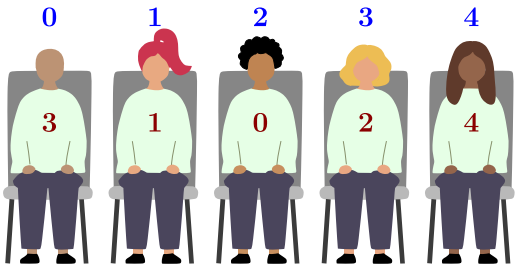
# D. План седења (seatingplan)

Завршној церемонији овог EGOI-а присуствоваће $N$ важних гостију (тј. дигиталаца). Сви они треба да седе у првом реду у једном веома специфичном распореду који одговара свим нијансама дипломатског протокола Голубира Шпијунира. Ноеми је провела две непроспаване ноћи одређујући тачан распоред седења, након чега је почела да говори РУУУУС.

Вероника надгледа завршну церемонију. Једна од њених бројних обавеза је да провери да ли су на седиштима у првом реду постављене исправне таблице са именима. Постоји само један мали проблем: Ноеми јој никада није рекла тачан распоред седења, а сада јој се губи сваки траг. Срећом, Дорка, фотографкиња, има апликацију која би могла да помогне.

Дорка је морала да припреми камере тако да може да фотографише одређене госте у првом реду. За подешавање јој је било потребно да зна колико ће свака фотографија бити широка, па јој је Ноеми направила апликацију која брзо исписује информације које су јој потребне. Вероника сада жели да искористи ту апликацију како би пронашла исправан распоред седења.

$N$ важних гостију нумерисано је бројевима од 0 до $N-1$. Седишта у првом реду такође су нумерисана бројевима од 0 до $N-1$, слева надесно. За свако $I$ ($0 \leq I \leq N-1$), нека $g_I$ означава госта који треба да седи на седишту $I$, а нека $s_I$ означава седиште на ком гост $I$ треба да седи.




Слика 1: Ред са пет гостију. За овај ред важи $g = [3, 1, 0, 2, 4]$ и $s = [2, 1, 3, 0, 4]$.

Апликација ради на следећи начин:

- Дорка уноси бројеве $I$, $J$, $K$ тачно три различита госта.
- Апликација јој каже минималан број гостију који ће бити видљиви ако су сва три изабрана госта на фотографији. Формално, апликација ће приказати вредност $\max(s_I, s_J, s_K) - \min(s_I, s_J, s_K) + 1$.

На пример, погледај ситуацију приказану на Слика 1:

- Гости $I = 0$, $J = 2$ и $K = 4$ седе на местима $s_I = 2$, $s_J = 3$ и $s_K = 4$. Ако их Дорка изабере, апликација ће приказати вредност $\max(2, 3, 4) - \min(2, 3, 4) + 1 = 3$.

  Другим речима, најужа фотографија која садржи госте 0, 2 и 4 има тачно ова три госта.

- Гости $I = 0$, $J = 4$ и $K = 3$ седе на местима $s_I = 2$, $s_J = 4$ и $s_K = 0$. Ако их Дорка изабере, апликација ће приказати вредност $\max(2, 4, 0) - \min(2, 4, 0) + 1 = 5$.

  Другим речима, фотографија која садржи ова три госта мора да садржи свих 5 гостију.

Помози Вероники да одреди исправан распоред седења користећи Доркину апликацију. Прецизније, твој програм треба да одреди и испише низ $g_0, g_1, ..., g_{N-1}$. Увек постоје тачно два исправна

одговора (један је обрнут редослед другог), и можеш исписати било који од њих. Твој резултат ће зависити од броја упита које твоје решење постави апликацији.

## Имплементација

> ⇨ Ово је интерактиван задатак. Твој програм ће користити стандардни улаз и излаз за комуникацију са системом за оцењивање у формату описаном испод.

Твој програм треба да почне читањем једног реда улаза који садржи позитиван цео број $T$, број тест примера који следе.

За сваки тест пример, твој програм треба да почне читањем једног реда улаза који садржи позитиван цео број $N$, број седишта, што је уједно и број гостију.

Да би поставио упит, твој програм треба да испише један ред облика "? $I$ $J$ $K$", где су $0 \leq I, J, K \leq N-1$ три **различита** броја.

Након постављања упита, твој програм треба да прочита један ред који садржи један позитиван цео број, одговор на твој упит.

Да би исписао исправан распоред седења, твој програм треба да испише ред облика "! $g_0$ … $g_{N-1}$".

Након што реши свих $T$ тест примера, твој програм треба нормално да се заврши.

Обрати пажњу да званични систем за оцењивање који CMS користи може бити **адаптиван**. То значи да за неке тест примере распоред гостију није унапред одређен. Уместо тога, систем може мењати своје понашање у зависности од конкретних упита које твој програм поставља.

**Флашовање.** Ако не користиш приложене шаблоне, обавезно испразни стандардни излаз након исписивања сваког реда, иначе твој програм може добити оцену *Not correct*. У Python-у се то дешава аутоматски ако користиш `input()` за читање редова. У C++-у, `cout << endl;` поред новог реда врши и флашовање; ако користиш printf, користи `fflush(stdout)`.

## Ограничења

- $1 \leq T \leq 10$.
- $N$ ће бити 5 (само пример из текста задатка), 8, 40 или 2000.
- За сваки тест пример можеш поставити највише 10 000 упита.

## Бодовање

Твој програм ће бити тестиран на више тест примера груписаних у подзадатке. Да би добио поене за подзадатак, мораш тачно решити све тест примере које он садржи.

- **Подзадатак 0 [ 0 поена]**: Пример из текста задатка($N = 5$).
- **Подзадатак 1 [ 9 поена]**: $N = 8$.
- **Подзадатак 2 [11 поена]**: $N = 2000$, и гости 0 и 1 седе један поред другог.
- **Подзадатак 3 [15 поена]**: $N = 40$.
- **Подзадатак 4 [65 поена]**: $N = 2000$.

За подзадатке 1 и 2, свако решење које тачно реши све тест примере добиће све поене.

За подзадатке 3 и 4, твоје решење мора тачно решити све тест примере да би добило било какве поене, а резултат ће зависити од $Q_s$, највећег броја упита које је твоје решење морало да постави да би решило неки тест пример. Нека је $X_s = \max(1, Q_s/N)$. Резултати за подзадатке 3 и 4 рачунају се на следећи начин:

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

Вредност $\text{score}_s$ се заокружује на најближи цео број за сваки подзадатак, а укупан резултат је њихов збир. Да би добио максималан број поена, потребно је да решиш подзадатак 3 са највише 55 упита и подзадатак 4 са највише 2597 упита. Примери вредности $Q_s$ и резултата за подзадатке 3 и 4 приказани су испод.

| $Q_s$ | 55 | 56 | 60 | 70 | 80 | 100 | 150 | 10000 |
|---|---|---|---|---|---|---|---|---|
| $\text{score}_3$ | 15 | 14 | 13 | 11 | 10 | 8 | 6 | 3 |

| $Q_s$ | 2597 | 2800 | 3000 | 4000 | 5000 | 6000 | 8000 | 10000 |
|---|---|---|---|---|---|---|---|---|
| $\text{score}_4$ | 65 | 58 | 53 | 35 | 26 | 21 | 14 | 11 |

## Примери улаза/излаза

| Грејдер | Решење |
|---|---|
| 1<br>5 | |
| | ? 0 2 4 |
| 3 | |
| | ? 3 0 1 |
| 3 | |
| | ? 0 4 3 |
| 5 | |
| | ! 3 1 0 2 4 |

## Објашњење

Пример улаза садржи један тест пример ($T = 1$) са $N = 5$ гостију. Скривени распоред гостију у овом тест примеру одговара Слика 1.

Први упит који пример решења поставља је 0, 2, 4. Одговор 3 на овај упит нам говори да ови гости седе, у неком непознатом редоследу, на три узастопна седишта једно поред другог.

Одговор 3 на други упит нам говори исто за госте 3, 0 и 1.

Сада можемо закључити да гост 0 мора седети у средини, са гостима 2 и 4 са једне стране и гостима 1 и 3 са друге стране.

Након трећег упита, већ можемо бити сигурни да гости морају седети или редом $[3, 1, 0, 2, 4]$ или обрнутим редом $[4, 2, 0, 1, 3]$. Можемо исписати било који од та два распореда.

## Детаљи оцењивања у CMS-у

Ако твоје решење прекрши протокол (на пример, постави неважећи упит или премаши дозвољени број питања), систем за оцењивање који комуницира са твојим решењем пријавиће грешку и прекинути рад, али твоје решење можда неће. Прекид рада система за оцењивање може довести до тога да се твој програм сруши или настави да чека одговор, па уместо очекиване поруке "Output isn't correct" можеш добити друге поруке као што су "Execution timed out (wall clock limit exceeded)", јер се пад програма или прекорачење временског ограничења пријављује раније. Имај то на уму када читаш и тумачиш CMS поруке.

Топло препоручујемо да користиш приложени алат за тестирање (погледај испод) како би тестирао своје решење локално пре слања. Излаз алата за тестирање биће кориснији од CMS порука. Конкретно, алат за тестирање проверава излаз твог решења и пријављује кршења протокола.

## Алат за тестирање

Да бисмо олакшали тестирање твог решења, обезбедили смо једноставан алат који можеш преузети са CMS-а. Коришћење алата није обавезно. Обрати пажњу да се званични систем за оцењивање на CMS-у разликује од алата за тестирање.

Да би користио алат, потребан ти је улазни фајл. Можеш користити приложени пример улаза `seatingplan.input0.txt` или направити свој. Улазни фајл треба да почиње редом који садржи број $T$ тест примера, а затим треба да има по два реда за сваки тест пример: један ред са бројем $N$ и један ред са бројевима $g_0, g_1, ..., g_{N-1}$.

За Python програме, рецимо `seatingplan.py` (који се иначе покреће као `pypy3 seatingplan.py`) покрени алат за тестирање овако:

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

За C++ програме, прво компајлирај своје решење:

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

а затим покрени алат за тестирање:

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