Ejercicio 3. Gestión de una tienda (4 puntos)

Nos piden implementar un sistema para gestionar la adquisición y venta de productos por parte de una tienda. De vez en cuando la tienda recibe nuevos productos identificados por un código (cadena sin espacios) y señalados con una fecha. Las unidades del producto adquirido se guardan en el almacén, salvo que haya clientes que hubieran intentado comprar ese determinado producto cuando no había existencias y se hubieran colocado en la lista de espera. En ese caso, los clientes son servidos en riguroso orden de llegada. También puede haber venta de productos en existencia. En ese caso, se venden siempre las unidades con una fecha menor. El sistema estará encapsulado en un TAD `Tienda` que proporciona las siguientes operaciones:

- constructora: al comienzo el almacén está vacío y no hay clientes en espera.
- `adquirir(COD, F, CANT)`: gestiona la adquisición de `CANT` unidades (un entero no negativo) del producto `COD` con fecha `F`. Si hubiera clientes esperando la llegada de este producto, devuelve un vector con los identificadores de los clientes que han podido ser servidos, en el orden en que hicieron la petición. El resto de unidades (si las hay) se guardan en el almacén.
- `vender(COD, CLI)`: gestiona la venta de una unidad del producto `COD` al cliente `CLI` (un string sin espacios). Si hay existencias, la operación devuelve `true` y la fecha del producto vendido (la menor entre las disponibles). Si no hay existencias, devuelve `false` y añade al cliente a la lista de espera de este producto (un cliente puede aparecer varias veces en la lista de espera).
- `cuantos(COD)`: devuelve cuántas unidades tiene la tienda del producto `COD` (independientemente de la fecha). Si se trata de un producto que nunca se ha adquirido simplemente se devuelve `0`.
- `hay_esperando(COD)`: indica si hay clientes en la lista de espera del producto COD. Si se trata de un producto que nunca se ha adquirido simplemente se devuelve `false`.

Se pide elegir una representación adecuada e implementar de manera eficiente las operaciones del TAD descrito, utilizando los contenedores vistos de la librería estándar de C++. En la plantilla se proporciona el tipo `Fecha` con operadores de orden y de E/S. Indica y justifica brevemente el coste de cada una de las operaciones. Ninguno de los métodos del TAD debe realizar operaciones de E/S (el manejo de E/S se hace en la función `resuelveCaso`).

Entrada

La entrada consta de una serie de casos de prueba. Cada caso consiste en una serie de líneas donde se muestran las operaciones a realizar, sobre una tienda inicialmente vacía: el nombre de la operación seguido de sus argumentos. Cada caso termina con una línea con la palabra FIN.

Salida

Para cada caso de prueba, se escribirá una línea por operación de la siguiente manera:

- `adquirir`: muestra `PRODUCTO ADQUIRIDO` seguido de los códigos de los clientes que estuvieran en la lista de espera (si los había) y hayan podido llevarse una unidad;
- `vender`: si hay existencias en ese momento se muestra `VENDIDO` y la fecha del producto vendido; si no, se muestra `EN ESPERA`;
- `cuantos`: muestra el número devuelto por la operación;
- `hay_esperando`: si hay clientes esperando que llegue ese producto escribe `SI`, y en caso contrario escribe `NO`.

Observa que ninguna operación debe generar error. Cada caso termina con una línea con tres guiones (—).

Entrada de ejemplo

```
vender lapiz Ana
adquirir lapiz 10/06/19 3
vender lapiz Pedro
adquirir boli 20/06/19 3
adquirir boli 15/06/19 2
vender boli Pedro
vender boli Luis
vender boli Marta
cuantos boli
hay_esperando boli
FIN
vender boli Ana
hay_esperando boli
hay_esperando lapiz
vender boli Pedro
vender boli Luis
adquirir boli 20/06/19 2
cuantos boli
hay_esperando boli
FIN
```

Salida de ejemplo

```
EN ESPERA
PRODUCTO ADQUIRIDO Ana
VENDIDO 10/06/19
PRODUCTO ADQUIRIDO
PRODUCTO ADQUIRIDO
VENDIDO 15/06/19
VENDIDO 15/06/19
VENDIDO 20/06/19
2
NO
---
EN ESPERA
SI
NO
EN ESPERA
EN ESPERA
PRODUCTO ADQUIRIDO Ana Pedro
0
SI
---
```