Concurso de Programación
AdaByron 2019

http://www.ada-byron.es

Cuadernillo de problemas

Patrocinado por

accenture

25 AÑOS 15

UAM UNIVERSIDAD AUTONOMA DE MADRID

Realizado en la **Escuela Politécnica Superior (UAM)**
8-9 de marzo de 2019

In almost every computation a great variety of arrangements for the succession of the processes is possible, and various considerations must influence the selections amongst them for the purposes of a calculating engine. One essential object is to choose that arrangement which shall tend to reduce to a minimum the time necessary for completing the calculation.
Ada Byron

8-9 de marzo de 2019

Listado de problemas

Autores de los problemas:

- Carlos Aguirre Maeso (Universidad Autónoma de Madrid)
- José Ramón Dorronsoro Ibero (Universidad Autónoma de Madrid)
- Luis Fernando Lago Fernández (Universidad Autónoma de Madrid)
- Gonzalo Martínez Muñoz (Universidad Autónoma de Madrid)

A
La galería de tiro

La federación de tiro con pistola de Ruritania nos ha encargado el diseño de una galería de tiro donde la distribución de los tiradores se ha de hacer de la siguiente manera:

- El primer tirador que llegue ha de situarse en uno de los dos extremos, izquierdo o derecho, de la galería.
- Los siguientes tiradores han de situarse de manera que maximicen el número de puestos de tiro vacíos hasta el tirador más cercano.
- No se admiten más tiradores si no es posible tener al menos un puesto vacío entre dos tiradores.

Por ejemplo, si hay 6 puestos, los dos primeros tiradores se situarán en los extremos mientras que un tercero podrá ponerse en el tercer o en el cuarto puesto. Tras esto no será posible acomodar a más tiradores. Si hay 8 puestos de tiro, se puede acomodar a 4 tiradores: dos en los extremos y, por ejemplo, uno en el cuarto puesto y otro en el sexto. En una galería con 9 puestos podrían disparar hasta 5 tiradores simultáneamente.

La federación no está segura de su presupuesto, por lo que nos pide determinar, para un cierto número N de puestos de tiro, cuál va a ser el número máximo de tiradores a acomodar.

Entrada

La entrada está formada por distintos casos de prueba, uno por línea. Cada línea contiene un único número entero $1 \leq N \leq 1000000$ que representa el número de puestos en la galería.

El final de las entradas se indica con una línea con un 0, que no se debe procesar.

Salida

Para cada caso de prueba se escribirá una línea con el número máximo de tiradores se pueden situar en la galería.

Entrada de ejemplo

```
2
5
10
13
0
```

Salida de ejemplo

```
1
3
5
5
```

B
Mensaje secreto

Para enviarse mensajes secretos, Ana y Tomás (estudiantes de programación) escriben notas en papel en las que cambian cada letra o símbolo por su valor numérico (decimal) de acuerdo al código ascii. Así, la letra `A` la escriben como `65`, la `B` como `66`, y así sucesivamente. Ana ha recibido el siguiente mensaje de Tomás:

```
771013210311711511697115321099711532113117101321081111153211211711011610111411111546
```

Y necesita ayuda para descifrarlo. ¿Puedes echarle una mano?

Entrada

La entrada está formada por un conjunto de mensajes codificados, cada uno en una línea distinta. Los mensajes pueden contener sólo letras mayúsculas y minúsculas del alfabeto inglés, además de los símbolos *espacio*, *coma* y *punto*. La longitud un mensaje, después de ser descifrado, no es nunca mayor de 1000 caracteres.

Salida

Para cada mensaje codificado, se escribirá en una línea el mensaje descifrado.

Entrada de ejemplo

```
651009766121114111110321091111089746
771013210311711511697321121141111031149710997114321011103267 46
```

Salida de ejemplo

```
AdaByron mola.
Me gusta programar en C.
```

C
Snake

Snake es un videojuego clásico que se creó en los años 70 y se popularizó algunos años más tarde. Consiste en controlar a una serpiente evitando que choque contra las paredes y contra su propia cola. La serpiente siempre está en movimiento, pero el jugador puede controlar la dirección en la que su cabeza avanza (izquierda, derecha, arriba o abajo). El cuerpo de la serpiente sigue exactamente el movimiento de la cabeza. Para complicar las cosas, la cola de la serpiente va creciendo y cada vez queda menos espacio disponible. El juego termina cuando la cabeza de la serpiente choca, o bien contra una pared, o bien contra su propio cuerpo.

Suponiendo un panel de juego cuadrado de tamaño 21×21 casillas, nuestro objetivo es determinar en qué instante acaba el juego dada la secuencia de movimientos realizados por el jugador. El juego siempre empieza con una serpiente de longitud 10, alineada en dirección vertical y con la cabeza en el centro del tablero, que se mueve hacia arriba. La serpiente siempre se mueve a razón de una casilla por unidad de tiempo, y la longitud de su cola aumenta en una unidad cada 10 unidades de tiempo.

Entrada

La entrada está formada por distintos casos de prueba, cada uno en una línea distinta. En cada línea aparece en primer lugar un número entero N ($1 \leq N \leq 500$), que indica el número de movimientos que siguen a continuación. Tras esto la línea contiene una lista con la descripción de los N movimientos. Un movimiento viene descrito por la pareja (t_i, l_i), $i = 1, ..., N$, con t_i un número entero y l_i una letra mayúscula. El número, $1 \leq t_i \leq 2000$, indica el instante en el que se realiza el movimiento. La letra, $l_i \in \{U, D, L, R\}$, indica la dirección del mismo de acuerdo al siguiente esquema: U = arriba (up), D = abajo (down), L = izquierda (left), R = derecha (right). Los movimientos aparecen siempre ordenados por el instante en que se realizan, y tienen efecto de manera instantánea.

El final de la entrada se indica con una línea con un único 0 que no se debe procesar.

Salida

Para cada caso de prueba, se escribirá en una línea el instante de tiempo en el que el juego finaliza porque la serpiente se choca. Este instante siempre existirá, pues después del último movimiento, en el supuesto de que se llegue a este punto, la serpiente continúa en línea recta hasta chocar. Por otra parte es posible que en algunos casos el juego termine antes de que se realicen todos los movimientos.

Entrada de ejemplo

```
1 1 R
1 10 L
3 10 R 12 D 14 L
10 10 R 13 D 16 L 20 U 21 R 24 U 25 R 28 D 29 L 35 D
2 1 R 5 L
1 1 D
3 1 R 2 D 6 L
3 6 R 7 D 11 L
0
```

Salida de ejemplo

```
11
20
15
31
5
1
17
11
```

D
Traduciendo los apuntes

El profesor de Programación II tiene mucho trabajo traduciendo los apuntes al inglés, así que ha pedido a sus compañeros que le ayuden con tan apasionante tarea. Se han repartido el trabajo de tal modo que cada uno tiene que traducir una parte del temario. Pero, como son un poco despistados, a algunos se les ha olvidado la parte que les tocaba traducir y al final cada uno ha traducido las páginas que le ha dado la gana. Como resultado tienen un conjunto de intervalos de páginas traducidas que en algunos casos solapan, mientras que otras páginas se han quedado sin traducir. Así las cosas, el profesor se pregunta cuántas páginas están traducidas al inglés.

Entrada

La entrada está formada por varios casos de prueba, cada uno en una línea. Al principio de cada línea aparece un número N ($1 \leq N \leq 10000$), el número de profesores que colaboran en la traducción. A continuación aparecen N parejas de números (x_i, y_i), $i = 1, ..., N$, que indican la primera y la última página traducidas por cada uno de los profesores. Siempre se cumple que $1 \leq x_i \leq y_i \leq 10^6$.

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

Salida

Para cada caso de prueba, se escribirá en una línea el número total de páginas traducidas.

Entrada de ejemplo

```
3 10 20 30 40 15 25
2 12 24 3 12
0
```

Salida de ejemplo

```
27
22
```