Programmēšana: RAM moduļu permutācijas

Problēmas apraksts

Savā brīvajā laikā ārpus darba Lemon AI Jānis tirgojas ar RAM moduļiem, lai gūtu papildu ienākumus.

Kādu dienu Jānis atrada neticamu piedāvājumu augstas veiktspējas RAM moduļu komplektā un nolēma uzlabot uzņēmuma serveri. Katram modulim ir unikāls sērijas numurs (no 1 līdz n).

Tomēr ir viena nianse: serverim ir savdabīga BIOS konfigurācijas sistēma. Optimālai veiktspējai RAM moduļiem jāinstalē noteiktā secībā, ko nosaka mātesplates atmiņas kontrolieris, bet instrukcijas nekur nav!

Par laimi, BIOS piedāvā diagnostikas režīmu: Jānis var testēt RAM moduļu izkārtojumu, un sistēma dos punktu skaitu, kas norāda, cik slotiem ir pareizais modulis.

Būdams sacensību programmētājs, Jānis atpazīst to kā **kombinatorisku optimizāciju (Combinatorial Optimization, CO)** problēmu — pareizā izkārtojuma atrašanu diskrētā telpā.

> ***Kombinatoriska optimizācija (Combinatorial Optimization, CO)*** *— joma, kas koncentrējas uz optimālu risinājumu atrašanu diskrētās telpās (piem., permutācijas, kombinācijas, grafi). CO problēmas ir ārkārtīgi grūtas, jo iespējamo kombināciju skaits eksponenciāli aug ar elementu skaitu. Tās bieži tiek risinātas, izmantojot neironu risinātājus un pastiprinājuma mācīšanos (Reinforcement Learning, RL), lai gan gudri algoritmi var efektīvi atrisināt konkrētus gadījumus.*

Permutācija ir noteikts doto elementu sakārtojums. Piemēram, iespējamās skaitļu 1, 2, 3 permutācijas ir:

```
- 1 - 2 - 3
- 1 - 3 - 2
- 2 - 1 - 3
- 2 - 3 - 1
- 3 - 1 - 2
- 3 - 2 - 1
```

Jūsu uzdevums ir palīdzēt Jānim efektīvi noskaidrot pareizo RAM konfigurāciju!

Uzdevums

Šī ir interaktīva problēma.

Servera BIOS ir slepena optimālā konfigurācija C ar n RAM moduļiem ($1 \leq n \leq 31$).

Katrs modulis ir apzīmēts ar unikālu sērijas numuru ($1 \leq \mathrm{id} \leq n$).

Jāņa mērķis ir atklāt pareizo konfigurāciju, izmantojot BIOS diagnostikas režīmu.

Katrā vaicājumā Jānis testē izkārtojumu P un saņem punktu skaitu S.

Punktu skaits S tiek aprēķināts šādi:

- Katrai pozīcijai i ($1 \leq i \leq n$) pievienojam 2^i punktiem S, ja modulis pozīcijā i izkārtojumā P atbilst modulim pozīcijā i konfigurācijā C.

Piemēram, ja:

- Optimālā BIOS konfigurācija $C = [2, 1, 3, 4]$
- Jāņa testa izkārtojums $P = [4, 1, 3, 2]$

Tad punktu skaits $S = 2^2 + 2^3 = 12$, jo moduļi pozīcijās 2 un 3 sakrīt (abiem ir modulis 1 pozīcijā 2, un abiem ir modulis 3 pozīcijā 3).

Jūsu uzdevums ir palīdzēt Jānim atklāt konfigurāciju C ne vairāk kā n vaicājumos.

Ievads

Pirmajā rindā ir vesels skaitlis n ($1 \leq n \leq 31$).

Mijiedarbība

Varat testēt izkārtojumu, izmantojot BIOS diagnostiku, vaicājot `? p_1 p_2 ... p_n` caur doto `query` funkciju, kur:

- `p_1, p_2, ..., p_n`: piedāvātais izkārtojums P ar n moduļiem

BIOS atbildēs ar veselu skaitli S (punktu skaitu).

Izvads

Kad esat noteicis optimālo konfigurāciju, vaicājiet `! c_1 c_2 ... c_n`, kur $c_1, c_2, \ldots, c_n$ ir konfigurācija C.

Ierobežojumi

Ierobežojumi uz ievadi ir:

- testa gadījumu skaits $\leq 2 \times 10^4$
- n summa visiem testa gadījumiem $\leq 1 \times 10^6$

Izpildes laika ierobežojums ir 20 sekundes. Atmiņas ierobežojums ir 512 MB.

Piemērs

Ņemiet $n = 3$, $C = [2, 1, 3]$.
Uzdosim divus vaicājumus, lai noteiktu C, un pēc tam izvadīsim rezultātu.

Ievads:

```
3
```

Mijiedarbība:

```
? 1 2 3
8
```

```
? 2 3 1
2
! 2 1 3
```

Vērtēšana

Jūsu risinājums tiks pārbaudīts uz slēptu testu skaita.

Rezultāts = nokārtotie testi / kopējie testi