Problem A. Addition

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Étant donné deux nombres a et b, calcule $a + b$

Input

L'entrée consiste en deux lignes. Sur la première se trouve un entier a, et la deuxième un entier b $(-100 \leq a, b \leq 100)$

Output

Imprime une ligne avec la valeur $a + b$.

Examples

standard input	standard output
7 5	12
-10 3	-7

Problem B. Autocollants

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

En regardant sur ton bureau tu te rends rends compte que tu as a autocollants rouges et b autocollants verts.

Ton but, est de déterminer si tu as plus d'autocollants rouges que verts.

- Si tu as plus d'autocollants rouge que vert, imprime $>$
- Si tu as moins d'autocollants rouge que vert, imprime $<$
- Si tu as autant d'autocollants rouge que vert, imprime $=$

Input

L'entrée consiste en deux lignes. Sur la première se trouve un entier a, et la deuxième un entier b $(0 \leq a, b \leq 10^9)$ — le nombre d'autocollants rouges et verts respectivement.

Output

Imprime une ligne soit $>$ soit $<$ soit $=$ selon si tu as plus, moins, ou autant, d'autocollants rouges que verts.

Examples

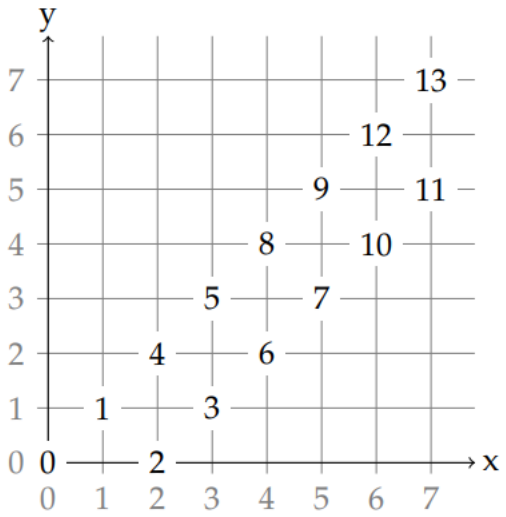
standard input	standard output
6 3	>
453 453	=

Problem C. La Grande Grenouille Grégoire

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

La Grande Grenouille Grégoire a inventé une promenade avec un motif de sauts étrange. Ton amie Camille est très impressionnée. Ci-dessous tu peux voir la positions des sauts de Grégoire dans le plan.

Elle commence avec le saut 0 au point $(0, 0)$, continue avec le saut 1 au point $(1, 1)$, ensuite le saut 2 au point $(2, 0)$ et ainsi de suite. La Grande Grenouille Grégoire est tellement fière de sa promenade qu'elle va la continuer à l'infini (sur l'image sont montrés seulement quelques premiers pas).



Camille veut déterminer si Grégoire arrive au point (x, y) et, si oui, après combien de sauts.

Peux-tu l'aider?

Input

La seule ligne d'entrée contient deux nombres x et y $(0 \le x, y \le 10^9)$ — les coordonnées du point (x, y).

Output

Écris le numéro du pas du point donné ou écris "JAMAIS"si Grégoire ne marche jamais sur ce point.

Examples

standard input	standard output
4 2	6
3 4	JAMAIS

Problem D. Multi-Adder

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Ton but est de résoudre le problème addition pour t tests.

Input

La première ligne consiste en un entier $t(1 \leq t \leq 100)$ — le nombre de tests.

Les t lignes suivantes consisent en deux entier a et b séparé par un espace. $(-10^9 \leq a, b \leq 10^9)$ – les deux entiers à additioner.

Output

Imprime t lignes. Sur la i-ème affiche l'addition de a et b dans le i-ème test.

Example

standard input	standard output
5 1 1 10 33 -15 -49 100000 200000 -214234 -92855	2 43 -64 300000 -307089

Problem E. FizzBuzz

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Ecris un programme qui imprime les nombres de 1 à N sur des lignes séparées. Mais pour les multiples de 3, écris "Fizz"à la place du nombre et pour les multiples de 5 imprime "Buzz".
Pour les nombres multiples à la fois de 3 et 5, imprime "FizzBuzz".

Input

Il y a une seule ligne d'entrée contenant un entier $N(1 \leq N \leq 2 \cdot 10^5)$

Output

Les N lignes comme décrites ci-dessus.

Example

standard input	standard output
15	1 2 Fizz 4 Buzz Fizz 7 8 Fizz Buzz 11 Fizz 13 14 FizzBuzz

Problem F. Suite de Syracuse

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

La suite de Syracuse (parfois référée en tant que suite de collatz) pour un nombre naturel donné est définie comme suit.

— si n est pair, passe à $n/2$

— si n est impair, passe à $3 \cdot n + 1$

Par exemple, si nous commençons par 7, la suite serait $7 \to 22 \to 11 \to 34 \to 17 \to 52 \to 26 \to 13 \to 40 \to 20 \to 10 \to 5 \to 16 \to 8 \to 4 \to 2 \to 1 \to 4$ ...

Une conjecture ouverte (non prouvée) affirme que cette suite se termine toujours par un cycle 4, 2, 1.

Pour un nombre naturel donné n, combien d'étapes sont-elles nécessaires avant d'arriver au nombre 1 ?

Input

Il y a une seule ligne d'entrée contenant un entier n ($1 \leq n \leq 10^6$) — le nombre de la suite de Syracuse

Output

Imprime un entier, le noombre d'étapes nécessaires avant d'arriver à 1.

Example

standard input	standard output
3	7

Problem G. Devine le nombre

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Il s'agit d'un problème interactif. Tu dois utiliser une opération `flush` *juste après avoir imprimé chaque ligne. Par exemple, en C++, tu devrais utiliser la fonction* `fflush(stdout)`*, en Java —* `System.out.flush()`*, en Pascal —* `flush(output)` *et en Python —* `sys.stdout.flush()`*.*

Dans ce problème, le jury a un certain nombre x, et tu dois le deviner. Le nombre x est toujours un entier entre 1 et n, où n te est donné au début.

Tu peux poser des questions au système de test. Chaque question est un seul entier entre 1 et n. Vide le flux de sortie après avoir imprimé chaque question. Le programme de test peut fournir deux réponses différentes :

- la chaîne de caractères "`<`" (sans guillemets), si le nombre du jury est inférieur à l'entier dans ta question ;
- la chaîne de caractères "`>=`" (sans guillemets), si le nombre du jury est supérieur ou égal à l'entier dans ta question.

Lorsque ton programme aura deviné le nombre x, imprime la chaîne de caractères "`! x`", où x est la réponse, et **termine ton programme normalement** immédiatement après avoir vidé le flux de sortie.

Ton programme est autorisé à poser au plus 25 questions (sans inclure l'impression de la réponse) au système de test.

Input

Utilise l'entrée standard pour lire les réponses aux questions.

La première ligne contient un entier n ($1 \le n \le 10^6$) — le nombre maximum possible du jury.

Les lignes suivantes contiendront les réponses à tes questions — les chaînes de caractères "`<`" ou "`>=`". La i-ème ligne est la réponse à ta i-ème question. Lorsque ton programme devinera le nombre, imprime "`! x`", où x est la réponse, et termine ton programme.

Le système de test te permettra de lire la réponse à la question uniquement après que ton programme aura imprimé la question pour le système et effectué l'opération de `flush`.

Output

Pour poser les questions, ton programme doit utiliser la sortie standard.

Ton programme doit imprimer les questions — des nombres entiers x_i ($1 \le x_i \le n$), une question par ligne (n'oublie pas la "*fin de ligne*" après chaque x_i). Après avoir imprimé chaque ligne, ton programme doit effectuer l'opération `flush`.

Chacune des valeurs x_i représente une question pour le système de test. La réponse à la question sera donnée dans le fichier d'entrée après avoir vidé la sortie. Si ton programme devine le nombre x, imprime la chaîne de caractères "`! x`", où x est la réponse, et termine ton programme.

Example

standard input	standard output
20 ~ < ~ >= ~ >= ~	~ 5 ~ 3 ~ 4 ~ ! 4