IMPLEMENTACIÓN DE CLASES SENCILLAS I

Proyecto Netbeans 1: Figuras Geométricas

En esta tarea se implementarán varias clases en Java y se probará su funcionamiento. Crea un proyecto Netbeans llamado ***FigurasGeometricas*** que contengan las clases especificadas en los ejercicios siguientes.

Ejercicio 1: La clase Punto

Esta clase modela la posición de un punto en el plano (2D). Su especificación es la siguiente:

- Necesitaremos los **atributos privados**:
  - **x** (double): Posición en el eje de abscisas.
  - **y** (double): Posición en el eje de ordenadas.
- Implementaremos los siguientes **métodos públicos**:
  - **Punto()**: Constructor por defecto. Establece el punto en el origen de coordenadas (0,0).
  - **Punto(double x, double y)**: Constructor que inicializa los atributos a los valores pasados por parámetro.
  - **Punto(Punto p)**: Constructor copia que inicializa los atributos a los valores del objeto *Punto* pasado por parámetro.
  - Métodos **get** y **set** para los dos atributos.
  - Redefine el método **String toString()**: para mostrar la información del objeto en formato texto.
  - Redefine el método **boolean equals(Punto p)**. Para determinar si dos puntos son iguales.

Implementa la clase principal **MainPunto** que pruebe el funcionamiento de esta clase creando objetos llamando a los diferentes constructores y haciendo uso de los diferentes métodos de la misma.

Ejercicio 2: La clase Circulo

Esta clase modela la figura geométrica de un Circulo. Su especificación es la siguiente:

- Necesitaremos los **atributos privados**:
  - **centro** (Punto): Objeto de la clase Punto creada anteriormente que determina el centro del círculo
  - **radio** (double): Radio del círculo. Debe ser mayor o igual que cero.
- Implementaremos los siguientes **métodos públicos**:
  - **Circulo()**: Constructor por defecto. Establece el centro en el origen de coordenadas (0,0) y el radio a cero.
  - **Circulo(Punto p, double radio)**: Constructor que inicializa los atributos a los valores pasados por parámetro. Si el radio es negativo se lanzará la excepción IllegalArgumentException.
  - **Circulo(Circulo c)**: Constructor copia que inicializa los atributos a los valores del objeto

Circulo pasado por parámetro.
- **Circulo(double origen_x, double origen_y, double radio)**: Constructor que inicializa los atributos a los valores pasados por parámetro. Si el radio es negativo se lanzará la excepción *IllegalArgumentException*.
- Métodos **get** y **set** para los dos atributos.
- **double getArea()**: Devuelve el valor del área del círculo.
- **double getCircunferencia()**: Devuelve la longitud de la circunferencia.
- Redefine el método **String toString()**: Para escribir el círculo en pantalla.
- Redefine el método **boolean equals(Circulo c)**: Para determinar si dos círculos son iguales.

Implementa la clase principal **MainCirculo** que pruebe el funcionamiento de esta clase creando objetos llamando a los diferentes constructores y haciendo uso de los diferentes métodos de la misma.

Ejercicio 3: Otras figuras geométricas

Creo otras clases que modelen otras figuras geométricas más complejas y la clase principal para probar el funcionamiento de las mismas.

Proyecto Netbeans 2: Academia

En esta tarea se implementarán varias clases en Java y se probará su funcionamiento. Crea un proyecto Netbeans llamado ***Academia*** que contengan las clases especificadas en los ejercicios siguientes.

Ejercicio 1: La clase Asignatura

Esta clase modela una asignatura de estudio. Su especificación es la siguiente:
- Necesitaremos los **atributos privados**:
  - **nombre** (String): Nombre de la asignatura.
  - **horas** (int): Número de horas a la semana. Debe ser un entero positivo.
- Implementaremos los siguientes **métodos públicos**:
  - **Asignatura(String nombre, int horas)**: Constructor que establece los valores de los atributos. Si el valor de horas no es correcto se lanzará la excepción correspondiente.
  - **Asignatura(Asignatura a)**: Constructor copia que establece los valores de los atributos a los del objeto pasado por parámetro.
  - Métodos **get** y **set** para los dos atributos.
  - Redefine el método **String toString()**: Para escribir la asignatura en pantalla.
  - Redefine el método **boolean equals(Asignatura a)**. Para determinar si dos asignaturas son iguales.

Implementa la clase principal **MainAsignatura** que pruebe el funcionamiento de esta clase.

Ejercicio 2: La clase Estudiante

Esta clase modela un estudiante. Su especificación es la siguiente:

- Necesitaremos los **atributos privados**:
  - **nombre** (String): Nombre del estudiante.
  - **apellido1** (String): Primer apellido del estudiante.
  - **apellido2** (String): Segundo apellido del estudiante.
  - **listaAsignaturas** (Asignatura []): Asignaturas en las que se encuentra matriculado el alumno.
- Implementaremos los siguientes **métodos públicos**:
  - **Estudiante(String nombre, String apellido1, String apellido2)**: Constructor que establece los valores de los atributos nombre y apellidos.
  - **Estudiante(Estudiante e)**: Constructor copia que establece los valores de los atributos a los del objeto pasado por parámetro.
  - Métodos **get** y **set** para los atributos.
  - **boolean añadeAsignatura(Asignatura a)**: Añade la asignatura a la lista de asignaturas matriculadas. Un alumno no puede estar matriculado de más de 30 horas, por lo que se debe comprobar que no se supera este límite antes de añadir la asignatura. Devuelve **true** si la asignatura se añade correctamente y **false** en caso contrario.
  - **int getNumeroAsignaturasMatriculadas()**: Devuelve el número de asignaturas matriculadas.
  - **int getNumeroHorasMatriculadas()**: Devuelve el número de horas matriculadas.
  - **Asignatura getAsignatura(int posicion)**: Devuelve la asignatura que ocupa la posición indicada por parámetro en la lista de asignaturas.
  - Redefine el método **String toString()**: Para escribir el Estudiante en pantalla.
  - Redefine el método **boolean equals(Estudiante e)**. Para determinar si dos estudiantes son iguales (son iguales si se llaman igual).

Implementa la clase principal **MainEstudiante** que pruebe el funcionamiento de esta clase.