黄冈中学信息奥赛训练题

测试时间 8:30—12:00

（请仔细阅读本页面内容）

一．题目概况

中文题目名称	差分约束	红绿灯	红石电路
英文题目与子目录名	differ	traffic	redstone
可执行文件名	differ	traffic	redstone
输入文件名	differ.in	traffic.in	redstone.in
输出文件名	differ.out	traffic.out	redstone.out
每个测试点时限	1s	1s	1s
测试点数目	10	10	10
每个测试点分值	100	10	10
附加样例文件	有	有	有
结果比较方式	全文比较（过滤行末空格及文末换行）		
题目类型	传统	传统	传统
运行内存上限	512M	512M	512M

二．提交源程序文件名

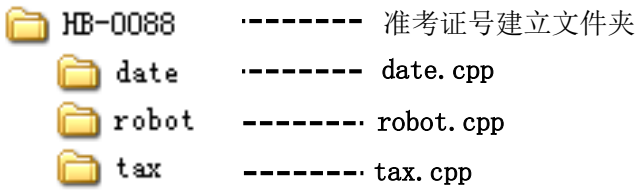
对于 C++文件	differ.cpp	traffic.cpp	redstone.cpp
对于 C 文件	differ.c	traffic.c	redstone .c

三．编译命令

对于 C++文件	g++.exe %s.cpp -o %s.exe -lm –O2
对于 C 文件	gcc.exe %s.c -o %s.exe -lm

注意事项：

1. 文件名（程序名和输入输出文件名）必须使用英文小写。
2. C/C++中的函数 main()的返回值类型必须是 int，程序正常结束时的返回值必须是 0。
3. 评测时允许使用万能头文件 <bits/stdc++.h>，除特别说明除外，c++默认标准为 c++14。
4. 程序执行时堆栈空间限制与内存空间限制相同，Linux 系统下使用终端命令 ulimit –s 修改堆栈大小。
5. 提交的文件目录如下图所示，HB-0088 为考生准考证号，date、robot、tax 为题目规定的英文名称，目录中只包含源程序，不能包含其他任何文件。



差分约束(differ)

【题目描述】

这是一道差分约束基础练习题。

给出 n 个约束条件，每个约束条件属于

$$xi - xj \leqslant w, \quad xi - xj \geqslant w, \quad xi - xj = w$$

三种形式之一。

给出 Q 个询问，每个询问都为：求$xi - xj$的最大值。

【输入格式】

从文件 differ.in 中读入数据。

第一行输入两个正整数 m,n，表示变量的个数和约束的个数。

接下来 n 行，每行四个整数t，i，j，w：

t 为 0,1,2 分别表示≤，≥，=；i，j，w的意义同题目描述。

接下来一行一个正整数 Q。

接下来 Q 行，每行两个正整数i，j，表示询问$xi - xj$的最大值。

【输出格式】

输出到文件 differ.out 中。

输出 Q 行，对于每个询问，输出答案，如果可以无限大，输出 Infinity 。

【样例输入】

```
4 3
0 1 2 10
1 2 3 -5
2 1 3 -2
2
1 3
1 4
```

【样例输出】

```
-2
Infinity
```

【数据范围与约定】

对于 30%的数据，n,m<=5。

对于 100%的数据，m<=200，n<=10000，Q<=100000，|w|<=10000。

保证约束条件不存在冲突。

红绿灯(traffic)

【题目描述】

家住 D 市的小 D 喜欢开车。

今天他想从 A 路口开车到 B 路口。D 市的道路可以简化成一个无向简单图，n 个路口通过 m 条马路相连。

每个路口都安装了红绿灯，为了简化模型，我们假设所有路口的红绿灯都只有红绿灯交替亮起。交替规则为红灯亮ri秒，绿灯亮gi秒，以此循环。

在小 D 从 A 路口出发的时刻，所有的路灯都处在绿灯恰好熄灭红灯恰好亮起的状态。

小 D 可以在任意时刻到达某个路口，但是只有当绿灯亮起时，才能从所在路口离开。

小 D 想知道从 A 出发最快多久能到达 B。

【输入格式】

从文件 traffic.in 中读入数据。

第一行四个正整数 n, m, A, B，表示路口数、马路数、起点和终点。

接下来 n 行，每行两个正整数ri, gi，意义如题目描述。

接下来 m 行，每行三个数 u, v, w，表示 u 路口和 v 路口之间有一条通过耗时为 w 的双向马路。

保证存在 A 到 B 的路径。

【输出格式】

输出到文件 traffic.out 中。

输出一行一个整数，即 A 到 B 的最短用时。

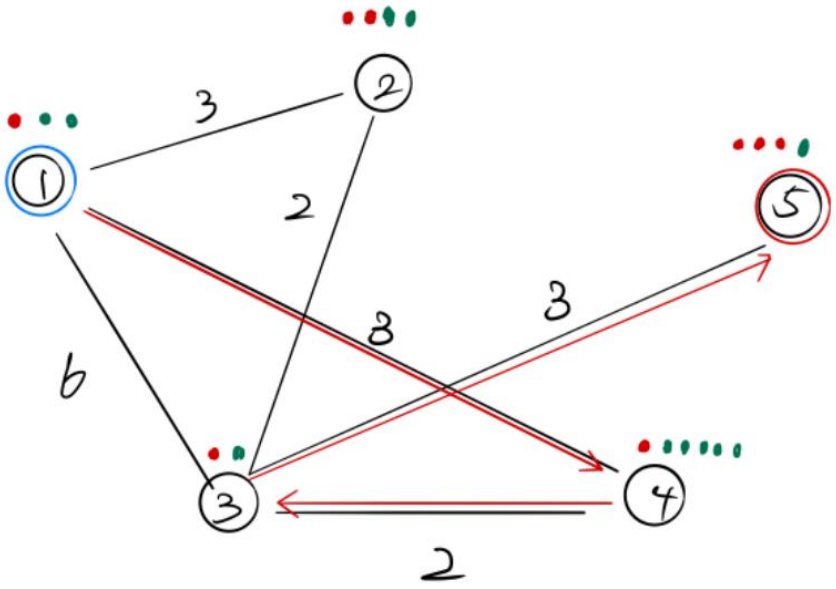
【样例输入】

```
5 6 1 5
1 2
2 2
1 1
1 5
3 1
1 4 3
4 3 2
1 2 3
3 5 3
2 3 2
1 3 6
```

【样例输出】

```
10
```

【样例解释】



【数据范围与约定】

对于 30%的数据，n<=10, m<=20。
对于 100%的数据，n<=250000, m<=500000, ri,gi<=10000, w<=1000000。

红石电路(redstone)

【题目描述】

小 M 喜欢 Minecraft。

小 M 利用 Minecraft 里的红石搭建了一个电路。电路有 n 个延时器，m 条红石导线。每条导线连接两个不同的延时器。导线可以单向传导电信号。我们认为电信号在导线上的传导是瞬时的。

每个延时器连接若干条入导线和若干条出导线（如一条导线连接$a \rightarrow b$，则其是a的出导线和b的入导线）。延时器自身也有延时参数 t。当延时器从连接的所有入导线都接收到电信号时，它会开始计时 t 秒。计时结束后，它会向所有出导线发出电信号。如果一个延时器没有入导线连接，它实际上会连接到电源，在开始实验的那一时刻（0 秒时）开始它的计时工作。保证导线的连接不成环。

现在，小 M 有一个红石电路，他想知道从开始实验接通电源（0 秒时），经过多少时间后，所有的延时器都能完成其计时工作。同时，他还想知道，是否存在某些“关键延时器”，使得仅仅减小其延时参数就能使上一问的答案减小。他希望输出所有的“关键延时器”编号。

【输入格式】

从文件 redstone.in 中读入数据。

输入的第一行包含一个整数 n，表示延时器的数量。

接下来 n 行。

每行第 1 个整数为$ti(1 \leqslant ti \leqslant 10000)$，表示编号为 i 的延时器的延时参数。

第 2 个整数为 $ci(1 \leqslant ci \leqslant 10000)$，表示该延时器的入导线数量。接下来 ci个互不相同的整数，表示该入导线的另一端延时器的编号。相邻两数之间用一个空格隔开。

保证导线的连接不形成有向环。

【输出格式】

输出到文件 redstone.out 中。

第一行一个整数，表示所有的延时器都能完成其计时工作的最短时间。

第二行一个整数 Num，表示“关键延时器”的个数。

第三行 Num 个整数，从小到大输出所有“关键延时器”的编号。

【样例输入】

```
5
6 3 4 5 2
1 1 5
9 2 2 5
10 0
6 1 4
```

【样例输出】

26
4
2 3 4 5

【样例解释】

4 号点开始计时，10s 时计时完成，5 号点可以开始计时。
5 开始计时，16s 时完成计时，2 号点可以开始计时。
2 号点开始计时，17s 时完成计时，1,3 号点可以开始计时。
1 号点、3 号点开始计时，1 号点 23s 时完成计时，3 号点 26s 时完成计时。
1 号点延时系数减少并不会影响 3 号点的最终完成。

【数据范围与约定】

对于 30%的数据，n<=10。
对于 50%的数据，n<=1000。
对于 100%的数据， n<=100000，∑ci<=1000000。