Programación, Curso 2017-2018

Problema 6: ArrayList y Excepciones

Datos personales	
Apellidos:	
Nombre:	

1 Pasar pseudocódigo a código usando ArrayList

Objetivo

Pasar un algoritmo que usa ArrayList de pseudocódigo a código y practicar con dos tipos de recorridos.

Descripción

Implementar en Java el siguiente algoritmo descrito en pseudocódigo, que normaliza los valores de un ArrayList que se pasa como parámetro. Los valores se normalizan dividiéndolos entre el máximo de los valores absolutos de los datos del ArrayList. Los valores normalizados se retornan en otro ArrayList de números reales:

```
método estático normalizar(ArrayList<Double> v) retorna ArrayList<Double>
        real max = 0.0
        ArrayList<Double> resultado = nuevo ArrayList<Double>
        // encontrar el valor absoluto máximo guardado en v
        para i desde 0 hasta longitud de v -1
                real valorAbs = valor absoluto de v[i]
                si valorAbs>max
                        max=valorAbs;
                fin si
        fin para
        // normalizar cada valor del array v y guardarlo en res
        para cada x en v hacer
                añade a resultado el valor x/max
        fin para
        retorna resultado
fin del método
```

Respuesta que se pide

El código Java del método

2 Búsqueda de múltiples elementos

Descripción

Escribir un método al que se le pasa como parámetro un array de Strings y que retorne un ArrayList de Strings, conteniendo aquellos Strings del array original que comiencen por una letra mayúscula.

Observar que la búsqueda de múltiples elementos no tiene que ver con el algoritmo de búsqueda en tablas visto en clase.

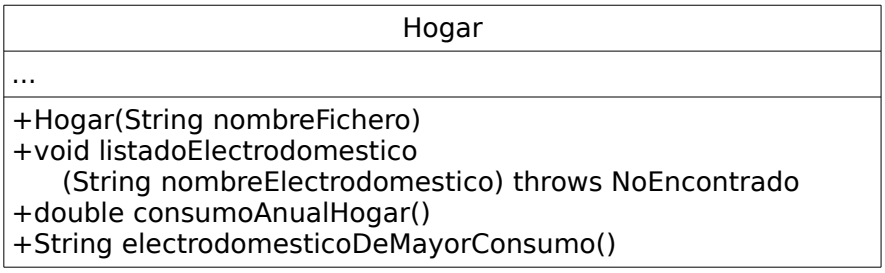
Nota: puede utilizarse el método estático isUpperCase() de la clase Character para determinar si un carácter es una letra mayúscula.

Respuesta que se pide

<Poner aquí el código Java del método>

3 Excepción leve

Se dispone de una clase que permite conocer los consumos eléctricos de los electrodomésticos de un hogar a lo largo del año. Obedece al diagrama de clases que se muestra



Puede observarse que el método ListadoElectrodomestico() tiene una cláusula throws que anuncia que el método puede lanzar la excepción NoEncontrado, definida en una clase aparte. Esta cláusula nos obliga a tratar esta excepción si invocamos al método.

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 fichero de nombre "electrodomesticos.txt"

b) Invoca a listadoElectrodoméstico() con el nombre "Lavadora"

c) Invoca a listadoElectrodoméstico() con el nombre "Luces"

Si en alguno de los pasos b) o c) se lanza NoEncontrado se muestra un mensaje de error y luego se sigue por el paso e), es decir, no se hace el d)

d) Muestra en pantalla el consumo anual del hogar obtenido mediante consumoAnualHogar()

e) Muestra en pantalla el nombre del electrodoméstico de mayor consumo, obtenido mediante electrodomesticoDeMayorConsumo().

Respuesta que se pide

<Poner aquí el código Java de la clase principal>

4 Excepciones leve y grave en el mismo método

Descripción

Se dispone de un método declarado según la cabecera de abajo, perteneciente a la clase

Experimento, que puede lanzar dos excepciones declaradas como clases independientes.

```
public static double medida() throws MedidaInestable, MedidaErronea
```

Escribir otro método que, desde otra clase diferente, trate de retornar una medida de un experimento llamando al método ya existente y tratando las excepciones del siguiente modo:

- si se lanza MedidaInestable pone un mensaje de error y retorna 0.0
- si se lanza MedidaErronea pone un mensaje de error y retorna Double.NaN

Respuesta que se pide

<Poner aquí el código Java del método>