Nombre: ..... Fecha: / 10 / 2009

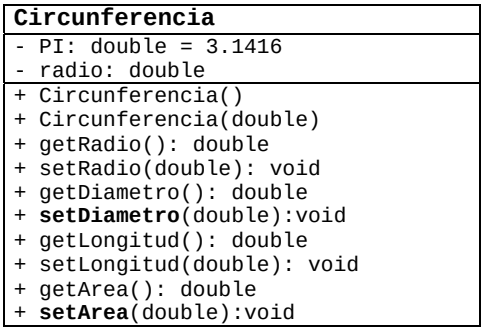
Grupo: 1 ☐ 2 ☐ 3 ☐ 4 ☐

PRÁCTICA 2
DECLARACIÓN Y DEFINICIÓN DE CLASES EN JAVA

En esta práctica se buscarán los siguientes objetivos. En primer lugar, conseguir la familiarización del alumno con el IDE para Java JCreator LE 3.50 (disponible en http://www.jcreator.com). Asímismo, aprender a traducir diagramas de clases en el lenguaje de especificación UML a Java. Teniendo en cuenta que es la primera vez que los alumnos usan Java, se pretende que el alumno se familiarice con la sintaxis propia del lenguaje. En la práctica el alumno debe aprender la sintaxis de Java para la declaración y definición de clases. Por último, el alumno ha de ser capaz de definir clientes (la función "main") que hagan uso de las clases anteriormente creadas; estos clientes nos servirán para introducir la sintaxis propia de la entrada y salida de datos por teclado en Java.

La forma de conseguir los anteriores objetivos será traduciendo diversos diagramas de UML al código correspondiente en Java utilizando JCreator como IDE. Los ejemplos propios de esta práctica nos servirán en prácticas posteriores para realizar con ellos construcciones más elaboradas.

1. A partir del siguiente diagrama que especifica en UML los atributos y métodos de una clase **Circunferencia**, crea un fichero *Circunferencia.java* donde se declare y defina la clase anterior.



2. Crea un programa principal que actúe como cliente de la clase **Circunferencia** (guárdalo en un fichero *Principal_prac1_1.java*). Más concretamente, crea tres objetos distintos de la clase **Circunferencia**. Crea primero un objeto de la clase **Circunferencia** haciendo uso del constructor sin parámetros. En segundo lugar, un objeto de la clase **Circunferencia** cuyo radio inicial sea 7.5. Por último, crea un objeto de la clase **Circunferencia** cuyo radio inicial sea 4.8.

3. Utiliza los métodos de acceso de la clase para mostrar por pantalla el valor del área y la longitud de los objetos que has creado. Comprueba que no puedes acceder desde tu cliente a ninguno de los atributos de la clase declarados como "**private**".

4. Utiliza los métodos de modificación para hacer que las circunferencias pasen a tener las siguientes propiedades:

La primera circunferencia debe tener como área la que el usuario introduzca por teclado

La segunda circunferencia debe tener un radio mayor en tres unidades al inicial

La tercera circunferencia debe pasar a tener la longitud que el usuario introduzca por teclado

5. Seguimos con un ejemplo distinto. Imagina que quieres informatizar un sistema de préstamo de películas por parte de una biblioteca. Aparte de los requerimientos más

complejos del sistema, de momento nos quedamos únicamente con que hacen falta dos clases llamadas **Pelicula** y **Usuario** cuya especificación en UML sea la siguiente:

Pelicula
- titulo: string - director: string - duracion: integer
+ Pelicula() + Pelicula(string,string,integer) + getTitulo(): string + setTitulo(string): void + getDirector(): string + **setDirector**(string):void + getDuracion(): integer + setDuracion (integer): void

Usuario
- nombre: string - dni: integer
+ Usuario() + Usuario(string,integer) + getNombre(): string + setNombre(string): void + getDNI(): integer + **setDNI**(integer):void

Guarda las clases en ficheros distintos, *Pelicula.java* y *Usuario.java*. La forma más sencilla de codificar en Java los campos definidos en UML como "string" es usar la clase **String**.

6. Define un cliente de las clases **Pelicula** y **Usuario** en la forma de un programa principal (guarda el fichero como *Principal_prac1_2.java*) y comprueba que las clases están bien programadas (es decir, se pueden crear objetos, se puede acceder a los atributos de las mismas, se pueden modificar, ...). Define una función auxiliar en el propio fichero *Principal_prac1_2.java* que reciba como parámetro un objeto de tipo **Usuario** y actualice el nombre del mismo con una nueva cadena leída por teclado.

7. Vamos a definir en un nuevo ejemplo una clase para representar puntos en el plano. La codificación de la misma comprende dos coordenadas, una para x y otra para y, y los métodos de acceso y modificación de dicha clase. La representación en UML de la clase **Punto** sería:

Punto
- coord_x: double - coord_y: double
+ Punto() + Punto(double,double) + getX(): double + setX(double): void + getY(): double + **setY**(double):void

8. Define un cliente de dicha clase, de nuevo en forma de un programa principal (guarda el fichero como *Principal_prac1_3.java*), que te permita crear objetos de la case **Punto**, acceder a sus atributos, modificarlos, ...

ENTREGAR los archivos *Circunferencia.java, Principal_prac1_1.java, Pelicula.java, Usuario.java, Principal_prac1_2.java, Punto.java, Principal_prac1_3.java*. Los archivos deben contener **todos los cambios** realizados durante los ejercicios de la práctica. La entrega es a través de Belenus en el repositorio de Julio Rubio. También deben contener unas cabeceras que permitan identificarte:

/* Nombre: ___
Grupo: _____
Nombre del fichero: ____
Descripción: _____*/

Guarda una copia de los ficheros para las prácticas sucesivas

Nota: Para compilar programas en Java necesitas descargar la JDK (Java Development Kit) que puedes encontrar, por ejemplo, en http://java.sun.com/javase/downloads. El IDE JCreator está disponible bajo licencia GNU; puedes descargarlo desde la dirección http://www.jcreator.com