

Assignment 5 – 2012

Create public class *DemographyException* that represents an unchecked exception (extends the *RuntimeException* class) and has:

- Public constructor that receives, as its input parameter, an error message and passes this parameter to the constructor of the parent class.

Create public class *Region* that can be serialized and has the following elements:

- Private attribute *name*.
- Private attribute *birthRate* that represents the total number of children born since the latest census.
- Private attribute *deathRate* that represents the total number of people who died since the latest census.
- Private attribute *migrationBalance* that represents the change in the population size due to migration (it is an integer value)
- Get and set methods for these attributes. Invalid values for the *name* attribute include null and strings containing less than 2 letters, while the *birthRate* and the *deathRate* attributes have to be greater than zero. In case of an invalid value being entered, a *DemographyException*, with an appropriate error message, should be thrown.
- Redefined toString method (of the Object class). The method returns a piece of text (String) with all the data about the region. The text to be returned should also contain information about the change in the population size, which is computed as follows: $\text{change} = \text{birthrate} - \text{death rate} + \text{migration balance}$.

Create visual class *RegionsGUI* that looks like the one shown on the figure below. The title of the GUI window should be “Regional demographic data”, and the central part of the window should contain text editor. When a user resizes the window, the central part (with the text editor) should be enlarged/shrunk, while the other components should stay unchanged.

- The *RegionGUI* class should contain private attribute *regions*, which is a list of objects of the *Region* class; the list should be initialized right away.
- When the “Delete” button is pressed, the content of all the text fields and the text area should be deleted.
- When the “Save” button is pressed, all the elements of the *regions* list should be written (serialized) into one of these two files: “growing_regions.out” and “dying_regions.out”, depending on the change in the population size (this change is computed using the formula: $\text{change} = \text{birthrate} - \text{death rate} + \text{migration balance}$). Those regions where the change is positive should be written into the first file, while the others should be written to the second file.
- When the “Add” button is pressed, all the data about a region should be collected from the input fields of the GUI, and a new instance of the *Region* should be created and added to the list, and also printed in the text editor. The new region should be added only if the list doesn’t contain the same region (regions are considered to be the same if they have the same name). The new region, if not in the list, should be added to it in such a way that the descending order based on the regions’ birthrate is preserved in the list. If the list already contains the same region or an exception is thrown while entering or transforming the data, the word “ERROR” should be printed in the text editor.

The screenshot shows a graphical user interface window titled "Regional demographic data". On the left side, there are four labeled input fields: "Name:", "Birth rate:", "Death rate:", and "Migration balance:". Each label is followed by a text input box. To the right of these fields is a large, empty text area for output. At the bottom of the window, there are three buttons: "Add", "Save", and "Delete". The window has a standard macOS-style title bar with red, yellow, and green control buttons.