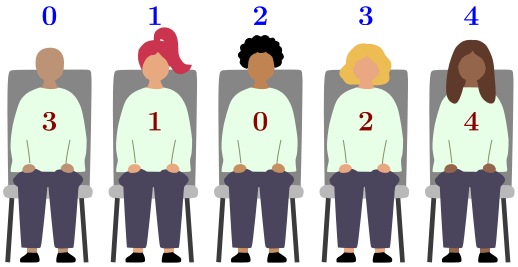
D. Plani i Uljes (seatingplan)

Ceremonia mbyllëse e kësaj EGOI do të ndiqet nga N mysafirë të rëndësishëm. Të gjithë këta duhet të ulen në rreshtin e parë në një renditje shumë specifike që korrespondon me të gjitha nuancat e protokollit diplomatik. Përcaktimi i renditjes së saktë të uljes i mori Noemit dy netë pa gjumë.

Veronika po mbikëqyr ceremoninë mbyllëse. Një nga përgjegjësitë e saj të shumta është të sigurohet që karriget e rreshtit të parë të kenë emrat e duhur. Ka vetëm një problem të vogël: Noemi kurrë nuk i tregoi renditjen e saktë të uljes, dhe tani Noemi nuk gjendet askund. Për fat të mirë, Dorka, fotografja, ka një aplikacion që mund të jetë i dobishëm.

Dorka duhej të përgatiste kamerat e saj në mënyrë që të mund të bënte disa foto specifike të mysafirëve në rreshtin e parë. Për konfigurimin, ajo duhej të dinte se sa e gjerë do të ishte secila foto, kështu që Noemi i bëri një aplikacion që nxjerr shpejt informacionin që i nevojitet. Veronika tani dëshiron të përdorë aplikacionin për të gjetur caktimin e saktë të vendeve.

N mysafirët e rëndësishëm janë të numëruar nga 0 deri në $N-1$. Karriget në rreshtin e parë janë gjithashtu të numëruara nga 0 deri në $N-1$, nga e majta në të djathtë. Për çdo I ($0 \leq I \leq N-1$), le të jetë g_I mysafiri që duhet të ulet në karrigen I, dhe le të jetë s_I karrigia në të cilën mysafiri I duhet të ulet.



Figurë 1: Një rresht me pesë mysafirë. Për këtë rresht, $g = [3, 1, 0, 2, 4]$ dhe $s = [2, 1, 3, 0, 4]$.

Aplikacioni funksionon si më poshtë:

- Dorka fut numrat I, J, K të saktësisht tre mysafirëve të ndryshëm.
- Aplikacioni i tregon asaj numrin minimal të mysafirëve që do të jenë të dukshëm nëse të tre mysafirët e zgjedhur janë në foto. Formalisht, aplikacioni do të shfaqë vlerën $\max(s_I, s_J, s_K) - \min(s_I, s_J, s_K) + 1$.

Për shembull, shihni situatën e treguar në Figurë 1:

- Mysafirët $I = 0$, $J = 2$, dhe $K = 4$ janë në karriget $s_I = 2$, $s_J = 3$, dhe $s_K = 4$. Nëse Dorka i zgjedh ata, aplikacioni do të shfaqë vlerën $\max(2, 3, 4) - \min(2, 3, 4) + 1 = 3$.

  Me fjalë të tjera, fotoja më e ngushtë që përmban mysafirët 0, 2, dhe 4 ka vetëm këta tre mysafirë.

- Mysafirët $I = 0$, $J = 4$, dhe $K = 3$ janë në karriget $s_I = 2$, $s_J = 4$, dhe $s_K = 0$. Nëse Dorka i zgjedh ata, aplikacioni do të shfaqë vlerën $\max(2, 4, 0) - \min(2, 4, 0) + 1 = 5$.

  Me fjalë të tjera, një foto që përmban tre mysafirët e dhënë duhet të përmbajë të gjithë 5 mysafirët.

Ndihmoni Veronikën të përcaktojë renditjen e saktë të uljes duke përdorur aplikacionin e Dorkës. Më konkretisht, programi juaj duhet të përcaktojë dhe të printojë sekuencën $g_0, g_1, \ldots, g_{N-1}$. Gjithmonë ka saktësisht dy përgjigje të sakta (njëra është me radhitje të kundërt), dhe ju mund të printoni cilëndo prej tyre. Rezultati juaj do të varet nga numri i pyetjeve që zgjidhja juaj i bën aplikacionit.

Implementimi

⇨ Ky është një problem interaktiv. Programi juaj do të përdorë standard input dhe output për të komunikuar me një vlerësues në formatin e përshkruar më poshtë.

Programi juaj duhet të fillojë duke lexuar një rresht inputi që përmban një numër të plotë pozitiv T, numrin e rasteve të testimit që pasojnë.

Për çdo rast testimi, programi juaj duhet të fillojë duke lexuar një rresht inputi që përmban një numër të plotë pozitiv N, numrin e karrigeve, që është gjithashtu numri i mysafirëve.

Për të bërë një pyetje, programi juaj duhet të printojë një rresht të formës "? I J K", ku $0 \leq I, J, K \leq N-1$ janë tre numra **të ndryshëm**.

Pas bërjes së një pyetjeje, programi juaj duhet të lexojë një rresht që përmban një numër të plotë pozitiv, përgjigjen e pyetjes suaj.

Për t'u përgjigjur me një renditje të saktë të uljes, programi juaj duhet të printojë një rresht të formës "! g_0 ... g_{N-1}".

Pas zgjidhjes së të gjitha rasteve të testimit T, programi juaj duhet të terminohet normalisht.

Vini re se vlerësuesi zyrtar i përdorur në CMS për të testuar zgjidhjen tuaj mund të jetë **adaptiv**. Kjo do të thotë që, për disa raste testimi, permutacioni i mysafirëve nuk përcaktohet paraprakisht. Në vend të kësaj, vlerësuesi mund të vendosë se cilin nga permutacionet e mbetura përdor në varësi të pyetjeve që programi juaj ka bërë tashmë.

Zbrazja (Flushing). Nëse nuk po përdorni shabllonet e dhëna, sigurohuni që të zbrazni standard output pas printimit të çdo rreshti, përndryshe programi juaj mund të gjykohet si *Jo i saktë*. Në Python, kjo ndodh automatikisht nëse përdorni input() për të lexuar rreshtat, dhe mund të përdorni print(..., flush=True) për ta detyruar atë të zbrazë standard output. Në C++, cout << endl; zbrazë standard output përveç printimit të një rreshti të ri; nëse përdorni printf, përdorni fflush(stdout).

Kufizimet

- $1 \leq T \leq 10$.
- N do të jetë 5 (vetëm tek shembulli), 8, 40, ose 2000.
- Për çdo rast testimi, mund të bëni maksimum 10 000 pyetje.

Vlerësimi

Programi juaj do të testohet në disa raste testimi të grupuara në nën-detyra. Për të marrë rezultatin për një nën-detyrë, duhet të zgjidhni saktë të gjitha testet që ajo përmban.

- **Nëndetyra 0 [0 points]**: Shembull ($N = 5$).
- **Nëndetyra 1 [9 points]**: $N = 8$.
- **Nëndetyra 2 [11 points]**: $N = 2000$, dhe mysafirët 0 dhe 1 ulen pranë njëri-tjetrit.
- **Nëndetyra 3 [15 points]**: $N = 40$.
- **Nëndetyra 4 [65 points]**: $N = 2000$.

Për nën-detyrat 1 dhe 2 çdo zgjidhje që zgjidh saktë të gjitha rastet e testimit do të vlerësohet me të gjitha pikët.

Për nën-detyrat 3 dhe 4, zgjidhja juaj duhet të zgjidhë saktë të gjitha rastet e testimit për të marrë ndonjë pikë, dhe rezultati juaj do të varet nga Q_s, numri më i madh i pyetjeve që zgjidhja juaj duhej të bënte për të zgjidhur një rast testimi. Le të jetë $X_s = \max(1, Q_s/N)$. Rezultatet për nën-detyrat 3 dhe 4 llogariten si më poshtë:

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

Vlera e score_s rrumbullakoset në numrin e plotë më të afërt për secilën nën-detyrë, dhe rezultati juaj total është shuma e këtyre. Për të marrë një rezultat të plotë, duhet të zgjidhni nën-detyrën 3 në maksimum 55 pyetje dhe nën-detyrën 4 në maksimum 2597 pyetje. Vlerat shembull të Q_s dhe rezultatet për nën-detyrat 3 dhe 4 janë treguar më poshtë.

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

Shembujt

Vlerësuesi	Zgjidhja
1 5	
	? 0 2 4
3	
	? 3 0 1
3	
	? 0 4 3
5	
	! 3 1 0 2 4

Shpjegimi

Inputi shembull përmban një rast testimi ($T = 1$) me $N = 5$ mysafirë. Konfigurimi i fshehur i mysafirëve në këtë rast testimi korrespondon me Figurë 1.

Pyetja e parë e bërë nga zgjidhja shembull është 0, 2, 4. Përgjigjja 3 për këtë pyetje na tregon se këta mysafirë ulen, në një renditje të panjohur, në tri karrige radhazi njëri pranë tjetrit.

Përgjigjja 3 për pyetjen e dytë na tregon të njëjtën gjë për mysafirët 3, 0 dhe 1.

Tani mund të konkludojmë se mysafiri 0 duhet të ulet në mes, me mysafirët 2 dhe 4 në njërën anë dhe mysafirët 1 dhe 3 në anën tjetër.

Pas pyetjes së tretë, ne mund të jemi të sigurt se mysafirët duhet të ulen ose në renditjen $[3, 1, 0, 2, 4]$ ose në renditjen e kundërt $[4, 2, 0, 1, 3]$. Ne mund të printojmë cilëndo nga renditjet.

Shabllonet e Kodit dhe Detajet e Vlerësimit në CMS

Ne rekomandojmë fuqimisht përdorimin e shablloneve të kodit të ofruara për C++ dhe Python. Këto kontrollojnë nëse komunikimi me vlerësuesin ishte i suksesshëm dhe përfundojnë në mënyrë të rregullt kur nuk ishte.

Nëse nuk i përdorni shabllonet e dhëna, në rastet kur zgjidhja juaj është e pasaktë, CMS mund të shfaqë përfundimin e gabuar. Për shembull, në vend të “Output isn’t correct” mund të merrni “Execution killed by signal” ose “Execution timed out (wall clock limit exceeded)”.

Ne gjithashtu rekomandojmë mjetin e testimit (shih më poshtë) për të testuar zgjidhjen tuaj lokalisht përpara se ta dorëzoni. Mjeti i testimit kontrollon rezultatet e zgjidhjes suaj dhe raporton përdorimin e pyetjeve të pavlefshme.

Mjeti i Testimit

Për të lehtësuar testimin e zgjidhjes suaj, ne ofrojmë një mjet të thjeshtë që mund ta shkarkoni nga CMS. Mjeti është opsional për t'u përdorur. Vini re se vlerësuesi zyrtar në CMS është i ndryshëm nga mjeti i testimit.

Për të përdorur mjetin, ju duhet një input file. Mund të përdorni inputin shembull të dhënë seatingplan.input0.txt ose të bëni tuajën. Input file duhet të fillojë me një rresht që ka numrin T të rasteve të testimit dhe më pas duhet të ketë dy rreshta për çdo rast testimi: një rresht me numrin N dhe më pas një rresht me numrat $g_0, g_1, \ldots, g_{N-1}$.

Për programet Python, p.sh. seatingplan.py (zakonisht ekzekutohet si pypy3 seatingplan.py) ekzekutoni mjetin e testimit si më poshtë:

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

Për programet C++, së pari përpiloni zgjidhjen tuaj:

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

dhe pastaj ekzekutoni mjetin e testimit:

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