"First, solve the problem. Then, write the code" John Johnson.

Programa-Me 2014
Regional de Madrid y Reus
Problemas

Ejercicios realizados por

Universidad Complutense
de Madrid

I.E.S. El Cañaveral
(Móstoles)

Realizado en el **I.E.S. El Lago (Madrid)** y en el **I.S. Baix Camp (Reus)**
19 de marzo de 2014

19 de marzo de 2014
http://www.programa-me.com

Listado de problemas

Autores de los problemas:

- Marco Antonio Gómez Martín (Universidad Complutense de Madrid)
- Pedro Pablo Gómez Martín (Universidad Complutense de Madrid)
- Patricia Díaz García (I.E.S. El Cañaveral - Móstoles)

Revisores:

- Ferran Borrell Micola (I.S. Baix Camp - Reus)
- Cristina Gómez Alonso (I.S. Baix Camp - Reus)

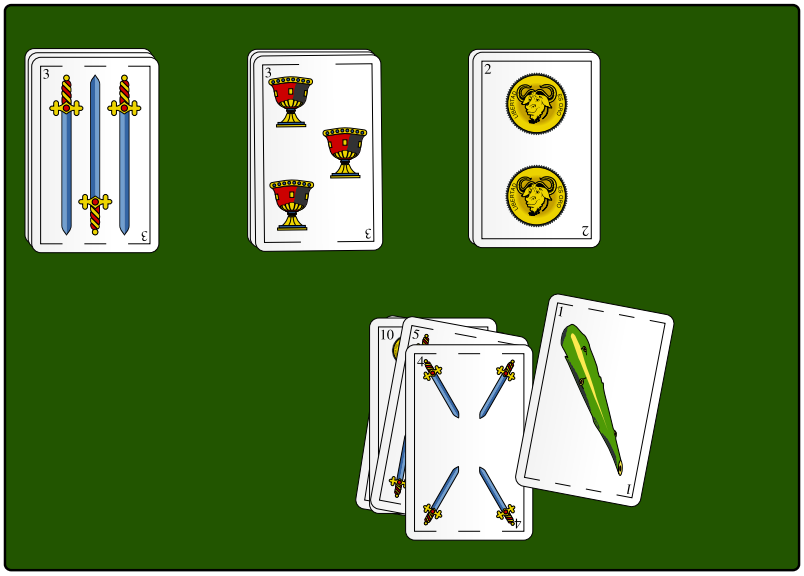
A
Solitario

Maneras de jugar al solitario hay tantas como tipos de barajas y países donde se juega con ellas. Siendo muy pequeña, mi abuelo materno me enseñó a jugar a una variedad del solitario muy sencilla utilizando la baraja española. Como en la mayoría de los solitarios, el objetivo es crear cuatro pilas de cartas, una para cada palo, en orden ascendente, es decir del as al rey.

En la modalidad de mi abuelo, se baraja el mazo de cartas y se coloca boca abajo sobre la mano. En cada jugada, se cogen simultáneamente *dos* cartas de ese mazo y se dan la vuelta sobre la mesa, formando una pila de cartas visibles. De esa pila, sólo podrá retirarse en cada momento la carta situada más arriba.

Durante la partida, se construyen también cuatro pilas de cartas, una para cada palo. Para eso, es necesario primero tener la suerte de que el as de cada palo quede en la parte de arriba de las cartas según van siendo llevadas desde el mazo de la mano a la mesa. Cuando eso ocurre, el as se retira y se coloca iniciando su propia pila, sobre la que podrá luego colocarse el dos de ese palo, luego el tres, etcétera, siempre que queden visibles en el mazo de cartas descubiertas.

Cuando se retira la carta de la parte superior del mazo de cartas descubiertas, la que queda inmediatamente debajo pasa a quedar visible; si es posible colocarla en alguna de las pilas de los palos que se están construyendo, deberá ser colocada también; el proceso continuará hasta que no queden cartas visibles, o la superior no pueda ser colocada.



Por ejemplo, imagina que, tras varias cartas descubiertas, hemos conseguido iniciar las pilas de los palos espadas, copas y oros colocando varias de sus cartas, llegando a la situación de la figura. En la penúltima jugada, extragimos del mazo de cartas boca abajo que tenemos en la mano la sota de oros y el cinco de espadas, que no pudieron ser colocadas. En el último hemos tenido más suerte al sacar el cuatro de espadas y el as de bastos. Éste queda en la parte superior y podemos colocarlo sobre la mesa, iniciando su propia pila. Además, al retirar el as, queda al descubierto el cuatro de espadas, que también colocamos inmediatamente (sobre el tres), y éste a su vez deja visible al cinco, que también extraemos. La sota de oros, que pasa a ser la superior, no se puede colocar, por lo que llega el momento de probar suerte descubriendo dos cartas nuevas del mazo de la mano.

Es importante darse cuenta de que si en lugar del as de bastos hubiera quedado en la parte superior, por ejemplo, el dos de bastos, al no poderlo retirar no podríamos tampoco colocar el cuatro de espadas de debajo.

Lo habitual es que tras llevar todas las cartas por parejas del mazo boca abajo a la mesa, no hayamos sido capaces de construir completas las cuatro pilas de cartas de cada palo. En ese caso, recogemos, sin

barajar, las cartas visibles que no hemos colocado y les damos la vuelta, de modo que la última carta que extragimos se convierte en la inferior del mazo boca abajo. Después, repetimos el proceso sacando las cartas de dos en dos otra vez. Es posible que al terminar de descubrir todas las cartas nos quede una única carta en la mano, y no dos. En ese caso, se colocará esta última sobre la pila de cartas visibles y se comprobará si se puede colocar siguiendo el procedimiento habitual.

El jugador gana la partida si consigue colocar todas las cartas en las pilas de los palos, y pierde en caso contrario, lo que ocurre cuando da toda una vuelta a las cartas pendientes sin haber sido capaz de colocar ninguna.

Entrada

La entrada consta de una serie de casos de prueba terminados con un `0`. Cada caso de prueba consiste en una configuración de baraja formada por uno, dos, tres o cuatro palos; es decir, 10, 20, 30 o 40 cartas. Cada palo consta de 10 cartas con valores desde el `1` (As) hasta el `7`, `10` (sota), `11` (caballo) y `12` (rey).

El caso de prueba comenzará por un número que indica el número de palos con el que jugamos el solitario. En la línea siguiente aparecerá la configuración del mazo de cartas inicial, ya barajado, como una secuencia de cartas. El mazo se considera colocado boca abajo, de manera que la primera carta que aparece será la primera que tengamos levantar.

Cada carta se representa con un número y un carácter, que indican el valor de la carta y el palo al que pertenece respectivamente. Se utilizará `O` para los oros, `C` para las copas, `E` para las espadas y `B` para los bastos. Entre el número y el palo aparecerá un espacio, al igual que entre una carta y la siguiente.

Salida

Para cada caso de prueba se mostrará una línea en la que se escribirá `GANA` si un jugador del solitario que comience con la configuración del caso de prueba acabará ganando la partida. En caso contrario se escribirá `PIERDE`.

Entrada de ejemplo

```
1
1 B 2 B 3 B 4 B 5 B 6 B 7 B 10 B 11 B 12 B
1
12 E 2 E 3 E 7 E 11 E 10 E 6 E 5 E 4 E 1 E
2
2 O 1 O 2 C 1 C 4 O 3 O 4 C 3 C 12 O 11 O 12 C 11 C 10 O 7 O 10 C 7 C 6 O 5 O 6 C 5 C
0
```

Salida de ejemplo

```
PIERDE
GANA
GANA
```

B
Encadenando palabras

A Samuel y a Clara les encanta jugar a encadenar palabras. Si Samuel dice *Mata*, Clara sigue diciendo *Tapa*. Samuel le sigue el juego diciendo *Papa* y Clara remata diciendo *Pato*.

Normalmente no tarda mucho en estallar la discusión cuando alguno piensa que el otro lo ha hecho mal. En realidad Samuel acaba de aprender a leer y a Clara todavía le queda un poco para empezar... así que es normal que tengan conflictos, pero lo cierto es que sus padres acaban cansados de tantas discusiones.

¿Puedes hacer un programa que les diga a Samuel y a Clara si su lista de palabras encadenadas está bien? No te preocupes por la existencia o inexistencia de las palabras que usan, de eso seguirán ocupándose sus sufridos padres.

Entrada

La entrada consta de un conjunto de casos de prueba, cada uno formado por una serie de entre 1 y 50 palabras en una misma línea. Cada palabra, de un mínimo de 2 caracteres y un máximo de 24, está separada de la siguiente mediante un espacio. Clara y Samuel no tienen aún demasiado vocabulario, por lo que podemos asegurar que las palabras que utilizan están formadas por sílabas formadas por dos letras.

Salida

Para cada caso de prueba se escribirá una línea que mostrará un `SI` si todas las palabras de la serie están correctamente encadenadas, y un `NO` en caso contrario.

Se considera que dos palabras están encadenadas si la última sílaba de la primera palabra es igual que la primera de la segunda. Para las palabras que tienen una única sílaba, ésta se considera simultáneamente la primera y la última.

Nos interesa que los niños aprendan ortografía, así que exigiremos que no sólo el sonido sea igual, sino que la grafía también lo sea. No obstante, todas las palabras se escribirán en minúscula y no tendrán vocales con tilde u otros símbolos no pertenecientes al alfabeto inglés.

Entrada de ejemplo

```
gugutata
mata tapa papa pato
seto taco coma matute
sien encima mapa patuco comida
cata tasama malote tejaba batama
kiosko comida
```

Salida de ejemplo

```
SI
SI
NO
SI
SI
NO
```

C
Embarque en un transatlántico

Los transatlánticos son buques gigantes con infinidad de *cubiertas* ("pisos") con camarotes, que pueden transportar simultáneamente a un ingente número de viajeros. Esto hace que se necesite una gran cantidad de tiempo hasta que todos los tripulantes embarcan, organizándose grandes líos en los pasillos interiores.

Queen Mary 2 (fotografía de Brian Burnell)

Para intentar paliar el problema, los responsables de la empresa *Cruz Ana Do* han decidido que, independientemente de cómo se sitúen los pasajeros en la cola para embarcar, los irán llamando por *cubiertas*.

Por organizarse, cada vez que hagan embarcar a los tripulantes de una cubierta, los organizadores desean saber cuánta gente queda aún en el puerto esperando. También están interesados en saber, algunas veces, a qué cubierta va alguno de los pasajeros de la cola.

Entrada

La entrada consta de una serie de casos de prueba. Cada uno comienza con una línea con un único número que indica la cantidad de pasajeros, $1 \leq n \leq 1.000.000$, a la espera de embarcar. A continuación, en una segunda línea, aparece un número por cada uno de los n pasajeros indicando en qué cubierta tiene su pasaje. Los transatlánticos tienen numeraciones de las cubiertas bastante caprichosas (por ejemplo, por superstición algunos no tienen cubierta número 13) de modo que no se debe asumir que sean consecutivos. Al menos sabemos que siempre son positivos menores que 2^{31}.

Tras la descripción de los pasajeros aparece una nueva línea con un número indicando cuántas acciones realizan los responsables del embarque. Estas acciones aparecen enumeradas a continuación, cada una en una línea.

A la acción de embarcar a los tripulantes de una determinada cubierta se la denomina `EMBARQUE` y viene seguida por el número de la cubierta. A la acción de averiguar la cubierta a la que está destinado uno de los pasajeros que aún quedan en la cola se la denomina `CONSULTA`, y viene seguido por su posición en la cola (un número entre 1 y el número de pasajeros restantes).

Los responsables del embarque a veces se confunden y llaman a los pasajeros de cubiertas que no existen (o que ya han sido embarcadas). Sin embargo, siempre preguntan la cubierta de pasajeros existentes.

El programa terminará cuando se llegue a un caso de prueba sin pasajeros.

Salida

Para cada acción `EMBARQUE` el programa indicará cuántos pasajeros quedan aún esperando en el puerto para subir al transatlántico una vez que han embarcado todos los viajeros que iban a la cubierta dada.

Para cada acción `CONSULTA` se indicará la cubierta del pasajero situado en esa posición de la cola de viajeros pendiente.

Se escribirá un `*` después de cada caso de prueba.

Entrada de ejemplo

```
2
3 4
2
EMBARQUE 0
CONSULTA 2
10
0 1 0 2 0 3 0 4 0 5
5
CONSULTA 1
CONSULTA 2
EMBARQUE 0
CONSULTA 1
CONSULTA 2
0
```

Salida de ejemplo

```
2
4
*
0
1
5
1
2
*
```

D
Dividir factoriales

Es sabido que el factorial de un número positivo n, escrito $n!$, es la multiplicación de todos los números entre 1 y n. Por ejemplo, 5! es $5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 = 120$.

Con un ordenador, calcular el factorial de un número es fácil, de modo que esta vez te pedimos que nos digas el resultado de dividir dos factoriales. ¿Eres capaz?

Entrada

La entrada está compuesta de múltiples casos de prueba, uno por línea.

Un caso de prueba se compone de dos números separados por un espacio, *num* y *den*. Ambos serán números positivos mayores que 0 y menores que 1.000.000.

La entrada termina con un caso en el que $num < den$, que no debe procesarse.

Salida

Para cada caso de prueba se mostrará, en una línea independiente, el resultado de dividir *num*! y *den*!, es decir $\frac{num!}{den!}$. Se garantiza que el resultado será siempre menor que 2^{63}.

Entrada de ejemplo

```
5 2
6 5
8 3
17 8
1 2
```

Salida de ejemplo

```
60
6
6720
8821612800
```

E
Los problemas de ser rico

Aunque no acostumbra a pasearse por las cocinas, tiene sirvientes para eso, una noche de insomnio el magnate Rick Achón sintió algo de hambre y decidió darse una vuelta y buscar algo de comer, con la esperanza de que el paseo le sirviera además para que por fin Morfeo acudiera a su encuentro.

Cuando entró en la despensa le entró cierto desasosiego; había viandas por todos sitios, jamones, chorizos, marisco, legumbres, ... pero no había nada que se pudiera considerar *vivo*.

Por si sufría más noches de insomnio, al día siguiente se propuso poner solución a este problema y no reparó en gastos al comprarse un gigantesco acuario.

El acuario tenía varios compartimentos escalonados, cada uno menor que el anterior. Tras su instalación, los sirvientes tuvieron que encargarse de su llenado. Y esto no resultó ser una tarea sencilla porque el magnate quería que sus nuevos peces vivieran en agua mineral, por lo que tuvieron que averiguar la capacidad del acuario para saber cuántos litros de agua comprar.

Entrada

La entrada comienza con un número no negativo indicando el número de casos de prueba que vendrán a continuación. Por cada caso de prueba se proporcionará, en una única línea, la descripción del acuario con tres números: el número de compartimentos del acuario (>0), la capacidad en litros del más grande (>0), y la diferencia de litros entre dos compartimentos adyacentes (≥ 0). Ninguno de los tres números será mayor de 10.000.

Salida

Para cada caso de prueba de la entrada, el programa escribirá el número de litros de agua mineral que debieron comprarse para llenar el acuario.

Entrada de ejemplo

```
3
5 300 10
4 400 20
3 500 30
```

Salida de ejemplo

```
1400
1480
1410
```

F
Por 3 o más 5

Cuenta la leyenda que un famoso matemático, tras aprender a sumar y multiplicar a la tierna edad de 3 años en apenas 5 días, se dio cuenta de que, empezando por 1, podía generar un montón de números sin más que multiplicar por 3 o sumar 5 a alguno de los que ya hubiera generado antes.

Por ejemplo, el 23 (edad a la que se casaría) lo obtuvo así:

$$((1 + 5) \times 3) + 5$$

Por su parte el 77 (edad a la que tendría su primer bisnieto) lo consiguió:

$$(((1 \times 3 + 5) \times 3) \times 3) + 5$$

Por mucho que lo intentó, algunos números, sin embargo, resultaron ser imposibles de obtener, como por ejemplo el 5, el 7 o el 15.

Entrada

La entrada estará compuesta de una serie de números positivos no mayores que 20.000, cada uno en una linea. El programa terminará al leer el número 0, que no deberá procesarse.

Salida

Para cada número de la entrada, el programa escribirá `SI` si el número se puede escribir como una sucesión (quizá vacía) de multiplicaciones por 3 y sumas de 5 al número 1 como las de los ejemplos, y `NO` en otro caso.

Entrada de ejemplo

```
5
7
15
23
77
18000
0
```

Salida de ejemplo

```
NO
NO
NO
SI
SI
NO
```

G
Números reversibles

Se denomina *número reversible* a aquél que al ser sumado a sí mismo tras invertir sus dígitos da como resultado un número en el que todos los dígitos son impares.

Por ejemplo, el número `36` es *reversible* pues `36 + 63 = 99`, y los dos dígitos de `99` son impares. Fíjate que esto significa que también el número `63` es reversible. También lo son el `409` y el `904`.

Para ser considerado número reversible, la cantidad de dígitos del número y de su versión invertida debe ser el mismo. Por tanto, el número `1010` *no* es reversible, incluso aunque `1010 + 0101 = 1111`. No se considera válido porque `0101` es en realidad el número `101`, que tiene menos dígitos que `1010`.

Hay más números reversibles de lo que podría parecer. Por ejemplo, hay 20 de 2 dígitos, y 100 de 3.

Entrada

La entrada está compuesta de una serie de casos de prueba. Cada uno contendrá una línea, con un número positivo menor que 10^9.

Un caso de prueba con un `0` indica el final, y no deberá procesarse.

Salida

Para cada caso de prueba el programa deberá escribir `SI` si el número es reversible, y `NO` si no lo es.

Entrada de ejemplo

```
36
904
1010
37
209
0
```

Salida de ejemplo

```
SI
SI
NO
NO
SI
```

H
Salvemos al lince ibérico

El lince ibérico (*Lynx pardinus*) es un animal que habita en la península ibérica. Desgraciadamente, a día de hoy quedan muy pocos individuos (no muchos más de 300) lo que hace que sea la especie de felino más amenazada del mundo.

Una de las amenazas principales de esta especie son los atropellos en las carreteras. Por eso es importante adecuar las infraestructuras de las pocas zonas donde habitan para minimizar las posibilidades de que estos preciosos animales terminen cruzando por el asfalto.

Hace años se hizo el esfuerzo de colocar, en muchas carreteras que atraviesan el hábitat de los linces, vallas protectoras que no permiten que éstos salten desde los laterales hacia la carretera. Se instalaron en las zonas de baja visibilidad donde era probable que al conductor no le diera tiempo a ver con la antelación suficiente a un despistado lince cruzando.

Sin embargo, no todo el mundo estuvo a favor, pues, aseguraban, hay mejores soluciones que las vallas. Los ecologistas, por ejemplo, consideran que debido a la imposibilidad de ser cruzadas, las carreteras valladas interrumpen las rutas de migración y segmentan las poblaciones de vida silvestre. Por su parte, los expertos en seguridad vial las ven peligrosas, pues ante un accidente impiden que los accidentados puedan abandonar la carretera y dirigirse al campo.

La polémica ha vuelto a abrirse, porque se quiere ampliar el programa de protección para garantizar que no habrá más muertes de linces, asegurando también el resto de tramos que aún no tienen valla. La opción que se está planteando en estos momentos es la construcción de pequeños *pasos subterráneos* o túneles por debajo de las carreteras. Cuando existen, los animales que quieren cruzar, en vez de pisar el frío asfalto, se decantan por ir por el túnel, salvándose así de una muerte casi segura.

Los ingenieros de caminos han dividido las carreteras en secciones de 100 metros de longitud, y ahora tienen que decidir dónde colocan esos pasos subterráneos. Los biólogos expertos en linces han determinado que un túnel colocado en una sección da servicio a tres secciones; es decir, si un lince llega a una sección sin vallar y sin paso subterráneo, *no* se lanzará a cruzar por la carretera si una de las secciones adyacentes tiene túnel.

Aún se está en una fase muy temprana del proyecto y, de momento, se quiere plantear un presupuesto. Éste depende únicamente del número de túneles y no de su posición. Dada la descripción de una carretera, ¿cuál es el mínimo número de túneles que hay que construir para que los linces no crucen por el asfalto?

Entrada

La entrada comienza con un número que indica cuántas descripciones de carreteras vendrán a continuación. Cada descripción consta de una única línea de no más de 10.000 caracteres. Cada uno de esos caracteres representa una sección de la carretera y puede ser o bien `.` o bien `X`. El símbolo `X` indica que esa sección ya tiene colocada una valla y que por lo tanto los linces no podrán cruzar por ahí. Los `.` indican secciones por las que un lince podría intentar cruzar y que, por tanto, hay que proteger con un paso subterráneo.

Salida

Para cada caso de prueba se escribirá una única línea con un número que indique el número mínimo de pasos subterráneos que hay que construir. Ten en cuenta que lo que se quiere evitar es que los linces crucen la carretera por la superficie en cualquier punto, algo que ya consiguen las vallas en sus tramos (`X`). No obstante, es posible, si así se considera oportuno, construir un paso subterráneo en un tramo vallado.

Entrada de ejemplo

```
3
...
...X.
XXX.X.
```

Salida de ejemplo

```
1
2
1
```

Saltos de trampolín

El reglamento de la FINA (*Federation Internationale de Natation*) para las competiciones oficiales de saltos de trampolín individual dice que, en los Juegos Olímpicos y Campeonatos Mundiales, los saltos serán evaluados por 7 jueces. Cada uno de ellos proporcionará una nota dentro de una escala entre 0 y 10, con incrementos de medio punto de acuerdo a la siguiente escala:

- Completamente fallido: 0 puntos.
- Insatisfactorio: de 0.5 a 2 puntos.
- Deficiente: de 2.5 a 4.5 puntos.
- Satisfactorio: de 5 a 6 puntos.
- Bueno: de 6.5 a 8 puntos.
- Muy bueno: de 8.5 a 10 puntos.

De las 7 notas, se eliminarán las dos notas más altas, y las dos más bajas.

Las notas restantes son sumadas y multiplicadas por el *Coeficiente de Dificultad* (CD) del salto. Los saltos más complejos tienen un mayor CD, de manera que se compensa la mayor dificultad de conseguir puntuaciones altas por parte de los jueces.

Por ejemplo, en un salto cuyo CD es 2.0, si las puntuaciones de los jueces son:

~~7.5~~ ~~7.0~~ ~~7.5~~ 7.5 7.5 ~~8.0~~ 7.5

tras quitar las dos notas mayores y las dos menores (que aparecen tachadas) la puntuación queda:

$$(7.5 + 7.5 + 7.5) \times 2 = 45.0$$

Entrada

La entrada estará compuesta de múltiples casos de prueba, cada uno en una linea.

Cada caso de prueba consistirá en 7 números, entre 0 y 10 en intervalos de 0.5, separados por espacios, indicando las 7 notas de los jueces en una competición de salto de trampolín. Los 7 números tendrán un único dígito decimal.

Salida

Para cada caso de prueba el programa indicará la puntuación final del competidor. Como *coeficiente de dificultad* se asumirá un valor de 2.0. Gracias a eso, la puntuación final nunca tendrá decimales, por lo que se deberá escribir únicamente su parte entera.

Entrada de ejemplo

```
7.5 7.0 7.5 7.5 7.5 8.0 7.5
8.0 9.5 7.5 9.0 7.0 8.0 8.5
```

Salida de ejemplo

```
45
49
```

J

Sudokus vacíos

Un *sudoku* es un tipo de pasatiempo numérico que se popularizó en Japón en 1986 e internacionalmente en 2005, aunque sus raíces alcanzan hasta el propio Leonhard Euler, en el siglo XVIII. Consiste en una tabla de 9 × 9 celdas, en la que se distinguen 9 "*regiones*" de 3 × 3.

En cada una de las celdas se debe escribir un dígito entre 1 y 9. Inicialmente se proporcionan algunas de las celdas ya rellenas, y se deja al jugador la responsabilidad de completar las demás, sabiendo que cada fila, cada columna y cada región de 3 × 3 debe contener todos los dígitos del 1 al 9 una única vez.

			5		9	2		7
6	7		3					4
					2			
	4	9	2				8	5
2	3				4	1	6	
			1					
5					8		9	1
8		4	9		7			

Los creadores de *sudokus* se encargan de que, dada la distribución de partida, sólo exista un modo de completar el resto de celdas. Además, *Nikoli*, la compañía japonesa que bautizó y popularizó el pasatiempo, impuso una restricción extra: para ser considerado un buen *sudoku*, éste no debe proporcionar más de 32 celdas rellenas, y debe tener *simetría rotacional*. Esto significa que si la distribución inicial del *sudoku* se rota 180 grados, las *celdas ocupadas* son las mismas, aunque no necesariamente con los mismos dígitos. Por ejemplo, tras rotarlo 180 grados, el *sudoku* anterior queda:

			7		9	4		8
1	6		8					5
					1			
	6	1	4				3	2
5	8				2	9	4	
			2					
4					3		7	6
7		2	6		5			

Si se superponen las dos figuras, se observa que, aunque con números diferentes, las celdas que están rellenas son las mismas. Los buenos aficionados a resolver *sudokus* aprovechan esto en su beneficio. Cada vez que completan una celda, se plantean si es posible rellenar la celda simétrica, algo que, aseguran, ocurre con bastante frecuencia.

Entrada

La entrada comienza con un primer número indicando el número de casos de prueba que vienen a continuación. Cada caso de prueba consiste en la distribución inicial de un *sudoku*.

Cada *sudoku* se proporciona con 9 líneas. Cada una está compuesta a su vez de 9 caracteres contiguos, cada uno representando el valor de una de las celdas de esa fila. El símbolo - se utiliza para indicar una celda vacía.

Dos casos de prueba consecutivos se separan por una línea en blanco. También hay una línea en blanco antes del primer *sudoku*.

Salida

Para cada caso de prueba se escribirá `SI` si el *sudoku* es válido, es decir si no supera los 32 dígitos rellenos y tiene *simetría rotacional*. En otro caso, se escribirá `NO`.

Ten en cuenta que *no* hay que preocuparse de si, con la configuración de partida, el *sudoku* es o no resoluble, por lo que los números leídos son indiferentes y el programa debe únicamente preocuparse de qué casillas están llenas y cuáles no.

Entrada de ejemplo

```
2

---5-92-7
67-3----4
-----2---
-492---85
---------
23---416-
---1-----
5----8-91
8-49-7---

---5-92-7
67-3----4
-----2---
-492---85
---------
23---416-
---------
5----8-91
8-49-7---
```

Salida de ejemplo

```
SI
NO
```