Hello World ! --zoj

KM 二分图带权匹配

题意：两个字符串的similarity定义为：

Similarity(S, T) = sum(Si == Ti) / L

L = Length(S) = Length(T), i = 1, 2,... L, where sum(Si == Ti) indicates the total number of equal labels in corresponding positions.

就是算两个串的最大similarity.

思路：按字母定义点。边权用上式算，其实只要扫一遍即可，在build里面算边权.

区间DP 好题

题意：跟经典的关路灯一样

思路：因为这题之前做了hdu\_4283，“悟”除了区间DP的状态f[i,j]要保证满足局部独立性，然后做这题的时候一眼定义了f[i,j]表示关掉区间[i,j]的

路灯花费最少的电费，而且严格控制是[i,j]这个区间，跟外面的灯无关。 然后接下来就不知道怎么转移了...因为历时时间无法计算也无法记录。

然后还是看了题解。发现又是自己的f[i,j]状态的含义定义不合适...=\_=

正解的f[i,j]状态定义为，关掉[i,j]这个区间的灯时，全局已经累积了多少电费。

这个状态的定义对“历时”比较好解决，如果当前人类关掉i灯后走向j灯，则全局电费增量为dis(i,j) \* sum\_of\_rest\_lights\_power，就是距离

乘上剩下的灯的功率和。

状态方程就好解决了，略.

摘抄：先说下本题的模型，送餐的顺序是一个从x位置开始的从1...n的一个全排列，我们要做的是找一个全排列它的总坑爹值最小

int work(int s)

{

for(int i = 1; i <= n; i++)

for(int j = 1; j <= n; j++) f[0][i][j] = f[1][i][j] = INF;

f[0][s][s] = f[1][s][s] = 0;

for(int i = s; i > 0; i--)

for(int j = s; j <= n; j++) if(i != j) { //TODO

int tmp = (Sum(1, i-1) + Sum(j+1, n));

f[0][i][j] = min(f[0][i][j], f[0][i+1][j] + (a[i+1].x - a[i].x) \* (tmp + a[i].b));

f[0][i][j] = min(f[0][i][j], f[1][i+1][j] + (a[j].x - a[i].x) \* (tmp + a[i].b));

f[1][i][j] = min(f[1][i][j], f[0][i][j-1] + (a[j].x - a[i].x) \* (tmp + a[j].b));

f[1][i][j] = min(f[1][i][j], f[1][i][j-1] + (a[j].x - a[j-1].x) \* (tmp + a[j].b));

}

return min(f[0][1][n], f[1][1][n]);

}

int main()

{

int V, X;

while(scanf("%d%d%d", &n, &V, &X) != EOF) { //V分钟跑1m，那就以V为单位

for(int i = 1; i <= n; i++) {

scanf("%d%d", &a[i].x, &a[i].b);

//a[i].x \*= V;

}

//X \*= V;

a[++n] = Person(X, 0);

sort(a+1, a+n+1, cmp);

for(int i = 1; i <= n; i++) sum[i] = sum[i-1] + a[i].b;

for(int i = 1; i <= n; i++) if(a[i].x == X && a[i].b == 0) {

printf("%d\n", V \* work(i)); //留到后面乘，之前在前面乘因为溢出WA了两次

break;

}

}

return 0;

}

枚举然后贪心判定

题意：n个人从1到n编号，每个人可以投一票但是不能投给自己。1号是主角，想做班长，但票数必须严格大于其他人，但是主角可以任何人，收买

每个人都有一个代价。求主角做到班长的最小花费。注意主角一开始没有投票，它可以选择投给任何人除开自己

思路：主角的票一定是投给当前票数最少的人，这样对他的影响最小。

另外主角最后当选班长，他拥有的票数为v的话，其他人的票数必须全部 <= v-1，如果全部人的票数<=v-1的情况下，主角的票数已经>v，那么这

个枚举是失败的（因为我们是假定主角的票数确切为v），然后将其他票拉过来给主角的时候，就贪心地选。

没心情写题解了...这道水题硬是卡了我好久...最后还是对拍找出来的bug

我一开始是直接让1号投给票数最少的那位同学，这是没错的，但问题是，不能在贪心判定之前就投！甩个数据，懒得说:

4

4 3 2

46 167 927

直接不管1号的票就ok了

还有一点，收买的同学必须是原本没有投给1号的的同学，这点要判断一下，网上有个别人的程序是错的，但数据水了吧，让过了，but it's wrong