Examen de Programación
(Grados en Física y Matemáticas)

Septiembre 2017

Primera parte (1.25 puntos por cuestión, 50% nota del examen)

1) Escribir el código Java de un método que retorna el número de la semana de una fecha, dados el día (de 1 a 31), mes (de 1 a 12) y año (de 2000 a 2099) que se pasan como parámetros. Se supone que el 1 de enero da comienzo a la semana 1. Para calcular el número de la semana de forma simplificada (el cálculo completo es más complejo) se puede usar esta expresión:

$$semana = \left\lceil \frac{diasInicioMes(mes) + dia}{7} \right\rceil$$

donde la función techo $\lceil x \rceil$ es el redondeo por arriba y `diasInicioMes` es una tabla que contiene el número de días transcurridos en el año hasta el día anterior al primer día de cada mes. Sus valores se obtienen de esta forma:

Año	Ene	Feb	Mar	Abr	May	Jun	Jul	Ago	Sep	Oct	Nov	Dic
bisiesto	0	31	60	91	121	152	182	213	244	274	305	335
normal	0	31	59	90	120	151	181	212	243	273	304	334

Para determinar si el año es bisiesto o no en el rango de interés que es de 2000 a 2099, basta comprobar si es múltiplo de 4. La cabecera del método será:

```
public static int semana (int dia, int mes, int año)
```

2) La tabla siguiente muestra la tarifa adicional de una línea aérea para la elección de asiento. Escribir un método Java que dado el medio por el que se hace la reserva (web, teléfono o aeropuerto) y la elección del usuario nos retorna la tarifa aplicable (en euros). Se valorará la eficiencia. Se dispone de las constantes estáticas definidas en la figura. La cabecera del método será:

```
public static int tarifaAsiento(int medio, int eleccion)
```

```
// Medios de reserva
public static int WEB=0
public static int TELEFONO=1
public static int AEROPUERTO=2

// Elección del usuario
public static int ALEATORIO=10
public static int TRASERO=11
public static int DELANTERO=12
public static int XL=13
```

Tarifa adicional en euros:

Elección	Web	Teléfono	Aeropuerto
Aleatorio	0	0	0
Trasero	2	4	9
Delantero	3	5	10
XL	6	9	14

3) Hacer un método Java para leer de un fichero de texto cuyo nombre se pasa como parámetro los contenidos de dos `ArrayList` de números reales que se pasan como parámetros. Inicialmente se deben vaciar los dos `ArrayList` usando su método sin

parámetros clear(). El primer array contendrá frecuencias (Hz) y el segundo voltajes de una señal eléctrica (V). En el fichero hay en primer lugar una línea de encabezamiento que explica los datos que vienen a continuación, y que deberá ignorarse. A continuación aparecen los contenidos de los arrays en dos columnas del mismo tamaño: la columna izquierda contiene los datos del primer array y la columna derecha los del segundo. La cabecera del método será:

```
public static void leeFichero (String nombreFich, ArrayList<Double> frec,
                               ArrayList<Double> volt)
```

Ejemplo de fichero

```
Frec(Hz)   Volt(V)
120000     3.4
125000     3.8
200000     4.5
250000     5.1
```

Contenidos finales de los ArrayList para este ejemplo

frec	volt
120000	3.4
125000	3.8
200000	4.5
250000	5.1

4) En un computador con sistema operativo Linux se desea escribir un *script* para copiar determinados archivos desde el disco duro a un *pen drive* situado en la carpeta /mnt/disk1. Los archivos a copiar se encuentran en tres carpetas dentro de la carpeta del usuario:
   - scripts: contiene scripts en ficheros cuyos nombres no tienen ninguna extensión
   - estadistica: contiene datos estadísticos en ficheros con extensión .dat
   - graficas: contiene gráficas en ficheros con extensión .graph

   En primer lugar hacer un esquema de la distribución inicial de carpetas y ficheros.

   En segundo lugar escribir el *script*. Utilizar para los ficheros del disco duro rutas relativas desde la carpeta del usuario, y para los ficheros del *pen drive* rutas absolutas. El *script* debe hacer los siguientes pasos:
   - cambiar el directorio de trabajo poniéndolo en el del usuario
   - crear en el *pen drive* tres carpetas llamadas scripts2, estadistica2 y graficas2.
   - copiar en la nueva carpeta scripts2 todos los *scripts* de la carpeta scripts que comiencen por estad
   - copiar en la nueva carpeta estadistica2 todos los ficheros de la carpeta estadistica que contengan la secuencia 2017 y acaben en .dat
   - mover a la nueva carpeta graficas2 todos los ficheros de la carpeta graficas que contengan la secuencia 2017 y acaben en .graph
   - borrar de las carpetas estadistica y graficas todos los ficheros que contengan la secuencia 2016

   En tercer lugar hacer un esquema de la distribución final de carpetas y ficheros.

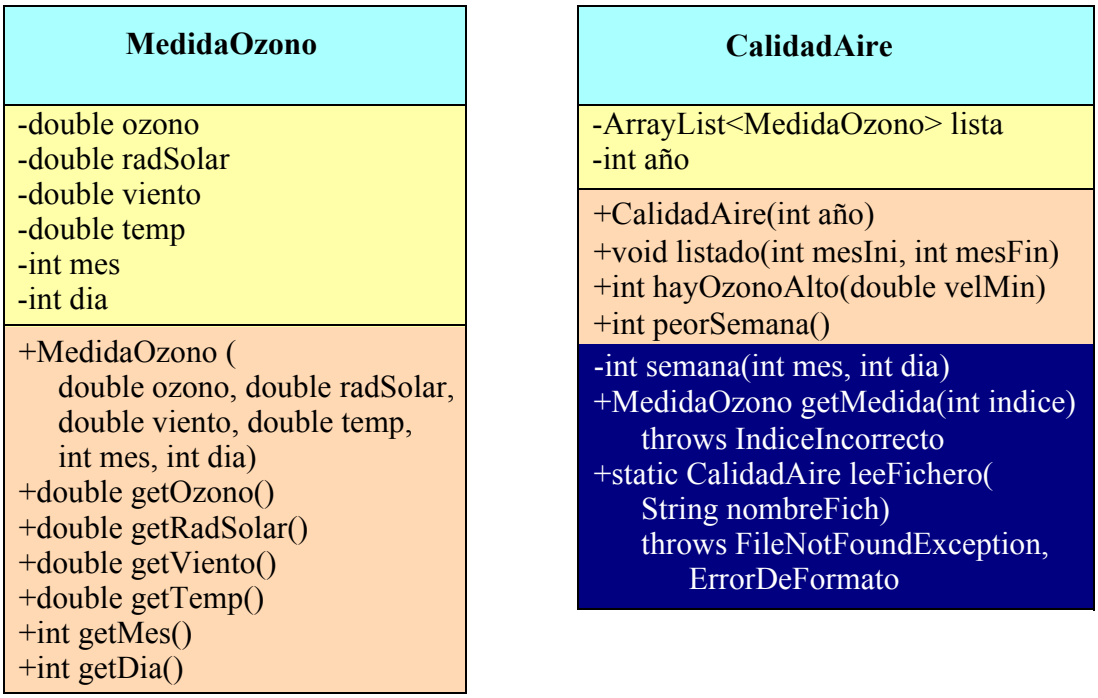
Examen de Programación
(Grados en Física y Matemáticas)

Septiembre 2017

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

Se desea hacer parte del software de análisis de la calidad del aire en una ciudad. Se dispone para ello de los datos de la calidad del aire almacenados en un fichero con formato "csv".

Se dispone de la clase MedidaOzono que ya está hecha y almacena los datos de la medida de ozono superficial para un día concreto. Se puede ver su diagrama de clase en la figura. La clase tiene un constructor al que se le pasan los valores iniciales de los atributos: concentración de ozono en $\mu g/m^3$, radiación solar en W/m^2, velocidad del viento en m/s, temperatura en ºC y el mes (entre 1 y 12) y día (entre 1 y 31) de la medida. Dispone también de un observador para cada atributo.



La clase CalidadAire recoge en el ArrayList lista los datos de varias medidas de ozono almacenadas en objetos de la clase MedidaOzono, ordenadas por fecha de más antigua a más moderna. Su diagrama de clases se muestra en la figura. Los métodos getMedida() y leeFichero() están ya implementados. Se pide escribir el encabezamiento de la clase, los atributos, el constructor y los métodos listado(), hayOzonoAlto(), y peorSemana(). La descripción de los métodos es:

- *constructor*: Se le pasa el año al que corresponden las medidas y lo guarda en el atributo año. También crea la lista vacía.
- listado(): Escribe en pantalla un listado de los datos de aquellas medidas del ozono cuyo mes esté comprendido entre mesIni y mesFin, ambos incluidos. Para ello, en primer lugar pone una línea de título que muestra el texto "Medidas del Ozono en" seguido del año de las medidas. En la siguiente línea pone un encabezamiento explicativo de las columnas que aparecerán a continuación. Finalmente pone en columnas los datos de las medidas del

ozono, una medida por línea, con todos sus datos. Utilizar para los números reales una anchura de 14 caracteres y dos decimales.

- hayOzonoAlto(): Retorna el índice de la primera casilla de la lista cuya velocidad del viento es superior a velMin y que contiene una concentración de ozono superficial superior al máximo recomendado por la OMS de 100 µg/m^3, o -1 si no hay ninguna casilla que cumpla estas condiciones.
- peorSemana(): Retorna el número de la semana (dentro del año) para la que la media de valores de concentración de ozono es mayor, siempre que para esa semana tengamos al menos tres datos. Si no hay suficientes datos retorna -1. Para este método utilizar este pseudocódigo que incluye un método privado auxiliar que comprueba si la media de concentracion de ozono de una semana es máxima:

```
método privado esMaximoOzono(entero numDatos, real sumaOzono,
  real ozonoMaximo) retorna booleano
    // solo hay que calcular la media si hay tres datos al menos
    si numDatos>=3 entonces
       // comparamos el ozono medio con el máximo
       retorna sumaOzono/numDatos>ozonoMaximo
    fin si
    // si no hay tres o mas datos no consideramos el máximo
    retorna falso
fin método

método peorSemana() retorna entero
   entero semanaAnterior=-1
   entero semanaActual=-1
   real sumaOzono=0
   entero numDatos=0
   real ozonoMaximo= menor valor real
   entero peorSemana=-1
   // Recorrer todas las medidas de la lista
   para cada MedidaOzono med en lista
      semanaActual=semana(mes y día de la medida med)
      si semanaActual=semanaAnterior entonces
         // no hemos cambiado de semana y simplemente sumamos los datos
         sumaOzono= sumaOzono+ozono de la medida med
         numDatos++
      si no
         // hemos cambiado de semana
         si semanaAnterior>0 entonces
            // Si la media es máxima actualizamos el máximo y la peor semana
            si esMaximoOzono(numDatos,sumaOzono,ozonoMaximo) entonces
               ozonoMaximo=sumaOzono/numDatos
               peorSemana=semanaAnterior
            fin si
         fin si
         // inicializar datos para empezar a calcular la media de la nueva semana
         sumaOzono=ozono de la medida med
         numDatos=1
      fin si
      semanaAnterior=semanaActual
   fin para
   // Procesar la media de la ultima semana, si es máxima
   si esMaximoOzono(numDatos,sumaOzono,ozonoMaximo) entonces
      peorSemana=semanaActual
```

```
    fin si
    retorna peorSemana
fin método
```

Los métodos que **no** se piden son:

- semana(): Retorna el número de la semana dentro del año, a partir del día y mes. Se supone que el 1 de enero da comienzo a la semana 1.
- getMedida(): Retorna la medida de ozono cuyo índice se indica. Si el índice no está comprendido entre 0 y el número de elementos de la lista menos uno se lanza IndiceIncorrecto.
- leeFichero(): Lee del fichero en formato "csv" cuyo nombre se indica los datos de la calidad del aire y los retorna en un objeto de la clase CalidadAire. Lanza FileNotFoundException si el fichero no existe o ErrorDeFormato si el formato del fichero csv es incorrecto.

Las excepciones IndiceIncorrecto y ErrorDeFormato están ya definidas en el proyecto en clases aparte.

Finalmente, se pide hacer un programa principal en una clase aparte que haga lo siguiente:

a. Lee el fichero CSV llamado madrid_2017.csv obteniendo un objeto de la clase CalidadAire que se usará en los siguientes pasos.

b. Hace un listado de los datos entre enero y junio (ambos incluidos)

c. Busca si existe alguna casilla con concentración de ozono mayor que la permitida para días con viento entre flojo y fuerte (>3m/s) y muestra en pantalla el resultado. Si la casilla existe muestra en pantalla sus datos.

d. Lo mismo que en el paso anterior pero para días con viento fuerte (>10m/s)

e. Muestra en pantalla el número de la peor semana

Tratamiento de errores:

- Si en el paso a) se lanzase FileNotFoundException o ErrorDeFormato se abandonarán las instrucciones restantes del main para, a continuación, poner un mensaje en pantalla indicando cuál es el error que ha ocurrido.
- Si en los pasos c) o d) se lanza IndiceIncorrecto se abandonan esos dos pasos con un mensaje de error pero luego se continúa con el paso e)

Valoración:

- encabezamiento de la clase, atributos y constructor: 0.5 puntos
- métodos listado(), hayOzonoAlto(), y peorSemana(): 1 punto cada uno
- programa principal: 1.5 puntos