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

Programa-Me 2019
Final Nacional
Problemas

Patrocinado por

Ejercicios realizados por

Universidad Complutense
de Madrid

Realizado en la **Facultad de Informática (U.C.M.)**
14 de junio de 2019

14 de junio de 2019
http://www.programa-me.com

Listado de problemas

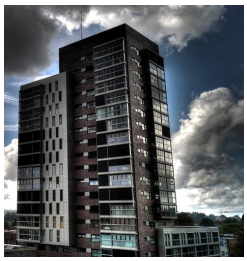
Autores de los problemas:

- Marco Antonio Gómez Martín (Universidad Complutense de Madrid)
- Pedro Pablo Gómez Martín (Universidad Complutense de Madrid)
- Antonio Pérez-Aradros Herrero (IES Comercio)

Revisores:

- Isabel Pita Andreu (Universidad Complutense de Madrid)
- Alberto Verdejo López (Universidad Complutense de Madrid)

A
¿Podemos empezar?



Siempre que hay reunión de vecinos en el portal, tenemos el mismo problema. Para poder empezar la reunión tiene que haber *cuórum*, lo que significa que tiene que estar presente una persona de, al menos, la mitad de las viviendas. Para saber si se ha alcanzado, no basta con contar cuántos somos, porque de algunas viviendas bajan a la reunión más de una persona, de modo que hay que apuntar de donde es cada uno de los asistentes para echar la cuenta.

El secretario de la comunidad es el responsable de decidir si se puede o no empezar, pero suele hacerse un poco de lío con la lista y se tiene la sospecha de que no siempre lo hace bien.

Entrada

El programa recibirá, por la entrada estándar, varios casos de prueba. Cada uno comienza con tres números, $1 \leq P \leq 30$, $1 \leq L \leq 26$ y $1 \leq A \leq 1.000$ indicando respectivamente el número de pisos del portal, el número de letras (viviendas) por piso, y el número de asistentes a la reunión.

A continuación aparece la vivienda de cada uno de los A asistentes. Una vivienda se especifica con el número de piso (entre 1 y P) y la letra, separados por un espacio. Se utilizan únicamente las letras del alfabeto inglés en mayúscula desde la '`A`' hasta la última en función del valor de L.

La entrada termina con tres ceros.

Salida

Por cada caso de prueba el programa escribirá `EMPEZAMOS` si hay al menos una persona de la mitad de las viviendas, y `ESPERAMOS` en otro caso. Si el número de viviendas es impar, debe utilizarse la mitad *por exceso*.

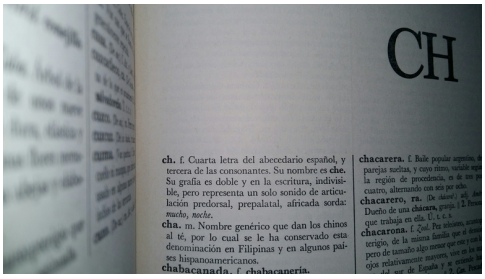
Entrada de ejemplo

```
4 1 3
1 A 2 A 1 A
1 5 3
1 E 1 E 1 C
0 0 0
```

Salida de ejemplo

```
EMPEZAMOS
ESPERAMOS
```

B
La desconfianza de la ASALE



Cuando en el lejano 1754 se publicó la *Ortografía de la lengua española*, los *digrafos ch* y *ll* (conocidos como *che* y *elle*) fueron considerados *letras*. Es por ello que cuando en 1803 se publicó la cuarta edición del *Diccionario de la lengua española* estos dos digrafos tuvieron secciones separadas dentro de la ordenación alfabética[1]. En particular, los niños en la escuela aprendían que el alfabeto era *a*, *b*, *c*, **ch**, *d*, *e*, *f*, *g*, *h*, *i*, *j*, *k*, *l*, **ll**, *m*, *n*, *ñ*, *o*, *p*, *q*, *r*, *s*, *t*, *u*, *v*, *w*, *x*, *y*, y *z*[2].

La consecuencia práctica a la hora de buscar en esos diccionarios era que, por ejemplo, "chocolate" aparecía *después* que "cuchillo" a pesar de que su segundo carácter (*h*) va antes en el alfabeto latino que el segundo carácter de "cuchillo" (la *u*). Ese mismo funcionamiento se da cuando los digrafos aparecen en medio de una palabra. Por ejemplo "cacique" aparecía antes que "cacho".

Durante el X congreso de la Asociación de Academias de la Lengua Española (ASALE) celebrada en 1994, la historia dio un vuelco. En aquellas reuniones en Madrid se acordó reordenar los digrafos y colocarlos en el lugar que determina el alfabeto latino clásico por lo que la siguiente edición del diccionario (2001) no tuvo secciones separadas para ellos y "cacho" apareció antes que "cacique". Por contar la historia completa, en 2010 se publicó una nueva ortografía de la lengua española, en donde se quitó a los digrafos la categoría de letra.

Aunque la ASALE seguramente justificará su decisión por otros motivos, nuestras sospechas son que desconfía por completo de los informáticos que tienen que programar los algoritmos de ordenación de cadenas en nuestro idioma. Y es que no resulta fácil. Además de manejar correctamente la *ñ*, vocales con tilde y las diéresis, teníamos que controlar los digrafos.

Es hora de demostrar a la ASALE que se equivocaba.

Entrada

La entrada está compuesta por distintos casos de prueba, cada uno en una línea.

En cada línea aparecen dos palabras distintas con letras en minúscula de no más de 10 caracteres. Para no tentar mucho a la suerte (y tener que terminar dando la razón a la ASALE) ninguna de las palabras tiene *ñ*, tildes o diéresis. Las palabras, eso sí, no tienen por qué ser válidas en nuestro idioma (podrían, por ejemplo, no tener vocales), aunque se garantiza que nunca aparecerán más de dos *l*'s seguidas.

Salida

Por cada caso de prueba se escribirá la palabra que debería aparecer antes en un diccionario anterior a la supresión de los digrafos.

Entrada de ejemplo

```
allanar alunizar
cacho cacique
cuchillo chocolate
peruano peru
```

[1] Es curioso que se tardara tanto tiempo en realizar esta separación; las tres ediciones anteriores del diccionario también fueron publicadas después de 1754 pero mantenían integradas en las secciones de la *c* y la *l* las palabras que comenzaban por *ch* y *ll*.

[2] Algunos incluso añadían, incorrectamente, la *rr* después de la *r*.

Salida de ejemplo

```
alunizar
cacique
cuchillo
peru
```

C
Aburrimiento en la autopista

De los 700 kilómetros que separan su casa de su lugar de vacaciones, la inmensa mayoría son de autopista. Edu Ermo no tiene claro si eso es una bendición o una maldición. Es verdad que las autopistas son cómodas y seguras, pero también son terriblemente aburridas, y Edu tiene que luchar duro contra el sopor. Por más que lo intenta, no consigue que su mujer, Sue Ñosmil, le sustituya al volante. A ella le ocurre exactamente lo mismo y se pasa los 700 kilómetros durmiendo en el sitio del copiloto. Solo la voz del GPS le hace, muy de vez en cuando, algo de compañía.

Para las próximas vacaciones ha ideado una estrategia. Se va a entretener mirando las matrículas de todos los coches con los que se cruce y comparándolas con la del suyo propio para contar cuántos coches son más antiguos y cuántos más nuevos. Quiere matar dos pájaros de un tiro: entretenerse para no dormirse, y tener una prueba de que su coche se está quedando viejo y poder así convencer a Sue de que hay que comprar otro.

Entrada

La entrada comienza con una línea que contiene el número de casos de prueba que vendrán a continuación.

Cada caso de prueba consta de una línea con una serie de matrículas. La primera de ellas se corresponde con la matrícula del coche de Edu y el resto con la de los coches que se cruza (se garantiza que Edu no se cruzará consigo mismo). La línea termina con un `0`.

El formato de todas las matrículas es igual: comienza con un número con cuatro dígitos (desde el 0000 hasta el 9999) al que siguen 3 letras en mayúsculas que nunca serán vocales, Q ni $\tilde{N}$.

Salida

Para cada caso de prueba se escribirá una línea con dos números separados por un espacio indicando, respectivamente, cuántos coches se han visto más antiguos que el de Edu, y cuántos más modernos. Edu no tiene buena memoria, por lo que si la misma matrícula aparece varias veces en la entrada deberá contarse tantas veces como lo haga.

Recuerda que un coche es más antiguo que otro si las tres letras de su matrícula son menores lexicográficamente o si, en caso de empate, el número es menor.

Entrada de ejemplo

```
2
5555CPP 5558CPB 5554CPX 0
2019PRG 3030PRG 3030PRG 0
```

Salida de ejemplo

```
1 1
0 2
```

D
Compe y Tencia

Hasta hace bien poco la vida era sencilla en mi pequeño pueblo situado en la serranía de Cuenca. Un bar, un colmado y una farmacia eran los únicos negocios presentes en la localidad, que disfrutaban de sus respectivos monopolios poniendo unos precios quizá un poco excesivos pero merecidos. Al fin y al cabo que te traigan desde la capital los productos que quieres hay que pagarlo.

Sin embargo, hace unos pocos meses la vida se torció para Doña *Compe*, la dueña del colmado. La culpable es la Señora *Tencia*, que ha decidido abrir una tienda en la otra esquina del pueblo.

Pero el trastorno no se queda en *Compe* y *Tencia*. También a nosotros los aldeanos nos ha trastocado la paz que disfrutábamos desde hacía generaciones. Ahora cuando vamos a comprar, tenemos que decidir a qué tienda vamos.

Mi objetivo, como el de todos, es pagar lo menos posible. Para eso he cogido los catálogos de precios de cada una de las tiendas y he planificado mis siguientes compras. Para cada una de ellas he establecido a cuánto ascendería mi cuenta en ambas tiendas, de forma que me ayude a decidir a cuál de las dos voy en cada momento. Eso sí, en aras de la paz y armonía de nuestro pueblo, me he marcado un número mínimo de visitas a cada una de las tiendas, para que ninguna de las dos empresarias se moleste conmigo (¡las dos son amigas de la infancia!).

Entrada

La entrada está compuesta por distintos casos de prueba, cada uno representando la planificación de una serie de compras.

Cada caso de prueba comienza con el número N de compras planificadas (como mucho 10.000). En la siguiente línea aparecen dos números, C y T, indicando el número mínimo de veces que tengo que ir a la tienda de *Compe* y *Tencia* respectivamente para que no se enfaden ($C + T \leq N$). Por último aparecerán dos líneas con N números cada una, indicando el precio de la cesta de cada compra en la tienda de *Compe* y *Tencia* respectivamente. Se garantiza que todas las cantidades son enteras entre 1 y 1.000.

La entrada termina con un 0, que no debe procesarse.

Salida

Por cada caso de prueba se escribirá una línea con el gasto mínimo en las compras.

Entrada de ejemplo

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

Salida de ejemplo

```
12
18
```

E
Multas desde el cielo

La Dirección General de Tráfico ha lanzado al aire infinidad de *drones* de larga autonomía para que vigilen desde el aire la red vial. Suspendidos en el aire durante días sin repostar, vigilan tramos de carretera a la caza de imprudentes conductores que creen estar a salvo al no divisar radares en tierra.

A veces, los drones se despistan y terminan colocándose de manera poco eficiente, solapándose parcialmente en los tramos que vigilan. Para mejorar el rendimiento del sistema (es decir, el número de multas impuestas) se quiere empezar detectando esas situaciones.

Entrada

La entrada comienza con un primer número que indica cuántos casos de prueba deberán ser procesados. Un caso de prueba está formado por dos parejas de números enteros positivos, cada una indicando el tramo cubierto por un dron para una determinada carretera. Los tramos se especifican con su punto kilométrico inicial y final que serán siempre números distintos entre 1 y 10.000.

Salida

Por cada caso de prueba el programa escribirá `SEPARADOS` si los drones vigilan tramos independientes, y `SOLAPADOS` en otro caso.

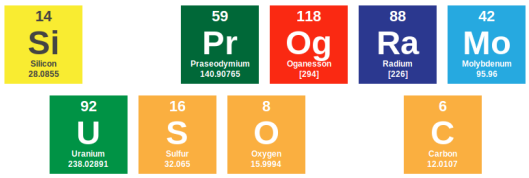
Entrada de ejemplo

```
3
10 20 30 40
40 20 10 30
35 55 70 55
```

Salida de ejemplo

```
SEPARADOS
SOLAPADOS
SEPARADOS
```

F
Escribiendo con átomos



En 1869 el ruso Dmitri Mendeléyev puso orden en el caos y estableció un criterio para colocar en una tabla los elementos químicos teniendo en cuentas las características de sus átomos. Su acierto fue tal que incluso dedujo la existencia de elementos aún no descubiertos y dejó hueco para ellos en su *tabla periódica*.

Cincuenta años después se fundó la Unión Internacional de Química Pura y Aplicada (conocida por sus siglas en inglés, IUPAC) que tiene entre sus cometidos dar nombre y establecer el símbolo químico de los elementos que se han ido descubriendo y sintetizando. En el 150 aniversario de la tabla periódica el número de elementos ascendía a los 118 de la figura.

Tabla periódica de los elementos

H																	He
Li	Be											B	C	N	O	F	Ne
Na	Mg											Al	Si	P	S	Cl	Ar
K	Ca	Sc	Ti	V	Cr	Mn	Fe	Co	Ni	Cu	Zn	Ga	Ge	As	Se	Br	Kr
Rb	Sr	Y	Zr	Nb	Mo	Tc	Ru	Rh	Pd	Ag	Cd	In	Sn	Sb	Te	I	Xe
Cs	Ba		Hf	Ta	W	Re	Os	Ir	Pt	Au	Hg	Tl	Pb	Bi	Po	At	Rn
Fr	Ra		Rf	Db	Sg	Bh	Hs	Mt	Ds	Rg	Cn	Nh	Fl	Mc	Lv	Ts	Og

La	Ce	Pr	Nd	Pm	Sm	Eu	Gd	Tb	Dy	Ho	Er	Tm	Yb	Lu
Ac	Th	Pa	U	Np	Pu	Am	Cm	Bk	Cf	Es	Fm	Md	No	Lr

Como se ve, los símbolos son siempre de una o dos letras que luego se juntan para construir las fórmulas químicas. Por ejemplo, la sal común o *cloruro de sodio* tiene como fórmula `NaCl` (`Na` es el sodio y `Cl` el cloro).

Esta combinación de letras hace que algunas fórmulas terminen formando una palabra válida. Por ejemplo, si ignoramos los números que aparecen en las fórmulas que indican el número de átomos de cada elemento, tenemos que el *carbonato cálcico* forma la palabra *caco* (en realidad $\texttt{CaCO}_3$), mientras que el *sulfato cálcico* forma *caso* ($\texttt{CaSO}_4$).

Si ignoramos el hecho de que no todas las combinaciones de elementos químicos son posibles, dada una frase ¿se puede escribir como una sucesión de símbolos químicos?

Entrada

La entrada está compuesta por varios casos de prueba, cada uno en una línea.

Cada línea contiene una cadena (de al menos 1 y como mucho 1.000 caracteres) con letras del alfabeto inglés tanto mayúsculas como minúsculas. Puede haber varios espacios separándolas aunque se garantiza que no habrá espacios ni al principio ni al final de la línea.

Salida

Por cada caso de prueba se escribirá `SI` si se puede escribir el texto (ignorando los espacios y las mayúsculas) utilizando los símbolos de los 118 elementos químicos disponibles en el 150 aniversario de la tabla periódica, y `NO` en caso contrario.

Entrada de ejemplo

```
Si programo
uso C
y no Java
```

Salida de ejemplo

```
SI
SI
NO
```

G
Trozos divisibles

Hay en Internet un protocolo de transporte llamado UDP (del inglés *User Datagram Protocol* o *protocolo de datagramas de usuario*) que a los profanos en la materia les sorprende cuando lo descubren.

En él, el emisor puede enviar el mensaje sin haber establecido previamente una conexión con el receptor, por lo que es muy *ligero* en cuanto a recursos consumidos. El precio que hay que pagar por esa ligereza es doble: los mensajes enviados pueden no llegar (y no hay forma de saberlo) y además pueden llegar en desorden.

Aunque se puede pensar que esas dos pegas lo convierten en un protocolo inservible, la realidad es que es muy utilizado. No sólo se utiliza para protocolos de la capa de aplicación como DHCP o DNS; también es utilizado para transmisión de audio y vídeo en tiempo real o juegos on-line.

Como ejemplo de que los clientes pueden recuperarse ante los fallos de comunicación que pueda haber, pensemos en un escenario concreto. Imaginemos que el emisor está enviando números grandes que, sabemos, son siempre divisibles por 900. Aún cuando los dígitos lleguen en desorden, es posible saber si, al menos aparentemente, no se ha perdido ningún dígito. Incluso en el caso de perderse alguno, se puede especular sobre cuáles habrán sido.

Entrada

La entrada comienza con una línea que contiene el número de casos de prueba que vendrán a continuación.

Cada caso de prueba es una línea de hasta 100 caracteres con los dígitos (quizá en desorden) recibidos.

Salida

Por cada caso de prueba se escribirá `COMPLETO` si los dígitos recibidos se pueden reordenar de forma que se tenga un número divisible por 900. En otro caso, se escribirán los dígitos que han podido perderse por el camino. Si hay más de un dígito, se escribirán en orden creciente. En caso de haber varias alternativas, se elegirá aquella que requiera menos dígitos.

Entrada de ejemplo

```
4
9
1080
108
08
```

Salida de ejemplo

```
00
COMPLETO
0
01
```

H
El extraño caso de la multiplicación

Si no admitimos números negativos, cuando sumamos dos números a y b, el resultado es siempre mayor o igual que ambos sumandos (será igual si alguno de ellos es 0).

Con la multiplicación, sin embargo, no ocurre lo mismo. Dependiendo de los valores que se multipliquen, el resultado no tiene por qué ser mayor o igual que ellos. De hecho, pueden ocurrir las tres cosas siguientes:

- El resultado es mayor o igual que los dos multiplicandos. Pasa por ejemplo si hacemos 3×5.
- El resultado se queda entre ambos multiplicandos. Por ejemplo al hacer 0.2×10 tenemos como resultado 2, que está entre ambos valores iniciales (sin coincidir con ninguno de los dos).
- El resultado es menor o igual que ambos valores. Por ejemplo en 0.9×0.8, que da 0.72.

Dado un conjunto de números, ¿en cuántas parejas de números se tiene que al multiplicar ambos valores el resultado *no* queda entre ambos?

Entrada

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

Cada caso de prueba comienza con una línea con el número de elementos N que se deben considerar ($2 \leq N \leq 200.000$).

La siguiente línea contiene N números entre 0 y 1.000 con como mucho tres decimales.

La entrada termina con un 0 que no debe ser procesado.

Salida

Por cada caso de prueba se escribirá una única línea con la cantidad de parejas de valores en los que el resultado de multiplicar ambos es o bien mayor o igual o bien menor o igual a ambos.

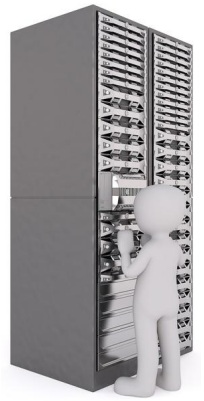
Entrada de ejemplo

```
4
4 5 6 7
4
4 0.1 5 0.1
8
0 1 4 5 6 0.2 0.3 1
0
```

Salida de ejemplo

```
6
2
22
```

Login, logout



Como administrador de una plataforma de aprendizaje *online*, una de tus responsabilidades es controlar la carga de los servidores, para conocer los picos de uso y poder detectar tendencias crecientes que te permitan adquirir nuevo hardware con antelación suficiente.

Eres buen programador pero, por desgracia, tus habilidades de administración de sistemas no son muy destacables. La única monitorización que tienes desplegada es el registro de los instantes en los que los usuarios hacen *login* y *logout* para entrar y salir del sistema. Cada vez que un usuario abre la sesión, escribes en el fichero de bitácora una `I` (de log**i**n). De la misma forma, cuando un usuario se desconecta, escribes una `O` (de log**o**ut).

Lo tienes así desde hace años y nunca has hecho mucho caso al resultado. Pero últimamente el sistema está empezando a ir lento, y has recuperado el fichero para intentar intuir qué está pasando contando los picos de uso.

Entrada

La entrada comienza con un número con la cantidad de casos que vendrán a continuación. Cada caso de prueba es una línea conteniendo únicamente las letras `I` y `O`, indicando que un usuario ha hecho *login* o *logout* respectivamente.

Cada caso de prueba está cogido de una porción de algún fichero de bitácora extraído de las copias de seguridad. Se garantiza que siempre tienen al menos una letra y no más de 10.000. Eso sí, al ser porciones aleatorias no se puede garantizar que el caso de prueba termine después de que todos los usuarios que han hecho *login* hayan hecho el correspondiente *logout*. De hecho, también podría ocurrir al contrario, que haya usuarios que hagan *logout* sin haber visto su *login*.

Salida

Por cada caso de prueba, el programa deberá escribir el máximo número de usuarios que podemos demostrar que han estado autentificados en el sistema simultáneamente.

Entrada de ejemplo

```
3
IIOIOO
IIIIO
O
```

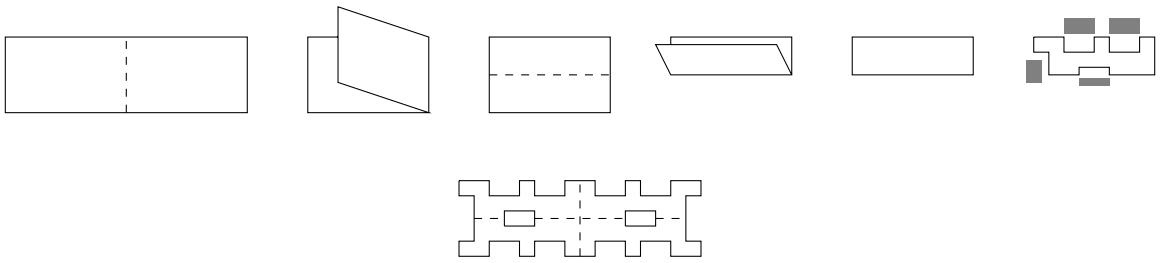
Salida de ejemplo

```
2
4
1
```

J
Concurso de simetrías

Cuando la directora del colegio te pidió que organizaras un concurso de *simetrías con papel* quisiste cambiar de colegio. Pero tras unas pocas horas, ya tienes lanzada la convocatoria:

> “
> Se convoca un concurso de *simetrías con papel*, consistente en realizar formas simétricas utilizando únicamente papel y tijeras. El proceso es el siguiente:
>
> - Coger una hoja de papel rectangular (o cuadrada), y doblarla a lo largo y a lo ancho para formar un rectángulo de forma que quedarán cuatro láminas de papel unidas por dos de los bordes.
> - Utilizando unas tijeras, hacer los cortes que se consideren oportunos a los cuatro bordes para eliminar pequeños trozos de papel.
> - Desdoblar el papel para descubrir la imagen final construida, que será la presentada al concurso.
>
> Se deben cumplir las siguientes normas:
>
> - La simetría entregada debe estar formada por una única pieza de papel.
> - Las tijeras solo deben hacer giros de 90 grados o lo que es lo mismo, los cortes serán siempre paralelos a alguno de los bordes.
>
> ”



Ahora ya solo queda esperar a recibir las piezas presentadas al concurso y comprobar que cumplen las condiciones anteriores.

Entrada

La entrada estará compuesta por distintos casos de prueba, cada uno formado por varias líneas. Cada caso de prueba se corresponde con una foto de una figura presentada al concurso.

La primera línea de cada caso contiene el tamaño de la foto en píxeles, primero el ancho, *tx*, y luego el alto, *ty*. Ninguno de ellos será mayor que 100.

A continuación vendrán *ty* líneas de *tx* caracteres en donde una ‘`X`’ indica que en esa posición hay papel, y un ‘`.`’ debe interpretarse como ausencia del mismo.

Salida

Para cada caso de prueba se escribirá `CORRECTO` si la entrega cumple las restricciones del enunciado y `TRAMPOSO` en caso contrario.

Entrada de ejemplo

```
16 5
XX..X..XX..X..XX
.XXXXXXXXXXXXX.
.XX..XXXXX..XX.
.XXXXXXXXXXXXX.
XX..X..XX..X..XX
10 4
..........
.XX....XX.
.XX....XX.
..........
2 2
X.
..
```

Salida de ejemplo

```
CORRECTO
TRAMPOSO
TRAMPOSO
```