1. INTRODUCCIÓN

2. FASES Y ENTREGABLES DE LA PRÁCTICA

3. DOCUMENTACIÓN A ENTREGAR

3.1. FASE I: DISEÑO DE LAS ESTRUCTURAS DE DATOS

Se pide:

a) **Representar** gráficamente (utilizando el diagrama de clases o diagrama de objetos de UML [5]) la estructura de las clases (principalmente, nombre + atributos + constructores + operaciones) necesarias para representar las entidades y la información más importante de la práctica. En cada caso, el grupo deberá seleccionar alguna de las estructuras de datos vistas en clase para organizar los datos y **argumentar las decisiones** tomadas.

b) **Desarrollar** en Java[6] el apartado a), sin llegar a implementar las operaciones. En este caso, especificar mediante JavaDoc[7] la pre- y post-condiciones. Para ello, se tendrán en cuenta las características de la práctica así como las operaciones principales que se van a necesitar, detalladas en el capitulo 4. El grupo deberá documentar cada clase y sus atributos para facilitar la comprensión y la utilización de la misma.

Recordar, que en esta fase no deberíamos de olvidar los principios de la programación, tales como, la Modularización y la Abstracción.

4. DESCRIPCIÓN DE LA PRÁCTICA

Se desea realizar un sistema informático que ayude a gestionar una colección de figuras realizadas a mano. A cada figura se le asociaría un nombre al ahora de guardar, para que en un futuro podamos consultar si la tenemos o no, poder recuperarla o modificarla.

Las figuras representan caracteres (o, en general, símbolos) escritos a mano mediante una secuencia de trazos. Supongamos, por simplificar, que los trazos disponibles son D (derecha), B (bajar), I (izquierda) y S (subir). Ejemplos:

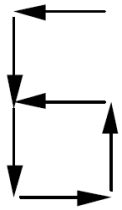

Figura 1 IBBDSI

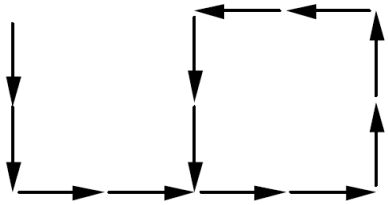

Figura 2 BBDDDDSSIIBB

Se quiere tener la opción de manipular las figuras, tales como poder girar toda la figura a la derecha (90º grados) y realizar la homotecia de factor 2 (multiplicar por 2 el tamaño de la figura). Ejemplo: la figura 2 es el resultado de aplicar 3 giros y una homotecia a la figura 1. También se quiere tener la opción de componer mediante las operaciones de:

- *fusionar*: dado dos figuras se añade la segunda figura al final de la primera
- *insertar*: dado dos figuras y una posición entre [*0..n*], se añade la segunda figura a la primera después del trazo que se encuentra en la posición dada. La posición 0 se refiere a la posición antes del primer trazo de la figura, y el valor de *n* a la longitud de la primera figura, es decir, al número total de trazos en la primera figura.

A la hora de analizar las figuras, además de la longitud, nos interesaría saber cual es su altura máxima, anchura máxima y su superficie. Ejemplos: la *Figura 1* tiene altura 2, anchura 1 y superficie 2; la *Figura 2* tiene altura 2, anchura 4 y superficie 8. Además de poder comparar si dos figuras son iguales, homotéticas o semejantes:

- Se dice que una figura es igual que otra si todas las secuencias de trazos que las representan también son iguales.
- Se dice que una figura es homotética de otra si las dos figuras tienen la misma orientación y si mediante un número de homotecias es posible obtener una figura de la otra. (Ayuda: utilizar la altura y la anchura para averiguar el posible número de homotecias de factor 2)
- Se dice que una figura es semejante a otra si existe una secuencia de giros y homotecias que permiten transformar la primera figura en la segunda. Diseñar un algoritmo que devuelve *False* si la primera figura no es semejante al segundo o devuelve *True*, si la primera se puede transformar en la segunda a través de giros y homotecias.

5. DISEÑO Y FUNCIONALIDAD DE LA PRÁCTICA

Después de haber finalizado la Fase I, el alumno recibirá el diseño y la especificación de la funcionalidad que tiene que implementar para realizar la Fase II.

6. BIBLIOGRAFÍA

1. JUnit. http://www.junit.org
2. Tutorial de JUnit, ver 'Material complementario' en eGela.
3. eGela. http://egela.ehu.es
4. Web-CAT. http://lsi.vc.ehu.es/Web-CAT/WebObjects/Web-CAT.woa
5. Resumen de UML, ver 'Material complementario' en eGela.
6. Java. https://www.oracle.com/java/index.html
7. JavaDoc, ver 'Material complementario' en eGela.
8. Eclipe. http://www.eclipse.org