D. Uhádni, kde bývam!

Problem Name	Guessing Game
Time Limit	4 seconds
Memory Limit	1 gigabyte

V historickom centre Lundu je ulica, pozdĺž ktorej stojí N domov v rade vedľa seba. Tieto domy majú za radom popisné čísla od 0 po $N-1$. Emma býva v jednom z týchto domov.

Jej kamaráti Anna a Bertil chcú zistiť, kde Emma býva. Emma im to však nechce povedať. Namiesto toho sa s nimi chce zahrať hru.

Pred samotným začiatkom hry Anna a Bertil poznajú len číslo N a nič iné. Skôr, ako hra začne, môžu si zvoliť celé číslo K a dohodnúť sa na spoločnej stratégii. Akonáhle hra začne, už spolu nesmú komunikovať inak ako hraním samotnej hry.

Hra sa skladá z dvoch fáz.

V prvej fáze si Emma zvolí poradie, v ktorom postupne navštívi domy na ulici tak, aby svoj vlastný dom navštívila posledný. Potom zoberie Annu so sebou a postupne ju vo zvolenom poradí privedie do všetkých domov. (Anna sa poradie nedozvie celé dopredu, len postupne chodí za Emmou a navštevuje domy.)

Pri každej návšteve domu iného ako toho, kde býva Emma, môže Anna napísať na dvere kriedou číslo z rozsahu od 1 po K. Keď nakoniec prídu k domu Emmy, tá si zoberie kriedu a na svoj dom si číslo z toho istého rozsahu napíše sama.

Nasleduje druhá fáza hry. V tejto fáze sa Bertil prejde po ulici a prečíta si čísla napísané Annou a Emmou na všetkých domoch v poradí od domu 0 k domu $N-1$. Bertil teraz musí uhádnuť, kde Emma býva. Môže si tipnúť dvakrát. Ak je ľubovoľný z jeho dvoch tipov správny, Anna a Bertil hru vyhrali, ak sú oba zlé, vyhrala Emma.

Tvojou úlohou je naprogramovať stratégie pre Annu a Bertila tak, aby zaručene vždy túto hru vyhrali. Počet bodov, ktorý získaš, závisí od čísla K, ktoré si zvolíš: čím menšie K ti stačí, tým lepšie.

Implementation

Musíš implementovať obe stratégie: aj Anninu, aj Bertilovu. Program, ktorý napíšeš, bude postupne spustený viackrát. Pri každom spustení mu bude oznámené, ktorú jednu stratégiu má práve vykonávať.

V prvom riadku vstupu budú vždy dve celé čísla: P a N. Číslo P (phase) udáva fázu hry: $P = 1$ je Anna, $P = 2$ je Bertil. Číslo N udáva počet domov.

Až na ukážkový test (ktorý sa pri bodovaní úlohy ignoruje) **vo všetkých testovacích vstupoch je N rovné 100 000.**

Zvyšok vstupu závisí na P, teda na tom, ktorá fáza hry sa práve hrá.

Fáza 1 (Anna)

Tvoj program by mal začať tým, že vypíše jeden riadok so zvoleným číslom K. Pre toto číslo musí platiť $1 \leq K \leq 1\,000\,000$.

Potom prebehne $N - 1$ kôl. V každom kole tvoj program najskôr načíta riadok s číslom domu i (platí $0 \leq i < N$). Následne tvoj program vypíše riadok s celým číslom A_i $(1 \leq A_i \leq K)$, ktoré chceš na dom i napísať kriedou.

Každé číslo domu okrem Emminho sa na vstupe tvojho programu objaví práve raz. Poradie, v ktorom ti budú čísla domov dané, si zvolí grader.

Fáza 2 (Bertil)

Tvoj program si zo vstupu prečíta riadok, v ktorom je N celých čísel: $A_0, A_1, \ldots, A_{N-1}$. Číslo A_i je číslo, ktoré Bertil vidí napísané kriedou na dome číslo i.

Tvoj program by následne mal vypísať riadok s dvomi medzerou oddelenými celými číslami s_1 a s_2 (kde $0 \leq s_i < N$): Bertilove dva tipy pre číslo domu, v ktorom býva Emma. (Je povolené vypísať dvakrát to isté číslo.)

Technické detaily implementácie

Pre každú fázu tvoj program spustíme úplne nanovo, aby si si nemala ako nič pamätať v pomocných premenných ani nikde inde.

Daj si pozor, aby si flushla výstup po každom riadku, ktorý vypíšeš. Ak tak neurobíš, môžeš dostať verdikt Time Limit Exceeded. V Pythone by malo stačiť používať `print()`, ktorý flushuje automaticky. V C++ `cout << endl;` po vypísaní znaku nového riadku aj flushne výstup. Ak používaš `printf`, po ňom sprav `fflush(stdout)`.

Grader pre túto úlohu **môže byť adaptívny**. Toto znamená, že správanie gradera sa môže meniť v závislosti od výstupu tvojho programu. Grader sa aktívne bude snažiť, aby rôzne heuristické

riešenia, ktoré nie sú vždy správne, dostali vstupy, ktoré budú pre ne ťažké.

Navyše pri testovaní tvojho riešenia môže testovač „cvične“ spustiť jeho prvú fázu, pozrieť si jej výstup, a potom znova spustiť jeho prvú fázu „naostro“ a pri tomto behu používať informácie, ktoré zistil z cvičného prvého behu.

Tvoj program musí byť deterministický. (Ak ho dvakrát spustíme na tom istom vstupe, musí dať ten istý výstup.)

Vo svojom programe ešte stále môžeš, ak chceš, používať pseudonáhodné čísla. Daj si ale pozor, aby si generátor, ktorý používaš, inicializovala nejakým pevne zvoleným seedom. Napr. v C++ môžeš zavolať `srand` alebo v Pythone zavolať `random.seed` tak, že ako parameter použiješ konkrétne celé číslo. (Nesmieš teda použiť napr. `srand(time(NULL))`.)

Ak sa tvoj program počas testovania nebude správať deterministicky, dostaneš verdikt Wrong Answer.

Tvoj program môže byť postupne spustený až trikrát (cvičná prvá fáza, ostrá prvá fáza, druhá fáza). Ak súčet časov behu za všetky tieto spustenia dokopy prekročí časový limit, dostaneš verdikt Time Limit Exceeded.

Scoring

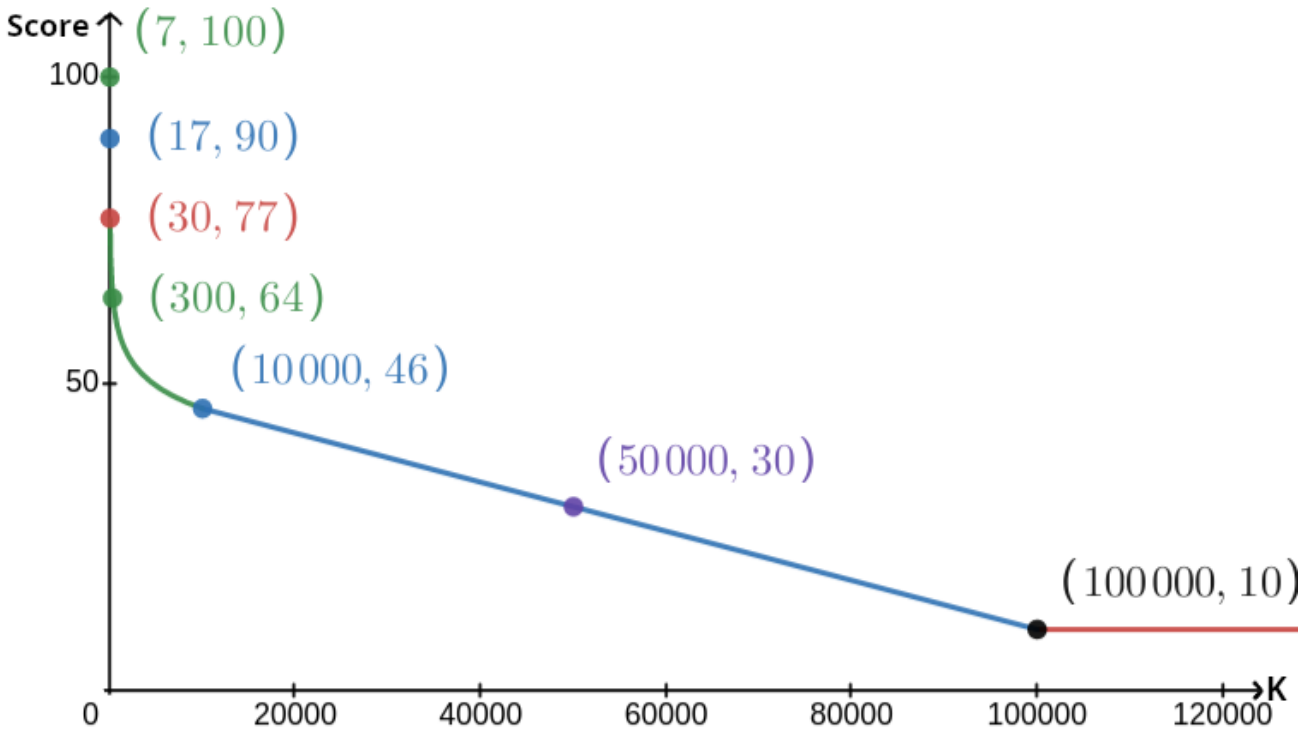
Tvoje riešenie bude postupne testované na viacerých testoch. Ak čo i len jeden z nich nevyrieši správne, dostaneš príslušný negatívny verdikt a nula bodov.

Ak správne vyriešiš všetky testy, dostaneš verdikt Accepted a tvoje skóre za túto úlohu bude závisieť od K, ktoré tvoj program použil. (Keďže všetky oficiálne testy majú to isté N, tvoj deterministický program musí vo všetkých použiť to isté K.)

Skóre sa podľa tvojho K určí nasledovne:

	Score
$K > 99\,998$	10 bodov
$10\,000 < K \leq 99\,998$	$10 + \lfloor 40(1 - K/10^5) \rfloor$ bodov
$30 < K \leq 10\,000$	$46 + \lfloor 31(4 - \log_{10}(K))/(4 - \log_{10}(30)) \rfloor$ bodov
$7 < K \leq 30$	$107 - K$ bodov
$K \leq 7$	100 bodov

Graf tejto funkcie je na nasledujúcom obrázku:



Pripomíname, že ukážkový test sa pri určovaní tvojho skóre ignoruje. Tvoje riešenie ho dokonca ani len nemusí správne vyriešiť.

Testing Tool

Z Kattisu si môžeš stiahnuť testing tool k tejto úlohe. Nájdeš ho na spodku stránky pre túto úlohu (v sekcii Attachments).

Používanie tohto nástroja je dobrovoľné. Ak chceš, môžeš si ho ľubovoľne upraviť. (Pri skutočnom testovaní odovzdaného programu v Kattise sa používa iný testovač ako ten, čo ste dostali vy.)

Príklad použitia (pre $N = 4$ domy, pričom na Emmin dom chceme kriedou napísať číslo $s = 2$):

Ak programuješ v Pythone a tvoj program je v súbore `solution.py` (normálne by si ho spúšťala príkazom `pypy3 solution.py`), spusti:

```
python3 testing_tool.py pypy3 solution.py <<< "4 2"
```

Ak programuješ v C++, svoj program najskôr skompiluj

(napr. príkazom `g++ -g -O2 -std=gnu++17 -static solution.cpp -o solution.out`)

a potom spusti:

```
python3 testing_tool.py ./solution.out <<< "4 2"
```

Testing tool, ktorý dostaneš, navštevuje domy v náhodnom poradí. Ak chceš namiesto toho použiť nejaké iné vlastné poradie, môžeš v testing toole príslušne upraviť časť označenú “MODIFY HERE”.

Example Interaction

Ešte raz pripomíname, že tento ukážkový test sa pri určovaní tvojho skóre ignoruje. Tvoje riešenie ho dokonca ani len nemusí správne vyriešiť.

Majme $N = 4$ a nech Emma býva v dome 1. Nech A je pole obsahujúce čísla napísané kriedou na domoch, pričom používame 0 pre domy, ktoré ešte nemajú kriedou napísané číslo. Na začiatku teda máme $A = [0, 0, 0, 0]$.

Pri prvom spustení tvoj program spustíme vo fáze 1 a oznámime mu $N = 4$.

Tvoj program odpovie voľbou $K = 3$.

Testovač sa opýta na dom 2. Tvoje riešenie si zvolí číslo $A_2 = 3$. A je teraz $[0, 0, 3, 0]$.

Testovač sa opýta na dom 0. Tvoje riešenie si zvolí číslo $A_0 = 1$. A je teraz $[1, 0, 3, 0]$.

Testovač sa opýta na dom 3. Tvoje riešenie si zvolí číslo $A_3 = 2$. A je teraz $[1, 0, 3, 2]$.

Už ostal len Emmin dom. Pre ten si testovač zvolí $A_1 = 2$, čím dostane výsledné pole A = $[1, 2, 3, 2]$.

Pri druhom spustení tvoj program spustíme vo fáze 2 a znova mu oznámime $N = 4$.

Následne mu oznámime postupnosť `1 2 3 2`.

Tvoj program odpovie tipmi `1 3`.

Keďže jeden z nich je správnym indexom Emminho domu (1), tvoj program hrajúci postupne za Annu a Bertila hru vyhral.

grader output	your output
1 4	
	3
2	
	3
0	
	1
3	
	2

grader output	your output
2 4	
1 2 3 2	
	1 3