CCF NOI 2018 冬令营

IOI 2018 中国国家候选队选拔赛

正式赛

时间：2018 年 2 月 8 日 08:00 ~ 13:00

题目名称	通道	州区划分	即时战略
题目类型	传统型	传统型	交互型
目录	`tunnel`	`walk`	`rts`
可执行文件名	`tunnel`	`walk`	`rts`
输入文件名	`tunnel.in`	`walk.in`	N/A
输出文件名	`tunnel.out`	`walk.out`	N/A
每个测试点时限	4.0 秒	10.0 秒	2.0 秒
内存限制	2 GB	1 GB	512 MB
测试点数目	31	20	20
每个测试点分值	见题面	5	5

提交源程序文件名

对于 C++ 语言	`tunnel.cpp`	`walk.cpp`	`rts.cpp`
对于 C 语言	`tunnel.c`	`walk.c`	`rts.c`
对于 Pascal 语言	`tunnel.pas`	`walk.pas`	`rts.pas`

编译选项

对于 C++ 语言	`-O2 -lm -std=c++11`	`-O2 -lm`	`-O2 -lm`
对于 C 语言	`-O2 -lm -std=c11`	`-O2 -lm`	`-O2 -lm`
对于 Pascal 语言	`-O2`	`-O2`	`-O2`

通道（tunnel）

【题目描述】

11328 年，C 国的科学家们研发了一种高速传送通道，可以在很短的时间内把居民从通道的一端送往另一端，这些通道都是双向的。

美中不足的是，这种传送通道需要进行大量的维护和检修。经过规划，C 国总统决定在 M 城中新建这种通道，在 M 城中，建立了 n 个传送站和 $3\times(n-1)$ 条传送通道，这些传送通道被分为 3 组，每一组都包含了 $(n-1)$ 条通道。

当任意一组通道运行时，居民都可以通过这组通道从任意一个传送站前往任意的另一个传送站。也就是说，所有的传送站都会被通道所连通。

三组通道按照 1、2、3 的顺序轮流运行，循环反复。在任意一个时刻，都有且只有一组传送通道可以使用。形式化地，在第 i 天中，有且只有第 $((i-1) \bmod 3+1)$ 组通道运行。

C 国著名科学家 Access Globe 正在进行一项社会调查实验：调查两个传送站之间的传送通道使用者的信息。Access Globe 的计划是这样的：

- 选定两个传送站 a、b
- 第一天，他从 a 出发，使用正在运行的这组通道沿最短路径到达 b，并调查经过的所有通道上使用者的信息
- 第二天，他从 b 出发，使用正在运行的这组通道沿最短路径到达 a，并调查经过的所有通道上使用者的信息
- 第三天，他从 a 出发，使用正在运行的这组通道沿最短路径到达 b，并调查经过的所有通道上使用者的信息

Access Globe 知道每一条传输线路在运行时的使用者人数。他希望找出一对 a、b，使得在整个实验过程中所有经过的通道的使用者数量之和最大。Access Globe 希望参加 CCF NOI 2018 冬令营的你帮他解决这个简单的小问题。如果你成功地解决了这个问题，Access Globe 会送你一份小礼物——100 分！

【输入格式】

从文件 ***tunnel.in*** 中读入数据。

输入文件的第 1 行包含一个正整数 n，表示传送站的个数，传送站从 1 到 n 编号；

输入文件的第 2 到第 n 行，每行包含 3 个数 u,v,w，表示第一组通道中有一条连接 u,v 的通道，其运行时使用者数量为 w 人；

输入文件的第 $(n+1)$ 到第 $(2n-1)$ 行，每行包含 3 个数 u,v,w，表示第二组通道中有一条连接 u,v 的通道，其运行时使用者数量为 w 人；

输入文件的第 $2n$ 到第 $(3n-2)$ 行，每行包含 3 个数 u,v,w，表示第三组通道中有一条连接 u,v 的通道，其运行时使用者数量为 w 人。

【输出格式】

输出到文件 ***tunnel.out*** 中。
输出文件共 1 行，包含一个整数，表示最大的使用者数量之和。

【样例 1 输入】

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

【样例 1 输出】

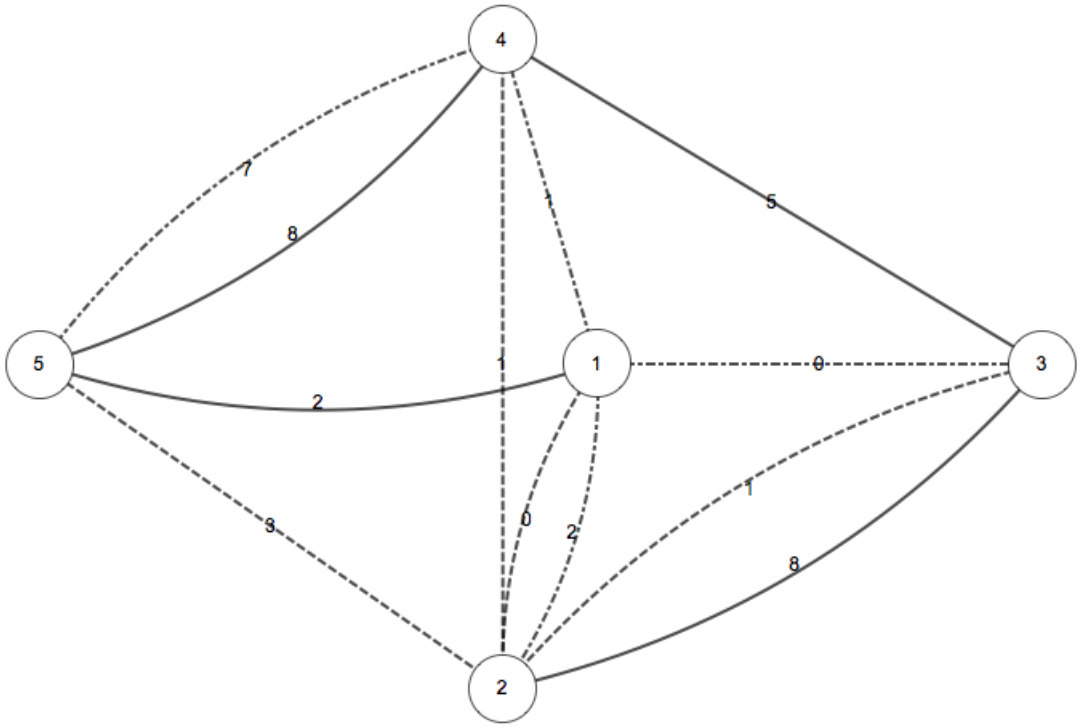
```
27
```

【样例 1】

见选手目录下的 ***tunnel/tunnel1.in*** 与 ***tunnel/tunnel1.ans***。

【样例 1 解释】

下图为样例中 M 城的传送站和传输线路情况。其中点和虚线交替的线条、虚线条和实线条分别表示第一组、第二组和第三组通道。



一种可行的方案是选择 $a=2, b=5$，这样的使用者数量之和为 $(3)+(8+5+1)+(2+1+7)=27$。

【样例 2】

见选手目录下的 ***tunnel/tunnel2.in*** 与 ***tunnel/tunnel2.ans***。

【评分方式】

本题采用**子任务捆绑测试**。一个**子任务**可能包含多个**测试点**，只有通过一个子任务中的**全部测试点**，才能使该子任务得满分，否则该子任务得 0 分。

【子任务】

对于所有数据，$2 \le n \le 10^5, 0 \le w \le 10^{12}$。

特殊性质 0：任意两组通道构成完全相同。

特殊性质 1：第二组通道和第三组通道构成完全相同。

特殊性质 2：对于第二组的每一个传送站，最多只有两个通道可以到达它，且编号为 x, y 的传送站之间通过**一条**通道直接连接充要条件是 $|x-y|=1$。

特殊性质 3：对于第三组的每一个传送站，最多只有两个通道可以到达它。

特殊性质 4：$n \le 3000$。

子任务	分值	特殊性质
0	6	4
1	3	0, 1, 2, 3
2	10	0, 1
3	16	1, 2, 3
4	11	1
5	15	2, 3
6	13	3
7	26	无

【提示】

- 在两组通道中，可能都包含了连接传送站 x, y 的通道，此时我们认为这两条通道是**不同**的。
- 特殊性质中，A 组通道和 B 组通道的“构成完全相同”是指：如果在 A 组中 u, v 之间存在一条使用人数为 w 的通道，那么在 B 组中 u, v 之间一定也存在一条使用人数为 w 的通道。是否相同**与描述方式与描述顺序均无关**。即在构成完全相同的两组通道 A 和 B 中，**通道输入的顺序不一定相同，每条通道的端点的输入顺序也不一定相同**（对于 A、B 组中一条连接 u, v 的使用人数为 w 的通道，一种可能出现的输入为：A 组通道中输入 $u\ v\ w$，而 B 组通道中输入 $v\ u\ w$）。

州区划分（walk）

【题目背景】

小 S 现在拥有 n 座城市，第 i 座城市的人口为 w_i，城市与城市之间可能有双向道路相连。

现在小 S 要将这 n 座城市划分成若干个州，每个州由至少一个城市组成，每个城市在恰好一个州内。

假设小 S 将这些城市划分成了 k 个州，设 V_i 是第 i 个州包含的所有城市组成的集合。定义一条道路是一个州的内部道路，当且仅当这条道路的两个端点城市都在这个州内。如果一个州内部存在一条起点终点相同，不经过任何不属于这个州的城市，且经过这个州的所有内部道路都恰好一次的路径（路径长度可以为 0），则称这个州是不合法的。

定义第 i 个州的满意度为：第 i 个州的人口在前 i 个州的人口中所占比例的 p 次幂，即：

$$\left(\frac{\sum_{x\in V_i} w_x}{\sum_{j=1}^{i}\sum_{x\in V_j} w_x}\right)^p$$

定义一个划分的满意度为所有州的满意度的乘积。

求所有合法的划分方案的满意度之和。

答案对 998244353 取模。

两个划分 $\{V_1...V_k\}$ 和 $\{C_1...C_s\}$ 是不同的，当且仅当 $k \neq s$，或存在某个 $1 \leq i \leq k$，使得 $V_i \neq C_i$。

【输入格式】

从文件 ***walk.in*** 中读入数据。

输入第一行包含三个整数 n, m, p，表示城市个数、城市之间的道路个数以及题目描述中的常数 p。

接下来 m 行，每行两个正整数 u, v，描述一条无向的道路，保证无重边无自环。

输入第 $m+2$ 行有 n 个正整数，第 i 个正整数表示 w_i。

【输出格式】

输出到文件 ***walk.out*** 中。

输出一行一个整数表示答案在模 998244353 意义下的取值。

即设答案化为最简分式后的形式为 $\frac{a}{b}$，其中 a 和 b 的互质。输出整数 x 使得 $bx \equiv a \bmod 998244353$ 且 $0 \leq x < 998244353$。可以证明这样的整数 x 是唯一的。

【样例 1 输入】

```
3 2 1
1 2
2 3
1 1 1
```

【样例 1 输出】

```
1
```

【样例 2】

见选手目录下的 ***walk/walk2.in*** 与 ***walk/walk2.ans***。

【提示】

$x^{p-1} \equiv 1 \mod p$，其中 p 为质数，$x \in [1, p)$。

【评分方式】

本题采用**子任务捆绑测试**。一个**子任务**可能包含多个**测试点**，只有通过一个子任务中的**全部测试点**，才能使该子任务得满分，否则该子任务得 0 分。

【子任务】

保证对于所有数据有：$0 \le n \le 21$，$0 \le m \le \frac{n*(n-1)}{2}$，$0 \le p \le 2$，$1 \le w_i \le 100$。

测试点	n	p
1	5	2
2	10	
3	15	0
4		1
5		2
6	21	0
7		
8		

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。在游戏的过程中，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 $1 \le u, v \le n$ 且 $u \ne v$，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，$2 \le n \le 3\times10^5, 1 \le T \le 5\times10^6, 1 \le \underline{dataType} \le 3$。不同 $\underline{dataType}$ 对应的数据类型如下：

- 对于 $\underline{dataType} = 1$ 的测试点，没有特殊限制
- 对于 $\underline{dataType} = 2$ 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 $(1 < i \le n)$ 与结点 $a_{\lfloor i/2 \rfloor}$ 之间有一条边相连
- 对于 $\underline{dataType} = 3$ 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 $(1 < i \le n)$ 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	$n=$	$T=$	$\underline{dataType}=$
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。