Examen Final de Fundamentos de Computadores y Lenguajes (Licenciado en Física)

Septiembre 2009

Primera parte (5 cuestiones, 50% nota del examen)

1) Se dispone de la clase `Fecha` con el diagrama de clases que se muestra abajo. Se desea escribir la clase `CalendarioLaboratorio` que permite guardar en un array las fechas en las que está reservado un laboratorio. Su diagrama de clases se muestra abajo

Fecha
public Fecha(int dia, int mes, int año) public int numDias(Fecha f)

CalendarioLaboratorio
private Fecha[] fecha
public CalendarioLaboratorio(int max) public boolean ponFecha(Fecha f, int i)

El constructor de `CalendarioLaboratorio` debe crear el array `fecha` con el tamaño `max`.

El método `ponFecha` debe anotar la fecha `f` que se le pasa como parámetro en la casilla `i` del array `fecha`. Antes debe comprobar que `i` corresponde a una casilla válida y no ocupada. Si es así, anota la fecha y retorna `true`. En caso contrario, no anota la fecha y retorna `false`.

Se considera que la casilla no es válida si su índice cae fuera del array. Se considera que una casilla no está ocupada si vale `null`.

2) Indicar el valor final de `y` en esta instrucción condicional múltiple para cada combinación de valores de `x` e `i` de la tabla:

x	i	y
0.5	1	
6.0	1	
-2.0	1	
30.0	1	
0.5	2	
6.0	2	
-2.0	2	
30.0	2	

```
if (x<0.0 || x>100.0) {
    if (i%2 ==0) {
        y=x*2.0;
    } else {
        y=x;
    }
} else if (x>5.0 && x<10.0) {
    if (i%2 ==0) {
        y=x*6.0;
    } else {
        y=x*3.0;
    }
} else {
    y=x;
}
```

3) Codificar en Java el siguiente algoritmo descrito mediante pseudocódigo, que calcula el desarrollo en serie del arcotangente de x, para |x|<1, según la ecuación

$$\sum_{i=0}^{n} \frac{(-1)^i}{2i+1} x^{2i+1}$$

```
método arcotangente(real x, entero n) retorna real
   si (|x|>=1) entonces
      poner mensaje de error en pantalla
      retorna Double.NaN
   si no
      real atan=0.0
      entero signo=1;
      real numerador=x;
      real denominador=1.0;
      para cada i desde 0 hasta n
         // añade a atan el nuevo término del desarrollo
         atan=atan+signo*numerador/denominador
         // calcula los valores para el siguiente término
         signo=-signo
         numerador=numerador*x*x
         denominador=denominador+2
      fpara
      retorna atan
   fsi
fin del método
```

4) Indicar utilizando la notación O(n) el tiempo de ejecución del algoritmo de la cuestión 3. Razonar la respuesta.

5) Escribir las órdenes que sería necesario dar a un intérprete de órdenes en Linux/Unix para recuperar de una memoria USB montada en el directorio `/media/disk1` los siguientes ficheros de su directorio `8-sep-2009`, metiéndolos en el directorio `/home/pedro`

- los acabados en `.txt`
- la carpeta `fuentes` completa, con todos sus contenidos

Suponer que el directorio de trabajo es `/home/juan`

Examen de Fundamentos de Computadores y Lenguajes
(Licenciado en Física)

Septiembre 2009

Segunda parte (5 puntos, 50% nota del examen)

Se desea implementar una clase que almacena los resultados de un experimento en el que se miden magnitudes eléctricas de tensión e intensidad. Las medidas se obtienen a intervalos de 1 ms y se guardan en dos arrays "paralelos", llamados `tension` e `intensidad`, usando como unidades voltios y amperios. La medida número `i` por tanto corresponde al milisegundo `i` del experimento; su tensión se guarda en `tension[i]` y su intensidad en `intensidad[i]`.

Los atributos y la interfaz de la clase se muestran a continuación:

```
public class MedidasElectricas
{
    private double tension[]; //voltios
    private double intensidad[]; // amperios

    /**
     * Constructor al que se le pasa el tiempo del experimento
     * en milisegundos
     */
    public MedidasElectricas (int milisegundos) {...}

    /**
     * Leer los datos del experimento de un fichero de texto
     */
    public void leeDeFichero (String nombreFichero)
       throws NoExiste, ErrorDeLectura   {...}

    /**
     * Calcular la potencia media en vatios
     */
    public double potenciaMedia() throws NoHayDatos{...}

    /**
     * Calcular la energía eléctrica empleada en el experimento,
     * en julios
     */
    public double energia() {...}

    /**
     * Verificar que las tensiones están dentro de sus límites
     * inferior y superior, en voltios
     */
    public boolean tensionesCorrectas
       (double limInferior, double limSuperior)  {...}
}
```

Las excepciones son clases ya hechas con un constructor sin parámetros.

Se pide implementar los métodos de la clase, de acuerdo a lo indicado en la interfaz y a la siguiente descripción detallada:

1) constructor (0.5 puntos): crea los dos arrays `tension` e `intensidad` de tamaño `milisegundos+1`. El motivo es que si, por ejemplo el experimento dura 100 ms, se guarden 101 datos, desde el milisegundo 0 al 100, ambos incluidos.

2) `leeDeFichero` (2 puntos): lee de un fichero de texto cuyo nombre se pasa como parámetro los datos del experimento, y los guarda en los arrays `tension` e `intensidad`. Cada línea del fichero contiene dos números reales separados por espacios en blanco: el primero es la tensión en voltios, y el segundo la intensidad en amperios. La primera línea tiene los datos del milisegundo 0, la segunda línea los del milisegundo 1, y así sucesivamente,

   Si el fichero no existe debe ponerse un mensaje de error en pantalla y luego lanzar la excepción `NoExiste`.

   Si al leer el fichero se detecta un error debe lanzarse la excepción `ErrorDeLectura` sin poner ningún mensaje de error. Los errores a tener en cuenta serán los de formato del número (`NumberFormatException`), la inexistencia de espacios en blanco para separar los dos números, y el hecho de que el fichero se termine antes de haber rellenado por completo los arrays.

3) `potenciaMedia` (1 punto): La potencia eléctrica en el instante `i` es `tensión[i]*intensidad[i]`. La potencia media se obtiene sumando todas las potencias instantáneas y dividiendo por el número de medidas. Si la duración del experimento es menor o igual que 2 ms se considera que hay pocos datos y hay que lanzar la excepción `NoHayDatos`.

4) `energia`: (0.5 puntos): La energía eléctrica se calcula como el producto de la potencia media por el tiempo del experimento. La potencia media se calcula con el método anterior. Si se lanza `NoHayDatos`, retornar cero.

5) `tensionesCorrectas` (1 punto): este método debe buscar en el array `tension` un valor de tensión incorrecto, es decir que sea superior a `limSuperior` o inferior a `limInferior`. Si encuentra un valor de tensión incorrecto, retorna `false`. Si no encuentra ninguno, retorna `true`.