

2012 年全国青少年信息学奥林匹克冬令营

竞赛时间：2012 年 2 月 11 日 8:00–13:00

题目名称	最小生成树	记忆中的水杉树	卡坦传奇
目录	mst	memory	catan
可执行文件名	mst	memory	catan
输入文件名	mst.in	memory.in	catan1.in ~ catan10.in
输出文件名	mst.out	memory.out	catan1.out ~ catan10.out
每个测试点时限	4 秒	3 秒	/
内存限制	256MB	256MB	/
测试点数目	10	10	10
每个测试点分值	10	10	10
是否有部分分	否	是	是
题目类型	传统型	传统型	提交答案型

提交源程序须加后缀

对于 C++ 语言	mst.cpp	memory.cpp	/
对于 C 语言	mst.c	memory.c	/
对于 Pascal 语言	mst.pas	memory.pas	/

注意：最终测试时，所有编译命令均不打开任何优化开关。

最小生成树

【问题描述】

给定无向带权连通图 G ，我们希望通过修改边的权值，使它的最小生成树唯一。已知减小、增加一条边的权值的单位代价分别为 a 和 b ，且修改后的权值必须为非负整数。

例如，对某个图 G ，如果将一条边的权值减 3、另一条边的权值加 2 之后，它的最小生成树唯一，则此时的代价之和是 $3a+2b$ 。试计算代价之和的最小值。

【输入格式】

输入文件 *mst.in* 的第一行包含数据编号，对于第 i 个数据，第一行将包含字符串 “mst i ”。

第二行包含 4 个正整数 n, m, a, b ，分别表示图 G 顶点的个数、边的条数，以及对一条边的权值减 1、加 1 的代价。

接下来 m 行，每行 3 个正整数 x, y, w ，表示顶点 x 和顶点 y 之间连有一条初始权值为 w 的边。顶点由 1 至 n 编号。

【输出格式】

输出文件 *mst.out* 仅包含一行，包含一个非负整数，即要求的最小值。如果无需修改，即图本身的最小生成树就是唯一的，则输出 0。

【样例输入】

```
mst 0
4 5 2 3
1 2 1
1 3 1
2 3 1
2 4 2
3 4 2
```

【样例输出】

```
5
```

【样例说明】

将边(2, 4)的权值减 1，边(2, 3)的权值加 1 之后，图 G 的最小生成树唯一，且此时的代价之和取到最小值。

【数据规模和附加文件】

测试数据编号	规模和约定
1	$n \leq 7, m \leq 10, a = 10^9, b = 1$
2	$n \leq 50,000, m \leq 1,000,000, a = 10^9, b = 1$
3	$n \leq 100, m \leq 1,000, a = 1, b = 10^9$
4	$n \leq 500,000, m \leq 1,000,000, a = 1, b = 10^9$ 数据见用户目录
5	
6	
7	
8	
9	
10	

对于 100% 的数据中所有的边，有 $1 \leq w \leq 1,000,000$ 。

记忆中的水杉树

【问题描述】

江苏省常州高级中学是一所百年名校，这里萦绕着无数人难以忘怀的回忆。

Will 记得，在他小的时候，常州高级中学改建以前，学校里有一片高大的水杉林，每到水杉落叶之时，针状的叶子会像毯子一样盖在地上，走在上面浪漫而又闲适。那时，Will 和同学们还喜欢用这些针叶，在水杉树下，玩“取叶子”的游戏。

游戏一开始，大家先将 n 片针叶平铺在地上。接着，每一轮可以有一个同学选择一片针叶，按水平或者垂直方向将针叶移走（也就是平移到无穷远处）——当然，前提是移动过程中不被任何尚未移走的针叶所阻碍。如果某一轮针叶的移动会被阻碍，那么这次移动就是非法的，是不被允许的。 n 轮过后，当针叶都被移走时，游戏也就结束了。

针叶并不是任何时刻都可以被移动的。当针叶很多的时候，判断每一轮中一片针叶是否可以按一个特定的方向移动是一件很麻烦的事情。

现在我们将地面抽象为平面直角坐标系， n 片针叶抽象为平面上 n 条互不相交的线段，并将其从 1 到 n 编号，Will 还将给出每一轮游戏中，他想要移动的针叶编号以及移动方向，请你帮助他：

1) 找出最早的一次非法移动出现在哪一轮；

2) 给出一个合法的移动方案完成这个游戏。

注意：在线段移动时仅端点接触不会造成阻碍，具体请参见样例。

【输入格式】

输入文件 *memory.in* 的第一行包含一个正整数 n ，表示针叶的数量。

接下来 n 行，每行 4 个整数，描述针叶的位置信息。其中第 i 行的整数为 a_i , b_i , c_i , d_i ，表示编号为 i 的针叶所抽象成的线段的端点为 (a_i, b_i) 和 (c_i, d_i) 。

接下来 n 行，每行 2 个整数，描述移动操作。其中第 i 行的整数为 p_i , q_i ，表示第 i 轮移动的针叶编号为 p_i ，方向为 q_i 。其中 q_i 为一个 0 到 3 之间的整数，0 表示向左平移（即 x 轴负方向），1 表示向上平移（即 y 轴正方向），2 表示向右平移，3 表示向下平移。

输入数据保证：

- 所有线段长度为正，两两之间没有公共点，且不存在垂直或者水平的线段；
- p_1 到 p_n 恰好组成一个 1 到 n 的排列；
- Will 所给出的移动操作中一定存在非法移动；
- n 轮均合法的移动操作总是存在的。

【输出格式】

输出文件 *memory.out* 一共包含 $n + 1$ 行。

输出的第一行包含一个 1 到 n 之间的整数，表示最早出现非法移动的是哪一轮。

接下来 n 行，每行两个整数，内容同输入格式所述，描述一个合法的移动序列。

【评分标准】

对于一个测试点：

- 如果非法移动判断正确，但是给出的移动方案错误，可以得到 5 分；
- 如果非法移动判断错误，但是给出的移动方案正确，可以得到 5 分；
- 如果非法移动的判断与给出的移动方案均正确，则可以得到 10 分；
- 否则，得 0 分。

注意：如果程序的输出格式不正确，将被直接判作 0 分。

【样例输入 1】

```
5
2 5 5 8
2 1 3 5
5 2 6 5
7 0 4 2
3 1 4 0
2 0
3 0
4 0
1 2
5 1
```

【样例输出 1】

```
3
2 0
3 0
4 3
1 2
5 1
```

【样例说明 1】

在 Will 给出的移动方案的第 3 轮中，编号为 4 的针叶向左移动会被编号为 5 的针叶阻碍。

【样例输入 2】

```

4
-1 1 2 3
13 5 9 8
10 10 15 14
10 17 0 20
3 1
2 1
1 1
4 1

```

【样例输出 2】

```

2
4 1
3 1
2 1
1 1

```

【数据规模和约定】

测试数据编号	n	其他约定	
1	$n = 3$	对任意 $2 \leq i \leq n$	对任意 $1 \leq i \leq n$ 有 $q_i = 1$ $ a_i , b_i , c_i , d_i \leq 10^4$
2	$n \leq 8$	有 $b_{i-1} < d_{i-1} < b_i < d_i$	
3	$n \leq 100$	对任意 $1 \leq i \leq n$	
4	$n \leq 2,000$	有 $a_i < c_i, d_i - b_i = 1$	
5	$n \leq 2,000$	/	对任意 $1 \leq i \leq n$ 有 $0 \leq q_i \leq 3$ $ a_i , b_i , c_i , d_i \leq 10^9$
6	$n \leq 20,000$	/	
7	$n \leq 30,000$	/	
8	$n \leq 50,000$	/	
9	$n \leq 80,000$	/	
10	$n \leq 100,000$	/	

卡坦传奇

【问题描述】

栋栋最近迷上了一款名为卡坦岛的游戏，这个游戏在一个地图上进行，这个地图上每个时刻会产生一些资源，玩家的主要目标是获取资源和建造房屋。栋栋最近正在研究游戏的策略，他希望从一些简单的情况入手。为此，栋栋简化了游戏并给出了一些游戏的局面，希望你能帮助他找到这些游戏局面的优秀解法。

栋栋简化后的游戏是在一个由 $n \times m$ 个方格组成的地图上进行的，如下图所示。



为了描述方便，我们用 $[r, c]$ 表示上图中的第 r 行 ($1 \leq r \leq n$) 第 c 列 ($1 \leq c \leq m$) 的方格。 $[r, c]$ 的左上角坐标用 $(r-1, c-1)$ 表示，右下角坐标用 (r, c) 表示（请注意圆括号和方括号的区分）。每一个方格 $[r, c]$ 都有一个资源属性 $S[r, c]$ 和一个点数属性 $D[r, c]$ 。其中资源属性 $S[r, c]$ 只能是 1, 2, 3, 4 中间的一个，而点数属性 $D[r, c]$ 可以是任意非负整数。

游戏玩家可以通过建造房屋来获得方格中产生的资源。房屋有两种：普通房和城堡，都只能建在方格的顶点上。要建造一个普通房需要的四种资源数量分别为 h_1, h_2, h_3, h_4 。普通房可以升级为城堡，升级所需要的四种资源数量分别为 l_1, l_2, l_3, l_4 。如果直接建造一个城堡，需要的四种资源数量分别为 $h_1+l_1, h_2+l_2, h_3+l_3, h_4+l_4$ 。在建造房屋时，任意两个房屋之间的直线距离必须严格大于 1，且除首次外的建造位置必须与某个已建路段相连。

路段也需要由玩家建造，每个路段都是某个方格的一条边。建造一个路段所需要的四种资源数量分别为 a_1, a_2, a_3, a_4 ，且必须与已建造的路段或房屋有公共点。路段可以重复建造，重复建造时花费的资源数量仍为 a_1, a_2, a_3, a_4 （重复建造路段一般是没有意义的，但规则允许这么做）。

每个时刻 t ，游戏生成一个数字 V_t ，每个点数 $D[r, c]$ 为 V_t 的方格都会产生编号为 $S[r, c]$ 的资源，它四个角上的房屋（如果有的话）会得到相应的资源：任意普通房将获得 1 个 $S[r, c]$ 号资源，任意城堡将获得 2 个 $S[r, c]$ 号资源。这些资源都汇总到玩家手中。例如，在一个资源属性为 4 的方格的四个角上，有一个普通房和一个城堡，则在时刻 t ，如果这个方格的点数等于 V_t ，玩家会从这个方格获得 3 个资源 4。在同一时刻，一个房屋可能从多个方格得到资源。

在游戏开始时，玩家有 4 种资源数量分别为 h_1, h_2, h_3, h_4 。利用这些资源，玩家可以在任意位置建造一个普通房。

任何时刻, 玩家可以用 K 个相同的资源换成 1 个另一种资源, 其中 K 是游戏规定的值。比如他可以使用 K 个资源 3 换 1 个资源 4。

现在, 栋栋告诉了你地图和每个时刻游戏生成的数字 V_t , 需要你在尽量少的时间内, 使得城堡个数的两倍 + 普通房个数至少为 10。

【输入格式】

这是一道提交答案的试题, 在你的目录下有 10 个输入文件 *catan1.in* ~ *catan10.in*。

输入的第一行包含两个整数 n, m , 分别表示方格的行数和列数。

接下来 n 行表示所有方格的资源属性, 其中第 r 行有 m 个整数, 第 r 行第 c 列的整数表示 $S[r, c]$ 。

接下来 n 行表示所有方格的点数属性, 其中第 r 行有 m 个整数, 第 r 行第 c 列的整数表示 $D[r, c]$ 。

接下来的一行包含 4 个整数 h_1, h_2, h_3, h_4 , 分别表示建造普通房所需要的 4 种资源数量。

接下来的一行包含 4 个整数 l_1, l_2, l_3, l_4 , 分别表示从普通房升级城堡所需要的 4 种资源数量。

接下来的一行包含 4 个整数 a_1, a_2, a_3, a_4 , 分别表示建造路段所需要的 4 种资源数量。

接下来的一行包含 1 个整数 K , 表示玩家可以使用 K 个相同的资源替换成 1 个另一种资源。

接下来的一行包含 1 个整数 G , 表示给出的时间长度。

最后一行包含 G 个整数 V_1, V_2, \dots, V_G , 分别表示每个时刻游戏生成的数字。

【输出格式】

对于每一个输入文件, 在目录下给出对应的输出文件 *catan*.out*。

输出的第一行包含一个整数 ans , 表示建造足够数量的房屋 (即城堡个数的两倍 + 普通房个数至少为 10) 所需要的时间长度。

接下来两行, 表示游戏开始时玩家的方案。其中第一行包含一个大写字母 **B** 和两个整数 x_0, y_0 , 表示初始时在 (x_0, y_0) 位置建造一个普通房。其中第二行包含一个大写字母 **E**。

接下来 ans 个部分, 表示游戏的方案。

其中第 i 个部分包含若干行, 表示在时刻 i 的资源产生后玩家的动作。

- 若一行包含一个大写字母 **B** 和两个整数 x, y , 表示在 (x, y) 位置建造一个普通房。
- 若一行包含一个大写字母 **C** 和两个整数 x, y , 表示在 (x, y) 位置建造一个城堡。
- 若一行包含一个大写字母 **U** 和两个整数 x, y , 表示将 (x, y) 位置的普通房升级为城堡。
- 若一行包含一个大写字母 **R** 和四个整数 x_1, y_1, x_2, y_2 , 表示在 (x_1, y_1) 和 (x_2, y_2) 两个位置间建造一个路段。其中 (x_1, y_1) 和 (x_2, y_2) 的距离必须为 1。
- 若一行包含一个大写字母 **X** 和两个整数 p, q , 表示将 K 个资源 p 换成一个资源 q 。

- 若一行包含一个大写字母 **E**，表示该部分的结束。

注意：你的输出文件的大小不能超过 1MB。

【评分标准】

对于每个测试点，如果你没有输出、输出不合法或在给定的时间 G 内不能建造足够数量的房屋，则得 0 分。

否则，对于每个数据，我们设有 9 个评分参数 m_2, \dots, m_{10} 。

- 若 $ans \leq m_{10}$ ，得 10 分；
- 若 $m_{i+1} < ans \leq m_i$ ，得 i 分；
- 若 $ans > m_2$ ，得 1 分。

【如何测试你的输出】

在你的目录下有一个程序 `checker` 可以用来测试你的输出结果，你可以在终端中使用以下命令来检查你的输出结果：

```
./checker N
```

其中 N 为测试点的编号，例如，要测试第 3 个测试点可以使用

```
./checker 3
```

该程序会检测你的输出是否合法。对于合法的输出，`checker` 会输出“Right.” 否则会输出错误信息。

【样例输入】

```
1 10
1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1
1 0 0 0
1 0 0 0
1 0 0 0
100
10
1 1 1 1 1 1 1 1 1 1
```

【样例输出】

```
6
B 0 0
E
U 0 0
E
R 0 0 0 1
R 0 1 0 2
E
B 0 2
U 0 2
E
R 0 2 0 3
R 0 3 0 4
B 0 4
U 0 4
E
R 0 4 0 5
R 0 5 0 6
B 0 6
U 0 6
R 0 6 0 7
R 0 7 0 8
E
B 0 8
U 0 8
E
```

【样例说明】

以上是一种可能的解法，需要的总时间为 6。请注意这个答案不一定是最优的，可能存在更优的解法。