Preparación

ADA BYRON

MADRID

SPECIALIST

Competitive Programming

UPV

Problema número 563

Teclado de televisor

Tiempo máximo: 1,000-3,000 s Memoria máxima: 4096 KiB

http://www.aceptaelreto.com/problem/statement.php?id=563

Queremos buscar series y películas en la televisión de la forma más eficiente posible. Cuando se busca un texto en la televisión aparece un teclado virtual por el que nos desplazamos con las flechas del mando a distancia: arriba, derecha, abajo e izquierda. El objetivo es calcular el número mínimo de desplazamientos que hay que realizar dado un teclado y un texto a introducir determinados. Debe tenerse en cuenta que hay teclas de distinto tamaño y que se trata de un teclado circular: cuando el cursor se sale por un extremo del teclado, aparece por el extremo opuesto. El cursor se situa inicialmente sobre la tecla que ocupa la esquina superior izquierda.

A continuación se muestra un ejemplo de teclado:

```
AAABCD
AAAGHI
LEEEEK
```

Un teclado tiene forma rectangular, y todas sus teclas son también rectángulos, aunque puede haber teclas de diferentes tamaños. Así, en el teclado del ejemplo la tecla `A` tiene dimensiones 2×3, la tecla `B` tiene dimensiones 1×1, la `E` 1×4, etc. El efecto de pulsar cada una de las flechas del mando a distancia se describe a continuación:

- Derecha: el cursor se desplaza a la tecla de la derecha, y si hay varias a la de más arriba. En el ejemplo, si se pulsa *derecha* cuando el cursor está sobre la tecla `A`, el cursor se situará sobre la tecla `B`.
- Izquierda: el cursor se desplaza a la tecla de la izquierda, y si hay varias a la de más arriba. En el ejemplo, si se pulsa *izquierda* cuando el cursor está sobre la tecla `A`, el cursor se situará sobre la tecla `D`.
- Arriba: el cursor se desplaza a la tecla de arriba, y si hay varias a la de más a la izquierda. En el ejemplo, si se pulsa *arriba* cuando el cursor está sobre la tecla `E`, el cursor se situará sobre la tecla `A`.
- Abajo: el cursor se desplaza a la tecla de debajo, y si hay varias a la de más a la izquierda. En el ejemplo, si se pulsa *abajo* cuando el cursor está sobre la tecla `A`, el cursor se situará sobre la tecla `L`.

Para el teclado del ejemplo, una posible forma de teclar el texto `ADKEAB` con el mínimo número de desplazamientos podría ser: izquierda, arriba, izquierda, arriba y derecha.

Entrada

La entrada está formada por la descripción de una serie de teclados, y hay varios textos de prueba para cada teclado. La descripción de un teclado empieza con una línea con tres números enteros: F, el número de filas del teclado; C, el número de columnas; y N, el número de textos a introducir usando el teclado. Se garantiza que $1 \leq F \leq 8$, $1 \leq C \leq 20$ y $1 \leq N \leq 100$. A continuación aparecen F líneas, con C caracteres cada una, describiendo el teclado según el formato del ejemplo anterior. Los caracteres válidos son las letras mayúsculas y los números, además del punto ("`.`"), que representa el *espacio*. Las teclas son siempre rectangulares y nunca hay dos teclas distintas asociadas al mismo carácter.

Tras la descripción del teclado aparecen N líneas, cada una con una cadena de texto a introducir usando el teclado. Todas las cadenas están formadas con los caracteres asociados a las teclas del teclado, y nunca tienen una longitud mayor que 100. Al empezar a escribir cada palabra, el cursor está en la letra superior izquierda del teclado, independientemente de dónde terminara con la anterior.

El final de la entrada se indica con una línea con tres ceros que no se debe procesar.

Salida

Para cada texto de prueba se debe imprimir, en una línea distinta, el número mínimo de desplazamientos necesarios para teclearlo.

Después de cada caso de prueba se escribirá una línea con tres guiones (`---`).

Entrada de ejemplo

```
3 6 4
AAAB.D
AAAGHI
LEEEEK
ADKEA
B.D
ELE
AA
1 4 2
ABBC
ABC
CBA
0 0 0
```

Salida de ejemplo

```
4
3
4
0
---
2
3
---
```

Autor: Luis Fernando Lago Fernández.

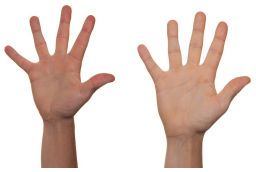
Revisores: Marco Antonio Gómez Martín y Pedro Pablo Gómez Martín.

Problema número 500

Votaciones capicúa

Tiempo máximo: 1,000-2,000 s Memoria máxima: 4096 KiB

http://www.aceptaelreto.com/problem/statement.php?id=500



Ruritania tiene N senadores, con N entre 1000 y 9999, número que suele cambiar según la voluntad del presidente ruritano, que es bastante veleta. También cambia a voluntad del presidente el *quorum* Q con el mínimo número de senadores que deben participar en una votación para que se considere válida, aunque siempre se ha de cumplir que $1000 \leq Q$.

En el senado sólo se puede votar sí o no, y por supuesto no todos los senadores votan siempre. Pero cuando el número concatenado de las cuatro cifras con los votos a favor y las cuatro cifras con los votos en contra es capicúa, tras la votación todos se van al bar del senado para celebrarlo.

Para la concatenación se cuentan los ceros a la izquierda; esto es una votación 1000 – 1 es capicúa, pues la concatenación es `10000001`, y también lo es una votación 1 – 1000, pues la concatenación es `00011000`.

La pregunta es: si en esta legislatura Ruritania tiene N senadores ($1000 \leq N \leq 9999$) y el *quorum* es Q ($1000 \leq Q \leq N$), ¿cuántas votaciones capicúa son posibles?

Entrada

La entrada está formada por varias líneas con dos enteros N, Q. Se garantiza $1000 \leq Q \leq \mathrm{N} \leq 9999$. El final de la entrada se indica con una línea con dos ceros que no se debe procesar.

Salida

Para cada caso de prueba N, Q debe escribirse una línea con los números N, Q y M, donde M es el número de posibles votaciones capicúa con N senadores y un *quorum* de Q.

Entrada de ejemplo

```
1001 1000
2100 2000
5324 4999
0 0
```

Salida de ejemplo

```
1001 1000 2
2100 2000 3
5324 4999 156
```

Autor: José Ramón Dorronsoro Ibero.

Revisores: Luis Fernando Lago Fernández, Pedro Pablo Gómez Martín y Marco Antonio Gómez Martín.

Problema número 554

Escaleranos

Tiempo máximo: 1,000-2,000 s Memoria máxima: 4096 KiB

http://www.aceptaelreto.com/problem/statement.php?id=554

Los habitantes de *Escalera* son famosos por su capacidad de subir de golpe varios peldaños de una escalera. Además les gusta especializarse en subir un número determinado de peldaños. Juan, por ejemplo, está especializado en subir 1, 3 o 5 peldaños en cada paso (lo que representamos mediante la lista [1, 3, 5]). En esta lista de peldaños siempre aparecerá el 1 en primer lugar, pues todos los escaleranos son capaces de subir los peldaños de uno en uno.

Así las cosas nos surge la curiosidad de saber, por ejemplo, de cuántas maneras distintas podría subir una escalera de 10 peldaños un escalerano especializado en subir [1, 3, 5] peldaños. De forma más general, si un escalerano se especializa en subir [p_1=1, p_2, ..., p_k] peldaños, ¿de cuántas maneras diferentes podrá subir una escalera de n peldaños llegando arriba de forma exacta?

Entrada

La entrada está formada por distintos casos de prueba, uno por línea. Cada caso de prueba contiene un primer número entero $1 \leq n \leq 100$ con el número total de peldaños de una cierta escalera. A continuación hay un segundo número entero $1 \leq k \leq 10$ con el número de diferentes peldaños que un escalerano puede subir de golpe. Finalmente hay k números enteros $p_1 = 1,...,p_k$ con los valores de dichos números de peldaños. Se cumple que todos los p_i son distintos entre sí y menores o iguales a n.

El final de la entrada se indica con una línea que empieza con un 0 y que no debe procesarse.

Salida

Para cada caso de prueba se escribirá una línea con el número de formas diferentes en las que el escalerano puede subir la escalera, llegando al último peldaño de forma exacta. Como el resultado puede ser muy grande se dará el resto que queda tras dividir el número total entre 1.000.000.007 (10^9+7).

Entrada de ejemplo

```
5 2 1 3
5 2 1 4
5 2 1 5
5 3 1 3 2
3 2 1 4
0
```

Salida de ejemplo

```
4
3
2
13
1
```

Autor: Luis Fernando Lago Fernández.

Revisores: Marco Antonio Gómez Martín y Pedro Pablo Gómez Martín.

Problema número 392

Amigo invisible

Tiempo máximo: 1,000-2,000 s Memoria máxima: 4096 KiB

http://www.aceptaelreto.com/problem/statement.php?id=392

El *amigo invisible* es un modo muy popular de que un grupo de personas se haga regalos entre sí en fechas señaladas, como por ejemplo en Navidad. El organizador escribe los nombres de cada participante en una papeleta y los introduce en una bolsa. Uno a uno, todos van sacando uno de los papelillos, que contendrá el nombre de la persona a la que deberán hacer el regalo. Si una persona extrae su propio nombre, el proceso tendrá que repetirse desde el principio: ser el amigo invisible de uno mismo hace que el juego pierda todo el interés.

Los padres del pequeño Samuel le han propuesto organizar un amigo invisible familiar, y le ha tocado regalar a su madre. Lleva varios días un poco desconcertado, porque, aunque no es capaz de explicar por qué, tiene la sospecha de que a su madre le ha tocado su padre, y a su padre le ha tocado regalarle a él. Esto del amigo invisible no parece tener ninguna gracia cuando juegan tres.

Le ha contado su preocupación a su prima Alana, que ha jugado también al amigo invisible en casa, con sus padres y su hermana Irene. A ella le tocó su madre... que le pidió ayuda para elegir algún juguete que pudiera gustarle a Irene, que le había tocado a ella. Alana tampoco supo explicar cómo, pero también supo quién era el amigo invisible de todos en su familia.

Está claro que esto del amigo invisible no tiene ninguna gracia. ¿Por qué lo llaman invisible si se puede sacar la asignación tan fácilmente?

Entrada

El programa deberá leer, de la entrada estándar, múltiples casos de prueba, cada uno relativo a la realización de un *amigo invisible* entre un grupo de personas.

La primera línea del caso de prueba contiene dos números. El primero $2 \leq p \leq 50$ indica el número de participantes en el amigo invisible. El segundo $1 \leq a \leq 50$ indica el número de asignaciones conocidas por uno de los participantes, por su propia papeleta y por suposiciones o confesiones directas de los demás.

A continuación vendrán a líneas, cada una indicando una de las asignaciones conocidas. Cada asignación se proporciona con dos números, el primero indicando la persona que regala, y el segundo a quién lo hace. Los participantes se numeran de 1 a p.

La entrada termina con un caso de prueba sin participantes, que no deberá procesarse.

Salida

Para cada caso de prueba, el programa deberá escribir "`SI`" si es posible conocer, sin posibilidad de error, la asignación de todos los participantes en el amigo invisible, y "`NO`" en otro caso.

Entrada de ejemplo

```
3 1
1 2
4 2
1 2
2 3
4 2
1 2
4 3
0 0
```

Salida de ejemplo

```
SI
SI
NO
```

Autor: Pedro Pablo Gómez Martín.

Revisores: Alberto Verdejo y Marco Antonio Gómez Martín.

Problema número 247

Saliendo de la crisis

Tiempo máximo: 1,000-2,000 s Memoria máxima: 4096 KiB

http://www.aceptaelreto.com/problem/statement.php?id=247

La abeja reina dijo hace unos cuantos meses que la colmena estaba por fin saliendo de la crisis. Ahora Maya quiere comprobar cómo de ciertas eran aquellas declaraciones, así que ha recopilado el histórico de distintos indicadores económicos desde el día de la declaración hasta hoy para ver si, efectivamente, todos ellos han ido creciendo día a día desde entonces.

Entrada

La entrada estará compuesta de distintos indicadores económicos, cada uno de ellos en dos líneas distintas. La primera línea indica el número de *muestras* recogidas del indicador ($0 < n \leq 100$). La segunda línea contiene n números positivos con los valores económicos (entre 1 y 10.000.000) medidos desde el día de la declaración de la abeja reina hasta el día de hoy.

La entrada termina con un indicador sin muestras (`0`) que no debe procesarse.

Salida

Por cada caso de prueba se dirá si según ese indicador la abeja reina tenía razón (`SI`) o las cosas no están tan bien como ella cree (`NO`).

Entrada de ejemplo

```
3
1 3 6
4
1 3 2 5
3
6 6 6
0
```

Salida de ejemplo

```
SI
NO
NO
```

Autores: Marco Antonio Gómez Martín y Pedro Pablo Gómez Martín.

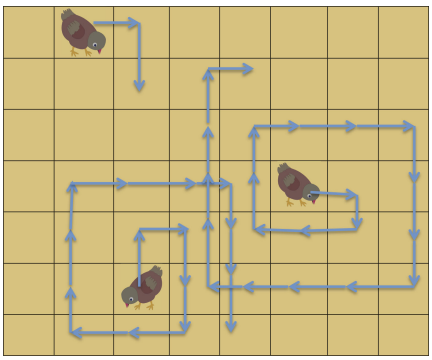
Revisores: Alberto Verdejo y Isabel Pita.

Problema número 328

La comida de los pollitos

Tiempo máximo: 1,000-3,000 s Memoria máxima: 4096 KiB

http://www.aceptaelreto.com/problem/statement.php?id=328



Los pollitos pasan todo el día picoteando el suelo del gallinero para comer los granos que van encontrando. Has estado varios días observando sus movimientos y has descubierto que todos ellos siguen un curioso patrón basado en las baldosas que hay en el suelo. Se despiertan mirando en una dirección (norte, sur, este, oeste) y comienzan a andar en esa dirección siguiendo un movimiento en espiral en el sentido de las agujas del reloj. El paseo termina cuando se cansan (cada pollito tiene un aguante distinto) o en su recorrido se topan con el borde del gallinero, momento en el que quedan aturdidos y se duermen hasta el día siguiente.

Además, en cada baldosa que pisan (incluída la que ocupan al despertarse) miran si hay pienso y si lo hay, comen un grano antes de dar el siguiente paso. Si no hay, simplemente siguen avanzando. Como son pequeños, cuando coinciden en un punto comen al mismo tiempo sin molestarse y a veces incluso duermen en el mismo sitio.

Para conseguir que tus pollitos crezcan lo más rápidamente posible sin gastar de más, has decidido distribuir los granos en el gallinero de forma que en cada paso cada pollito se encuentre un grano y pueda comerlo. Conoces la dirección en que se despiertan los pollitos, y el número de pasos que pueden dar en la espiral antes de cansarse y parar hasta el día siguiente. Ahora debes averiguar los granos que tienes que colocar en cada punto para que, al acabar el día, no quede ninguno y todos los pollitos hayan comido el máximo posible.

Entrada

La entrada comienza con una línea con el número de casos de prueba que deberán procesarse. Cada caso comienza con una línea con tres números, f, c y n indicando, respectivamente, el tamaño del gallinero en la dirección norte-sur, el tamaño en la dirección oeste-este y el número de pollitos ($1 \leq f, c \leq 50$; $0 \leq n \leq 500$). Las n líneas siguientes contienen la información de cada pollito. El primer número, v, indica la posición en la dirección norte-sur ($1 \leq v \leq f$), el segundo, h, la posición en la dirección oeste-este ($1 \leq h \leq c$), después se indica la dirección en la que empieza a moverse el pollito (`N`, `S`, `E`, `W`) y finalmente el número máximo de pasos que aguanta antes de dormirse a descansar (al menos uno).

Salida

Para cada caso de prueba se escribirán f líneas. En cada una se escribirán c valores, separados por un espacio en blanco, con el número de granos que hay que poner en cada punto. Tras cada caso de prueba se escribirá una línea con tres guiones (`---`).

Entrada de ejemplo

```
2
7 8 3
1 2 E 2
6 3 N 25
4 6 E 21
4 3 2
1 1 N 3
4 2 W 3
```

Salida de ejemplo

```
0 1 1 0 0 0 0 0
0 0 1 1 1 0 0 0
0 0 0 1 1 1 1 1
0 1 1 2 2 1 1 1
0 1 1 2 2 1 1 1
0 1 1 2 2 1 1 1
0 1 1 1 1 0 0 0
---
1 0 0
0 0 0
1 1 0
1 1 0
---
```

Autor: Isabel Pita.

Revisor: Pedro Pablo Gómez Martín.

Problema número 558

CamelCasi

Tiempo máximo: 1,000 s Memoria máxima: 4096 KiB

http://www.aceptaelreto.com/problem/statement.php?id=558

En programación es habitual utilizar el estilo *camel case* para escribir los nombres de variables que concatenan varias palabras. La norma de estilo indica que la primera letra de cada una de las palabras concatenadas debe ser mayúscula, mientras que el resto deben escribirse en minúscula. Algunos ejemplos de nombres de variables utilizando esta notación son `CamelCase`, `AdaByron` o `ConcursoDeProgramacion`.

Jaime ha decidido utilizar el estilo *camel case* para nombrar todas sus variables. Sin embargo ha cometido errores al teclear, y algunas de las letras que deberían ser mayúsculas aparecen en minúscula. Como consecuencia el compilador ha mostrado un montón de errores, y Jaime ha decidido llamar a su estilo *camel casi*.

Entrada

La entrada está formada por distintos casos de prueba, cada uno ocupando varias líneas.

Cada caso de prueba comienza el número N de identificadores que hay en el programa ($1 \leq N \leq 1.000$). A continuación aparecen N líneas con los identificadores (de hasta 50 caracteres) en formato *camel casi*. Todas las variables aparecen al menos una vez bien escritas (en formato *camel case* correcto). Además no existen dos variables que se diferencien sólo en el uso de mayúsculas. Por ejemplo las cadenas `AdaByron` y `adaByron` se refieren a la misma variable.

Salida

Para cada identificador de la entrada en formato *camel casi* se imprimirá una línea con el nombre de la variable en formato *camel case*.

El final de cada caso de prueba vendrá seguido de una línea con tres guiones, `---`.

Entrada de ejemplo

```
2
AdaByron
adabyron
4
camelcase
concursoDeProgramacion
CamelCase
camelCase
```

Salida de ejemplo

```
AdaByron
AdaByron
---
CamelCase
concursoDeProgramacion
CamelCase
CamelCase
---
```

Autor: Luis Fernando Lago Fernández.

Revisores: Marco Antonio Gómez Martín y Pedro Pablo Gómez Martín.

Problema número 564

El misterioso caso de los múltiplos de 7

Tiempo máximo: 2,000 s Memoria máxima: 4096 KiB

http://www.aceptaelreto.com/problem/statement.php?id=564

El belga Hércules Poirot es conocido como el mejor detective privado del mundo. Además de resolver los asesinatos que se van cruzando por su camino, últimamente en su tiempo libre ha tomado especial interés por la teoría de números.

Hoy Poirot está investigando las propiedades de las potencias de 2. Además, como el 7 es su número de la suerte, está interesado en aquellos que sean múltiplos de 7.

Sin embargo, como sabe hasta el más ignorante de los villanos con los que Poirot se cruza, no existe ninguna potencia de 2 que sea múltiplo de 7, obviamente. Pero utilizando sus pequeñas células grises, se ha dado cuenta de que restándole una unidad a una potencia de 2, sí podemos llegar a veces a un múltiplo de 7. Poirot está deseando saber cuántos números positivos menores que 2^N de la forma $2^k - 1$ (con k un entero positivo) son múltiplos de 7.

Entrada

La entrada comienza por el número de casos de prueba que vendrán a continuación. Cada caso aparece en una línea que contiene un único valor N ($1 \leq N \leq 500.000$), el valor conocido por Poirot.

Salida

Para cada caso de prueba, se escribirá una única línea con la respuesta al problema de Poirot, es decir, cuántos números positivos menores que 2^N de la forma $2^k - 1$ son múltiplos de 7.

Entrada de ejemplo

```
3
2
3
10
```

Salida de ejemplo

```
0
1
3
```

Autores: Vladyslav Lyeuta y Alberto Verdejo.

Revisor: Pedro Pablo Gómez Martín.

Problema número 452

Bingo infantil

Tiempo máximo: 2,000 s Memoria máxima: 4096 KiB

http://www.aceptaelreto.com/problem/statement.php?id=452

Para que los niños que tiene en clase practiquen las restas que acaba de enseñarles, Mavi ha pensado en una versión especial del juego del Bingo. En la versión tradicional, cada jugador recibe un cartón con una serie de números, y se van extrayendo de un bombo bolas con números impresos hasta que alguien asegura que todos los números de su cartón han salido ya.

En la variante que ha pensado Mavi, en cada jugada extraerá dos números en lugar de solo uno. El valor jugado, que los niños tendrán que tachar de sus cartones, es la resta del mayor menos el menor. Tras cada jugada, los dos números serán incorporados de nuevo al bombo, en contra de lo que ocurre en el juego tradicional.

Aunque la idea es interesante, Mavi se enfrenta a un problema. El bingo que va a utilizar lleva en el armario de la clase muchos años y ha pasado por muchas manos... algunas un poco descuidadas que han hecho que se pierdan bolas. De modo que necesita saber la lista de números que pueden "salir" en su particular bingo, para ponerlos en los cartones y que todos tengan la posibilidad de ganar.

Mavi es consciente de que seguramente algunos números tengan más posibilidades de salir que otros, pero no le importa mucho. De hecho más bien lo considera una virtud, porque así podrá crear cartones con números más probables para los niños que restan con dificultad y que tengan también posibilidades de ganar.

Entrada

Cada caso de prueba comienza con un número indicando cuántas bolas quedan aún en el bingo de la clase (al menos 2). A continuación aparece el número de cada una de ellas. Todos los números son valores entre 1 y 2.000 y no hay ninguno repetido.

La entrada termina con un 0 que no debe procesarse.

Salida

Para cada caso de prueba, el programa deberá escribir, por la salida estándar los números que pueden formarse, ordenados de menor a mayor y separados por un espacio. No debe haber espacio tras el último número.

Entrada de ejemplo

```
4
1 3 4 5
3
4 1 8
0
```

Salida de ejemplo

```
1 2 3 4
3 4 7
```

Autor: Pedro Pablo Gómez Martín.

Revisores: Marco Antonio Gómez Martín y Alberto Verdejo.

Problema número 447

Limpiaparabrisas de los híbridos

Tiempo máximo: 1,000-3,000 s Memoria máxima: 4096 KiB

http://www.aceptaelreto.com/problem/statement.php?id=447

Me he comprado un coche híbrido. Es uno de esos que tiene un motor de combustión y un motor eléctrico, y dependiendo de las circunstancias usa uno u otro. La verdad es que son una chulada.

Pero cualquier cosa de mecánica que se tenga que hacer con él requiere un cuidado infinito. El otro día un camión me puso perdido el cristal delantero y al intentar limpiarlo vi que no tenía agua en el "*limpia*". Cuando llegué a casa, me acordé en el aparcamiento, y aprovechando que tenía en el maletero un par de garrafas grandes decidí llenarlo con el agua de un grifo que tenemos abajo. Al mirar las instrucciones del coche me asusté, porque decía que bajo ningún concepto derramara el agua fuera del depósito del "*limpia*", porque si caía en el motor eléctrico podía estropearlo (y, aún peor, sufrir una descarga). También decía que en el depósito entraban exactamente dos litros de agua. Yo tenía una garrafa de cinco litros y otra de cuatro, pero ninguna de dos, así es que preferí no arriesgarme y no llené el depósito.

Cuando lo conté en casa se rieron de mí, porque, aseguraban, podría haber conseguido tener exactamente dos litros de agua en una de las garrafas. Les dije que las garrafas no tenían ningún tipo de escala y yo no tenía nada con lo que marcar, pero lo único que conseguí fueron más risas. No entiendo nada.

Entrada

El programa tendrá que procesar múltiples casos de prueba. Cada uno comienza con un primer número l mayor que 0, indicando cuántos litros entran en el depósito del agua del coche, seguido del número $1 \leq r \leq 4$ de recipientes que tenemos a nuestra disposición.

En la siguiente línea aparecen r números indicando la capacidad, en litros, de cada recipiente disponible (entre 1 y 15).

Todos los números serán enteros. Además se garantiza que al menos uno de los recipientes tendrá una capacidad igual o mayor al del depósito del agua del coche.

La entrada termina con un `0`.

Salida

Para cada caso de prueba se escribirá "`SI`" si es posible conseguir tener exactamente l litros de agua en alguno de los recipientes y "`NO`" en otro caso. Ten en cuenta que no es posible marcar de ninguna forma los niveles de agua en los recipientes. Lo único que se puede hacer es vaciar o llenar un recipiente completamente, o hacer trasvases de uno a otro.

Entrada de ejemplo

```
2 2
5 4
4 2
12 15
0
```

Salida de ejemplo

```
SI
NO
```

Autor: Pedro Pablo Gómez Martín.

Revisores: Alberto Verdejo y Marco Antonio Gómez Martín.

Problema número 496

La estación de esquí

Tiempo máximo: 1,000-4,000 s Memoria máxima: 16384 KiB

http://www.aceptaelreto.com/problem/statement.php?id=496

A Andrés le encanta esquiar. A veces necesita ir de un punto a otro de la estación en un tiempo determinado y, como buen esquiador, siempre busca la ruta que le permita conseguirlo maximizando los metros de descenso.

Este año ha ido a una estación muy grande de los Alpes y necesita ayuda para encontrar las mejores rutas.

Entrada

La entrada está formada por distintos casos de prueba, cada uno compuesto por varias líneas.

La primera línea contiene un único número entero N, que representa el número de remontes mecánicos (sólo de subida) en la estación. A continuación aparecen N líneas que describen cada uno de los N remontes. En cada una aparecen tres números enteros: la altura en metros del punto inicial del remonte (I), la altura en metros del punto final del remonte (F) y el tiempo en segundos que se tarda en subir (T, incluye el posible tiempo de espera haciendo cola). La última línea contiene 3 números enteros: la altura del punto de salida de Andrés (O), la altura del punto de destino (D), y el tiempo máximo en segundos para llegar (M). Se supone que Andrés siempre se puede desplazar hacia abajo esquiando, a razón de un segundo por cada metro de desnivel.

Se cumple $1 \leq N \leq 25$, $0 \leq O, D \leq 3.000$ y $0 < M \leq 10.000$. Para cada remonte se tiene $0 \leq I < F \leq 3.000$ y $500 \leq T \leq 2.000$.

El final de la entrada se indica con una línea con un único cero que no se debe procesar.

Salida

Para cada caso de prueba, se escribirá una línea indicando los metros de bajada de la mejor solución (aquella que consigue ir desde el punto de origen al punto de destino en un tiempo menor o igual que el indicado y maximiza los metros de bajada). Si es imposible llegar al destino en el tiempo indicado se imprimirá la palabra `IMPOSIBLE`.

Entrada de ejemplo

```
3
0 100 500
0 300 1000
200 400 500
0 300 5000
3
0 100 500
0 300 1000
200 400 500
0 400 1700
3
0 100 500
0 300 1000
200 400 500
0 400 1300
0
```

Salida de ejemplo

```
1000
100
IMPOSIBLE
```

Autor: Luis Fernando Lago Fernández.

Revisores: Pedro Pablo Gómez Martín y Marco Antonio Gómez Martín.

Problema número 383

El alcance de las historias

Tiempo máximo: 1,000-2,000 s Memoria máxima: 4096 KiB

http://www.aceptaelreto.com/problem/statement.php?id=383

A David le encantan los libros de la colección "Elige tu propia aventura", en los que hay páginas con opciones para continuar la historia de formas distintas. Además, en un mismo libro puede haber varias páginas de comienzo de historias. En estos libros hay contenidas varias historias posibles y le encanta explorar todas y cada una de ellas. Cuando lee este tipo de libros escribe en un cuaderno la secuencia de las páginas que ha leído desde una de las posibles páginas de comienzo hasta completar una historia y le fascina contemplar los saltos que ha dado por el libro, sobre todo los que se producen hacia el principio del libro. Como le gusta jugar con los números, para cada secuencia de páginas calcula la diferencia mayor entre cada página de la secuencia y cualquier otra página más adelante en la secuencia. Es lo que él llama el "alcance" de la historia. En un libro normal de n páginas, que empezase en la página 1 y en el que se avanzase de uno en uno hasta el final, el alcance sería –1. Si se leyese de final al principio el alcance sería en cambio n–1. Ayuda a David a calcular el alcance de sus historias.

Entrada

La primera línea contiene un número que indica el número de casos de prueba que aparecen a continuación.

Cada caso de prueba se compone de dos líneas. La primera de ellas tiene un único entero con el número de páginas leídas (entero mayor o igual que 2 y menor o igual que 300.000), mientras que la segunda línea contiene la secuencia de páginas (enteros mayores o iguales que 1 y menores o iguales que 300.000).

Salida

Para cada caso de prueba se escribirá el alcance de la historia, es decir, la mayor diferencia entre una página de la secuencia y las que le siguen en la secuencia.

Entrada de ejemplo

```
4
8
1 2 3 8 9 4 5 7
7
10 3 7 8 1 2 11
6
1 2 5 6 2 10
5
1 3 5 7 9
```

Salida de ejemplo

```
5
9
4
-2
```

Autor: Clara Segura.

Revisores: Alberto Verdejo y Pedro Pablo Gómez Martín.

Problema número 324

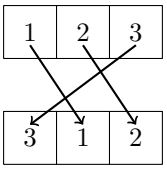
El teorema del punto fijo

Tiempo máximo: 1,000-2,000 s Memoria máxima: 4096 KiB

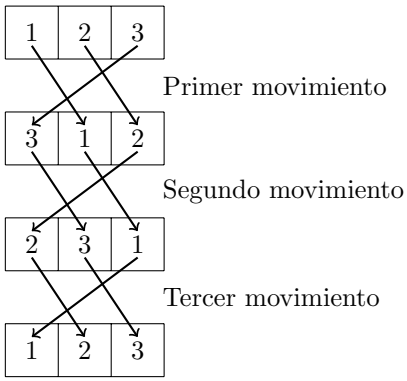
http://www.aceptaelreto.com/problem/statement.php?id=324

En la familia de teoremas matemáticos llamados *del punto fijo*, Brouwer estudió uno que, parece ser, explicaba con la analogía de remover una taza de café con azúcar. Brouwer afirmaba que en todo momento hay un punto/molécula de la taza que no habrá cambiado de lugar.

Nosotros no estudiaremos ese teorema, pero nos quedamos con la idea de que, al mover la taza de café, sus moléculas intercambian sus lugares para terminar en sitios completamente distintos. Por poner un ejemplo reducido (en el que ni siquiera se cumple el teorema del punto fijo), la molécula 1 podría haber pasado a ocupar el lugar que tenía la molécula 2, la molécula 2 pasar donde estaba la molécula 3, y la molécula 3 ocupar el lugar que originalmente tenía la 1.



Si somos capaces de replicar el mismo movimiento una y otra vez, llega un momento en el que la taza de café vuelve a su estado original:



La pregunta que nos hacemos es, dada la descripción del intercambio de moléculas que conseguimos con el movimiento de la taza de café, ¿cuántas veces tendremos que repetirlo para que el estado de la taza vuelva a ser el mismo que al principio?

Entrada

La entrada estará compuesta por distintos casos de prueba. Cada uno de ellos consta de dos líneas, una primera con el número de moléculas en la taza de café ($1 \leq n \leq 100$) y otra indicando qué movimiento de moléculas se realizan cada vez que se mueve la taza.

El movimiento viene definido con n enteros que indican, para cada posición de una molécula, en qué posición termina la molécula que ocupaba ese lugar.

La entrada termina con una taza de café sin moléculas, que no debe procesarse.

Salida

Para cada caso de prueba se escribirá el número de veces que hay que mover la taza de café para que cada molécula recupere su lugar original. Se garantiza que el número no excederá 10^9.

Entrada de ejemplo

```
3
1 2 3
3
2 3 1
4
2 1 4 3
0
```

Salida de ejemplo

```
1
3
2
```

Autor: Marco Antonio Gómez Martín.

Revisores: Alberto Verdejo y Pedro Pablo Gómez Martín.