

## 即时战略 (rts)

这是一道交互题。

### 【题目描述】

小 M 在玩一个即时战略 (Real Time Strategy) 游戏。不同于大多数同类游戏，这个游戏的地图是树形的。也就是说，地图可以用一个由  $n$  个结点， $n - 1$  条边构成的连通图来表示。这些结点被编号为  $1 \sim n$ 。

每个结点有两种可能的状态：“已知的”或“未知的”。游戏开始时，只有 1 号结点是已知的。在游戏中，小 M 可以尝试探索更多的结点。具体来说，小 M 每次操作时需要选择一个已知的结点  $x$ ，和一个不同于  $x$  的任意结点  $y$ （结点  $y$  可以是未知的）。然后游戏的自动寻路系统会给出  $x$  到  $y$  的最短路径上的第二个结点  $z$ ，也就是从  $x$  走到  $y$  的最短路径上与  $x$  相邻的结点。此时，如果结点  $z$  是未知的，小 M 会将它标记为已知的。

这个游戏的目标是：利用至多  $T$  次探索操作，让所有结点的状态都成为已知的。然而小 M 还是这个游戏的新手，她希望得到你的帮助。

为了让游戏过程更加容易，小 M 给你提供了这个游戏的交互库，具体见【任务描述】和【实现细节】。

另外，小 M 也提供了一些游戏的提示，具体见题目的最后一节【提示】。

### 【任务介绍】

你需要实现一个函数 `play`，以帮助小 M 完成游戏的目标。

- `play(n, T, dataType)`
- $n$  为树的结点个数；
- $T$  为探索操作的次数限制；
- `dataType` 为该测试点的数据类型，具体见【数据规模和约定】。

在每个测试点中，交互库都会调用恰好一次 `play` 函数。该函数被调用之前，游戏处于刚开始的状态。

你可以调用函数 `explore` 来帮助你在游戏中探索更多结点，但是这个函数的调用次数不能超过  $T$  次。

- `explore(x, y)`
- $x$  为一个已知的结点；
- $y$  为一个不同于  $x$  的任意结点（可以不是已知的结点）；
- 这个函数会返回结点  $x$  到  $y$  的最短路径上的第二个结点的编号。

在函数 `play` 返回之后，交互库会检查游戏的状态：只有当每个结点都是已知的，才算游戏的目标完成。

## 【实现方法】

你需要且只能提交一个源文件 `rts.cpp/c/pas` 实现上述函数，且遵循下面的命名和接口。

### 对 C/C++ 语言的选手：

源代码中需要包含头文件 `rts.h`。

你需要实现的函数 `play`：

```
[] void play(int n, int T, int dataType);
```

函数 `explore` 的接口信息如下：

```
[] int explore(int x, int y);
```

### 对 Pascal 语言的选手：

你需要使用单元 `graderhelperlib`。

你需要实现的函数 `play`：

```
[] procedure play(n, T, dataType : longint);
```

函数 `explore` 的接口信息如下：

```
[] function explore(x, y : longint) : longint;
```

## 【如何测试你的程序】

### 对 C/C++ 语言的选手：

你需要在本题目录下使用如下命令编译得到可执行程序：

对于 C 语言：

```
gcc grader.c rts.c -o rts -O2 -lm
```

对于 C++ 语言：

```
g++ grader.cpp rts.cpp -o rts -O2 -lm
```

### 对 Pascal 语言的选手：

你需要在本题目录下使用如下命令编译得到可执行程序：

```
fpc grader.pas -o"rts" -O2
```

可执行文件将从标准输入读入以下格式的数据：

第一行包含三个整数  $n$ ,  $T$ ,  $dataType$ ，需要保证  $n$  在  $[2, 3 \times 10^5]$  之间， $T$  在  $[1, 5 \times 10^6]$  之间， $dataType$  在  $[1, 3]$  之间。

接下来  $n - 1$  行，每行两个整数  $u, v$ ，需要保证  $1 \leq u, v \leq n$  且  $u \neq v$ ，表示一条  $u$  和  $v$  之间的边。

你的输入需要保证这  $n - 1$  条边构成一棵树。

读入完成之后，交互库将调用 `play` 函数。如果此时你调用 `explore` 的次数超过  $T$  次，则交互库会输出详细的错误信息，并退出。

接下来交互库会判断游戏目标是否完成。如果完成，则会输出 "Correct"，否则会输出相应的错误信息。

如果传入 `explore` 函数的参数非法 ( $x, y$  不在 1 到  $n$  的范围内, 或  $x$  不是已知结点, 或  $x$  等于  $y$ ), 那么交互库会输出详细的错误信息, 并退出。

如果要使用自己的输入文件进行测试, 请保证输入文件符合以上格式要求, 否则不保证程序能正确运行。

### 【如何使用样例源代码】

本题目录下, 有针对每种语言的样例源代码 `rts_sample.cpp/c/pas`。选择你所需的语言, 将其复制为 `rts.cpp/c/pas`, 按照上文中提到的方式进行编译, 即能通过编译得到可执行程序。

对于非正式选手, 你只能选择一种语言进行作答, 即你本题的试题目录下不能同时存在多个语言的 `rts.cpp/c/pas`, 否则系统将任选一份源代码进行评测并作为最终结果。

接下来你需要修改这个文件的实现, 以达到题目的要求。

### 【样例 1 输入】

```
4 100 1
1 3
3 4
3 2
```

### 【样例 1 输出】

```
Correct
```

### 【样例 1 解释】

这是使用试题目录的 `grader` 和正确的源程序得到的输出文件。

对于此样例, 一种可能的 `explore` 函数的调用顺序为:

- `explore(1, 2)`, 返回 3
- `explore(3, 2)`, 返回 2
- `explore(2, 4)`, 返回 3
- `explore(3, 4)`, 返回 4

### 【样例 2】

见选手目录下的 `rts/rts2.in` 与 `rts/rts2.ans`。

该组样例的数据范围同第 5 个测试点。

### 【样例 3】

见选手目录下的 *rts/rts3.in* 与 *rts/rts3.ans*。

该组样例的数据范围同第 8 个测试点。

### 【评分方式】

最终评测时只会收取 *rts.cpp/c/pas*，修改选手目录中的其他文件对评测无效。

题目首先会受到和非交互式程序题相同的限制。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。你只能访问自己定义的和交互库给出的变量及其对应的内存空间，尝试访问其他空间将可能导致编译错误或运行错误。

若程序正常结束，则会开始检验正确性。只有当游戏目标完成时，该测试点得满分，其他情况该测试点得 0 分。

题目中所给的时间、空间限制为你的代码和交互库加起来可以使用的时间和空间。我们保证，对于任何合法的数据及在限制范围内的调用，任何语言任何版本的交互库（包括下发给选手的和最终评测使用的），运行所用的时间不会超过 0.5s，运行所用的空间不会超过 64MB，也就是说，选手实际可用的时间至少为 1.5s，实际可用的空间至少为 448MB。

### 【子任务】

一共有 20 个测试点，每个测试点 5 分。

对于所有测试点，以及对于所有样例， $2 \leq n \leq 3 \times 10^5$ ,  $1 \leq T \leq 5 \times 10^6$ ,  $1 \leq \underline{dataType} \leq 3$ 。不同 dataType 对应的数据类型如下：

- 对于 dataType = 1 的测试点，没有特殊限制
- 对于 dataType = 2 的测试点，游戏的地图是一棵以结点 1 为根的完全二叉树，即，存在一个  $1 \sim n$  的排列  $a$ ，满足  $a_1 = 1$ ，且结点  $a_i$  ( $1 < i \leq n$ ) 与结点  $a_{\lfloor i/2 \rfloor}$  之间有一条边相连
- 对于 dataType = 3 的测试点，游戏的地图是一条链，即，存在一个  $1 \sim n$  的排列  $a$ ，满足结点  $a_i$  ( $1 < i \leq n$ ) 与结点  $a_{i-1}$  之间有一条边相连

对于每个测试点， $n, T, \underline{dataType}$  的取值如下表：

测试点编号	$n =$	$T =$	<u><math>dataType =</math></u>
1	2	10000	1
2	3	10000	
3	10	10000	
4	100	10000	
5	1000	10000	2
6	20000	300000	
7	250000	5000000	
8	1000	20000	3
9	5000	15500	
10	30000	63000	
11	150000	165000	
12	250000	250100	
13	300000	300020	
14	1000	50000	1
15	5000	200000	
16	30000	900000	
17	150000	3750000	
18	200000	4400000	
19	250000	5000000	
20	300000	5000000	

### 【提示】

这里是小 M 给你的一些贴心的提示：

- 图（无向图）由结点和边构成，边是结点的无序对，用来描述结点之间的相互关系
- 路径是一个结点的非空序列，使得序列中相邻两个结点之间都有边相连
- 两个结点是连通的，当且仅当存在一条以其中一个结点开始、另一个结点结束的路径
- 一个图是连通的，当且仅当这个图上的每对结点都是连通的
- 一棵  $n$  个结点的树，是一个由  $n$  个结点， $n - 1$  条边构成的连通图
- 两个结点的最短路径，是指连接两个结点的所有可能的路径中，序列长度最小的
- 在一棵树中，连接任意两个结点的最短路径，都是唯一的
- 通过访问输入输出文件、攻击评测系统或攻击评测库等方式得分属于作弊行为，所得分数无效。