Proyecto UT3 (para hacer en casa y entregar en GitHub)

Objetivos

Saber:

- ☐ *definir y utilizar constantes y atributos (variables de instancia) dentro de una clase*
- ☐ *definir constructores*
- ☐ *definir y utilizar parámetros en los constructores y métodos*
- ☐ *definir métodos accesores y mutadores*
- ☐ *usar las sentencias de asignación y escritura*
- ☐ *construir una sentencia if*
- ☐ *construir una sentencia switch*
- ☐ *expresar el algoritmo correspondiente a un método*
- ☐ *usar operadores aritméticos y relacionales*
- ☐ *usar librería Math*

Antes de empezar

☐ Este ejercicio es para realizar de forma individual en casa.

☐ El proyecto de partida está en https://github.com/aetxabao/Caldera Deberás hacer un *fork* a tu cuenta y clonarlo en tu PC desde BlueJ tal y como se explicó en clase

☐ Una vez completado desde BlueJ haz un *push* del último *commit* a GitHub

☐ No olvides hacer el pull request indicando "*Terminado proyecto UT3 Caldera*" .

☐ Se valorará en la corrección que el programa esté probado (compila y ejecuta bien) y que esté claramente escrito y organizado (se respetan las reglas de estilo del lenguaje Java, nombres descriptivos, código no duplicado, ...)

☐ La fecha tope de entrega es el **Domingo 16 de Octubre** hasta las **23,30h**.

☐ Se anulará automáticamente la corrección del ejercicio y se **evaluará con un 0** si se detecta que ha sido copiado o dejado copiar a algún compañero/a

☐ Se penalizará si no se siguen las normas de entrega del ejercicio

- ☐ no se ha hecho un *fork* y no se sube vía *commit, push y pull request*
- ☐ hay algún *commit* posterior a esta fecha de entrega

☐ ***El profesorado podrá convocar al alumno/a para defender oralmente el proyecto***

Especificaciones

En este proyecto vamos a modelar mediante una clase el gasto de una caldera de una comunidad de vecinos. La clase va a permitir registrar información acerca de los consumos de gas, el mantenimiento de la caldera y los gastos de agua y luz que tiene. Las instancias que se creen de esta clase serán para comunidades de vecinos que tendrán un presupuesto inicial y estarán formadas por un número de vecinos. Interesa saber cuál ha sido el resultado global y el resultado para cada vecino, así como datos estadísticos.

Haz el *fork* del proyecto **Caldera** desde https://github.com/aetxabao a tu cuenta GitHub y desde BlueJ clona el proyecto a tu PC.

Abre el proyecto BlueJ. Tienes que completar únicamente la clase `Caldera`. La clase `DemoCaldera` no tienes que modificarla, te servirá para probar la otra.

No olvides escribir tu nombre después de la etiqueta *@author*.

Define dentro de la clase `Caldera` las siguientes constantes y atributos (deduce los tipos):

- dos constantes que indican los tipos de impuestos
  - IMP_IVA con el valor asociado 0.22
  - IMP_HIDROCARBUROS con el valor asociado 0.20
- tres constantes que indicas los posibles conceptos de gastos
  - AGUA con el valor asociado A
  - LUZ con el valor asociado L
  - NADA con el valor asociado N
- una constante para indicar ningún mes
  - NINGUNO con el valor asociado 0
- cuatro constante para indicar el periodo
  - PERIODO_OCTUBRE_DICIEMBRE con el valor asociado 1
  - PERIODO_ENERO_MARZO con el valor asociado 2
  - PERIODO_ABRIL_JUNIO con el valor asociado 3
  - PERIODO_JULIO_SEPTIEMBRE con el valor asociado 4

- los siguientes atributos o variables de instancia

  ☐ *vecinos* - guarda el número de vecinos que conforman la comunidad
  ☐ *presupuesto* - guarda el dinero con el que se pretendía hacer frente a los gastos

  ☐ *acumuladoConsumo* - acumula el importe sin impuestos del valor da la cantidad de gas consumida al precio que se haya adquirido (€).
  ☐ *acumuladoMantenimiento* – acumula los gastos debidos al mantenimiento de la caldera (€).
  ☐ *gastoAgua* – acumula el gasto en el concepto relativo al agua (€).
  ☐ *gastoLuz* – acumula el gasto en el concepto relativo a la luz (€).

  ☐ *mesMasConsumo* – almacena el número que indica el mes en el que se ha pagado más dinero debido al consumo de gas maxCosumo. Valores entre 1 y 12, el 1 se corresponde con enero y el 12 con diciembre.
  ☐ *maxConsumo* – almacena el valor del dinero que se ha pagado en el mes que más ha costado pagar el gas teniendo en cuenta la cantidad adquirida y el precio al que se adquirió (€).

  ☐ *mesMasCaro* – almacena el número que indica el mes en el que se ha pagado más caro el gas que se corresponde con maxPrecio. Valores entre el 1, que es enero, y el 12, que es diciembre.
  ☐ *maxPrecio* – almacena el valor del precio más caro del gas pagado en algún mes (€/KWh).

  ☐ *mesMasBarato* – almacena el número que indica el mes en el que se ha pagado más barato el gas que se corresponde con minPrecio. Valores entre el 1, que es enero, y el 12, que es diciembre.
  ☐ *minPrecio* – almacena el valor del precio más barato del gas pagado en algún mes (€/KWh).

  ☐ *periodoMasMantenimiento* – almacena el número que indica el periodo en el que se ha pagado más caro el mantenimiento que se corresponde con maxMantenimiento. Valores entre el 1 y el 4, que se corresponden con las constantes definidas.
  ☐ *maxMantenimiento* – almacena el valor del precio más caro del mantenimiento pagado en algún periodo (€).

  ☐ *mesMasGasto* – almacena el número que indica el mes en el que se ha pagado más dinero debido al consumo de agua o luz, maxGasto. Valores entre 1 y 12.
  ☐ *maxGasto* – almacena el valor del precio más caro del gasto pagado en algún mes como concepto de agua o luz (€).
  ☐ *conceptoMasGasto* – almacena el valor del del concepto definido en las constantes para representar el agua, la luz o nada.

No debes incluir más atributos. Solo los indicados. Respeta los nombres que se te dan

Completa los siguientes métodos:

- un **constructor** sin parámetros inicializa los atributos a 0, los meses y periodo con las constante NINGUNO y el concepto en el que se produce el mayor gasto con NADA. Utiliza las constantes.
- otro **constructor** con parámetros que inicializa los atributos igual que el constructor sin parámetros, pero fijando el número de vecinos y el valor del presupuesto con los valores pasados en los parámetros.

- accesores (**getters**) y mutadores (**setters**) para los atributos vecinos y presupuesto.

- **public void consumo(int mes, int gas, double precio)**

  Este método recibe tres parámetros que supondremos correctos. Son los datos del consumo de gas (KWh) realizado en un mes al precio indicado (Euros/KWh):

  - ☐ *mes* – nº del mes. Será un valor entre 1 y 12.
  - ☐ *gas* – cantidad de gas consumida ese mes en KWh.
  - ☐ *precio* – precio al que se ha adquirido el gas en Euros/KWh.

  Ej. *consumo(9, 15496, 0.067668);* significa que en septiembre se han consumido 15496 KWh a un precio de 0.067668 Euros/KWh.

  consumo(1, 98024, 0.127802); significa que en enero se han consumido 98024 KWh a un precio de 0.127802 Euros/KWh.

  consumo(3, 71668, 0.132327); significa que en marzo se han consumido 71668 KWh a un precio de 0.132327 Euros/KWh.

  consumo(8, 14469, 0.202504); significa que en agosto se han consumido 14469 KWh a un precio de 0.202504 Euros/KWh.

  A partir de estos parámetros el método debe hacer cálculos para actualizar el estado de la caldera (sus atributos) adecuadamente:
  - ☐ qué atributos hay que actualizar? *acumuladoConsumo*, *maxConsumo*, *mesMasConsumo*, etc....
  - ☐ el consumo (Euros) se calcula sin impuestos como multiplicación del gas (KWh) y el precio (Euros/KWh).
  - ☐ deberás utiliar sentencias **if** para determinar cuándo se realiza el máximo consumo y el precio máximo y mínimo y respectivamente actualizar esos estadísticos

- **public void mantenimiento(int periodo, double importe)**

Este método recibe dos parámetros que supondremos correctos. Son los datos relativos al gasto de mantenimiento en cada periodo.

- ☐ *periodo* – nº del periodo. Será un valor entre 1 y 4.
- ☐ *importe* – valor del gasto relativo al mantenimiento de la caldera en Euros.

Ej. *mantenimiento(1, 1552.10);* significa que en el periodo 1, de octubre a diciembre, el gasto de mantenimiento de la caldera ha supuesto 1552.10 Euros.

mantenimiento(2, 912.86); significa que en el periodo 2, de enero a marzo, el gasto de mantenimiento de la caldera ha supuesto 912.86 Euros.

A partir de estos parámetros el método debe hacer cálculos para contabilizar al acumulado del gasto de mantenimiento, determinar en qué periodo se produce el mayor gasto de mantenimiento y cuál es su valor. Análogamente al método anterior, para este caso será necesario incorporar un condicional.

- **public void gasto(int mes, char concepto, double importe)**

Este método recibe tres parámetros que supondremos correctos. Son los datos relativos al gasto en agua o luz en un mes.

- ☐ *mes* – nº del mes. Será un valor entre 1 y 12.
- ☐ *concepto* – agua o luz definida por un carácter, A para agua y L para luz.
- ☐ *importe* – valor del gasto en Euros.

Ej. *gasto(2, 'L', 558.34);* significa que en febrero se ha gastado en luz 558.34 Euros.

gasto(2, 'A', 239.42); significa que en febrero se ha gastado en agua 239.42 Euros.

Además de acumular el gasto de agua y luz se debe determinar cuál es el mayor gasto, en qué mes se produce y a qué concepto se corresponde.

- el método **printResultados()** - muestra en pantalla el resultado del periodo

```
==================
 RESULTADO GLOBAL
==================
Presupuesto:    38638.0
Consumo gas:    61688.26
Impuestos g.:   25909.07
Mantenimiento:  4157.58
Iva manto.:     914.67
Gasto agua:     2647.83
Iva agua:       582.52
Gasto luz:      4663.01
Iva luz:        1025.86
------------------
TOTAL : -62950.8 Euros.
------------------
==================
RESULTADO X VECINO
==================
Vecinos:        48
Aporte v.:      804.96
Gasto v.:       2116.43
Resultado:      -1311.47
------------------
El resultado ha sido NEGATIVO,
se tiene que pagar 1311.47 Euros.
El pago se pasara en
5 cuotas de 262.29 Euros.
------------------
```

- el método **printEstadisticas()** - muestra en pantalla información acerca del consumo máximo, meses en los que el gas ha sido más caro y más barato, en qué concepto se ha tenido el mayor gasto y cuándo ha sido el mantenimiento más caro.

```
==================
   ESTADISTICAS
==================
Max. consumo:   ENERO   12527.66
Mes mas caro:   AGOSTO  0.202504
Mes mas barato: SEPTIEMBRE      0.067668
Mayor gasto en: ABRIL   679.94  LUZ
P. mas manto.:  OCTUBRE-DICIEMBRE       1552.1
------------------
```

Los siguientes métodos deben ser llamados por los métodos **printResultados** y **printEstadisticas**:

- el método **String getNombreMes(int numMes)** - devuelve el nombre del mes asociado al parámetro pasado. Los nombres de los meses se devolverán en mayúsculas sin tildes. En caso de que numMes no esté comprendido entre 1 y 12 se devolverá el valor "NINGUNO". Se debe utilizar una sentencia **switch**, no se puede utilizar if.

- el método **String getNombreConcepto(char concepto)** - devuelve el nombre del concepto asociado al parámetro pasado. Se deberá devolver "AGUA", "LUZ" o "NADA". Utiliza **if**. En las comparaciones se deberá utilizar las **constantes** definidas.

- el método **String getNombrePeriodo(int numPeriodo)** - devuelve el nombre del periodo asociado al parámetro pasado. Se deberá devolver "OCTUBRE-DICIEMBRE", "ENERO-MARZO", "ABRIL-JUNIO", "JULIO-SEPTIEMBRE" o "NINGUN PERIODO". Utiliza **switch**. En las comparaciones se deberá utilizar las **constantes** definidas.

- el método **String analisisResultado(double resultado)** - devuelve un String con múltiples líneas informando de la forma de pago.

  Si el resultado es positivo devolverá un mensaje parecido:

```
El resultado ha sido POSITIVO,
se devolvera 45.52 Euros.
El pago se realizara en breve en
una transferencia.
```

  Si el resultado es negativo y menor o igual que 200 el mensaje será similar a:

```
El resultado ha sido NEGATIVO,
se tiene que pagar 114.56 Euros.
El pago se pasara en un solo cobro.
```

  En el caso que el dinero a pagar sea meno o igual que 600, habrá que pagar 1 o 2 cuotas de 200 Euros y el resto en otra cuota, por ejemplo:

```
El resultado ha sido NEGATIVO,
se tiene que pagar 513.33 Euros.
El pago se pasara en
2 cuotas de 200 Euros y
otro cobro de 113.33 Euros.
```

  En el caso contrario, cuando el dinero a pagar sea mayor que 600, habrá que pagar 5 cuotas del mismo valor para afrontar la deuda. Por ejemplo:

```
El resultado ha sido NEGATIVO,
se tiene que pagar 1311.47 Euros.
El pago se pasara en
5 cuotas de 262.29 Euros.
```

- el método **double redondeo2decimales(double valor)** - devuelve el valor redondeado con uno o dos decimales. Se puede hacer utilizando la función round de la librería Math.

```
38638 -> 38638.0
61688.255730000004 -> 61688.26
-62950.79553660001 -> -62950.8
```

- el método **int divisionEntera(double dividendo, int divisor)** - devuelve el cociente sin decimales de la división entera del dividendo entre el divisor. Ej. Si el dividendo es 447.55 y el divisor es 200, el valor devuelto sería 2.

- el método **double restoDivisionEntera(double dividendo, int divisor)** - devuelve el resto con decimales de la división entera del dividendo y el divisor. Ej. Si el dividendo es 447.55 y el divisor es 200, el valor devuelto sería 47.55.

Posible ejecución

Para probar que la clase Caldera funciona correctamente:

a) crea un objeto de la clase DemoCaldera

a) llama al método **iniciar()** y luego al método **printResultados()** y **printEstadisticas()**

Tendrás que obtener los resultados de la figura:

```
==================
 RESULTADO GLOBAL
==================
Presupuesto:    38638.0
Consumo gas:    61688.26
Impuestos g.:   25909.07
Mantenimiento:  4157.58
Iva manto.:     914.67
Gasto agua:     2647.83
Iva agua:       582.52
Gasto luz:      4663.01
Iva luz:        1025.86
------------------
TOTAL : -62950.8 Euros.
------------------
==================
RESULTADO X VECINO
==================
Vecinos:        48
Aporte v.:      804.96
Gasto v.:       2116.43
Resultado:      -1311.47
------------------
El resultado ha sido NEGATIVO,
se tiene que pagar 1311.47 Euros.
El pago se pasara en
5 cuotas de 262.29 Euros.
------------------
==================
   ESTADISTICAS
==================
Max. consumo:   ENERO   12527.66
Mes mas caro:   AGOSTO  0.202504
Mes mas barato: SEPTIEMBRE      0.067668
Mayor gasto en: ABRIL   679.94  LUZ
P. mas manto.:  OCTUBRE-DICIEMBRE       1552.1
------------------
```

Rúbrica evaluación	
constantes	1,00
atributos	5,00
construtores	6,00
accesores y mutadores	4,00
consumo	8,00
mantenimiento	4,00
gasto	6,00
printResultados	20,00
printEstadisticas	12,00
getNombreMes	3,50
getNombreConcepto	3,00
getNombrePeriodo	3,50
analisisResultado	15,00
redondeo2decimales	3,00
divisionEntera	3,00
restoDivisionEntera	3,00
Puntuación sobre	100
No compila	-5
Copia	-100