Concurso de Programación

AdaByron 2018

`http://www.ada-byron.es`

Cuadernillo de problemas

Patrocinado por

accenture

Realizado en la **Escuela Politécnica Superior (UAM)**
20-21 de abril de 2018

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

20-21 de abril de 2018

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
¡Mío!

Marta y Daniel están aprendiendo las tablas de multiplicar y juegan con su padre al siguiente juego. Cada uno de ellos elige un número (por ejemplo Marta elige el 3 y Daniel el 5) y su padre empieza a contar desde 1 hasta que se cansa. Cada vez que el padre dice un número múltiplo del número elegido por uno de los niños, ese niño debe gritar: *¡Mío!*.

A continuación se muestra un ejemplo del juego:

```
Uno...
Dos...
Tres... Mio (Marta)
Cuatro...
Cinco... Mio (Daniel)
Seis... Mio (Marta)
Siete...
```

A los niños les hace mucha gracia cuando los dos gritan ¡Mío! a la vez (por ejemplo con el 15 en el caso anterior), y se preguntan cuántas veces ocurrirá para valores arbitrarios de M (el número elegido por Marta), D (el número elegido por Daniel) y N (el número hasta el que cuenta el padre).

Entrada

La entrada está formada por distintos casos de prueba, cada uno en una línea diferente. Cada caso de prueba consiste en tres números: M, el número elegido por Marta; D, el número elegido por Daniel; y N, el número hasta el que cuenta el padre (un entero positivo menor que 10^9). Los números M y D son enteros positivos menores o iguales que N. El final de la entrada se indica con una línea con tres ceros que no se debe procesar.

Salida

Para cada caso de prueba, se escribirá una línea con el número de veces que los dos niños gritan a la vez *¡Mío!*

Entrada de ejemplo

```
3 5 100
5 12 100
2 4 16
0 0 0
```

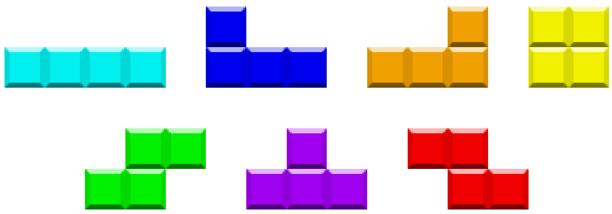
Salida de ejemplo

```
6
1
4
```

B
Tetris

El Tetris es un videojuego que se popularizó en los años 80. Consiste en colocar una serie de piezas con distintas formas que van cayendo en un tablero, de tal modo que queden encajadas de la forma más compacta posible. En este problema vamos a suponer una secuencia de piezas que caen, cada una en una determinada posición y con una determinada orientación que no se pueden cambiar. Las piezas se van amontonando según caen y no se eliminan las filas completas (como ocurre en el juego original). El objetivo es determinar la altura final de cada columna del tablero después de que caigan todas las piezas.

Hay un total de 7 piezas diferentes, mostradas en la figura.



Entrada

La entrada está formada por distintos casos de prueba, y cada caso de prueba ocupa varias líneas. La primera línea contiene dos números, el número de columnas del tablero (C) y el número de piezas que van a caer (N). A continuación hay N líneas, cada una de ellas con la descripción de una pieza. La descripción de una pieza consta de tres números: I, R y P. El primer número, I, es el identificador de la pieza (un número entre 1 y 7, en el mismo orden que en la figura). El segundo número, R, es la rotación de la pieza, que puede tomar los valores 0, 90, 180 o 270. Representa el ángulo de rotación de la pieza en el sentido contrario a las agujas del reloj. El tercer número, P, indica la posición de la pieza. Representa la columna más a la izquierda ocupada por la pieza. La numeración de las columnas empieza en 1.

Los valores mínimo y máximo para C y N son $4 \leq C \leq 100$, $1 \leq N \leq 100000$. A modo de ejemplo, se muestra a continuación la descripción de dos piezas en un tablero de 8 columnas.

```
1 90 3                    2 180 6
  |  X     |                |        |
  |  X     |                |        |
  |  X     |                |     XXX|
  |  X     |                |       X|
  |12345678|                |12345678|
```

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

Salida

Para cada caso de prueba, se escribirá una línea con la altura alcanzada en cada columna del tablero después de que caigan todas las piezas.

Entrada de ejemplo

```
8 4
6 270 4
1 180 5
1 90 6
7 0 4
5 3
1 0 1
4 0 1
5 90 4
0 0
```

Salida de ejemplo

```
0 0 0 9 9 8 3 3
3 3 1 3 2
```

C
Sigue la serie

0, 1, 2, 4, 5, 6, 8, 9, 10, 12,... ¿Qué número sigue en la serie?

Entrada

La entrada está formada por distintos casos de prueba. Cada caso de prueba consta de dos líneas. En la primera línea aparece un único número N ($5 \leq N \leq 200000$), el número de elementos en la serie. En la segunda línea aparecen los N números de la serie, que siempre empieza en 0 y es estrictamente creciente. Los incrementos son siempre entre 0 y 10, y siguen un patrón regular que se repite cíclicamente. Se garantiza que el ciclo completo aparece al menos dos veces en la serie.

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 con el número que sigue en la serie.

Entrada de ejemplo

```
10
0 1 2 4 5 6 8 9 10 12
6
0 2 4 6 8 10
8
0 1 3 6 7 9 12 13
0
```

Salida de ejemplo

```
13
12
15
```

D
Escalando el Everest

Juanito quiere subir al Everest y está planeando la ascensión desde el campo base. Para llegar a la cumbre tendrá que instalar varios campamentos de altura en los que hacer noche. Los campamentos sólo se pueden instalar en determinados puntos de la montaña, y la instalación de cada uno de ellos lleva un coste asociado que depende de su ubicación, el espacio disponible, la exposición al viento, etc. Juanito quiere minimizar los costes de la expedición, pero necesita que no haya más de 650 metros de desnivel entre dos campamentos consecutivos, ni tampoco entre el campo base y el primer campamento ni entre el último campamento y la cumbre de la montaña. ¿Puedes ayudarle a calcular el coste mínimo de la expedición?

Entrada

La entrada está formada por distintos casos de prueba. Cada caso de prueba consiste en 4 líneas. La primera línea contiene un único número entero N ($5 \leq N \leq 1000$), que representa el número de posibles ubicaciones para los campamentos de altura. La segunda línea contiene N números enteros, correspondientes a la altura de cada una de las ubicaciones, medida desde el campo base. Las ubicaciones están ordenadas por altura desde el campo base, y la última ubicación corresponde a la cima de la montaña. La tercera línea contiene N números enteros entre 10 y 100, correspondientes a los costes asociados a la instalación de un campamento en cada una de las ubicaciones. El último de estos números siempre será 0, pues la última ubicación es la cumbre y en la cumbre no se instala ningún campamento. La cuarta línea contiene un único número entero M ($100 \leq M \leq 1000$), que representa el máximo desnivel permitido entre dos campamentos consecutivos (incluyendo el campo base), o entre el último campamento y la cumbre de la montaña. Se garantiza que el problema siempre tiene solución.

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 con el coste mínimo de la expedición.

Entrada de ejemplo

```
5
100 200 300 400 500
10 10 10 10 0
100
11
300 600 1000 1500 1700 2000 2150 2400 2700 2950 3300
10 20 30 40 50 60 70 80 90 100 0
650
0
```

Salida de ejemplo

```
40
250
```

E
La estación de esquí (reto accenture)

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 caso de prueba consiste en 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 $1 \leq N \leq 25$, $0 \leq O \leq 3000$, $0 \leq D \leq 3000$ y $0 \leq M \leq 10000$. Para cada remonte se tiene $0 \leq I < F \leq 3000$ y $500 \leq T \leq 2000$.

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
```

F
La estación central

En las películas de acción todo ocurre siempre en una estación de tren o metro concurrida y llena de gente. Pero las líneas de metro o tren tienen cientos de estaciones. ¿Cuál es la estación más importante de la red?

Un grupo de inteligencia ha decidido estudiar todas las redes de metro y tren del mundo y localizar en cada una de ellas la estación más importante, a la que van a llamar Estación Central.

Para decidir cuál es la estación central, primero deciden cuales son las estaciones más importantes de una red y tras una discusión deciden que las estaciones más importantes van a ser aquellas que pertenecen al menos a dos líneas de metro o tren distintas.

Finalmente deciden que la Estación Central será aquella de las estaciones importantes desde la cual se tarda menos en llegar a todas las demás estaciones importantes de la red, suponiendo siempre que el tiempo que se tarda en llegar de una estación a la siguiente (en cualquier línea) es de 10 minutos. Es decir, la estación central será aquella estación importante para la cual la suma de los tiempos para llegar a las demás estaciones importantes de la red es menor.

Entrada

La entrada consiste en varios casos de prueba, cada uno con varias líneas. Para cada caso de prueba, la primera línea contiene dos números enteros: el primero indica el número de estaciones en la red (N) y el segundo el número de líneas de la red (L). A continuación aparecen L líneas adicionales, cada una con información acerca de una línea de la red, en la cual se indican las estaciones que pertenecen a esa línea separadas por un espacio. Cada estación está representada por un número entero entre 1 y N. Al final de cada línea aparece un 0 que no corresponde a ninguna estación y se debe ignorar.

El número máximo de líneas de la red es de 100 y el número máximo de estaciones por línea es 100.

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

Salida

Para cada red del fichero de entrada se indicará cual es el número de estación correspondiente a la estación central. En caso de haber varias estaciones que cumplan los requisitos, se indicará aquella que tenga un número más bajo. Se garantiza que la red es conexa y que siempre hay al menos una estación central.

Entrada de ejemplo

```
13 3
1 2 3 4 5 6 7 0
8 9 4 10 13 0
11 2 12 9 6 7 0
6 2
2 5 3 6 1 4 0
4 1 6 3 5 2 0
5 2
1 2 3 4 5 0
3 5 1 4 2 0
7 2
3 5 1 2 4 7 6 0
3 6 1 0
0 0
```

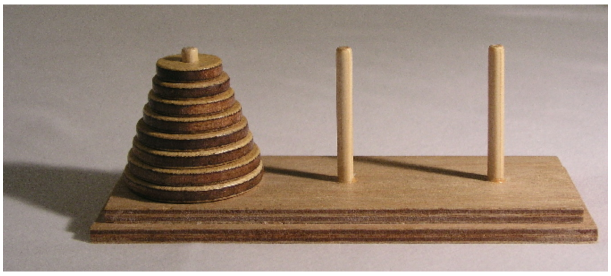
Salida de ejemplo

```
9
3
4
6
```

G
Los monjes de Hanoi y el monasterio Ada-Byron

Todo el mundo (bueno, más o menos) sabe que los monjes de Hanoi deben efectuar $2^{64} - 1$ movimientos para llevar los 64 discos del primero al segundo de los tres postes a su disposición respetando, naturalmente, la regla de no poner un disco encima de otro que tenga un diámetro menor, y que para ello basta mover recursivamente:

- 63 discos del poste 1 al 3 usando el poste 2 como auxiliar,
- 1 disco del poste 1 al 2,
- 63 discos del poste 3 al 2 usando el poste 1 como auxiliar.



Antes de pasar al monasterio de Hanoi, los monjes deben practicar en el monasterio Ada-Byron, donde se trabaja con solo 32 discos. Para facilitar el aprendizaje, los monjes han numerado los discos del 1 al 32, y se han repartido en 32 grupos donde cada grupo se encarga en exclusiva de mover uno de los discos cuando le llega el turno al disco en cuestión. Además, y para estar atentos, los monjes del disco k han elaborado una lista con los movimientos específicos donde tocará mover el disco k-ésimo.

En esa lista hay dos movimientos destacados: cuando el disco k se mueve por k-ésima vez y cuando, tras moverlo, se entra en los últimos k movimientos del disco. Tras ambos movimientos (y, naturalmente, si se dan), los monjes deben tocar una campana.

Para entrenarse a fondo, los monjes del disco k van a practicar el toque de campana trabajando con un número n de discos con $k \leq n \leq 32$. Para uno de estos n, ¿en qué movimientos m_1, m_2 tocarán los monjes del disco k la campana?

Entrada

La entrada está formada por varias líneas con dos enteros k, n. Se garantiza $1 \leq k \leq n \leq 32$. El final de la entrada se indica con una línea con dos ceros que no se debe procesar.

Salida

Para cada caso de prueba k, n debe escribirse una línea con los valores k, n, m_1 y m_2, donde k y n son los datos de entrada, m_1 es un entero con el movimiento tras el que se toca la campana por primera vez (-1 si no es posible) y m_2 es un entero con el movimiento tras el que se toca la campana por segunda vez (-1 si no es posible).

Entrada de ejemplo

```
1 3
2 6
4 8
2 3
7 9
16 21
0 0
```

Salida de ejemplo

```
1 3 1 5
2 6 6 54
4 8 56 184
2 3 6 -1
7 9 -1 -1
16 21 1015808 1015808
```

H
Modificación de tablas

En ocasiones es necesario modificar los valores numéricos de una tabla a través de un fichero que contiene instrucciones de cómo modificar dicha tabla. Por ejemplo, una tarea a realizar de la forma anterior sería la de aumentar o disminuir el valor de un rango consecutivo de casillas que se encuentran dentro de una misma columna de la tabla.

Entrada

La entrada está formada por distintos casos de prueba, y cada caso de prueba ocupa varias líneas. La primera línea contiene tres números, el número de filas de la tabla (F), el número de columnas de la tabla (C) y el número de modificaciones que se van a hacer sobre la tabla (N). A continuación hay N líneas, cada una de ellas con la descripción de una modificación a la tabla. Cada una de estas líneas consta de cuatro números: I, A, B y M. El primer número, I, es el índice de la columna que se va a modificar. Los números A y B son los índices de la primera y la última fila a modificar. El número M es el valor a añadir a las casillas anteriores.

Se garantiza que $F \geq 1$, $C \geq 1$, $F \times C \leq 1000000$ y $1 \leq N \leq 100000$. Los índices de las filas y las columnas empiezan en 0. Los valores mínimo y máximo para M son $-10 \leq M \leq 10$.

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

Salida

Para cada caso de prueba, se escribirá una tabla con F filas y C columnas en la cual se han realizado todas y cada una de las operaciones indicadas. Cada número dentro de una misma fila se separará mediante un espacio y al final de cada fila se insertará un salto de línea. La tabla se supone inicialmente rellena de ceros.

Entrada de ejemplo

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

Salida de ejemplo

```
0 0 1 0 0
-1 0 3 0 0
-1 0 1 0 0
0 0 1 0 0
```

I
Votaciones capicúa

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 quórum Q de senadores necesarios para que una votación sea válida, aunque siempre se ha de cumplir que $Q > 1000$.

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, con $1000 \leq N \leq 9999$ y el quórum es Q, con $1000 \leq Q \leq N$, ¿cuántas votaciones capicúa son posibles?

Entrada

La entrada está formada por varias líneas con dos enteros N, Q. Se garantiza $1000 \leq Q \leq N \leq 9999$. 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 quórum de Q.

Entrada de ejemplo

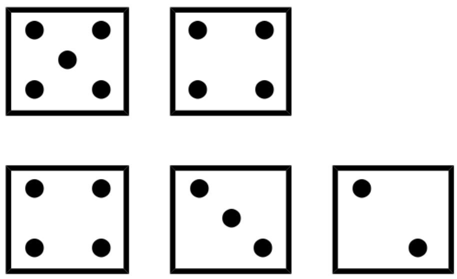
```
1001 1000
2100 2000
5324 4999
0 0
```

Salida de ejemplo

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

J
Pequeña batalla de dados

En el juego del Risk se conquistan territorios con una estrategia basada en dados. Las batallas empiezan cuando el atacante mueve un determinado número de tropas al territorio que desea conquistar. A continuación, la batalla se divide en oleadas. Para cada oleada, el atacante y el defensor lanzan un determinado número de dados. El número de dados que tira cada contrincante no puede ser mayor que el número de tropas disponibles, ni mayor que el número de dados disponibles. Tras tirar los dados cada bando ordena sus dados de mayor a menor y luego se alinean con los dados del enemigo. Finalmente, para cada par de dados defensor-atacante el atacante pierde una tropa si el valor de su dado es igual o menor que el valor del dado del defensor. En caso contrario, es el defensor quien pierde una tropa. Después de cada oleada, y si quedan tropas en ambos bandos, el atacante decide si retirarse o continuar con la batalla. Veamos un ejemplo en el que el defensor tiene 2 tropas y el atacante tiene 3. El defensor tira dos dados y saca 4 y 5. El atacante tira tres dados y obtiene 3, 4 y 2. Tras ordenarlos y emparejarlos, los dados quedan como en la figura.



En este caso, en la primera y segunda pareja de dados el defensor tiene un valor mayor o igual que el atacante, por lo que este último pierde dos tropas. El tercer dado del atacante se descarta ya que no está emparejado con ninguno del defensor. Si el atacante continua para una segunda oleada, entonces el defensor lanza dos dados y el atacante uno. Si, por ejemplo, obtuvieran el defensor 1 y 5 y el atacante 6 entonces, dado que 6 es mayor que 5, el defensor perdería una tropa. En una tercera oleada cada bando lanza un único dado. Si ambos sacaran 5, entonces el defensor ganaría y la batalla terminaría con una tropa sobreviviente en el bando defensor y ninguna en el bando atacante.

Entrada

Cada batalla se describe en dos líneas. La primera línea tiene cinco números enteros con la información siguiente: número de tropas de defensa (td), número de tropas de ataque (ta), máximo número de dados a usar por la defensa (dd), máximo número de dados a usar por el ataque (da) y número de oleadas (no). Los límites para estos valores son: $1 \leq td \leq 100000$, $1 \leq ta \leq 100000$, $1 \leq dd \leq 1000$ y $1 \leq da \leq 1000$. La segunda línea incluye toda la secuencia de tiradas de dados en la batalla como: valores obtenidos por la defensa en la primera oleada, valores obtenidos por el ataque en la primera oleada (siempre se tira el máximo número de dados posible), y a continuación de forma equivalente para las sucesivas oleadas. Las oleadas están siempre completas, pero el atacante puede retirarse antes del final de la batalla sin que haya un ganador. Siempre hay al menos una oleada. El máximo número de lanzamientos de dados en una batalla es 200000.

Salida

Para cada batalla, el programa debe imprimir una línea con el número de tropas que sobreviven para el defensor y el número de tropas que sobreviven para el atacante.

Entrada de ejemplo

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

Salida de ejemplo

```
1 0
5 5
0 3
```