EJOY

EJOY 2067

Задача 2. Подкуп

⏳0,67 сек. 💾256 MB

Боби има граф, но не знае какъв е. Той може да попита журито дали в графа има връзка между върхове u и v ($1 \leq u, v \leq N$) с пряко ребро между тях, и ако да, какво е теглото му. Целта на Боби е да намери минимално покриващо дърво на графа си. Проблемът е, че на журито му е скучно и решава да се гаври с вас. Журито може да ви излъже за това дали има ребро между два върха. В началото шансът да Ви излъже е 67%. Боби е богат и има безкрайни пари, с които може да подкупи журито да не ви лъже. Цената на един подкуп е 1 лев, а един подкуп намаля шанса да ви излъже с 1%. Вие трябва да помогнете на Боби да даде възможно най-малко пари (дори и да е богат) и едновременно да намери най-минималното покриващо дърво, което може.

Задача

Напишете програма **`bribe`**, съдържаща функция `solve`, която ще се компилира с програмата на журито и ще комуникира с нея, като задава въпроси за ребра и подкупва журито. Вашата програма трябва да намери минималното покриващо дърво с информацията, която има.

Детайли по имплементацията

Функцията `void solve(int n)`, която трябва да напишете, ще бъде извикана само веднъж от програмата на журито и като аргумент ще получи броя върхове в графа N и парите на Боби M. За комуникация с програмата на журито Ви се предоставят следните функции:

```
int query(int u, int v);
void bribe();
void submit(int l);
```

При всяко извикване на функцията `query`, тя ще върне теглото на реброто ако журито твърди, че съществува, а в противен случай връща -1. Функцията `bribe` извършва описания подкуп. След като откриете минималното покриващо дърво, вашата функция трябва да извика `submit` и да предаде като аргумент сумата на теглата в откритото минимално покриващо дърво. След това изпълнението на вашата функция ще бъде прекратена.

Вашата програма `bribe.cpp` трябва да имплементира функцията `solve`. Тя може да съдържа и друг код, и функции, необходими за работата Ви, но не трябва да съдържа главната функция `main`. Също така, не трябва да четете от стандартния вход или да отпечатвате на стандартния изход. Програмата Ви трябва да включва хедър файла `bribe.h` чрез указание към предпроцесора:

```
#include "bribe.h"
```

Ограничения и оценяване

- $1 \leq N \leq 1000$
- $1 \leq$ тегло на ребро ≤ 10000
- Точките на даден тест се пресмятат по някакви глупави сметки, които отнеха прекалено много време за измисляне и няма да се обясняват тук.

EJOY 2067

Примерна комуникация

Функция на участника	Програма на журито
	`solve(5)`
`query(1, 2)`	`1`
`query(2, 3)`	`-1`
`bribe()`	
`query(3, 4)`	`1`
`query(4, 5)`	`1`
`submit(4)`	

Пояснение: В примера графът е пръчка, а на втората заявка журито ни лъже.

Локално тестване

Предоставен Ви е файлът `Lgrader.cpp`, който може да компилирате заедно с вашата програма, за да я тествате. При стартиране програмата ще чете от стандартния вход стойността на N и после броя ребра M. На следващите M реда чете по 3 числа - двата върха, които свързва, и теглото. След това ще отпечатва комуникацията, която се извършва. Може да модифицирате предоставения файл, както искате.