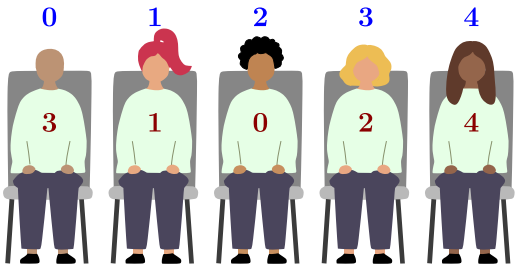
D. Stoelindeling (seatingplan)

De sluitingsceremonie van deze EGOI zal worden bijgewoond door N belangrijke gasten. Ze moeten allemaal op de voorste rij zitten in een heel specifieke volgorde die aansluit bij alle nuances van het diplomatieke protocol. Het vaststellen van de juiste stoelindeling heeft Noemi twee slapeloze nachten gekost.

Veronica is verantwoordelijk voor de sluitingsceremonie. Een van de vele dingen die ze moet doen, is ervoor zorgen dat de naamkaartjes op de voorste rij kloppen. Er is alleen één klein probleempje: Noemi heeft haar nooit verteld wat de juiste stoelindeling is, en nu is Noemi nergens te bekennen. Gelukkig heeft Dorka de fotograaf een app die zou kunnen helpen.

Dorka moest haar camera's zo instellen dat ze een aantal specifieke foto's kon maken van gasten op de eerste rij. Voor de opstelling moest ze weten hoe breed elke foto zou zijn, en daarom heeft Noemi voor haar een app gemaakt die snel de benodigde informatie geeft. Veronica wil de app nu gebruiken om de juiste stoelindeling te vinden.

De N belangrijke gasten zijn genummerd van 0 tot en met $N-1$. De stoelen op de voorste rij zijn ook genummerd van 0 tot en met $N-1$, van links naar rechts. Voor elke I ($0 \leq I \leq N-1$) is g_I de gast die op stoel I hoort te zitten, en s_I is de stoel waarop gast I hoort te zitten.



Figuur 1: Een rij met vijf gasten. Voor deze rij geldt $g = [3, 1, 0, 2, 4]$ en $s = [2, 1, 3, 0, 4]$.

De app werkt als volgt:

- Dorka voert de nummers I, J, K van precies drie verschillende gasten in.
- De app vertelt haar wat het minimale aantal gasten is dat zichtbaar zal zijn als alle drie de geselecteerde gasten op de foto staan. Formeel gesproken zal de app de waarde $\max(s_I, s_J, s_K) - \min(s_I, s_J, s_K) + 1$ geven.

Zie bijvoorbeeld de situatie in Figuur 1:

- Gasten $I = 0$, $J = 2$, en $K = 4$ zitten op stoelen $s_I = 2$, $s_J = 3$, en $s_K = 4$. Als Dorka hen selecteert, geeft de app de waarde $\max(2, 3, 4) - \min(2, 3, 4) + 1 = 3$.

  Met andere woorden, op de smalste foto waarop de gasten 0, 2 en 4 staan, staan precies deze drie gasten.

- Gasten $I = 0$, $J = 4$, en $K = 3$ zitten op stoelen $s_I = 2$, $s_J = 4$, en $s_K = 0$. Als Dorka hen selecteert, geeft de app de waarde $\max(2, 4, 0) - \min(2, 4, 0) + 1 = 5$.

  Met andere woorden, op een foto waarop de drie gegeven gasten staan, moeten alle 5 gasten staan.

Help Veronica de juiste stoelindeling vast te stellen met Dorka's app. Om precies te zijn: je programma moet de rij $g_0, g_1, ..., g_{N-1}$ bepalen en uitvoeren. Er zijn altijd precies twee juiste antwoorden (de één is de omgekeerde van de ander), en je mag één van beide uitvoeren. Je score hangt af van het aantal vragen die je oplossing aan de app stelt.

Implementatie

⇨ Dit is een interactief probleem. Je programma gebruikt standaardinvoer en -uitvoer om te communiceren met een grader op de manier die hieronder beschreven is.

Je programma moet beginnen met het lezen van één regel invoer waarop een positief geheel getal T staat, het aantal testgevallen dat daarna komt.

Voor elk testgeval moet je programma beginnen met het lezen van één regel invoer waarop een positief geheel getal N staat, het aantal stoelen en dus ook het aantal gasten.

Om een vraag te stellen, moet je programma een regel uitvoeren in de vorm "? I J K", waarbij $0 \leq I, J, K \leq N-1$ drie **verschillende** getallen zijn.

Na het stellen van een vraag moet je programma één regel lezen waarop één positief geheel getal staat, het antwoord op je vraag.

Om de juiste de stoelindeling als resultaat te geven, moet je programma een regel uitvoeren in de vorm "! g_0 … g_{N-1}".

Nadat alle T testgevallen zijn opgelost, moet je programma normaal afsluiten.

Let op: de officiële grader die in CMS wordt gebruikt om je oplossing te testen, kan **adaptief** zijn. Dat betekent dat voor sommige testgevallen de permutatie van gasten niet vooraf is bepaald. In plaats daarvan kan de grader zijn gedrag veranderen afhankelijk van de specifieke vragen die je programma stelt.

Buffers leegmaken (flushing). Als je de meegeleverde templates niet gebruikt, zorg er dan voor dat je de standard output leegmaakt (flusht) na het afdrukken van elke regel, anders wordt je programma mogelijk beoordeeld als *Not correct*. In Python gebeurt dit automatisch als je `input()` gebruikt om regels te lezen, en je kan `print(..., flush=True)` gebruiken om het te forceren om te flushen. In C++ zorgt `cout << endl;` ervoor dat standard output wordt leeggemaakt, naast het afdrukken van een nieuwe regel; als je printf gebruikt, gebruik dan `fflush(stdout)`.

Randvoorwaarden

- $1 \leq T \leq 10$.
- N zal gelijk zijn aan 5 (alleen voorbeeld), 8, 40, of 2000.
- Voor elk testgeval mag je maximaal 10 000 vragen stellen.

Scoring

Je programma wordt getest op verschillende testgevallen die zijn gegroepeerd in subtasks (deelopgaven). Om de punten voor een subtask te behalen, moet je alle tests die de subtask bevat correct oplossen.

- **Subtask 0 [0 punten]**: Voorbeeld ($N = 5$).
- **Subtask 1 [9 punten]**: $N = 8$.
- **Subtask 2 [11 punten]**: $N = 2000$, en gasten 0 en 1 zitten naast elkaar.
- **Subtask 3 [15 punten]**: $N = 40$.
- **Subtask 4 [65 punten]**: $N = 2000$.

Voor subtasks 1 en 2 krijgt elke oplossing die alle testgevallen correct oplost alle punten.

Voor subtasks 3 en 4 moet je oplossing alle testgevallen correct oplossen om punten te scoren. Bovendien hangt je score zal af van Q_s, het grootste aantal vragen dat je oplossing nodig had om een testgeval op te lossen. We definiëren $X_s = \max(1, Q_s/N)$. De scores voor subtasks 3 en 4 worden als volgt berekend:

$$\text{score}_3 = \min\left(15, 3 + \frac{19}{X_s^{1.5}}\right), \qquad \text{score}_4 = \min\left(65, 3 + \frac{91}{X_s^{1.5}}\right)$$

De waarde van score_s wordt afgerond op het dichtstbijzijnde gehele getal per subtaak, en je totale score is de som over alle subtasks. Om alle punten te behalen, moet je subtask 3 in maximaal 55 vragen oplossen en subtask 4 in maximaal 2597 vragen. Voorbeeldwaardes van Q_s en punten voor subtaken 3 en 4 worden hieronder getoond.

Q_s	55	56	60	70	80	100	150	10000
score_3	15	14	13	11	10	8	6	3

Q_s	2597	2800	3000	4000	5000	6000	8000	10000
score_4	65	58	53	35	26	21	14	11

Voorbeelden

Grader	Oplossing
1 5	
	? 0 2 4
3	
	? 3 0 1
3	
	? 0 4 3
5	
	! 3 1 0 2 4

Uitleg

De voorbeeldinvoer bevat één testgeval ($T = 1$) met $N = 5$ gasten. De verborgen stoelindeling voor de gasten in dit testgeval komt overeen met Figuur 1.

De eerste vraag van de voorbeeldoplossing is 0, 2, 4. Het antwoord 3 op deze vraag vertelt ons dat deze gasten, in een onbekende volgorde, op drie opeenvolgende stoelen naast elkaar zitten.

Het antwoord 3 op de tweede vraag vertelt ons hetzelfde over de gasten 3, 0 en 1.

We kunnen hieruit afleiden dat gast 0 in het midden moet zitten, met gasten 2 en 4 aan de ene kant en gasten 1 en 3 aan de andere kant.

Na de derde vraag kunnen we er al zeker van zijn dat de gasten ofwel in de volgorde $[3, 1, 0, 2, 4]$ of in de omgekeerde volgorde $[4, 2, 0, 1, 3]$ moeten zitten. Het maakt niet uit welke van deze volgordes je uitvoert.

Code templates en details van de beoordeling in CMS

We adviseren met klem om de beschikbare code templates voor C++ en Python te gebruiken. Deze controleren of de communicatie met de grader succesvol is en stoppen netjes als dat niet zo is.

Als je niet de beschikbare templates gebruikt, kan het gebeuren dat CMS een verkeerde beoordeling geeft, als je oplossing incorrect is. Bijvoorbeeld: in plaats van "Output isn't correct" zou je "Execution killed by signal" of "Execution timed out (wall clock limit exceeded)" kunnen krijgen.

We adviseren je ook om de meegeleverde testing tool te gebruiken (zie hieronder) om je oplossing lokaal te testen voordat je haar inlevert. De testing tool controleert de uitvoer van je oplossing en laat het weten als je ongeldige vragen stelt.

Testing tool

Om het makkelijker te maken je oplossing te testen, bieden we je een eenvoudige tool aan die je kunt downloaden van CMS. Het gebruik van de tool is optioneel. Let op: de officiële grader op CMS is anders dan de testing tool.

Om de testing tool te gebruiken, heb je een invoerbestand nodig. Je kunt de meegeleverde voorbeeldinvoer `seatingplan.input0.txt` gebruiken of je eigen invoer maken. Het invoerbestand moet beginnen met een regel waarop het aantal T testgevallen staat en daarna per testgeval twee regels: één regel met het aantal N en één regel met de nummers $g_0, g_1, ..., g_{N-1}$.

Voor Python-programma's, bijvoorbeeld `seatingplan.py` (normaalgesproken uitgevoerd als `pypy3 seatingplan.py`), voer je het testprogramma als volgt uit:

```
python3 testing_tool.py pypy3 seatingplan.py < seatingplan.input0.txt
```

Voor C++-programma's compileer je eerst je oplossing:

```
g++ -DEVAL -std=gnu++20 -O2 -pipe -static -s -o seatingplan seatingplan.cpp
```

en voer je daarna het testprogramma uit:

```
python3 testing_tool.py ./seatingplan < seatingplan.input0.txt
```