2026 河南省实验中学 暑期集训

CSP-S / NOIP 模拟赛

第一场

时间：2026 年 8 月 25 日 08:30 ～ 13:00

题目名称	完美匹配	寻宝游戏	防御阵列	通讯网络
题目类型	传统型	交互型	传统型	传统型
目录	`match`	`finale`	`station`	`telecom`
可执行文件名	`match`	`finale`	`station`	`telecom`
输入文件名	`match.in`	`finale.in`	`station.in`	`telecom.in`
输出文件名	`match.out`	`finale.out`	`station.out`	`telecom.out`
每个测试点时限	1.0 秒	5.0 秒	2.0 秒	1.0 秒
内存限制	512 MiB	512 MiB	512 MiB	512 MiB
测试点数目	10	20	20	25
测试点是否等分	是	是	是	是

提交源程序文件名

对于 C++ 语言	`match.cpp`	`finale.cpp`	`station.cpp`	`telecom.cpp`

编译选项

对于 C++ 语言	`-O2 -std=c++14 -static`

注意事项（请仔细阅读）

1. 文件名（程序名和输入输出文件名）必须使用英文小写。
2. 请使用文件读写操作。
3. `main` 函数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。
4. 提交的程序代码文件的放置位置不包含子文件夹。
5. 若无特殊说明，结果的比较方式为全文比较（过滤行末空格及文末回车）。
6. 选手提交的程序源文件必须不大于 100KB。
7. 程序可使用的栈空间内存限制与题目的内存限制一致。
8. 评测机以出题人电脑为准：12th Gen Intel(R) Core(TM) i5-1240P 1.70 GHz，16.0 GB 运行内存（实测稍快于 Luogu 评测机，请各位选手放心并正常作答）。
9. 评测在 LemonLime 上进行，以 CCF 官方编译器版本为准。

完美匹配（match）

【题目背景】

小 F 是一个讨厌试题难度与学生能力不符的人，他现在有 n 套题目和 m 个学生，每个学生和试题之间的关系达到一定要求，小 F 才会让学生做这套题，否则，小 F 认为这是一种浪费题目也是浪费学生时间的行为。

【题目描述】

小 F 有 n 套卷子，每套卷子的难度系数为 a_i，同时有 m 个学生，每个学生的能力系数为 b_i，当试卷系数和能力系数满足以下关系时，小 F 才会让学生做这套卷子：

$$a_i + b_j \geq S$$

$$|\, a_i - b_j| \leq D$$

这里的 S 和 D 都是小 F 给出的平衡系数。

每套卷子最多只能给一个学生做，每个学生最多也只能做一套卷子（也可以不匹配）。小 F 想知道最多能成功匹配多少对（试卷, 学生）。

【输入格式】

从文件 ***match.in*** 中读入数据。

第一行包含四个正整数 n, m, S, D，分别表示试卷数量，学生数量，两个平衡系数。

第二行包括 n 个正整数 a_i 表示试卷的难度系数。

第三行包括 m 个正整数 b_i 表示学生的能力系数。

【输出格式】

输出到文件 ***match.out*** 中。

输出一行一个整数，表示最大的匹配数。

【样例 1 输入】

```
4 4 10 3
5 8 2 7
4 6 7 2
```

【样例 1 输出】

```
3
```

【样例 1 解释】

给定平衡参数 $S = 10, D = 3$。

试卷的难度系数为 $a = [5, 8, 2, 7]$，学生的能力系数为 $b = [4, 6, 7, 2]$。

一种可行的最大匹配方案如下（共匹配 3 对）：

- 将第 4 套试卷（难度 $a_4 = 7$）分配给第 1 个学生（能力 $b_1 = 4$）。
- 将第 1 套试卷（难度 $a_1 = 5$）分配给第 2 个学生（能力 $b_2 = 6$）。
- 将第 2 套试卷（难度 $a_2 = 8$）分配给第 3 个学生（能力 $b_3 = 7$）。

由于试卷 3（难度 2）和学生 4（能力 2）无法再与其他未匹配项组成满足条件的匹配，因此最多只能成功匹配 3 对，输出 3。

【样例 2】

见选手目录下的 ***ex_match / ex_match1.in*** 与 ***ex_match / ex_match1.ans***，其满足测试点 $1 \sim 2$。

【样例 3】

见选手目录下的 ***ex_match / ex_match2.in*** 与 ***ex_match / ex_match2.ans***，其满足测试点 $3 \sim 4$。

【数据范围】

对于 100% 的数据，保证 $1 \le n, m \le 2 \times 10^5, 1 \le a_i, b_i, S, D \le 10^9$。

测试点编号	$n, m \le$	特殊性质
$1 \sim 2$	10	无
$3 \sim 4$	1000	
$5 \sim 6$	2×10^5	A
$7 \sim 10$	2×10^5	无

对于特殊性质 A：满足 $D = 10^9$。

寻宝游戏（finale）

【题目背景】

这是一道交互题。

【题目描述】

小 F 太无聊了，他来到了一个综艺节目的现场，参加一个寻宝游戏。

游戏内容是这样的，宝藏被埋藏在形态和 n 个节点的树一样的道路，树以 1 为根，节目组在游戏开始前选择了 x 点，并在此埋了宝藏。

因此树的形态和宝藏埋藏地点在游戏开始前都是确定的。

你需要帮助小 F 赢得宝藏，他承诺和你平分。

在游戏开始前，节目组为小 F 准备了一张地图，换言之，他已经知道了整颗树的形态。为了防止小 F 漫无目的的找宝藏太过困难，节目组准备了一些提问次数，可以帮助小 F 找到宝藏。

具体地，小 F 可以向节目组提出以下两种询问：

- 给定节点 u，询问 u 与 x 的最短距离（简单路径上的边数）。
- 给定节点 u，询问 u 到 x 的简单路径上的第二个节点（即 u 之后的下一个节点）。
  - 特殊限制：小 F 只能在 u 为 x 的严格祖先的时，才能提出询问，否则直接判定为失败。

定义：我们宣称 a 是 b 的严格祖先，当且仅当 $a \neq b$ 且根节点 1 到 b 的简单路径经过节点 a。

你需要帮助小 F 在尽可能少的提问次数内找到宝藏，否则就算得到宝藏，也会被万恶的节目组抽走一部分。

【实现细节】

选手不需要，也不应该实现 `main` 函数。

选手需要确保提交的程序包含头文件 `finale.h`，即在程序开头加入以下代码：

```
#include "finale.h"
```

选手需要在提交的程序源文件 `finale.cpp` 中实现以下两个函数：

```
void init(int c, int t);
```

- c, t 分别表示测试点编号与测试数据组数。$c = 0$ 表示该测试点为样例。
- 对于每个测试点，该函数会在程序开始运行时被交互库调用恰好一次。

```
int find_node(int n, std::vector<int> u, std::vector<int> v);
```

- n 表示树的节点个数。

- u, v 是两个长度为 $n-1$ 的数组，分别表示树上的 $n-1$ 条无向边 (u_i, v_i)（下标从 0 开始，$0 \le i \le n-2$）。
- 该函数需要返回一个整数，表示你找到的隐藏节点 x 的编号。
- 对于每个测试点，该函数会被交互库调用**恰好 t 次**。

选手可以通过调用以下两个函数向交互库提出询问：

```
int query_d(int u);
```

- 参数 u 表示询问的节点编号。选手需要保证 $1 \le u \le n$。
- 该函数会返回节点 u 与隐藏节点 x 之间的最短距离。

```
int query_s(int u);
```

- 参数 u 表示询问的节点编号。选手需要保证 $1 \le u \le n$ 且 u 为 x 的严格祖先。
- 该函数会返回从 u 到 x 的简单路径上的第二个节点。

注意：在任何情况下，最终测试时所用的交互库运行所需时间均不会超过 0.2 秒，所用内存为固定大小，且均不超过 64 MiB。

【测试程序方式】

试题目录下的 `grader.cpp` 是交互库参考实现，最终测试时所用的交互库实现与该参考实现有所不同，因此选手的解法不应该依赖交互库实现。

选手可以在本题目录下使用如下命令编译得到可执行程序：

```
g++ grader.cpp finale.cpp -o finale -std=gnu++14 -O2 -static
```

【输入格式】

对于编译得到的可执行程序：

- 可执行文件将从**标准输入**读入以下格式的数据：
  - 输入的第一行包含两个非负整数 c, t，分别表示测试点编号和测试数据组数。
  - 接下来依次为每组测试数据，对于每组测试数据：
    - 第一行包含两个整数 n, x，分别表示树的节点数与隐藏节点的编号。
    - 接下来 $n-1$ 行，每行包含两个正整数 u, v，表示树上的一条无向边。

【输出格式】

- 可执行文件将输出以下格式的数据至**标准输出**：
  - 对于每组测试数据：
    - 第一行包含一个字符串，表示测试的结果。其中，
      - `Correct` 表示选手返回的结果正确；
      - `Wrong answer` 表示选手返回的结果错误；
      - `Invalid operation` 表示选手对询问函数的调用不合法（包括越界、违反祖先限制或询问次数超限）。
    - 若结果为 `Correct`，则第二行包含一个非负整数，表示选手在所有测试数据中调用 `query_d` 与 `query_s` 次数之和的最大值。

【样例 1 输入】

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

【样例 1 输出】

```
Correct.
2
```

【样例 1 解释】

该样例共包含一组测试数据。

对于第一组测试数据，树的形态如下，隐藏节点为 $x = 5$：

- 节点 1 与节点 2,3 相连；
- 节点 3 与节点 4,5 相连。

以下是一种可能的交互过程：

- 交互库调用 `find_node(5, [1, 1, 3, 3], [2, 3, 4, 5])`。
- 选手调用 `query_d(2)`，交互库返回 2 到 5 的距离 3。
- 选手调用 `query_s(3)`，由于 3 是 5 的严格祖先，交互库返回路径上的下一个节点 5。
- 选手返回节点编号 5。

选手程序共进行了 2 次询问，给出了正确答案。

【样例 2】

见选手目录下的 ***ex_finale / ex_finale1.in***，其满足测试点 $1 \sim 3$。

【样例 3】

见选手目录下的 ***ex_finale / ex_finale2.in***，其满足测试点 $10 \sim 12$。

【样例 4】

见选手目录下的 ***ex_finale / ex_finale3.in***，其满足测试点 $13 \sim 15$。

【下发文件说明】

在本试题目录下：

1. `grader.cpp` 是提供的交互库实现（与评测时保持一致）。
2. `finale.h` 是头文件，选手不用关心具体内容。
3. `template_finale.cpp` 是提供的示例代码，选手可参考并实现自己的代码。

选手注意对所有下发文件做好备份。最终评测时只测试本试题目录下的 `finale.cpp`，对该程序以外文件的修改不会影响评测结果。

【数据范围】

对于 100% 的数据，保证 $t = 10, 2 \le n \le 2 \times 10^5$，给定的边构成一棵以 1 为根的合法树。

测试点编号	$n \le$	特殊性质
$1 \sim 3$	20	无
$4 \sim 6$	2×10^5	A
$7 \sim 9$		B
$10 \sim 12$		C
$13 \sim 15$		D
$16 \sim 20$		无

- 特殊性质 A：树是一条以 1 为一端端点的链。
- 特殊性质 B：树是一棵以 1 为中心的菊花图（即所有节点 $2 \le i \le n$ 均与 1 直接连边）。
- 特殊性质 C：树是一棵深度不超过 18 的满二叉树。
- 特殊性质 D：树按照均匀随机方式生成。

【评分方式】

注意：

- 选手不应当通过非法方式获取交互库的内部信息，如直接与标准输入、输出流进行交互。此类行为将被视为作弊；
- **最终的评测交互库与样例交互库的实现相同。**

本题首先会受到和传统题相同的限制，例如编译错误会导致整道题目得 0 分，运行时错误、超过时间限制、超过空间限制等会导致相应测试点得 0 分等。选手只能在程序中访问自己定义的变量以及交互库给出的变量，尝试访问其他地址空间将可能导致编译错误或运行错误。

每次调用 `find_node` 函数时，若返回的节点不正确，或询问函数调用不合法，或单组测试数据内调用询问函数的总次数超过 2×10^5，则相应测试点得 0 分。

在上述条件基础上：

- 在测试点 $1 \sim 3$ 中，程序得到满分当且仅当每次调用 `find_node` 函数时，询问总次数不超过 10；否则获得 0 分。

- 在测试点 $4 \sim 20$ 中，设 m 表示某测试点各组数据中调用询问函数次数总和的最大值，则程序在该测试点将获得 $5 \cdot f(m)$ 分，其中 $f(m)$ 的计算方式如下表所示：

m	$f(m) =$
$m \le 36$	1
$36 < m \le 72$	$1 - \frac{\sqrt{m-36}}{30}$
$72 < m \le 500$	$0.8 - \frac{(m-72)^{1.5}}{16000}$
$500 < m \le 2 \times 10^5$	$0.2 - 0.18 \times \frac{m-500}{2\times 10^5-500}$
$m > 2 \times 10^5$	0

防御阵列（station）

【题目背景】

小 F 穿越到了一个平行世界，这里的人们所用的装备和我们不一样，小 F 发现这个世界的军事装备是一种名叫魂导器的东西，具体是什么他也说不清楚，需要充能释放，而且似乎是有些限制的。

小 F 所在的地方正处于战场，他被一方军队的人抓住了，小 F 为了使得自己活下来，说明了自己是一个有用的人，他可以帮助军队解决实际问题。

于是元帅告诉了他一个问题，现在全军列装了 n 个联动魂导防御护罩，但是由于魂导器产生的能量波动会影响旁边的魂导器，现在小 F 需要找出使得魂导器两两之间最大的最短距离。

他如果不解决掉就会被杀掉，请你帮助小 F 解决这个问题。

【题目描述】

给定 n 个魂导器的候选坐标 a_i, b_i，换言之，第 i 个魂导器只能布置在 a_i 或 b_i 的其中一个位置上，即 $p_i \in \{a_i, b_i\}$。

定义抗干扰度为：

$$\min_{1 \le i < j \le n} |p_i - p_j|$$

请你合理安排每个魂导器的部署位置，使得两两之间的**最小距离最大化**。

【输入格式】

从文件 ***station.in*** 中读入数据。

第一行包含一个整数 n，表示魂导器的数量。

接下来 n 行，每行两个非负整数 a_i, b_i 表示第 i 个魂导器的候选位置。

【输出格式】

输出到文件 ***station.out*** 中。

输出一个整数，表示最小距离最大的答案。

【样例 1 输入】

```
3
1 8
3 12
6 10
```

【样例 1 输出】

```
5
```

【样例 1 解释】

一种最优的部署方案为：

- 第 1 个魂导器选择坐标 $a_1 = 1$；
- 第 2 个魂导器选择坐标 $b_2 = 12$；
- 第 3 个魂导器选择坐标 $a_3 = 6$。

此时最终部署坐标为 $\{1, 6, 12\}$，各对魂导器之间的距离分别为：

- $|1 - 6| = 5$；
- $|6 - 12| = 6$；
- $|1 - 12| = 11$。

两两距离的最小值为 $\min(5, 6, 11) = 5$。可以证明，不存在使得两两最小距离严格大于 5 的部署方案。

【样例 2】

见选手目录下的 ***ex_station / ex_station1.in***，其满足测试点 $3 \sim 6$。

【样例 3】

见选手目录下的 ***ex_station / ex_station2.in***，其满足测试点 $9 \sim 20$。

【数据范围】

对于 100% 的数据，保证 $2 \le n \le 5 \times 10^4$，$0 \le a_i, b_i \le 10^9$，对于任意 $1 \le i \le n$，保证 $a_i \ne b_i$。

测试点编号	$n \le$	特殊性质
$1 \sim 2$	16	无
$3 \sim 6$	1000	
$7 \sim 8$	5×10^4	A
$9 \sim 20$		无

- 特殊性质 A：存在常数 L，使得所有 $b_i = a_i + L$。

通讯网络（telecom）

【题目背景】

小 F 正在帮助帝国构建通讯网络，可是由于帝国的地理位置原因，帝国一共有 n 个城市，但是只有 $n-1$ 条边。

帝国的首都在 1 号节点，现在为了保证通讯的高效性，小 F 决定将帝国的信息传输变为若干条链（自上而下的链）的传输，他认为这样可以提高效率。

可是每一条传输链又有不同的消耗和花费，他一个人解决不了，于是小 F 决定求助于你，帮他求出最小的维护花费。

【题目描述】

帝国是一个 n 个节点的有根树，节点编号分别是 $1 \sim n$，帝国首都 1 号节点即为根。

除根节点 1 之外，每一个节点 $i(2 \le i \le n)$ 均有父节点 $p_i(1 \le p_i < i)$，且这条道路的长度为 d_i。

你需要将树上的 n 个节点划分为若干条自上而下的链。

定义：我们称自上而下的链为一个节点序列 $X = (x_1, x_2, \cdots, x_k)(k \ge 1)$ 满足：对于任意 $2 \le j \le k$，均有 $p_{x_j} = x_{j-1}$。

维护一条传输链需要支付一定的费用，这个费用由**基础费用**和**距离费用**两部分组成，具体来说如下：

$$W(X) = C + \left(\sum_{j=2}^{k} d_{x_j} \right)^2$$

特别的，当 $k = 1$ 的时候，$W(X) = C$。

并且你需要保证每一个节点恰好被划分进一个链内，换言之，如果划分为了 m 条链 $X^{(1)}, X^{(2)}, \cdots, X^{(m)}$，则每个节点必须只在一条链内出现一次。

小 F 希望你求出所有链维护费用之和的最小值，即 $\min \sum_{i=1}^{m} W(X^{(i)})$。

【输入格式】

从文件 ***telecom.in*** 中读入数据。

第一行包含两个正整数 n, C，分别表示节点数和基础费用。

接下来 $n-1$ 行，第 i 行（对应编号为 $i+1$ 的节点）包含两个正整数 p_{i+1}, d_{i+1}，分别表示 $i+1$ 节点的父节点和向父节点方向道路的长度。

【输出格式】

输出到文件 ***telecom.out*** 中。

一行一个整数，表示划分所有节点的最小总维护费用。

【样例 1 输入】

```
6 7
1 2
1 1
3 2
3 1
5 1
```

【样例 1 输出】

```
29
```

【样例 1 解释】

一种最优的划分方案是将 6 个节点划分为如下 3 条信息传输链：

- 链 1 为 $(1,2)$，费用为 $C+(d_2)^2=7+2^2=11$
- 链 2 为 $(3,5,6)$，费用为 $C+(d_5+d_6)^2=7+(1+1)^2=11$
- 链 3 为 (4)，费用为 $C+0^2=7$

所有节点均恰好属于一条链，总维护费用为 $11+11+7=29$。可以证明不存在总费用更小的划分方案。

【样例 2】

见选手目录下的 ***ex_telecom / ex_telecom1.in*** 与 ***ex_telecom / ex_telecom1.ans***，其满足测试点 1。

【样例 3】

见选手目录下的 ***ex_telecom / ex_telecom2.in*** 与 ***ex_telecom / ex_telecom2.ans***，其满足测试点 $2\sim3$。

【数据范围】

对于 100% 的数据，保证 $1\le n\le 10^5$，$1\le d_i\le 10^4$，$1\le p_i<i$，$1\le C\le 10^{12}$。

测试点编号	$n \leq$	$C \leq$	特殊性质
1	15	10^6	无
$2 \sim 3$	5000	10^{12}	
$4 \sim 5$	10^5	10^{12}	A
$6 \sim 7$	10^5	10^{12}	B
$8 \sim 25$	10^5	10^{12}	无

- 特殊性质 A：整棵树中最多只有一个节点的度数大于 1。
- 特殊性质 B：整棵树退化为一条链，且根节点 1 为链的一个端点。