Comercio fluvial

Primavera 2024

1. Introducción

Tenemos una cuenca fluvial formada por un río principal y varios afluentes. Hay ciudades en sus fuentes/nacimientos y en los puntos en que dos afluentes se unen, incluida la desembocadura, donde casualmente se unen dos afluentes. No hay ciudades en tramos del río que no sean nacimientos o uniones de afluentes.

De cara a una posible representación gráfica, vamos a suponer que la desembocadura está al norte y las fuentes al sur, es decir, que las aguas van de sur a norte y cuando se va río arriba, se va de norte a sur. Las ciudades a partir de la desembocadura tienen dos ciudades "siguientes" mirando río arriba, una a mano izquierda y otra a mano derecha, y así sucesivamente hasta llegar a las de los nacimientos.

El río y sus afluentes son unas importantes arterias comerciales entre las ciudades situadas en ellos. Las ciudades comercian entre ellas a través del río con unos productos que tienen un identificador numérico que va desde 1 a un cierto valor máximo. Los atributos de un producto son su peso y volumen.

Cada ciudad tiene un identificador `string` que comienza por una letra y solo tiene letras y dígitos. Cada ciudad dispone de un inventario de productos, que no son necesariamente todos los posibles. Cada producto en el inventario de una ciudad tiene como atributos cuantas unidades de dicho producto tiene la ciudad y cuantas necesita. Puede pasar que tenga más unidades de las necesarias, exactamente las que necesita o menos (incluso 0). Todos los productos de un inventario de una ciudad se necesitan, es decir, el número de unidades necesitadas es mayor que 0.

La cuenca dispone de un barquito dedicado a realizar viajes comerciales. En cada viaje, este barco intenta vender una cierta cantidad de un determinado producto e intenta comprar una cantidad (posiblemente diferente) de otro producto. Ambas cantidades serán no negativas y al menos una de ellas será estrictamente positiva. El barco no puede intentar vender y comprar el mismo producto, tienen que ser necesariamente diferentes. Solo se emplea el barco para una de las acciones de comercio entre las ciudades (ver la acción "Viajar" en la descripción posterior).

Se comienza leyendo cuantos productos diferentes hay. Luego, se leen correlativamente sus pesos y volúmenes respectivos. Posteriormente se lee la estructura de la cuenca. Luego se leen los datos del barco, es decir, qué producto quiere comprar y cuantas unidades, y qué producto quiere vender y cuantas unidades. Estos datos iniciales no contienen situaciones erróneas.

A partir de este momento las acciones que se pueden realizar son las siguientes:

- A veces se fundan o se abandonan ciudades por lo que cambia la configuración de las ciudades de la cuenca y se ha de leer de nuevo.
- Se puede leer el inventario de una ciudad concreta.
- Se pueden leer los inventarios de varias ciudades.
- Se pueden modificar los datos del barco.
- Se pueden escribir los datos del barco, junto con los viajes que ha realizado en la cuenca actual.
- Se puede consultar cuantos productos hay.
- Se pueden añadir productos nuevos, dando su peso y volumen.
- Se pueden consultar los datos de un producto.
- Se pueden escribir los datos de una ciudad, es decir, el inventario y el peso y volumen total de los productos que almacena.
- Se puede añadir un producto al inventario de una ciudad diciendo cuantas unidades tiene y cuantas necesita.
- Se puede modificar los datos de un producto en el inventario de una ciudad, es decir, cambiar la cantidad de unidades que tiene o necesita. El número de unidades que se necesitan puede disminuir, pero siempre será mayor que 0.
- Se puede eliminar un producto del inventario de una ciudad. Las unidades que tuviera de dicho producto desaparecen.
- Se pueden consultar los datos de un producto en el inventario de una ciudad.
- Se puede finalizar la actividad en el río.

Además hay otras acciones más complicadas porque involucran dos o más ciudades. Las detallamos a continuación.

1.1. Comerciar

Dos ciudades diferentes pueden comerciar entre ellas. Una ciudad le dará a la otra todos los productos que le sobren hasta alcanzar si es posible los que la otra necesite, y viceversa.

1.2. Redistribuir

A partir de la ciudad de la desembocadura, cada ciudad comerciará con las dos ciudades inmediatamente situadas río arriba (mirando río arriba habrá una ciudad a mano derecha y otra a mano izquierda). Primero comerciará la ciudad de la desembocadura con la ciudad río arriba a mano derecha y luego con la ciudad río arriba a mano izquierda, y así sucesivamente, hasta llegar a las ciudades de los nacimientos.

1.3. Viajar

El barco buscará una ruta partiendo de la desembocadura que maximice el número de productos comprados y vendidos. Una ruta acaba en la ciudad más alejada de la desembocadura en la que haya comprado o vendido algo. Si dos rutas son igual de provechosas, se escoge la más corta en número de ciudades visitadas; si tienen la misma longitud se escoge la que pasa por la ciudad que está a mano derecha mirando río arriba.

Una vez escogida la ruta, se procede a hacer el viaje donde el barco compra y vende en las ciudades del recorrido, modificándose los productos en las ciudades en los que se realicen compras o ventas.

2. Operaciones

Los detalles exactos del formato de la entrada y la salida se han de consultar en el juego de pruebas público del jutge. Puede haber líneas con comentarios, que comienzan por `//` y un espacio y luego el comentario propiamente dicho que ha de acabar en la misma línea. Estas líneas no hay que escribirlas en la salida.

Para cada operación se deben comprobar las posibles situaciones de error en el orden en que aparecen descritos (en algunas de las operaciones pueden producirse varios errores incluso simultáneamente). Si durante la ejecución de una operación se produce alguna de dichas situaciones, la operación deja de ejecutarse sin comprobar los siguientes errores y se imprime un mensaje. Se supone que no habrá más situaciones de error.

1. **leer_rio** o **lr**. Se leerán los identificadores de las ciudades indicando la estructura de la cuenca. No se escribe nada. Los inventarios de las ciudades quedan vacíos. El barco conserva sus atributos de productos pero no ha realizado viajes en la nueva cuenca.

2. **leer_inventario** o **li**. Se leerá el identificador de una ciudad. Si la ciudad no existe se escribirá un mensaje de error. Si la ciudad existe, se leerá un número que indica la cantidad de elementos del inventario y para cada uno de ellos se leerá el identificador del producto, cuantas unidades tiene la ciudad y cuantas necesita. El número de unidades necesitadas siempre ha de ser mayor que 0. Aunque la ciudad no exista, habrá datos de su inventario que igualmente se han de leer aunque no se usen, para pasar a la siguiente operación.

3. **leer_inventarios** o **ls**. Se leerán los inventarios de ciudades del río. Todas las ciudades existirán. Los datos del inventario son como en la funcionalidad anterior. No necesariamente todas las ciudades del río tendrán inventario.

4. **modificar_barco** o **mb**. Se leerá el identificador del producto que se quiere comprar y la cantidad, y el identificador del producto que se quiere vender y la cantidad. Si algún producto no existe, se escribirá un mensaje de error. Si los dos productos son el mismo, se escribirá un mensaje de error. Se garantiza que ambas cantidades serán no negativas y al menos una de ellas será estrictamente positiva.

5. **escribir_barco** o **eb**. Se escriben los cuatro valores mencionados en la anterior operación y los viajes realizados en la cuenca actual, en orden cronológico. Para cada viaje solo se ha de escribir la última ciudad visitada de la ruta escogida.

6. **consultar_num** o **cn**. Escribe cuantos productos diferentes hay.

7. **agregar_productos** o **ap**. Se lee el número de productos nuevos, mayor que 0. Sus identificadores serán correlativos a partir del último producto existente. Se leerán sus pesos y volúmenes respectivos.

8. **escribir_producto** o **ep**. Se lee el identificador de un producto. Si no existe el producto se escribe un mensaje de error. En caso contrario se escribe el peso y volumen del producto.

9. **escribir_ciudad** o **ec**. Se leerá el identificador de una ciudad. Si la ciudad no existe se escribirá un mensaje de error. Si la ciudad existe, se escribirá su inventario, y el peso y el volumen total de los productos almacenados.

10. **poner_prod** o **pp**. Se leerá el identificador de una ciudad, de un producto y las unidades que tiene y que quiere. Si el producto no existe escribe un mensaje de error. Si la ciudad no existe, escribe un mensaje de error. Si el producto ya está en el inventario de la ciudad, escribe un mensaje de error. Si no hay errores, los datos de ese producto se añaden a la ciudad, modificándose el peso y el volumen total si hace falta. Se escribe el peso y el volumen total. El número de unidades necesitadas siempre ha de ser mayor que 0.

11. **modificar_prod** o **mp**. Se leerá el identificador de una ciudad, de un producto y las unidades que tienen y que quiere. Si el producto no existe escribe un mensaje de error. Si la ciudad no existe, escribe un mensaje de error. Si el producto no está en el inventario de la ciudad, escribe un mensaje de error. Si no hay errores, los datos de ese producto sustituyen a los que había en la ciudad, modificándose el peso y el volumen total si hace falta. Se escribe el peso y el volumen total. El número de unidades necesitadas se puede modificar, pero siempre ha de ser mayor que 0.

12. **quitar_prod** o **qp**. Se leerá el identificador de una ciudad y de un producto. Si el producto no existe escribe un mensaje de error. Si la ciudad no existe, escribe un mensaje de error. Si el producto no está en el inventario de la ciudad, escribe un mensaje de error. Si no hay errores, se quitan los datos de este producto en la ciudad, modificándose el peso y el volumen total si hace falta. Se escribe el peso y el volumen total.

13. **consultar_prod** o **cp**. Se leerá el identificador de una ciudad y de un producto. Si el producto no existe escribe un mensaje de error. Si la ciudad no existe, escribe un mensaje de error. Si el producto no está en el inventario de la ciudad, escribe un mensaje de error. Si no hay errores, se escribe cuantas unidades de ese producto tiene y quiere la ciudad.

14. **comerciar** o **co**. Se leerán los identificadores de dos ciudades. Si no existe alguna de las dos (o las dos), se escribe un mensaje de error. Si las ciudades existen y son

la misma, se escribe un mensaje de error. Si las ciudades existen y son diferentes, comercian entre ellas.

15. **redistribuir** o **re**. No se leen datos. La ciudad de la desembocadura comercia con su ciudad río arriba a mano derecha y luego con la ciudad río arriba a mano izquierda, y así sucesivamente.

16. **hacer_viaje** o **hv**. No se leen datos. El barco busca la ruta a partir de la desembocadura que le permita comprar y vender el mayor número posible de productos. En caso que haya más de una ruta que lo cumpla, se queda con la más corta, y en caso de que tengan la misma longitud, se queda con la que viene río arriba a mano derecha. Una vez encontrada la ruta, se hace el viaje y se compran y venden los productos a lo largo de la ruta, modificándose los inventarios de las ciudades. Se escribe el total de unidades de productos compradas y vendidas por el barco.

17. **fin**. Acaba la ejecución.