Assignment 4

DTU Compute
Technical University of Denmark
02102 Introductory Programming
03/04/2022

Group 32
Polly Clémentine Nielsen Boutet-Livoff - 214424

As I am the only member of the group, all the work was done by me.

1 Artikler

This software is not a much a program as it is a library. I have created classes that could be useful in representing scientific articles in software. Along with this there is a class called `ArtikelTest` that contains code which "tests" these classes. It instantiates various objects and runs some of the methods provided.

```
$ javac ArtikelTest.java && java ArtikelTest
A. Abe & A. Turing: "A". Journal of Logic
B. Bim: "B". Journal of Logic
C. Boutet-Livoff, P. Olly & C. lémentine: "C for yourself". Brain
```

Looking at the source code we can see that output reflects what we expect. Three articles are created with different data and are correctly displayed using the `toString` method. The code is written with OOP (Objected Oriented Programming) in mind; I encapsulate data, provide methods to interact with the objects and separate unrelated concepts into different classes. i.e. an article and a journal are related but different concepts and are therefor in different classes. All in all, it is a very simple library and a good showcase of some OOP practices. The testing could be improved by having software verify the expected functionality. This can be done by using assertions: simply checking that the output of some code (like `toString()`) is equal to the expected value (`B. Bim: "B". Journal of Logic`) and throwing an error if it isn't. I have elected not to do this as I don't believe it is required and I am capable of verifying the simple functionality of this program on my own.

2 Mandelbrot

To use the program run the Mandelbrot class and input the real component of the midpoint, the imaginary component and then the side length of the square area that the program is to render. After doing this the program will render a picture of the Mandelbrot fractal centered on the given complex number and with the given side length. The program supports more parameters but they require modifying the source code slightly. To change the resolution modify `GRID_SIZE`. To change the maximum iteration count (and thereby the accuracy) modify `MAX`. To change the colors used modify line 83's file path, or override the line with `Color[] colors = loadRandomColors();` in order to have random colors.

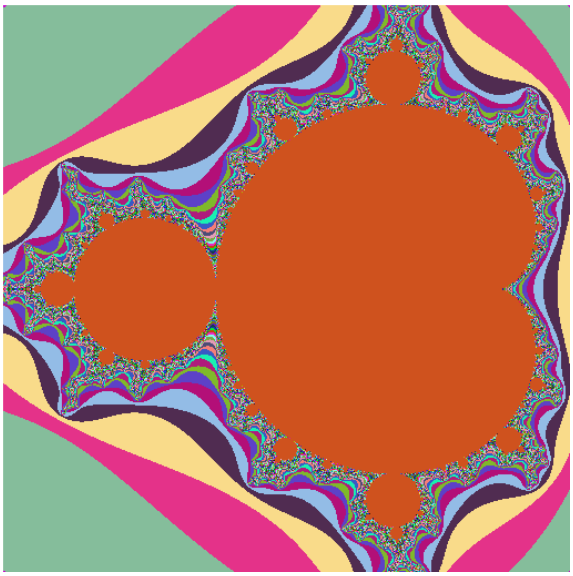

Figure 1: The Mandelbrot set using random colors.

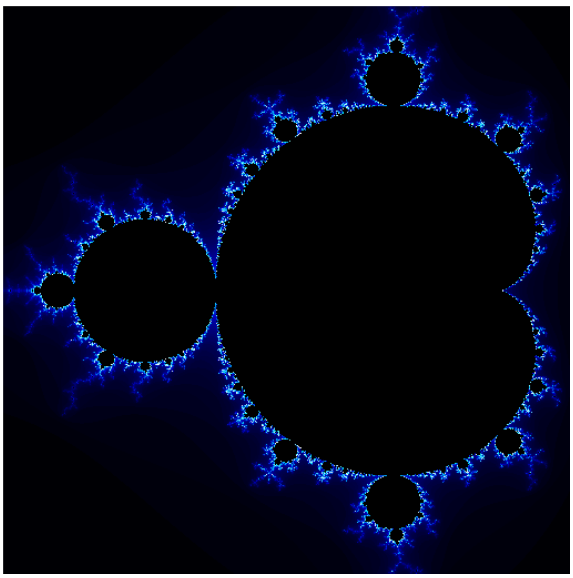

Figure 2: The Mandelbrot set shown using blues.mnd.

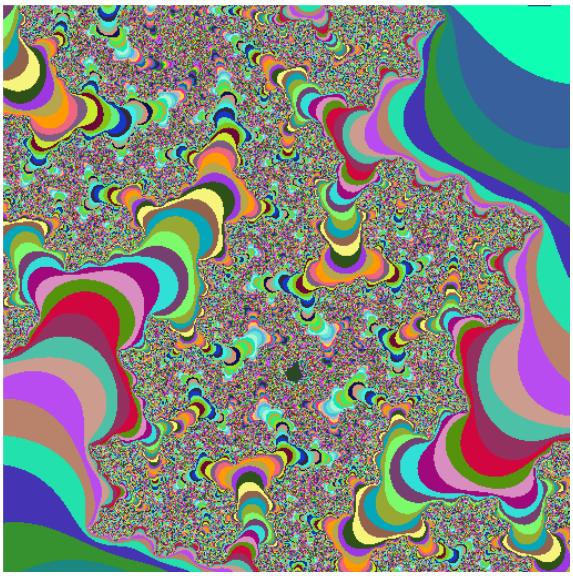

Figure 3: A closer look at 0.10087-0.63198i where the side length is 0.003. The random colors make it hard to see what parts of the inside of the fractal and what parts are the outside.

Some of the rendered images are shown in figure 1, 2 and 3. A lot of the code for this program is in the main function. You could see this as sloppy, it doesn't follow OOP. However none of the functionality is reused, so moving it into other functions would not reduce reuse. As for readability, I believe the code is easily understood using the provided comments and that moving it into a function would simply increase the complexity of the program. I elected not to add any additional functionality beyond what was required. However, the `Complex` class is a prime example of OOP providing clean and readable code.

3 Game of life

The program loads whatever file is specified in its source code, and required no additional configuration. If the source code is slightly modified the program supports random board generation. The `GameOfLife` class encapsulates its data and functionality such as the board and the rendering. This leaves the main function very simple and short. The calls to `StdDraw` can be somewhat inscrutable, but I would argue this is a flaw of the library. It is unclear what line 18 and 21 do. The board functions as a torus. This is implemented entirely in `liveNeighbours` through the use of modular arithmetic. In figure 4 we can see a test run of the program. The animation can be weirdly jittery, which I believe is either a flaw in the `StdDraw` library or my utilization of it, however I don't know how to fix this as the documentation for the library is lacking.

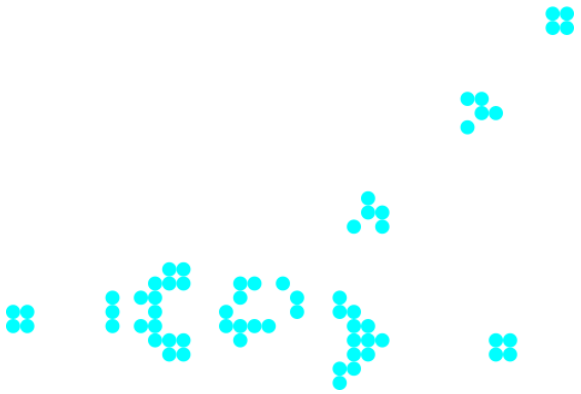

Figure 4: The program running a simulation of glider_gun.gol