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

Programa-Me 2021
Final Nacional
Problemas

Concurso online *multisitio*

Ejercicios realizados por

Facultad de Informática
Universidad Complutense de Madrid

Coordinado desde la **Facultad de Informática (U.C.M.)**
28 de abril de 2021

Universidad Complutense Madrid

28 de abril de 2021
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)

Revisores:

- Alberto Verdejo López (Universidad Complutense de Madrid)
- Jaime Bocio Sanz (Institut Baix Camp – Reus)
- Anton Dalmau Mines (Institut Baix Camp – Reus)
- Eduard Sorita Calatayud (Institut Baix Camp – Reus)

A
Honor y distribución

El *bridge* es un juego de cartas jugado con la baraja inglesa por cuatro jugadores compitiendo en parejas. Al inicio de cada partida ("*mano*" en terminología del juego), se reparten las 52 cartas (13 a cada jugador), y, durante la fase de *carteo*, los jugadores utilizan una de ellas en cada *baza*. Los equipos compiten por conseguir más bazas que el adversario, siguiendo una serie de reglas que recuerdan al juego del *tute* español. Las necesidades estratégicas y de planificación del *bridge* lo han convertido en el único juego de cartas en ser reconocido como deporte por el Comité Olímpico Internacional, siguiendo la estela de juegos como el ajedrez o el billar.

Aparte del concepto de "*muerto*" (*dummy* en inglés), la característica más significativa del *bridge* es la *subasta*, en la que los equipos aventuran el número de bazas que serán capaces de ganar, a la vez que eligen el *palo del triunfo*.

Cartas del declarante y del muerto en una partida de bridge

Para saber cuánto hacer subir la subasta, es necesario estimar la *fortaleza* de las cartas que se poseen. Para eso, se utilizan los llamados *puntos de honor*, inventados por Milton Work. Esta valoración asigna cuatro puntos a cada *as* (`A`), tres a cada *rey* (`K`), dos a cada *reina* o *dama* (`Q`) y uno a cada *sota*, *jack* o, en francés, *valet* (`J`).

Además, debido al funcionamiento del propio juego, en ocasiones se cuentan también los *puntos de distribución*, que hacen referencia a la distribución general de las cartas de un jugador y su potencial más allá del poder individual de cada carta. Aunque el modo de evaluar este aspecto depende de los lugares, una alternativa extendida es añadir un punto de distribución si se tienen únicamente dos cartas de un palo (*doubleton*), dos si se tiene solo una (*singleton*) y tres si no se tiene ninguna (*void*).

Entrada

El programa recibirá por su entrada estándar una primera línea con un número indicando la cantidad de casos de prueba que vendrán a continuación.

Cada caso de prueba contendrá, en una única línea, la descripción de las 13 cartas de un jugador en una partida de *bridge*. Cada carta se describirá indicando su número y su palo separados por espacio. Las cartas estarán también separadas entre sí por un espacio.

Los números de las cartas serán valores entre 2 y 10, así como las letras `A`, `K`, `Q` y `J` descritas antes. Para los palos se utilizará la letra `C` para *corazones* (♡), `D` para *diamantes* (♢), `P` para *picas* (♠) y `T` para *tréboles* (♣).

Salida

El programa escribirá, en la salida estándar, la valoración total de las cartas de cada caso de prueba. Cada valoración se escribirá en una línea, y contendrá la suma de los puntos de honor y de distribución.

Entrada de ejemplo

```
2
Q C A C 3 T Q T K T 2 D 4 D 5 D 7 D Q D 9 P Q P K P
6 D J P 7 P 6 P 5 P 4 P K C J C 9 C 8 C 4 C 2 C 10 T
```

Salida de ejemplo

```
19
9
```

B
Ordenando el armario

Después de un montón de años sin organizar la ropa, el estado de tu armario ha llegado a niveles de película de terror. Has estado buscando formas de mantener el orden, técnicas para doblar la ropa y cualquier tipo de ayuda que te permita coger una prenda sin que las demás te ataquen saltándote encima.

La primera conclusión a la que has llegado es que necesitas hacer inventario y saber, al menos, si tienes más ropa de verano o de invierno. Te has hecho una lista y ahora toca contar.

Entrada

El programa deberá procesar varios casos de prueba, cada uno en una línea.

Cada caso de prueba es una sucesión de letras "`V`", "`I`" o "`A`" indicando, respectivamente, que una prenda es de verano, de invierno o que sirve para ambas estaciones indistintamente. La sucesión termina con un punto ("`.`").

El último caso de prueba, que no debe procesarse, es una lista sin prendas.

Salida

Para cada caso de prueba el programa escribirá "`VERANO`", "`INVIERNO`" o "`EMPATE`" dependiendo de la estación para la que más ropa tengas.

Entrada de ejemplo

```
V V A .
I V V I A I .
A .
.
```

Salida de ejemplo

```
VERANO
INVIERNO
EMPATE
```

C
El precio de la gasolina

Luis Insuer T. es un pobre desgraciado al que todo le sale mal. Es un hombre que apostó un millón al rojo, y salió negro. Es un hombre que vio como la mujer de sus sueños se casaba con su mejor amigo. Es un hombre que vendió sus acciones justo antes de que subieran.

Pero es un hombre que conduce un Golpagüen Sol. Porque todos, dice, necesitamos en la vida, algo en qué confiar.

Claro que también dice que, desde que lo tiene, la gasolina no ha hecho más que subir. Yo le insisto en que es verdad que la tendencia general es que va subiendo poco a poco, pero también hay veces que baja. Pero él dice que no, que siempre que ha echado gasolina era más cara que la vez anterior.

Yo tengo apuntado el precio diario de la gasolina desde hace un montón de tiempo y sé que tengo razón y a veces baja. Pero con sus precedentes, voy a tener que creerle. Como su Golpagüen Sol no tiene que repostar todos los días, seguro que ha tenido la mala suerte de hacerlo de manera que los precios que *él ha visto* eran siempre crecientes.

Entrada

Cada caso de prueba contiene una primera línea con la cantidad de días (entre 1 y 1.000) que llevo anotando el precio de la gasolina. A continuación, en otra línea, viene el precio de la gasolina esos días, separados por espacio. Los precios vienen expresados en céntimos, y son valores entre 1 y 1.000.000.000.

La entrada termina con un caso sin días, que no debe procesarse.

Salida

Por cada caso de prueba el programa escribirá el mayor número de días que Luis puede haber echado gasolina (no necesariamente consecutivos) sabiendo que siempre ve un precio mayor que el anterior.

Entrada de ejemplo

```
4
1 2 3 4
5
1 2 2 1 10
6
2 3 2 4 8 6
5
8 1 3 2 9
0
```

Salida de ejemplo

```
4
3
4
3
```

D
Minimizando el castigo

A Larry, un inquieto muchacho del Bronx, le pusieron en cierta ocasión como castigo en el Instituto escribir 100 veces "Lawrence Gordon Tesler no volverá a copiar". La frase resultaba francamente irónica pues al tenerla que repetir tantas veces no hacía otra cosa que... copiarla.

Aquel castigo le pareció un sinsentido de modo que decidió buscar un atajo. En aquella época ya existían las fotocopias, aunque por entonces aún se las conocía como *xeroxcopias* debido a la marca que las inventó e hizo populares[1]. Se le ocurrió que podía escribir la frase una sola vez, hacer una copia, recortarla y pegarla justo debajo de la original. Una vez que tuviera las dos juntas, podía hacer una nueva copia, recortarla, y poner, debajo de las anteriores, las dos nuevas frases. Con un poco de paciencia conseguiría tener las 100 frases "escritas" en un momento, y dedicar la tarde a algo más productivo.

Nadie pareció darse cuenta de la trampa, ni siquiera el profesor que, a la larga, marcaría su vida al introducirle en el mundo de la programación. Con el tiempo, ya trabajando en Xerox PARC, recordaría con cariño aquella tarde de fotocopias, tijeras y pegamento de barra que le inspiró para inventar el *copiar y pegar*.

Entrada

La entrada comienza con un número indicando cuántos casos de prueba deberán ser procesados. Cada uno será un número entre 1 y 10^9 indicando cuántas veces tiene Larry que escribir, como castigo, una determinada frase.

Salida

Por cada caso de prueba el programa escribirá el mínimo número de operaciones de fotocopia, recorte y pegado que tendrá que hacer Larry para terminar el castigo completo, partiendo de una única línea, escrita a mano, de la frase.

Entrada de ejemplo

```
4
1
2
3
4
```

Salida de ejemplo

```
0
1
2
2
```

[1] Curiosamente, Larry terminaría trabajando, durante más de 10 años, para Xerox.

E
Peligro por hielo

Mi madre se ha comprado un flamante coche nuevo, y he heredado su viejo carcamal. Le tengo mucho cariño y me hace mucha ilusión ser el dueño del coche que me vio crecer sentado en el asiento de atrás. Pero debo reconocer que le faltan muchas prestaciones.

En particular, hoy virtualmente todos los coches avisan cuando la temperatura desciende más allá de un límite, para que el conductor esté prevenido de la posible existencia de hielo en la calzada. Pero mi antigualla no lo hace, y, como conductor novel, me vendría muy bien en los meses fríos. Para resolverlo, he decidido que, con mi soldador y mucha ilusión, voy a añadirle esa funcionalidad usando un pequeño microcontrolador programable. El problema es que la electrónica se me da bien, pero la programación es otra historia.

Entrada

Cada caso de prueba comienza con un número $1 \leq n \leq 1.000$ indicando cuántas medidas de temperatura ha tomado el dispositivo a lo largo de un determinado viaje. A continuación aparecen n números con esas mediciones, todas entre –20 y 40 y dadas como números enteros.

La entrada termina con un 0.

Salida

Por cada caso de prueba el programa escribirá cuántas veces debe hacer sonar la señal de *peligro por hielo* el microcontrolador. Debe hacerlo cuando la temperatura descienda hasta los 4 grados o menos, siempre que la temperatura haya estado por encima de los 6 desde que se dió el último aviso o no se haya dado ningún aviso por frío todavía.

Entrada de ejemplo

```
4
8 5 4 3
6
7 5 3 1 6 3
6
7 5 3 1 7 3
3
4 2 -1
0
```

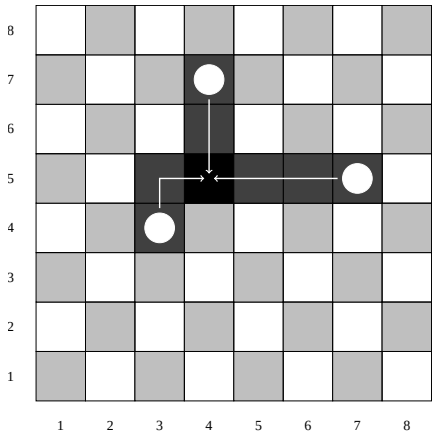
Salida de ejemplo

```
1
1
2
1
```

F
Reunión de torres

En los últimos días, todas las piezas que hay guardadas en el club de ajedrez están un poco inquietas. Parece ser que están cansadas de que los humanos las sobemos mientras dura la partida y luego las devolvamos a sus cajas oscuras y frías.

No tenemos muy claro cómo lo han hecho, pero las torres se han puesto de acuerdo y han decidido tener una reunión en el tablero de ajedrez que hay en la trastienda para tratar el asunto. Con la seguridad que da la oscuridad de la noche, han salido de sus cajas y han ido todas hacia ese tablero para verse. Pero han cometido un pequeño error: no han decidido en *qué casilla* coincidir, así que ahora está cada una en una celda distinta y tienen que decidir (a gritos...) dónde se ven. Obviamente es una reunión clandestina, por lo que quieren elegir la casilla que minimice la suma de las distancias recorridas por todas ellas, teniendo en cuenta que cada torre solo se mueve en horizontal o vertical.



Entrada

La entrada estará compuesta por distintos casos de prueba, cada uno ocupando varias líneas.

La primera línea de cada caso contiene dos números. El primero, N, indica el número de celdas en horizontal y vertical que tiene el tablero en el que se han reunido ($2 \leq N \leq 10.000$). El segundo indica el número T de *torres* que hay sobre él ($2 \leq T \leq min(100.000, N \times N)$).

A continuación aparecerán T líneas con las coordenadas ocupadas por cada una de las torres. El primero indica la fila y el segundo la columna, siendo la coordenada `1 1` la esquina inferior izquierda. Se garantiza que, en el instante inicial, no hay más de una torre en la misma posición.

Salida

Por cada caso de prueba se escribirá la celda en la que se deben reunir para minimizar la suma de la distancia recorrida por todas las torres. Se utilizará el mismo formato que en la entrada: la fila y la columna separadas por un espacio en una única línea.

Si hay varias opciones posibles se dará la posición que esté más a la izquierda, es decir la que tenga un menor valor en la columna. Si sigue habiendo más de una respuesta, se dará la que esté más abajo, es decir la de menor fila. Se admite que la celda de reunión esté ya ocupada por una torre (que no tendrá que moverse).

Entrada de ejemplo

```
8 3
4 3
7 4
5 7
8 4
1 1
8 1
8 8
1 8
```

Salida de ejemplo

```
5 4
1 1
```

G
¡No quiero irme, Sr. Stark!

Una vez conseguidas las *Seis Gemas del Infinito* y colocadas en su *Guantalete*, Thanos decide borrar a la mitad de los seres vivos del universo, para salvarlos de la superpoblación y, naturalmente, para demostrar su amor por la *Señora Muerte*.

Aunque no está claro el modo en el que Thanos elige quiénes sobrevivirán y quiénes serán borrados, hay una teoría que dice que los coloca mentalmente en círculo, cuenta k individuos, que se salvan (al menos momentáneamente) y borra al siguiente. Repite este proceso una y otra vez, dando vueltas en círculo, hasta que ha eliminado a la mitad de la población.

Dependiendo del k y de la colocación de unos y otros en el círculo, algunos se salvarán y otros no. Sabemos que, un instante antes de ser eliminado, *Spiderman* se abraza con miedo a *Iron Man* y le dice "¡No quiero irme, Sr. Stark!"[2]. Pero las cosas podrían haber sido muy diferentes.

Entrada

La entrada comienza con un número que indica cuántos casos deberán ser procesados.

Cada caso de prueba está compuesto por cuatro números, n, s, p y k. El primero, $2 \leq n \leq 1.000$ indica el tamaño de la población que será reducida a la mitad. Los dos siguientes, $1 \leq s, p \leq n$ indican las posiciones en el círculo de *Iron Man* (el Señor Stark) y de *Spiderman* (Peter Parker) respectivamente. Se garantiza que ambos serán diferentes. El último, $0 \leq k \leq 20$ indica el número de personas que Thanos se salta cada vez en su recorrido antes de eliminar al siguiente.

Salida

Por cada caso de prueba el programa escribirá "`No quiero irme, Sr. Stark!`" si, tras eliminar a la mitad de la población, *Spiderman* está entre los asesinados y *Iron Man* no. Por su parte, escribirá "`No quiero irme, Peter!`" si es *Spiderman* el que sobrevive y *Iron Man* el que muere. Finalmente, se escribirá "`No hay abrazo`" si ambos sobreviven o ambos mueren.

Thanos no escatima en muertes. Si el número de personas total es impar, eliminará a una persona más que las que sobrevivan.

Entrada de ejemplo

```
3
2 1 2 1
2 2 1 1
5 1 5 2
```

Salida de ejemplo

```
No quiero irme, Sr. Stark!
No quiero irme, Peter!
No hay abrazo
```

[2] Al menos eso es lo que ha quedado en la memoria colectiva. Como ocurre con *Tócala otra vez, Sam*, de Casablanca, la frase real pronunciada no es exactamente así.

H
Conan Doyle y Sean Connery

El sistema de honores británico permite recompensar a personas individuales que han realizado alguna acción destacada relacionada con el Reino Unido. Uno de los rangos dentro de este sistema es el conocido como *Knight Bachelor* o *Caballero*. El rey o reina pueden premiar a cualquiera, incluso personas que no son del Reino Unido, dotándoles así del título de *Sir*.

De esta forma en el año 1998 Elton John pasó a ser Sir Elton John, o en 2000 Sean Connery se convirtió en Sir Sean Connery. Mucho antes de eso, nada menos que en 1902 el autor de las novelas de Sherlock Holmes pasó a ser conocido como Sir Arthur Conan Doyle.

Aunque tras el nombramiento se puede seguir utilizando el nombre original, lo habitual es que aparezca el *Sir* delante. Si uno quiere ser preciso, no obstante, se debe tener cuidado al hacer la lista de las obras de esos premiados. Por ejemplo, no es correcto decir que en la película Indiana Jones y la última cruzada aparecía Sir Sean Connery pues en aquel año de 1989, todavía no había sido nombrado caballero.

Como editores de una nueva reedición de todos los libros de Sherlock Holmes, queremos respetar la tradición y añadir el *Sir* únicamente en aquellos que fueron escritos cuando ya Doyle tenía el título otorgado.

Entrada

La entrada estará compuesta por varios casos de prueba. Cada caso de prueba comienza con una línea que contiene el año y el nombre completo de una persona a la que se le concedió el título de Caballero y por tanto *desde ese año en adelante* comenzó a llamársele *Sir*. A continuación aparecerá una línea indicando el número de obras publicadas por esa persona (entre 1 y 100), a la que le sigue una línea por cada obra con el año y el título.

Se garantiza que los años serán números positivos no mayores que 10.000, que después de cada año habrá un único espacio, y que la longitud de cada línea no superará nunca los 100 caracteres.

Ningún autor es lo suficientemente prolífico como para tener más de una obra por año.

Salida

Para cada caso de prueba se debe indicar cuál es la primera obra del autor en la que debe aparecer el título de *Sir*. Si no hay ninguna se indicará `NINGUNA` y si todas las obras fueron creadas cuando el autor ya contaba con el título de caballero, se escribirá `TODAS`.

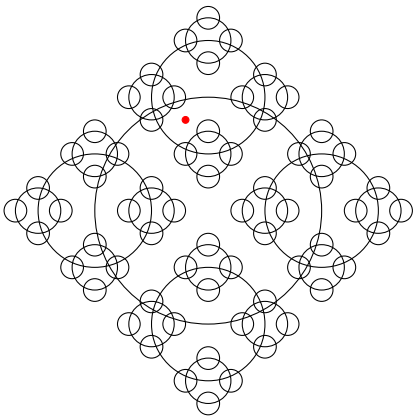
Entrada de ejemplo

```
1902 Connan Doyle
3
1892 Las aventuras de Sherlock Holmes
1905 El retorno de Sherlock Holmes
1902 El sabueso de los Baskerville
2000 Sean Connery
3
1989 Indiana Jones y la ultima cruzada
1963 Desde Rusia con amor
1987 Los intocables de Eliot Ness
```

Salida de ejemplo

```
El sabueso de los Baskerville
NINGUNA
```

Punto dentro de círculos



Un objeto geométrico es fractal cuando, entre otras cosas, es autosimilar, es decir, su forma está construida a base de copias más pequeñas de la misma figura.

En muchos casos esos objetos geométricos se pueden definir mediante algoritmos recursivos, aunque si se hace así hay que poner un "caso base" para que esa recursión termine. Ese caso base suele expresarse en base a la longitud de alguna de las primitivas con las que se construye el objeto.

Un ejemplo de dibujo recursivo comienza pintando un círculo de radio R en el centro del área de dibujado, es decir en la posición $(0, 0)$. Después vuelve a pintarse el mismo dibujo cuatro veces, en las posiciones $(R, 0)$, $(-R, 0)$, $(0, R)$ y $(0, -R)$ pero utilizando círculos de radio $R/2$. Esos cuatro dibujos, a su vez, consistirán en círculos tras los que se pintan cuatro copias de sí mismo con la longitud de sus radios reducidos a la mitad... hasta llegar a ese caso base en el que los círculos son de radio 1.

Para garantizar que siempre terminaremos dibujando círculos de radio 1, eso sí, la división que utilizaremos será *división entera* por lo que si el primer círculo es de radio 5, los cuatro siguientes serán de radio 2 y los siguientes 16 de radio 1. Los puristas dirían que entonces el dibujo no es *autosimilar*, pero el resultado es un dibujo con algunas "imperfecciones" que no quedan mal a la vista.

Con esta definición, un punto dado en el plano puede quedar dentro de varios círculos. ¿De cuántos?

Entrada

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

Cada caso de prueba consiste en tres números. El primero R indica el radio del primer círculo del dibujo recursivo ($1 \leq R \leq 10^8$). Los dos siguientes representan las coordenadas enteras del punto por el que se pregunta ($-2{\times}10^8 \leq X, Y \leq 2{\times}10^8$).

Salida

Para cada caso de prueba se escribirá una línea con un número que indica dentro de cuántos círculos está el punto dado. Si el punto está tocando la circunferencia, también se considera que está dentro del círculo.

Entrada de ejemplo

```
1 1 0
1 1 1
10 -2 8
```

Salida de ejemplo

```
1
0
2
```

J
Unos

Los múltiplos de un número n son todos aquellos números que se pueden conseguir multiplicando a n por cualquier número natural.

Los múltiplos tienen algunas curiosidades. Por ejemplo, el 0 solo se tiene a sí mismo como múltiplo; todos los demás números tienen infinitos múltiplos. También sabemos desde pequeños que los múltiplos de 2 acaban en 0, 2, 4, 6 u 8, y los llamamos *pares*. Y los múltiplos de 5 acaban siempre en 0 o en 5.

Algo mucho menos conocido es que todos los números que *no* son múltiplos de 2 o 5 tienen a un múltiplo que, escrito en decimal, es una secuencia de unos. Por ejemplo, el 3 tiene entre sus múltiplos al 111, y el 7 al 111.111.

Dado un número impar no terminado en 5, ¿cuál es su menor múltiplo que, al ser escrito en base 10, es una serie de unos?

Entrada

El programa deberá procesar, de la entrada estándar, múltiples casos de prueba.

Cada caso de prueba es un número impar mayor que 0 y menor que 1.000.000. Ninguno será múltiplo de 5.

Salida

Por cada caso de prueba n el programa escribirá el número de dígitos que tiene el menor múltiplo de n tal que sea positivo y compuesto únicamente de dígitos a 1.

Se garantiza que la salida siempre es menor que 1.000.000.

Entrada de ejemplo

```
3
7
11
```

Salida de ejemplo

```
3
6
2
```

● K
Próximo tren corto

Aunque el jefe de la estación de Cercanías pone en todos los carteles luminosos y anuncia por megafonía que el próximo tren en llegar a la estación será corto, los pasajeros insisten en distribuirse a lo largo de todo el andén. Cuando llega el tren, son comunes las carreras de aquellos que se han colocado mal y no faltan los tropezones y accidentes. Una vez la maleta de un pasajero voló por los aires ¡y toda su ropa terminó esparcida por las vías!

Salvador D. Pasa Jeros, un conductor de cercanías con buen corazón, quiere parar el tren de modo que se minimice la suma total recorrida por los usuarios mal colocados. Para eso cuenta con las cámaras del andén de la próxima estación, con las que consigue saber cuánta gente hay en cada punto.

Entrada

Cada caso de prueba comienza con dos números, $1 \leq T \leq E \leq 200.000$ indicando, respectivamente, la longitud del tren y del andén.

A continuación aparecen, en otra línea, E números (entre 1 y 1.000) indicando cuántos pasajeros hay en cada posición.

La entrada termina con dos ceros.

Salida

Por cada caso de prueba el programa escribirá la menor longitud total acumulada que tendrán que recorrer los pasajeros en el andén, asumiendo que el tren para en el mejor sitio posible.

Si un pasajero está situado en una zona del andén en la que para el tren, se asume que no deberá desplazarse hasta llegar a una puerta.

Entrada de ejemplo

```
2 4
1 1 1 1
2 5
1 1 1 1 1
3 6
10 5 0 0 10 10
0 0
```

Salida de ejemplo

```
2
4
35
```