Concurso Regional Programame Zaragoza

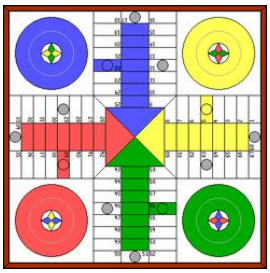
15 de Marzo de 2019

Cuaderno de Problemas

Índice de problemas

Ejercicios realizados por:

Santiago Faci (Centro San Valero)



A. Jugando al parchís

A Diana le encanta el parchís y practica a todas horas. Es por eso que quiere diseñar un algoritmo en el ordenador para poder competir contra él y asi mejorar sus estrategias cuando quede con sus amigos para jugar.
Para comenzar, ha diseñado una primera versión aplicando algunas reglas de comportamiento que se definen a continuación. Asi hará las primeras pruebas de su algorimo pasándole diferentes estados de una partida para ver como responde para el caso de partidas de dos jugadores.
Conviene recordar que todas las casillas del tablero están numeradas y que existen las casillas 'refugio' donde las fichas no pueden comerse. En el tablero de Diana están casillas son las numeradas con números múltiplo de 5.
El ordenador seleccionará la ficha a mover aplicando, revisados en orden, los siguientes criterios:

- Poder comer la ficha de un adversario. Y siempre primero con la ficha que esté más alejada.
- Caer en una casilla 'refugio'
- Hacer avanzar la ficha más retrasada en el tablero

Hay un caso particular y es que, en el caso de que un jugador tenga dos de sus fichas en una casilla 'refugio', y saca un 6, estará obligado a mover una de esas fichas.

Hay que tener en cuenta que en ningún caso pueden coexistir dos fichas de diferente color en una misma casilla y que el jugador que se come la ficha de otro, contará 20 con la misma ficha que comió. Si una casilla está ocupada por dos casillas del mismo color no es posible comerse ninguna de ellas.

Entrada

Se recibe como entrada una primera línea, que se corresponde con el estado del tablero, indicando el número de casilla y color de la ficha que allí se encuentra, teniendo en cuenta que sólo se indican aquellas casillas que contienen alguna ficha. Si hay dos fichas del mismo color se indica dos veces el color de la misma. A continuación se indica el color de la ficha del jugador que lanza el dado y el valor de éste.

Salida

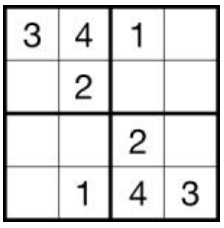
Como salida se debe generar el estado del tablero después de la jugada que se indica como entrada, y aplicando únicamente las reglas que se han explicando en el enunciado para este primer prototipo del algoritmo.

Entrada de ejemplo

```
4v 5a 10v 11a 30aa 34v 35v
v 1
```

Salida de ejemplo

```
4v 5a 30aa 31v 34v 35v
```



B. Sudoku

El sudoku es un juego matemático en el que se tiene que rellenar una cuadrícula con números (generalmente del 1 al 9) sin repetirse ninguno para todas las filas y columnas de dicho cuadro y de los cuadros interiores (que son más pequeños).
En este caso vamos a trabajar con una variante más sencilla que es el sudoku de 4x4, donde se trabaja solamente con los números que van del 1 al 4 y donde las casillas a rellenar son menos.
El sudoku está tan de moda que hay multitud de webs donde puedes pasarte el día jugando. Es por eso que necesitamos que prepares un programa que permita corregir rapidamente algunos sudokus erróneos para notificarlo a los usuarios que juegan todos los días.
Por ahora solamente se corregirán algunos casos concretos, asi que este algoritmo, por ahora solamente tiene que corregir los casos en los que los usuarios repitan algún número en las columnas de los sudokus de 4x4 que realicen. Asi, al algoritmo sólo le llegarán sudokus con estos fallos, aunque también habrá que tener en cuenta que puede que llegue sudokus correctos, que deberán ser devueltos tal cual.

Entrada

Se recibe como entrada un sudoku de 4x4 donde, en algunas de sus columnas puede que se haya repetido un número en vez de poner el que tocaba. En todos los casos el error está en el número que aparece repetido si comenzamos a revisar cada columna desde arriba.

Salida

Se debe generar como salida el sudoku corregido y correcto.

Entrada de ejemplo

```
1 2 3 4
4 2 2 1
1 1 4 3
2 4 4 2
```

Salida de ejemplo

```
1 2 3 4
4 3 2 1
2 1 4 3
3 4 1 2
```

C. Jugando al Risk

En el juego del Risk se llevan a cabo las más feroces batallas de dados de la historia. Dos ejércitos combaten por ganar cada batalla lanzando unos dados esperando conseguir la máxima puntuación para eliminar el mayor número de unidades del ejército contrario.
En programame os proponemos una versión reducida del juego real. En este caso se enfrentan siempre dos equipos que vienen identificados por un color. Se presentan con un número de ejércitos determinado y se enfrentan hasta que uno de los dos equipos pierde todas sus unidades.
Para pelear disponen de dados, pudiendo tirar 3 veces la unidad que ataca y 2 la que defiende. En el caso de atacar, el equipo tendrá que elegir los dos valores más altos para pelear contra su contricante.
Las jugadas se evalúan siempre comparando los valores altos y bajos de los dados entre ambos, y se perderá un ejército por cada unidad en la que una tirada se vea superada por la tirada del contricante, independientemente de si se está atacando o defendiendo.
La partida termina cuando uno de los dos equipos se queda sin ningún ejército.

Entrada

Como entrada se recibe una primera línea indicando los colores y ejércitos de cada equipo. A continuación se indican todos los combates. Cada combate equivale a dos líneas donde se indicará siempre el color del jugador, si ataca (A) o defiende (D) y el valor del lanzamiento de sus dados.
Todas las jugadas terminan siempre con un ganador. Siempre empieza atacando el equipo que aparece en primer lugar en la primera línea de la entrada.

Salida

Como salida se irá obteniendo el resultado de cada batalla en forma de unidades perdidas por cada equipo o 'nadie pierde' cuando empaten con los dados. Cuando la batalla termine se indica mencionando al equipo que gana la batalla.

Entrada de ejemplo

```
rojo 3 azul 5
rojo A 5 4 1
azul D 3 2
azul A 1 1 1
rojo D 1 1
rojo A 3 3 3
azul D 4 2
```

Salida de ejemplo

```
azul pierde 4 unidades
nadie pierde
rojo pierde 1 unidad,azul pierde 1 unidad
rojo gana batalla
```

D. Ayudando a domjudge

En la fase local de un centro de Formación Profesional han tenido un problema con domjudge, la aplicación que hace de juez y 'corrige' los problemas en directo.
El problema está en que no se muestra el marcador final y nadie sabe quién ha ganado. Por ello, los encargados han recopilado la información de los problemas resueltos por cada equipo uno a uno y lo han anotado en una hoja de papel.
Ahora tienen un listado de equipos, problemas resueltos y el tiempo asignado a cada uno, y necesitan hacer las cuentas para saber en qué orden han acabado. Para esa parte han pensado que la mejor forma de hacerlo es escribir un programa de ordenador que lo haga automáticamente, y acuden a vosotros para que les ayudeis. Ellos os pasarán la información que han conseguido anotar y vosotros debéis mostrar el marcador con los nombres de los equipos, el número de problemas resueltos y el tiempo total empleado. Todo ello en orden del primer al último equipo.

Entrada

Se recibe como entrada una primera línea con el número de problemas del concurso y el número de concursantes. A continuación se recibe una línea por participante con el nombre del equipo y el tiempo que les llevó hacer cada problema (en caso de que no lo hayan resuelto aparecerá el caracter guión '-'). El tiempo mostrado incluye ya las penalizaciones por intento fallido.

Salida

Como resultado se debe mostrar la clasificación del concurso, mostrando nombre de equipo, número de problemas resueltos y tiempo total empleado, recordando que el primer criterio a tener en cuenta es el número de problemas resueltos y, en caso de empate, el tiempo total empleado para realizarlos.

Entrada de ejemplo

```
8 4
equipo1 - - - - - - - 20
equipo2 - - - - 10 - - -
equipo3 - - - 30 25 120 80 200
equipo4 20 30 40 60 120 - - -
```

Salida de ejemplo

```
equipo4 5 270
equipo3 5 455
equipo2 1 10
equipo1 1 20
```

E. Haciendo palomitas

A Diana le encanta también ver películas tranquilamente en su casa. Y siempre acompañada de todas las palomitas que pueda comer. En su afán por seguir optimizándolo todo quiere poder calcular cuánto tiempo le llevará cocinar todas las palomitas que quiera comer antes de comenzar, y asi calcular a qué hora estará todo listo para disfrutar de su película favorita.
Tiene calculado que las palomitas tardan en saltar y estar listas 5 minutos pero también se ha dado cuenta que cada vez sólo consigue que se cocinen la mitad de las que ha puesto en la olla. Es por eso que retira las que están listas y vuelve a cocinar las que quedan, y asi repetidamente hasta que todas las palomitas están listas (Tiene calculado que cuando queda un solo grano de maíz en la olla siempre acaba saltando correctamente).
Por eso, necesita idear un algoritmo que permita calcular, para una cantidad de granos de maíz para palomitas, cuanto tiempo necesita pasar en la cocina para tener listas sus palomitas.
Habrá que tener en cuenta que no existe el caso en que un grano se cocine a medias. En cada hornada se tiene que obtener un número de granos completamente cocinados.

Entrada

Se reciben como entradas diferentes cantidades de granos de maíz que se cocinarán.

Salida

Como salida se debe mostrar la cantidad en minutos que costará cocinar cada una de las cantidades que se pasan como entrada, teniendo en cuenta los cálculos de Diana.

Entrada de ejemplo

```
100
50
20
101
```

Salida de ejemplo

```
40 minutos
35 minutos
30 minutos
40 minutos
```

F. Baloncesto

El equipo de baloncesto de Diana quiere planear bien la estrategia y asi no perder ningún partido esta temporada. Quiere idear un algoritmo que le permita conocer la mejor y peor estrategia a seguir para remontar los partidos cuando parezca que está todo perdido.

El algoritmo analiza el estado del marcador y define la forma más rápida de llegar a un resultado ganador (colocarse por delante del adversario en 1 punto) a base de canastas de 2 y 3 puntos. Y lo hace proponiendo la mejor y peor jugada. Asi el entrenador está preparado también para cuando no pueda contar con sus mejores tiradoras de triples.

Por supuesto, todo pasa por evitar que el equipo adversario no meta ni una sola canasta durante la remontada, pero eso ya es cosa del entrenador, que tendrá que asegurarse de practicar en defensa todo lo posible.

Entrada

Como entrada se reciben los diferentes escenarios en forma de marcador local - visitante.

Salida

Como salida se debe generar la peor (más tiros) y mejor (menos tiros) combinación de tiros que deberían acertar para ganar al adversario solamente por la distancia mínima (1 punto).

Entrada de ejemplo

```
20 - 30
10 - 30
0 - 15
```

Salida de ejemplo

```
4 de 2 y 1 de 3 / 3 de 3 y 1 de 2
9 de 2 y 1 de 3 / 7 de 3
8 de 2 / 4 de 3 y 2 de 2
```

G. Rutas de montaña

Un grupo de amigos aficionados al ciclismo se pasan los fines de semana montados en la bici. Quedan todos los sábados y domingos para hacer una ruta. Viven en Benasque, en el corazón de los Pirineos, por lo que están rodeados de montañas.

Tienen una competición interna que consiste en ver quién hace más desnivel positivo (subiendo) en cada ruta. Asi, se dedican a hacer salidas en bici que consisten en encadenar un puerto de montaña tras otro para hacer el mayor desnivel posible. Como la tecnología no es lo suyo se limitan a anotar las cotas de los puertos que ascienden y la cota más baja antes de comenzar la subida del siguiente. Además, saben a que altura viven, y todas las etapas comienzan y terminan en casa.

Cuando llegan a casa hacen los cálculos para saber cuántos metros han ascendido ese día. No estaría mal que les ayudaras resolviendo este problema.

Entrada

Como entrada se recibe una línea que indica la altura a la que vive uno de estos ciclistas. A continuación, se recibe una línea por cada ruta que ha realizado, donde se indican las alturas de los puertos y de la zona más baja que hay entre dos puertos seguidos. Asi, entre dos valores altos siempre aparece un valor bajo que indica la zona más baja hasta la que desciende del primer puerto para comenzar a subir el siguiente, y asi sucesivamente, como si se tratara de una montaña rusa.

Salida

Como salida se tiene que mostrar, para cada etapa, el total de metros ascendidos por el ciclista.

Entrada de ejemplo

```
250
1200 1000 1500 600 700 600 900 800 1900
1100 1000 2000
300 200 400
600 500 740
```

Salida de ejemplo

```
2950
1850
250
590
```

H. Programaphone

Debido a la popularidad del concurso, Programame ha decidido sacar su propia línea de telefonía virtual, llamada Programaphone, con tarifas especiales para sus concursantes.

Y en esta edición del concurso vamos a tratar de diseñar el algoritmo para calcular el precio de la factura de un usuario de esta nueva línea.

La compañia sólo dispone de una tarifa por ahora, la tarifa *globo*, y que tiene las siguientes condiciones:

- El establecimiento son 15 céntimos.
- Los primeros 15 minutos son gratuitos.
- A partir del minuto 15, se tarifica por minutos y se cobra a 3 céntimos. Se cobran minutos completos, aunque no se hayan consumido del todo.

A modo de ejemplo, si se realiza una llamada con una duración de 15 minutos y 25 segundos, el precio seria: 15 cts + 1 min * 3 cts = 0,18 €

Entrada

Se recibe como entrada el detalle de llamadas de un cliente, donde se muestran las horas de inicio y fin de cada una de ellas, indicando hora, minuto y segundo (hh:mm.ss).

Salida

Se debe generar como salida, según las tarifas que se han explicado, el precio total de la factura, en euros, utilizando dos decimales.

Entrada de ejemplo

```
654321987 10:00.00h 10:30.20h
123456789 11:00.15h 11:02.23h
612123443 00:13.59h 01:01.01h
657547644 13:00.10h 13:09.00h
```

Salida de ejemplo

```
2,61 euros
```