**TEMA 8 ABSTRACCIÓN Y GENERALIZACIÓN**

1. Defínase una interfaz pública genérica en el paquete “interfaces” de nombre “Interfaz” con los siguientes miembros:
   1. T copiar(int)
   2. T eliminar()
2. Defínase una clase abstracta pública en el paquete “abstractas” de nombre “Abstracta” con los siguientes miembros de objeto:
   1. Un atributo variable privado de tipo int con el nombre de “valor” e inicializado a 1.
   2. Un atributo constante público de tipo int con el nombre de MAXIMO.
   3. Un método público de nombre getValor que devuelve el valor del atributo “valor”.
   4. Un método abstracto protegido de nombre setMaximo sin parámetros que devuelve un int.
   5. Un método público de nombre setValor con un parámetro de tipo int que devuelve un objeto de tipo abstracta. En dicho método se devolverá una excepción de tipo IllegalArgumentException si el valor del parámetro no estuviera entre los valores 1 y MAXIMO, ambos inclusive, con el mensaje de tu nombre y apellidos. De no lanzarse el error, se le asigna como valor del atributo “valor” el del parámetro. Dicho método devuelve el objeto que invoca el método.
   6. Un constructor público sin parámetros, que asignará a la constante MAXIMO el valor devuelto por el método setMaximo.
   7. Un método abstracto público de nombre “mostrar” sin parámetros que devuelve un objeto de tipo abstracta.
3. Defínase una clase pública en el paquete “clases” de nombre “edad” que derive de la clase “Abstracta” e implementa la interfaz “Interfaz<edad>” reemplazando los métodos del siguiente modo:
   1. setMaximo 🡪 Devuelve el valor de 120.
   2. mostrar 🡪 Muestra por consola saltando de línea el texto “La edad actual es ” seguido del valor del atributo “valor”. Dicho método devolverá un objeto de tipo edad.
   3. copiar 🡪 Asigna al atributo “valor” el valor del parámetro de dicho método.
   4. eliminar 🡪 Asigna el valor 1 al atributo “valor”.
   5. Defínase una constante estática publica de tipo Comparator<edad> con el nombre EDAD que compare dos objetos de tipo edad por el valor de su atributo valor (utilícese expresiones lambdas)

Los métodos “mostrar”, “eliminar” y “copiar” devolverán el objeto que invoca el método.

1. Hágase una aplicación de nombre “Aplicacion” que tenga el siguiente código en el método de “inicio”:
   1. Se defina una variable de tipo edad de nombre “edad” que extienda, mediante una clase anónima, la clase edad de forma que reemplace el método de mostrar del siguiente modo:
      * Muestre por consola saltando de línea el texto

“\*\*\*\*\*\*\*\*\*\*\* SALIDA POR CONSOLA \*\*\*\*\*\*\*\*\*\*\*\*\*”

* + - Invoque el método “mostrar” de la clase padre.
    - Muestre por consola saltando de línea el texto “\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*”
  1. Copie en la variable “edad” el valor de 50 invocando el método de “copiar”
  2. Muestre el valor almacenado en la variable “edad” invocando el método de mostrar.

ENTRADA/SALIDA

