CSP-S 模拟赛

2019 年 11 月 04 日　08:00 - 11:30

题目概况

题目名称	学农	平面嵌入	小球
源程序文件名	`farm.c/cpp/pas`	`planar.c/cpp/pas`	`ball.c/cpp/pas`
每个测试点时限	1 秒	1 秒	2 秒
测试点数目	捆绑测试	捆绑测试	捆绑测试
每个测试点分值			
运行内存上限	512 MB	512 MB	1024 MB
题目类型	传统题	传统题	传统题
结果比较方式	Special Judge	Special Judge	全文比较

※ 除特殊说明外，输入、输出文件的数字之间均以恰一个空格分隔，行末字符为 Line Feed（`'\n'`，ASCII 10），且该字符也存在于输入文件的最后一行末尾。

※ 如无特殊说明，全文比较会过滤行末空格及文末回车。

编译命令（以第一题为例）

C	`gcc -o farm farm.c -O2 -lm`
C++	`g++ -o farm farm.cpp -std=c++11 -O2 -lm`
Pascal	`fpc farm.pas -O2`

Problem A. 学农

输入文件: farm.in
输出文件: farm.out
时间限制: 1 秒
内存限制: 512 MB

OM 和 IO 竞赛退役的小 U 开启了他的学农之旅。当小 U 的同学们正在地里挖红薯时，教官突然喊了一句“集合了！”。

此时，还有很多同学仍然在挖红薯。把铲子还掉，把红薯放进袋子里，是一件繁重、费力、又费时间的工作。为了不让教官生气，小 U 班级的 n 个人想要在尽量短的时间内完成自己挖红薯的任务，并回到一个指定的地点。

如果我们把学农的场地看成一个无限大的二维平面的话，那么第 i 个同学所在位置是 (x_i, y_i)，且他完成挖红薯任务的时间为 t_i 个单位。所有同学会同时开始挖红薯。

班长可以决定小 U 班级的 n 个人的集合位置 (X, Y)，他需要让完成挖红薯任务并且回到指定位置 (X, Y) 的时间最晚的人回到指定位置时的时间距离开始挖红薯的时间最短（X, Y 可以不为整数）。

值得注意的是，由于农田是方方正正的，所以同学们很难沿着不平行于坐标轴的方向行走。因此，第 i 个人从完成任务到走到指定集合位置的时间为 $|x_i - X| + |y_i - Y|$。

当然，小 U 的班级都忙着挖红薯的任务，这样的问题就交给你了！

输入格式

第一行包含一个整数 n, 表示小 U 班级的人数。

接下来 n 行，每行有 3 个整数 x_i, y_i, t_i，表示第 i 个人所在的位置和他完成挖红薯任务所需要的时间。

输出格式

输出一行一个小数，表示最后一个回到集合位置的人回到集合位置的时间距离开始挖红薯的时间的最小值。如果你输出的答案和正确答案的差距不超过 10^{-6}，那么就认为你的答案正确。

样例 1

farm.in	farm.out
2 1 1 1 0 0 1	2.000000

样例解释 1

选择集合位置为 $(0.5, 0.5)$，则每个人挖红薯用了 1 单位的时间，且回到集合位置也用了 1 单位的时间。

样例 2

见选手文件夹中的 farm2.in/ans。

数据范围

对于所有测试数据，满足 $n \leq 10^6, |x_i|, |y_i| \leq 10^9, 0 \leq t_i \leq 10^9$。

子任务 1 (分值: 33)

$n, |x_i|, |y_i| \leq 10^2$.

子任务 2 (分值: 33)

$x_i = 0$.

子任务 3 (分值: 34)

无特殊限制.

Problem B. 平面嵌入

输入文件:	planar.in
输出文件:	planar.out
时间限制:	1 秒
内存限制:	512 MB

本题中的距离均指欧几里得距离。

小 Z 有一个 $r \cdot c$ 个点无向图，点的编号为 $1 \dots r \cdot c$，每条边有一个权值。

小 Z 的无向图非常特殊，这个无向图可以由正整数 r, c 和 m 个不超过 rc 的正整数 $a_1, a_2 \dots a_m$ 确定。具体来说，这个无向图上的边可以分为三类：

1. 对于整数 $1 \le i < r, 1 \le j \le c$，点 $(i-1)c+j$ 与点 $ic+j$ 之间有一条权值为 1 的边。
2. 对于整数 $1 \le i \le r, 1 \le j < c$，点 $(i-1)c+j$ 与点 $(i-1)c+j+1$ 之间有一条权值为 1 的边。
3. 对于整数 $1 \le i < r, 1 \le j < c$，如果存在整数 $1 \le k \le m$ 使得 $(i-1)c+j = p_k$，那么点 $(i-1)c+j$ 与点 $ic+j+1$ 之间有一条权值为 $\sqrt{2}$ 的边。

这个无向图上不存在上述三类边以外的边。小 Z 希望，你能给这个无向图上的每个点指定一个平面上的坐标，把这个无向图画在平面上：在对应的坐标上画出每个点，如果有一条边 (u, v)，就以点 u 和点 v 为端点画一条线段。

小 Z 认为，一种把这个无向图画在平面上的方案是合法的，当且仅当对于任意一条权值为 w 的边 (u, v)，点 u 和点 v 的坐标在平面上的距离恰好为 w，且任意两个点的坐标不同。小 Z 认为一个方案是优美的，当且仅当这个方案是合法的，且存在整数 $1 \le i < r, 1 \le j < c$，点 $(i-1)c+j$ 与点 $ic+j+1$ 之间的距离不等于 $\sqrt{2}$。

小 Z 想得到一个优美的方案，于是他找到了你，希望你能帮助他找到一个满足他的要求的方案。

本题下发文件中的 .ans 文件仅包含一行 Yes 或 No 表示是否存在满足要求的方案。

输入格式

第一行包含三个正整数 r, c, m，意义如题目描述所述。

第二行包含 m 个整数，第 i 个整数表示 p_i 的值。

输出格式

如果不存在满足小 Z 的要求的方案，输出一行 No。

否则，第一行输出 Yes。接下来的 rc 行，每行两个浮点数 x_i 和 y_i，表示点 i 的横坐标和纵坐标。你必须以小数（如果是整数，小数点可以省略）的形式输出 x_i, y_i，并保证 $|x_i|, |y_i| \le 10^6$，且小数位数不超过 8，否则你的程序可能会在这个测试点得到 WA。请尽量不要使用 cout，以免 x_i, y_i 被以科学计数法的形式输出。

评分标准

如果你试图绕过 SPJ 得分，你的得分将被人工修改为 0 分。

如果你的输出了 inf 或 nan，或你的输出不符合题目中要求的输出格式导致 SPJ 无法正确读取，你可能会因此获得 0 分。

对于一个测试点，如果这个测试点存在一种满足要求的方案，而你在第一行输出了 No，你会在这个测试点获得 0 分；如果你在第一行输出了 Yes，SPJ 会读入你的方案，并检验你的方案是否满足要求。由于浮点数可能存在误差，在判定两点间的距离是否等于边权或判定两点坐标是否相等时，一个实数 x 被认为等于实数 w，当且仅当 $|x-w| \leq 10^{-6}$；在检验是否存在 i, j 使得点 $(i-1)c+j$ 到点 $ic+j+1$ 的距离不等于 $\sqrt{2}$ 时，一个实数 x 被认为等于实数 w，当且仅当 $|x-w| \leq 10^{-4}$。如果你的方案被认为是满足要求的，你将在这个测试点获得 1 分，否则你将在这个测试点获得 0.5 分。如果这个测试点不存在一种满足要求的方案，而你在第一行输出了 Yes，你会在这个测试点获得 0 分；如果你在第一行输出了 No，你将获得 1 分。

对于满分为 S 的子任务，设你在这个子任务的所有测试点中得分的最小值为 k，则你在这个子任务中的得分为 kS。

更多细节见下发文件中的 planar_spj.cpp。

在评测时使用的 SPJ 可能与下发的 SPJ 有不同。

如何测试你的程序

以 Linux 下为例。

首先，编译下发的 planar_spj.cpp：

```
g++ planar_spj.cpp -lm -o planar_spj
```

请把需要测试的样例的.in 文件，.ans 文件，你的程序输出的.out 文件和编译好的可执行文件 planar_spj 放在同一目录下。假设.in 文件名为 planar1.in，.ans 文件名为 planar1.ans，.out 文件名为 planar1.out，执行以下命令：

```
./planar_spj planar1.in planar1.out planar1.ans 1 score.txt report.txt
```

你的得分将被输出到 score.txt 中，具体信息会被输出到 report.txt 中。

样例 1

planar.in	planar.out
2 3 1 2	Yes 0 0 0.8660254 -0.5 1.8660254 -0.5 0 -1 0.8660254 -1.5 1.8660254 -1.5

样例解释 1

这个方案如图所示。

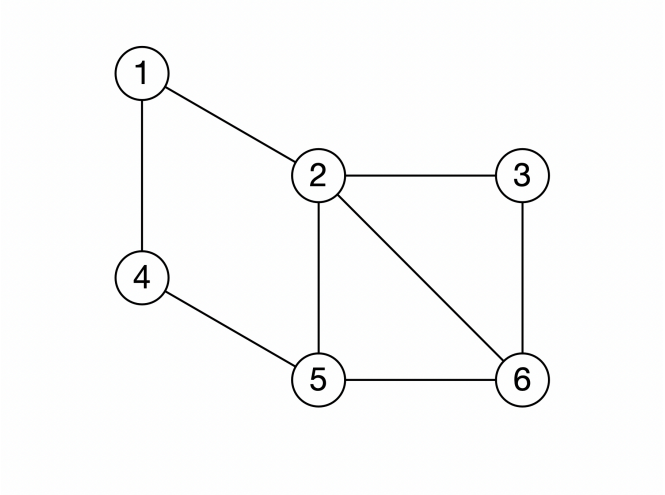

图 1: 样例 1 中的方案

样例 2

见选手文件夹中的 planar2.in/ans。

样例 3

见选手文件夹中的 planar3.in/ans。

样例 4

见选手文件夹中的 planar4.in/ans。

数据范围

对于所有测试数据，满足 $2 \leq r, c \leq 500, m \leq 2.5 \cdot 10^5, 1 \leq p_i \leq rc$，$p_i$ 的值两两不同。

子任务 1 (分值: 10)

$m = 0$.

子任务 2 (分值: 10)

$m = 2$.

子任务 3 (分值: 10)

$r = c = 2$.

子任务 4 (分值: 10)

$r = 2$.

子任务 5 (分值: 20)

$r = c = 3$.

子任务 6 (分值: 40)

无特殊限制.

提示

本题的数据范围中，p_i 的范围**仅保证** $1 \leq p_i \leq rc$ 和 p_i 两两不同。

Problem C. 小球

输入文件:	ball.in
输出文件:	ball.out
时间限制:	2 秒
内存限制:	1024 MB

小 Z 有一个 n 个点的有根树，点被编号为 $1 \dots n$，其中点 1 为根。

初始时树上每个点都没有放小球。小 Z 会进行 q 次操作，操作有两种：

- `1 u` 在点 u 上放一个小球，保证执行这个操作时，点 u 上没有小球。
- `2 u` 移除点 u 上的小球，保证执行这个操作时，点 u 上有小球。

任意时刻，如果存在一个小球，他所在的点存在一个没有小球的儿子，那么这个小球会开始下落。具体来说，设 p_u 是点 u 子树中没有小球的点编号的最小值（如果 u 子树中不存在没有小球的点，那么 $p_u = +\infty$），小球会移动到当前所在的点的儿子中，p_v 最小的儿子 v。可以证明，任意时刻，至多只有一个小球所在的点存在没有小球的儿子，所以按照这个规则，小球的移动方式是唯一的。下落是瞬间完成的，在每次操作开始时，树上一定不会存在可以下落的小球。

小 Z 想知道，对于每个操作 `1 u`，放置在点 u 的小球，在放置这个小球引起的所有下落完成之后，这个小球所在的点的编号。因为小 Z 是一个很菜的 IOer，所以他不知道怎么解决这个问题，希望你能帮帮他。

输入格式

第一行包含两个整数 n, q，表示点数和操作个数。

第二行包含 $n-1$ 个整数，第 i 个整数表示点 $i+1$ 的父亲的编号。

接下来的 q 行每行包含两个整数 op, u，意义如题目描述中所述。

输出格式

对每个 $op = 1$ 的操作，输出一个整数表示放置的小球在所有下落完成之后所在的点的编号。

提示

本题读入量较大，建议使用较快的读入方式。

样例 1

ball.in	ball.out
5 9 5 1 3 3 1 1 1 1 1 1 1 1 1 1 2 5 2 4 2 2 1 3	2 4 5 3 1 5

样例 2

见选手文件夹中的 ball2.in/ans。

样例 3

见选手文件夹中的 ball3.in/ans。

样例 4

见选手文件夹中的 ball4.in/ans。

数据范围

对于所有测试数据，满足 $n, q \leq 7 \cdot 10^5, op \in \{1, 2\}$。保证输入的是一棵树。

子任务 1 (分值: 10)

$n, q \leq 100$.

子任务 2 (分值: 20)

$n, q \leq 1000$.

子任务 3 (分值: 20)

$n, q \leq 10^5, op = 1, u = 1$.

子任务 4 (分值: 20)

$n, q \leq 10^5$.

子任务 5 (分值: 30)

无特殊限制。