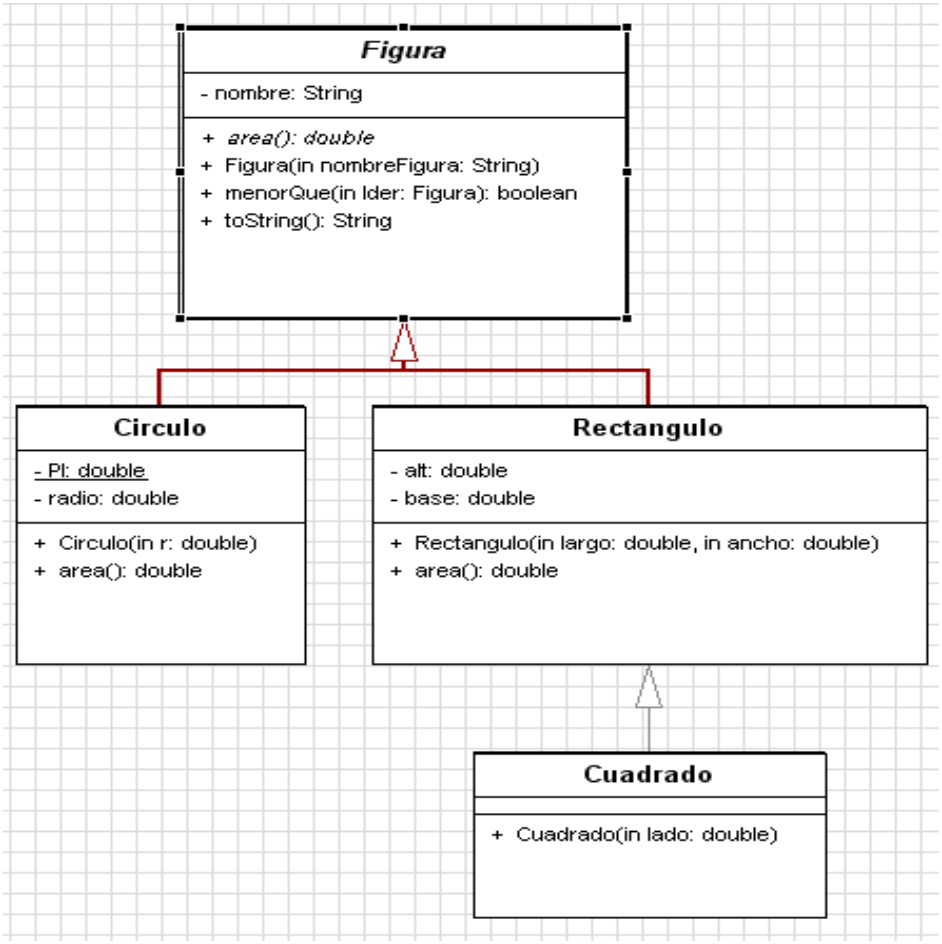
PRÁCTICA 2. Documentación en Java.
Herramienta de documentación automática *javadoc*.
UML. Diagramas de clases. *UML Amateras*.
Casos de Prueba: *JUnit*.

Ejercicios a realizar:

1) En el archivo zip (Figura.zip), que habrá descargado junto con esta práctica, encontrará las clases Figura, Circulo, Rectángulo, Cuadrado, PruebaFigura. Resuelve el siguiente problema: 'Ordenar Figuras'. cuya especificación es: 'Léanse los datos de N figuras (círculos, cuadrados o rectángulos) y muéstrense ordenados por área' . Deberá estudiar su funcionamiento, documentando el diseño con ayuda de *javadoc*, según la norma de la plantilla de prácticas.

Nota: nombre paquete com.mp.practica2.ejercicio1



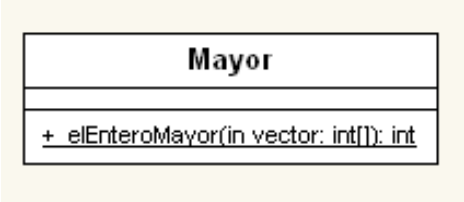
2) Reescriba el código del ejercicio sobre la empresa de cobros de la primera práctica. Introduzca los comentarios *javadoc* necesarios para cumplir con la norma de la plantilla de prácticas. Ejecute *javadoc* y compruebe la salida mediante un navegador.

3) Utilizando la herramienta UML Amateras instalada como un plugin sobre Eclipse, genere el diagrama de clases. Interaccione con el diagrama probando los niveles de visualización, formatos de visualización, manejo de código fuente, impresión, exportación a imagen, etc…

   a) Genere el diagrama de clases del ejemplo Figura.

   b) Genere el diagrama de clases del ejercicio tercero de la primera práctica.

4) Junto con esta practica se acompañaba de un juego de prueba realizado con JUnit para probar la clase Mayor que tiene el método *elEnteroMayor,* este método devuelve el entero mayor de una lista de enteros que se le pasa como entrada, la lista esta soportada por un array.



   a) El alumno deberá ejecutar los juegos de prueba utilizando la vista JUnit de la perspectiva java del IDE Eclipse. Compruebe el funcionamiento del juego de prueba para la clase Mayor.

   b) Corrija a través de las pruebas los errores encontrados en el código.

   Nota: el sistema de nombrado para el paquete es **com.mp.practica2.ejercicio4**

   Y para las pruebas de unidad : **com.mp.practica2.ejercicio4.test**

5) A partir del código del ejercicio 3 de la primera práctica, trasládelo al paquete de este ejercicio y realice las modificaciones necesarias para que pase el juego de pruebas.

   Nota: el sistema de nombrado para el paquete es **com.mp.practica2.ejercicio5**

   Y para las pruebas de unidad : **com.mp.practica2.ejercicio5.test**

6) Este ejercicio se refiere al sistema informático de una residencia de ancianos. En lo que se refiere al apartado de software, se ha decidido empezar con un sistema nuevo. Este se ha concretado, en una primera etapa, en el desarrollo de un sistema de gestión de reserva de habitaciones.

   Después de una investigación preliminar, basada en varias entrevistas y en la revisión de documentación, el dominio de problema se ha concretado en una especificación del software textual con el objetivo de una primera versión del sistema. De la cual se extrae el siguiente resumen:

   La residencia cuyo nombre será “Las acacias”, se estructura en habitaciones. Las habitaciones están todas equipadas igual, disponiendo de dormitorio, cuarto de baño y una pequeña salita de estar. Aún está por determinar el número de habitaciones finales que se construirán. Todas serán individuales. Y se identificarán de la forma habitual en los establecimientos hoteleros, con un número cuyo primer dígito identifica la planta y los dígitos que le siguen el número de habitación.

   De los ancianos, a los que se referirá como residentes, se guardarán los datos Dni, Nombre, Fecha nacimiento y Sexo.

Del personal encargado de sus cuidados, a los que se referirá como cuidadores, se guardará su Dni y Nombre.

La funcionalidad básica que se pide en este prototipo es:

La operación de ingreso en la residencia de un residente consiste en tomarle sus datos y añadirlo a la residencia y en asignarle una habitación mediante una reserva con una fecha de entrada y una fecha de salida posterior.

La operación de salida de la residencia de un residente consiste en dar de baja (eliminar) el residente, y en buscar su reserva y asignarle la fecha de salida.

La operación de cambio de habitación de un residente consiste en asignar la fecha de salida a la reserva actual del residente y en crear una nueva reserva con el residente, la nueva habitación y como fecha de entrada la fecha de salida.

Cada vez que se crea una nueva reserva, esta constará de un número de reserva entero positivo único que se generará autoincrementado, es decir 1, 2, 3, …. Además constará con los datos de residente, habitación, fecha de entrada y fecha de salida.

Y obtener una serie de listados con información sobre el sistema:

1. Un listado ordenado por el nombre de todos los residentes y las habitaciones que tienen asignadas en una fecha.

2. Un listado de las habitaciones libres en una fecha.

3. Un listado con la edad media en años de los residentes según el sexo.

Restricciones de diseño: Se utilizará para la repetición la estructura array. Por ejemplo: El número de residentes puede ser desde 0 hasta n (un número indeterminado). Por defecto los arrays se inicializan a un tamaño de 5.

Junto con este documento se ha incluido un juego de pruebas para probar este ejercicio. Ejecute el juego de pruebas hasta que sea pasado. Haga las modificaciones necesarias hasta que pase el juego de prueba sin modificar este. Para realizar este apartado debe utilizar JUnit en la perspectiva Java de Eclipse.

Nota: Para las pruebas de unidad, el paquete se situará en : **com.mp.practica2.ejercicio6.test**

Recuerde: No se puede modificar el código del juego de pruebas para que se adapte a su código. En el paradigma de la programación extrema el juego de pruebas se realizan con frecuencia antes que la propia implementación de las clases.