ダンジョン 2 (Dungeon 2)

あなたは Just Ordinary Inventions 社を知っているだろうか? この会社の業務は「ただ平凡な発明 (just ordinary inventions)」をすることである.

JOI 君は, Just Ordinary Inventions 社で開発された最新のゲームで遊んでいる.

このゲームは, いくつかの部屋といくつかの道からなるダンジョンを探索するゲームである. 道はダンジョン内の異なる 2 つの部屋を結んでおり, 双方向に移動可能である. どの異なる 2 つの部屋についても, それらを結ぶ道は高々1 本しかなく, 両端が同じ部屋であるような道は存在しない. また, ダンジョンのどの 2 つの部屋の間も, 何本かの道を使うことで互いに移動できることがわかっている. 各部屋同士は非常によく似ており, 同じ本数の道が出ている部屋同士は, 部屋の様子を見るだけではまったく区別ができない.

このゲームでは, 攻略を助けるために各部屋に目印と台座が用意されている. 部屋から出ている道は, 目印を基準として 1 本目, 2 本目, ... と数えることができる. ゲーム中にダンジョンの構造が変化することはない. したがって, 同じ部屋から同じ番号の道を使って移動すると, いつも同じ部屋にたどり着く. 台座にはプレイヤーが色を変更することができる宝石が 1 個飾られている. 宝石の色は色 1, 色 2, ..., 色 X のいずれかであり, ゲーム開始時の各部屋の宝石の色は色 1 である. 宝石の色はプレイヤーが操作しない限り変更されることはない.

JOI 君は, もしこのダンジョンの構造, すなわちダンジョンの部屋同士がどのように道で結ばれているかがわかれば, このゲームは簡単に攻略できてしまうことに気づいた. しかし, JOI 君がいろいろ試してもダンジョンの構造を決定することはできなかった. そこで, あなたは JOI 君に代わってダンジョンの構造を決定するプログラムを書くことにした.

課題

ダンジョンを探索して, ダンジョンの構造を決定するプログラムを作成せよ. しかし, JOI 君はダンジョンの構造を完全に明かされることを望まなかったため, プログラムはダンジョンの構造を直接答えるのではなく, 1 以上 R 以下の各整数 i に対して「最小でちょうど i 本の道を使うことで移動できる 2 つの部屋の組は何個あるか」の値 (部屋の順序を入れ替えたものは同じ組とみなす) を解答しなければならない.

ダンジョンを探索するために, あなたには次の行動を行うためのライブラリが提供される.

- 現在いる部屋から, 何本の道が出ているかを知る.
- 現在いる部屋の台座に飾られている宝石の色を知る.
- 現在いる部屋の台座に飾られている宝石の色を, 指定した色にする (現在の色と同じ色を指定することもできる). その後, 部屋から出ている道を 1 本選び, その道を使って他の部屋へ移動する.
- 最後に使った道が, 現在いる部屋から出ている道のうち何本目かを知る.

実装の詳細

あなたは，JOI 君に解答する方法を実装した 1 個のプログラムを書かねばならない．プログラムは dungeon2.h をインクルードすること．

プログラムは，以下のルーチンを実装しなければならない．

- `void Inspect(int R)`

  このルーチンは，最初に 1 回だけ呼び出される．

  - 引数 R は，1 以上 R 以下の各整数 i に対して「最小でちょうど i 本の道を使うことで移動できる 2 つの部屋の組は何個あるか」の値 (部屋の順序を入れ替えたものは同じ組とみなす) を解答しなければならないことを表す．

また，次のルーチンを呼び出し，質問に答えなければならない．

- `void Answer(int D, int A)`

  - 引数 D，A は，この呼び出しにおいて,「最小でちょうど D 本の道を使うことで移動できる 2 つの部屋の組は A 個ある」ことを解答することを表す．

  ただし，Answer の呼び出しは以下の条件を満たすこと．

  - D は 1 以上 R 以下の整数でなければならない．これを満たさない場合，不正解 **[1]** となる．
  - Answer を同じ引数 D で 2 回以上呼び出してはならない．これを満たさない場合，不正解 **[2]** となる．
  - Answer はちょうど R 回呼び出さなければならない．これを満たさない場合，不正解 **[3]** となる．
  - A は最小でちょうど D 本の道を使うことで移動できる 2 つの部屋の組の個数でなければならない．これを満たさない場合，不正解 **[4]** となる．

  Answer の呼び出しが不正解と判定された場合，それ以降のプログラムは実行されるとは限らない．

これに加え，プログラム中では，以下のルーチンを呼び出すことができる．

- `void Move(int I, int C)`

  - 引数 I は，プレイヤーが移動するために選ぶ道の番号である．この呼び出しの直後，プレイヤーは現在いる部屋から出ている I 本目の道を使って他の部屋まで移動する．
  - 引数 C は，部屋の移動を行う前に，現在いる部屋の台座に飾られている宝石の色を色 C にすることを表す．

  ただし，Move の呼び出しは以下の条件を満たすこと．

  - 引数 I は，プレイヤーが現在いる部屋から出ている道の本数を K として，1 以上 K 以下の整数でなければならない．これを満たさない場合，不正解 **[5]** となる．

- 引数 C は，宝石の色の種類が X 種類として，1 以上 X 以下の整数でなければならない．X の値は小課題ごとに定まっている．これを満たさない場合，不正解 [6] となる．
- Move を 1 500 000 回より多く呼び出してはならない．呼び出した場合，不正解 [7] となる．

- `int NumberOfRoads()`
  - この関数は，プレイヤーが現在いる部屋から出ている道の本数を返す．
- `int LastRoad()`
  - この関数は，プレイヤーが最後に使った道が，現在いる部屋から出ている道のうち何本目の道であるかを返す．ただし，Move を一度も呼び出す前にこの関数が呼び出された場合は，−1 を返す．
- `int Color()`
  - この関数は，プレイヤーが現在いる部屋の台座に飾られている宝石の色を返す．

ルーチン Inspect が呼び出された後，答えの判定が行われる．

内部での使用のために他のルーチンを実装したり，グローバル変数を宣言するのは自由である．ただし，あなたの提出は標準入力・標準出力，あるいは他のファイルといかなる方法でもやりとりしてはならない．

コンパイル・実行の方法

作成したプログラムをテストするための，採点プログラムのサンプルが，コンテストサイトからダウンロードできるアーカイブの中に含まれている．このアーカイブには，提出しなければならないファイルのサンプルも含まれている．

採点プログラムのサンプルは 1 つのファイルからなる．そのファイルは grader.c または grader.cpp である．作成したプログラムをテストするには， 次のようにコマンドを実行する．

- C の場合

```
gcc -std=c11 -O2 -o grader grader.c dungeon2.c -lm
```

- C++ の場合

```
g++ -std=c++11 -O2 -o grader grader.cpp dungeon2.cpp
```

コンパイルが成功すれば， grader という実行ファイルが生成される．

実際の採点プログラムは，採点プログラムのサンプルとは異なることに注意すること．採点プログラムのサンプルは単一のプロセスとして起動する．このプログラムは，標準入力から入力を読み込み，標準出力に結果を出力する．

採点プログラムのサンプルの入力

採点プログラムのサンプルは標準入力から以下のデータを読み込む．

- 1 行目には，整数 N, X, R が空白を区切りとして書かれている．これは，ダンジョンに部屋 1，部屋 2，…，部屋 N の N 個の部屋があり，宝石の色が X 種類あり，プログラムが解答すべき値が R 個であることを表す．
- 続く $2N$ 行のうちの $2i-1$ 行目 $(1 \leqq i \leqq N)$ には，整数 D_i が書かれており，部屋 i から D_i 本の道が出ていることを表す．$2i$ 行目 $(1 \leqq i \leqq N)$ には，D_i 個の整数 $T_{i1}, T_{i2}, \ldots, T_{iD_i}$ が空白を区切りとして書かれている．これは，部屋 i から出ている j $(1 \leqq j \leqq D_i)$ 本目の道を使って移動すると部屋 T_{ij} にたどり着くことを表す．
- 続く R 行のうちの j 行目 $(1 \leqq j \leqq R)$ には，整数 A_j が書かれている．これは，最小でちょうど j 本の道を使うことで移動できる 2 つの部屋の組が A_j 個あることを表す．つまり，各 j $(1 \leqq j \leqq R)$ に対して，`Answer` の引数 `D` として j，引数 `A` として A_j を与えて呼び出した場合に採点プログラムのサンプルが正解と判定し，それ以外の場合に不正解と判定することを表す．

採点プログラムのサンプルは，プレイヤーの初期位置を部屋 1 として，あなたの作成したルーチンを呼び出す．

採点プログラムのサンプルの出力

プログラムの実行が正常に終了した場合，採点プログラムのサンプルは標準出力へ以下の情報を 1 行で出力する (引用符は実際には出力されない)．

- 正解の場合，関数 `Move` を呼び出した回数が “`Accepted : #move = 8`” のように出力される．
- 不正解の場合，不正解の種類が “`Wrong Answer [1]`” のように出力される．

制限

すべての入力データは以下の条件を満たす．N, D_i, T_{ij} の意味については「採点プログラムのサンプルの入力」の項目を参照せよ．

- $2 \leqq N \leqq 200$.
- $3 \leqq X \leqq 100$.
- $1 \leqq R \leqq 200$.
- $1 \leqq D_i \leqq N-1$ $(1 \leqq i \leqq N)$.
- $1 \leqq T_{ij} \leqq N$ かつ $T_{ij} \neq i$ $(1 \leqq i \leqq N, 1 \leqq j \leqq D_i)$.

- $T_{i1}, T_{i2}, \ldots, T_{iD_i}$ $(1 \leqq i \leqq N)$ は互いに異なる．
- 各 i, j $(1 \leqq i \leqq N, 1 \leqq j \leqq D_i)$ に対し，$T_{T_{ijk}} = i$ を満たす k $(1 \leqq k \leqq D_{T_{ij}})$ が存在する．
- どの 2 つの部屋の間も，何本かの道を使うことで互いに移動できる．

小課題

以下では，入力データにおけるダンジョンの部屋の数を N，道の本数を M とする．

小課題 1 [17 点]

以下の条件を満たす．

- $N \leqq 50$.
- $M \leqq 100$.
- $X = 100$.

小課題 2 [27 点]

以下の条件を満たす．

- $N \leqq 50$.
- $M \leqq 100$.
- $X = 3$.

小課題 3 [56 点]

- $X = 3$ を満たす．

この小課題では，以下に従い得点が決定される．

- この小課題の全てのテストケースにおける，次の値の最大値を L とおく．
  - `Move` の呼び出し回数を C としたときの，$\dfrac{C}{M}$.
- このとき，この小課題の得点は，
  - $L \leqq 14$ のとき，56 点．
  - $14 < L \leqq 32$ のとき，$\lfloor 70 - L \rfloor$ 点．

- $32 < L \leqq 64$ のとき，$\left\lfloor 54 - \frac{L}{2} \right\rfloor$ 点.
- $64 < L$ のとき，0 点.

ここで，$\lfloor x \rfloor$ は x を超えない最大の整数を表す.

採点システム上では，プログラムが正しく終了し，正解と判定された場合には詳細の欄に `Accepted` と表示される．ただし，小課題 3 の採点で，上の C, M が $64 < \frac{C}{M}$ となるテストケースについては，結果の欄には不正解と表示されるので注意せよ.

やりとりの例

採点プログラムのサンプルが読み込む入力の例と，それに対応するルーチンの呼び出しの例を以下に示す.

入力例	ルーチンの呼び出しの例	
`4 3 3`	呼び出し	戻り値
`1`	`Inspect(3)`	
`2`	`NumberOfRoads()`	`1`
`3`	`LastRoad()`	`-1`
`1 3 4`	`Move(1,2)`	
`2`	`Color()`	`1`
`2 4`	`LastRoad()`	`1`
`2`	`NumberOfRoads()`	`3`
`2 3`	`Move(1,3)`	
`4`	`Color()`	`2`
`2`	`Answer(1,4)`	
`0`	`Answer(2,2)`	
	`Answer(3,0)`	

この例での関数の呼び出しは，必ずしも意味のある呼び出しとは限らないことに注意せよ.

回転寿司 (Sushi)

回転寿司屋 JOI では寿司の乗った皿を環状のベルトコンベアで運んでいる．ベルトコンベアは反時計回りに回っている．現在，店内には客 1 から客 N までの N 人の客がいて，客はベルトコンベアの周りに番号順に反時計回りに並んでいる．客 N の隣には客 1 が座っている．

客は 1 人 1 枚ずつの皿を持っている．各皿には**価格**と呼ばれる値が定まっており，店を出る際に，客は，自分の持っている皿の価格に等しい金額を支払う．

回転寿司屋 JOI では，一風変わったタイムセールを実施している．このタイムセールでは，Q 回に分けて，順番に板前から皿が提供される．i 回目 ($1 \leqq i \leqq Q$) の皿の提供の内容は 3 個の整数の組 (S_i, T_i, P_i) で表される．

タイムセールのルールは次の通りである．タイムセールが始まる直前に，板前はベルトコンベア上の皿を全て回収する．以下の 1〜3 を $i = 1, 2, \ldots, Q$ まで繰り返す．

1. 板前がベルトコンベアの客 S_i の前の位置に，価格 P_i の皿を置く．
2. 皿は客 S_i の場所から客 T_i の場所までベルトコンベア上を移動する．それぞれの客は，自分の前の皿に対し，次の行動を行う．
   - もし，その皿の価格が自分の持っている皿の価格より小さいならば，自分の皿とベルトコンベア上の皿を交換する．
   - もし，その皿の価格が自分の持っている皿の価格以上であれば，皿の交換は行わない．
3. 皿が客 T_i の前を通過した後，板前がその皿を回収する．

あなたは板前のもとで職人修業をしている弟子であり，店の皿洗いを任せられている．回転寿司屋 JOI の皿は価格によって洗い方が異なる．あなたは皿洗いの準備のために，タイムセールにおける Q 回の提供において板前がどの価格の皿を回収するのかをあらかじめ知っておきたい．

補足 (競技終了後に追記)

$S_i = T_i$ のときは，客 S_i のみが 2 つ目の行動を行う．

課題

それぞれの客がタイムセールの直前に持っている皿の情報と，タイムセールにおける皿の提供の情報が与えられたとき，それぞれの皿の提供で板前が回収する皿の価格を求めるプログラムを作成せよ．

入力

標準入力から以下のデータを読み込め．

- 1 行目には，整数 N, Q が空白を区切りとして書かれている．これは，客の人数が N 人，タイムセールにおける皿の提供回数が Q 回であることを表す．
- 続く N 行のうちの i 行目 $(1 \leqq i \leqq N)$ には，整数 X_i が書かれている．これは，客 i がタイムセールの直前に価格 X_i の皿を持っていることを表す．
- 続く Q 行のうちの i 行目 $(1 \leqq i \leqq Q)$ には，整数 S_i, T_i, P_i が空白を区切りとして書かれている．これは，i 回目の皿の提供が組 (S_i, T_i, P_i) で表されることを表す．

出力

出力は Q 行からなる．標準出力の i 行目 $(1 \leqq i \leqq Q)$ には，i 回目の皿の提供で板前が回収する皿の価格を表す整数を出力せよ．

制限

すべての入力データは以下の条件を満たす．

- $1 \leqq N \leqq 400\,000$.
- $1 \leqq Q \leqq 25\,000$.
- $1 \leqq X_i \leqq 1\,000\,000\,000$ $(1 \leqq i \leqq N)$.
- $1 \leqq S_i \leqq N$ $(1 \leqq i \leqq Q)$.
- $1 \leqq T_i \leqq N$ $(1 \leqq i \leqq Q)$.
- $1 \leqq P_i \leqq 1\,000\,000\,000$ $(1 \leqq i \leqq Q)$.

小課題

小課題 1 [5 点]

以下の条件を満たす．

- $N \leqq 2\,000$.
- $Q \leqq 2\,000$.

小課題 2 [15 点]

以下の条件を満たす.

- $S_i = 1$ $(1 \leqq i \leqq Q)$.
- $T_i = N$ $(1 \leqq i \leqq Q)$.

小課題 3 [80 点]

追加の制限はない.

入出力例

入力例 1	出力例 1
6 7 8 6 7 4 5 9 2 4 5 4 1 4 6 2 7 1 5 2 3 4 8 4 3 1 3 1 3	7 9 8 7 8 6 5

客 1 から客 6 までの持っている皿の価格は，それぞれの皿の提供の後，以下のようになる.

- 1 回目の皿の提供の後，8, 5, 6, 4, 5, 9
- 2 回目の皿の提供の後，8, 5, 6, 4, 4, 5
- 3 回目の皿の提供の後，7, 5, 6, 4, 4, 5
- 4 回目の皿の提供の後，2, 5, 6, 4, 4, 5
- 5 回目の皿の提供の後，2, 5, 6, 4, 4, 5
- 6 回目の皿の提供の後，2, 5, 5, 1, 4, 4

- 7 回目の皿の提供の後，2, 5, 3, 1, 4, 4

入力例 2	出力例 2
4 2 5 2 4 7 1 4 3 1 4 1	7 5

入力例 2 は，小課題 2 の制限を満たす．

入力例 3	出力例 3
10 10 19 5 8 17 14 3 9 10 7 6 1 8 4 7 3 2 5 9 10 4 8 3 10 3 6 8 7 4 6 6 3 2 9 12 6 3 7 9 6 3	19 10 14 17 8 10 3 12 7 9

電報 (Telegraph)

JOI 諸島は太平洋に浮かぶ小さな島国である．JOI 諸島には N 個の島があり，1 から N までの番号が付けられている．

JOI 諸島では島同士の通信は主に無線によって行われる．それぞれの島には電波の送信機と受信機がひとつずつある．送信機は全方向に電波を送ることができるが，受信機は特定の方向からの電波しか受け取ることができない．そのため，それぞれの受信機は，特定のひとつの島からの電波しか受け取ることができない．ただし，受信機の向きを変えることで，どの島からの電波を受け取ることができるかを変えることができる．

現在，島 i $(1 \leqq i \leqq N)$ の受信機は島 A_i $(A_i \neq i)$ からの電波を受け取ることができる．また，島 i の受信機の向きを変えるのにかかるコストは，どう向きを変えるかによらず，C_i である．

JOI 諸島では，公共事業として電報サービスを行っている．島 i $(1 \leqq i \leqq N)$ からの電波を島 j $(1 \leqq j \leqq N, j \neq i)$ の受信機が受け取ることができるとき，島 i から島 j に無線通信によって電報を送ることができる．また，電報はいくつかの島を経由して送ってもよい．すなわち，島 i，島 j，島 k $(1 \leqq i, j, k \leqq N$ かつ i, j, k はそれぞれ相異なる) について，島 i から島 j に電報を送ることができ，島 j から島 k に電報を送ることができるとき，島 i から島 k に電報を送ることができる．無線通信以外の方法で電報を送ることはできない．

JOI 諸島の逓信大臣であるあなたは，任意の島から任意の島へ電報を送れるようにしたい．そのためには，いくつかの島の受信機の向きを変える必要があるかもしれない．いくつかの島の受信機の向きを変えるのにかかるコストは，それぞれの受信機の向きを変えるのにかかるコストの総和である．

任意の島から任意の島へ電報を送れるようにするためにかかるコストの最小値を計算せよ．

課題

JOI 諸島の島の数と，それぞれの島の受信機に関する情報が与えられたとき，任意の島から任意の島に電報を送れるようにするためにかかるコストの最小値を求めるプログラムを作成せよ．

入力

標準入力から以下のデータを読み込め．

- 1 行目には，整数 N が書かれている．これは，JOI 諸島には N 個の島があることを表す．
- 続く N 行のうちの i 行目 $(1 \leqq i \leqq N)$ には，整数 A_i, C_i が空白を区切りとして書かれている．これは，島 i の受信機は，現在，島 A_i からの電波を受け取ることができ，向きを変えるのにかかるコストが C_i であることを表す．

出力

標準出力に，任意の島から任意の島に電報を送れるようにするためにかかるコストの最小値を 1 行で出力せよ．

制限

すべての入力データは以下の条件を満たす．

- $2 \leqq N \leqq 100\,000$.
- $1 \leqq A_i \leqq N$ $(1 \leqq i \leqq N)$.
- $A_i \neq i$ $(1 \leqq i \leqq N)$.
- $1 \leqq C_i \leqq 1\,000\,000\,000$ $(1 \leqq i \leqq N)$.

小課題

小課題 1 [10 点]

- $N \leqq 10$ を満たす．

小課題 2 [30 点]

- $N \leqq 15$ を満たす．

小課題 3 [30 点]

- $N \leqq 3\,000$ を満たす．

小課題 4 [30 点]

追加の制限はない．

入出力例

入力例 1	出力例 1
4 2 2 1 4 1 3 3 1	4

島 2 の受信機の向きを変え，島 4 からの電波を受け取れるようにする．すると任意の島から任意の島へ電報を送れるようになり，かかったコストは 4 である．

どのように受信機の向きを変えてもコストが 4 を下回ることはないので，4 を出力する．

入力例 2	出力例 2
4 2 2 1 6 1 3 3 1	5

まず，島 1 の受信機の向きを変え，島 4 からの電波を受け取れるようにする．次に，島 3 の受信機の向きを変え，島 2 からの電波を受け取れるようにする．すると任意の島から任意の島へ電報を送れるようになり，かかったコストは $2 + 3 = 5$ である．

どのように受信機の向きを変えてもコストが 5 を下回ることはないので，5 を出力する．

入力例 3	出力例 3
4 2 2 1 3 4 2 3 3	4

島 1 と島 3 の受信機の向きを変えればよい．

入力例 4	出力例 4
3 2 1 3 1 1 1	0

どの島の受信機の向きも変えなくてよい．