Examen de Introducción al Software
(Ingeniería Informática)

Septiembre 2017

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

1) Escribir un método Java que calcule y retorne el coste de la entrada a un parque natural dada la tarifa que se muestra. El método tiene como entradas un booleano que indica si es temporada alta o baja, un `String` que indica el tipo de entrada (`"Adulto"`, `"Infantil"` o `"Tarifa escolar"`) y un booleano que indica si la entrada es de tarde o de día completo. Si el `String` del tipo de entrada es incorrecto se retornará `-1`. Se valorará la eficiencia.

Temporada alta, del 01/04 al 30/09

	TARDE	DÍA COMPLETO
Adulto	20€	30€
Infantil	11€	17€
Tarifa escolar	12€	13€

Temporada baja, del 01/10 al 31/03

	TARDE	DÍA COMPLETO
Adulto	16€	23€
Infantil	9€	14€
Tarifa escolar	7€	8€

2) Escribir un método Java que reciba como parámetros los siguientes valores reales relativos a la rotación de una onda electromagnética polarizada linealmente al atravesar un plasma. El método recibe como parámetros:
   - e: carga eléctrica de las partículas del plasma, en C
   - m: masa de las partículas, en Kg
   - N_t: densidad superficial de partículas, en m^{-2}
   - B: densidad de flujo magnético, en Wb m^{-2}
   - λ: longitud de onda, en m
   - ϕ: ángulo entre la dirección de propagación y las líneas del campo magnético, en grados

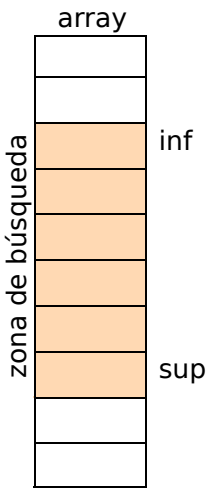
   El método calcula y retorna la rotación, θ, en grados, obtenida mediante la siguiente expresión, siendo $c = 3 \cdot 10^8$ m/s la velocidad de la luz en el vacío y $\varepsilon_0 = 8.85 \cdot 10^{-12}$ farad/m la permitividad del vacío. La expresión calcula radianes, pero el método debe retornar grados:

$$\theta = \frac{e^3 B \lambda^2 \cos\phi}{8\pi^2 c^3 \varepsilon_0 m^2} N_t$$

3) Escribir el *pseudocódigo* de un método recursivo llamado `busquedaBinaria` que busca el entero de valor `x` dentro de una parte de un array de enteros cuyos valores están ordenados de menor a mayor. La zona de búsqueda está delimitada por los índices `inf` y `sup`, ambos incluidos. Si se encuentra el valor en la zona de búsqueda, se devuelve el índice de la casilla en que se encuentra. Si no se encuentra, se devuelve `-1`. El método tiene por tanto como parámetros al array, el valor buscado `x`, y los límites de la zona de búsqueda: `inf` y `sup`.

En el algoritmo de búsqueda binaria se divide la zona de búsqueda en dos mitades. En la versión recursiva, el caso directo se da cuando `inf` y `sup` son iguales, y en ese caso se retorna `inf` si esa casilla del array vale `x`, y `-1` en caso contrario. En el caso recursivo se calcula el punto medio `med=(inf+sup)/2` y si `x` es mayor que la casilla `med` del array se retorna el valor retornado por la búsqueda binaria para el mismo array y valor `x` pero entre los límites `med+1` y `sup`, y en caso contrario se retorna el valor retornado por la búsqueda binaria para el mismo array y valor `x` pero entre los límites `inf` y `med`.

Se supone que los índices `inf` y `sup` son correctos, `inf<=sup`, y el array es correcto estando siempre ordenado de menor a mayor.



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

   a. En una hoja de cálculo tenemos esta fórmula escrita en la celda `C2`: `=$B3*C1`. ¿Qué fórmula aparecerá en la celda `D2` si copiamos allí esta fórmula? ¿Y si la copiamos en `F2`?

   b. Disponemos de una hoja de cálculo con valores entre las filas 1 a 50 en las columnas `A` a `G`. Indicar los pasos que darías para que en la columna `H` aparezca el promedio de las casillas no vacías de la fila correspondiente.

   c. Se parte de una hoja de cálculo con las columnas `A`, `B`, `C` y `D` con datos ya metidos. Describe brevemente los pasos a realizar para que en la columna `E` cada elemento contenga la suma de las casillas anteriores de la fila solo si esta resulta positiva o cero. Si la suma es negativa esa casilla se debe quedar en blanco.

   d. Explica con brevedad los pasos para crear una consulta que incluya un criterio de selección de información.

   e. En una base de datos relacional se puede obtener información tanto de una tabla o de una consulta. Explica las principales diferencias entre hacerlo con una o con otra.

   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 2017

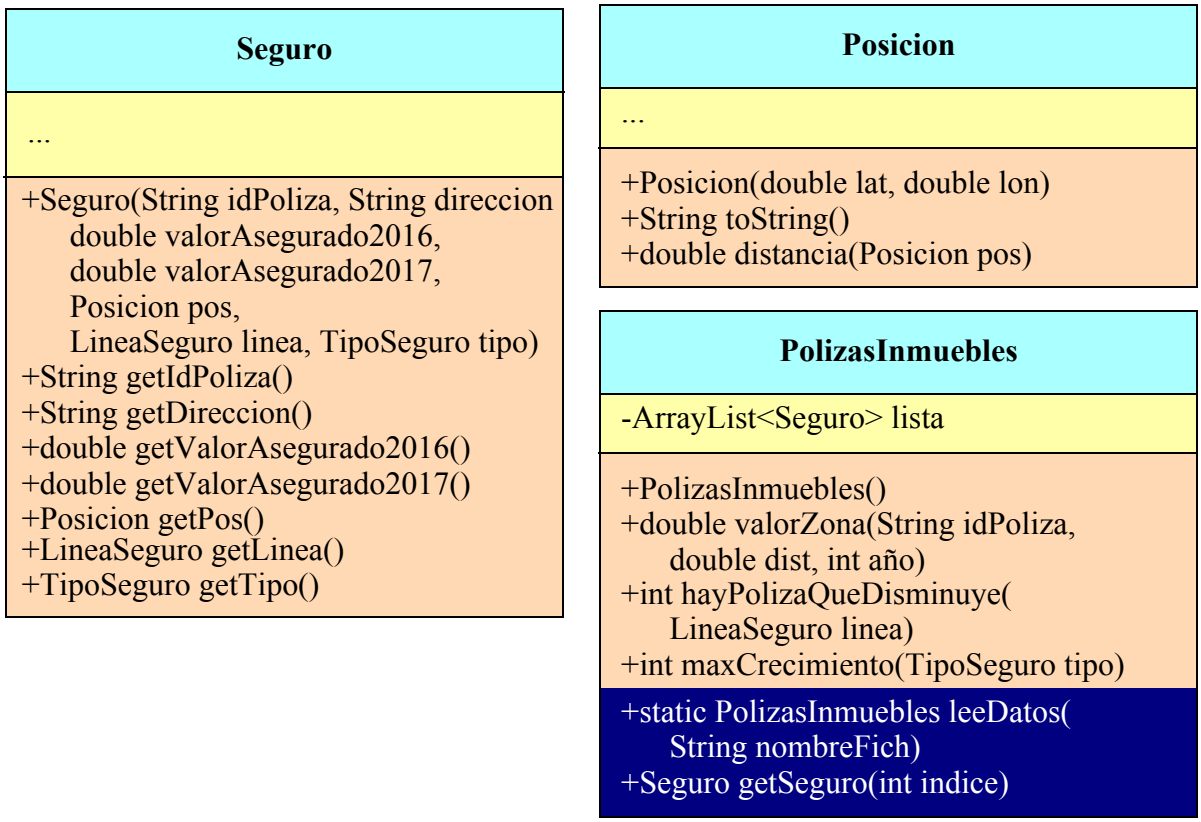
Segunda parte (5 puntos, 50% nota del examen)

Se desea realizar una parte del software perteneciente a una compañía de seguros de inmuebles. Para ello se dispone de la clase `Seguro` ya realizada, cuyos objetos almacenan datos relativos a una póliza de seguros. Dispone de métodos para construir el seguro y observadores de los atributos. Las cantidades aseguradas están en euros. La línea de seguro y el tipo de seguro se expresan como datos de las siguientes clases enumeradas que están definidas aparte:

```
public enum LineaSeguro {RESIDENCIAL, COMERCIAL}

public enum TipoSeguro {INCENDIO, INCENDIO_E_INUNDACION, TODO_RIESGO}
```

También se dispone de la clase `Posicion`, que almacena las coordenadas del inmueble: latitud (`lat`) y longitud (`lon`) en grados. Dispone de un método que retorna la distancia entre la posición del objeto actual y la del objeto `pos`, en kilómetros, y otro método que retorna la posición convertida a `String`. Los diagramas de estas clases se muestran aquí:



Se pide escribir en Java parte de una clase llamada `PolizasInmuebles` que dispondrá de un atributo que es un `ArrayList` de objetos de la clase `Seguro` y que contiene la lista de las pólizas de seguros de los inmuebles asegurados por la compañía. Sus métodos deberán hacer lo siguiente:

- *Constructor*: Crea la `lista` vacía.

- valorZona(): Retorna el sumatorio de los valores asegurados para el año indicado, de aquellos inmuebles situados a una distancia menor o igual a dist respecto a la posición de la póliza cuyo identificador se indica. Si esa póliza no existe o si el año no es 2016 o 2017 se retorna Double.NaN para indicar el error.
- hayPolizaQueDisminuye(): Busca una póliza de la línea de seguros indicada cuyo valor asegurado sea menor en 2017 al que tuvo en 2016. Si la encuentra retorna su índice dentro de la lista. Si no la encuentra retorna -1.
- maxCrecimiento(): Calcula la póliza del tipo de seguro indicado cuya diferencia entre el valor asegurado en 2017 y el de 2016 es máxima. Retorna su índice dentro de la lista o -1 si la lista está vacía.

Por otro lado hay dos métodos de la clase PolizasInmuebles que ya están hechos:

- leeDatos(): Lee del fichero cuyo nombre se indica los datos de las pólizas y los retorna en un objeto de la clase PolizasInmuebles. Si hay algún error durante la lectura se retorna null.
- getSeguro(): Retorna el seguro que ocupa en la lista el índice indicado en el parámetro, o null si ese parámetro es menor que 0 o mayor o igual al tamaño de la lista.

Escribir además en una clase aparte un programa de prueba con un main que haga:

- Lee el fichero "inmuebles.csv" obteniendo un objeto de la clase PolizasInmuebles. Si el objeto obtenido es null el programa finalizará con un mensaje de error. En otro caso el objeto se usará en los siguientes pasos.
- Muestra en pantalla el valor total asegurado en un radio de 20 km de la póliza de identificador "788543" para el año 2016.
- Muestra en pantalla si hay alguna póliza cuyo valor asegurado disminuya de 2016 a 2017, de entre las de línea RESIDENCIAL. Si la hay, muestra en pantalla sus identificador, dirección, y tipo de seguro
- Muestra en pantalla los datos de la póliza TODO_RIESGO que experimentó el mayor crecimiento de 2016 a 2017. Los datos a mostrar son: identificador, dirección, y tipo de seguro.

Valoración (sobre 5):

1) Encabezamiento de la clase, atributo y constructor: 0.5 puntos

2) valorZona(), hayPolizaQueDisminuye(), maxCrecimiento(): 1 punto cada uno

3) main: 1.5 puntos