Σχέδιο Παιχνιδιού (Toy Design)

Ονομασία Προβλήματος	Σχέδιο Παιχνιδιού
Αρχείο εισόδου	Διαδραστικό Πρόβλημα (Interactive Task)
Αρχείο εξόδου	Διαδραστικό Πρόβλημα (Interactive Task)
Χρονικό όριο	1 δευτερόλεπτο
Όριο μνήμης	256 megabytes

Εργάζεστε σε μια εταιρεία που σχεδιάζει παιχνίδια. Ένα καινούριο παιχνίδι που δημιουργήθηκε παίζεται ως εξής: Υπάρχουν n ακίδες, αριθμημένες από 1 έως n, που προεξέχουν από ένα κουτί. Μερικά ζεύγη ακίδων συνδέονται με καλώδια στο εσωτερικό του κουτιού. (Με άλλα λόγια, οι ακίδες και τα καλώδια σχηματίζουν έναν μη κατευθυνόμενο γράφο, όπου οι ακίδες είναι οι κορυφές και τα καλώδια οι ακμές). Τα καλώδια δεν είναι ορατά από έξω και και ο μόνος τρόπος για να μάθετε κάτι για αυτά είναι να χρησιμοποιήσετε έναν **ελεγκτή** στις ακίδες: Μπορείτε να επιλέξετε δύο ακίδες i και j τέτοιες ώστε $i \neq j$, και ο ελεγκτής θα σας πει αν αυτές οι δύο ακίδες συνδέονται άμεσα ή έμμεσα μέσα στο κουτί. (Έτσι, ο ελεγκτής ενημερώνει εάν υπάρχει ένα μονοπάτι μεταξύ αυτών των δύο ακίδων στον γράφο).

Ας ονομάσουμε **σχέδιο** το σύνολο των συνδέσεων μέσα στο κουτί του παιχνιδιού.

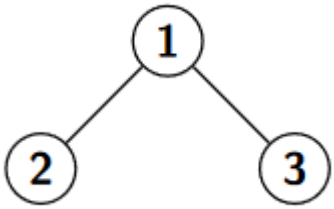
Για να θέσετε ερωτήματα και να δημιουργήσετε αυτά τα σχέδια χρησιμοποιείτε ένα εξειδικευμένο λογισμικό. Αυτό το λογισμικό δουλεύει ως εξής: Ξεκινά με κάποιο σχέδιο του παιχνιδιού, το οποίο συμβολίζουμε ως "σχέδιο 0". Δεν σας δείχνει τις συνδέσεις μέσα στο κουτί για αυτό το σχέδιο. Αντί αυτού μπορείτε να εκτελέσετε επανειλημμένα την ακόλουθη λειτουργία τριών-βημάτων:

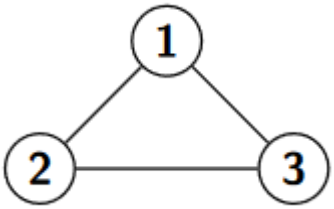
1. Επιλέγετε έναν αριθμό σχεδίου a και δύο αριθμούς ακίδων i και j τέτοια ώστε $i \neq j$.
2. Το λογισμικό σας ενημερώνει τι θα συνέβαινε εάν χρησιμοποιούσατε τον ελεγκτή σε αυτές τις δύο ακίδες. Με άλλα λόγια, σας λέει εάν οι ακίδες i και j είναι (άμεσα ή έμμεσα) συνδεδεμένες στο σχέδιο a.
3. Επίσης, εάν οι ακίδες δεν ήταν, άμεσα ή έμμεσα συνδεδεμένες, στο σχέδιο a, τότε δημιουργεί ένα νέο σχέδιο, το οποίο έχει όλες τις συνδέσεις από το σχέδιο a και επιπρόσθετα άλλη μια άμεση σύνδεση μεταξύ των i και j. Σε αυτό το σχέδιο δίνεται ο επόμενος διαθέσιμος αριθμός σχεδίου. (Έτσι, το πρώτο σχέδιο που δημιουργήθηκε με

αυτόν τον τρόπο θα έχει τον αριθμό 1, μετά τον αριθμό 2 και ούτω καθεξής). Σημειώστε ότι αυτό δεν αλλάζει το σχέδιο a, απλώς δημιουργεί ένα νέο σχέδιο που έχει την επιπρόσθετη σύνδεση.

Στόχος σας είναι να μάθετε όσο το δυνατόν περισσότερα για το σχέδιο 0 χρησιμοποιώντας αυτήν τη λειτουργία.

Σημειώστε ότι δεν είναι πάντα δυνατό να προσδιοριστεί το ακριβές σύνολο συνδέσεων για το σχέδιο 0, επειδή δεν υπάρχει τρόπος να διακριθούν οι άμεσες και οι έμμεσες συνδέσεις. Για παράδειγμα, εξετάστε τα ακόλουθα δύο σχέδια με $n = 3$:





Ο ελεγκτής θα αναφέρει οποιοδήποτε ζεύγος ακίδων ως συνδεδεμένο και για τα δύο σχέδια, επομένως δεν θα μπορείτε να τα διακρίνετε χρησιμοποιώντας το λογισμικό που περιγράφεται παραπάνω.

Στόχος σας είναι να προσδιορίσετε οποιοδήποτε σχέδιο, το οποίο είναι ισοδύναμο με το σχέδιο 0. Δύο σχέδια είναι **ισοδύναμα** εάν ο ελεγκτής αναφέρει το ίδιο αποτέλεσμα και στα δύο σχέδια για όλα τα ζεύγη ακίδων.

Υλοποίηση

Αυτό είναι ένα διαδραστικό πρόβλημα (interactive problem). Πρέπει να υλοποιήσετε μια συνάρτηση

```
void ToyDesign(int n, int max_ops);
```

που καθορίζει ένα σχέδιο που είναι *ισοδύναμο* με το σχέδιο 0. Η υλοποίησή σας θα πρέπει να επιτύχει αυτόν τον στόχο καλώντας δύο συναρτήσεις, όπως περιγράφονται παρακάτω. Η πρώτη συνάρτηση που μπορείτε να καλέσετε είναι:

```
int Connected(int a, int i, int j);
```

όπου $1 \leq i, j \leq n$, $i \neq j$, $a \geq 0$, και ο a δεν πρέπει να υπερβαίνει το πλήθος των σχεδίων που υπάρχουν μέχρι τώρα. Εάν οι ακίδες i και j είναι (άμεσα ή έμμεσα) συνδεδεμένες στο σχέδιο a, τότε θα επιστρέψει a. Διαφορετικά, θα επιστρέψει τον αριθμό των σχεδίων που έχουν δημιουργηθεί μέχρι τώρα συν ένα, που γίνεται ο αριθμός που εκχωρείται στο νέο σχέδιο που έχει όλες τις συνδέσεις του σχεδίου a συν τη σύνδεση μεταξύ i και j. Η συνάρτηση `Connected` μπορεί να κληθεί το πολύ `max_ops` φορές.

Όταν το πρόγραμμά σας τελειώσει με τις λειτουργίες `Connected`, θα πρέπει να περιγράψει ένα σχέδιο που ισοδυναμεί με το σχέδιο 0. Για να περιγράψει ένα σχέδιο, το πρόγραμμα θα πρέπει να καλεί την:

```
void DescribeDesign(std::vector<std::pair<int,int>> result);
```

Η παράμετρος `result` είναι ένας πίνακας (vector) ζευγών ακεραίων που περιγράφουν τις άμεσες συνδέσεις μεταξύ των ακίδων. Κάθε ζεύγος αντιστοιχεί σε μία σύνδεση και πρέπει να περιέχει τους δύο αριθμούς ακίδων της σύνδεσης. Πρέπει να υπάρχει το πολύ μία άμεση σύνδεση μεταξύ κάθε (χωρίς συγκεκριμένη διάταξη) ζεύγους ακίδων και καμία άμεση σύνδεση μεταξύ μιας ακίδας και του εαυτού της. Η κλήση αυτής της συνάρτησης τερματίζει την εκτέλεση του προγράμματός σας.

Περιορισμοί

- $2 \leq n \leq 200$

Βαθμολόγηση

- Subtask 1 (10 βαθμοί): $n \leq 200$, *max_ops* $= 20\,000$
- Subtask 2 (20 βαθμοί): $n \leq 8$, *max_ops* $= 20$
- Subtask 3 (35 βαθμοί): $n \leq 200$, *max_ops* $= 2\,000$
- Subtask 4 (35 βαθμοί): $n \leq 200$, *max_ops* $= 1\,350$

Παράδειγμα Διάδρασης (Sample Interaction)

Ενέργεια υποψηφίου	Ενέργεια βαθμολογητή	Επεξήγηση
	`ToyDesign(4, 20)`	Υπάρχουν 4 ακίδες στο παιχνίδι. Πρέπει να προσδιορίσετε οποιοδήποτε σχέδιο, το οποίο είναι ισοδύναμο με το σχέδιο 0 καλώντας την `Connected` το πολύ 20 φορές.
`Connected(0, 1, 2)`	`Επιστρέφει 1.`	Οι ακίδες 1 και 2 δεν συνδέονται άμεσα ή έμμεσα στο σχέδιο 0. Το νέο σχέδιο 1 δημιουργείται.
`Connected(1, 3, 2)`	`Επιστρέφει 2.`	Οι ακίδες 3 και 2 δεν συνδέονται άμεσα ή έμμεσα στο σχέδιο 1. Το νέο σχέδιο 2 δημιουργείται.

`Connected(0, 3, 4)`	`Επιστρέφει 0.`	Οι ακίδες 3 και 4 συνδέονται άμεσα ή έμμεσα στο σχέδιο 0. Δεν δημιουργείται κανένα νέο σχέδιο.
`DescribeDesign({{3, 4}})`	-	Περιγράφουμε ένα σχέδιο που έχει μόνο μία σύνδεση: τις ακίδες 3 και 4.

Παράδειγμα Βαθμολογητή (Sample Grader)

Το παράδειγμα βαθμολογητή που σας δίνεται, `grader.cpp`, στο επισυναπτόμενο του προβλήματος `ToyDesign.zip`, διαβάζει τα δεδομένα εισόδου από την τυπική είσοδο (standard input) με την ακόλουθη μορφή:

- Η πρώτη γραμμή περιέχει τον αριθμό των ακίδων, n, τον αριθμό των άμεσων συνδέσεων, m και *max_ops*
- Οι επόμενες m γραμμές περιέχουν τις άμεσες συνδέσεις ως πλειάδες (tuples) από ακίδες.

Το παράδειγμα βαθμολογητή διαβάζει την είσοδο και καλεί την συνάρτηση `ToyDesign` στη λύση σας. Ο βαθμολογητής θα εμφανίσει ένα από τα παρακάτω μηνύματα, ανάλογα με τη συμπεριφορά της λύση σας:

- "Wrong answer: Number of operations exceeds the limit.", εάν ο αριθμός των κλήσεων της `Connected` υπερβαίνει το *max_ops*
- "Wrong answer: Wrong design id.", εάν η παράμετρος a σε μια κλήση της `Connected` είναι ο αριθμός ενός σχεδίου που δεν υπάρχει τη στιγμή που έγινε η κλήση.
- "Wrong answer: Incorrect design.", εάν το σχέδιο που περιγράφεται μέσω της `DescribeDesign` δεν είναι ισοδύναμο με το σχέδιο 0.
- "OK!" εάν το σχέδιο που περιγράφεται μέσω της `DescribeDesign` είναι ισοδύναμο με το σχέδιο 0.

Για να μεταγλωττίσετε (compile) το παράδειγμα βαθμολογητή με τη λύση σας, μπορείτε να χρησιμοποιήσετε την ακόλουθη εντολή στη γραμμή εντολών (terminal prompt):

```
g++ -std=gnu++11 -O2 -o solution grader.cpp solution.cpp
```

όπου `solution.cpp` είναι το αρχείο της λύσης σας που πρέπει να υποβληθεί στο CMS. Για να εκτελέσετε το πρόγραμμα με το δείγμα εισόδου που παρέχεται στο συνημμένο, πληκτρολογήστε την ακόλουθη εντολή στη γραμμή εντολών:

```
./solution < input.txt
```