**1、医院设置**

本题的求解任务十分明了：求一个最小路径之和。

根据题意，对n个结点，共有n个路径之和：用记号Si表示通向结点i的路径之和，则，其中Wj为结点j的居民数，g(i,j)为结点j到结点i的最短路径长度。下面表中反映的是样例的各项数据：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| j  g(i,j)  i | 1 | 2 | 3 | 4 | 5 | Si |
| 1 | 0 | 1 | 1 | 2 | 2 | 0×13+1×4+1×12+2×20+2×40=136 |
| 2 | 1 | 0 | 2 | 3 | 3 | 1×13+0×4+2×12+3×20+3×40=217 |
| 3 | 1 | 2 | 0 | 1 | 1 | 1×13+2×4+0×12+1×20+1×40=81 |
| 4 | 2 | 3 | 1 | 0 | 2 | 2×13+3×4+1×12+0×20+2×40=130 |
| 5 | 2 | 3 | 1 | 2 | 0 | 2×13+3×4+1×12+2×20+0×40=90 |

从表中可知S3=81最小，医院应建在3号居民点，使得所有居民走的路径之和为最小。

由此可知，本题的关键是求g[i,j]，即图中任意两点间的最短路径长度。

求任意两点间的最短路径采用下面的弗洛伊德（Floyd）算法。

（1）数据结构：

w:array[1..100]of longing; 描述个居民点人口数

g:array[1..100, 1..100]of longint 初值为图的邻接矩阵，最终为最短路径长度

（2）数据的读入：

本题数据结构的原形为二叉树，数据提供为孩子标识法，分支长度为1，建立带权图的邻接矩阵，分下面两步：

①g[i,j]←Max {Max为一较大数，表示结点i与j之间无直接相连边}

②读入n个结点信息：

for i:=1 to n do

begin

g[i,j]:=0;

readln(w[i],l,r);

if l>0 then begin

g[i,l]:=l; g[l,i]:=l

end;

if r>0 then begin

g[i,r]:=l; g[r,i]:=l

end;

（3）弗洛伊德算法求任意两点间的最短路径长度

for k:=1 to n do

for i:=1 to n do

if i<>k then for j:=1 to n do

if (i<>j)and(k<>j)and(g[i,k]+g[k,j]<g[i,j]) then g[i,j]:=g[i,k]+g[k,j];

（4）求最小的路程和min

min:=max longint;

for i:=1 to n do

begin

sum:=0;

for j:=1 to n do sum:=sum+w[i]\*g[i,j];

if sum<min then min:=sum;

end;

（5）输出

writeln(min);

**仔细思考一个问题：现在n<=100，当数据范围变成了n<=100000，这道题又应该怎么做呢？提示：这是一棵树！！**

**2、工程规划**

本题是一类称为约束满足问题的典型问题，问题描述成n个子任务的起始时间Ti及它们之间在取值上的约束，求一种满足所有约束的取值方法。

将工程的n个子任务1，2，…，n作为一有向图G的n个顶点，顶点Vi（i＝1，…，n）的关键值为子任务i的起始时间Ti，我们并不需要关心顶点之间的弧及其弧长，而是要确定顶点的关键值Ti的取值，满足所有的约束条件。本题的约束条件由m个不等式Ti-Tj≤b给出，这样的有向图称为约束图。

为了确定每一个Ti的值，先假设某一个子任务的起始时间为零，如设Tj＝0，则其余子任务的起始时间Ti相对于T1可设置其起始时间为一区间[-maxt，maxt]。

下面分析不等式Ti-Tj≤b。此不等式可变形为如下两种形式：

（1）Ti≤Tj+b意味Ti的最大值为Tj+b；

（2）Tj≥Ti-b意味Tj的最大值为Ti-b；

因此，根据题中给出的m个不等式，逐步调整各个Ti的最小值和最大值。

设high[i]为Ti当前的最大值，low[i]为Ti当前的最小值。

high[j]为Tj当前的最大值，low[j]为Tj当前的最小值。

若high[i]-high[j]>b，则high[i]=high[j]+b(根据Ti≤Tj+b)，

若low[i]-low[j]<b，则low[j]＝low[i]-b(根据Ti≥Ti-b)。

以上的调整终止视下列两种情况而定：

（1）对所有的不等式Ti-Tj≤b，不再有high[i]或low[j]的调整；

（2）若存在high[i]<low[i]或high[j]<low[j]则表示不存在T1，T2，…，Tn能满足所有m个不等式Ti-Tj≤b，即问题无解。

根据以上思路，先建立约束图，每个结点代表一个起始时间值，并记录其可能的取值范围：

数组high，low:array[1..maxn]of longint；{n个子任务起始时间的取值范围}

high[1]=0; low[1]=0; {设置n个起始时间的初值，其中Ti=0}

for i:=2 to n do begin

high[i]:=maxt; {Ti的上界}

low[i]:=-maxt; {Tj的下界}

end;

约束条件（m个不等式）用记录数组表示：

type {不等式结构}

Tinequ=record

i,j,b:longint;

end;

var

arrinequ:array[1..maxm]of Tinequ; {存放m个不等式}

利用约束条件，逐一调整每一个起始时间的取值范围，直到无法调整为止。

主要算法如下：

|  |
| --- |
| flag:=true; {调整状态标记}  noans:=false; {解的有无标记}  while (flag) do {进行约束传递，根据不等式调整各个起始时间值}  begin  flag:=false;  for k:=1 to m do  with arrinequ[k] do  begin  if (high[i]-high[j]>b) then begin high[i]:=high[j]+b; flag:=true; end; {调整Ti的上界}  if (low[i]-low[j]>b) then begin low[j]:=low[i]-b; flag:=true; end; {调整Tj的下界}  if (low[i]>high[i]) or (low[j]>high[j])  then begin {无法满足当前不等式，则调整终止}  noans:=true; {问题无解noans=true}  flag:=false;  break;  end;  end;  end; |

下面以样例说明：

**【样例1】**

8个不等式如下

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| i | 1 | 1 | 2 | 3 | 4 | 4 | 5 | 5 |
| j | 2 | 5 | 5 | 1 | 1 | 3 | 3 | 4 |
| b | 0 | -1 | 1 | 5 | 4 | -1 | -3 | -3 |

顶点的关键值Ti的调整记录：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 初值 | 第1轮调整 | 第2轮调整 | 第3轮调整 |
| high[1] | 0 | 0 | 0 | 0 |
| low[1] | 0 | 0 | 0 | 0 |
| high[2] | 100000 | 100000 | 2 | 2 |
| low[2] | -10000 | 2 | 2 | 2 |
| high[3] | 100000 | 5 | 5 | 5 |
| low[3] | -10000 | 4 | 5 | 5 |
| high[4] | 100000 | 4 | 4 | 4 |
| low[4] | -10000 | 4 | 4 | 4 |
| high[5] | 100000 | 1 | 1 | 1 |
| low[5] | -10000 | 1 | 1 | 1 |
| 调整状态 | | 有变化 | 有变化 | 无变化 |

**【样例2】**

5个不等式如下

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号 | 1 | 2 | 3 | 4 | 5 |
| i | 1 | 1 | 2 | 5 | 4 |
| j | 2 | 5 | 5 | 1 | 1 |
| b | -3 | -1 | -1 | -5 | 4 |

顶点关键值Ti的调整记录：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | 初值 | 第一轮调整 | 第二轮调整 |
| 1 | high | 0 | 0 |  |
| low | 0 | 0 |  |
| 2 | high | 100000 | 99999 |  |
| low | -10000 | 3 |  |
| 3 | high | 100000 | 10000 |  |
| low | -10000 | -10000 |  |
| 4 | high | 100000 | 4 |  |
| low | -10000 | -10000 |  |
| 5 | high | 100000 | -5 |  |
| low | -10000 | 1 |  |
| 调整情况 | | | high[5]<low[5]终止 | |

经第一轮调整，调整过程终止，即问题无解。

从样例2所给不等式也可看出，因为：

T1-T2≤-3，→T2>T1

T2-T5≤-1，→T5>T2

T5-T1≤-5，→T1>T5

这三个不等式不能同时成立，因此问题无解。

**3、公路修建**

三条规则中的第二条是故弄玄虚。如果三个或三个以上的城市申请修建的公路成环，那么这些公路的长度必然都相同，否则不满足“每个城市选择一个与它最近的城市，申请修建通往该城市的公路”。所以，如果成环，其实是任意去掉一条路。

本题要我们求的实际上是最小成成树，也就是说，按规则生成的是图的最小生成树。为什么呢？很显然，按规则生成的应该是树。根据规则：每个城市选择一个与它最近的城市，申请修建通往该城市的公路。那么，对于图中任意的环，环上最长边必被舍弃。这就与求最小生成树的“破环法”完全相符了。

用Prim算法求图中的最小生成树，最小生成树上各边的长度只和即是所求的答案。时间复杂度为O(n2)。

但是，本题还有其特殊性。本题是在Euclid空间求最小生成树，Euclid空间最小生成树有O(nlog2n)的算法，是用Voronoi图+Kruskal算法（或用Prim+heap代替Kruskal）实现的。

**4、K-联赛**

看一个队是否有希望夺冠，首先，这个队此后的比赛自然是赢越多越好，所以先让这个队把以后的比赛都赢下来，算出这个队最高能拿多少分。下面关键就看是否有可能让其他队的积分都低于刚才计算出的最高分。

建立一个网络，所有的球队作为图中的节点，每两个队之间的比赛也作为图中的节点。从网络的源各连一条边到“比赛的节点”，容量为两队间还剩的比赛场数。从“每个队的节点”都连一条边到网络的汇，容量为这个队当前的积分与最高分之差。如果一个“比赛的节点”代表的是A与B之间的比赛，那么从这个节点连两条边分别到“A队的节点”和“B队的节点”，容量为无穷大。

如果这个网络的最大流等于所有还未进行的比赛的场次之和，那么需要我们判断的那个队抗有可能夺得冠军。

本题要我们找出所有可能夺冠的队，那么只需枚举所有的球队，判断它们是否有可能夺冠即可。

**5、速度限制**

首先，利用预处理计算任意两个节点之间只经过无限速标志的路的最短距离。这可以用F1ovd算法得到，时间复杂度为O(n3)。

计算城市1到城市D之间最快路径时，只需对Dijkstra稍作修改即可：在Dijkstra算法中，用一个已计算出最短路径的节点去刷新其他节点当前最短路径长度时，除了要枚举有限速标志的路以外，还要在此路的基础上，枚举通过此路后要经过无限速标志的路到达的节点。时间复杂度为O(n2+mn)，即O(mn)。

**6、机器调度**

本题所求的是工作模式的最少切换次数，实际上也就是求最少需要使用多少个工作模式，因为一个工作模式被切换两次肯定是不合算的，一旦切换到一个工作模式就应该把这个工作模式可以完成的工作都完成。

将两台机器的工作模式分别看成n个和m个节点。jobi分别和机器A和B的mode\_x和mode\_y相关：jobi要被完成，就必须切换到机器A的mode\_x或切换到机器B的mode\_y。将jobi看作图中的一条边——连接节点x和节点y的边，那么这条边就要求x和y两个节点中至少要有一个节点被取出来。这正符合覆盖集的性质。

我们构成的图是二分图，要求最少的切换次数，就是要使覆盖集最小。二分图的最小覆盖集问题等价于二分图的最大匹配问题。因此，只需对此二分图求一个最大匹配即是我们要求的答案。时间复杂度。

**7、邮递员送信**

此题要求的目标：sum(dist(1,i))+sum(dist(I,1))，只需要以1为起点做一次最短路就可以求出sum(dist(1,i))，对于sum(dist(I,1))，需要将所有边都反向，然后也以1为起点做一次最短路即可。

**8、拐弯**

最短路问题，构造N\*M\*4个点，期中点(I,j,k)表示的意义是在(I,j)这个格子时的方向为k,然后对于(I,j,kk),如果k和kk是两个旋转90读能转化的方向，就连一条边权威1的边，而对于(I,j,k),和（i+dx[k],j+dy[k],k），连接一条边权威0的边，表示从（i，j）在方向为k的情况下能向k这个方向走一步到达（i+dx[k],j+dy[k]），因为起始和终止的方向不确定，故再添加一个源点和汇点，源点向起始位置的四个方向表示的点连边权为0的边，从终止位置的4个方向表示的点向汇点连边，然后SPFA求最短路即可。

**思考：**当N与M比较大时怎么办？

——可以构造图时，可以只构造拐点。

类似习题：最短路线（Walk, ACM/ICPC Jinhua 2012, UVa1666）

平面上有n（n≤50）个建筑物，求从(x1,y1)到(x2,y2)的一条路，使得转弯次数最少。建筑物都是坐标平行于坐标轴的矩形，可以相互接触但不会重叠（接触的点或者边都不能通过）。你只能沿着平行于坐标轴的直线走，可以沿着建筑物的边走，但不能穿过建筑物。无解输出-1。

提示：本题在细节上容易出错。