Smart farming

El sector agrícola está empezando a aplicar técnicas de bigdata para mejorar la productividad. Vamos a diseñar un sistema que nos permita detectar diversas plagas para proceder a fumigar adecuadamente las zonas en las que se producen. La cooperativa agrícola ha dividido sus tierras en sectores, en los que ha situado unos robots capaces de diferenciar distintos tipos de plagas. Cuando detectan una plaga los robots envían al sistema la plaga detectada y el sector en el que está. La cooperativa de vez en cuando alquila una avioneta y fumiga una determinada plaga en todos los sectores que están infectados.

La implementación hará uso de los tipos `sector` y `plaga` que se representan con un `string`.

Las operaciones son las siguientes:

- `alta(id)`: da de alta un nuevo sector `id`. Si el sector ya estaba dado de alta la operación no tiene efecto.
- `datos(id,p)`: el sistema recibe notificación de una plaga `p` desde el sector `id`. Si el `id` del sector no está dado de alta en el sistema se lanzará una excepción con el mensaje `Sector no existente`. Si la plaga no se encuentra registrada se registrará y si la plaga ya ha sido notificada para ese sector y todavía no se ha fumigado se lanzará una excepción con el mensaje `Plaga repetida`.
- `fumigar(p)`: Obtiene un vector con los sectores de los que se recibió noticia de que tienen la plaga `p` en el orden en que se recibieron los avisos. El vector no debe tener sectores repetidos. Los sectores fumigados no necesitan volver a fumigarse hasta que vuelvan a avisar sus robots. Si la plaga no existe o no tiene sectores afectados se lanzará una excepción con el mensaje `Plaga no existente`.
- `plagas(id)`: Obtiene en un vector todas las plagas de un sector que han sido notificadas pero que todavía no se han tratado en orden alfabético. Si el `id` del sector no está dado de alta en el sistema se lanzará una excepción con el mensaje `Sector no existente`.

Requisitos de implementación.

Seleccionar un tipo de datos adecuado para representar la información. En la cabecera de cada función debe indicarse el coste de la misma.

Los métodos del TAD no deben mostrar nada por pantalla. El manejo de la entrada y salida de datos se realizará en funciones externas al TAD.

Debe justificarse el tipo representante elegido y dar el coste de todas las operaciones

Entrada

La entrada consta de una serie de casos de prueba. Cada caso está formado por una serie de líneas, en las que se muestran las operaciones a llevar a cabo, una por cada línea: el nombre de la operación seguido de sus argumentos. La palabra FIN en una línea indica el final de cada caso.

Salida

Para cada caso de prueba se escribirán los datos que se piden. Las operaciones que generan datos de salida son:

- `fumigar`, que debe escribir `Fumigar la plaga` *nombre de la plaga* `en los sectores:` seguido de los sectores que hay que fumigar, separados por un caracter blanco y en el orden en que llegó la notificación de la plaga al sistema.
- `plagas`, que debe escribir `Plagas del sector` *nombre del sector* `:` seguido de las plagas separadas por un caracter blanco y en orden alfabético. Debe dejarse un caracter blanco a ambos lados de los dos puntos.

Cada caso termina con una línea con tres guiones (`---`).

Si alguna operación produce una excepción se mostrará el mensaje de la excepción como resultado de la operación.

Entrada de ejemplo

```
alta sector3
alta sector2
datos sector2 pulgon
datos sector3 cochinilla
datos sector2 cochinilla
fumigar pulgon
datos sector3 cochinilla
fumigar cochinilla
datos sector2 cochinilla
fumigar cochinilla
plagas sector3
datos sector2 pulgon
plagas sector2
FIN
```

Salida de ejemplo

```
Fumigar la plaga pulgon en los sectores : sector2
ERROR: Plaga repetida
Fumigar la plaga cochinilla en los sectores : sector3 sector2
Fumigar la plaga cochinilla en los sectores : sector2
Plagas del sector sector3 :
Plagas del sector sector2 : pulgon
---
```

Autor: Facultad Informática (UCM)