Práctica Evaluable Tema 2.

Estructuras de control

Objetivos.

- Repasar los conceptos estudiados hasta ahora.

Consideraciones iniciales.

- Cada ejercicio tiene que estar en un fichero con el nombre indicado en el enunciado y debe contener una clase con ese mismo nombre, pero sin la extensión ".cs"

Código implementado

Para cada archivo fuente entregado se deberá incluir como comentario en las primeras líneas del archivo el nombre del autor, la indicación de la práctica evaluable y el número del ejercicio.

Además se incluirá un listado de todos los apartados, indicando si han sido implementados totalmente, parcialmente o no ha sido realizado.

Por ejemplo:

```
/*
Perez Gomez, Andres
Practica Evaluable Tema 2
Ejercicio 1    parcialmente

*/
```

```
/*
Perez Gomez, Andres
Practica Evaluable Tema 2
Ejercicio 2    si

*/
```

Entrega.

Se debe entregar un archivo comprimido ZIP con los archivos fuente (extensión .cs) de los ejercicios propuestos.

- Nombre del archivo: **Apellidos_Nombre_PracT2.zip**

  Por ejemplo, si te llamas Andrés Pérez Gómez el archivo debe llamarse *Perez_Gomez_Andres_PracT2.zip*

Desarrollo.

Ejercicio 1.

Nombre del fichero: "PracT2_E1.cs"

Puntuación máxima: 4 puntos

Implementa un programa que solicite al usuario dos números enteros que representen el día y el mes de una fecha y, muestre por pantalla la estación del año correspondiente. El programa pedirá de forma recurrente más parejas de números hasta que se introduzca un “0” como día o mes.

Suponemos que el mes de Febrero siempre tendrá 28 días, y la siguiente distribución de las estaciones del año:

- Primavera : del 21 de Marzo al 20 de Junio
- Verano : del 21 de Junio al 21 de Septiembre
- Otoño: del 22 de Septiembre al 21 de Diciembre
- Invierno: del 22 de Diciembre al 20 de Marzo

En caso de que la entrada introducida no se corresponda con ninguna estación, se debe mostrar el mensaje "Fecha incorrecta" y continuar la ejecución del programa.

Utilizando excepciones (bloques try..catch) se deberá controlar si los datos introducidos no son números, mostrar un mensaje de error y continuar la ejecución del programa.

Ejemplo de ejecución:

```
Introduce el dia: 22
Introduce el mes: 9

La estación del año es OTOÑO

Introduce el dia: 1
Introduce el mes: 1

La estación del año es INVIERNO

Introduce el dia: 31
Introduce el mes: 4

Fecha incorrecta

Introduce el dia: 0
Introduce el mes: 4

FIN
```

Ejercicio 2.

Nombre del fichero: "PracT2_E2.cs"

Puntuación máxima: 3 puntos

Escribe un programa que pida un intervalo [a,b] (validar que el primer número sea menor que el segundo) y muestre todos los números perfectos que se encuentran en él. Un número perfecto es un número que es igual a la suma de todos sus divisores excepto él mismo. Por ejemplo, 6 es perfecto porque 1 + 2 + 3 = 6.

No es necesario utilizar excepciones para comprobar que los valores “a” y “b” son números. Se asume que siempre se introducirán números enteros.

Ejemplos de ejecución:

```
Introduce “a”: 5
Introduce “b”: 50
6 28

------

Introduce “a”: 20
Introduce “b”: 1000
28 496

------

Introduce “a”: 1
Introduce “b”: 1000
6 28 496
```

Ejercicio 3

Nombre del fichero: "PracT2_E3.cs"

Puntuación máxima: 3 puntos

Realiza un programa que le pida al usuario el tamaño del lado de un hexágono regular, y luego lo dibuje en pantalla con asteriscos. Por ejemplo, si el usuario elige un hexágono de lado 3, se dibujará esto:

Ejemplo de ejecución:

```
  ***
 *****
*******
 *****
  ***
```