Ejercicio de valoración de conocimientos. Primera Evaluación
2º DAMM. Programación de Servicios y Procesos

Se dispone de un array n caracteres mensaje01. Este array se cargará con un texto en claro. Dicho texto se va a cifrar mediante una sistema de sustitución almacenado en un array alfabetoCode

```
char [] mensaje01  = new char[n]
char [] mensaje02  = new char[n]
char [] alfabetoClaro = {'a', 'b', 'c', 'd', 'e'...};
char [] alfabetoCode  = {'b', 'h', 'l', 'a', 'i'...};
```

Para codificar el mensaje01 en mensaje02 vamos a crear n hilos, de forma que cada hilo codifique un elemento i del array mensaje01 y lo deposite en mensaje02[i]. Para codificar un carácter se buscará en alfabetoClaro y se sustituirá por el equivalente del alfabetoCode. El mensaje01 se introducirá por teclado y deberán imprimirse para verificar el texto a cofificar.

Rado que los arrays alfabetoClaro y alfabetoCode no van a ser modificados no es necesario aplicar ningún mecanismos de concurrencia, sin embargo a la hora de guardar los valores queremos limitar el número de procesos que acceden concurrentemente en escritura al array mensaje02. Para ello se debe utilizar un semáforo permitiendo que un número de procesos puedan acceder concurrentemente en escritura al array. El número de procesos dependerá de una variable max_concurrencia que deberá ser definida previamente.

La creación de los hilos se realizará generando un array de hilos dependiendo de una variable, por lo que modificando dicho valor podemos cambiar el número de hilos a crear.

```
Codifica [] t = new Codifica[maximo_hilos];
```

Al finalizar la ejecución de cada hilo se imprimirá un mensaje indicando que el hilo i_ésimo ha terminado y mostrará las posiciones y valores en claro y cofidicados. Al terminar todos los hilos la aplicación indicar que se ha terminado imprimiendo el texto en claro y codificado. Por ello es necesario estar seguros que todos los hilos han finalizado (isAlive()).

Nuestra aplicación constará al menos de la siguientes clases (no son necesarias más clases):

1. **Test**, que es la aplicación principal que inicia todos los hilos e imprime el resultado.
2. **Encode**, que es la clase que va a sumar los valores.
3. **Resource**, utilizada para implementar la concurrencia.

Criterios de evaluación:

1. Las clases se ha implementado adecuadamente, realizan la función sugerida y los parámetros de las clases son coherentes con su funcionalidad. (2 puntos)
2. La suma de los valores se ha realizado adecuadamente evitando problemas de concurrencia. (2 puntos)
3. La aplicación genera el resultado final del texto codificado en la forma adecuada.(2 puntos)
4. Se ha aplicado correctamente el mecanismo de concurrencia sugerido. (2 puntos)
5. La aplicación presenta una lógica coherente y funciona correctamente. (2 puntos)

Entrega del ejercicio
El ejercicio deberá ser entregado en un archivo comprimido con el nombre y apellidos, y deberá contener únicamente las clases del ejercicio con los nombres solicitados.