Twin Cookies

Problem name	Twin Cookies
Input file	standard input
Output file	standard output
Time limit	1 second
Memory limit	256 megabytes

Aceasta este o problema interactiva. Programul tau va interactiona cu graderul prin afisarea de mesaje la standard output si citirea mesajelor de la standard input

Sophie pregateste petrecerea de ziua gemenilor ei. Gemenii iubesc biscuitii. De ziua lor, ei ar vrea sa incerce ceva nou: biscuitii de la Unique Cookie Tastiness Company (UCTC).

Fiecare biscuite produs de UCTC are asociata o valoare cuprinsa intre 1 si 10^{16} inclusiv. Cum gemenii lui Sophie devin gelosi unul pe celalalt, ei trebuie sa primeasca biscuiti care au sumele valorilor egale.

UTCT accepta doar comenzi de **exact** n biscuiti. Pentru fiecare comanda, clientul specifica valoarea fiecaruia dintre cei n biscuiti.

Unique Cookie Tastiness Company refuza sa produca doi biscuiti de aceeasi valoare pentru acelasi client. Sophie trebuie sa se asigure ca nu comanda niciodata biscuiti de aceeasi valoare -- nici in aceeasi comanda si nici in comenzi diferite. Sophie nu a mai cumparat niciodata de la UCTC, deci poate comanda biscuiti de fiecare valoare disponibila.

In calea lui Sophie se mai afla inca un obstacol: este bine cunoscut faptul ca serviciul de livrare al UTCT este groaznic. De fiecare data cand un client comanda n biscuiti, doar unul dintre cei n biscuiti va ajunge la client. Restul vor fi mancati pe drum de angajatii serviciului de livrare. Clientul nu poate alege care dintre cei n biscuiti va fi livrat.

Cum ziua de nastere se apropie rapid, Sophie are timp sa faca cel mult 101 comenzi. Sarcina ta este sa o ajuti.

Mai specific, trebuie sa faci urmatoarele:

1. Mai intai, trebuie sa comanzi biscuitii. Poti sa faci cel mult 101 comenzi, fiecare formata din exact n valori. Poti face o singura comanda la un moment de timp. **Imediat dupa fiecare comanda ti se va da valoarea biscuitelui pe care l-ai primit.**

   Tine minte ca nu ai voie sa folosesti aceeasi valoare de mai multe ori, nici macar in comenzi diferite. (In particular, daca ai comandat un biscuite cu valoarea t care nu este livrat, **nu poti** sa mai comanzi un biscuite cu aceeasi valoare).

2. Apoi, imparte biscuitii. Odata ce ai primit suficienti biscuiti, trebuie sa distribui **cativa** dintre acesti biscuiti primiti intre gemeni. Fiecare geaman trebuie sa primeasca cel putin un biscuite si fiecare dintre ei trebuie sa primeasca biscuiti de aceeasi valoare totala. **Nu trebuie folositi toti biscuitii primiti!**

Output

De fiecare data cand programul tau afiseaza una sau mai multe linii la standard output, aceasta actiune trebuie urmata prin a face **flush la output stream**. Acest lucru este necesar pentru a asigura ca datele ajung la grader imediat.

De exemplu:

- in C++, se poate folosi `fflush(stdout)` sau `cout << flush`
- in Java, se poate folosi `System.out.flush()`
- in Python, se poate folosi `sys.stdout.flush()`

Interaction Protocol

Programul tau trebuie sa efectueze urmatoarea secventa de operatii:

1. Citeste valoarea n de la standard input.
2. De cel mult 101 de ori:
   1. Mai intai, afiseaza la standard output o linie care descrie o comanda de n biscuiti.
   2. Apoi, afiseaza valoarea biscuitelui primit. Se garanteaza ca aceasta valoare se afla intre cele n din comanda respectiva.
3. Afiseaza trei linii descriind un mod valid de a impartii intre gemeni unii dintre biscuitii primiti.

Graderul va afisa fiecare intreg pe cate o linie separata.

Pentru a comanda biscuiti, afiseaza o singura linie formata din '?' urmat de n numere intregi: valorile biscuitilor pe care vrei sa ii comanzi. Afiseaza cate un singur spatiu inainte de fiecare dintre cele n numere.

Tine minte ca poti sa faci cel mult 101 comenzi si ca nu este permis sa folosesti aceeasi

valoare de doua ori.

Odata ce ai comandat suficienti biscuiti, afiseaza cele 3 linii care descriu biscuitii pe care Sophie ar trebui sa ii dea gemenilor.

Prima dintre aceste linii trebuie sa fie de forma "! m k" cu $m, k > 0$: numarul de biscuiti pe care ii va primi primul, respectiv al doilea geaman.

A doua linie va contine m numere intregi separate prin cate un spatiu reprezentand valorile biscuitilor pe care ii va primi primul geaman.

Similar, a treia linie va contine k numere intregi separate prin cate un spatiu: valorile biscuitilor pe care ii va primi al doilea geaman.

Outputul trebuie sa indeplineasca urmatoarele conditii:

1. Fiecare geaman trebuie sa primeasca cel putin un biscuite.
2. Fiecare geaman trebuie sa primeasca biscuiti care au aceeasi suma a valorilor.
3. Doar biscuitii care au fost livrati dupa ce ai facut comenzi pot fi folositi.
4. Fiecare dintre acei biscuiti poate fi dat cel mult unuia dintre gemeni.

Orice output care satisface aceste conditii va fi acceptat. In particular, poti afisa biscuitii alesi in orice ordine.

Dupa afisarea ultimelor trei linii, trebuie sa faci flush la output stream pentru o ultima oara si lasa programul sa se termine normal.

Scoring

Subtask 1 (8 puncte): $n = 1$

Subtask 2 (9 puncte): $1 \leq n \leq 2$

Subtask 3 (18 puncte): $1 \leq n \leq 25$

Subtask 4 (16 puncte): $1 \leq n \leq 200$

Subtask 5 (13 puncte): $1 \leq n \leq 1000$

Subtask 6 (36 de puncte): $1 \leq n \leq 5000$

Examples

standard input	standard output
1 13 7 31 12 5 3	? 13 ? 7 ? 31 ? 12 ? 5 ? 3 ! 2 3 7 13 12 5 3
2 7 2 5	? 3 7 ? 2 8 ? 1 5 ! 2 1 2 5 7

Note

Exemplele de input si output trebuie citite rand cu rand. Programul tau citeste cate o valoare de la standard input si afiseaza o linie (sau trei linii la final) la standard output alternativ.

Graderul alege biscuitele pe care il va returna arbitrar. Aceasta inseamna ca graderul isi adapteaza raspunsurile in functie de intrebarile tale, dar poate si sa aleaga aleator un biscuite in altele. De exemplu, pentru $n = 2$, daca faci aceeasi secventa de comenzi ca in al doilea exemplu, poti primi o multime diferita de biscuiti.