EJERCICIOS TEMA 5

1.

Clase Caja

Diseñar la clase Caja
Propiedades:
Alto, largo y profundo de tipo entero
Métodos:
volumen, t.q. devuelva el valor del volumen de la caja.
duplicaTamaño, t.q. duplique el valor del alto, largo y profundo de la caja actual.
ampliaTamaño, t.q. dada una cantidad entera, amplía el valor de largo, alto y profundo d e la caja actual con ese valor.
toString, que devuelve una cadena cuyo contenido sea el valor del alto, largo y profundo de la caja actual.
setCaja que cambie el valor del alto, largo y profundo de la caja actual, por tres valores de tipo entero como datos de entrada.

2.

Clase Triángulo

Diseña una clase que se llame triángulo, cuyas propiedades son la base y la altura de tipo entero.
Métodos:
area, devuelve el valor del área del triángulo actual.
perimetro, devuelve el perímetro del triángulo actual.
hipotenusa, devuelve la hipotenusa del triángulo actual.
duplicaTamaño, duplica el valor de la base y la altura del triángulo actual.
ampliaTamaño, dada una cantidad entera, amplía el valor de la base y la altura del triángulo actual con esa cantidad.
toString, devuelve una cadena que contiene los valores del triángulo actual (con modificador público).
equals, t.q. dado otro triángulo, devuelve true, si las propiedades del triángulo actual coinciden con el de entrada, o false, si no lo hacen. De carácter público.
setTriangulo, t.q. dada una base y una altura, se cambian los valores actuales por los nuevos.
cambioTriangulo, t.q. dado un triángulo, se cambien los valores de la base y altura actuales por los valores de la base y altura del triángulo dado.

En main():
Crea dos triángulos: t1 y t2.
La base y la altura de t1son 12 y 5, respectivamente.
Mostrar los valores de ese triángulo t1.
La base y la altura de t2son 10 y 4, respectivamente.
Mostrar los valores de ese triángulo t2.
Duplicar el tamaño de t1.
Mostrar los valores de ese triángulo t1.
Ampliar el tamaño de t2 en 7 unidades.
Mostrar los valores de ese triángulo t1.
Calcula el área, hipotenusa y perímetro del triángulo t1.
Muestra los valores del área, hipotenusa y perímetro del triángulo t1.
Muestra los valores del área, hipotenusa y perímetro del triángulo t2.
Cambia los valores de la base y la altura de t1 por los valores 50 y 30.
Mostrar los valores del triángulo t1.
Cambia los valores de t2 por los de t1.

Mostrar los valores del triángulo t2.
Comprobar si t1 y t2 son iguales.

Un objeto es el encapsulamiento de un conjuntos de operaciones (métodos) y de un estado (datos), con lo cual, la encapsulación consiste en agrupar atributos (propiedades) y métodos en una clase.

Con dicho encapsulamiento se consigue que los programadores que vayan a utilizar dicha clase, solo tengan que conocer cómo llamar a los métodos, sin tener que saber cómo están implementadas.

La encapsulación está asociada con la visibilidad, determinando con ellos qué miembros de la clase son visibles fuera de la clase e, incluso, fuera del paquete donde está dicha clase. Lo aconsejable es que los datos sean privados y que se puedan acceder a ellos a través de los métodos de la propia clase.

3.

Crear el proyecto: 'IES', que esté formado por dos paquetes: paquete1 y paquete2.

Dentro del **paquete1**, vamos a meter dos ficheros java: uno que guardará la clase Alumno y el otro, guardará la clase Profesor.

Clase Profesor PUBLICA

Propiedades: nombre y especialidad de tipo String **privadas**.

Métodos:

- pedirNombre, que introduzca por teclado el nombre del profesor PACKAGE.
- pedirEspecialidad, que introduzca por teclado la especialidad PACKAGE.
- toString, devuelva una cadena con los valores del profesor PUBLIC.

Clase Alumno PUBLICA

Propiedades: nombre de tipo String y edad de tipo entero **publicas.**

Métodos:

- pedirNombre, que introduzca por teclado el nombre del alumno PRIVADO.
- pedirEdad, que introduzca por teclado la edad PRIVADO.
- iniciarDatos, dados como datos de entrada un nombre y una edad, el alumno tomará ese nombre y esa edad. PACKAGE
- toString, devuelva una cadena con los valores del alumno. PUBLIC

Dentro del **paquete2** se va a incluir un fichero con la clase Principal. Exactamente, dicho fichero tendrá el siguiente código:

-importar la clase Alumno del paquete1 (import paquete1.Alumno)

-importar la clase Profesor del paquete1 (import paquete1.Profesor)

-En el método main de la clase Principal:

- Crea un alumno alum1.
- Crea un profesor prof1.

-Escribe en el método main: alum1. saldrán todos los miembros que puedes usar de Alumno: las propiedades: nombre y edad, y el métodos: toString –el único public-.

-Dentro del método iniciarDatos de la clase Alumno, añade la línea de código: Profesor prof1=new Profesor(); . Después escribe prof1. no saldrán las propiedades pero sí los métodos: toString, pedirNombre y pedirEspecialidad, toString por ser público y el resto por ser PACKAGE.

-Dentro de la clase Profesor, en cualquier método escribe **this.** saldrán todos los miembros de la clase Profesor, sean propiedades como métodos. Al igual ocurre con la clase Alumno.

-Modifica las condiciones de visibilidad de las clases Alumno y Profesor. Las nuevas condiciones son:

La clase Alumno se puede usar fuera del paquete, pero la clase Profesor solo se puede usar dentro del paquete al que pertenece.

Dentro de la clase Alumno:

-El método iniciarDatos() se puede usar fuera de la clase, pero no fuera del paquete.

La POO es una colección de clases. Siempre habrá una clase que tendrá el método main, que es por donde se empezará a arrancar la ejecución del programa.

---

4. Clase Punto

Propiedades: Las coordenadas x e y, de tipo entero y privadas.

Métodos (utilizad this donde creáis conveniente):

Constructor Punto, t.q. Inicialice x e y a -1.

getX, para obtener la coordenada x del punto actual.

getY, para obtener la coordenada y del punto actual.

setX, para cambiar el valor de x.

setY, para cambiar el valor de y.

toString, devuelve una cadena con los valores de las coordenadas del pto.

distancia, t.q. devuelva la distancia existente entre el pto. actual y otro dado.

ptoIntermedio, t.q. devuelve el punto situado entre dos puntos, el actual y otro dado.

ptoSuma, t.q. devuelve el pto. resultado de la suma del pto. actual más otro dado.

setPunto, t.q. cambie el valor de x e y, con dos valores dados.

inicializa, t.q. dado dos cadenas, las transforma a enteros, e inicializa así las coordenadas.

ptoOpuesto, t.q. devuelva un punto con las coordenadas opuestas del punto actual con respecto del origen.

equals, tal que dado otro punto, devuelve true, si las coordenadas son las mismas que el punto actual, o false, si no lo son. De carácter público.

Métodos sobrecargados:

Constructor **Punto**, t.q. inicialice x e y con dos valores de tipo entero.

Constructor **Punto**, t.q. inicialice x e y con las coordenadas de otro punto.

setPunto, t.q. cambie el valor de x e y, con los valores de un pto. dado.

distancia, t.q. dados dos valores enteros correspondientes a las coordenadas x e y , devuelva el valor de la distancia entre el punto actual y punto formado por esos valores.

distancia, devuelve la distancia existente entre el punto actual y el origen.

equals, tal que dadas dos coordenadas de tipo entero que forman un punto, devuelve true, si coinciden con las del punto actual, o false, si no lo hacen.

ptoSuma, t.q. dando dos coordenadas enteras, devuelva el punto suma del punto actual y el punto cuyas coordenadas son las dadas.

ptoIntermedio, t.q. dando dos coordenadas enteras, devuelve el punto intermedio con respecto del punto actual y el formado por las coordenadas dadas.

USO DE LA CLASE: Punto

a) Crear los puntos: p, q, r, t.
b) Las coordenadas de p son: 10 y 3
c) Las coordenadas de q son por defecto.
d) Las coordenadas de r coinciden con las de p.
e) Las coordenadas de t son introducidas por teclado y las cadenas leídas se consideran como entrada en el constructor.
f) Mostrar los valores de las coordenadas de los 4 ptos.
g) Calcula la distancia existente entre el punto q y el punto origen.

h) Calcula la distancia existente entre el punto r y el punto formado por las coordenadas por los valores 8 y 7.
i) Calcula la distancia existente entre el punto q y el punto r.
j) Cambia las coordenadas del punto q por los valores de la coordenada x del punto p y la coordenada y del punto r.
k) Cambia las coordenadas del punto t por los valores de las coordenadas del punto p.
l) Crea un punto o, cuyos valores sean por defecto.
El punto o será el punto opuesto al punto t.
Mostrar las coordenadas del punto o.
m) Crea un punto s, cuyos valores sean por defecto.
El punto s será el punto suma resultante de la suma entre el punto t y q.
Mostrar las coordenadas del punto s.
n) Crea un punto i, cuyos valores sean por defecto.
El punto i será el punto intermedio entre p y t.
Mostrar las coordenadas del punto i.
o) Crea un punto z, cuyos valores sean por defecto.
El punto z será el opuesto del punto intermedio entre los puntos q y t.
Mostrar las coordenadas del punto z.
p) Comprobar si el punto q y el punto t son iguales.

5.

Clase Complejo

Diseña la clase: **Complejo**.
Propiedades de carácter privado y de tipo entero: **r** e **i**, siendo r la parte real e i la imaginaria.

Constructores:

Complejo, t.q. inicialice el complejo a la unidad real (1,0).
Complejo, t.q. inicialice el complejo a dos valores considerados como datos de entrada.
Complejo, t.q. inicialice el complejo a los valores de otro complejo considerado como dato de entrada.

Métodos de acceso a las propiedades:

getReal, t.q. devuelve la parte real del complejo actual.
getImaginaria, t.q. devuelve la parte imaginaria del complejo actual.
setReal, t.q. cambia el valor de la parte real del complejo actual por un valor considerado como entrada.
setImaginaria, t.q. cambia el valor de la parte imaginaria del complejo actual por un valor considerado como entrada.
setComplejo, t.q. dados dos valores enteros, cambie la parte real y la imaginaria del complejo actual por ellos.
setComplejo, t.q. dado un complejo, cambie la parte real y la imaginaria del complejo actual por las respectivas del primero.

Métodos:

toString, t.q. devuelva una cadena con el valor de las propiedades del complejo actual, siendo de carácter público.
equals, t.q. dado un complejo, devuelva true, si son iguales o false si no lo son. De carácter público.
equals, t.q. dados dos valores de tipo entero, devuelve true si los valores del complejo actual coinciden o false si no lo hacen. De carácter público.
complejoSuma, t.q. devuelva el complejo resultado de la suma del complejo actual y otro

considerado como entrada. Fórmula: (a + bi) + (c + di) = (a+c) + (b+d)i

complejoSuma, t.q. devuelva el complejo resultado de la suma del complejo actual y los valores considerados como entrada, correspondientes a la parte real y la imaginaria.

complejoResta, t.q. devuelva el complejo resultado de la resta del complejo actual y los valores considerados como entrada, correspondientes a la parte real y la imaginaria.
Fórmula: (a + bi) – (c + di) = (a-c) + (b-d)i

complejoResta, t.q. devuelva el complejo resultado de la resta del complejo actual y otro considerado como entrada.

complejoProducto, t.q. devuelva el complejo resultado de la prodcuto del complejo actual y otro considerado como entrada. Fórmula: (a + bi) x (c + di) = (ac – bd) + (ad + bc)i

complejoProducto, t.q. devuelva el complejo resultado de la producto del complejo actual y los valores considerados como entrada, correspondientes a la parte real y la imaginaria.

complejoCociente, t.q. devuelva el complejo resultado del cociente del complejo actual y otro considerado como entrada.
Fórmula: $(a + bi) / (c + di) = ((ac + bd) / (c^2 + d^2)) + ((bc – ad)/(c^2 + d^2))\ i$

complejoCociente, t.q. devuelva el complejo resultado del cociente del complejo actual y los valores considerados como entrada, correspondientes a la parte real y la imaginaria.

USO DE LA CLASE: COMPLEJO

a) Crear los complejos: w, x, y, z de la siguiente forma:
b) Los valores de las propiedades de w son: 12 y 3, parte real e imaginaria, respectivamente.
c) Los valores de las propiedades de x son por defecto.
d) Los valores de las propiedades de y coinciden con las de w.
e) Los valores de las propiedades de z son introducidas por teclado y las cadenas leídas se consideran como entrada en el constructor.
f) Mostrar los valores de las propiedades de los 4 complejos.
Cambia los valores de las propiedades del complejo z por los valores de la parte imaginaria del complejo y y la parte real del complejo w.
Cambia las coordenadas del complejo w t por los valores de las propiedades del complejo y.
g) Crea un complejo v, cuyos valores sean por defecto.
El punto v será el complejo resultante de la suma de los complejos w y z.
Mostrar las propiedades del complejo v.
h) Crea un complejo u resultante de la suma entre el complejo w y el formado por los valores 8 y 5, (parte real e imaginaria, respectivamente).
Mostrar las propiedades del complejo u.
i) Mostrar las propiedades del complejo resultante de la suma de los complejos z y u.
j) Crea un complejo t, cuyos valores sean por defecto.
Calcular t= w-(y+(8,5i))
Mostrar las propiedades del completo t.
k) Crea un complejo s, con los valores de sus propiedades sean las de un complejo generado de la fórmula: w (t + y) .
Mostrar las propiedades del complejo s.
l) Crea un complejo r, con los valores de sus propiedades sean: la real -> la parte imaginaria del complejo cociente de t entre y, la imaginaria -> 10.
Mostrar las propiedades del complejo r.
m) Comprobar si los complejos r y t son iguales.
n) Comprobar si los complejos w y el formado por los valores 5+6i son iguales.

6.

Clase Fraccion

Diseña la clase: **Fraccion**.
Propiedades de carácter privado y de tipo entero: **num** y **den**, siendo **num** el numerador y **den** el denominador.

Constructores:

Fraccion, t.q. inicialice la fraccion a la unidad (1,1).
Fraccion, t.q. inicialice la fraccion a dos valores considerados como datos de entrada.
Fraccion, t.q. inicialice la fraccion a los valores de otra fraccion considerado como dato de entrada.

Métodos de acceso a las propiedades:

getDen, t.q. devuelve el denominador de la fraccion actual.
getNum, t.q. devuelve el numerador de la fraccion actual.
setDen, t.q. cambia el valor del denominador de la fraccion actual por un valor considerado como entrada.
setNum, t.q. cambia el valor del numerador de la fraccion actual por un valor considerado como entrada.
setFraccion, t.q. dados dos valores enteros, cambie el numerador y el denominador de la fraccion actual por ellos.
setFraccion, t.q. dado una fraccion, cambie el numerador y el denominador de la fraccion actual por las respectivas de la primera.

Métodos:

toString, t.q. devuelva una cadena con el valor de las propiedades de la fracción actual, siendo de carácter público.
equals, t.q. dada otra fracción, devuelve true si son iguales su denominador como su numerador, o false si no lo son. De carácter público.
equals, t.q. dados dos valores enteros considerados como numerador y numerador, devolviendo true si coinciden con los de la fracción actual, o false si no lo hacen. De carácter público.
fraccionSuma, t.q. devuelva la fracción resultado de la suma de la fracción actual y otra considerado como entrada. Fórmula: (a/b) + (c/d) = (a*d + b*c) /(b*d)
fraccionSuma, t.q. devuelva la fracción resultado de la suma de la fracción actual y los valores
considerados como entrada, correspondientes al numerador y denominador.
fraccionResta, t.q. devuelva la fraccion resultado de la resta de la fraccion actual y los valores considerados como entrada, correspondientes al numerador y el denominador.
Fórmula: (a/b) + (c/d) = (a*d - b*c) /(b*d)

fraccionResta, t.q. devuelva la fraccion resultado de la resta de la fracción actual y otra considerada como entrada.
fraccionProducto, t.q. devuelva la fracción resultado de la prodcuto de la fracción actual y otra considerada como entrada. Fórmula: (a/b) * (c/d) = (a*c) / (b*d)
fraccionProducto, t.q. devuelva la fracción resultado de la producto de la fraccción actual y los valores considerados como entrada, correspondientes al numerador y denominador.
fraccionCociente, t.q. devuelva la fraccción resultado del cociente de la fracción actual y otra considerada como entrada. Fórmula: (a/b) / (c/d) = (a*d) / (b*c)
fraccionCociente, t.q. devuelva la fracción resultado del cociente de la fracción actual y los valores considerados como entrada, correspondientes al numerador y denominador.

reducir, de carácter privado, tal que reduzca la fracción actual hasta hacerla irreducible.
mcd, tal que dados dos números enteros, x e y, calcula y devuelve el valor del máximo común divisor de ambos. De carácter privado y estático.
mcd, tal que dada una fracción, calcula y devuelve el valor del máximo común divisor de ambos. De carácter privado y estático.

USO DE LA CLASE: FRACCION

a) Crear los quebrados: f, g, h, j de la siguiente forma:
b) Los valores de la fracción f son: 27 y 5, numerador y denominador, respectivamente.
c) Los valores de la fracción g son por defecto.
d) Los valores de la fracción h coinciden con las de g.
e) Los valores de las propiedades de j son introducidas por teclado y las cadenas leídas se consideran como entrada en el constructor.
f) Mostrar los valores de las 4 fracciones.

Reducir la fracción j.
Cambia los valores de la fracción f por los valores del denominador de g y del numerador de h.
Cambia los valores de la fracción g por los valores de la fracción h.

g) Crea un quebrado k, cuyos valores sean por defecto.

El quebrado k será el quebrado resultante de la resta de los quebrados g y f.
Mostrar las propiedades de la fracción k.

h) Crea un quebrado l resultante del cociente entre el quebrado h y el formado por los valores 18 y 5.

Mostrar las propiedades de la fracción l.

i) Mostrar las propiedades de la fracción resultante del producto de los quebrados h y k.
j) Crea un quebrado m, cuyos valores sean por defecto.

Calcular m= j-(l+(23,8))
Mostrar las propiedades del quebrado m.

k) Crea un quebrado n, con los valores del quebrado generado de la fórmula: g /(h - f) .

Mostrar las propiedades de la fracción n.

l) Crea un quebrado o, con los valores del quebrado son: el numerador -> el denominador de la fracción suma de f y h, y denominador -> el denominador de la fracción resta de g y m.

Mostrar las propiedades de la fracción o.

7.

Clase Hora

Diseñar la clase: **Hora**.
Propiedades:

h, **m** y **s**. Donde h es la hora, m los minutos y s los segundos. De tipo entero y carácter privado.

Constructores:

Se pide implementarlos con su propio **código** y tb. utilizando **this**. Indicar qué constructor o constructores se eligen como patrón.

Hora, t.q. inicialice las propiedades 0. Constructor por defecto.
Hora, t.q. inicialice las tres propiedades a tres valores de tipo entero considerados como datos de entrada.
Hora, t.q. inicialice la hora a un sólo valor entero considerado como dato de entrada, y los minutos y segundos a 0.

Hora, t.q. inicialice la hora y los minutos a dos valores enteros considerados como datos de entrada y los segundos a 0.
Hora, t.q. inicialice las propiedades a las de otra hora.

Métodos de acceso a las propiedades:

getH, t.q. devuelve el valor de h.
getM, t.q. devuelve el valor de m.
getS, t.q. devuelve el valor de s.
setH, t.q. cambia el valor de h por otro de tipo entero considerado como entrada.
setM, t.q. cambia el valor de m por otro de tipo entero considerado como entrada.
setS, t.q. cambia el valor de s por otro de tipo entero considerado como entrada.
setHora, t.q. cambia los valores de todas las propiedades de la hora actual por tres valores considerados como entrada.
setHora, t.q. cambia los valores de las propiedades de la hora actual por las de otra hora.
toString, t.q. devuelve una cadena con los valores de las propiedades. El formato ha de ser el siguiente: hh:mm:ss. El método es de carácter público.
equals, t.q. dada otra hora, devuelve true, si son iguales o false, si no lo son. De carácter público.
equals, t.q. dados tres datos de entrada considerados como entrada, representando a la hora, minutos y segundos, devuelve true, si los valores coinciden o false si no lo hacen. Método de carácter público.

Métodos:

toSegundos, t.q. devuelve el número de tipo entero en segundos en total que posee la hora actual.
addMinutos, t.q. devuelve la hora completa, añadiendo a la hora actual un valor entero considerado como entrada, expresado en minutos.
addSegundos, t.q. devuelve una Hora completa, dando como entrada un valor de tipo entero expresado en segundos.

USO DE LA CLASE : Hora.

a) Crear las horas: (indicar en todo momento los valores de cada hora)
b) Hora **h1**, que tome la hora por defecto.
c) Hora **h2**, que inicialice la hora a los valores 23, 25 y 48.
d) Hora **h3**, que inicialice la h a 4 horas, minutos y segundos a 0.
e) Hora **h4**, que inicialice la **h** a la hora (h) de h1, **m** a 34 y **s** a 0.
f) Hora **h5**, que inicialice las propiedades de la hora introducidas por teclado.
g) Hora **h6**, que inicialice las propiedades a los valores de las de h4.
h) Hora **h7**, que inicialice **h** a la hora de h2, **m** a los minutos de h3 y **s** a la hora de h4.
i) Hora **h8**, que inicialice sus propiedades: **h** a hora de h7, **m** y **s** a 0.
j) Hora **h9**, que inicialice sus propiedades a la hora resultado de añadir 14 segundos a la hora h7.
k) Mostrar las propiedades de todas las horas.
l) Cambiar los minutos a hora (h) de h8 por los segundos de h7.
m) Cambiar la hora completa de h7: h toma h de h6, m es 14 y s toma s de h4.
n) Cambiar la hora completa de h9 por los valores de la hora h4.
o) Cambiar la hora completa de h6 por la hora resultado de añadir segundos en h4, tantos como segundos de h5.

8.

Clase Cuenta

Diseña también la clase **Escritura**:
Métodos:
write →t.q. dado un String, lo muestra por pantalla, sin salto de línea. Público.
writeln →t.q. dado un String, lo muestra por pantalla, con salto de línea. Público.
Diseña la clase **Cuenta**:
Propiedades (todas privadas):
beneficio →es la cantidad que da el banco a todos los titulares de la cuenta. Esta cantidad es la misma en todas las cuentas. Real.
titular →nombre del titular de la cuenta. String.
saldo →saldo de la cuenta. Real.
Métodos:
iniciaCuenta →dados un nombre de persona y un saldo, se dan datos al objeto actual. Público.
ingreso → t.q. dada una cantidad, la añada al saldo de la cuenta actual.
reintegro → t.q. dada una cantida, le reste al saldo de la cuenta actual. Es necesario comprobar si es suficiente saldo.
pedirBeneficio → t.q. pida mediante un mensaje el beneficio que da la empresa por ctas. y el beneficio que ésta ofrece a todas las cuentas.
toString → t.q. devuelva en una cadena los valores de la cuenta actual.

Diseña la clase **Principal**:
Crea dos cuentas: cliente1, cliente2
Da valores a cliente1: "Arancha Pérez", 100
Da valores a cliente2: "Emilio González", 500
Pedir el beneficio que ofrece el banco a las cuentas de los clientes.
Mostrar la cuenta cliente1
Mostrar la cuenta cliente2
Ingresar la cantidad de 1000 a la cuenta cliente1.
Reintegrar en la cuenta cliente2 la cantidad de 1000
Mostrar la cuenta cliente1 y la cuenta cliente2.

10.

Clase Familia

Escribe un programa que pida los datos de los cuatro miembros de una familia. Los datos que nos interesa de cada uno de ellos son: nombre, dirección y edad.

La dirección de todos los miembros de la familia será siempre la misma. Con lo cual, este dato solo se pedirá una vez y si se cambia la dirección de un familiar, esto tendrá que afectar al resto de la familia.

Visualizar los datos de cada uno de los miembros de la familia.

Posteriormente, la familia cambia de dirección, con lo cual, el programa pedirá esa nueva dirección.

Vuelve a visualizar los datos de todos los miembros de la familia con el fin de comprobar que el cambio de de dirección ha afectado a todos los miembros. Por último, se visualizará la media de edad de dicha familia.