Práctica 2: Fechas

1. Objetivo

El objetivo de la práctica es la elaboración de un programa **modular** que permita al usuario realizar algunas operaciones con fechas. Para ello, habrá que poner en práctica cuestiones sobre:

- Estructuras alternativas o condicionales.
- Estructuras repetitivas.
- Programación modular
- Implementación de programas compuestos de varias clases.

2. Introducción.

El programa permitirá al usuario realizar varias tareas entre las que podrá elegir a través de un menú:

1.- Validar una fecha: Se solicitará al usuario una fecha y el programa le dirá si es válida o inválida.

2.- Comparar dos fechas: Se solicitará al usuario dos fechas y el programa le dirá cuál de ellas es menor.

Para ello se implementarán dos clases. En el proyecto que has descargado para la práctica ya están creadas y parcialmente implementadas. Hay que completarlas:

- **La clase utiles.Fechas** (clase Fechas del paquete utiles). Esta es una clase de librería (repasa lo que era una clase de librería en el comienzo del Tema 2). Se trata de una clase que contendrá métodos que pueden resultar de utilidad para trabajar con fechas. Los hacemos en una clase aparte porque podrían resultar útiles para otros programas en el futuro. **Esta clase no debe hacer entrada/salida por teclado/pantalla.**
- La clase practica.ValidarFecha (clase ValidarFecha del paquete practica). Es una clase de programa (repasa lo quer era una clase de perograma en el comienzo del Tema 2). Se trata de la clase que contendrá al método main y a otros métodos en los que el main se apoya, pero que creemos que no pueden resultar útiles para otros programas. Esta clase será la responsable de comunicarse con el usuario (entrada/salida). Para realizar la tarea que se pide esta clase llamará a métodos de la clase Fechas

3. La clase Fechas

Como hemos dicho se trata de una clase de librería con métodos de utilidad relacionados con fechas. De momento vamos a implementar métodos que vamos a necesitar para hacer esta práctica, pero en el futuro podríamos ampliarla y añadir más métodos.

Tienes que implementar los métodos que aparecen en la clase y que están incompletos. Hazlos en el orden en que aparecen en la clase. Antes de cada método hay un comentario explicando qué tiene que hacer el método, el significado de cada parámetro (@param) y qué tiene que devolver (@return)

Estos comentarios siguen una estructura en base a ***anotaciones*** (@param, @return, …) que posibilita crear de la documentación de la clase de manera automática. Una herramienta llamada javadoc es capaz de procesar esos comentarios y construir con ellos una página web con la documentación de la clase, con el mismo estilo que hemos visto en la documentación de las clases de Java.

Implementa todos los métodos de la clase Fechas antes de continuar con la clase ValidarFecha

4. La clase ValidarFecha

El programa mostrará un menú con aspecto similar al siguiente:

1.- Validar fecha.
2.- Comparar fechas.
0.- Salir

Indique la acción a realizar:

- Si el usuario elige una opción correcta (1 o 2), el programa solicitará la información necesaria para llevar a cabo la acción elegida, según se describe en los apartados siguientes.
- Si el usuario elije salir (0), el programa terminará.
- Si el usuario se equivoca e indica una opción incorrecta, el programa le informará del error y se mostrará de nuevo el menú.

5. Validar una fecha (opción 1 del menú).

Cuando el usuario elije la primera opción del menú, se le solicita la introducción de una fecha. El usuario debe introducir una fecha con el aspecto “dd/mm/aaaa”. (dd, mm y aaaa pueden tener longitud variable, por ejemplo 1/1/2009 , 31/12/2009, etc).

El programa indicará al usuario si la fecha introducida es o no correcta. En caso de ser correcta, se mostrará la fecha con el siguiente formato:

dia de ***nombreDelMes*** de *año: Fecha correcta.*

Por ejemplo:

*1 de **enero** de 2009: Fecha correcta*

Es decir, se mostrará el **nombre del mes** en lugar del número del mes, que es lo que el usuario ha introducido.

- Si la fecha es incorrecta, se indicará al usuario. Por ejemplo:

  Día: 30, Mes: 2 Año: 2012. Fecha incorrecta

De forma esquemática, lo que hay que hacer si el usuario elije la opción 1 del menú es lo siguiente:

- Pedir al usuario una fecha y leerla en un String
- Descomponer el String en tres partes: dia, mes, anyo, utilizando para ello los métodos extraerDia, extraerMes y extraerAnyo de la clase Fechas.
- Comprobar si la fecha (dia,mes, anyo) es correcta, utilizando el método esFechaValida de la clase Fechas.
- Mostrar al usuario el correspondiente mensaje, según se ha descrito anteriormente.

6. Comparar fechas (opción 2 del menú).

Cuando el usuario elije la segunda opción del menú, se le solicita la introducción de dos fechas.

Hay que asegurarse de que las fechas introducidas son correctas. Para ello, **cada una de las fechas se leerá en un bucle**, de manera que si la fecha introducida no es válida, se le volverá a pedir al usuario. Para comprobar si la fecha es válida se usará el método esFechaValida de la clase Fechas.

Una vez introducidas las dos fechas, el programa tiene que mostrar un mensaje por pantalla indicando cual de las dos fechas es la menor. Para ello se hará uso del método compararFechas de la clase Fechas.

7. Ampliacion voluntaria (2 puntos): Diferencia entre dos fechas

Añadir al programa una tercera opción de menú “3.- Diferencia entre fechas”.

Se trata de que el usuario introduzca dos fechas y se le indique, en valor absoluto, cuántos días de diferencia hay entre ellas. Como en el apartado 6, habrá que asegurarse de que las dos fechas introducidas son correctas.

8. Proyecto y entrega

Cambia “ApellidoNombre” por tus datos en el proyecto, comprímelo y súbelo al aula virtual