

ToruŇin kaltevasta tornista kerrotaan monia legendoja. Tornin seinämä muodostaa ympyrän, jossa on $N \geq 3$ tasavälein sijoitettua ovea (toisin sanoen ovet muodostavat säännöllisen N -kulmion kärkipisteet). Ovet on numeroitu luvusta 0 lukuun $N - 1$, mutta **satunnaisessa järjestyksessä**. Lisätietoja tästä löytyy pisteytysosioista.

Erään vähemmän tunnetun legendan mukaan tornin jokaisen uuden asukkaan piti suorittaa tietty haaste. Haasteen tavoitteena oli luetella ovet, alkaen jostakin ovesta, ja sitten kulkemalla ympäri ympyrää (myötä- tai vastapäivään), käyden jokaisella ovella tarkalleen kerran.

Tämä täytyy tehdä näkemättä tornia. Uusi asukas voi kuitenkin esittää seuraavanlaisia kyselyitä: “Annetuna kolme erillistä ovea x, y, z , mikä ovipari on lähimpänä toisiaan: $\{x, y\}$, $\{y, z\}$ vai $\{z, x\}$?”. Vastauksena annetaan kaikki ne oviparit (joukosta $\{x, y\}$, $\{y, z\}$ ja $\{z, x\}$), joiden välinen etäisyys on pienin. Etäisyys on yksinkertaisesti lyhyimmän ovia yhdistävän janan pituus. Tehtävänäsi on kirjoittaa ohjelma, joka esittää vain vähän tällaisia kyselyitä ja selvittää ovien järjestyksen.

Vuorovaikutus

Tämä on interaktiivinen tehtävä. Sinun tulee kirjoittaa ohjelma, joka löytää oikean ratkaisun tehtävään ja kommunikoi vuorovaikuttajan kanssa lukemalla standardisyötteestä ja kirjoittamalla standarditulosteeseen.

Vuorovaikutuksen alussa ohjelmasi lukee kaksi kokonaislukua t ja k ($1 \leq t \leq 100$, $1 \leq k \leq 12\,000$) standardisyötteestä, missä t on testitapausten määrä ja k suurin sallittu keskimääräinen kyselyiden määrä. Katso pisteytysosio lisätietoja varten.

Jokaisessa testitapauksessa ohjelmasi lukee ensin yhden kokonaisluvun n ($3 \leq n \leq 500$) standardisyötteestä, joka ilmaisee ovien lukumäärän tornissa.

Sen jälkeen ohjelmasi voi esittää kyselyitä seuraavassa muodossa:

- Ohjelmasi tulostaa rivin muodossa

? x y z

standarditulosteeseen, missä x, y ja z ovat eri kokonaislukuja ($0 \leq x, y, z \leq n - 1$). Tämä rivi edustaa yksittäistä kyselyä ovista x, y ja z .

- Vastaus annetaan muodossa:

r
 a_1 b_1
...
 a_r b_r

missä r on kokonaisluku ($1 \leq r \leq 3$), joka kertoo pienimmän etäisyyden oviparien määrän. Jokainen tällainen pari kuvataan kahdella kokonaisluvulla a_i ja b_i ($a_i, b_i \in \{x, y, z\}$ ja $a_i < b_i$).

Kun olet määrittänyt ovien järjestyksen, sinun tulee tulostaa yksi rivi muodossa

! x_0 x_1 ... x_{n-1}

standarditulosteeseen, missä x_0, x_1, \dots, x_{n-1} on ovien järjestys kuten tehtävänannossa kuvattiin. Huomaa, että täsmälleen $2n$ järjestystä kelpaa, koska järjestyksen voi aloittaa mistä tahansa ovesta ja kulkea kumpaan tahansa suuntaan. Kaikki nämä hyväksytään.

Muista, että jokaisen kyselyn tai vastauksen jälkeen sinun tulee tyhjentää tulostepuskuri komennolla `cout.flush()` (tai `fflush(stdout)` `printf` käyttäessä) C++:ssa tai `sys.stdout.flush()` Pythonissa. Muuten ohjelmasi saattaa saada `Time Limit Exceeded`-tuloksen.

Kun ohjelma on tulostanut vastauksen vuorovaikuttajalle, sen tulee siirtyä seuraavaan testitapaukseen tai lopettaa vuorovaikutus, jos kaikki testitapaukset on käsitelty.

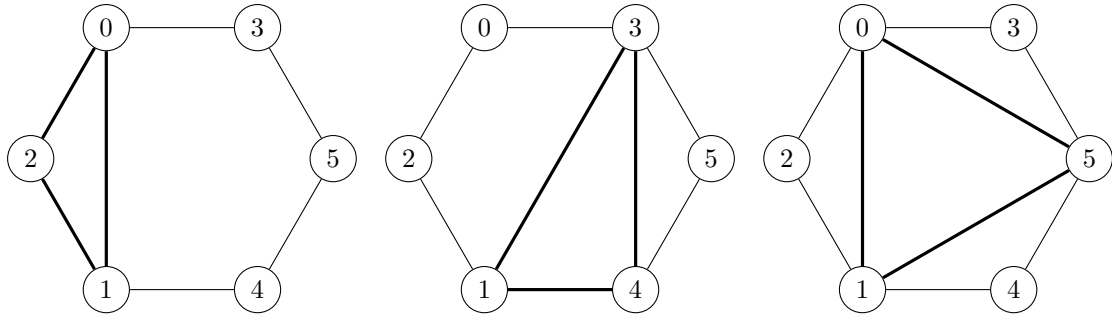
Ohjelmasi ei saa avata tiedostoja tai käyttää muita resursseja. Virheiden selvittämistä varten saa kirjoittaa virhetulosteeseen, mutta huomaa, että sen kirjoittaminen vie aikaa.

Lisäksi huomaa, että vuorovaikuttaja ei ole adaptiivinen, eli ovien alkuperäinen järjestys on kiinteä jokaisessa testitapauksessa eikä muutu vuorovaikutuksen aikana.

Esimerkki vuorovaikutuksesta

Oletetaan, että on vain yksi testitapaus, jossa $n = 6$, ja ovien järjestys on 5, 3, 0, 2, 1, 4. Vuorovaikutus voi näyttää seuraavalta:

Vuorovaikuttuja	Ohjelmasi	Kommentti
1 100		$t = 1$ and $k = 100$.
6		Vuorovaikuttuja antaa määrän ovia ensimmäisessä tapauksessa.
	? 0 1 2	Ohjelmasi kysyy mitkä parit ovia ovat lähimpänä.
2 0 2 1 2		Oviparit $\{0, 2\}$ ja $\{1, 2\}$ ovat lähimpänä.
	? 4 1 3	Ohjelmasi kysyy mitkä parit ovia ovat lähimpänä.
1 1 4		Pari $\{1, 4\}$ on lähimpänä.
	? 0 5 1	Ohjelmasi kysyy mitkä parit ovia ovat lähimpänä.
3 0 5 0 1 1 5		Parit $\{0, 5\}$, $\{0, 1\}$, ja $\{1, 5\}$ ovat lähimpänä.
	! 4 5 3 0 2 1	Ohjelmasi tulostaa ovien järjestyksen oikein.



Selitys: Yllä olevissa kuvissa ovet ja niiden numerot näkyvät tornin seinämällä. Ensimmäisessä kuvassa vasemmalta on kolmio, jonka muodostavat ovet 0, 1, 2, vastaten ohjelmasi ensimmäistä kyselyä. Siitä nähdään, että parit $\{0, 2\}$ ja $\{1, 2\}$ ovat lähimpänä. Toisessa kuvassa näkyy kolmio ovista 1, 4, 3, joka vastaa ohjelmasi toista kyselyä. On selvästi nähtävissä, että pari $\{1, 4\}$ on lähimpänä. Kolmannessa kuvassa kolmio ovista 0, 1, 5, joka vastaa ohjelmasi kolmatta kyselyä, ja kaikki oviparit ovat yhtä lähellä toisiaan.

Huomaa, että järjestykset 0, 2, 1, 4, 5, 3 tai 5, 4, 1, 2, 0, 3 (ja muutama muu) ovat myös hyväksytyjä ratkaisuja tässä tapauksessa.

Pisteytys

Pisteytys on jaettu osatehtäviin. Jokaisessa osatehtävässä on tarkalleen yksi testi, joka sisältää tarkalleen $t = 100$ testitapausta. Jokaisessa testissä ohjelmasi kyselyiden määrän keskiarvo lasketaan jakamalla kyselyiden kokonaismäärä testitapausten määrällä. Jos tämä keskiarvo ylittää annetun k :n osatehtävässä, saat 0 pistettä siitä osatehtävästä. Muuten, osatehtävissä 1–4 saat täydet pisteet osatehtävästä.

Viimeisessä osatehtävässä pisteet lasketaan seuraavasti. Olkoon k^* ohjelmasi todellinen kyselyiden keskiarvo. Tällöin pistemäärä lasketaan kaavalla:

$$\left\lceil 56 \cdot \min \left(1, \frac{12000 - k^*}{7800} \right) \right\rceil,$$

eli saat pisteitä lineaarisesti välillä 0–56 kun k^* muuttuu välillä 12000–4200.

Huomaa, että jos ohjelmasi antaa virheellisen vastauksen johonkin testitapaukseen, saat 0 pistettä kyseisestä osatehtävästä riippumatta kyselyiden määrästä.

Osatehtävien lisärajoitukset on esitetty alla olevassa taulukossa.

Osatehtävä	Rajat	Pisteet
1	$k = 8000, 3 \leq n \leq 9$	6
2	$k = 4500, 40 \leq n \leq 50$	7
3	$k = 3000, 90 \leq n \leq 100$	9
4	$k = 4500, n = 400$, on olemassa oikea vastaus x_0, \dots, x_{n-1} missä $x_i = i$ kaikille $200 \leq i \leq 399$	22
5	$k = 12\,000, n = 500$	56 asti

Lisäksi voidaan olettaa, että jokainen testitapaus on luotu ensin valitsemalla n **tasaisesti satunnaisesti** kaikista sallitun osatehtävän ehdoilla kelpaavista arvoista, ja sitten valitsemalla ovien järjestys **tasaisesti satunnaisesti** kaikista järjestyksistä, jotka täyttävät osatehtävän ehdot.