Solution 12

**1、素数（CF 396B）**

我们把 1 / { u(i)\*v(i) }拆开->  (1/(u(i)-v(i)) \* ( 1/v(i) - 1/u(i) )

若n +1  是素数，则显然(1/(u(i)-v(i)) \* ( 1/v(i) - 1/u(i) ) 这样完全相同的式子有 u(i)-v(i) 个

那么就可以把前面系数约掉，那么剩下的式子就是 1/2 - 1/(n+1)

若不是，则我们找到第一个<=n的素数，即v(n)

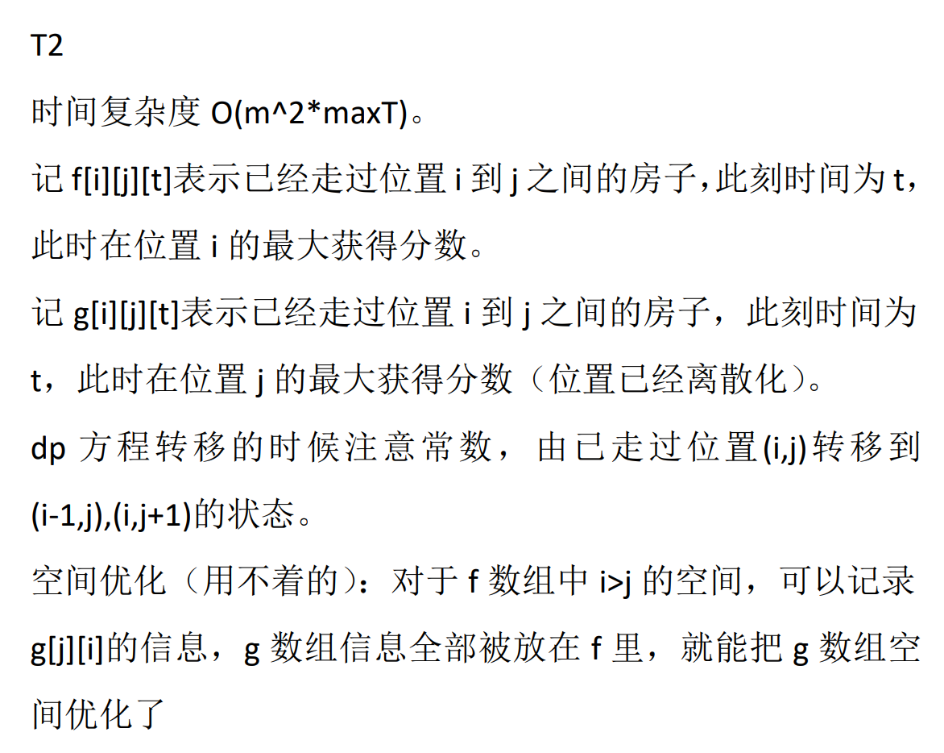
和第一个>n的素数,即 u(n)

然后前面的 2-v(n)求和，即 1/2 - 1/v(n)

后面共有 n - v(n)项，即 (n-v(n)) \* (1/ (u(n)\*(v(n) )

同分即可。

**2、精灵**



**3、秘密通道**

我们发现，一个点可以向上下左右最近的一面墙开一枪，再向最近的墙开一枪，走到最近的墙边再走一步就能传送过去  
  
走到最近的墙旁边并再走一步就等于最近的墙与当前点的距离

首先，可以预处理出每个格子到上下左右第一堵墙在哪里

然后考虑点连边跑最短路

①向它的四周连一条权值为1的边

②向上下左右第一堵墙连一条与它距离的墙为权值的边

最后跑一边dijkstra最短路就好了

（所以有人写spfa是可以被卡掉的吧）

然后是BFS

它好像是错的

我没写过我不知道它为什么错了

大概可能是因为你写记忆化的话不一定是最优解

因为传送门的边权有很大可能不为1

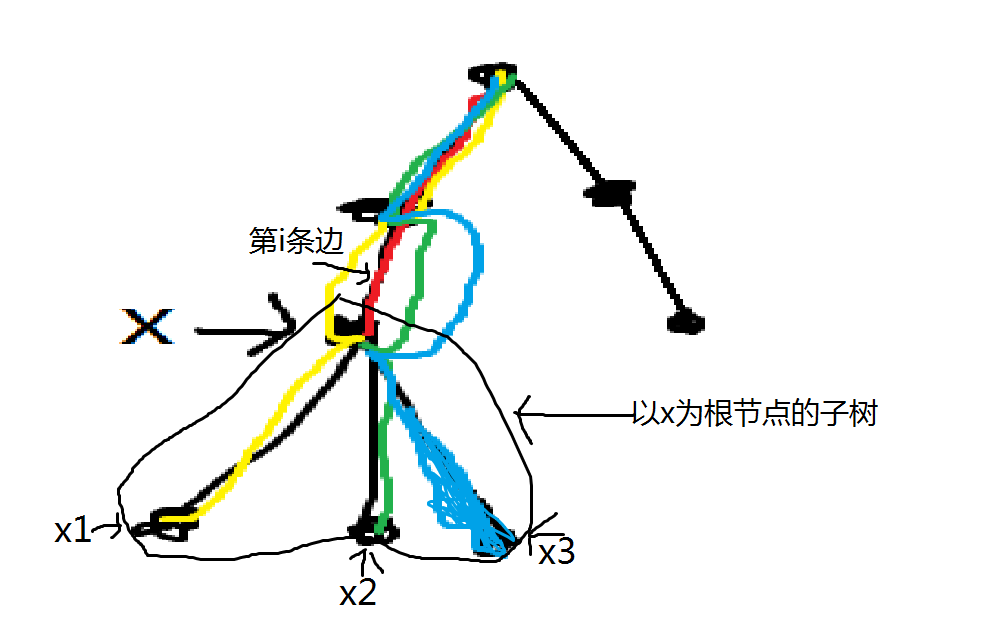
**4、小x游世界树题解**

题目大意：

给出一棵树，求从起点到所有节点的最小距离和。

40分做法：

n只有1000，所以可以用邻接表或邻接矩阵来存边，暴力枚举根节点，每次建一棵树，找出最优的，输出就行。至于答案怎么求请看下面。

我们设f[i]是以i为根的子树的消耗能量值的总和(包括i)，zs[i]是以i为根的子树的节点个数(包括i)。当我们在建树时其实就可以求出当1为根节点时的答案，因为每个节点都要走一次，我们不如求每条边走了多少次，而一条边走的次数则是由这条边链接的两个点中深度大的节点的那颗子树的节点数来决定的，设这个节点为x，因为根节点要到达这棵子树上的每一个点，那么这条边就要走zs[x]次，原因见下图。

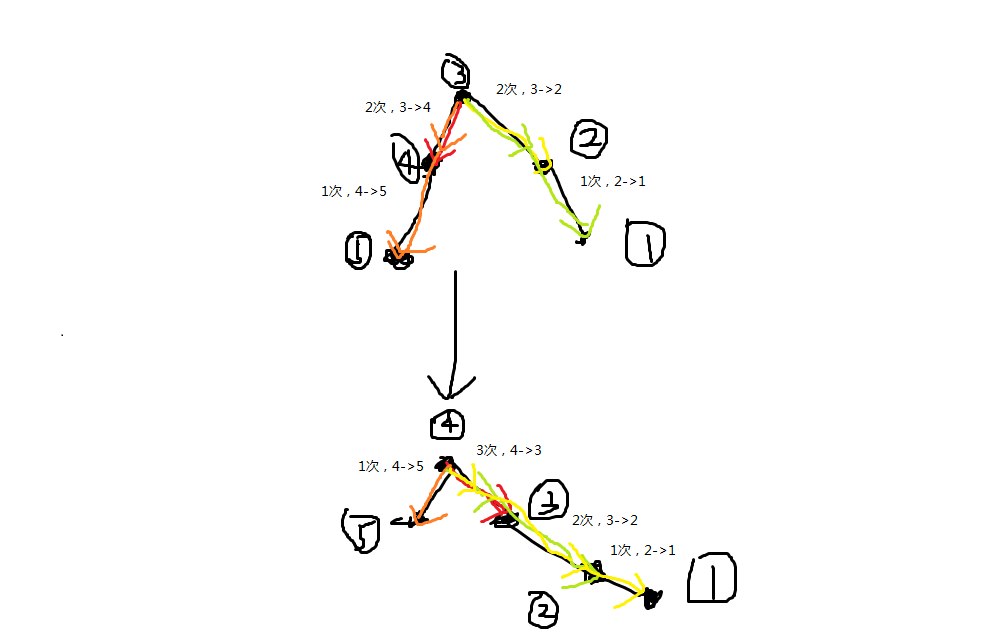
很容易可以看出，根节点要到达以x为根节点的子树的每一个节点，也就是x,x1,x2和x3，要经过这条四次（已用不同颜色标出）。根据这个我们可以得知，f[i]由i的所有儿子的f值加上i与i的父亲节点fa[i]之间的那条边走的次数得出。也就是f[i]=sum（f[son[i]]）+zs[i]\*(s[j]-a[fa[i]])。son[i]表示i的所有子节点，sum（f[son[i]]）就是i的子节点的f值总和，s[j]表示i与fa[i]之间的那条边的权值，a[fa[i]]表示i的父亲节点的加速器性能，因为是从上往下走，所以减的是i的父亲的加速器。当然，不同的根节点会导致某个点的父亲也有所不同，所以需要重建一棵树。

时间复杂度O(n^2+n)，空间复杂度O(n^2+n)。

60分做法：

因为邻接矩阵存不下边，而邻接表链表又有人不会（比如我），那么就可以用到前向星了，链式前向星只需要n的空间(对于这道题)，我们用h[i]表示以i为起点的第一条边在数组中的位置，l[i]表示第i条边的下一条边(也是以第i条边为起点的边)，g[i]表示第i条边的终点,s[i]表示第i条边的权值。因为是双向边，所以一条边要存两次。每读入一条边，就将它拆成两条单向边，先将边的总数加一，然后将这条边的下一条边(l[ss],ss表示边的总数)更新为其中一个点的第一条边，然后将这个点的第一条边位置更新，g[ss]为另一个点，s[i]就是权值，至于以另一个点为起点的操作跟这个差不多。时间复杂度O(n^2)，空间复杂度O(n)。

100分做法：

其实并不是每个点都要建一次树，我们可以在一个点为整棵树的根节点时，以它的答案来更新它的儿子节点的答案。

可以看出(画的丑别介意，看得懂就行)，如果将根节点的儿子换为根，会变动的只有根节点与被换的那个儿子之间的边，原来的次数=zs[4]=2，现在的次数=zs[3]（新的）=3，而且由3到4变成了由4到3。虽然改变子树的子节点数会方便一些，但可能会出一些迷之bug(相信su某感触很深)，所以最好用n-zs[4]来代表zs[3]，则新的f[4]则等于f[3]+（n-zs[4]）\*(s[i]-a[4])-zs[4]\*(s[i]-a[3])，将它套进去就行。