Seminar 2: Designing an application
24292-Object Oriented Programming

1 Introduction

The objective of this seminar is to learn how to design an entire program, in which multiple classes interact in order to solve a given problem. The design is in the form of a class diagram, which contains class definitions and relations between classes of the different types that we have covered in the theory lectures.

As always, each class has a number of attributes and methods that are to be set to complete the design. Also remember that each completed class diagram should be a connected graph. We will practice the use of relations that we have seen in class, and we will pay special importance to the fact that some relations must appear as attributes in a given class. Remember also to add the cardinality of relations whenever appropriate.

A partial solution to these exercises will be implemented in Java during lab session 2.

2 Point, Geometric Point and Vector classes

Based on the classes that we designed in the previous seminar, now design three classes that model the Point, Vector and GeometricPoint classes, as well as the relations between these classes.

- A point has two coordinates x, y and a distance operator, as in the previous seminar.
- A vector also has two coordinates x, y, and defines vector operations like sum, scalar product and norm.
- A geometric point has a name, in addition to two coordinates, and can be drawn on the screen using an instance of the Graphics class.

In Java, the Graphics class is useful to access the screen window of our application and draw shapes on it. The Graphics class appears in a Java library called the Abstract Window Toolkit (awt) classes: java.awt.Graphics.

Below is the API of some relevant methods defined in the Graphics class:

```
drawString(String s, int x, int y);
drawLine(int x1, int y1, int x2, int y2);
drawOval(int x, int y, int width, int height);
```

This is an example of using the Graphics class from the theory lecture:

```
public class TextMessage {
  private String msg;
  ...
  public void draw( java.awt.Graphics graphics ) {
    graphics.drawString( msg, 0, 0 );
  }
}
```

3 Country and city classes

To model geographic regions, we will approximate each region using a geometric shape. Concretely, there exist two types of regions: Polygonal and Ellipsoid. A polygonal region is a polygonal shape made up of a sequence of points. An ellipsoidal region must specify its two diameters and its center. All regions can be drawn to the screen, using the methods from the Graphics class.

A city is a particular case of a GeometricPoint which adds a number of inhabitants. A country is a region that, in addition, specifies a name and a collection of cities. A particular city plays the special role of being the capital of that country. Countries are related to each other with a relation of neighborhood: each country is related to its neighbors. A country can be drawn on the screen through the Graphics class.

4 Map application design

All countries of the application are part of a single World. This world is subdivided into continents, which in turn contain regions. A continent can compute the total area it occupies based on the regions it contains.

A map is a class that we use to draw a world on the screen. While drawing the map, it can be shifted by a certain coordinate, making all the elements of the world drawn relative to the shift.