PROGRAMACIÓN - EXAMEN 1ª EVALUACIÓN

Entrega del examen: carpeta comprimida con nombre y apellidos que contenga este archivo, los pseudocódigos (en formato psc) y los ficheros fuente de las clases.

1. **Realiza el pseudocódigo y la codificación en lenguaje de programación Java de un algoritmo que solicite las longitudes de los catetos de un triángulo rectángulo** (no es necesario validar) **y muestre por pantalla la hipotenusa, el área y el perímetro del rectángulo.**

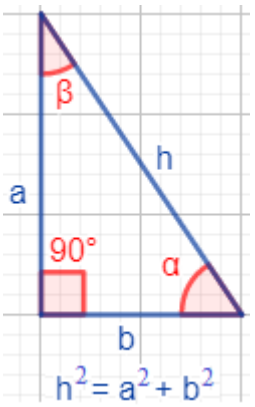
   Sean **a** y **b** los catetos.

   Para calcular la hipotenusa: $h=\sqrt{a^2+b^2}$

   Para calcular el área: $A=\frac{a*b}{2}$

   Para calcular el perímetro: $P=a+b+h$

   Cálculo raíz cuadrada
   PseInt: raiz(x)
   Java: Math.sqrt(x)

   Ejemplo:

   Entrada: Cateto a: 3
   Cateto b: 4

   Salida: h = 5.0
   A = 6.0
   P = 12.0

2. **Realiza el pseudocódigo y la codificación en lenguaje de programación Java de un algoritmo que solicite números enteros positivos por teclado (valida la entrada) y por cada número introducido nos diga si es un número de Harshad o no. El proceso acabará cuando se introduzcan 3 números de Harshad.**

   Un número es número de Harshad si es divisible por la suma de sus dígitos.

   Ejemplo:

   Entrada: 45 4+5=9 ⇒ 45%9=0 ⇒ Salida: ES NÚMERO DE HARSHAD
   Entrada: 9476 9+4+7+6=26 ⇒ 9476%26=12 ⇒ Salida: NO ES NÚMERO DE HARSHAD
   Entrada: 2020 2+0+2+0=4 ⇒ 2020%4=0 ⇒ Salida: ES NÚMERO DE HARSHAD
   Entrada: 99972 9+9+9+7+2=36 ⇒ 99972%36=0 ⇒ Salida: ES NÚMERO DE HARSHAD

3. **Escribe un programa en Java que solicite por teclado una frase y muestre por pantalla si se trata de un tautograma o no. Una frase será un tautograma si todas las palabras comienzan por la misma letra.**

   La frase podrá estar en mayúsculas o minúsculas.

   Se asegura que las palabras no tienen tildes.

   La frase no contendrá símbolos ni dígitos.

   Ejemplos:

   Entrada: Antes alegre andaba Salida: TAUTOGRAMA
   Entrada: Nada ni nadie nos necesita Salida: TAUTOGRAMA
   Entrada: Francisco frunce la frente Salida: NO

4. Implementa en lenguaje de programación Java la clase PolinomioG2 representada en el siguiente diagrama de clases.

PolinomioG2
-a: float -b: float -c: float
+PolinomioG2(a: float, b: float, c: float) +getA(): float +getB(): float +getC(): float +setA(a: float): void +setB(b: float): void +setC(c: float): void +evaluar(x: float): float -getDiscriminante(): float +getRaices(): String +toString(): String

Un polinomio de grado 2 corresponde a la siguiente expresión: ax^2+bx+c donde ***x*** es la variable y ***a***, ***b*** y ***c*** representan los coeficientes.

El constructor ***PolinomioG2(float a, float b, float c)*** crea un polinomio con los valores ***a***, ***b*** y ***c*** especificados. Si ***a*** es 0 lo sustituye por 1.

El método ***setA(float a)*** cambia el valor del atributo ***a*** si el valor recibido es distinto de 0. Si el valor recibido es 0 no cambia el valor del atributo ***a***.

✔ El método ***evaluar(float x)*** devuelve el valor del polinomio para el valor ***x*** pasado como argumento. El valor se obtiene de la expresión: ax^2+bx+c

✔ El método privado ***getDiscriminante()*** devuelve el resultado de la siguiente expresión: b^2-4ac

✔ El método ***getRaices()*** devuelve las posibles soluciones (raíces) de la ecuación de segundo grado $ax^2+bx+c=0$

El formato de la cadena devuelta por el método es el siguiente: **[x₁,x₂]**

La fórmula general para resolver una ecuación de segundo grado es la siguiente:

$x=\frac{-b\pm\sqrt{b^2-4ac}}{2a}$ donde ***a***, ***b*** y ***c*** son los coeficientes de la ecuación.

b^2-4ac es el discriminante y puede tomar tres valores: positivo, negativo o cero y será lo que determine el número de raíces de la ecuación.

- Si el discriminante es cero la ecuación tiene una solución real repetida: $x_1=\frac{-b}{2a}$ $x_2=\frac{-b}{2a}$
- Si el discriminante es positivo la ecuación tiene dos soluciones reales: $x_1=\frac{-b+\sqrt{b^2-4ac}}{2a}$ $x_2=\frac{-b-\sqrt{b^2-4ac}}{2a}$
- Si el discriminante es negativo la ecuación tiene dos soluciones complejas: $x_1=parteReal + parteImaginaria\ i$ $x_2=parteReal - parteImaginaria\ i$

  Primero calcularemos la parte real: $parteReal=\frac{-b}{2a}$

  y luego la parte imaginaria, para lo que tendremos que calcular el valor absoluto del discriminante: $parteImaginaria=\frac{\sqrt{|discriminante|}}{2a}$

✔ El método ***toString()*** devuelve una representación del polinomio con el siguiente formato: ***ax^2+bx+c*** ***(cuando los coeficientes a o b son 1 o -1, no deben aparecer)***

Ejemplos:

POLINÓMIO	x₁	x₂	a	b	c	evaluar(2)	getDiscriminante()	getRaices()	toString()
$2x^2+4x+2$	$x_1=-1$	$x_2=-1$	2	4	2	18	0	[-1.0,-1.0]	2x^2+4x+2
$-2x^2-x+1$	$x_1=-1$	$x_2=0.5$	-2	-1	1	-9	9	[-1.0,0.5]	-2x^2-x+1
x^2-2x+5	$x_1=1+2i$	$x_2=1-2i$	1	-2	5	5	-16	[1.0+2.0i,1.0-2.0i]	x^2-2x+5

CALIFICACIÓN				
		Correcto. Se ajusta a especificaciones.	Correcto pero con problemas de claridad, legibilidad o eficiencia. Correcto pero formato de salida incorrecto.	Sin realizar. No se ajusta a especificaciones. Errores de lógica. Errores de ejecución.
Ejercicio 1	Pseudocódigo	1	0,5	0
	Codificación en Java	1	0,5	0
Ejercicio 2	Pseudocódigo	1	0,5	0
	Codificación en Java	1	0,5	0
Ejercicio 3	Codificación Java	1,5	0,75	0
Ejercicio 4	Declaración de la clase, atributos, constructor, getters y setters	0,5	0,25	0
	método evaluar	0,5	0,25	0
	método getDiscriminante	0,5	0,25	0
	método getRaices	2	1	0
	método toString	1	0,5	0