Solución al Examen de Fundamentos de Computadores y Lenguajes

Examen Parcial. Junio 2007

Cuestiones (5 cuestiones, 5 puntos en total)

1) Indicar qué texto se mostrará en la pantalla al ejecutar el siguiente programa para cada uno de los siguientes casos:
   a) La segunda vez que se llama a `calculaValor()` se lanza `Apagado`
   b) La segunda vez que se llama a `calculaValor()` se lanza `ValorIncorrecto`
   c) No se lanza ninguna excepción

```
public static void main (String[] args) {
  double suma=0.0;
  int num=0;
  Experimento exp=new Experimento();

  try {
     for (int i=1; i<4; i++) {
        try {
           suma=suma+exp.calculaValor();
           num++;
           System.out.println("Calculado valor "+num);
        } catch (ValorIncorrecto v) {
           System.out.println("Valor incorrecto");
        }
     } // for
     System.out.println("Numero de datos obtenidos: "+num);
  } catch (Apagado a) {
      System.out.println("Experimento apagado");
  }
}
```

---

a) La segunda vez que se llama a `calculaValor()` se lanza `Apagado`

```
Calculado valor 1
Experimento apagado
```

b) La segunda vez que se llama a `calculaValor()` se lanza `ValorIncorrecto`

```
Calculado valor 1
Valor incorrecto
Calculado valor 2
Numero de datos obtenidos: 2
```

c) No se lanza ninguna excepción

```
Calculado valor 1
Calculado valor 2
Calculado valor 3
Numero de datos obtenidos: 3
```

---

2) Implementar en Java el siguiente algoritmo descrito en pseudocódigo:

```
método estático normalizar(array de reales v)
  retorna array de reales
comienzo
    max=0.0
    res = nuevo array de reales del mismo tamaño que v
    // calcular el máximo en valor absoluto
```

```
    para cada i entre 0 y longitud de v -1 lazo
        valorAbs= valor absoluto de v[i]
        si valorAbs>max entonces
            max=valorAbs;
        fin de si
    fin de lazo
    // calcular el resultado normalizando los valores de v
    para cada i entre 0 y longitud de v -1 lazo
        res[i]=v[i]/max
    fin de lazo
    retorna res
fin del método
```

```
public static double[] normalizar(double[] v) {
    double max=0.0;
    double valorAbs;
    double[] res = new double[v.length];
    // calcular el máximo en valor absoluto
    for (int i=1; i<v.length; i++) {
        valorAbs=Math.abs(v[i]);
        if (valorAbs>max) {
            max=valorAbs;
        }
    }
    // calcular el resultado normalizando los valores de v
    for (int i=1; i<v.length; i++) {
        res[i]=v[i]/max;
    }
    return res;
}
```

3) Indicar razonadamente el ritmo de crecimiento del tiempo de ejecución del algoritmo anterior mediante la notación $O(n)$, siendo n el tamaño de `v`.

Análisis de cada parte del método:

a) La 1ª declaración crea e inicializa una variables real simple por lo que es $O(1)$

b) La 2ª declaración crea una variables real simple por lo que es $O(1)$

c) La 3ª declaración crea un array de tamaño n, que Java inicializa a cero de forma automática, por lo que es $O(n)$

d) El primer bucle `for` se hace n veces. Su interior consta de una instrucción simple, una comparación, y una asignación, por lo que es $O(1)$. En total el bucle es $O(n)$

e) El segundo bucle `for` se hace n veces. Su interior consta de una instrucción simple por lo que es $O(1)$. En total el bucle es $O(n)$

f) La instrucción `return` retorna una referencia al array, por lo que es una instrucción que maneja un dato simple (la referencia): $O(1)$

La ejecución secuencial de todas estas partes, aplicando la regla de las sumas, es $O(n)$

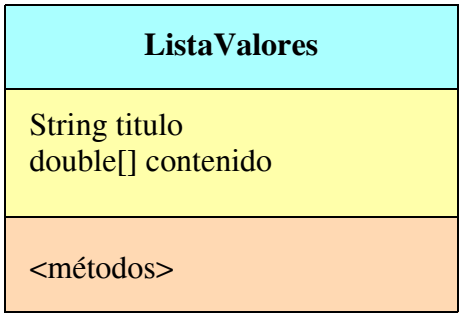
4) Se dispone de la siguiente clase enumerada. Escribir un método al que se le pase como parámetro un objeto de la clase enumerada y que retorne un número real de acuerdo con la tabla de la derecha.

```
public enum Temperatura {
   baja, media, alta, muyAlta;
}
```

0.0	baja
15.0	media
25.0	alta
33.0	muyAlta

```
public static double valor(Temperatura t) {
   switch (t) {
       case baja: return 0.0;
       case media: return 15.0;
       case alta: return 25.0;
       case muyAlta: return 33.0;
       default: return -1.0; // no se puede dar, pero el
                             // compilador lo exige
   }
}
```

5) Se dispone de la clase con los atributos que aparecen en la figura. Escribir un método para esa clase que escriba en un fichero de texto cuyo nombre se le pasa como parámetro, el `titulo` y cada valor del array `contenido`. Cada dato en una línea.




```
public void escribeEnFichero(String nombre) {
  try {
    FileWriter fileout = new FileWriter(nombre);
    PrintWriter fich = new PrintWriter(fileout);

    try {
      fich.println(titulo);
      for (double d:contenido) {
         fich.println(d);
      }
    } finally {
      fich.close();
      fileout.close();
    }
  } catch (IOException e) {
     System.out.println("Error de salida "+e);
  }
}
```

Examen de Fundamentos de Computadores y Lenguajes

Examen Parcial. Junio 2007

Problema (5 puntos)

Se desea finalizar la implementación de la clase que implementa las operaciones necesarias para manipular un sensor de presión, y cuya interfaz se muestra a continuación:

```
/**
 * Clase que representa un sensor de presion
 */
public class SensorPresion
{
    // datos de calibración del sensor
    private double calib[];

    /**
     * Constructor al que se le pasa el nombre del fichero que
     * contiene los datos de calibración
     */
    public SensorPresion(String nombreFichCalib) {...}

    /**
     * Retorna la temperatura actual en grados
     */
    public double temperatura() {...}

    /**
     * Retorna la medida actual del sensor de presion, sin calibrar,
     * en bares
     */
    public double midePresion() throws Desconectado {...}

    /**
     * Retorna la medida actual del sensor de presion, calibrada,
     * en bares
     */
    public double mideCalibrado() throws FueraRango, Desconectado {...}

}
```

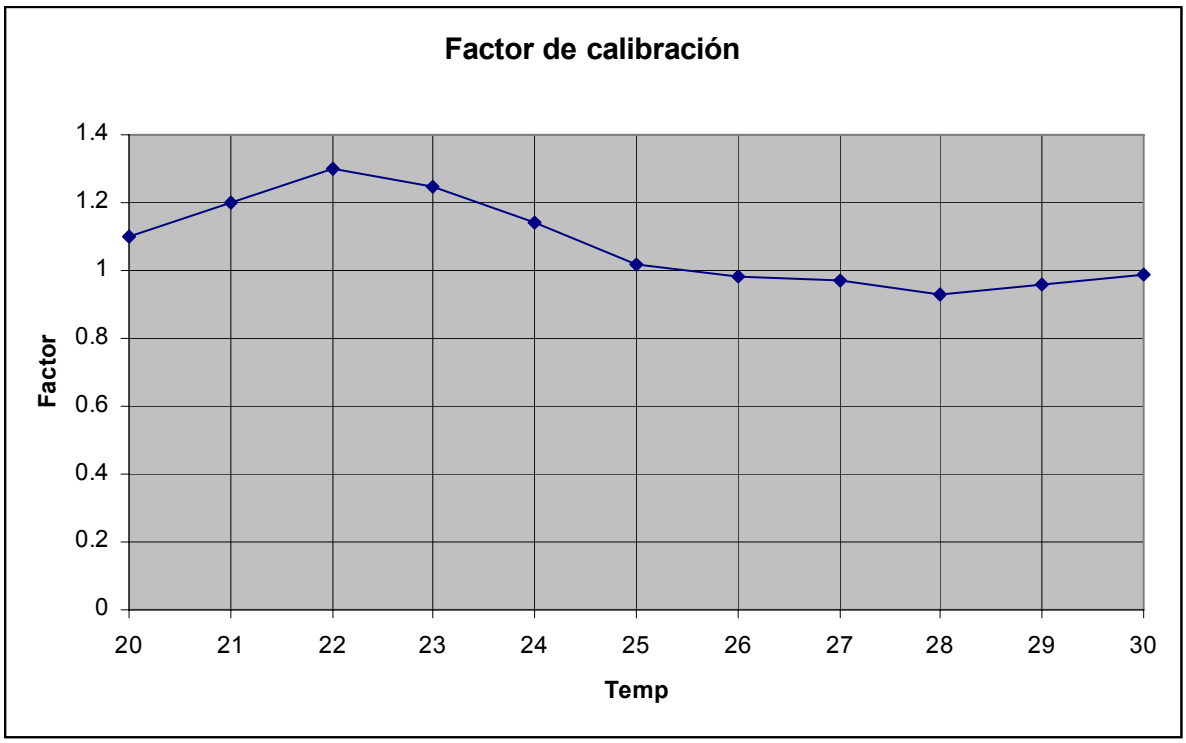
Las excepciones están definidas en clases externas de la forma:

```
public class Desconectado extends Exception {}

public class FueraRango extends Exception {}
```

Los métodos `temperatura()` y `midePresion()` están ya implementados. Lo que se pide es escribir el constructor y el método `mideCalibrado()`.

El sensor de presión obtiene un valor en bares que debe ser calibrado en función de la temperatura. Para calibrar el valor se dispone de una lista de valores de calibración, que son factores por los que hay que multiplicar la presión leída para obtener la presión calibrada. La figura muestra un ejemplo de esta lista de valores, para temperaturas entre 20 y 30 grados centígrados, disponiendo de un valor para cada grado (11 valores en total). Para calibrar valores intermedios de temperatura, se realizará una interpolación lineal entre los valores de calibración de los grados más próximos. Se supone que el sensor de presión funciona siempre entre 20º y 30º.



Constructor (2 puntos). Debe hacer lo siguiente

- Crear el array `calib` con 11 casillas
- Leer del fichero de texto cuyo nombre es `nombreFichCalib` los datos de la calibración y meterlos uno por uno en el array `calib`. El fichero contiene 11 números reales, uno por línea.
- Cerrar el fichero al terminar de leer o si hay algún error
- Si se eleva cualquier excepción al operar con el fichero o sus datos poner el pantalla el mensaje "Fallo al leer fichero de calibración", y poner todos los valores de `calib` a 1.0.

Método `mideCalibrado`() (3 puntos). Debe hacer lo siguiente:

- Leer la temperatura con el método `temperatura`() y si está fuera del rango [20º,30º) lanzar una excepción `FueraRango`
- Leer la presión con `midePresion`(), de modo que si se lanza `Desconectado` reintentar la operación un máximo de 4 veces más. Al quinto intento fallido lanzar la misma excepción `Desconectado`. Si alguna de las lecturas tiene éxito pasar al siguiente punto.
- Si se ha conseguido leer la presión retornar el valor leído multiplicado por el factor de calibración, que se obtiene por interpolación con la fórmula:

$$factor = c_i + (c_{i+1} - c_i) \cdot (t - t_i)$$

donde c_i es el valor de la casilla `i` de `calib`, t es la temperatura actual, t_i es la temperatura actual redondeada por abajo al entero más próximo, e i es el entero igual a t_i - 20º. Para redondear un número real por abajo se puede usar `Math.floor`().

```java
import java.io.*;
public class SensorPresion
{
    // datos de calibración del sensor
    private double calib[];

    /**
     * Constructor al que se le pasa el nombre del fichero que
     * contiene los datos de calibración
     */
    public SensorPresion(String nombreFichCalib) {
       calib=new double[11];
       try {
         FileReader filein = new FileReader(nombreFichCalib);
         BufferedReader in = new BufferedReader(filein);
         try {
           for (int i=0; i<11; i++) {
             calib[i]=Double.parseDouble(in.readLine());
           }
         } finally {
           in.close();
           filein.close();
         }
       } catch (Exception e) {
         System.out.println("Fallo al leer fichero de calibración");
         for (int i=0; i<11; i++) {
           calib[i]=1.0;
         }
       }
    }

    public double mideCalibrado() throws FueraRango, Desconectado {

        // leer la temperatura
        double temp=temperatura();
        if ((temp>=30.0) || (temp<20.0)) {
            throw new FueraRango();
        }

        //leer la presion
        double presion=0.0;
        // máximo 5 intentos
        for (int i=5; i>0; i--) {
           try {
               presion=midePresion();
               break;
           } catch (Desconectado d) {
               if (i==1) {
                    throw d;
               }
           }
        }

        // calibrar la presion
        double ti=Math.floor(temp);
        int i=(int)ti-20;
        double factor=calib[i]+(calib[i+1]-calib[i])*(temp-ti);
        return factor*presion;
    }
    ...// otros métodos
}
```