Toy Design

Tehtävän nimi	ToyDesign
Syötetiedosto	Interaktiivinen tehtävä
Tulostetiedosto	Interaktiivinen tehtävä
Aikaraja	1 sekunti
Muistiraja	256 megabytes

Työskentelet yrityksessä joka suunnittelee leluja. Uusi lelu jota ollaan tekemässä toimii näin: On n tappia, numeroitu 1 :stä n :ään, jotka törröttävät laatikosta. Jotkut parit tappeja ovat kiinnitetty toisiinsa narulla boksin sisällä (Toisin sanoen, tapit ja langat muodostavat suuntaamattoman verkon, missä tapit ovat solmuja ja langat kaaria). Langat eivät näy laatikon ulkopuolelle, joten ainoa tapa saada selville jotain niistä on käyttää **testaajaa** tapeille: Voimme ottaa kaksi tappia i ja j siten että $i \neq j$, ja testaaja kertoo jos nämä kaksi tappia ovat laatikon sisällä toisiinsa yhdistettyjä, joko suorasti tai epäsuorasti (Eli, testaaja kertoo jos näiden kahden tapin välillä on polku verkossa).

Kutsumme yhteyksien joukkoa laatikon sisällä lelun **muotoiluksi**.

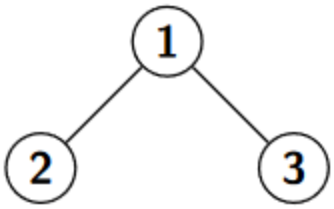
Käytät erikoistunutta ohjelmistoa kyselläksesi ja luodaksesi näitä muotoiluita. Ohjelmisto toimii näin: Se aloittaa jollain lelun muotoilulla jota kutsumme "muotoilu 0":ksi. Se ei näytä sinulle yhteyksiä laatikon sisällä. Sen sijaan, voit toistaen tehdä seuraavan kolmevaiheisen operaation:

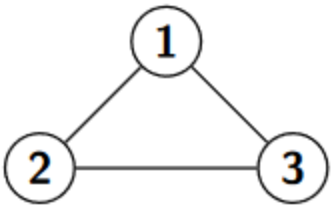
- 1. Voit ottaa muotoilun numero a ja kaksi tapin numeroa i ja j siten että $i \neq j$.
- 1. Ohjelmisto kertoo sinulle mitä tapahtuu kun käytämme testeriä näille tapeille. Toisin sanoen, se kertoo sinulle jos tapit i ja j ovat (suorasti tai epäsuorasti) yhdistetty muotoilussa a.
- 1. Myös, jos tapit eivät ole suorasti tai epäsuorasti yhdistettyjä muotoilussa a, se luo uuden muotoilun jossa on kaikki yhteydet muotoilusta a sekä lisäksi yksi suora kaari i:n ja j:n välillä. Tälle muotoilulle annetaan seuraava saatavissa oleva muotoilun numero. (Joten, ensimmäinen muotoilu joka tehdään tällä tavalla saa numeron 1, seuraava 2, ja niin edelleen). Muista että tämä ei muuta muotoilua a, vaan tekee vain uuden muotoilun jossa on ylimääräinen yhteys.

Sinun tehtäväsi on oppia niin paljon kuin mahdollista muotoilusta 0 käyttämällä tätä operaatiota.

Muista että aina ei ole mahdollista määrittää eksaktia joukkoa yhteyksiä muotoilulle 0, koska koska ei ole mahdollista erotella suoria ja epäsuoria yhteyksiä. Esimerkiksi, katso seuraavia kahta

muotoilua kun $n = 3$:





Testaaja raportoi kaikkien tappien olevan yhdistettyjä, joten emme osaa erotella niitä käyttäen yllä kuvattua ohjelmistoa.

Tavoitteesi on määritellä mikä tahansa muotoilu joka on ekvivalentti muotoilun 0 kanssa. Kaksi muotoilua ovat **ekvivalentteja** jos testaaja raportoi saman tuloksen molemmille muotoilulle kaikille tapeille.

Toteutus

Tämä on interaktiivinen tehtävä. Sinun pitää toteuttaa funktio

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

joka määrittää muotoilun joka on *ekvivalentti* muotoilun 0 kanssa. Toteutuksesi tulee saavuttaa tämä tavoite kahta funktiota alla kuvatulla tavalla.

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

missä $1 \leq i, j \leq n, i \neq j, a \geq 0$, ja a ei saa ylittää jo luotujen muotoilujen määrää. Jos tapit i and j on (suorasti tai epäsuorasti) yhdistetty muotoilussa a, se palauttaa takaisin saman muotoilun a. Muuten se palauttaa jo luotujen muotoilujen määrän plus yksi, mistä tulee uudelle muotoilulle annettu numero. Tässä uudessa muotoilussa on kaikki samat yhteydet kuin a:ssa sekä lisäksi yhteys i:n ja j:n välillä. Funktiota `Connected` voidaan kutsua korkeintaan `max_ops` kertaa.

Kun ohjelmasi on valmis `Connected`-operaatioiden kanssa, sen pitää määrittää muotoilu, joka on ekvivalentti muotoilun 0 kanssa. Määrittääkseen muotoilun, ohjelman pitää kutsua funktioita

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

Parametri `result` on vektori kokonaislukupareja jotka määrittävät suorat yhteydet tappien välillä. Jokainen pari vastaa yhtä yhteyttä ja sisältää kaksi tapin numeroa. On olemassa korkeintaan yksi suora yhteys kahden (järjestämättömän) tappiparin välillä, ja tapista itseensä ei ole koskaan suoraa yhteyttä. Tämän funktion kutsuminen lopettaa ohjelman suorituksen.

Rajoitukset

- $2 \leq n \leq 200$

Pisteytys

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

Esimerkki-interaktio

Kilpailijan toiminta	Arvioijan toiminta	Selitys
	`ToyDesign(4, 5)`	Lelussa on 4 tappia. Sinun pitää määrittää muotoilu joka on ekvivalentti muotoilun 0 kanssa kutsumalla `Connected` korkeintaan 5 kertaa.
`Connect(0, 1, 2)`	`Palauttaa 1.`	Tapit 1 ja 2 eivät ole yhdistettyjä suorasti tai epäsuorasti in design 0. Uusi muotoilu 1 luodaan.
`Connected(1, 3, 2)`	`Palauttaa 2.`	Tapit 3 ja 2 ei ole yhditetty suorasti tai epäsuorasti muotoilussa 1. Uusi muotoilu 2 luodaan
`Connected(0, 3, 4)`	`Palauttaa 0.`	Tapit 3 ja 4 ovat yhdistettyjä suorasti tai epäsuorasti muotoilussa 0. Yhtään uutta muotoilua ei luoda
`DescribeDesign({{3, 4}})`	-	Määritämme muotoilun jossa on vain yksi yhteys: Tapit 3 ja 4.

Esimerkkiarvioija

Annettu esimerkkiarvioija, `grader.cpp`, tehtävän liitteessä `ToyDesign.zip`, lukee syötteen standardisyötteestä seuraavassa formaatissa:

- Ensimmäinen rivi sisältää tappien määrän, n, suorien yhteyksien määrän, m, ja *max_ops*
- Seuraavat m riviä sisältää suorat yhteydet pareina tappeja.

Esimerkkiarvioija lukee syötteen ja kutsuu `ToyDesign`-funktiota käyttäjän ratkaisussa.

Kääntääksesi esimerkkiarvioijan ratkaisullasi aja seuraava komento komentorivillä:

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

missä `solution.cpp` on ratkaisutiedostosi submitoitavaksi CMS:ään. Ajaaksesi ohjelman liitteenä olevalla esimerkkisyötteellä, aja seuraava komento komentorivillä:

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