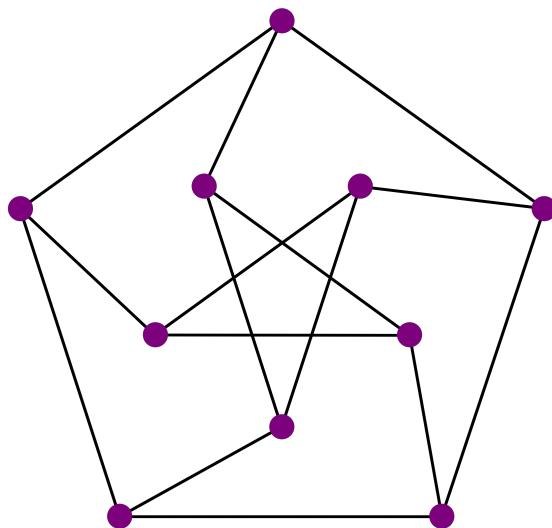


Concurso de Programación ComplICAUS V



Cuadernillo de problemas

Patrocinado por



Realizado en la Escuela Técnica Superior de Ingeniería Informática (US)

13 de febrero de 2026



Índice

| | |
|-------------------------------------|----|
| A - Dos Hermanas | 2 |
| B - La Guerra de Píxeles | 4 |
| C - Lema Técnico | 6 |
| D - El negocio definitivo | 7 |
| E - Operación Anunnakis | 9 |
| F - Crucigrama sevillano | 10 |
| G - Saber y Ganar | 12 |
| H - Pie | 13 |
| I - El Protocolo del Alcázar | 14 |
| J - Problemas de comunicación | 16 |
| K - El archivo de indias | 17 |
| L - Pautas de seguridad | 19 |



Problema A

Dos Hermanas

Basado en: Delft Distance [NWERC 2022 D]



Figura 2: Rotonda de dos hermanas.

Te encuentras en la casa de unos amigos, situada en la esquina noroeste de Dos Hermanas, y quieres llegar cuanto antes a la feria, ubicada en la esquina sureste de la ciudad. Para lograrlo, es necesario atravesar el núcleo urbano, cuyo trazado puede modelarse como una cuadrícula de $h \times w$ manzanas.

A diferencia de ciudades con un diseño completamente ortogonal, Dos Hermanas combina bloques de edificios rectangulares con un gran número de rotondas, elemento característico de su urbanismo. Todos los edificios rectangulares están alineados con los ejes y tienen una longitud de lado de 10m, mientras que las rotondas son perfectamente circulares y tienen un diámetro de 10m. Entre dos construcciones adyacentes existe justo el espacio necesario para un estrecho callejón de anchura despreciable.

Como no quieras perderte el inicio de la feria, necesitas encontrar el camino más corto desde la casa de tus amigos hasta el recinto ferial. Afortunadamente, dispones de un mapa detallado de la ciudad que te permitirá planificar la ruta óptima, evitando dar vueltas innecesarias alrededor de las innumerables rotondas.

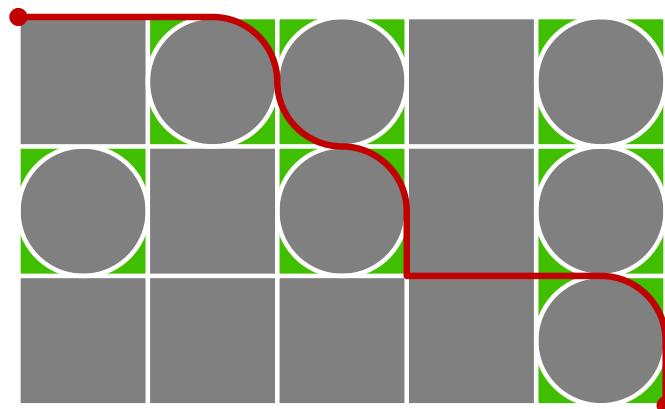


Figura 3: Ilustración de la Entrada de Ejemplo 1, con el camino más corto mostrado en rojo.



Entrada

La entrada consiste en:

- Una línea con dos enteros h y w ($1 \leq h, w \leq 700$), el número de filas y el número de columnas de edificios mostrados en el mapa de la ciudad.
- h líneas, cada una con w caracteres que son 0 (para rotundas) o X (para edificios cuadrados), describiendo las formas de los edificios.

El mapa está orientado con el lado norte hacia arriba.

Salida

Imprime la longitud del camino más corto desde la esquina noroeste hasta la esquina sureste de Dos Hermanas en metros. La respuesta puede tener un error de, como máximo, 10^{-6} m.

Ejemplos

Entrada 1

```
3 5
X00X0
OXOXO
XXXX0
```

Salida 1

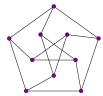
```
71.4159265359
```

Entrada 2

```
1 4
X00X
```

Salida 2

```
45.7079632679
```



Problema B

La Guerra de Píxeles

Basado en: Identifying Map Tiles [NWERC 2015 I]

En el evento masivo de internet conocido como la «Guerra de Píxeles», millones de usuarios luchan por dibujar en un lienzo digital del tamaño del mundo. Tu facción está intentando proteger un dibujo muy importante (un capibara gigante), pero el enemigo avanza rápido.

Para gestionar el inmenso tamaño del lienzo, el servidor lo divide en *teselas*. El nivel de zoom 0 es una única tesela que cubre todo el lienzo. El nivel de zoom 1 divide el mapa en cuatro teselas y, en general, el nivel de zoom n contiene 4^n teselas diferentes, cada una cubriendo una pequeña parte del dibujo.

El problema es la comunicación. Los generales de tu facción envían las órdenes de defensa usando un formato jerárquico llamado *quadkey*, pero tu script de defensa automática necesita coordenadas cartesianas (x, y) para saber dónde colocar el píxel.

Una *quadkey* es una cadena de dígitos que identifica de forma única una tesela en un cierto nivel de zoom. El primer dígito especifica en cuál de los cuatro cuadrantes del mapa completo se encuentra la tesela: 0 para el cuadrante superior izquierdo, 1 para el cuadrante superior derecho, 2 para el cuadrante inferior izquierdo y 3 para el cuadrante inferior derecho. Los dígitos siguientes especifican en qué subcuadrante del cuadrante actual se encuentra la tesela.

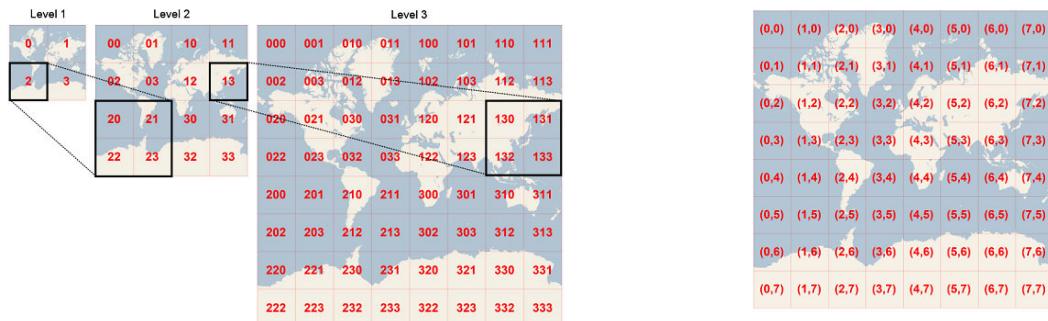


Figura 4: (a) Mapa de órdenes *quadkeys* para niveles de zoom 1 a 3. (b) Coordenadas (x, y) que necesita tu bot para el nivel 3. Nota que el origen $(0, 0)$ es la esquina superior izquierda.

Tu misión es programar el traductor: Dada la *quadkey* de la tesela que está siendo atacada, imprime el nivel de zoom y las coordenadas (x, y) exactas para que tu bot pueda defender la tesela.

Entrada

La entrada consiste en:

- Una línea con una cadena s ($1 \leq \text{longitud}(s) \leq 30$), la *quadkey* recibida desde el cuartel general.
- La cadena s está formada únicamente en los dígitos 0, 1, 2 y 3.

Salida

Imprime tres enteros para configurar el bot: el nivel de zoom y las coordenadas x e y de la tesela.

Ejemplos

Entrada 1

3

Salida 1

1 1 1

Entrada 2

Salida 2



130

3 6 2



Problema C

Lema Técnico

Autor: Pablo Reina Jiménez

Lema 1.28 (Lema Técnico) Sea \mathbb{X} un k -espacio vectorial de dimensión finita y sean x_1, \dots, x_n vectores linealmente independientes de \mathbb{X} . Entonces, existe un $c > 0$ tal que para cualesquiera escalares $\alpha_1, \dots, \alpha_n$ de k ,

$$\left\| \sum_{i=1}^n \alpha_i x_i \right\| \geq c \sum_{i=1}^n |\alpha_i|$$

Ejemplo de lema técnico real extraído de un texto de análisis funcional. Su inclusión es meramente ilustrativa y no guarda relación con el problema.

En matemáticas, la mayoría de los teoremas importantes no aparecen de la nada. En su lugar, se construyen pacientemente a partir de otros resultados más pequeños, muchas veces aparentemente irrelevantes. Estos resultados intermedios, conocidos como lemas técnicos, suelen consistir en fórmulas poco elegantes, difíciles de interpretar y, en ocasiones, carentes de cualquier significado intuitivo. Sin embargo, su verdadero valor no reside en su belleza, sino en el hecho de que permiten que demostraciones mucho más grandes y complejas puedan finalmente cerrarse.

Arnau está cursando la asignatura de Diferenciación de Funciones de Varias Variables y se ha encontrado con uno de estos lemas. El lema afirma que es posible aproximar el valor de la derivada de una función definida en un intervalo discreto utilizando únicamente información básica del intervalo.

Se consideran los siguientes valores:

- n_1 : inicio del intervalo,
- n_2 : final del intervalo,
- n_{12} : valor de la función en el punto medio del intervalo.

Según el lema, la derivada aproximada \hat{D} se calcula mediante la siguiente expresión:

$$\hat{D} = \left\lfloor \frac{(n_1 + 1)(n_2 + 1)}{n_{12} + 1} - 1 \right\rfloor$$

donde $\lfloor x \rfloor$ es el entero más cercano igual o menor a x .

Arnau sospecha que la fórmula carece de cualquier tipo de sentido matemático. Sin embargo, su profesor no da su brazo a torcer e insiste en que la expresión es correcta y debe aplicarse tal y como está definida.

Resignado, Arnau decide completar el boletín de problemas utilizando esta fórmula sin cuestionarla.

Entrada

La entrada consiste en una única linea con los enteros n_1, n_2, n_{12} en ese orden ($0 \leq n_1, n_2, n_{12} \leq 10000$).

Salida

Imprime el valor resultante al aplicar la fórmula.

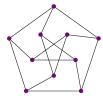
Ejemplos

Entrada

15 18 11

Salida

24



Problema D

El negocio definitivo

Basado en: *The SetStack Computer [NWERC 2006 B]*



En las polvorientas oficinas de Gadgetzan, el ingeniero goblin **Jefazo Noggenfogger** ha descubierto el negocio definitivo: vender «Aire Premium». Los aventureros compran cualquier cosa si es lo suficientemente rara, así que ha diseñado una máquina que empaqueta... ¡La nada absoluta!

El **«Saqueatrón 3000»** trabaja con la materia prima más barata del universo: el «Aire», una caja vacía. La genialidad del invento es su recursividad: puede meter una caja vacía dentro de otra caja, y esa caja dentro de otra, creando una infinidad de «paquetes» colecciónables cada vez más complejos y valiosos ¡Todo a partir de la nada!

El Saqueatrón opera sobre una **pila de estos contenedores**. Las operaciones programadas en la consola de mando son:

- **PUSH**: Crea una caja vacía y la pone en la cima.
- **DUP**: Duplica la caja de la cima. La nueva caja queda sobre la original.
- **UNION**: Toma las dos cajas superiores y vuelca su contenido en una nueva. Si las dos cajas tenían elementos iguales se eliminan los duplicados para ahorrar en materiales.
- **INTERSECT**: Toma las dos cajas superiores y crea una nueva solo con los elementos que hubiese duplicados en las dos. La nueva solo tiene una copia de estos elementos en común.
- **ADD**: Toma la caja de la cima (*A*) y la siguiente (*B*). Mete la caja *A* dentro de *B*. El resultado es el contenedor *B* con un «bulto» más: la caja *A*.

Para ilustrar la locura de esta tecnología, imagina que el escáner detecta que en la cima hay una caja *A* que contiene «Aire» y una caja con «Aire» ($A = \{\{\}, \{\{\}\}\}$), y justo debajo hay un contenedor *B* con «Aire» y una caja con una caja con «Aire» ($B = \{\{\}, \{\{\{\}\}\}\}$).

Si ejecutamos las operaciones sobre estos contenedores:

- **UNION**: El Saqueatrón quita *A* y *B* de la pila, y a cambio introduce una caja con tres cosas («Aire», caja con «Aire» y caja con caja con «Aire») ($\{\{\}, \{\{\}\}, \{\{\{\}\}\}\}$).
- **INTERSECT**: El Saqueatrón quita *A* y *B* de la pila, y a cambio introduce una caja con «Aire» ($\{\{\}\}$).
- **ADD**: El Saqueatrón mete la caja *A* dentro de la *B* y la añade a la pila ($\{\{\}, \{\{\{\}\}\}, \{\{\}, \{\{\}\}\}\}$).

Entrada

La primera línea contiene un natural T ($0 \leq T \leq 5$) que indica el número de casos de prueba. La primera línea de cada caso de prueba contiene el número de operaciones N ($0 \leq N \leq 2000$).

A continuación siguen N líneas, cada una conteniendo uno de los cinco comandos. Los goblins se han esforzado en depurar las instrucciones para que el Saqueatrón nunca necesite sacar contenedores de la pila y se la encuentre vacía.



Salida

Para cada operación especificada en la entrada, habrá una línea de salida que consiste en un solo natural. Este natural es la cantidad de elementos que tenga el contenedor en la cima de la pila, después de ejecutar el comando correspondiente. No es necesario contar el número de sub-elementos.

Después de cada caso de prueba, debe haber una línea con *** (tres asteriscos).

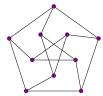
Ejemplo

Entrada

```
2
9
PUSH
DUP
ADD
PUSH
ADD
DUP
ADD
DUP
UNION
5
PUSH
PUSH
ADD
PUSH
INTERSECT
```

Salida

```
0
0
1
0
1
1
2
2
2
***
0
0
1
0
0
***
```



Problema E

Operación Anunnakis

Basado en: Biodiversity [SWERC 2019 B]

Rocío es una estudiante de doctorado en la Facultad de Biología de la Universidad de Sevilla. Actualmente participa en un proyecto de «Ciencia Ciudadana» a nivel europeo, donde voluntarios de todo el continente registran avistamientos de fauna urbana a través de una aplicación móvil.

En el laboratorio del campus de Reina Mercedes, Rocío se encarga de analizar los datos brutos. Como es un proyecto internacional coordinado desde Bruselas, todos los registros están en **inglés**. A estas alturas del día, después de ocho horas picando datos y tres cafés de máquina, Rocío ya no sabe si un **frog** es una rana o un estado de ánimo.

Ella tenía unas ganas locas de irse con sus amigos al **Anunnakis** a echarse un futbolín y tomar algo, pero su supervisora le acaba de mandar una tarea de extrema urgencia que ha truncado sus planes. Su misión es detectar si en algún ecosistema local hay una especie se esté volviendo dominante. Según el protocolo, una especie se considera dominante si el número de registros de dicha especie es **estrictamente mayor** que la suma de todos los demás registros juntos.

¿Puedes ayudar a Rocío a procesar estos listados internacionales para que pueda cerrar el portátil de una vez y llegar a tiempo a la siguiente partida de futbolín?

Entrada

La entrada consiste en las siguientes líneas:

- En la primera línea: un natural N ($1 \leq N \leq 2 \times 10^5$).
- En cada una de las siguientes N líneas: la especie de un animal como una cadena de longitud máxima 20, conteniendo solo caracteres alfanuméricos ASCII.

Salida

El nombre de la especie dominante, si la hay, o la cadena «NONE» en caso contrario.

Ejemplos

Entrada 1

```
3
frog
fish
frog
```

Salida 1

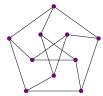
```
frog
```

Entrada 2

```
4
cat
mouse
mouse
cat
```

Salida 2

```
NONE
```



Problema F

Crucigrama sevillano

Basado en: Crosswords [SWERC 2018 C]



Figura 5: Fachada trasera del ayuntamiento de Sevilla

Para atraer a más turistas a Sevilla, el alcalde quiere organizar un gran espectáculo de luz y sonido nocturno en la fachada trasera del ayuntamiento. Se le ha ocurrido mostrar letras gigantes a través de las ventanas, una por ventana, de modo que formen palabras tanto horizontal como verticalmente.

En la fachada, las ventanas están organizadas en una cuadrícula rectangular con N filas y M columnas. Se ha preparado una lista de A palabras con N letras cada una y otra lista de B palabras con M letras cada una. El alcalde necesita nuestra ayuda para saber cuántas formas hay de mostrar simultáneamente N palabras de longitud M horizontalmente y M palabras de longitud N verticalmente en esa cuadrícula.

Entrada

La entrada consiste en lo siguiente:

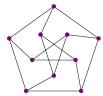
- La primera línea contiene dos enteros N y A separados por un espacio.
- La segunda línea contiene dos enteros M y B separados por un espacio.
- Las siguientes A líneas contienen palabras con N letras, una por línea.
- Las siguientes B líneas contienen palabras con M letras, una por línea.

Límites:

- La entrada satisface $2 \leq N, M \leq 4$ y $1 \leq A \times B \leq 4 \times 10^9$.
- Ninguna de las listas contiene palabras duplicadas.
- Las palabras consisten en letras minúsculas (a-z) del diccionario inglés.
- Las palabras de la primera lista se mostrarán verticalmente, de arriba a abajo.
- Las palabras de la segunda lista se mostrarán horizontalmente, de izquierda a derecha.
- La misma palabra se puede usar varias veces para construir una cuadrícula (es decir, en varias columnas si pertenece a la primera lista o en varias filas si pertenece a la segunda).
- Cuando $N = M$, no está permitido usar palabras de la primera lista horizontalmente (a menos que aparezcan también en la segunda lista) o palabras de la segunda lista verticalmente (a menos que aparezcan también en la primera lista).

Salida

La salida debe consistir en una única línea con el número total de cuadrículas distintas que se pueden formar.



Ejemplos

Entrada 1

```
3 4
4 5
war
are
yes
sat
says
area
test
ways
rest
```

Salida 1

```
2
```

Explicación del ejemplo 1

Las soluciones son:

$$\begin{pmatrix} s & a & y & s \\ a & r & e & a \\ t & e & s & t \end{pmatrix} \quad \begin{pmatrix} w & a & y & s \\ a & r & e & a \\ r & e & s & t \end{pmatrix}$$

Entrada 2

```
3 7
3 7
ran
age
now
its
the
set
ago
ran
age
now
its
the
set
ago
```

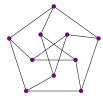
Salida 2

```
2
```

Explicación del ejemplo 2

Las soluciones son:

$$\begin{pmatrix} i & t & s \\ t & h & e \\ s & e & t \end{pmatrix} \quad \begin{pmatrix} r & a & n \\ a & g & o \\ n & o & w \end{pmatrix}$$



Problema G

Saber y Ganar

Autor: Juan Antonio Álvarez García (editado por Pablo Reina Jiménez)



Figura 6: El eterno presentador del programa

En el aclamado programa de televisión «Saber y Ganar» se quiere incluir una nueva prueba. Consiste en averiguar cuántos valores enteros positivos distintos (estrictamente mayores que 0) se pueden obtener aplicando las cuatro operaciones aritméticas básicas ($+$, $-$, $*$, $/$) a los números de un conjunto de tamaño K , que cambia cada día.

Para esta prueba, las operaciones deben aplicarse estrictamente una a una, sin permitir paréntesis ni ningún otro mecanismo que altere el orden de evaluación. Es decir, se comienza con un primer número, se aplica una operación con el segundo número, el resultado obtenido se combina con el tercer número mediante la siguiente operación y así sucesivamente hasta utilizar todos los números del conjunto.

Todos los números del conjunto deben utilizarse una única vez. El orden de los números y las operaciones aritméticas que se realizan en cada paso puede elegirse libremente. Se permite que durante el proceso aparezcan fracciones, siempre que el resultado final sea un entero estrictamente positivo.

Por ejemplo, para obtener el valor 5 a partir de las cifras 4, 1, 2, 3, una posible elección es el orden 4, 1, 2, 3 y las operaciones $*$, $/$, $+$, lo que da lugar a la evaluación secuencial: $\frac{4*1}{2} + 3 = 5$.

Entrada

La entrada consiste en lo siguiente:

- La primera línea contiene un entero N ($1 \leq N \leq 10$) indicando el número de casos de prueba.
- Para cada caso de prueba:
 - Una línea con un entero K indicando el tamaño del conjunto ($5 \leq K \leq 9$).
 - Una línea con K dígitos que forman el conjunto a estudiar.

Salida

La salida debe contener una línea por cada caso de prueba con el número de valores enteros positivos distintos que pueden expresarse siguiendo las reglas del problema.

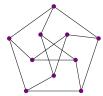
Ejemplo

Entrada

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

Salida

```
31
97
67
```



Problema H Pie

Basado en: Pie [NWERC 2006 C]

Se acerca mi cumpleaños y tradicionalmente sirvo tarta. No solo una, no. He preparado N tartas, de varios sabores y tamaños. F de mis amigos vendrán a la fiesta y cada uno de ellos recibirá un trozo de una única tarta. Podría darles varios trozos de distintas tartas, pero entonces perdería emoción elegir la ideal para cada uno.

Mis amigos son muy molestos y si uno de ellos recibe un trozo más grande que los demás, empiezan a quejarse. Por lo tanto, todos ellos deben recibir trozos del mismo tamaño, incluso si esto lleva a que se desaproveche algo de tarta. Por supuesto, yo también quiero un trozo de tarta para mí y ese trozo también debe ser del mismo tamaño.

¿Cuál es el mayor tamaño en el que podemos cortar los trozos de tarta sin que nadie se queje? Ten en cuenta que todas las tartas tienen forma cilíndrica y altura 1, pero sus radios pueden ser diferentes. Si el tamaño es correcto, no hay problema en que un trozo coincida exactamente con una tarta completa.

Entrada

La entrada comienza con un entero positivo en una línea propia indicando el número de casos de prueba. Luego, para cada caso de prueba:

- Una línea con dos enteros N y F con $1 \leq N, F \leq 10000$, el número de tartas y el número de amigos, respectivamente.
- Una línea con N enteros r_i con $1 \leq r_i \leq 10000$, los radios de las tartas.

Salida

Para cada caso de prueba, imprime una línea con el mayor volumen posible para los trozos de tarta V . La respuesta debe darse como un número de punto flotante con un error absoluto máximo de 10^{-3} .

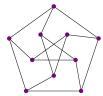
Ejemplo

Entrada

```
3
3 3
4 3 3
1 24
5
10 5
1 4 2 3 4 5 6 5 4 2
```

Salida

```
25.1327
3.1416
50.2655
```



Problema I

El Protocolo del Alcázar

Basado en: *Alphabetical Aristocrats [NWERC 2024 A]*



Figura 7: Real Alcázar de Sevilla

La luz de la tarde cae sobre los naranjos del Real Alcázar de Sevilla. Como secretario de la Capitanía General de Andalucía, se te ha encomendado una tarea de suma importancia diplomática: organizar la recepción de la Gran Delegación Comercial de los Países Bajos, que ha remontado el Guadalquivir para negociar los aranceles del comercio de lana y aceite.

El ambiente en la ciudad es vibrante, pero el protocolo es lo primero. Para evitar un conflicto diplomático en el banquete que se celebrará esta noche, debes organizar a los ilustres invitados holandeses siguiendo estrictamente sus propias costumbres heráldicas, ya que son especialmente sensibles con el orden de sus linajes.

Los embajadores han sido claros: en su tierra, los apellidos no se ordenan por la primera letra de la firma, sino por la *raíz del linaje*. Esto significa que debes ignorar las preposiciones o artículos en minúscula (como «van den» o «de»), empezando a comparar desde la *primera letra mayúscula* que aparezca en el apellido.

Tu tarea es ordenar lexicográficamente esta lista de nombres extranjeros para que el maestro de ceremonias pueda anunciarlos sin que ningún aristócrata de Utrecht o Ámsterdam se sienta ofendido en suelo andaluz.

El orden se determina lexicográficamente:

Espacio en blanco < Apóstrofe ' < Letras mayúsculas A-Z < Letras minúsculas a-z

van den Heken < van den Hecken the Younger

Entrada

La entrada consiste en:

- Una línea con un entero N ($1 \leq N \leq 1000$), el número de apellidos.
- N líneas, cada una con una cadena S ($1 \leq |S| \leq 50$), uno de los apellidos.

Los apellidos consisten en letras inglesas, espacios y apóstrofes (A-Z, a-z, " ", "'"). Se garantiza que la parte que comienza con la primera letra mayúscula es única. Los nombres no tienen espacios iniciales, finales ni consecutivos.



Salida

Imprime la lista de apellidos, ordenada según las reglas holandesas.

Ejemplos

Entrada 1

```
7
Bakker
van der Steen
fakederSteenOfficial
Groot Koerkamp
van den Hecken the Younger
de Waal
van 't Hek
```

Salida 1

```
Bakker
Groot Koerkamp
van den Hecken the Younger
van 't Hek
van der Steen
fakederSteenOfficial
de Waal
```

Entrada 2

```
5
aep Ceallach
var Emreis
an Gleanna
Terzieff Godefroy
of Rivia
```

Salida 2

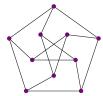
```
aep Ceallach
var Emreis
an Gleanna
of Rivia
Terzieff Godefroy
```

Entrada 3

```
7
van den Brand
Brand 'Heek
den Brand Heek
van Brand heek
der Brandheek
DeN BRAAnD hEeK
den brandHeek
```

Salida 3

```
van den Brand
Brand 'Heek
den Brand Heek
van Brand heek
der Brandheek
DeN bRAnD hEeK
den brandHeek
```



Problema J

Problemas de comunicación

Basado en: Jumbled Communication [NWERC 2015 J]

Adam, ha comprado recientemente una Raspberry Pi y algunos equipos, incluyendo un sensor de temperatura inalámbrico y un receptor de 433MHz para recibir las señales que envía el sensor. Como se le da muy bien la electrónica, en un abrir y cerrar de ojos ya tenía preparado el receptor para obtener la información proporcionada del sensor. La decepción llegó al comprobar los datos recibidos... ¡no entendía nada!

Después de un par de horas investigando en internet, encontró un documento que explica el origen del problema. Resulta que su sensor meteorológico ofusca los datos que envía, para evitar que los usuarios lo utilicen con productos de otros fabricantes (¡Ni que fuese marca Apple!). Afortunadamente, el documento también describe cómo el sensor ofusca su comunicación.

El documento indica que el sensor aplica la expresión $x \wedge (x \ll 1)$ a cada byte enviado. Para estos cálculos, ten en cuenta que cada byte consta siempre de 8 bits (ej. el número 2 se representa como 00000010).

- El operador \wedge es el XOR bit a bit. Por ejemplo, $10110000 \wedge 01100100 = 11010100$.
- El operador \ll es un desplazamiento de bits a la izquierda (no circular). Por ejemplo, $10111001 \ll 1 = 01110010$. Al hacer $x \ll j$, los bits de x se mueven j pasos a la izquierda, los bits más significativos de x se descartan, y se añaden j ceros como los bits menos significativos del resultado.

Para que la Raspberry Pi de Adam interprete correctamente los bytes enviados por el sensor meteorológico, la transmisión necesita ser descifrada. Sin embargo, Adam no es un gran programador, así que te ha pedido ayuda.

¿Puedes ayudar a Adam implementando el algoritmo de descifrado?

Entrada

La entrada consiste en:

- Una línea con un entero N ($1 \leq N \leq 10^5$), el número de bytes en el mensaje enviado por el sensor meteorológico.
- Una línea con N enteros b_1, \dots, b_N ($0 \leq b_i \leq 255$), los valores de byte del mensaje.

Salida

Imprime n valores de byte (en codificación decimal), el mensaje descifrado.

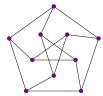
Ejemplo

Entrada

```
5
58 89 205 20 198
```

Salida

```
22 55 187 12 66
```



Problema K

El archivo de indias

Autor: Pablo Reina Jiménez



Figura 8: Exterior del Archivo de Indias

El Archivo General de Indias, situado en Sevilla, custodia uno de los mayores fondos documentales del mundo sobre la administración española en América y Filipinas entre los siglos XV y XIX. En sus estanterías se conservan millones de manuscritos, informes, cartas, mapas y registros oficiales, redactados a lo largo de varios siglos.

Un archivista del Archivo de Indias está digitalizando parte de este fondo para facilitar la labor de los historiadores. De cada manuscrito tenemos dos valores: el *año* en el que fue redactado y su *calidad histórica*, un número entero que mide el estado de conservación y la relevancia del documento.

Cada día, el archivista recibe numerosas consultas de historiadores que investigan períodos muy concretos, solicitando manuscritos de un año concreto con una calidad mínima. Buscar dichos manuscritos a mano requiere mucho tiempo, por lo que el necesita tu ayuda para diseñar un sistema eficiente.

Entrada

1. La primera línea contiene un natural T ($1 \leq T \leq 25$), el número de casos de prueba.
2. Para cada caso de prueba:
 1. Una línea con un natural N ($1 \leq N \leq 2 * 10^5$), el número de manuscritos.
 2. Las siguientes N líneas contienen dos naturales: A ($1400 \leq A \leq 1900$) y C ($1 \leq C \leq 10^9$), indicando el año y la calidad histórica de un manuscrito.
 3. Una línea con un natural Q ($1 \leq Q \leq 2 * 10^5$), el número de consultas.
 4. Las siguientes Q líneas contienen dos naturales X y K , que representan el año consultado y la calidad mínima requerida.

Salida

Para cada consulta, imprime:

- **SI** si existe al menos un manuscrito del año X con calidad $\geq K$.
- **NO** en caso contrario.



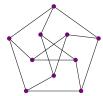
Ejemplo

Entrada

```
1
5
1400 10
1500 3
1500 8
1600 100
1600 1
4
1500 5
1500 9
1600 50
1700 1
```

Salida

```
SI
NO
SI
NO
```



Problema L

Pautas de seguridad

Basado en: Assigning Workstations [NWERC 2015 A]

Penélope forma parte del equipo de administración de la supercomputadora recién construida. Su trabajo es asignar estaciones de trabajo a los investigadores que vienen a ejecutar sus cálculos en la supercomputadora.

Penélope es muy perezosa y odia desbloquear máquinas para los investigadores que van llegando. Puede desbloquear las máquinas de forma remota desde su escritorio, pero no siente que esta tarea menor esté a la altura de sus cualificaciones. Si decidiera ignorar las pautas de seguridad, simplemente podría pedir a los investigadores que no bloqueen sus estaciones de trabajo cuando se vayan, y luego asignar a los nuevos investigadores a las estaciones de trabajo que ya no se usan pero que siguen desbloqueadas. De esa manera, solo necesita desbloquear cada estación de trabajo para el primer investigador que la use, lo cual sería una gran mejora para Penélope.

Desafortunadamente, las estaciones de trabajo no utilizadas se bloquean automáticamente si no se usan durante más de m minutos. Después de que una estación de trabajo se ha bloqueado, Penélope tiene que desbloquearla nuevamente para el siguiente investigador que la use.

Dado el horario exacto de llegada y salida de los investigadores, ¿puedes decirle a Penélope cuántos desbloqueos puede ahorrarse pidiendo a los investigadores que no bloqueen sus estaciones de trabajo al irse y asignando a los investigadores que llegan a las estaciones de trabajo de manera óptima?

Puedes asumir que siempre hay suficientes estaciones de trabajo disponibles.

Entrada

La entrada consiste en:

- Una línea con dos enteros N ($1 \leq N \leq 300000$), el número de investigadores, y M ($1 \leq M \leq 10^8$), el número de minutos de inactividad antes de que una estación de trabajo se bloquee.
- N líneas, cada una con dos enteros A y S ($1 \leq A, S \leq 10^8$), que representan a un investigador que llega después de A minutos y se queda exactamente S minutos.

Salida

Imprime el número máximo de desbloqueos que Penélope puede ahorrarse saltándose las pautas de seguridad.

Ejemplos

Entrada 1

```
3 5
1 5
6 3
14 6
```

Salida 1

```
2
```

Entrada 2

```
5 10
2 6
1 2
17 7
3 9
15 6
```

Salida 2

```
3
```