：

$$\sum_{l-1 \le l' < r' \le r} (r'-l')^{type}[label_{l'} == label_{r'}]$$

如果对于区间$[l,r]$已经有答案，我们能够方便的将区间拓展到$[l-1,r]$或者$[l,r+1]$，只需要在区间中维护每种$label$对应下标的数目、和、平方和就能够$O(1)$实现答案更新。时间复杂度$O(n\sqrt{n})$。
考虑分块，将$n+1$个前缀分成$\sqrt{n}$块，利用刚才的算法预处理每两个块之间的答案，然后再维护以每个块为结尾的前缀中每种$label$对应下标的数目、和、平方和，这样查询的时候对于整块部分我们直接调用答案，对于零散的部分我们利用预处理的前缀和信息暴力计算即可，单组询问时间复杂度$O(\sqrt{n})$。

### 4.1.3 时空复杂度

时间复杂度$O((n+q)\sqrt{n})$，空间复杂度$O(n\sqrt{n})$。

## 4.2 GRAPHCNT

### 4.2.1 题目大意

给定一张$n$个点，$m$条边的有向图，问存在多少对点$(X,Y)$，使得存在两条从

点1出发分别到$X$和$Y$的路径满足这两条路径只在点1处相交。
数据范围$1 \leq n \leq 10^5, 0 \leq m \leq 5 \times 10^5$。

#### 4.2.2 算法讨论

我们求出这张无向图以1为根的Dominator Tree，则点对$(X, Y)$合法当且仅当$X, Y$在Dominator Tree上的LCA为点1。
于是我们直接求出Dominator Tree，然后在树上进行简单的统计即可。
关于Dominator Tree的更多内容请参见李煜东的《图连通性若干拓展问题》。

#### 4.2.3 时空复杂度

时间复杂度$O(m\alpha(n, m))$，空间复杂度$O(m)$。

## 5 April Challange 2015

# 6 March Challange 2015

## 6.1 TREECNT2

### 6.1.1 题目大意

给定一棵$n$个点的边带权的树，有$q$次修改，每次修改一条边的权值，每次修改后都要询问树上有多少条路径使得路径上所有边的权值最大公约数为$1$。
数据范围$n \leq 10^5$，$q \leq 100$，任意时刻每条边的权值是$\leq w = 10^6$的正整数。

### 6.1.2 算法讨论

由于只需考虑权值的最大公约数是否为$1$，我们将所有权值的质因子的幂次都降为$1$，这样的话每个权值的约数个数最多为$2^c$个，其中$c$表示一个$10^6$以内的数最多含有的质因子数，且$c$只有$7$。
我们可以利用线性筛预处理每个数的最小质因子，这样就能在$O(\log w)$时间内完成分解。
考虑如何对于$d$求有多少条路径满足$d$整除路径上所有边的权值的最大公约数。
显然我们只需要考虑所有是$d$的倍数的边，并在每个连通块内根据有多少个点分别统计即可。
记这个答案为$f(d)$，则根据莫比乌斯反演，最终答案为$\sum f(d)\mu(d)$。
我们按照不同的$d$将图分层拆开。
有用的$d$一定是某条边的权值的约数，因此每条边最多能够提供$O(2^c)$层，每层两个点，因此所有层中点的总数是$O(2^c n)$级别的。
那么我们将这些点利用并查集维护连通块大小并合并起来，总时间复杂度为$O(2^c n\alpha(n))$。
但是还有$q$次修改，我们不妨首先处理所有没有被修改过的边，然后对于每次修改暴力枚举修改的$q$条边，按照这些边此时的权值加入到图中计算答案。我们依然利用并查集处理，注意这里的并查集是需要还原的，我们可以用一个栈记录下所有修改，然后最后将这些修改逆序还原。
时间复杂度$O(w + (n+q)\log w + 2^c n\alpha(n) + q^2 2^c \alpha(n))$。

### 6.1.3 时空复杂度

时间复杂度$O(w+(n+q)\log w+2^c n\alpha(n)+q^2 2^c\alpha(n))$，空间复杂度$O(w+2^c n)$。

# 7 Februray Challange 2015

# 8 January Challange 2015

## 8.1 XRQRS

### 8.1.1 题目大意

给定一个初始为空的序列，需要支持下面几个操作：
$0\ x$表示在当前序列的结尾插入一个数$x$。
$1\ L\ R\ x$表示在序列的$[L,R]$区间找到一个数$y$使得$x,y$的异或和最大。
$2\ k$表示删除序列的最后$k$个数。
$3\ L\ R\ x$表示询问序列$[L,R]$区间中有多少个数$\leq x$。
$4\ L\ R\ k$表示询问序列$[L,R]$区间中的第$k$小值。
数据范围：操作数目$M \leq 500000$，数字$0 \leq x \leq 500000$，保证所有询问合法。

### 8.1.2 算法讨论

我们用可持久化Trie维护序列前缀的Trie，则操作$0,2$已经能够在$O(\log x)$里处理。
对于询问$1,3,4$，我们都可以通过用两个版本的Trie树相减得到区间的Trie，再利用树上二分解决。
以询问1为例：我们从高到低考虑$x$的每一位，从根出发，我们贪心的每次走与$x$这一位不相同的数字的子树，这样最终就能得到最大的异或和。
这样询问也能在$O(\log x)$内解决。

### 8.1.3 时空复杂度

时间复杂度$O(M \log x)$，空间复杂度$O(M \log x)$。

# 9 December Challange 2014

## 9.1 RIN

### 9.1.1 题目大意

有$n$个课程，$m$个学期，每个课程都必须选择在某个学期学习，第$i$个课程在第$j$个学期学习将会得到$X_{i,j}$的价值。同时有$k$个限制条件，第$i$个限制条件给定$A_i, B_i$，要求课程$A_i$所在的学期必须在$B_i$所在的学期之前。
求所有课程价值的最大平均值。
数据范围$n, m, k \leq 100$，$0 \leq X_{i,j} \leq 100$，保证至少存在一种合法的解。

### 9.1.2 算法讨论

将每个课程拆成一条长度为$m+1$的链，分别表示第$0-m$个学期，对于课程$i$，第$j-1(1 \leq j \leq m)$个点到第$j$个点连一条边，容量为$100 - X_{i,j}$。
对于源点，连向所有链的第0个学期，容量为$\infty$。
所有链的第$m$个学期连向汇点，容量为$\infty$。
对于限制$A, B$，对于$1 \leq i \leq m$，从$A$链的第$i-1$个学期连向$B$链的第$i$个学期，容量为$\infty$，这样就能满足题目中的限制了。
于是我们求最小割，再用$100 \times n$减去就是最大的总和了。
再除以$n$就是最大平均值。

### 9.1.3 时空复杂度

时间复杂度$O(MaxFlow(n \times m, (n+k) \times m))$，空间复杂度$O((n+k) \times m)$。

# 10 November Challange 2014

## 10.1 FNCS

### 10.1.1 题目大意

给定一个长度为$n$的序列$a_i$，同时还有$n$个函数，第$i$个函数为$f_i = \sum_{j=l_i}^{r_j} a_j$，函数的值会随着$a_i$的变化而发生改变。
现在要支持一些操作，要么改变某个$a_i$的值，要么给定$l, r$，求$\sum_{i=l}^{r} f_i$。
数据范围$n, q \leq 10^5$。

### 10.1.2 算法讨论

考虑分块，将$n$个函数分成$\sqrt{n}$块，在块内维护所有函数的和函数中每个$a_i$的常系数以及块内的答案，利用打标记再扫一遍的方法，时间复杂度$O(n\sqrt{n})$。
同样用分块的方法维护序列$a_i$的前缀和，将序列分成$\sqrt{n}$块，当我们修改某个$a_i$的时候，在$a_i$后面的块打上标记，$a_i$所在的块暴力重建，时间复杂度$\sqrt{n}$，同时对于函数所在的每个块利用我们维护的$a_i$的常数$O(1)$修改，时间复杂度$O(\sqrt{n})$。
考虑如何回答询问，首先对于函数的整块直接更新答案，对于$O(\sqrt{n})$零散部分的函数，每个函数都能用$a_i$的两个前缀和相减，而我们已经维护了前缀和，可以$O(1)$计算，时间复杂度$O(\sqrt{n})$。
综上，我们就能在$O((n+q)\sqrt{n})$的时间解决此题。

### 10.1.3 时空复杂度

时间复杂度$O((n+q)\sqrt{n})$，空间复杂度$O(n\sqrt{n})$。

# 11 October Challange 2014

# 12 September Challange 2014

## 12.1 QRECT

### 12.1.1 题目大意

给定一个二维平面，共有$Q$组询问，支持插入或者删除一个矩形，或者给定一个矩形，询问目前平面上有多少个矩形和这个矩形有至少一个交点。
数据范围$Q \leq 10^5$。

### 12.1.2 算法讨论

考虑离线处理，将询问转化为给定的这个矩形与平面上的多少个矩形没有交点。
将在平面上插入的每个矩形权值设为$1$，删除的每个矩形权值设为$-1$，这样就只有插入操作了，同时将问题转化为求平面上与这个矩形没有交点的矩形的权值和。
与一个矩形没有交点的矩形可以用全部矩形减去完全在这个矩形上方的，在这个矩形下方的，完全在这个矩形左方的，完全在这个矩形右方的所有矩形权值和。但是这样四个角会被减去两次，所以还要加回来一次。
将这些问题的贡献分开考虑，四个方向的贡献可以用一个一维按时间分治解决；四个角的贡献可以用一个二维的按时间分治解决。
于是总时间复杂度$O(Q \log^2 Q)$。

### 12.1.3 时空复杂度

时间复杂度$O(Q \log^2 Q)$，空间复杂度$O(Q)$。

## 13 August Challange 2014

# 14 July Challange 2014

# 15 June Challange 2014

## 15.1 TWOCOMP

### 15.1.1 题目大意

给定一棵$n$个点的树，上面有$A, B$两种路径，每种路径不超过$m$条，其中每条路径都有一个权值。
现在可以在每种路径中各选出一些，使得$A$中被选中的任一条路径与$B$中被选中的任一条路径之间都没有交点，在此条件下最大化被选中的路径的权值之和。
数据范围$n \leq 10^5, m \leq 700$。

### 15.1.2 算法讨论

我们首先需要判断两条链之间有没有交点。
我们将链拆成两条（或一条）深度单调递减的路径，那么只需判定两条这样的路径是否有交即可。
不难发现，两条深度递增的路径有交，则必定有一条路径，其深度最小的点$x$在另一条路径上$(a, b)$。其中$a$表示另一条路径深度最小的点，$b$表示另一条路径深度最大的点。
那么只需判定是否满足$a$是$x$的祖先且$x$是$b$的祖先。
于是对两个点$x, y$，我们如何判定$x$是不是$y$的祖先呢？
我们显然可以预处理倍增数组，并倍增判断，但是存在更加优秀的算法。
我们对树进行DFS，预处理出每个点的入栈、出栈序，不妨记为$in_i, out_i$，则$x$是$y$的祖先当且仅当$in_x \leq in_y$且$out_x \geq out_y$。
这样的话，我们就能$O(1)$判定两条路径有没有交了。
我们枚举$A, B$的每条路径，若路径有交则将两条路径对应的点连边，我们能发现这个原问题其实就是一个二分图最大带权独立集问题，可以利用最大流解决。
时间复杂度为$O(n + m^2 + m^4)$。
对于网络流问题，虽然时间复杂度的上界非常高，但一般不容易达到瓶颈，所以还是能够通过的。

### 15.1.3 时空复杂度

时间复杂度$O(n + m^4)$，空间复杂度$O(n + m^2)$。

# 16 May Challange 2014

## 16.1 ANUDTQ

### 16.1.1 题目大意

给定一棵$n$个点的有根树，有$m$次询问，每次可以以某个树中的点为父亲新加一个叶子节点，也可以删除以某个点为根的子树中的所有点，也可以将以某个点为根的子树中的所有节点的权值全部加上一个数，也可以询问某个子树内所有节点的权值和。
数据范围：$n, m \leq 50000$。

### 16.1.2 算法讨论

我们维护树的DFS序，这样每棵子树中的所有节点在DFS序中都是连续一段。
但这道题目中还能加点和删除子树，我们使用Splay树动态维护DFS入栈出栈序，不难发现删除子树就是将序列中的连续一段删除，加点就是将一个节点的入栈出栈序插入他的父亲的入栈序之后。
修改和询问利用延迟标记实现即可。

### 16.1.3 时空复杂度

时间复杂度$O(m \log n)$，空间复杂度$O(n + m)$。

# 17 April Challange 2014

# 18 March Challange 2014

## 18.1 GERALD07

### 18.1.1 题目大意

给定一张$n$个点$m$条边的无向图，图中可能存在重边和自环，现在有$q$个询问，每次询问只保留标号在$[l,r]$区间内的边时，图中连通块的个数。
数据范围$n,m,q \leq 2 \times 10^5$。

### 18.1.2 算法讨论

一开始有$n$个连通块，对于一个区间中的边，那些两个端点之前没有连通的边，加入后会使得连通块的数目减少$1$。
所以我们只需知道哪些边是在加入这条边之前两个端点不连通的。
我们按照标号加入每条边，维护关于标号的最大生成树，并在加入这条边的时候，记录删掉了哪条边。
考虑区间$[l,r]$中的一条边$x$，若其弹掉了边$y$，则证明必须加入$y$这条边以及若干条之后的边，$x$的两个端点才能连通。因此，若$y < l$，则此时$x$的两个端点不连通，对答案产生$-1$的贡献；否则$y \leq l$，说明$[l,r]$中只使用$x$之前的某些边已经能让$x$的两个端点连通，对答案没有贡献。
使用Link-Cut Tree维护每个点弹掉了哪条边，对于每次询问其实就是询问区间$[l,r]$中弹掉边的标号$< l$的边的数目，可以用可持久化线段树维护。
时间复杂度$O((n+m+q)\log n)$。

### 18.1.3 时空复杂度

时间复杂度$O((n+m+q)\log n)$，空间复杂度$O(n+m\log m+q)$。

# 19 Februray Challange 2014

## 19.1 DAGCH

### 19.1.1 题目大意

给定一张$n$个点，$m$条边的有向图，保证从1号点出发能够到达图中的所有点。
随意从1号点出发开始DFS，从1开始给每一个点一个标号。给出的图中，用每个点的标号来代表原来图中的点。定义点$x$是点$y$的supreme vertex当且仅当存在一条从$x$出发到$y$的路径，使得$x$的标号小于$y$的标号，且路径上除了$x, y$两个点之外的点的标号均大于$y$的标号。定义$x$是$y$的superior vertex当且仅当$x$是$y$的所有supreme vertex中标号最小的。现在有$Q$组询问，每次给定一个标号为$P_i$的点，询问有多少个点的superior vertex是这个点。
数据范围：数据组数$\leq 10$，$n \leq 10^5$，$n-1 \leq m \leq 2 \times 10^5$，$Q \leq 10^5$。

### 19.1.2 算法讨论

首先需要对于给定的有向图进行标号小的点优先进行的DFS来还原搜索树。
不难发现superior vertex其实就是Domintor Tree中的semidominator，直接套用Lengauer Tarjan算法的部分就行了。

### 19.1.3 时空复杂度

时间复杂度$O(Tm\alpha(n,m))$，空间复杂度$O(m)$。

## 19.2 COT5

### 19.2.1 题目大意

Treap是一种二叉平衡树，维护一些节点$(k, w)$，其中$k$是键值、$w$是权值。
Treap的中序遍历要使得键值是递增的，还要保证每一个节点的权值都要小于他的父亲节点的权值。
现有$n$个询问，要么插入一个节点$(k, w)$，要么删除一个键值为$k$的节点，要么询问两个键值分别为$ku, kv$的节点在Treap上的距离。
数据范围$n \leq 200000$。
保证任意时刻树中不存在两个节点的键值相同或者权值相同。

### 19.2.2 算法讨论

首先假如我们有一个关于键值的有序序列，那么如何建树呢？
我们首先找出序列中权值最大的点作为根节点，然后将剩余的两部分作为左子树和右子树递归下去。
显然哪部分是左子树，哪部分是右子树对于答案是没有影响的。
那我们会发现，对于两个序列中的点$x, y$，如果以两个点为两端的区间中不存

在比两个点权值都大的点，那么$x, y$必定有一个是另一个的祖先——权值大的那个是祖先；否则$x, y$一定在某次递归中被划分到了两棵子树中。
同时区间中权值最大的那个点就是两个点的$\mathrm{lca}$，这个结论也是比较显然的。
那么我们要算两个点之间距离，在已经知道$\mathrm{lca}$的情况下，只要能算出每个点的深度就行了。
考虑我们如何找出一个点所有的祖先，祖先的权值应该大于这个点的权值，并且区间中的点的权值均小于祖先的权值。
我们发现只要从这个点在序列中的位置开始，分别找出向前和向后的权值单调递增链的长度就行了。
现在先只考虑向后权值递增链的长度。
不妨使用线段树来维护。
对于线段树上的每个节点，记录从这个节点的左端点开始向右权值单调递增的链的长度以及链上最后一个点的权值。
合并时，我们只需要计算左儿子链上最后一个点在右儿子链上的排名就行了。
而这个排名能利用类似树上二分的方法做到$O(\log n)$。
于是合并时间复杂度为$O(\log n)$，修改一个点的权值只需要将叶子节点到根的路径上的$O(\log n)$个节点都更新一下就行了，时间复杂度$O(\log^2 n)$。

#### 19.2.3 时空复杂度

时间复杂度$O(n \log^2 n)$，空间复杂度$O(n)$。

# 20 January Challange 2014

# 21 December Challange 2013

## 21.1 QTREE6

### 21.1.1 题目大意

给定一棵$n$个点的树，其中每个点都有一个颜色，初始均为黑色。
两个不同的点在同一个连通块中，当且仅当两个点之间的路径上的所有点的颜色相同。
现在有$q$个询问，要么将一个点的颜色取反（黑色变成白色，白色变成黑色），要么询问一个点所在的连通块的点的数目。
数据范围$n, q \leq 10^5$。

### 21.1.2 算法讨论

考虑利用轻重链树链剖分，这样的话，任意一个点到根的路径上最多只会经过$O(\log n)$条轻边。
对于每个节点，我们维护在有根树意义下，以这个节点为根的连通块的大小。
那么答案就应该是，在这个点所在的重链上，从这个点开始深度递增的连续一段的点的数目，再加上以这些点连出去的所有颜色相同的虚儿子为根的连通块的大小。
那么我们只需在线段树上维护最长的白色节点前缀长度以及这些节点的答案之和，最长的黑色节点前缀长度以及这些节点的答案之和，深度最大的的白色节点、黑色节点。
考虑询问，我们只需找出一个点$x$到根的路径上深度最大的与这个点颜色不同的节点$y$，并找到$x$的深度$> y$且最小的祖先$z$，回答以$z$为根的连通块的答案就行了。利用倍增处理，时间复杂度$O(\log n)$。
考虑修改，我们依次处理每条重链，修改会对这条重链深度最小的节点信息产生影响，然后这个深度最小的节点又会沿着轻边对他的父亲重链的某个点产生修改。依此类推，这样只会对$O(\log n)$条重链进行修改，每次修改都是一次线段树上的操作，因此总时间复杂度$O(\log^2 n)$。

### 21.1.3 时空复杂度

时间复杂度$O(n + q \log^2 n)$，空间复杂度$O(n \log n)$。

## 22 November Challange 2013

参见我的第一轮集训队作业题解。
数据和题解能在这里下载：
http://pan.baidu.com/s/1i3rqzq9

# 23 October Challange 2013

# 24 September Challange 2013

## 24.1 TWOROADS

### 24.1.1 题目大意

给定平面上的$n$个点，求两条直线使得每个点到这两条直线距离的最小值的平方和最小。
数据范围$1 \leq n \leq 100$，坐标范围的绝对值$\leq 1000$。

### 24.1.2 算法讨论

考虑任意两条不重合直线，则整个平面会被这两条直线的两条垂直的角平分线分为四个区域，其中两个相对的区域距离其中一条直线比较近，另外两个相对的区域距离另一条直线比较近。
其中一条角平分线将平面上的点集分成了两个部分，考虑两部分分别形成的凸包的公切线必定是两部分各选出一个点的连线，并且这条直线等价于一开始的角平分线，因此我们可以枚举任意两个点的连线作为第一条角平分线，我们将所有点按照这个点在第一条角平分线上的投影长度排序，这样第二条角平分线对于平面起到的划分效果只有$O(n)$种，于是一共只有$O(n^3)$种状态，我们依次枚举取最优解即可。
关于如何求最优解，我们实际上需要考虑这样一个问题，给定平面上若干个点，求一条直线使得这条直线到所有点距离的平方和最小，这个问题可以参考`http://www.zhihu.com/question/37942571`我的回答。这个公式是支持$O(1)$时间内在点集里加点和删点的，于是我们枚举第一条角平分线，再枚举第二条角平分线位置的同时就能顺便维护两个区域的答案了。

### 24.1.3 时空复杂度

时间复杂度$O(n^3)$，空间复杂度$O(n)$。

# 25 August Challange 2013

## 25.1 LYRC

### 25.1.1 题目大意

给定$W$个单词，每个单词长度不超过$|P|$，另有$N$句长度不超过$|S|$的歌词，问每个单词在所有歌词中一共出现的次数。
数据范围：单词和歌词包含大小写英文字母、数字和横杠，$W \leq 500, |P| \leq 1000, N \leq 100, |S| \leq 50000$。

### 25.1.2 算法讨论

直接利用AC自动机进行处理，对单词建出AC自动机，对于每句歌词在AC自动机上走，每走到一个节点都相当于要在这个节点在Fail树上到根节点的路径上都+1，我们可以通过在这个节点上打一个标记来实现，最后再DFS一遍Fail树就能得到答案了。

### 25.1.3 时空复杂度

时间复杂度$O(W|P| + N|S|)$，空间复杂度$O(63W|P|)$。

## 25.2 PRIMEDST

### 25.2.1 题目大意

给定一棵$n$个点的树，每条边的长度都为1，求在上面随机一条路径，这条路径的长度为质数的概率，误差不超过$10^{-6}$就会被认为是正确的。
数据范围$2 \leq n \leq 50000$。

### 25.2.2 算法讨论

我们只需求出每种长度的路径各有多少条，就能知道所有路径的数目以及长度为质数的路径的数目，用两者相除就能得到我们想要的概率。
我们利用点分治统计所有通过当前有根树的根节点的路径。
考虑所有只经过一棵子树的路径，我们只需要从根出发向下DFS一下就能求出这些路径的贡献。
考虑所有跨越两棵子树的路径，对于每棵子树，我们维护一个数组$C$，其中$C_i$表示子树内深度为$i$的节点的数目，我们对两棵子树的$C$数组做一个卷积$D$，则$D_i$表示的就是跨越两棵子树且长度为$i$的路径的数目。
我们可以依次枚举每棵子树，将这棵子树的$C$数组与前面的子树的$C$数组的前缀和的卷积$D$数组统计入答案。
我们分析复杂度：对于一棵子树，$C$数组的长度就是子树内节点的最大深度；对于两个长度分别为$n, m$的$C$数组，卷积花费的时间复杂度为：
$O(max(n, m) \log max(n, m))$。

因此，我们只要将所有子树按照最大深度从小到大的顺序进行排序，并按照上面的过程利用卷积统计答案，对于一棵点数为$n$的树，显然就能在不超过$O(n \log n)$的时间复杂度内完成统计。
再加上树分治，则总时间复杂度为$O(n \log^2 n)$。

#### 25.2.3 时空复杂度

时间复杂度$O(n \log^2 n)$，空间复杂度$O(n)$。

# 26 July Challange 2013

# 27 June Challange 2013

# 28 May Challange 2013

# 29 April Challange 2013

# 30 March Challange 2013

# 31 Februray Challange 2013

## 31.1 QUERY

### 31.1.1 题目大意

给定一棵$n$个点，点上有点权的树，支持在路径上加上一个等差数列，询问路径上的点权之和，还要支持版本回溯操作。
数据范围$n, Q \leq 10^5$，等差数列的首项、公差均$\leq 1000$。

### 31.1.2 算法讨论

考虑如何在序列的一个区间上加上一个等差数列，我们可以利用线段树维护，并设延迟标记$(a, b)$表示在线段树节点对应的区间上加上一个首项为$a$，公差为$b$的等差数列。这个标记能够很容易地下传，不妨令左儿子区间的长度为$l$，那我们将标记$(a, b)$下传，只需要在左儿子上打上$(a, b)$的标记，并在右儿子上打上$(a + lb, b)$的标记。并且利用标记能够$O(1)$更新答案。这样就解决了序列上的问题。
将序列问题拓展到树上，只需要用到轻重链树链剖分，将路径上的操作转化为$O(\log n)$个序列上的操作，这样就能在$O(n \log^2 n)$的时间内解决这个问题。
注意到还需要将数据结构可持久化，我们只需要将用到的线段树可持久化即可，那么空间复杂度$O(n \log^2 n)$。
在将线段树可持久化的时候，由于每走到一个结点，在下传标记的时候都必须新建儿子节点，空间的常数很大。而注意到这里的标记是可加的，我们可以将标记永久化，即不下传标记，进行修改时只在完全覆盖的节点上打上标记，同时更新这个节点到根节点上的路径上的信息；查询时在递归过程中维护一个根节点到这个节点的路径上的标记的和，当区间完全覆盖节点时，返回这个节点的答案再加上维护的标记对于这个节点产生的贡献。这样就不用下传标记了，减小了空间的常数。

### 31.1.3 时空复杂度

时间复杂度$O(Q \log^2 n)$，空间复杂度$O(Q \log^2 n)$。

# 32 January Challange 2013

# 33 December Challange 2012

## 33.1 DIFTRIP

### 33.1.1 题目大意

给定一棵$n$个点的有根树，定义两条深度递增的路径是相似的当且仅当两条路径长度相同并且对应位置的点的度数相同。
问不相似的路径一共有多少种。
数据范围$n \leq 10^5$。

### 33.1.2 算法讨论

相当于是一颗Trie树，其中的每个点都写着一个与这个点在树上的度数相等的的字符，那么只需要求出Trie树上有多少种不同子串就行了。
利用后缀自动机在Trie树上的拓展就能解决这个问题了。
注意这里的字符集大小是$O(n)$级别的，所以需要用$map$来存指针。

### 33.1.3 时空复杂度

时间复杂度$O(n \log n)$，空间复杂度$O(n)$。

# 34 November Challange 2012

## 34.1 COUNTARI

### 34.1.1 题目大意

给定一个长度为$n$的正整数序列$a$，问有多少三元组$(i,j,k)$满足$1 \leq i < j < k \leq n$且$a_j - a_i = a_k - a_j$。
数据范围$n \leq 10^5$，$1 \leq a_i \leq 30000$。

### 34.1.2 算法讨论

不妨将序列分成$m$块，每块$\frac{n}{m}$个元素。
考虑三元组分布的情况：
（1）分布在同一块中。
（2）前两个分布在同一块中，最后一个分布在另一块中。
（3）后两个分布在同一块中，第一个分布在另一块中。
这三种情况，我们都可以通过暴力枚举$j$和同一块中的$i$（或$k$），并利用$2a_j = a_i + a_k$统计此时合法的$k$（或$i$）的数目，只需维护一个带时间戳的计数数组就能完成统计。
时间复杂度$O(m \times (\frac{n}{m})^2 + n)$。
（4）三个元素分布在不同块中。
只需枚举$j$所在的块，对于两侧的两个序列做卷积，就能对于这个块中的每个$j$算出有多少对合法的$(i,k)$。
不妨令$w = max\{a_i\}$。
时间复杂度$O(m(n + w \log w))$。
实际测试中令$m = 30$就能通过全部测试数据了。

### 34.1.3 时空复杂度

时间复杂度$O(m \times (\frac{n}{m})^2 + n + m(n + w \log w))$，空间复杂度$O(n + w)$。

# 35 October Challange 2012

# 36 September Challange 2012

## 36.1 PARADE

### 36.1.1 题目大意

有一个$n$个点，$m$条边的有向图，边上是带权的，现在要找出若干条路径，首先花费所有路径上的边的权值之和（若某条边出现在$k$条路径中则计算$k$次），随后如果有一条路径的起点和终点不同额外花费$C$的费用，如果有一个点没有出现在任意一条路径中额外花费$C$的费用。现有$Q$组询问，每次给定一个$C$的取值，问这种情况下的最小花费。
数据范围$n \leq 250, m \leq 30000$。

### 36.1.2 算法讨论

我们用Floyd算法求出全局最短路，然后重新建图，将每两个点之间连一条边，边权为两个点之间的最短路。然后，我们在新图上求解这个问题，那么可以将问题转化为选出一些点不相交的路径，若路径上一共有$k$条边，则付出的费用为所有边的权值之和再加上$(n-k) \times C$。
那么这个问题类似于有向图的最小路径覆盖问题，我们将所有点拆成入点和出点，然后利用SPFA求最小费用流，那么每次增广选择的边的权值都是单调递增的，我们将这些权值记下来，那么对于一个给定的$C$，我们可以二分找到第一个$> C$的权值，只选择这个权值前面的权值对应的边。

### 36.1.3 时空复杂度

时间复杂度$O(n^3 + COSTFLOW(n, n^2) + Q \log n)$，空间复杂度$O(n^2)$。

# 37 August Challange 2012

# 38 July Challange 2012

## 38.1 DGCD

### 38.1.1 题目大意

给定一棵$n$个点的树，一开始时树上编号为$i$的点的权值为$v_i$，支持两种操作：要么将一条路径上的点的权值均加上一个数$d$，要么询问一条路径上的点的权值的最大公约数。
数据范围$n, q \leq 50000, v_i, d \leq 10000$。

### 38.1.2 算法讨论

首先注意到一个事实：

$$gcd(a_1, a_2, ..., a_n) = gcd(a_1, a_2 - a_1, ..., a_n - a_{n-1})$$

这个结论我们可以使用数学归纳法证明，这里略过。
那么考虑如何在序列上解决这个问题，考虑区间加上一个数对于区间的影响：

$$gcd(a_l + d, a_{l+1} + d, ...a_r + d) = gcd(a_l + d, a_{l+1} - a_l, ..., a_r - a_{r-1})$$

因此我们利用线段树在每个区间长度为$n$节点上维护区间左端点的值，以及$n-1$个差分数值的最大公约数，这个区间的最大公约数就是二者的最大公约数，对区间进行修改只需要改左端点的值就行了，利用延迟标记，我们就能在$O(\log n)$的时间里解决这个问题。
为了将序列拓展到树上，我们只需要利用树链剖分，将路径拆分为最多$O(\log n)$条链求解，同时最大公约数是容易合并的，那么只需依次在每条链上用序列的求解方法求得答案然后合并就行了。
单组询问时间复杂度$O(\log^2 n)$。

### 38.1.3 时空复杂度

时间复杂度$O(n + q \log^2 n)$，空间复杂度$O(n)$。

# 39 June Challange 2012

## 39.1 MATCH

### 39.1.1 题目大意

给定一个二分图，两侧分别有$n, m$个点，现在给定$P_{i,j}$表示左侧的第$i$个点到右侧的第$j$个点之间有边的概率，求这个二分图的期望最大匹配。
数据范围$n \leq 5, m \leq 100, 0 \leq P_{i,j} \leq 1$。

### 39.1.2 算法讨论

注意到$n \leq 5$，于是我们只对左侧的点进行考虑。
定义匹配状态表示一个长度为$n$的01串，第$i$位为1表示左侧的第$i$个点目前是匹配点，否则不是匹配点。显然，不同的匹配状态有$2^n$种。
定义匹配状态的集合表示一个长度为$2^n$的01串，第$i$位表示第$i$种匹配状态目前是不是合法的。
考虑$f_{i,s}$表示只考虑右侧前$i$个点，左侧匹配状态的集合为$s$时概率。
我们枚举所有的$s$，再$O(2^n)$枚举与右侧第$i+1$个点的连边情况，我们计算$f_{i,s}$的贡献：
对于当前的“匹配状态的集合”$s$，我们考虑其中所有的合法匹配状态$state$，枚举与第$i+1$个点相连的左侧的点$x$，若$state[x] = 0$，则我们令一个新的状态为$newstate = state, newstate[x] = 1$，并将$newstate$扩充入现在的匹配状态的集合。这样我们得到一个新的匹配集合$s'$，令枚举的这种连边的概率为$p$，则我们做转移$f_{i+1,s'} += f_{i,s} \times p$。
这是因为我们对于当前的匹配状态，只能至多再选一个左侧的点加进去与$i+1$匹配。
然而我们发现$s$的状态量是非常大的——能达到$O(2^{2^n})$！
但是实际上我们分析一下，会发现有用的状态很少。
现在假设从空集开始，进行了若干次状态的扩充，每次都是尝试将一系列的点加入匹配：
第一次：$\{2\}$；第二次：$\{1,3\}$，第三次：$\{2,4\}$。
不难发现第三次中的2对于匹配集合是没有任何影响的：也就是说**每个左侧的点只会在第一次尝试加入匹配集合时产生影响。**
于是我们对于一个匹配状态集合，我们只需要考虑尝试加入过匹配集合的左侧点的并集，以及并集的顺序。
由于$n$很小，总状态数不是很多，只有大概4000个。
对于每个匹配状态集合，容易知道此时的最大匹配，那么求出概率就能计算答案了。
于是总时间复杂度为$O(4000m2^n)$。

### 39.1.3 时空复杂度

时间复杂度$O(4000m2^n)$，空间复杂度$O(4000m)$。

## 39.2 CLOSEST

### 39.2.1 题目大意

给定三维空间中的$n$个点，另有$Q$组询问，每次给定一个三维空间中的点，求到这个点的欧几里得距离最小的点的下标。
数据范围$n, Q \leq 50000$，坐标数值范围绝对值$\leq 10^9$，得分与输出正确的数量有关。

### 39.2.2 算法讨论

考虑K-Dimension Tree来进行修改和查询，具体细节在这里不再赘述。
直接进行每次查询期望是$O(\sqrt{n})$的，但是点集可能并不是随机的，可能会超时，于是我们进行卡时，利用KDTree回答若干个询问直到时间所剩无几，然后对于剩下的询问输出随机数。

### 39.2.3 时空复杂度

时间复杂度$O(Q\sqrt{n})$，空间复杂度$O(n)$。

# 40 May Challange 2012

## 40.1 LEBOXES

### 40.1.1 题目大意

有$n$个箱子，第$i$个箱子打开之后有$\frac{P_i}{100}$的概率得到$V_i$单位的钱，有$1-\frac{P_i}{100}$的概率得到1个钻石。还有$m$个物品，购买第$i$个物品需要花费$C_i$个单位的钱以及$D_i$个钻石。现在打开所有的箱子，求最多能够买到物品数目的期望值。
数据范围：$n,m \leq 30, 1 \leq V_i, C_i \leq 10^7, 0 \leq D_i \leq 30, 0 \leq P_i \leq 100$。

### 40.1.2 算法讨论

我们不妨先利用DP预处理出$f_{i,j}$表示想买$i$个物品的时候，如果得到了$j$个钻石，那么最少还需要多少单位的钱。
考虑到$n$非常小，利用Meet in the middle，将所有箱子划分成两个尽可能大小相同的集合，依次在集合内部$2^n$枚举所有可能的情况，并存下在每个集合内得到固定数量的钻石的时候得到的钱数从小到大的排序。
我们枚举其中一个集合的所有可能情况，枚举另一个集合得到了多少个钻石，再枚举最终能得到多少物品，对于一个物品数，在另一个集合固定钻石数的所有情况中，钱数是一段连续区间，我们可以通过二分得到这段区间，在预处理时维护概率的前缀和就能用这段区间的概率和更新答案。

### 40.1.3 时空复杂度

时间复杂度$O(nm^2 + \lfloor \frac{n}{2} \rfloor 2^{\lfloor \frac{n}{2} \rfloor} + 2^{\lfloor \frac{n}{2} \rfloor} nm \lfloor \frac{n}{2} \rfloor)$，空间复杂度$O(nm + 2^{\lfloor \frac{n}{2} \rfloor})$。

## 40.2 TICKETS

### 40.2.1 题目大意

有$n$道菜和$m$个顾客，每个顾客都喜欢$n$道菜中不同的两种菜，现在要将这些菜供应给顾客，每道菜最多只能供应给一个顾客。现在要求最大化$k$，使得在$m$个顾客中任选$k$个顾客都能够存在一种供应方案使得每个顾客都至少被供应到一道喜欢的菜。
数据范围：数据组数$T \leq 15$，$2 \leq n \leq 200, 0 \leq m \leq 500$。

### 40.2.2 算法讨论

我们考虑构造一个无向图，对于每个顾客都看成是在无向图中两道菜对应的点之间连的一条无向边。
这样的话，对于一种顾客的选择如果存在满足要求的方案，当且仅当这些边以及这些边相关的点形成的子图中点数不小于边数。
那么事实上只需求出边数最小的边数大于点数的子图就行了。

我们考虑边数最小的这种子图使得边数>点数，若边数≥点数+2，那我们考虑删去一条边，如果这删去了两个点，证明这条边与剩下的图是不连通的，且剩下的图满足边数>点数且边数更小；如果这删去了一个点，边数和点数同时减少了1，剩下的图依然满足边数>点数且边数更小；如果这没有删去任何点，边数减少1，剩下的图依然满足边数≥点数+1即边数>点数，且边数更少。
所以我们证明了边数最小的使得边数>点数的子图必定满足边数=点数+1。
考虑这种子图有什么性质：
通过刚才的讨论我们已经证明这个子图中所有点的度数都至少为2，不然就可以删掉一条边了。
令点数为$|V|$，由于总度数为$2|V|+2$，那么就有两种情况：
情况1：两个点度数为3，剩下的所有点度数为2。
这种情况要么是两个点之间有三条点不相交的路径（A），要么是两个点不相交的环上连了一条边（B）。
情况2：一个点度数为4，剩下的所有点度数为2。
这种情况只能是两个环只同时存在一个点（C）。
对于A，我们枚举所有起点和终点求出最短的三条路径就行了，时间复杂度$O(n^2m)$。
对于B，C，我们可以枚举根做一棵有根树，再枚举所有非树边，得到这条非树边构成的环和路径的总长，然后只需要维护总长的最小值和次小值用总和来更新答案即可。最小值和次小值的路径相交是没有关系的，因为如果相交那么说明在A中一定能求出更小的解。实现中用LCA来求路径的长度，我们暴力$O(n)$来求，总时间$O(n^2m)$。

### 40.2.3 时空复杂度

时间复杂度$O(Tn^2m)$，空间复杂度$O(m)$。

# 41 April Challange 2012

## 41.1 TSUBSTR

### 41.1.1 题目大意

给定一棵$n$个点的有根树，每个节点上都有一个字母，我们说一个字符串出现在这棵树上当且仅当这个字符串能够由树上一条深度递增的路径中的节点上的字母顺次连接起来形成。首先求出现在这棵树上的字符串集合的大小，然后有$Q$组询问，每次给定一个字母之间的字典序大小，求字符串集合中的字典序第$k$小的字符串。
数据范围：$n \leq 250000, q \leq 50000, 1 \leq k < 2^{63}$，总输出大小不超过$800KB$。

### 41.1.2 算法讨论

我们不妨考虑后缀自动机在字典树上的拓展，对于每个节点我们都从他的树上的父亲结束插入的节点上开始插入，这样用一次BFS就能完成构造。
后缀自动机可以被看成一张拓扑图，从根节点出发的每条路径都对应着一个子串，那么我们只需要用一次拓扑序动态规划就能求出后缀自动机上从每个节点出发能走出的路径条数了。
对于每组询问，我们可以从根节点出发，类似树上二分的过程，每次按照字母的字典序大小开始走，由于总输出的大小是有限的，所以复杂度是有保证的。

### 41.1.3 时空复杂度

时间复杂度$O(n + q + 800000)$，空间复杂度$O(n|\alpha|)$。

## 41.2 CONNECT

### 41.2.1 题目大意

给定一个$n \times m$的矩阵，其中每个位置都有一个数，数的范围是$[-1, n \times m]$，每个位置还有一个选择这个位置需要付出的代价，每个代价都为正数。
现在要求选择矩阵中的若干个位置组成一个四连通的连通块，使得其中至少有$k$种不同的正数，同时最小化付出的总代价。
数据范围$n, m \leq 15, k \leq 7$。

### 41.2.2 算法讨论

如果数值的范围为$[-1, k]$之间，那么就很好做了，直接利用斯坦纳树模型就可以了。
我们考虑随机将$[1, n \times m]$区间中的数映射到$[1, k]$中，然后这样做一次，能得到正确答案的概率为$\frac{k!}{k^k}$，那么我们随机做$T$次，当$T = 300$的时候就能够通

过Tsinsen上的数据了。

#### 41.2.3 时空复杂度

时间复杂度$O(T(3^k+2^k\times SPFA(n\times m,n\times m)))$，空间复杂度$O(nm2^k)$。

# 42 March Challange 2012

# 43 Februray Challange 2012

# 44 January Challange 2012

## 44.1 CARDSHUF

### 44.1.1 题目大意

有一个长度为$n$的序列，现在要进行$m$个操作，要么将序列中的一段连续子序列换一个位置，要么将这段连续子序列翻转过来再换一个位置。最后输出这个序列。
数据范围$n, m \leq 10^5$。

### 44.1.2 算法讨论

直接利用Splay树维护这些操作即可。

### 44.1.3 时空复杂度

时间复杂度$O(n + m \log n)$，空间复杂度$O(n)$。

# 45 December Challange 2011

# 46 November Challange 2011

# 47 October Challange 2011

## 47.1 BAKE

### 47.1.1 题目大意

维护一些出售产品的信息，要么插入一条出售信息，要么查询所有在给定范围内的出售信息的总销售额，信息可能有省略，具体内容请参见原题。

### 47.1.2 算法讨论

用树状数组维护年龄这一维，剩下的维度没有区间查询，则可以被压成一个6元组进行状态表示，那我们每次插入时，对于给定的6元组，我们修改所有这个6元组被包含在的信息可能不确定的6元组，这样的6元组最多有$1 \times 2 \times 3 = 6$种，令最大年龄为$n$，则每次修改复杂度为$O(\log n)$；对于每次查询直接在对应的树状数组中查询就行了，复杂度也为$O(\log n)$。

### 47.1.3 时空复杂度

时间复杂度$O(S \log n)$，空间复杂度$O(n)$。

## 48 September Challange 2011

# 49 August Challange 2011

# 50 July Challange 2011

# 51 June Challange 2011

## 51.1 MINESREV

### 51.1.1 题目大意

给定一张扫雷地图，其中某些位置是空格，某些位置是地雷。
如果对于某个空格，以这个空格为中心的$3 \times 3$区域内存在地雷，那么这个空格将被替换为一个数字方格，表示这个区域内存在地雷的个数。
正常的扫雷规则是：
点开一个空格，则能够同时点开这个空格所在的连通块以及这个连通块周围的一圈数字。
点开一个数字，则只能点开这个数字本身。
点开一个地雷，则只能点开这个地雷本身。
现在要将扫雷规则反过来：
令$s(x)$表示点开格子$x$能够同时点开的格子集合。
对于两个格子$p, q$，如果存在一个格子$x$使得$p \in s(x)$且$q \in s(x)$，则在新规则下点开$p$的同时也能点开$q$。
现在给定扫雷地图，求在新规则下最少多少次操作能够点开所有的格子。
数据范围：数据组数$T \leq 50$，地图长宽$n, m \leq 50$。

### 51.1.2 算法讨论

首先所有的地雷都必须点一次。
点开一个空格的话，能够点开这个空格所在的连通区域以及周围的一圈数字。
点开一个数字的话，能够点开所有与这个数字相邻的空格区域以及他们周围的一圈数字。
每个空格连通区域周围都一定有数字，那么点开空格肯定没有点开数字划算。
将每个空格连通区域以及周围的一圈数字看做一个点，每个同时与两个空格连通区域相邻的数字看做一条边，这样形成了一个无向图。
容易证明，每个数字最多只能与两个空格连通区域相邻。
实际上，问题抽象成了求一个无向图的最小边覆盖。
而这个答案等于点数减去最大匹配，我们利用一般图的最大匹配求解就行了。
还有一些孤立的数字，按照地雷处理。 无向图的点数是$O(nm)$级别的，而这个图显然是一个平面图，于是边数也是$O(nm)$级别的。
一种一般图最大匹配的实现能够做到$O(n^2m^2)$。
总时间复杂度$O(Tn^2m^2)$。

### 51.1.3 时空复杂度

时间复杂度$O(Tn^2m^2)$，空间复杂度$O(nm)$。