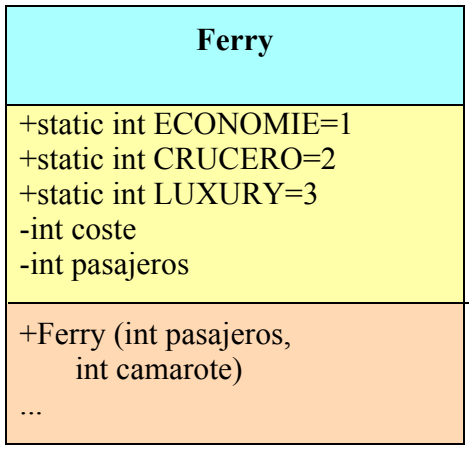
Examen de Introducción al Software
(Ingeniería Informática)

Septiembre 2015

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

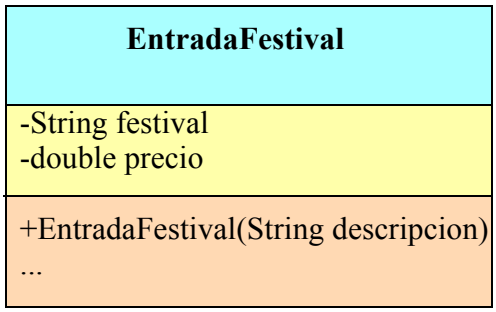
1) Se desea escribir el constructor de la clase Ferry cuyo diagrama se muestra y que contiene datos sobre una reserva de viaje en un Ferry. Este constructor copia en el atributo pasajeros el parámetro del mismo nombre. Por otro lado calcula y guarda en el atributo coste el coste total del billete en función del número de pasajeros (dado por el parámetro pasajeros) y del tipo de camarote (dado por el parámetro camarote), según el precio en euros de la siguiente tabla:



Tipo de camarote	Primer pasajero	Segundo pasajero	Tercer y sucesivos pasajeros
ECONOMIE	186.00	173.00	162.00
CRUCERO	223.00	212.00	198.00
LUXURY	330.00	310.00	291.00

Se valorará la eficiencia de la implementación.

2) Se desea escribir el constructor de la clase EntradaFestival cuyo diagrama se muestra. Este constructor recibe como parámetro un texto llamado descripcion que tiene el nombre del festival y el precio de la entrada, con el formato de siguiente ejemplo:




```
Pirineos SUR Precio=22.00
```

El constructor debe depositar en el atributo festival los caracteres de la descripción anteriores a la palabra "Precio=", eliminando espacios en blanco iniciales y finales (para esto último se cuenta con el método trim() de la clase String). Asimismo debe depositar en el atributo precio los caracteres posteriores a "Precio=" convertidos a número (para lo que se cuenta con el método estático Double.parseDouble()).

3) Escribir un método Java que reciba como parámetro un array de números reales conteniendo ángulos en grados, y que retorne otro array del mismo tamaño con todos los ángulos del original convertidos a radianes.

4) Escribir el *pseudocódigo* de un método iterativo que comprueba si un string que se le pasa como parámetro y que contiene una expresión matemática con paréntesis tal como:

((a + b) × ((c + d) - e))

tiene los paréntesis equilibrados. Para ello se deben recorrer los caracteres del string de izquierda a derecha y llevar en una variable la cuenta del número de paréntesis abiertos. Cada vez que encontramos un paréntesis izquierdo esta cuenta se aumenta en uno y cada vez que encontremos uno derecho se decrementa en uno. Para que los paréntesis estén equilibrados se tienen que dar estas dos propiedades:

- la cuenta final de paréntesis abiertos tiene que ser cero
- en ningún momento la cuenta puede ser negativa

Lógicamente, la cuenta de paréntesis abiertos debe comenzar en cero.

5) Contestar *razonadamente* a las siguientes preguntas. Utilizar un *máximo* de *5 líneas* para cada respuesta:

a. Convertir la siguiente fórmula de una hoja de cálculo a una fórmula equivalente con referencias absolutas. ¿Qué diferencia práctica hay entre una fórmula y otra?

```
=SI((C10-C9)>100;3;C18)
```

b. Indicar los pasos a dar para conseguir obtener en una hoja de cálculo una casilla que indique cuántos valores menores a 10^{-8} hay entre los valores de la columna E en las filas 1 a 120.

c. Se parte de una hoja de cálculo con diversas columnas de datos ya metidos. Describe brevemente los pasos a realizar para añadir una columna nueva que contenga en cada casilla la suma de los valores de cada fila.

d. Indica al menos tres tipos de datos reales que están disponibles en una base de datos de tipo `base`. ¿Por qué hay distintos tipos y no solo uno?

e. Explica brevemente qué mecanismo se usa en una base de datos para establecer relaciones entre registros de diferentes tablas de datos.

Nota: en esta cuestión las respuestas correctas suman 0.2 puntos, las incompletas o las que pasen de 5 líneas ni suman ni restan y las erróneas restan 0.1 puntos. Se valora la *precisión* de la respuesta.

Examen de Introducción al Software
(Ingeniería Informática)

Septiembre 2015

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

Desde el 1 de Julio de 2015 se ha puesto en marcha en España un sistema de facturación por horas del consumo eléctrico de los hogares. Con el nuevo sistema, las compañías facturarán los consumos por horas, teniendo en cuenta que cada hora tiene un precio distinto. Se desea hacer parte de un programa que permita al cliente adaptar los consumos de sus electrodomésticos y así reducir el importe de su recibo.

Cada electrodoméstico se almacena en el programa mediante un objeto de la clase `Electrodomestico` cuyo diagrama se muestra abajo y que ya está implementada. Sus atributos son:

- `nombre`: es un texto que describe el electrodoméstico
- `horaInicio`: hora de encendido (de 0 a 23); se enciende al principio de esta hora
- `horaFinal`: hora de apagado (de 0 a 23); se apaga al final de esta hora
- `consumo`: consumo promedio en kW

Si por ejemplo un electrodoméstico funciona solo durante una hora, pongamos las 20h, entonces `horaInicio=20` y `horaFinal=20`.

Como puede observarse, se dispone de un constructor al que se le pasan los datos del electrodoméstico. También hay métodos observadores, uno para cada atributo:

Electrodomestico
-String nombre -int horaInicio -int horaFinal -double consumo
+Electrodomestico (String nombre, int horaInicio, int horaFinal, double consumo) +String nombre() +int horaInicio() +int horaFinal() +double consumo()

Hogar
-ArrayList<Electrodomestico> lista -double[] precioLuz -boolean precioRellenado
+Hogar(String[] electrodomésticos) +void obtenerPrecioLuz() +void listadoElectrodomesticos() +double coste(String nombre) +double[] costeHorario()

Lo que se pide es implementar en Java la clase `Hogar` cuyo diagrama de clases se muestra arriba. La clase dispone de los siguientes atributos:

- `lista`: guarda una lista de objetos de la clase `Electrodomestico`
- `precioLuz`: es un array de 24 casillas conteniendo los precios de la luz en cada hora del día, en euros/kWh

- precioRellenado: booleano que indica si el array precioLuz tiene sus casillas rellenadas con datos válidos o no.

Los métodos de la clase hacen lo siguiente:

- *constructor*: recibe como parámetro un array de strings que contiene los datos de todos los electrodomésticos de la casa, crea un objeto de la clase Electrodoméstico con cada uno y añade estos objetos al atributo lista. El parámetro electrodomésticos contiene un electrodoméstico en cada casilla, con el formato que se muestra en este ejemplo, donde el orden de los datos es: Nombre, horaInicio, horaFinal, consumo(kW). Ejemplo:

  ```
  "Lavaplatos     20          21          1.3"
  ```

  Además, el constructor da valor al atributo precioLuz creando para él un array con 24 casillas, pero *no* rellena sus valores. Por último pone precioRellenado a False.

  Se puede suponer que el array de strings es correcto y que horaFinal>=horaInicio para cada caso.

- obtenerPrecioLuz(): este método se supone ya hecho. Se conecta por Internet a un servidor en el que se recogen los precios de la luz en cada hora y con ellos se rellena el array precioLuz. Además pone precioRellenado a true.

- listadoElectrodomésticos(): Muestra en pantalla un listado del precio total que supone encender cada electrodoméstico ese día. En cada línea se pondrá el nombre del electrodoméstico y el coste en euros. El coste diario de un electrodoméstico se calcula como

$$coste = \sum_{h = horaInicio}^{horaFinal} consumo \cdot precioLuz[h]$$

  En este método si precioRellenado es False se pone en pantalla un mensaje de error.

- coste(): Busca el electrodoméstico cuyo nombre coincide con el parámetro nombre. Si lo encuentra retorna el coste de ese electrodoméstico calculado con la fórmula de arriba. Si no lo encuentra o si precioRellenado es False retorna Double.NaN.

- costeHorario(): Retorna un array de 24 números reales en los que se recoge el gasto total de todos los electrodomésticos del hogar en cada hora del día. Para ello se seguirá el siguiente pseudocódigo:

```
método costeHorario() retorna real[0..23]
   real[] total = nuevo array de 24 números reales
   // recorrer cada electrodoméstico de la lista
   para cada Electrodomestico ele en lista hacer
      // recorrer todas las horas a las que está encendido ele
      para h desde horaInicio de ele hasta horaFinal de ele hacer
         // añadir al coste calculado hasta el momento el coste de ele en la hora h
         total[h]= total[h] +(consumo de ele)*precioLuz[h]
      fin para
   fin para
   retorna total
fin método
```

  Sin embargo, si precioRellenado es False se retornará null. Este detalle no figura en el pseudocódigo.

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

a. Crea un objeto de la clase `Hogar` a partir del siguiente array:

```
{"Lavaplatos     20          21          1.3",
 "Lavadora       21          23          1.8",
 "Aspiradora     10          11          2.3",
 "Cafetera        8           9          0.5"}
```

b. Invoca el método `obtenerPrecioLuz()` para obtener los precios del día publicados en Internet.

c. Muestra en pantalla las 24 casillas del array obtenido con `costeHorario()`. Si se obtiene el valor `null` se muestra un mensaje de error.

d. Muestra en pantalla el listado de todos los electrodomésticos.

e. Muestra en pantalla el coste del electrodoméstico `"Lavadora"`.

f. Para probar si `coste` retorna correctamente el valor `null` se intenta mostrar en pantalla el coste del electrodoméstico inexistente `"xxxx"`. Se comprueba que el resultado es `null` y se pone un mensaje de confirmación en pantalla o en caso contrario un mensaje de error.

Valoración:

- atributos y constructor: 1 punto
- `listadoElectrodomesticos`: 1 punto
- `coste`: 1 punto
- `costeHorario`: 1 punto
- programa principal: 1 punto