CS1 — Practical Session 8: Objects & Classes

Kurt Driessens

October 3rd, 2018

In this last practical session you will get experience with classes and objects in Java. There is only 1 exercise and an assignment. The Bonus is (according to me at least) good fun and gives you extra exercise on arrays and loops, and shows how easy it is to do complex things when you can use already available classes and objects.

This is the last time you'll see the teaching assistants in a lab, so if you have any remaining questions for them, make sure you ask them NOW!

Exercise 1

Complete the object started in the downloadable file HouseOnSale.java. The price of a house (in this exercise) is computed as follows:

- If the house has 2 or less bedrooms, the price is 2000 euros per squared meter.
- If the house has 3 or more bedrooms, the price is 1500 euros per squared meter.
- If the house has a garage, you must add 25000 euros to the price.

Write a main method that tests all the instance methods.

Assignment!

Design and implement a class Rectangle. This time you are also responsible for deciding which methods to include in the class and figuring out which parameters to use. Make sure the following abilities are supported:

- A default constructor that takes no parameters and makes rectangles of size 5 by 10.
- A constructor which allows specification of the size of the new rectangle.
- A method that returns the area of the rectangle.

If you want, you can add getters and setters for all instance variables too. It is good practice. Again write a main method to test your class.

Upload your `.java` file to get credit for your work.

Bonus — Image manipulation and hidden messages

In this exercise you get to combine your knowledge of arrays and objects to discovery some hidden messages. Downloadable from EleUM are three image files: "location1.bmp", "location2.bmp" and "location3.bmp". You can have a look at these images as they are originally, but you will probably learn little of the locations that they show. This is because the information in the image is hidden.

You can use the class **SimpleImageHandler** to help you decode the images. Objects of the class SimpleImageHandler have the following public interface:

- `public SimpleImageHandler(String path)`: This is the constructor. It loads the image file referred to by the "path" into memory.
- `public void showImage()`: This method shows the current image, as it is in memory, on the screen. Each time showImage() is invoked a new screen opens which displays the image as it exists in memory at that time. If the loaded image is bigger than 600×600 only the upper left part of the image is shown.
- `public int[][] getRedValues()`: This method returns all the intensity values of the red channel as a 2-dimensional integer array.
- `public void setRedValues(int[][] red)`: This method replaces all intensity values of the red channel by those of the 2-dimensional integer array parameter. Intensities larger than 255 are truncated to 255 (maximum intensity).
- `public int[][] getGreenValues()`: This method returns all the intensity values of the green channel as a 2-dimensional integer array.
- `public void setGreenValues(int[][] green)`: This method replaces all intensity values of the green channel by those of the 2-dimensional integer array parameter. Intensities larger than 255 are truncated to 255 (maximum intensity).
- `public int[][] getBlueValues()`: This method returns all the intensity values of the blue channel as a 2-dimensional integer array.
- `public void setBlueValues(int[][] blue)`: This method replaces all intensity values of the blue channel by those of the 2-dimensional integer array parameter. Intensities larger than 255 are truncated to 255 (maximum intensity).

In all of the 2-dimensional arrays, the first index refers to the height dimension (0 at the top) and the second to the width dimensions (0 at the left). You can use this class by putting the SimpleImageHandler.class file in the same directory as your own source file.

You will also need the following information on the three images to be able to decode them:

- For the first image "location1.bmp", which is almost completely black, the information about the location is simply hidden by lowering the intensity of the red channel. In this image, the green and blue channel are not used. You can get the image back by substituting the red values of all pixels by 10 times their current value.
- For the second image "location2.bmp", which is even more black, the information is hidden in all three color channels. This time the intensity was lowered extremely, so raising a single channel's intensity might not be sufficient to make the location recognisable. You will probably need to substitute the intensity of the values of all three channels by a larger multiple of their current value.
- For the third image "location3.bmp", which looks like coloured noise, the information is hidden in the green channel, but overpowered by the random values inserted in the red and blue channel. To recover the image, you will have to set the values of the red and blue channel to 0's and raise the intensity of the green values.