Programación

Act07_Clases

Francisco José García Cutillas | 1FPGS_DAM

Índice

Ejercicio A01

Punto.

- **Crea un programa con una clase llamada Punto que representará un punto de dos dimensiones en un plano. Solo contendrá dos atributos enteros llamadas x e y (coordenadas).**

```
package com.mycompany.garciacutillasfranciscojoseactut07clases;

/**
 *
 * @author Fran
 */
public class Punto {

    private int x;
    private int y;
```

- **Añade a la clase Punto un constructor con parámetros que copie las coordenadas pasadas como argumento a los atributos del objeto.**

```
    public Punto(int _x, int _y) {

        this.x = _x;
        this.y = _y;

    }
```

- **Los atributos de Punto van a ser private y debes generar sus getter y setter correspondientes que serán de tipo public.**

```
    public int getX() {
        return x;
    }

    public void setX(int x) {
        this.x = x;
    }

    public int getY() {
        return y;
    }

    public void setY(int y) {
        this.y = y;
    }
```

- **Añade a la clase Punto los siguientes métodos públicos**
  - **public void imprime() // Imprime por pantalla las coordenadas. Ejemplo: "(7, -5)".**

```
//Este método imprime por pantalla las coordenadas del punto
public void imprime() {

    System.out.println("(" + this.x + "," + this.y + ")");

}
```

  - **public void setXY(int x, int y) // Modifica ambas coordenadas. Es como un setter doble.**

```
//Este método permite modificar las dos coordenadas del punto a la vez
public void setXY(int _x, int _y) {

    this.x = _x;
    this.y = _y;

}
```

  - **public void desplaza(int dx, int dy) // Desplaza el punto la cantidad (dx,dy) indicada. Ejemplo: Si el punto (1,1) se desplaza (2,5) entonces estará en (3,6). Puede desplazarse distancias negativas.**

```
//Este método permite desplazar el punto tanto en x como en y
public void desplaza(int dx, int dy) {

    this.x += dx;
    this.y += dy;

}
```

  - **public int distancia(Punto p) // Calcula y devuelve la distancia entre el propio objeto (this) y otro objeto (Punto p) que se pasa como parámetro.**

```
//Este método devuelve como resultado la distancia entre el punto, y otro punto pasado como parámetro
public double distancia(Punto p) {

    double resultado = Math.sqrt((Math.pow(p.getX() - this.x, b: 2)) + (Math.pow(p.getY() - this.y, b: 2)));

    return resultado;

}
```

- **Necesitamos un método que nos permita crear un objeto Punto con coordenadas aleatorias. Esta funcionalidad no depende de ningún objeto concreto por lo que será estática. Deberá crear un nuevo Punto (utiliza el constructor) con x e y entre -100 y 100, y luego devolverlo (con return).**

  - **public static Punto nuevoPuntoAleatorio()**

```
public class Punto {

    private static final int MINALEATORIO = -100;
    private static final int MAXALEATORIO = 100;
```

```
//Este método genera un punto aleatorio en el rango que se haya introducido en las constantes
//presentes en los atributos de la clase Punto
public static Punto nuevoPuntoAleatorio() {


    int xAleatorio = (int) (Math.random() * ((MAXALEATORIO - MINALEATORIO) + 1)) + MINALEATORIO;
    int yAleatorio = (int) (Math.random() * ((MAXALEATORIO - MINALEATORIO) + 1)) + MINALEATORIO;


    return new Punto ( _x: xAleatorio, _y: yAleatorio);
}
```

- **Crea varios objetos de la clase Punto y aplica los diferentes métodos y comprueba que funcionan adecuadamente.**

```java
package com.mycompany.garciacutillasfranciscojoseactut07clases;

/**
 *
 * @author Fran
 */
public class GarciaCutillasFranciscoJoseActUt07Clases {

    public static void main(String[] args) {

        Punto p1 = new Punto( _x: 2, _y: 3);
        Punto p2 = new Punto ( _x: -7,  _y: 55);
        Punto p3 = new Punto ( _x: 23, _y: -9);
        Punto pAleatorio = Punto.nuevoPuntoAleatorio();

        System.out.println( x: "Coordenadas de p1");
        p1.imprime();

        p2.setXY( _x: 10,  _y: -50);
        System.out.println( x: "Coordenadas de p2");
        p2.imprime();

        p3.desplaza( dx: 2,  dy: 5);
        System.out.println( x: "Coordenadas de p3");
        p3.imprime();

        System.out.println("Distancia: "+ p2.distancia( p: p1));

        System.out.println( x: "Coordenadas punto aleatorio");
        pAleatorio.imprime();



    }
}
```

```
] --- exec-maven-plugin:3.0.0:exec (default-cli) @ GarciaCutillasFranciscoJoseActUt07Clases ---
Coordenadas de p1
(2,3)
Coordenadas de p2
(10,-50)
Coordenadas de p3
(25,-4)
Distancia: 53.600373133029585
Coordenadas punto aleatorio
(4,-10)
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  0.344 s
Finished at: 2023-03-01T20:07:07+01:00
------------------------------------------------------------------------
```

Ejercicio A02

Persona

- **Crea un programa con una clase llamada Persona que representará los datos principales de una persona: dni, nombre, primer apellido, segundo apellido y edad.**

```
package com.mycompany.garciacutillasfranciscojoseactut07clases;

import utilidades.Dni;

/**
 *
 * @author Fran
 */
public class Persona {

    private String dni;
    private String nombre;
    private String primerApellido;
    private String segundoApellido;
    private int edad;
```

- **Añade dos constructores de personas, una en la que recibe todos sus atributos y otro en el que no recibe el segundo apellido.**

```
public Persona(String _dni, String _nombre, String _primerApellido, String _segundoApellido, int _edad) {

    //Con el método compruebaDni de la clase Dni en el paquete utilidades, comprobamos que el dni
    //introducido sea correcto
    if (Dni.compruebaDni( dni: _dni)) {

        this.dni = _dni;

    } else {

        System.out.println( x: "El DNI introducido no es correcto");

    }

    this.nombre = _nombre;
    this.primerApellido = _primerApellido;
    this.segundoApellido = _segundoApellido;

    //Si la edad introducida es negativa, se asigna 0 a edad
    if (_edad < 0) {

        this.edad = 0;

    } else {

        this.edad = _edad;

    }


}

public Persona(String _dni, String _nombre, String _primerApellido, int _edad){

    this(_dni, _nombre, _primerApellido, _segundoApellido: " ", _edad);

}
```

```java
package utilidades;

/**
 *
 * @author Fran
 */
public class Dni {


    //Método para comprobar que introducen el dni correcto
    //Devuelve true si el dni tiene los 8 primeros caracteres numéricos y el último es una letra,
    //en cualquier otro caso devuelve false
    public static boolean compruebaDni(String dni) {

        char dniChar[] = new char[dni.length()];
        dniChar = dni.toCharArray();
        boolean salida = true;

        if (dni.length() != 9) {

            salida = false;

        } else {

            for (int i = 0; i < dniChar.length; i++) {

                if (i < 8 && !Character.isDigit(dniChar[i])) {

                    salida = false;

                }

                if (i == 8 && !Character.isLetter(dniChar[i])) {

                    salida = false;

                }

            }

        }

        return salida;

    }

}
```

- **Todos los atributos serán private y existirán getters y setters para cada atributo. Además, se definirá un getApellidos en el que devolverá ambos apellidos en una sola cadena.**

```java
public String getDni() {
    return dni;
}

public void setDni(String dni) {
    this.dni = dni;
}

public String getNombre() {
    return nombre;
}

public void setNombre(String nombre) {
    this.nombre = nombre;
}

public String getPrimerApellido() {
    return primerApellido;
}

public void setPrimerApellido(String primerApellido) {
    this.primerApellido = primerApellido;
}

public String getSegundoApellido() {
    return segundoApellido;
}

public void setSegundoApellido(String segundoApellido) {
    this.segundoApellido = segundoApellido;
}

public int getEdad() {
    return edad;
}

public void setEdad(int edad) {
    this.edad = edad;
}

public String getApellidos(){

    return this.primerApellido+" "+this.segundoApellido;

}
```

- **Añade a la clase Persona los siguientes métodos públicos:**
    - **public void imprime() // Imprime la información del objeto: "DNI:… Nombre:… etc."**

```java
//Método que muestra por pantalla la información de la persona
public void imprime(){

    System.out.println( x: "*** INFORMACIÓN DE LA PERSONA ***");
    System.out.println("DNI: " + this.dni);
    System.out.println("Nombre: " + this.nombre);
    System.out.println("Primer apellido: " + this.primerApellido);
    System.out.println("Segundo apellido: " + this.segundoApellido);
    System.out.println("Edad: " + this.edad);

}
```

```java
//Método que muestra por pantalla la información de la persona
public void imprime() {

    System.out.println( x: "*** INFORMACIÓN DE LA PERSONA ***");
    System.out.println("DNI: " + this.dni);
    System.out.println("Nombre: " + this.nombre);
    System.out.println("Apellidos: "+ getApellidos());
    System.out.println("Edad: " + this.edad);

}
```

En este caso hemos cambiado la forma de mostrar los apellidos, para que en el caso de que no se le introduzca segundo apellido, no salga un campo en blanco.

    - **public boolean esMayorEdad() // Devuelve true si es mayor de edad (false si no).**

```java
//Método que devuelve true si es mayor de edad y false si no lo es
public boolean esMayorEdad() {

    if (this.edad >= 18) {

        return true;

    } else {

        return false;

    }

}
```

- **public boolean esJubilado() // Devuelve true si tiene 65 años o más (false si no).**

```
//Método que devuelve true si tiene más de 65 años y false si no
public boolean esJubilado(){

    if (this.edad >= 65) {

        return true;

    } else {

        return false;

    }

}
```

- **public int diferenciaEdad(Persona p) // Devuelve la diferencia de edad entre 'this' y p.**

```
//Método que devuelve la diferencia de edad de dos personas
public int diferenciaEdad(Persona p){

    return Math.abs(this.edad - p.getEdad());

}
```

- **public boolean esMayor(Persona p) // Devuelve true si 'this" tiene la misma o más edad que p**

```
//Método que devuelve true si la persona tiene la misma edad que otra persona pasada por parámetro
public boolean esMayor(Persona p){

    if (this.edad >= p.getEdad()) {

        return true;

    } else {

        return false;

    }

}
```

- **El DNI de una persona no puede variar. Añade el modificador final al atributo dni y asegúrate de que se guarde su valor en el constructor.**

  - **Quita el método setDNI(…) que de todos modos ya no se podrá utilizar porque Java no te dejará modificar el atributo dni.**

```
public class Persona {

    private final String dni;
    private String nombre;
```

```java
public Persona(String _dni, String _nombre, String _primerApellido, String _segundoApellido, int _edad) {

    //Con el método compruebaDni de la clase Dni en el paquete utilidades, comprobamos que el dni
    //introducido sea correcto
    if (Dni.compruebaDni( dni: _dni)) {

        this.dni = _dni;

    } else {

        System.out.println( x: "El DNI introducido no es correcto");
        this.dni = null;

    }
```

- **La mayoría de edad a los 18 años es un valor común a todas las personas y no puede variar. Crea un nuevo atributo llamado mayoriaEdad que sea static y final. Tendrás que inicializarlo a 18 en la declaración. Utilízalo en el método que comprueba si una persona es mayor de edad.**

```java
public class Persona {

    private final String dni;
    private final static int mayoriaEdad = 18;
```

```java
    //Método que devuelve true si es mayor de edad y false si no lo es
    public boolean esMayorEdad() {

        if (this.edad >= mayoriaEdad) {

            return true;

        } else {

            return false;

        }

    }
```

- **Crea un método static boolean validarDNI(String dni) que devuelva true si dni es válido (tiene 8 números y una letra). Si no, devolverá false. Utilízalo en el constructor para comprobar el dni (si no es válido, muestra un mensaje de error y no guardes los valores).**

  Ya está creado anteriormente en el paquete utilidades, clase Dni, método compruebaDni.

- **Crea varios objetos de la clase Persona y aplica los diferentes métodos y comprueba que funcionan adecuadamente.**

```
//Ejercicio 2
Persona per1 = new Persona ( _dni: "11111111A",  _nombre: "Juan",  _primerApellido: "Cano",  _segundoApellido: "Fernández",  _edad: 30);
Persona per2 = new Persona ( _dni: "22222222B",  _nombre: "Alicia",  _primerApellido: "Pérez",  _edad: 17);
Persona per3 = new Persona ( _dni: "33333333C",  _nombre: "Pepe",  _primerApellido: "Cantero",  _segundoApellido: "Palazón",  _edad: 66);

per1.imprime();
per2.imprime();
System.out.println(per2.getNombre() + " es mayor de edad: "+per2.esMayorEdad());
System.out.println(per3.getNombre()+" es jubilado: "+per3.esJubilado());
System.out.println("Diferencia de edad entre "+ per1.getNombre()+" y "+ per2.getNombre()+" es: "+ per2.diferenciaEdad( p: per1));
System.out.println("¿"+ per2.getNombre()+ " es mayor que "+ per3.getNombre()+ "? "+per2.esMayor( p: per3));
per1.setNombre( nombre: "Antonio");
per1.setPrimerApellido( primerApellido: "Martínez");
per1.setSegundoApellido( segundoApellido: "Guirao");
per1.setEdad( edad: 55);
per1.imprime();
```

```
*** INFORMACIÓN DE LA PERSONA ***
DNI: 11111111A
Nombre: Juan
Apellidos: Cano Fernández
Edad: 30
*** INFORMACIÓN DE LA PERSONA ***
DNI: 22222222B
Nombre: Alicia
Apellidos: Pérez
Edad: 17
Alicia es mayor de edad: false
Pepe es jubilado: true
Diferencia de edad entre Juan y Alicia es: 13
¿Alicia es mayor que Pepe? false
*** INFORMACIÓN DE LA PERSONA ***
DNI: 11111111A
Nombre: Antonio
Apellidos: Martínez Guirao
Edad: 55
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  0.352 s
Finished at: 2023-03-02T16:38:26+01:00
```

Ejercicio A03

Artículo

- **Crea un programa con una clase llamada Artículo que contará con los siguientes atributos: nombre, precio (sin iva) e iva (será un dato entero)**

```
package com.mycompany.garciacutillasfranciscojoseactut07clases;

/**
 *
 * @author Fran
 */
public class Articulo {

    private String nombre;
    private double precioSinIva;
    private int iva;

}
```

- **Añade un constructor con todos los datos que consideres relevantes.**

```
public Articulo(String _nombre, double _precioSinIva, int _iva){

    this.nombre = _nombre;

    if (_precioSinIva < 0) {

        System.out.println( x: "Error. No se puede introducir un precio negativo a un articulo");
        this.precioSinIva = 0;

    } else {

        this.precioSinIva = _precioSinIva;

    }

    this.iva = _iva;

}
```

- **Encapsulamiento básico: Todos los atributos serán private y debes indicar los getters y setters necesarios.**

```
public String getNombre() {
    return nombre;
}

public void setNombre(String nombre) {
    this.nombre = nombre;
}

public double getPrecioSinIva() {
    return precioSinIva;
}

public void setPrecioSinIva(double precioSinIva) {
    this.precioSinIva = precioSinIva;
}

public int getIva() {
    return iva;
}

public void setIva(int iva) {
    this.iva = iva;
}
```

- **Diseña los siguientes métodos:**

  - **Método para imprimir la información del artículo por pantalla.**

```
public void muestraInformacion(){

    System.out.println( x: "*** INFORMACIÓN DEL ARTÍCULO ***");
    System.out.println("Nombre artículo: " + this.nombre);
    System.out.println("Precio sin iva: " + this.precioSinIva);
    System.out.println("Iva aplicado al producto: " + this.iva);

}
```

  - **Método getPVP que devuelva el precio de venta al público (PVP) con iva incluido.**

```
public double getPVP(){

    return this.precioSinIva * (((double)this.iva / 100) + 1);

}
```

- **Método getPVPDescuento que devuelva el PVP con un descuento pasado como argumento.**

```
public double getPVPDescuento(int descuento){

    return getPVP() - (getPVP() * ((double)descuento / 100)) ;

}
```

- **Tipo de IVA. Vamos a indicar que existen tres tipos de IVA**
  - **El IVA general (21%): para la mayoría de los productos a la venta.**
  - **El IVA reducido (10%): hostelería, transporte, vivienda, etc.**
  - **El IVA super reducido (4%): alimentos básicos, libros, medicamentos, etc.**
- **Estos tres tipos de IVA no pueden variar y a cada artículo se le aplicará uno de los tres. Razona qué cambios sería necesario realizar a la clase Articulo e impleméntalos.**

```
package com.mycompany.garciacutillasfranciscojoseactut07clases;

/**
 *
 * @author Fran
 */
public class Articulo {

    enum TipoIva {

        GENERAL, REDUCIDO, SUPERREDUCIDO;

    }

    private static final int IVAGENERAL = 21;
    private static final int IVAREDUCIDO = 10;
    private static final int IVASUPERREDUCIDO = 4;
```

Primero hemos creado un enumerado con los tres tipos de IVA, además de sus correspondientes atributos constantes. Con esto, podríamos cambiar dicha constante para toda la clase, en el caso de que variara el porcentaje de cada IVA.

```
public Articulo(String _nombre, double _precioSinIva, TipoIva _iva) {

    this.nombre = _nombre;

    //Comprobación de que no introduzcan precio en negativo
    if (_precioSinIva < 0) {

        System.out.println( x: "Error. No se puede introducir un precio negativo a un artículo");
        this.precioSinIva = 0;

    } else {

        this.precioSinIva = _precioSinIva;

    }

    if (_iva.equals( other: TipoIva.GENERAL)) {

        this.iva = IVAGENERAL;

    } else if (_iva.equals( other: TipoIva.REDUCIDO)) {

        this.iva = IVAREDUCIDO;

    } else {

        this.iva = IVASUPERREDUCIDO;

    }

}
```

El constructor quedaría de esta manera.

- **Crea varios objetos de la clase Artículo y aplica los diferentes métodos y comprueba que funcionan adecuadamente.**

```
//Ejercicio 3
Articulo a1 = new Articulo( _nombre: "Edredón",  _precioSinIva: 24.99,  _iva: Articulo.TipoIva.GENERAL);
Articulo a2 = new Articulo( _nombre: "Pan",  _precioSinIva: 0.90,  _iva: Articulo.TipoIva.SUPERREDUCIDO);
Articulo a3 = new Articulo( _nombre: "Pepsi",  _precioSinIva: 0.50,  _iva: Articulo.TipoIva.REDUCIDO);

a1.muestraInformacion();
System.out.println("Precio con iva de "+ a1.getNombre() + " :" + a1.getPVP());
System.out.println("Precio con descuento de "+ a1.getNombre() + " :" + a1.getPVPDescuento( descuento: 10));

System.out.println("Cambio nombre y precio del " + a2.getNombre());
a2.setNombre( nombre: "Huevos");
a2.setPrecioSinIva( precioSinIva: 2.15);
a2.muestraInformacion();

System.out.println("Precio sin iva de " + a3.getNombre() + ": " + a3.getPrecioSinIva());
```

```
*** INFORMACIÓN DEL ARTÍCULO ***
Nombre artículo: Edredón
Precio sin iva: 24.99€
Iva aplicado al producto: 21%
Precio con iva de Edredón :30.237899999999996
Precio con descuento de Edredón :27.214109999999998
Cambio nombre y precio del Pan
*** INFORMACIÓN DEL ARTÍCULO ***
Nombre artículo: Huevos
Precio sin iva: 2.15€
Iva aplicado al producto: 4%
Precio sin iva de Pepsi: 0.5
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  0.580 s
Finished at: 2023-03-03T12:46:07+01:00
------------------------------------------------------------------------
```

Ejercicio A04

Compra

- **Crea un programa con una clase llamada Compra que va a tener una persona, un array de artículos que se ha comprado, un array de enteros que indica las cantidades que se han comprado de dicho artículo (Por ejemplo, si en la posición 2 del array artículos existe un artículo llamado "Libro el temor de un hombre Sabio" y en esa posición 2 del array númeroArticulosComp hay un 3 de valor, significa que sean comprado 3 ejemplares de dicho libro) y una fecha en la que se ha realizado la compra (LocalDate).**

```java
package com.mycompany.garciacutillasfranciscojoseactut07clases;

import java.time.LocalDate;

/**
 *
 * @author Fran
 */
public class Compra {

    private Persona persona;
    private Articulo articulo[];
    private int cantidad[];
    private LocalDate fecha;
```

- **Debes indicar un constructor en la que se deben recibir todos los datos de la compra (los arrays ya deben llegar rellenos).**

```
public Compra(Persona _persona, Articulo _articulo[], int _cantidad[], LocalDate _fecha){

    this.persona = _persona;
    this.articulo = _articulo;
    this.cantidad = _cantidad;
    this.fecha = _fecha;

}
```

- **Encapsulamiento básico: Atributos privados y métodos getters y setters.**

```
public Persona getPersona() {
    return persona;
}

public void setPersona(Persona persona) {
    this.persona = persona;
}

public Articulo[] getArticulo() {
    return articulo;
}

public void setArticulo(Articulo[] articulo) {
    this.articulo = articulo;
}

public int[] getCantidad() {
    return cantidad;
}

public void setCantidad(int[] cantidad) {
    this.cantidad = cantidad;
}

public LocalDate getFecha() {
    return fecha;
}

public void setFecha(LocalDate fecha) {
    this.fecha = fecha;
}
```

- **Diseña los siguientes métodos**
    - **Mostrar información de la compra**

```java
//Método para mostrar la información de la compra
public void mostrarInfo(){

    System.out.println( x: "*** INFORMACIÓN DE LA COMPRA ***");
    System.out.println( x: "Articulo       Cantidad       PVP");
    System.out.println( x: "--------------------------------");

    for (int i = 0; i < this.articulo.length; i++) {

        System.out.println(articulo[i].getNombre()+ "                "
                + cantidad[i] + "      " + articulo[i].getPVP());

    }

    System.out.println( x: " ");
    System.out.println("Total de la compra          " + precioTotal() + "€");

    System.out.println( x: "--------------------------------");
    System.out.println("Fecha de compra: " + this.fecha);

}
```

    - **Método que devuelve el precio total de la compra (debe tener en cuenta el precio de los productos con su IVA y la cantidad comprada)**

```java
//Método para calcular el precio total de los productos de la compra
public double precioTotal(){

    double total = 0;

    for (int i = 0; i < this.articulo.length; i++) {

        total += this.cantidad[i] * this.articulo[i].getPVP();

    }

    return total;

}
```

```java
//Ejercicio 4
Articulo arrayAr[] = {a1, a2, a3};
int cantidades[] = {1, 2, 5};

Compra c1 = new Compra( _persona: per1, _articulo: arrayAr, _cantidad: cantidades, _fecha: LocalDate.now());
c1.mostrarInfo();
```

```
*** INFORMACIÓN DE LA COMPRA ***
Artículo       Cantidad       PVP
---------------------------------
Colcha                1     30.237899999999996
Huevos                2     2.2359999999999998
Pepsi                 5     0.55

Total de la compra          37.4599€
---------------------------------
Fecha de compra: 2023-03-03
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  0.325 s
```