# Lab exercise 2: Implementar un disseny de programa 24292-Object Oriented Programming

Simón Gasión: 240126
Jordi Polo: 239980

## 1.Introduction

En esta práctica se ha de crear un programa que sea capaz de imprimir en un plano de dos dimensiones representando un mundo una série de polígonos denominados “regiones”, cada una delimitada por una série de puntos, y agrupados en continentes.

Para crear cada una de estas regiones se necesita crear una estructura de datos que represente cada punto, cada uno de estos se guardan en una lista asociada a la región a la que representa. La clase “PolygonalRegion” tiene dos funcionalidades principales; la más importante es dibujar la región en el lienzo, para lo cual se utiliza la librería de gráficos de java, y como rige el enunciado del Lab ésta puede también calcular su propia área.
Más allá de “PolygonalRegion” tenemos la clase “Continent”, que está formada por una lista de regiones y sirve para gestionarlas en bloque.

Similarmente, la clase ‘world’, que representa el lienzo en dónde se pintan todas las regiones, actúa como un cúmulo de continentes en forma de lista enlazada, los cuales están compuestos por regiones geométricas gracias a una lista enlazada.

## 2.Description

Para implementar los métodos de “PolygonalRegion” es importante tener en cuenta una pieza de información que nos proporciona el enunciado del lab, y es que todas las regiones tendrán la forma de un polígono convexo. Gracias a este dato podemos averiguar el área con la siguiente fórmula matemática:

$$\text{Area} = \frac{1}{2}\begin{vmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \\ \vdots & \vdots \\ x_n & y_n \\ x_1 & y_1 \end{vmatrix} = \frac{1}{2}\left[(x_1y_2 + x_2y_3 + x_3y_4 + \cdots + x_ny_1) - (y_1x_2 + y_2x_3 + y_3x_4 + \cdots + y_nx_1)\right]$$

En el código del lab se añade el primer punto del polígono al último lugar de la lista también (por motivos que explicamos posteriormente), por lo tanto podemos traducir esta fórmula a código fácilmente usando bucles de N repeticiones sin tener en cuenta los términos $x_ny_1$ y $y_nx_1$.

Para dibujar las líneas de punto a punto el enunciado del lab sugiere usar la función drawLine de “graphics”, pero hemos observado que eso no es un método viable por lo que hemos usado una subclase de graphics llamada graphics2D, que es una versión más avanzada y reciente.

Posteriormente a la implementación de la clase “PolygonalRegion” hemos creado la clase continent la cual coge como atributo una lista enlazada de regiones poligonales, y conteniendo está el método constructor de la clase, otro método para conseguir el area de todas las formas geométricas que dispone el continente y otra para ser dibujadas.

Seguido a la clase “Continent” creamos la clase “World” la cual coje como atributo una lista enlazada de continentes, y contiene como métodos el propio constructor y el método para dibujar.

Para la implementación de las clases “Continent” y “World” no tuvimos que pensar demasiado al ser unas clases sencillas de implementar y especificada su implementación en la hoja de la práctica.

Finalmente para el cambio en la clase “MyMap” simplemente hemos tenido que definir diferentes puntos en diferentes ubicaciones para crear las distintas regiones poligonales, en nuestro caso 9 regiones, y creando diferentes continentes para añadir estas regiones en ellos, haciendo en nuestro caso que cada continente contenga 3 de las regiones, por tanto habiendo de crear 3 continentes. Y cuando estos han sido creados, añadimos estos a “World”. Claramente para crear tanto las regiones como los continentes y como el propio mundo, hemos generado distintas listas enlazadas y usado el método constructor para cada caso. A parte, imprimimos gracias al “getArea” para las regiones y al “getTotalArea” para los continentes, las áreas de tanto las regiones como los continentes. Dibujando finalmente el mundo a raíz de la función “draw” de la clase “World”.
Para los cambios en “MyMap” tuvimos libertad sobre todo en la distribución de los polígonos situando estos de manera que parezcan países en distintos continentes, estando los países de cada continente unidos entre ellos, y diferenciados de otros continentes. En el resto de los cambios fue simple pero largo al haber de implementar tantos puntos, y crear varias regiones para los continentes.

# 3.Conclusion

La práctica ha sido correctamente realizada, dibujando los polígonos y calculando las áreas de manera correcta, habiendo hecho varios ejemplos con polígonos de distintas medidas para comprobar que todo funciona correctamente.

La principal dificultad que hemos encontrado en la práctica surgió al intentar implementar el método “draw” de la clase “PolygonalRegion”, debido a que los métodos que sugiere el enunciado del lab sólo admiten argumentos de tipo “integer” mientras que los atributos de la clase “Point”, que són los datos que se utilizan para implementar dicho método, són de tipo “real”. Para solventar este problema necesitamos importar otra librería más actualizada que sí que permite operar con reales. Además, encontramos que, con la manera en la que escribimos el código, ningún polígono se acababa de cerrar, cosa que solucionamos fácilmente añadiendo el primer punto de la lista de “Points” de cada región en la cola de la lista.