

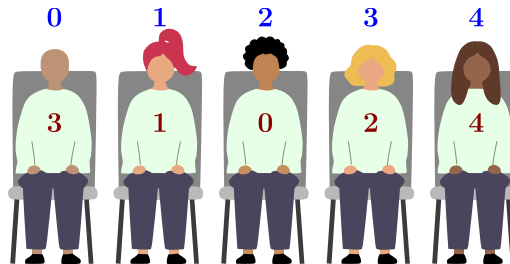
D. Σχέδιο Καθισμάτων (seatingplan)

Στην τελετή λήξης αυτού του ΕΓΟΙ θα παρευρεθούν N επίσημοι καλεσμένοι. Όλοι τους πρέπει να καθίσουν στην πρώτη γραμμή με μια πολύ συγκεκριμένη διάταξη, που να συμβαδίζει με όλες τις λεπτομέρειες του διπλωματικού πρωτοκόλλου. Η Νοέμι πέρασε δύο άπνες νύχτες για να βρει τη σωστή διάταξη.

Η Βερόνικα επιβλέπει την τελετή λήξης. Μια από τις πολλές αρμοδιότητές της είναι να βεβαιωθεί ότι στις θέσεις της πρώτης γραμμής υπάρχουν τα σωστά ονόματα. Υπάρχει όμως ένα μικρό προβληματάκι: η Νοέμι δεν της είπε ποτέ ποια είναι η σωστή διάταξη και τώρα η Νοέμι είναι άφαντη. Ευτυχώς, η φωτογράφος Ντόρκα έχει μια εφαρμογή που ίσως βοηθήσει.

Η Ντόρκα έπρεπε να προετοιμάσει τις κάμερές της ώστε να τραβήξει μερικές συγκεκριμένες φωτογραφίες των καλεσμένων στην πρώτη γραμμή. Για το στήσιμο, χρειαζόταν να ξέρει πόσο ευρεία θα είναι κάθε φωτογραφία, οπότε η Νοέμι της έφτιαξε μια εφαρμογή που βγάζει γρήγορα τις πληροφορίες που χρειάζεται. Η Βερόνικα θέλει τώρα να χρησιμοποιήσει αυτήν την εφαρμογή για να βρει τη σωστή τοποθέτηση των καλεσμένων στα καθίσματα.

Οι N επίσημοι καλεσμένοι είναι αριθμημένοι από 0 έως $N - 1$. Τα καθίσματα στην πρώτη γραμμή είναι επίσης αριθμημένα από 0 έως $N - 1$, από αριστερά προς τα δεξιά. Για κάθε I ($0 \leq I \leq N - 1$), έστω g_I ο καλεσμένος που υποτίθεται ότι κάθεται στο κάθισμα I , και έστω s_I το κάθισμα στο οποίο υποτίθεται ότι κάθεται ο καλεσμένος I .



Σχήμα 1: Μια γραμμή με πέντε καλεσμένους. Για αυτή τη γραμμή, $g = [3, 1, 0, 2, 4]$ και $s = [2, 1, 3, 0, 4]$.

Η εφαρμογή λειτουργεί ως εξής:

- Η Ντόρκα εισάγει τους αριθμούς I , J , K τριών διαφορετικών καλεσμένων.
- Η εφαρμογή της δείχνει τον ελάχιστο αριθμό καλεσμένων που θα είναι ορατοί αν και οι τρεις επιλεγμένοι καλεσμένοι βρίσκονται στη φωτογραφία. Τυπικά, η εφαρμογή θα εμφανίσει την τιμή $\max(s_I, s_J, s_K) - \min(s_I, s_J, s_K) + 1$.

Για παράδειγμα, δες την κατάσταση στο Σχήμα 1:

- Οι καλεσμένοι $I = 0$, $J = 2$ και $K = 4$ κάθονται στις θέσεις $s_I = 2$, $s_J = 3$ και $s_K = 4$. Αν η Ντόρκα τους επιλέξει, η εφαρμογή θα εμφανίσει την τιμή $\max(2, 3, 4) - \min(2, 3, 4) + 1 = 3$.

Με άλλα λόγια, η πιο στενή φωτογραφία που περιέχει τους καλεσμένους 0, 2 και 4 έχει μόνο αυτούς τους τρεις καλεσμένους.

- Οι καλεσμένοι $I = 0$, $J = 4$ και $K = 3$ κάθονται στις θέσεις $s_I = 2$, $s_J = 4$ και $s_K = 0$. Αν η Ντόρκα τους επιλέξει, η εφαρμογή θα εμφανίσει την τιμή $\max(2, 4, 0) - \min(2, 4, 0) + 1 = 5$.

Με άλλα λόγια, μια φωτογραφία που περιέχει τους τρεις αυτούς καλεσμένους πρέπει να περιλαμβάνει και τους 5 καλεσμένους.

Βοηθήστε τη Βερόνικα να βρει τη σωστή διάταξη καθισμάτων χρησιμοποιώντας την εφαρμογή της Ντόρκα. Πιο συγκεκριμένα, το πρόγραμμά σας πρέπει να προσδιορίσει και να εκτυπώσει τη σειρά g_0, g_1, \dots, g_{N-1} . Υπάρχουν πάντα ακριβώς δύο σωστές απαντήσεις (η μία είναι η αντίστροφη της άλλης), και μπορείτε να εκτυπώσετε οποιαδήποτε από τις δύο. Η βαθμολογία σας θα εξαρτηθεί από τον αριθμό των ερωτημάτων (queries) που θα κάνει η λύση σας στην εφαρμογή.

Υλοποίηση

⇒ Αυτό είναι ένα διαδραστικό πρόβλημα. Το πρόγραμμά σας θα χρησιμοποιεί standard input και output για να επικοινωνήσει με τον κριτή (grader) με τη μορφή που περιγράφεται παρακάτω.

Το πρόγραμμά σας πρέπει να ξεκινά διαβάζοντας μια γραμμή εισόδου που περιέχει έναν θετικό ακέραιο T , τον αριθμό των περιπτώσεων ελέγχου (test cases) που ακολουθούν.

Για κάθε περίπτωση ελέγχου (test case), το πρόγραμμά σας πρέπει να ξεκινά διαβάζοντας μια γραμμή εισόδου που περιέχει έναν θετικό ακέραιο N , τον αριθμό των καθισμάτων, που είναι και ο αριθμός των καλεσμένων.

Για να κάνετε ένα ερώτημα (query), το πρόγραμμά σας πρέπει να εκτυπώσει μια γραμμή της μορφής «? $I J K$ », όπου $0 \leq I, J, K \leq N - 1$ είναι τρεις **διαφορετικοί** αριθμοί.

Αφού κάνετε το ερώτημα, το πρόγραμμά σας πρέπει να διαβάσει μια γραμμή που περιέχει έναν θετικό ακέραιο, την απάντηση στο ερώτημά σας.

Για να δώσετε τη σωστή διάταξη καθισμάτων, το πρόγραμμά σας πρέπει να εκτυπώσει μια γραμμή της μορφής «! $g_0 \dots g_{N-1}$ ».

Αφού λύσετε όλα τις T περιπτώσεις ελέγχου (test cases), το πρόγραμμά σας πρέπει να τερματίσει κανονικά.

Σημειώστε ότι ο επίσημος κριτής (grader) που χρησιμοποιείται στο CMS για τον έλεγχο της λύσης σας μπορεί να είναι **προσαρμοστικός (adaptive)**. Δηλαδή, για ορισμένες περιπτώσεις ελέγχου (test cases), η μετάθεση των καλεσμένων δεν είναι προκαθορισμένη. Αντ' αυτού, ο κριτής (grader) μπορεί να αλλάζει τη συμπεριφορά του ανάλογα με τα ερωτήματα που θέτει το πρόγραμμά σας.

Flushing. Εάν δεν χρησιμοποιείτε τα παρεχόμενα πρότυπα, φροντίστε να εκκαθαρίσετε την τυπική έξοδο (standard output) μετά την εκτύπωση κάθε γραμμής, διαφορετικά το πρόγραμμά σας μπορεί να κριθεί ως *Not correct*. Στην Python, αυτό συμβαίνει αυτόματα εάν χρησιμοποιήσετε το `input()` για να διαβάσετε γραμμές και μπορείτε να χρησιμοποιήσετε το `print(..., flush=True)` για να το αναγκάσετε να εκκαθαρίσει. Στην C++, το `cout << endl`; εκκαθαρίζει εκτός από την εκτύπωση μιας νέας γραμμής. Εάν χρησιμοποιείτε `printf`, χρησιμοποιήστε το `fflush(stdout)`.

Περιορισμοί

- $1 \leq T \leq 10$.
- Το N θα είναι 5 (μόνο για παράδειγμα), 8, 40, ή 2000.
- Για κάθε περίπτωση ελέγχου (test case), μπορείτε να κάνετε το πολύ 10000 ερωτήματα.

Βαθμολογία

Το πρόγραμμά σας θα ελεγχθεί σε αρκετές περιπτώσεις ελέγχου (test cases) ομαδοποιημένες σε υποπροβλήματα (subtasks). Για να πάρετε τους βαθμούς ενός υποπροβλήματος, πρέπει να λύσετε σωστά όλες τις περιπτώσεις ελέγχου που περιέχει.

- **Υποπρόβλημα 0** [0 πόντοι]: Παράδειγμα ($N = 5$).
- **Υποπρόβλημα 1** [9 πόντοι]: $N = 8$.

- **Υποπρόβλημα 2 [11 πόντοι]:** $N = 2000$, και οι καλεσμένοι 0 και 1 κάθονται ο ένας δίπλα στον άλλον.
- **Υποπρόβλημα 3 [15 πόντοι]:** $N = 40$.
- **Υποπρόβλημα 4 [65 πόντοι]:** $N = 2000$.

Για τα υποπροβλήματα 1 και 2, οποιαδήποτε λύση που επιλύει σωστά όλα τις περιπτώσεις ελέγχου (test cases) θα πάρει όλους τους βαθμούς.

Για τα υποπροβλήματα 3 και 4, η λύση σας πρέπει να επιλύει σωστά όλες τις περιπτώσεις ελέγχου (test cases) για να πάρει βαθμούς, και η βαθμολογία σας θα εξαρτηθεί από το Q_s , τον μέγιστο αριθμό ερωτημάτων που χρειάστηκε η λύση σας για να επιλύσει μια περίπτωση ελέγχου (test case). Έστω $X_s = \max(1, Q_s/N)$. Οι βαθμοί για τα υποπροβλήματα 3 και 4 υπολογίζονται ως εξής:

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

Η τιμή του score_s στρογγυλοποιείται στον πλησιέστερο ακέραιο ανά υποπρόβλημα, και η συνολική σας βαθμολογία είναι το άθροισμα αυτών. Για να πάρετε το άριστα, πρέπει να λύσετε το υποπρόβλημα 3 με το πολύ 55 ερωτήματα και το υποπρόβλημα 4 με το πολύ 2597 ερωτήματα. Παραδείγματα τιμών του Q_s και οι βαθμολογίες για τα υποπροβλήματα 3 και 4 φαίνονται παρακάτω.

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

Παραδείγματα εισόδου/εξόδου

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

Εξήγηση

Το παράδειγμα εισόδου περιέχει μία περίπτωση ελέγχου (test case) ($T = 1$) με $N = 5$ καλεσμένους. Η κρυφή διάταξη των καλεσμένων σε αυτήν την περίπτωση ελέγχου (test case) αντιστοιχεί στο Σχήμα 1.

Το πρώτο ερώτημα που κάνει η ενδεικτική λύση είναι 0, 2, 4. Η απάντηση 3 σε αυτό το ερώτημα μας λέει ότι αυτοί οι καλεσμένοι κάθονται, με κάποια άγνωστη διάταξη, σε τρία συνεχόμενα καθίσματα δίπλα-δίπλα.

Η απάντηση 3 στο δεύτερο ερώτημα μας λέει το ίδιο για τους καλεσμένους 3, 0 και 1.

Μπορούμε τώρα να συμπεράνουμε ότι ο καλεσμένος 0 πρέπει να κάθεται στη μέση, με τους καλεσμένους 2 και 4 από τη μία πλευρά και τους καλεσμένους 1 και 3 από την άλλη.

Μετά το τρίτο ερώτημα, μπορούμε ήδη να είμαστε σίγουροι ότι οι καλεσμένοι πρέπει να κάθονται είτε στη σειρά [3, 1, 0, 2, 4] είτε στην αντίστροφη σειρά [4, 2, 0, 1, 3]. Μπορούμε να εκτυπώσουμε οποιαδήποτε από τις δύο σειρές.

Πρότυπα Κώδικα και Λεπτομέρειες Αξιολόγησης στο CMS

Σας συνιστούμε ανεπιφύλακτα να χρησιμοποιείτε τα παρεχόμενα πρότυπα κώδικα για C++ και Python. Αυτά ελέγχουν εάν η επικοινωνία με τον βαθμολογητή (grader) ήταν επιτυχής και τερματίζονται ομαλά όταν δεν ήταν.

Εάν δεν χρησιμοποιήσετε τα παρεχόμενα πρότυπα, σε περιπτώσεις όπου η λύση σας είναι λανθασμένη, το CMS ενδέχεται να εμφανίσει λανθασμένη ετυμηγορία. Για παράδειγμα, αντί για «Output isn't correct», ενδέχεται να εμφανιστεί το μήνυμα «Execution killed by signal» or «Execution timed out (wall clock limit exceeded)».

Σας συνιστούμε επίσης να χρησιμοποιήσετε το εργαλείο ελέγχου (δείτε παρακάτω) για να δοκιμάσετε τη λύση σας τοπικά πριν την υποβάλετε. Το εργαλείο ελέγχου (testing tool) εξετάζει τα αποτελέσματα της λύσης σας και αναφέρει τη χρήση μη έγκυρων ερωτημάτων.

Εργαλείο Ελέγχου (Testing Tool)

Για να διευκολύνουμε τον έλεγχο της λύσης σας, παρέχουμε ένα απλό εργαλείο που μπορείτε να κατεβάσετε από το CMS. Η χρήση του εργαλείου είναι προαιρετική. Σημείωσε ότι ο επίσημος κριτής (grader) στο CMS είναι διαφορετικός από το εργαλείο ελέγχου.

Για να χρησιμοποιήσετε το εργαλείο, χρειάζεστε ένα αρχείο εισόδου. Μπορείτε να χρησιμοποιήσετε το παρεχόμενο παράδειγμα εισόδου `seatingplan.input0.txt` ή να φτιάξετε το δικό σας. Το αρχείο εισόδου πρέπει να ξεκινά με μια γραμμή που έχει τον αριθμό T των περιπτώσεων ελέγχου (test cases) και στη συνέχεια να έχει δύο γραμμές ανά περίπτωση ελέγχου (test case): μια γραμμή με τον αριθμό N και ακολούθως μια γραμμή με τους αριθμούς g_0, g_1, \dots, g_{N-1} .

Για προγράμματα σε Python, ας πούμε `seatingplan.py` (συνήθως εκτελούνται ως `py3 seatingplan.py`), τρέξετε το εργαλείο ελέγχου ως εξής:

```
python3 testing_tool.py py3 seatingplan.py < seatingplan.input0.txt
```

Για προγράμματα σε C++, πρώτα κάνετε compile τη λύση σας:

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

και μετά τρέξετε το εργαλείο ελέγχου:

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