EJERCICIOS PROGRAMACION ORIENTADA A OBJETOS

Problema 1

Considera la clase Pelota. Una pelota se caracteriza por el nombre de su propietario, por su presión de inflado y por su operatividad (una pelota no operativa es una pelota que se ha reventado).Tiene 3 funciones:

- inflar: Una vez inflada la pelota, si la presión supera las 25 unidades, la pelota se hace inoperativa y la presión baja a 0.
- desinflar: Desinfla la presión de la pelota, respetando que la presión nunca sea negativa
- botar: Si la pelota no está operativa, muestra un mensaje. Si la pelota está operativa pero la presión es menor o igual a 5, muestra otro mensaje. Y si la presión es superior a 5 muestra otro mensaje

Se dan dos casos de uso, los programas Prog1 y Prog2, con sus respectivos resultados para ayudar en la construcción de la clase Pelota:

Prog1

```
public static void main (String[] args) {
    Pelota p1, p2;
    p1 = new Pelota("Maradona");
    p2 = new Pelota("Romario");
    p1.inflar(10);
    p1.inflar(10);
    p2.inflar(5);
    p2.inflar(5);
    System.out.println(p1.botar());
    System.out.println(p2.botar());
    p1.desinflar(5);
    p2.desinflar(16);
    System.out.println();
    System.out.println("Un rato más tarde...");
    System.out.println(p1.botar());
    System.out.println(p2.botar());
}
```

```
run:
la pelota de Maradona no bota porque se ha reventado
la pelota de Romario hace BOING-BOING
la pelota de Maradona no bota porque se ha reventado
BUILD SUCCESSFUL (total time: 0 seconds)
```

Prog2

```
public static void main (String[] args) {
    Pelota p1, p2, p3;
    p1 = new Pelota("Maradona");
    p2 = new Pelota("Romario");
    p3 = p1;
    p1.inflar(10);
    p2.inflar(10);
    p3.inflar(10);
    System.out.println(p1.botar());
    System.out.println(p2.botar());
    System.out.println(p3.botar());
}
```

```
run:
la pelota de Maradona no bota porque se ha reventado
la pelota de Romario hace BOING-BOING

Un rato más tarde...
la pelota de Maradona no bota porque se ha reventado
la pelota de Romario hace bing-boing
BUILD SUCCESSFUL (total time: 0 seconds)
```

Problema 2

A partir de los siguientes casos de uso, crea la clase Ascensor.

```
public static void main (String[] args) {
    Ascensor a1, a2;
    a1 = new Ascensor(0,10);
    a2 =  a1;
    a1.ir(6);
    a2.bajar();
    a1 = new Ascensor(-1,  a1.getMax()/2);

    a1.ir(a2.getPiso());
    a1.subir();
    a2.bajar();
    System.out.println("El ascensor 1 está en el piso: " + a1.getPiso());
    System.out.println("El ascensor 2 está en el piso: " + a2.getPiso());
}
```

```
run:
El ascensor 1 está en el piso: 5
El ascensor 2 está en el piso: 4
BUILD SUCCESSFUL (total time: 0 seconds)
```

```
public static void main (String[] args) {
    Ascensor a1, a2, a3;
    a1 = new Ascensor(10,20);
    a3 =  a1;
    a2 = new Ascensor(a3.getPiso()/2,  a1.getMin());
    a3.subir();
    a2.ir(a1.getPiso());
    a3 =  a2;
    a2 = new Ascensor(a3.getMin()-1,  a2.getMax());
    System.out.println(a1.getPiso()+a2.getPiso()+ a3.getPiso());
}
```

```
run:
20
BUILD SUCCESSFUL (total time: 0 seconds)
```

Problema 3

a) Indica cual será el resultado de la ejecución del siguiente programa:

```
public static void main (String[] args) {
    Ascensor ascl, asc2;
    int veces;
    ascl = new Ascensor(0, 4);
    asc2 = new Ascensor(-2, 7);
    ascl.ir(2);
    asc2.ir(ascl.getPiso()+2);
    veces   = 0;
    while (veces < 5)       {
        ascl.subir();
        asc2.bajar();
        veces = veces + 1;
    }
    System.out.println("El primer ascensor está en el piso: " + ascl.getPiso());
    System.out.println("El segundo ascensor está en el piso: " + asc2.getPiso());
}
```

b) Escribe la función escribirPiso a partir del siguiente caso de uso:

```
public static void main (String[] args) {
    Ascensor asc1, asc2, asc3;
    asc1 = new Ascensor(0, 19);
    asc2 = new Ascensor(-4, 4);
    asc3 = new Ascensor(-2, 10);

    asc1.ir(5) ;
    asc2.ir(asc1.getPiso()-10);
    asc3.ir(asc1.getPiso()+asc2.getPiso());

    while(asc1.getPiso()>0)         {
        asc1.bajar();
        asc2.subir();
        asc3.bajar();
    }

    System.out.print("Primer ascensor:");
    escribirPiso(asc1.getPiso());
    System.out.print("Segon ascensor:");
    escribirPiso(asc2.getPiso());
    System.out.print("Tercer ascensor:");
    escribirPiso(asc3.getPiso());
}
```

```
run:
Primer ascensor: se encuentra en la planta baja
Segon ascensor: se encuentra en la planta 1
Tercer ascensor: se encuentra en el subterraneo 2
BUILD SUCCESSFUL (total time: 0 seconds)
```

c) Escribir un programa que haga lo siguiente:

- Crea un ascensor que se pueda desplazar entre las plantas 10 y 30 (incluidas).
- Después, hace subir el ascensor creado hasta la planta 28 (directamente)
- Después hace bajar el ascensor 15 plantas, de una en una. Pero en esta parte sólo se puede utilizar el método bajar. Habrá que utilizar una iteración...
- Finalmente el programa escribe en que planta se encuentra el ascensor y se acaba.

d) Escribe un programa que haga lo siguiente:

- El programa estará preparado para trabajar con dos ascensores (dos variables de tipo Ascensor...)
- Primero de todo, consultará al usuario para pedirle por el primer ascensor, el piso más bajo y el más alto entre los cuales se puede desplazar.
- Construirá el primer ascensor con los límites indicados por el usuario.
- Llevará el ascensor construido a su planta central (si un ascensor puede parar en n plantas, la planta de en medio está n/2 plantas por encima de la más baja). Para esto se utilizara el método ir. Y para calcular a qué planta se utilizaran los métodos getMin y getMax.
- Construirá un segundo ascensor que se pueda desplazar desde la planta en que actualmente se encuentra el primer ascensor, hasta el límite superior más una unidad del primer ascensor.
- Hará subir, piso a piso, el segundo ascensor hasta la planta más alta a la que puede ir, utilizando sólo el método subir.
- Finalmente, el programa escribirá:
    - Limites, inferior y superior del primer ascensor
    - Planta en que se encuentra el primer ascensor
    - Limites inferior y superior del segundo ascensor

Problema 4

La clase TeleTonta ha sido programada por un programador inexperto que se ha hecho un lío considerable con el uso de this. Esto provoca que los objetos de esta clase no funcionen como se espera de ellos.

```
public static class TeleTonta {
    private boolean enMarcha;   //Si la tele está en funcionamiento
    private double frecuencia;  //Frecuencia sintonizada
    private int volumen;        //Volumen del sonido

    public TeleTonta (double frecuencia)
    {   //Constructor: una tele se fabrica sintonizando una determinada
        //frecuencia y siempre sale de fabrica parada y con el volumen a 0
        if  (frecuencia < 0)
            this.frecuencia = 0.0;
        else
            this.frecuencia = frecuencia;
        enMarcha = false;
        this.volumen = 0;
    }
    public boolean getEnMarcha()    {return(this.enMarcha);}
    public  int getVolumen()        {return(volumen);}
    public double getFrecuencia()   {return(this.frecuencia);}

    public void pulsarInterruptor()
    {   //Si la tele está funcionando la apaga y
        //si está apagada la pone en marcha.
        boolean enMarcha;
        enMarcha = this.enMarcha;
        if (enMarcha)   enMarcha  = !enMarcha;
        else            this.enMarcha =! enMarcha;
    }
    public void modificarVolumen (boolean arriba)
    {   //Controla el volumen. Si está en marcha aumenta o decrementa
        //una unidad el volumen
        if  (enMarcha){
            if (arriba) volumen = this.volumen + 1;
            else        this.volumen = volumen - 1;
        }
    }
    public void sintonizar(double frecuencia)
    {   //Si la tele está en marcha, se sincroniza a la frecuencia indicada
        //sólo si no es negativa
        if (this.enMarcha){
            if (this.frecuencia >= 0.0)
                frecuencia = this.frecuencia;
        }
    }
}
```

Indica donde crees que hay errores en los métodos que implementa la clase. Vuelve a escribir la clase de tal manera que los objetos funcionen correctamente (o sea, haciendo que el constructor y los métodos hagan aquello que indican los comentarios). Utiliza this sólo cuando sea imprescindible.

Problema 5

Haz una clase a la que llamarás CalculoNumeros. La clase tendrá cuatro atributos de tipo float. Estos atributos se podrán establecer a partir del constructor de la clase, o una vez creado el objeto, a través de un método que también escribirás (el método se llamará 'setNumeros'). La clase también tendrá cuatro métodos más, para devolver cada uno de los atributos de clase (Los métodos se llamarán 'getNumero1', 'getNumero2', etc.). Además, esta clase tendrá tres métodos adicionales para hacer cálculos con los atributos de clase: uno para calcular el mayor, otro para calcular el menor y otro para calcular el promedio.

a) Haz un programa que pida cuatro números por el teclado, y que diga cuál es el mayor, cual es el menor y cuál es su media. Para hacer los cálculos usarás la clase CalculoNumeros.

Problema 6

Crea una clase llamada FormatNumeros que estará en un paquete llamado numeros que incluirá cuatro métodos estáticos sobrecargados. Los métodos se llamarán formatNum, Uno recibirá un parámetro de tipo int, otro de tipo long, otro de tipo float y el último de tipo double. Estos métodos devolverán el número pasado como parámetro en forma de un objeto String con formato. Es decir, sí al primer método mencionado se le pasa el número 3400500 debería devolver el objeto String «3.400.500».

NOTA: Utiliza la clase NumberFormat

Problema 7

Considera la clase CuentaCorriente.

```
public class CuentaCorriente  {
    private String titular;       // Número del titular de la cuenta
    private int numero;           // Número de la cuenta
    private double saldo;         // Cantidad de euros en la cuenta
    private double limiteCredito; // Máximo saldo negativo permitido

    /*  Constructor */
    public CuentaCorriente (String titular, int numero) {
        this.titular     = titular;
        this.numero      = numero;
        this.saldo       = 0.0;     //  this no necesario
        this.limiteCredito = 0.0;   //  this no necessario
    }

    // Metodos de acceso.  En los 4 casos this no es necesario
    // No hay posibilidad de conflicto
    public String getTitular()      {return this.titular;}
    public int getNumero()          {return this.numero;}
    public double getSaldo()        {return this.saldo;}
    public double getLimitCredit()  {return this.limiteCredito;}

    public void setLimitCredit (double limiteCredito)       {
        if (limiteCredito>=0) {
            this.limiteCredito =  limiteCredito;  //  this necesario
        }
    }
    public void ingresar (double cuanto) {
        if (cuanto>0) {
            this.saldo = this.saldo + cuanto;    //this no necesario
        }
    }
```

```
    public boolean retirar (double cuanto) {
        double saldo;
        boolean ok;
        if (cuanto > 0) {
            saldo =  this.saldo - cuanto;   // this necesario
            if (saldo<0) {
                if (this.valAbs(saldo) <= this.limiteCredito) {
                    //this no necesario
                    ok = true;
                    this.saldo  =  saldo;   //  this necesario
                }
                else {
                    ok  =  false;
                }
            }
            else {
                ok =  true;
                this.saldo  =  saldo;       //  this necesario
            }
        }
        else  {
            ok  = false;
        }
        return ok;
    }
    private double valAbs (double n) {
        if (n>=0.0)
            return n;
        else
            return  -n;
    }
}
```

a) Escribir un procedimiento que dadas dos cuentas se iguale el saldo, pasando las cantidades de dinero que hagan falta de la cuenta que más tiene a la cuenta que menos tiene. El resultado es un valor booleano que indica si la operación se ha podido hacer

NOTA: Si en la primera cuenta hay un saldo de a euros y en la segunda hay un saldo de b euros lo que se pretende es que, al final, en cada cuenta haya un saldo de (a+b)/2 euros. Si suponemos que la cuenta que tiene más dineros es la que tiene saldo a, entonces se debe sacar de esta cuenta a-(a+b)/2 euros y pasarlos a la otra cuenta)

b) Escribir una función que dadas 3 cuentas indique si todas tres tienen el mismo saldo.

c) Escribir un programa que haga lo siguiente:

- Crear tres cuentas
- Ingresar 1000 euros en la primera cuenta, 500 en la segunda y 250 en el tercer
- Intentar igualar el saldo de la primera cuenta con la de la segunda (utilizar el procedimiento del apartado a)
- Escribir un mensaje indicando si ha sido posible o no igualar los saldos
- Intentar igualar el saldo de la segunda cuenta con la del tercero
- Escribir un mensaje indicando si ha sido posible o no igualar los saldos.
- Escribe un mensaje indicando las 3 cuentas tienen el mismo saldo o no (utilizar la función del apartado b)

Problema 8

Considera la clase CuentaCorriente del ejercicio anterior.

a) Escribe un procedimiento que dado una cuenta corriente (primer parámetro) y un número entero n (segundo parámetro) incremente el saldo de la cuenta en un n%. (n por ciento). Llamad al procedimiento darIntereses.

b) Considera la función que tiene por cabecera

```
private static int aleaJactaEst (int min, int max)
```

que devuelve un número entero aleatorio entre min y max (ambos comprendidos)

Escribe una función que dados 3 cuentas corrientes (parámetros), escoja al azar una de las tres y la devuelva como resultado. Para escoger uno de las cuentas esta función se basará en aleaJactaEst. Esta función, la que escoge uno de las 3 cuentas, se llamará rollTheDice.

c) Escribir un programa que esté preparado para trabajar con 3 cuentas corrientes, y que haga lo siguiente:

- Crear una cuenta corriente para Juana, con número de cuenta 1001
- Crear una cuenta corriente para María con número de cuenta 1002
- Crear una cuenta corriente para Raquel con número de cuenta 1003
- Ingresar en cada uno de las cuentas anteriores una cantidad de euros obtenida al azar y comprendida entre 100 y 200 (cada cuenta ha de tener su cantidad). Para generar la cantidad que se ingresa en cada cuenta utiliza la función aleaJactaEst
- Utilizando la función rollTheDice, escoger una de las 3 cuentas. La cuenta escogida se "guardará" en la variable llamada afortunada.
- Utilizando la función rollTheDice, escoger uno de las 3 cuentas. La cuenta escogida se "guardará" en la variable llamada malaPata
- Hacer que la cuenta afortunada reciba unos intereses del 10%
- Intentar retirar de la cuenta malaPata 150 euros
- Decir si la operación anterior se ha podido efectuar o no.

Problema 9

Considera la clase CuentaCorriente anterior.

Apartado A

Escribe un procedimiento especializado en la creación de una cuenta corriente a partir de los datos suministrados por el usuario. Los datos que obtendría del exterior son:

- titular de la cuenta

- numero de la cuenta

- saldo inicial de la cuenta

El resultado del procedimiento AbrirCuenta será una cuenta corriente acabada de crear, con el titular y el número obtenidos del usuario y al cual se le tendría ingresado el saldo inicial, también indicado por el usuario

Apartado B

Escribe un procedimiento sin retorno llamado abonarIntereses que dado una cuenta corriente ingrese los intereses. Los intereses son:

- si el saldo de la cuenta supera los 180000 €, un 5% del saldo

- si el saldo de la cuenta no supera los 180000 €, un 3% del saldo

Apartado C

Escribe un procedimiento de resultado booleano llamado cobrarComisión, que dado una cuenta corriente le cobre la comisión de mantenimiento. Aquí cobrar es sinónimo de retirar. La comisión es:

- Si la cuenta corriente tiene un límite de crédito superior a 0 entonces la comisión son 10 € fijos más un 1 % del límite de crédito.

- Si la cuenta corriente no tiene un límite de crédito superior a 0 entonces la comisión son los 10 € fijos y nada más.

El procedimiento devuelve true si ha podido retirar el dinero y false en caso contrario

Apartado D

Escribe una función, llamada masRico que dadas dos cuentas corrientes devuelva aquella que tiene más saldo (cualquiera de las dos si tienen exactamente el mismo saldo)

Apartado E

Escribe un programa que haga lo siguiente:

- Con la ayuda del procedimiento abrirCuenta crear dos cuentas corrientes que serán referenciados por las variables c1 y c2

- Abonar intereses a las cuentas referenciados por c1 y c2 (haciendo uso del procedimiento desarrollado a tal efecto)

- Cobrar la comisión a la cuenta c1. Si lo puede hacer ha de escribir un mensaje indicándolo e indicando el número de la cuenta. Lo mismo si no puede hacerlo.

- Con la ayuda de la función desarrollada en el apartado anterior, escribe el nombre del titular de la cuenta que tiene más dinero.

Problema 10

Se nos ha encargado que construyamos una clase llamada Boleto, con el objetivo de poderlo utilizar en un programa que simula sorteos de lotería. Los objetos de esta clase tendrán las siguientes características:

Atributos: un atributo (string) llamado nomProp que identificara al propietario, tres atributos (int) llamados priNum, segNum, terNum que contendrán los 3 números con que el boleto participara en un sorteo, un atributo (boolean) llamado validado que indicará si el boleto se valida o no para participar en un sorteo, y un último atributo (boolean) llamado gastado que indicará si el boleto ya ha sido gastado en un sorteo o no.

Constructor: con un único parámetro que será el nombre del propietario del boleto. Este constructor, además de dar nombre al propietario del boleto, hará lo siguiente: inicializará los atributos priNum, segNum y terNum con el valor-1, el atributo validado con el valor false y el atributo gastado con el valor false;

Métodos de acceso a los atributos: para cada atributo habrá el correspondiente método getxxx para consultar el valor

Métodos de modificación del valor de los atributos: los atributos priNum, segNum, terNum tendrán el correspondiente método setxxx para modificar su valor.

Estos métodos operaran de la siguiente manera: si el boleto está gastado o validado entonces no hacen nada. Si el billete no está gastado modifican el valor del atributo si el valor que se da (parámetro) está comprendido entre 1 y 49 (incluidos). Si no es así tampoco hacen nada.

Otros métodos:

- Un método llamado validar, de resultado booleano, que:
  - Devuelve false si el boleto está gastado.
  - Si el boleto no está gastado:
    - Mirara si priNum, segNum y terNum son todos diferentes de -1. Si alguno de ellos es -1 devolverá false
    - Si alguno de ellos está repetido devolverá false
    - Si ninguno de ellos no es -1 ni hay repeticiones, pondrá el valor true del atributo validado y devolverá true
- Un método llamado reaprovechar que:
  - Poner el valor false al atributo gastado (marca el boleto como no gastado)
  - Poner el valor false al atributo validado (marca el boleto como no validado)
- Un método llamado cuantosAciertos, de resultado int y con 3 parámetros (también int). Los tres parámetros representaran los tres números que han salido en el sorteo y se puede garantizar que serán diferentes entre ellos. Si el boleto no está validado o está gastado devolverá el valor 0, sin hacer nada más. Si el boleto está validado y no gastado, pondrá el valor true al atributo gastado y devolverá el número de aciertos que contiene el boleto.

Problema 11

Otro programador nos ha suministrado la clase Bombo. Esta clase nos permite simular sorteos:

```
public class Bombo{
    int NumBolas;
    int numExtracciones;

    public Bombo(int bolas)
    {   //Crea un bombo con el numero bolas, si es mayor que 0
        //Si es 0 o menor, crea un bombo de 49 bolas
        //Las bolas del bombo estaran numeradas correlativamente
    }
    public int sacarBola()
    {   //Saca al azar una bola del Bombo, si quedan
        //Si ya no quedan bolas, porque se han hecho
        //numBolas extracciones, entonces devuelve -1
        return -1;
    }
    public void reinicializar()
    {   //Vuelve a rellenar el bombo con todas las bolas
        //como si nunca se hubiera sacado ninguna

    }
}
```

Se pide construir un programa que haga lo siguiente:

- Construir tres boletos, una para Juan, otra para María y la última para Carla.
- Poner en cada boleto los siguientes números:
    - o Para el boleto de Juan: 17, 23, 45
    - o Para el boleto de María: 12, 33, 49
    - o Para el boleto de Carla: 6, 17, 35
- Validar los tres boletos. Cada vez que se valida un boleto se debe indicar si ha quedado validado o no.
- Crear un bombo de 49 bolas.
- Extrae tres bolas del bombo
- Escribe cuantos aciertos tiene cada participante en el sorteo

Vuelve a escribir el programa de manera que los números jugados por Juan, María y Carla sean elegidos al azar. ¿Cómo? Muy fácil: se puede utilizar un bombo de 49 bolas para generarlos al azar. Recordad que después de cada utilización del bombo este se debe de reinicializar.

Problema 12

Utilizando la clase Bombo del problema anterior, escribe un programa que simule un sorteo de la lotería primitiva (49 bolas; se han de hacer 7 extracciones: los 6 números de la combinación ganadora y el número complementario). El programa se debe limitar a escribir los números extraídos. Por ejemplo:

6, 19, 24, 31, 43, 44 Complementario: 28

Problema 13

Modificad el programa anterior para que, además, indique el número del reintegro. El reintegro se saca de un bombo en el que hay 10 bolas numeradas del 0 al 9. Atención: los objetos de la clase Bombo numeran las bolas a partir de 1. Ejemplo:

5, 12, 33, 37, 38, 41 Complementario: 35; Reintegro: 0

Problema 14

El famoso restaurante Chez Lentilles ofrece cada semana 5 platos, de entre todas sus especialidades, como promoción especial. El restaurante ha encargado un pequeño programa para saber:

a) que beneficio bruto le reportan las ventas de estos 5 platos

b) cual de los 5 platos son los más populares (el gerente del restaurante considera que un plato es popular cuando en una semana se sirven más de 20 unidades).

Para resolver este problema, un programador experto ha preparado la clase plato:

```
public class plato {
    private String nombre;       // Nombre del plato
    private boolean postre;      // true si es un postre
    private boolean vegetariano;// true si es un plato vegetariano
    private double coste;        // Lo que cuesta, en euros, elaborarlo
    private double precioVenta; // Precio, en euros, que figura en la carta
    private int ventas;          // Unidades vendidas
    public plato (String nombre, boolean postre, boolean vegetariano) {
        this.nombre = nombre;
        this.postre = postre;
        this.vegetariano = vegetariano;
        this.coste = 0.0;
        this.precioVenta = 0.0;
        this.ventas = 0;
    }
    public String getNombre() { return(nombre);}
    public boolean esPostre () { return(postre);}
    public boolean esVegetariano () { return(vegetariano);}
    public double getCoste() { return(coste);}
    public double getPrecioVenta() { return (precioVenta);}
    public int getVentas() { return(ventas);}
    public void setCoste (double coste) {
        if (coste>0.0) {
            this.coste=coste;
        }
    }
```

```
    public void setPrecioVenta (double precioVenta) {
        if (precioVenta>this.coste) {
            this.precioVenta = precioVenta;
        }
    }
    public void setVentas (int ventas) {
        if (ventas>0) {
            this.ventas=ventas;
        }
    }
    public double beneficioBruto () {
        return((this.precioVenta - this.coste)*this.ventas);
    }
}
```

Se pide resolver el problema planteado haciendo uso de esta clase y siguiendo los siguientes pasos:

a. Escribe un procedimiento especializado en lectura que haga una pregunta al usuario encargado de teclear los datos de los platos y proporcione como resultado el valor true si la pregunta ha sido respondida afirmativamente, o el resultado false si la pregunta ha sido respondida negativamente. La pregunta a realizar será el parámetro del procedimiento. Para responder afirmativamente, el usuario tendrá que teclear la letra s y para responder negativamente la letra n. Este procedimiento tendrá que insistir hasta que la letra tecleada sea alguna de estas dos. La cabecera de este procedimiento ha de ser:

`private static boolean leerSiNo (String pregunta)`

b. Escribe un procedimiento que cree y devuelva un plato, construido a partir de los datos obtenidos desde el teclado (nombre del plato, respuesta a la pregunta de si el plato es un postre y respuesta a la pregunta si el plato es vegetariano). Hacer uso del procedimiento leerSiNo. La cabecera de este procedimiento ha de ser:

`private static plato generarPlato()`

c. Escribe un procedimiento que dado un plato (como parámetro) incorpore la información referente al coste de su elaboración, el precio de venta y el número de unidades que se han servido. La cabecera de este procedimiento ha de ser:

`private static void añadirDatos (plato p)`

d. Escribe un procedimiento especializado en escritura que dado un plato (como parámetro) y un número entero (también como parámetro) que indica que cantidad de unidades vendidas se deben superar para considerar que el plato es popular, escriba el nombre, una indicación de si es un postre y una indicación de si es vegetariano, pero sólo en el caso de que el número de ventas del plato supere el límite dado. La cabecera de este procedimiento ha de ser:

`private static void escPlato (plato p,  int q)`

e. Con la ayuda de los procedimientos anteriores ya se puede resolver todo el problema. El punto de entrada en ejecución (private static void main) ha de hacer lo siguiente:

i. Crear los 5 platos, haciendo uso del procedimiento generarPlato (harán falta 5 variables de tipo plato para mantener las referencias a los platos creados).

ii. Obtener el resto de datos de los 5 platos creados en el paso anterior, haciendo uso del procedimiento añadirDatos.

iii. Calcular y escribir la ganancia bruta total en base al beneficio bruto de cada plato.

iv. Escribe los datos de los platos más populares utilizando el procedimiento escPlato.

Problema 15

Una productora de televisión quiere efectuar un casting para seleccionar participantes en un programa de televisión tipo "Operación Triunfo"

Durante la primera fase del casting, los aspirantes son evaluados, de 5 en 5, por un jurado de especialistas que valoran los siguientes aspectos de cada uno:

• Su timidez (de 0 a 10 en incrementos unitarios. 0 quiere decir nada tímido y 10 extremadamente tímido)

• Su aspecto físico (de 0 a 10 en incrementos unitarios. 0 quiere decir muy mal aspecto físico y 10 quiere decir muy buen aspecto físico)

• El resultado de una pequeña prueba de canto y baile (de 0 a 10 en incrementos unitarios. 0 quiere decir nada satisfactoria y 10 quiere decir muy satisfactoria)

Además, la productora conoce a priori, de cada aspirante:

- Su nombre
- Su edad
- Su sexo aparente (hombre o mujer)
- Si ha recibido o no clases de baile
- Si ha recibido o no clase de canto.

A partir de toda esta información, se otorga a cada aspirante una puntuación final, siguiendo los criterios siguientes:

- Otorgar 100 puntos por cada punto obtenido en la prueba de canto y baile.
- Restar 20 puntos por cada punto de timidez
- Sumar 100 puntos más si la puntuación de aspecto físico es igual o superior a 7
- Restar 100 puntos si la puntuación de aspecto físico es igual o inferior a 3
- Sumar 50 puntos más si es un hombre (los hombres tienen mejor aceptación entre el público que las mujeres)
- Sumar 50 puntos más si el aspirante ha hecho clases de baile
- Sumar 50 puntos más si el aspirante ha hecho clases de canto
- Restar 20 puntos por cada año en que el aspirante supera la treintena

Una vez los 5 aspirantes han estado sometidos a la valoración del jurado, se calcula la media de puntuación de los 5 y pasan a la siguiente fase del casting todos aquellos que superen esta mediana.

Se pide:

1. Codificar una clase a la que llamaremos Aspirante y que presentará las siguientes características:

a) El constructor de la clase recibirá como parámetros la siguiente información:

- el nombre del aspirante
- su edad
- si es o no un hombre

- si ha recibido o no clases de baile
- si ha recibido o no clases de canto

b) La timidez, aspecto físico y los puntos de la prueba, el constructor considerara que, de entrada tienen valor 0

c) El nombre, la edad, el hecho de si es o no un hombre, el hecho de si ha recibido o no clases de baile y el hecho de si ha recibido o no clases de canto no es podrán modificar después de la creación del aspirante.

d) Habrá un método público para consultar cada atributo.

e) Habrá un método público para dar valor a cada atributo que sea modificable.

f) Habrá un método público para calcular la puntuación final del aspirante.

2. Codificar un programa (y todos los procedimientos que se considere adecuados), que haga uso de la clase Aspirante y que permita:

a) Recaudar los datos correspondientes a cada uno de los 5 aspirantes que han de ser evaluados.

b) Recaudar las puntuaciones otorgadas por el jurado a cada aspirante.

c) Escribir los datos (como mínimo el nombre y la puntuación total) de aquellos aspirantes que han superado la fase.

- Además el programa (no la clase Aspirante) tendrá que tener las siguientes características de robustez en referencia a la introducción de datos:

- No permitir ninguna puntuación fuera del rango [0,10].

- No permitir que la edad de ningún aspirante sea menor de 18 ni mayor de 40.

Problema 16

Una empresa inmobiliaria, que vende viviendas de alto "standing" de segunda mano, en zonas costeras y al mejor postor, escoge, semanalmente, de su bolsa de viviendas no vendidas, los 4 que menos interés han suscitado entre sus clientes, la mayoría de los cuales son jubilados de origen británico. En el momento de la elección, la empresa conoce, de cada vivienda los siguientes datos: un numero identificador de la vivienda, un entero positivo, el pueblo (o ciudad) donde se encuentra, el precio mínimo de venda, expresado en euros, los metros cuadrados útiles de que dispone y si tiene o no vistas directas al mar.

Una vez efectuada la elección de estas viviendas poco interesantes, la empresa los promocionará ofertándolas a un precio mínimo más bajo. En concreto, las viviendas que no tengan vistas al mar serán rebajadas 1000 euros por metro cuadrado, hasta un máximo de 100000 euros (esto quiere decir que las viviendas de más de 100 metros cuadrados sólo serán rebajados en 100000 euros) mientras que las que tienen vistas al mar serán rebajados 750 euros por metro cuadrado, hasta un máximo de 75000 euros.

Durante un cierto periodo de tiempo, las viviendas escogidos son ofertadas a este precio mínimo más bajo, a la espera que algún cliente se interese. Pasado este periodo de tiempo puede pasar, por cada vivienda, que finalmente se haya vendido, o no. Si se ha vendido se dispondrá del nombre del comprador y de la cantidad que pagará (expresada en euros y que siempre será igual o mayor que el precio mínimo de oferta).

Diseña una clase ViviendaOferta y un programa que permita:

1. Introducir los datos de las 4 viviendas que serán ofertadas (número identificador, pueblo o ciudad donde se encuentra, metros cuadrados de que dispone, precio mínimo de venta y si tiene o no vistas al mar)

2. Escribe la lista de las 4 viviendas donde, por cada vivienda conste:
    a. Numero identificador de la vivienda
    b. Pueblo o ciudad donde se encuentra
    c. Precio mínimo de venda
    d. Precio mínimo de venta durante el período de oferta

3. Introducir el precio en euros de 1 libra esterlina

4. Por cada uno de las 4 viviendas introducir datos indicando
    a. si ha sido vendido o no
    b. (Y si ha sido vendido) nombre del comprador
    c. (Y si ha sido vendido) precio ofertado en euros

5. Escribe una lista de las viviendas no vendidas en que conste:
    a. Numero identificador de la vivienda
    b. Pueblo o ciudad donde se encuentra

6. Escribe una lista de las viviendas vendidas en que conste:
    a. Numero identificador de la vivienda
    b. Nombre del comprador
    c. Precio que pagará el comprador expresado en libras esterlinas

Problema 17

Dada la siguiente clase Libro

```
public class Libro {
    private String titulo;
    private int puntos;
    private int votantes;
    private int categoria;
    public Libro (String titulo, int categoria) {
        this.titulo = titulo;
        puntos = 0;
        votantes = 0;
        this.categoria = categoria;
    }
    public String getTitulo() {return titulo;}
    public int getPuntos()  {return puntos;}
    public int getVotantes() {return votantes;}
    public int getCategoria() {return categoria;}
    public void setCategoria (int categoria) {
        if (categoria>=1 && categoria<=4) {
            this.categoria  = categoria;
        }
    }
    public void votar (int puntos)  {
        if (puntos>=0 && puntos<=10)        {
            this.puntos = this.puntos + puntos;
            votantes = votantes + 1;
        }
    }
    public double calificacion () {
        if  (votantes==0) {return -1;}
        else {return puntos*1.0/votantes;}
    }
}
```

a) Escribe un procedimiento, llamado hacerVotacionAzar, que dado como parámetro un libro le otorgue una cantidad de puntos (votar) igual al producto de su categoría por un número entero entre 1 y 3 elegido al azar

b) Escribe una función llamada mejorDeTres que dados como parámetros 3 libros escoja (devuelva) aquel que tiene una calificación más alta.

c) Escribe un procedimiento especializado en escritura, de nombre felicitar, que dado como parámetro un libro, escriba la siguiente frase:

... and the winner is <titulo>, donde <titulo> ha de ser el título del libro dado como parámetro

d) Escribe un programa (punto de entrada en ejecución) que haga lo siguiente, utilizando siempre que se pueda los procedimientos de los apartados anteriores

- Crear (y mantener referenciados por variables) tres libros con los títulos “Objetos desinhibidos”, “Métodos alegres” y “Atributos en acción”. Para cada uno de ellos, la categoría será un número entre 1 y 4 elegido al azar
- Hacer una votación al azar para el primer libro, otra para el segundo y una última para el tercero
- De entre los tres libros, seleccionar el que tiene una calificación más grande
- "Felicitar" al libro ganador

Problema 18

Disponemos de la clase Libro y de la conocida función

```
private static int aleaJactaEst (int min,  int max)
```

que proporciona (devuelve) un número entero aleatorio entre min y max (ambos comprendidos)

a) Escribe un procedimiento sin retorno (no función), llamado votacionAzarosa, que dado como parámetros dos libros (los cuales se pueden llamar 11 y 12) le otorgue al primero una cantidad de puntos escogida al azar entre 0 y 10 y al segundo una cantidad de puntos, también escogida al azar entre 4 y 8

b) Escribe una función (de resultado Libro), llamada patitoFeo que dados como parámetros tres libros escoja (retorne) aquel que tiene una calificación más baja

c) Escribe un procedimiento especializado en escritura, de nombre reprender, que dado como parámetro un libro, escriba la siguiente frase:

> El libro titulado <titulo> ha recibido <puntos> puntos y ha obtenido la ridícula calificación de <calificación>
>
> donde <titulo> ha de ser el título del libro dado como parámetro, <puntos> su número de puntos y <calificacion> su calificación.

d) Escribe un programa (punto de entrada en ejecución) que haga lo siguiente, utilizando siempre que se pueda los procedimientos de los apartados anteriores

- Crear (y mantener referenciados por variables) tres libros con los títulos “Clases heredadas”, “Procedimientos públicos” y “Parametros descomunales”. Para cada uno de ellos, la categoría será un número entre 2 y 3 elegido al azar
- Hacer una votación azarosa para el primer y el segundo libro, otra para el segundo y el tercero, y una última para el primero y el tercero.
- De entre los 3 libros, seleccionar el que tiene una calificación más baja
- "reprender" el libre seleccionado en el punto anterior

Problema 19

A continuación se da la descripción de una clase. Se pide codificarla.

Nombre de la clase: Juguete.

Atributos: nombre (String), peso, precio (reales), edad (entero).

Constructor: recibe dos parámetros, el nombre y el peso del juguete. Precio y edad quedaran inicializados con el valor cero.

Métodos getXxx: uno para cada atributo.

Métodos setXxx: sólo para uno, el atributo edad. Se debe de verificar que la edad dada está comprendida entre 0 y 99. Si no es así no efectúa ningún cambio.

Otros métodos:

- Un método booleano llamado precioEstablecido que responde true si el precio es diferente de cero y false en caso contrario
- Un método (sin retorno) llamado establecerPrecio. Un único parámetro (double) que representa una cantidad de euros por gramo. El precio del juguete se establece como el producto de su peso por la cantidad de euros por gramo.

Problema 20

A continuación se da la descripción de una clase. Se pide codificarla.

Nombre de la clase: InformeEstudiante.

Atributos: nombre (String), ex1, ex2, comportamiento (entero), atEspecial (booleano).

Constructor: recibe dos parámetros, el nombre del estudiante y un booleano que indica si el estudiante requiere o no atención especial. El resto de atributos (ex1, ex2 y comportamiento) quedaran inicializados con el valor -1;

Métodos getXxx: para los atributos nombre, comportamiento y atEspecial

Métodos setXxx: sólo uno para el atributo comportamiento. No hará falta ningún tipo de verificación.

Otros métodos:

- Un método sin retorno llamado evalúa. Recibe dos parámetros enteros (ex y val). Si el primer parámetro tiene valor 1 entonces pone en el atributo ex1 el valor del segundo parámetro. Si el primer parámetro tiene el valor 2 entonces pone en el atributo ex2 el valor del segundo parámetro. Si el primer parámetro tiene cualquier otro valor, no hace nada
- Un método de retorno entero llamado miraExamen. Un único parámetro entero (llamado ex). Si el valor del parámetro es 1 devuelve el valor del atributo ex1. Si es 2, devuelve el valor del atributo ex2. En cualquier otro caso devuelve -1.
- Un método de retorno real (double), sin parámetros y llamado notaFinal. Este método calcula la nota final del estudiante de la siguiente manera:

  Si alguno de los atributos ex1, ex2 o comportamiento tiene valor -1 entonces el resultado es -1. En caso contrario la nota base es la media de ex1 y ex2 (hacer lo que haga falta para no perder decimales). Si el comportamiento es igual o superior a 5 entonces la nota base se multiplica per 1.1. En caso contrario se multiplica por 0.9. Finalmente, los estudiantes que requieren atención especial reciben un punto adicional.

Problema 21

Una empresa discográfica quiere hacer una encuesta para averiguar cuáles de sus cantantes (interpretes) es el más popular y cual el que menos. Para resolver este problema se nos suministra la clase Interprete y un programa que resuelve el problema. Nuestra tarea será escribir el código de algunos de los procedimientos utilizados por el programa.

```
public static class Interprete{
    private int id;
    private int puntos;
    private String nombre;
    public Interprete(int id, String nombre)
    {
        this.id=id;
        this.nombre=nombre;
        puntos=0;
    }
    public int getId(){return(id);}
    public int getPuntos(){return(puntos);}
    public String getNombre(){return(nombre);}
    public void añadirPuntos(int qp){
        puntos = puntos + qp;
    }
}
```

```
public static void main (String[] args) {
    int qCantantes;
    int ind;            //Para indexar la tabla durante la carga
    int identificador;  //Identificador del cantante que se vota
    int posMax, posMin; //Posición del máximo y del mínimo
    System.out.println();
    qCantantes = leerEnteroPositivo("Cuantos cantantes seran encuestados?");
    Interprete tCantante[] = new Interprete[qCantantes];
    //Carga de la tabla de cantantes
    ind = 0;
    while (ind < qCantantes) {
        tCantante[ind] = obtenerInterprete();
        ind = ind + 1;
    }
    //Proceso de votación
    System.out.println("Se inicia el proceso de votación: ");
    identificador = leerEnteroPositivo("Identificador del cantante (1000 para finalizar): ");
    while (identificador!=1000) {
        ind = buscarIdentificador(tCantante, identificador);
        if (ind == -1) {
            System.out.println("Cantante inexistente. Votación descartada");
        }
        else {
            tCantante[ind].añadirPuntos(1);
        }
        identificador = leerEnteroPositivo("Identificador del cantante (1000 per finalizar): ");
    }
```

```
    //Búsqueda del cantante más y menos votado
    ind = 0;
    posMax = 0;
    posMin = 0;
    while (ind < qCantantes - 1) {
        if (tCantante[ind].getPuntos()>tCantante[posMax].getPuntos()) {
            posMax=ind;
        }
        if (tCantante[ind].getPuntos()<tCantante[posMin].getPuntos()) {
            posMin=ind;
        }
        ind = ind + 1;
    }
    if (tCantante[ind].getPuntos() > tCantante[posMax].getPuntos()) {
            posMax=ind;
    }
    if (tCantante[ind].getPuntos() < tCantante[posMin].getPuntos()) {
            posMin=ind;
    }
    /* Dar resultdos */
    System. out. println();
    System. out. println("Cantante más votado: ");
    escCantante(tCantante[posMax]);
    System.out. println();
    System.out.println("Cantante menos votado: ");
    escCantante(tCantante[posMin]);
}
```

El programa hace lo siguiente:

• Averigua cuantos cantantes serán encuestados.

• Crea una tabla con capacidad para todos los cantantes.

• Pide los datos de cada cantante y va rellenando la tabla (un cantante en cada posición).

• Lee una secuencia de identificadores de Interprete, que se acaba con el identificador 1000 (este identificador se utiliza para marcar la finalización de la secuencia y se garantiza que no corresponde a ningún cantante de los que hay en la tabla). Para cada identificador leído:

- Se averigua en qué posición de la tabla está el cantante con aquel identificador.
- Se incrementa el número de votos de aquel cantante en una unidad.

• Una vegada recogidos todos los votos el programa determina en qué posición de la tabla se encuentra el cantante más votado y en cual el menos votado.

• Finalmente, escribe los datos de los cantantes en estas posiciones.

Escribe el código de los procedimientos obtenerInterprete, buscarIdentificador y escCantante.

Problema 22

En este problema se tiene que hacer uso de la clase Informe suministrada.

```
public class Informe {
    private String nombreEstudiante, nombreAsignatura;
    private int actitud, procedimientos;
    private int conocimientos1, conocimientos2;
    public Informe (String nombreEstudiante,  String nombreAsignatura){
        this.nombreEstudiante = nombreEstudiante;
        this.nombreAsignatura = nombreAsignatura;
        this.actitud = -1;
        this.procedimientos = -1;
        this.conocimientos1 = -1;
        this.conocimientos2 = -1;
    }

    public  String getNombreEstudiante() {return this.nombreEstudiante;}
    public  String getNombreAsignatura() {return this.nombreAsignatura;}
    public  int getActitud() {return this.actitud;}
    public  int getProcedimientos () {return this.procedimientos;}
    public  int getConocimientosl () {return this.conocimientos1;}
    public  int getConocimientos2() {return this.conocimientos2;}

    public  void setActitud (int actitud){
        if  (actitud==0 || actitud==5 || actitud==10) {
            this.actitud =  actitud;
        }
    }
    public  void setProcedimientos(int  procedimientos) {
        if  (procedimientos >=0 && procedimientos <= 10) {
            this.procedimientos = procedimientos;
        }
    }
```

```
    public  void setConocimientos (int cual, int cuanto) {
        if  (cuanto>=0 && cuanto<=10) {
            if (cual==1) { this.conocimientos1 = cuanto;}
            if (cual==2) { this.conocimientos2 = cuanto;}
        }
    }
    public boolean evaluable () {
        return (actitud!=-1 && procedimientos!=-1 &&
            conocimientos1!=-1 && conocimientos2 != -1);
    }
    public double calcularNotaConocimientos() {
        if (this.conocimientos1==-1 || this.conocimientos2==-1){
            return(-1);
        }
        else
            return ((this.conocimientos1 + this. conocimientos2)/2.0);
    }
    public double calcularNotaInforme (){
        if  (this.evaluable()) {
            return (this.actitud*0.2 + this.procedimientos*0.3 +
            calcularNotaConocimientos()*0.5);
        }
        else return(-1);
    }
}
```

Apartado A

Escribe un procedimiento de cabecera

private void rebajarConocimientos (Informe inf)

que reste 3 puntos de la nota conocimientos1 del informe dado y dos puntos de la nota conocimientos2, sólo si el informe es evaluable. Si no lo es no hace nada.

Apartado B

Escribe un procedimiento llamado mejorarActitud que dado un informe mejore la calificación de la actitud:

Si es 0 la pasa a 5, si es 5 la pasa a 10, si el informe no es evaluable no hace nada.

Apartado C

Escribe un procedimiento, llamado mostrarDatos, especializado en escritura, que dado un informe escriba:

- el nombre del estudiante
- el nombre de la asignatura
- la nota de conocimientos
- la nota del informe

Apartado D

Escribe una función, llamada mejorInforme que dados dos informes devuelva aquel que tiene una nota de informe más grande.

```
private Informe mejorInforme (Informe pr, Informe se) {
```

Apartado E

Escribe un punto de entrada en ejecución que haga lo siguiente:

- Crear dos informes (para Antonio y Bartomeu) de la asignatura Cocina Oriental
- A Antonio otorgarle la máxima calificación en todo
- A Bartomeu ponerle un 5 de actitud y un 6 al resto.
- A Antonio rebajarle los conocimientos
- A Bartomeu mejorarle la actitud
- Haciendo uso de la función mejorInforme escoger el mejor informe
- Mostrar los datos del mejor informe, haciendo uso del procedimiento mostrarDatos

Problema 23

A continuación se da la descripción de una clase. Codificadla.

Nombre de la clase: Revista.

Atributos: nombre, género (String), precio (real), páginas (entero) conCD (booleano).

Constructor: recibe tres parámetros, el nombre, el género y el valor booleano que indica si la revista lleva un CD o no. Los atributos correspondientes al precio y al número de páginas se inicializan con los valores 0.0 y 0, respectivamente.

Métodos getXxx: uno para cada atributo.

Métodos setXxx: sólo uno, para el atributo páginas.

Otros métodos:

• Un método booleano llamado especial que responda true si la revista lleva CD y tiene más de 300 páginas.

• Un método (sin retorno) llamado establecerPrecio. Este método, cuando se invoca, da valor al atributo precio, siguiendo el siguiente criterio:

- Si la revista tiene menos de 100 páginas: 3 euros
- Si la revista tiene entre 100 y 200 páginas: 4 euros
- Si la revista tiene más de 200 páginas: 4.5 euros
- Si la revista lleva CD 0.5 euros adicionales

Determina que parámetros hacen falta para este método, si es que hacen falta.

Problema 24

Se quiere codificar una clase los objetos de la cual recuerdan una visión simplificada de las neuronas del sistema nervioso. A grandes rasgos una neurona es una célula que recibe estímulos químicos. Estos estímulos van "cargando" la neurona. Cuando la carga supera un determinado umbral, la neurona responde "disparando" es decir, descargándose generando una señal eléctrica. Estas son las particularidades de la clase que se debe codificar.

Nombre de la clase: Neurona.

Atributos:

- acumuladora (booleano): si es true quiere decir que cuando recibe un estimulo este se suma con los recibidos anteriormente. Si es false quiere decir que un estimulo sustituye el anterior

- inputA, inputs (double): estímulos de entrada

- pesoA, pesoB (double): peso ponderación que tienen los estímulos A y B en la carga de la neurona

- limiteDescarga (double): valor a partir del cual se considera que la estimulación es suficiente intensa como para permitir la descarga

- descargaMaxima (double): máxima intensidad que puede tener el disparo de la neurona.

Constructor: recibe tres parámetros. Un valor booleano que servirá para inicializar el atributo acumuladora y dos atributos double que servirán para inicializar los atributos pesoA y pesoB. Los atributos inputA e inputB se inicializaran con el valor 0.0. El atributo limiteDescarga se inicializara con el valor 50.0 y descargaMaxima se inicializará con el valor 100.0

Métodos getXxx: para los atributos acumuladora, pesoA, pesoB, limiteDescarga y descargaMaxima

Métodos setXxx:

- uno para limiteDescarga

- uno para descargaMaxima. Este método set tiene la particularidad de que además de modificar el atributo descargaMaxima también modifica el atributo limiteDescarga, dándole un valor igual a la mitad de la descargaMaxima.

Otros métodos:

- un método sin retorno y de parámetro double llamado cargarA. Este método simula la estimulación de la entrada A (inputA) de la neurona. Si la neurona es acumuladora suma a inputA el valor del parámetro. Si no lo es, da a inputA el valor del parámetro

- un método llamado cargarB que hace exactamente lo mismo que el anterior pero con inputB

- un método de retorno double y sin parámetros llamado disparar. El funcionamiento es de la siguiente manera: Primero se debe calcular la potencia del disparo. La potencia del disparo es la suma ponderada de inputA e inputB según los pesos pesoA y pesoB. Esto se calcula como $inputA \cdot pesoA + inputB \cdot pesoB$

Si la potencia del disparo no supera el valor que hay en limiteDescarga, entonces el valor del disparo (valor devuelto) es 0.0. En este caso los valores de inputA e inputB no se modifican.

Si la potencia del disparo supera el valor que hay en limiteDescarga la neurona "dispara". El valor del disparo es la potencia que se ha calculado, si esta no supera el valor descargaMaxima. Y es el valor de descargaMaxima si la potencia calculada supera este valor. En este caso (potencia supera el límite) los valores de inputA e inputB pasan a ser 0.0 (la neurona se "relaja") un método sin retorno y sin parámetros llamado relajar. Este método deja los atributos inputA e inputB con el valor 0.0

Problema 25

En este problema se debe hacer uso de la clase Neurona anterior:

Apartado A

Escribe un procedimiento especializado en la creación de una neurona a partir de los datos suministrados por el usuario. Los datos que obtendrá del exterior son:

- un carácter para decidir si la neurona será o no acumuladora (si el carácter es 's' o 'S' entonces la neurona será acumuladora. Si el carácter es cualquier otra, no lo será.
- La descarga máxima

El resultado del procedimiento será una neurona creada en el propio procedimiento. Si es acumuladora o no dependerá de la elección del usuario. Los pesos de las entradas A y B serán 0.75 y 0.25 respectivamente. La descarga máxima será la indicada por el usuario y el límite de descarga será el 60% del valor de la descarga máxima.

Apartado B

Escribe un procedimiento sin retorno llamado cargarUnPoco que dados una Neurona y un número real (double) haga lo siguiente:

- Si la neurona es acumuladora, carga las entradas A y B con una carga igual al 50% del número dado como segundo parámetro
- Si la neurona no es acumuladora, carga las entradas A y B con una carga igual al 25% del límite de descarga de la neurona más el 50% del número dado como segundo parámetro

Apartado C

Escribe un procedimiento de resultado booleano llamado propagarCarga, que dados 3 neuronas llamadas fA, fB y destino haga lo siguiente:

- Cargar la entrada A de la neurona destino con el valor resultante de disparar la neurona fA

- Cargar la entrada B de la neurona destino con el valor resultante de disparar la neurona fB

El procedimiento devuelve true si alguna de las neuronas fA o fB cuando ha disparado ha producido un valor superior a 0.0. El retorno será false si ni fB ni fB han disparado por encima de 0.0

Apartado D

Escribe una función, llamada masPotente que dadas 2 neuronas devuelva aquella que tiene un valor del atributo descargaMaxima más grande (cualquier de las dos si tienen el mismo valor en el atributo)

Apartado E

Escribe un programa que haga lo siguiente:

- Con la ayuda del procedimiento hacerNeurona crear 6 neuronas. Nos referiremos a ellas (nombre de las variables que las referenciaran) como o1, o2, o3, o4, d1 y d2.

- Cargar un poco las neuronas o1, o2, o3, y o4 haciendo uso del procedimiento cargarUnPoco. En todos los casos, el valor del segundo parámetro será 200.0

- La carga de las neuronas o1 y o2 propagarla a la neurona d1

- La carga de las neuronas o3 y o4 propagarla a la neurona d2

- De entre las neuronas d1 y d2 elegir aquella que sea más potente

- Disparar la neurona más potente y mostrar (escribir) el valor de la potencia disparada.

Problema 26

Un monomio es una expresión algebraica de un único termino: $c \cdot x^n$ donde c es el coeficiente y n el exponente. Se quiere codificar una clase para representar monomios de coeficiente real y exponente (grado) natural o cero. Coeficiente y exponente serán los atributos.

El constructor de la clase Monomio recibirá dos parámetros que serán el coeficiente y el exponente.

El coeficiente y el exponente de un monomio se podrán consultar pero no se podrán modificar.

Las operaciones que se podrán hacer con un monomio serán las siguientes:

- evaluarlo en un punto. Dado un número real (a) el método evaluar calculará el valor del monomio en aquel punto, esto es: $c \cdot a^n$. Para calcular potencias, se tiene un método privado que calcula la base elevada al exponente:

```
private double pot (double base, int exponent)
```

- Preguntar si el monomio es una constante. Un monomio es una constante si su exponente es 0.

- Calcular la derivada. El método calcularDerivada, sin parámetros, dará como resultado el monomio derivada. El monomio derivada del monomio $c \cdot x^n$ es $c \cdot n \cdot x^{n-1}$, es decir un nuevo monomio de coeficiente $c \cdot n$ y exponente n-1. Este método no modifica el monomio sino que genera (crea) uno nuevo.

Problema 27

Diseña una interface llamada Ordenar. Que tenga los siguientes métodos:

- ordena
- setDatos que tendrá como parámetro un vector de tipo long,
- invierte
- getDatos que devolverá un vector de tipo long.

Estos serán métodos genéricos, para una posterior implementación en otra clase.

Problema 28

Haz un programa que genere 100 números aleatorios, luego los ordene por orden de menor a mayor y los muestre por pantalla. Para hacerlo deberás utilizar en la clase de tu programa, una clase que llamarás Ordenacion que implementará la interface creada en el ejercicio anterior.