CS5001 Practical 4 Report

Introduction

This practical is to help users to explore Mandelbrot Set by a Java program. This report will describe the design and implementation of this program. Firstly, the overall design will be discussed. Then, this report will focus on part of implementation and results.

Design and Implementation

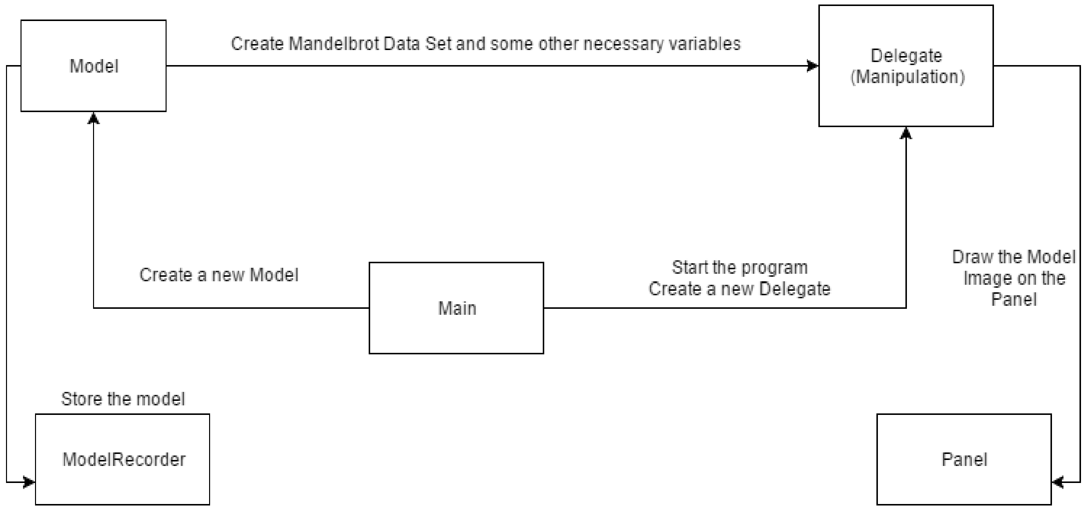


Figure 1 The Flowchart of CS5001-p4

Figure 1 is the flowchart of this program. Model-Delegate is selected to finish this task. JFrame is used as a fundamental. Components are added based on this JFrame. Basically, this program uses the structure of one code example on stures[1]. Functions of this program mainly are:

1. Show an initial model when the program starts to run with default values (X_Resolution is 1,000 and Y_Resolution is 800).
2. Update to have a new model with a new iteration times.
3. Zoom a model with a mouse.
4. Change colour for a model. To make the process a little interesting, random numbers are used to generate colours. Here the JPanel's size is set the same as that of JFrame. Thus, when one user is dragging the frame's border, the colours of the model will also change.
5. Undo, redo, and reset models. Here, "reset" only means back to the default model. History of models will not be cleared.
6. Save a picture in JPG format into current directory and notify results to the user.

There is one Object named Model used to initialise the first graph and do actions related to models (e.g. update one model, undo or redo). Every time a new model (excluding models generated by undo and redo), Object ModelRecorder with related information of the model will be added into an ArrayList<ModelRecorder>. This recorder is responsible to assist undo and redo methods.

Class Delegate has several components used to communicate with a user. It including five buttons, two textfileds, one panel and one menu button which contains two sub-buttons.

[1] The resource is: https://studres.cs.st-andrews.ac.uk/CS5001/Examples/L11-13_GUIs/CS5001_SimpleSwing_MDGuiExample/.

Class Panel is responsible to draw pictures regrading to Mandelbrot Models. In addition, it has a set of methods for zooming.

Results and Discussion

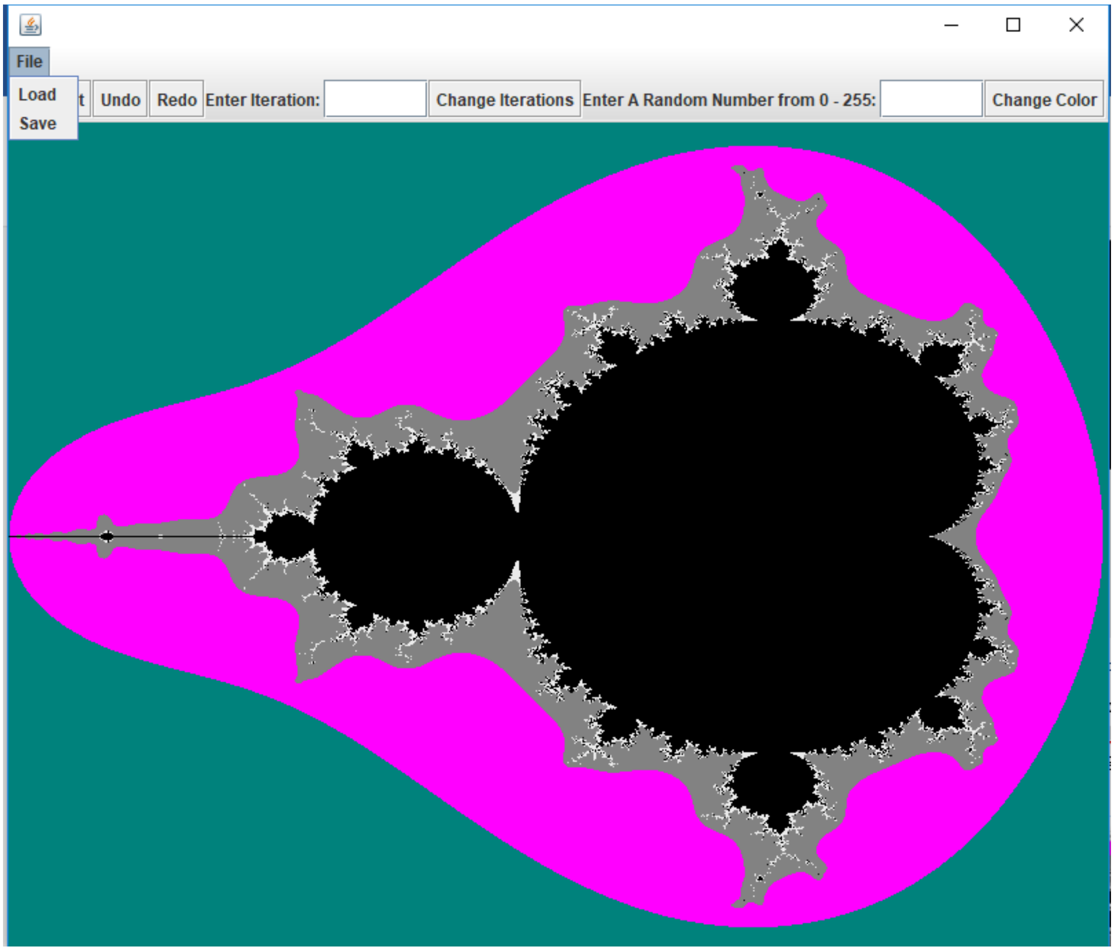


Figure 2 The Initial picture of this program

Figure 2 is the default picture of this program. The colour of this picture is not fixed. Thus, every time when a user run it, it will show various colours. Five buttons are added. Functions of Buttons “Reset”, “Undo”, and “Redo” are stated as their names. Button “Change iterations” is used for change the iterations of Mandelbrot Model. Button “change color” is used to change the parameter r in object Color. When r changes, g and b will change relatively.

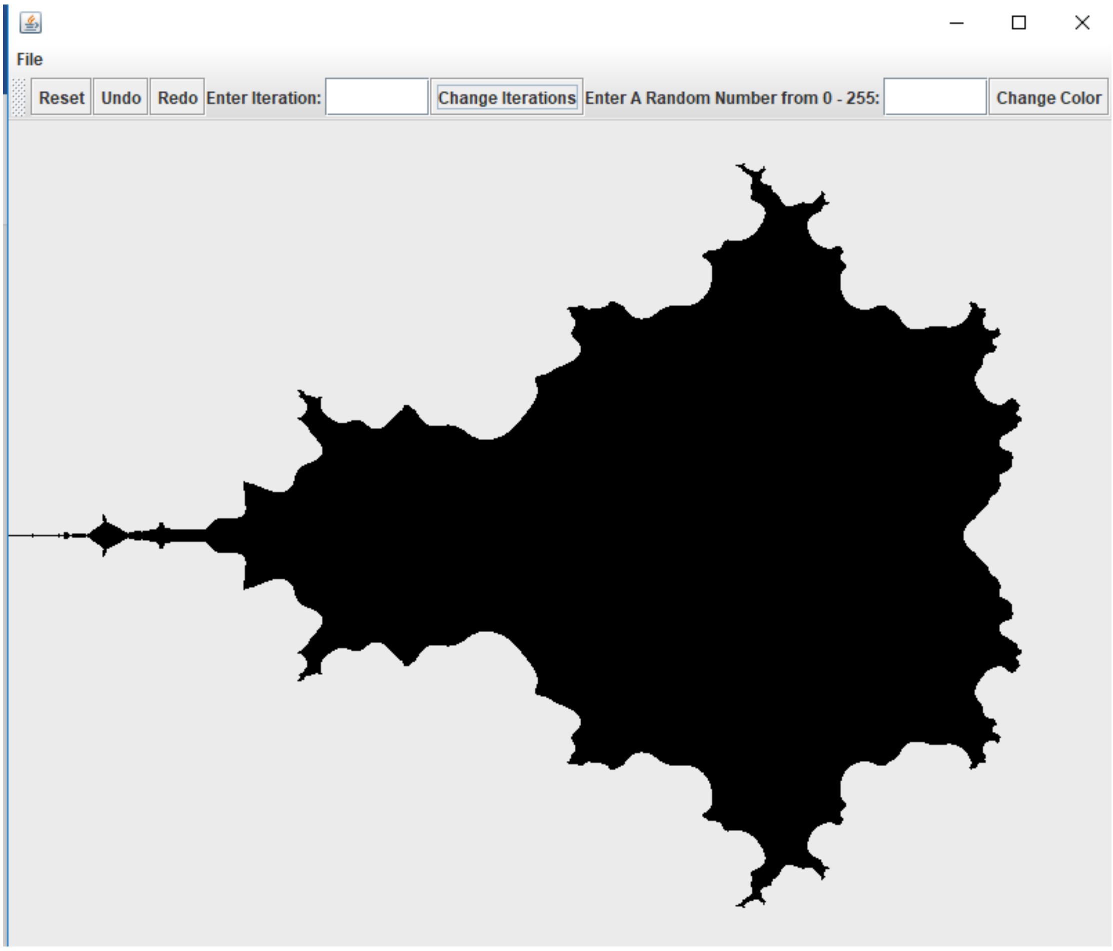


Figure 3 The Model Image when iteration is 11

Figure 3 is the image of a model whose iteration is 11. It can be seen that the colour changes even the user does not use Button “Change Color”. This is for fun but it is not very reasonable. If time permits, a choice of whether to change colour automatically should be provided to users.

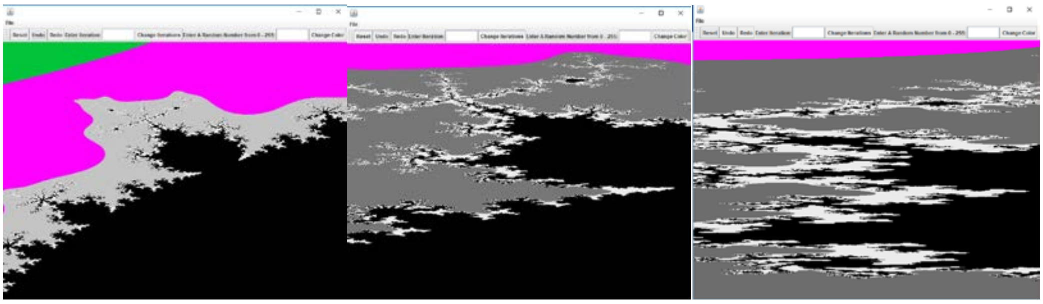
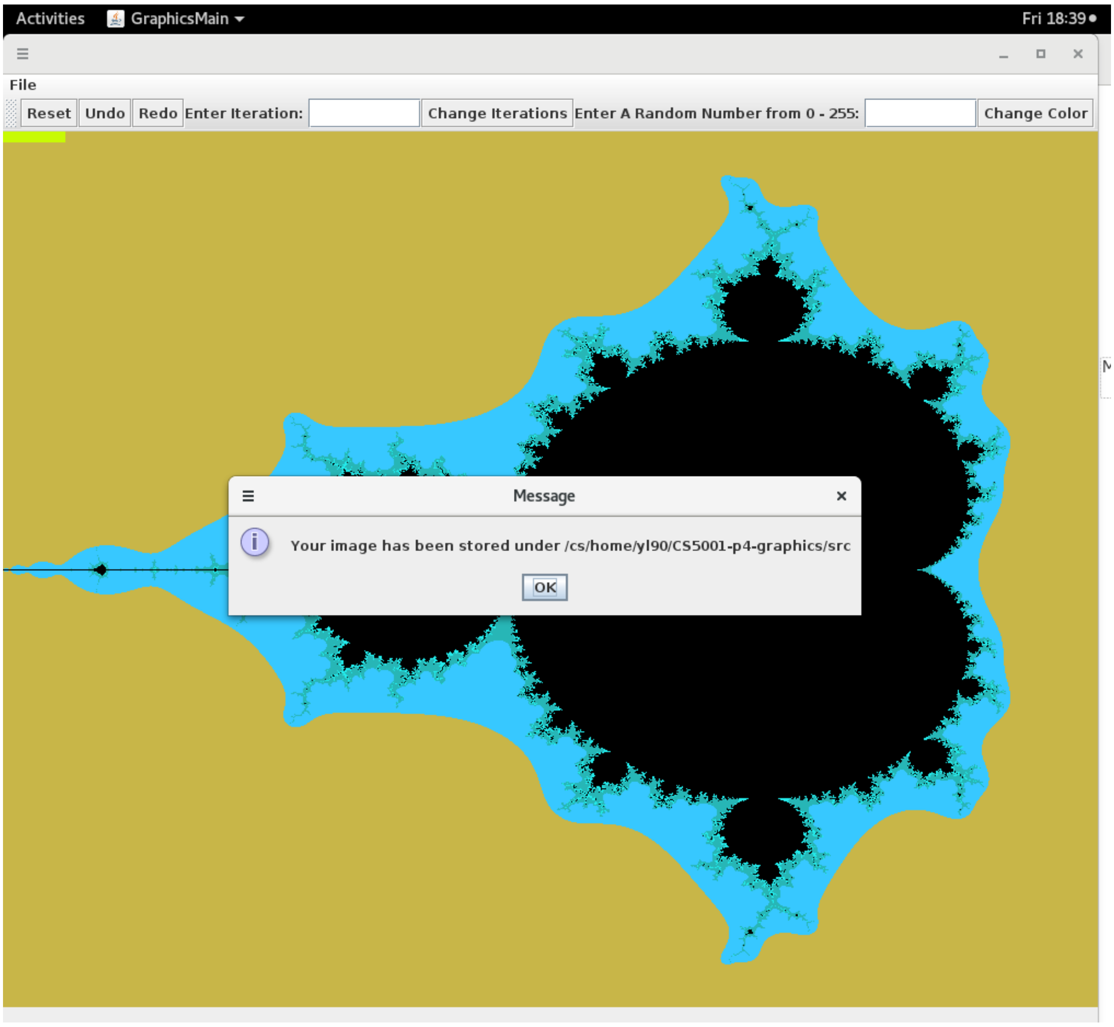


Figure 4 A set of pictures for the same model with different zooms

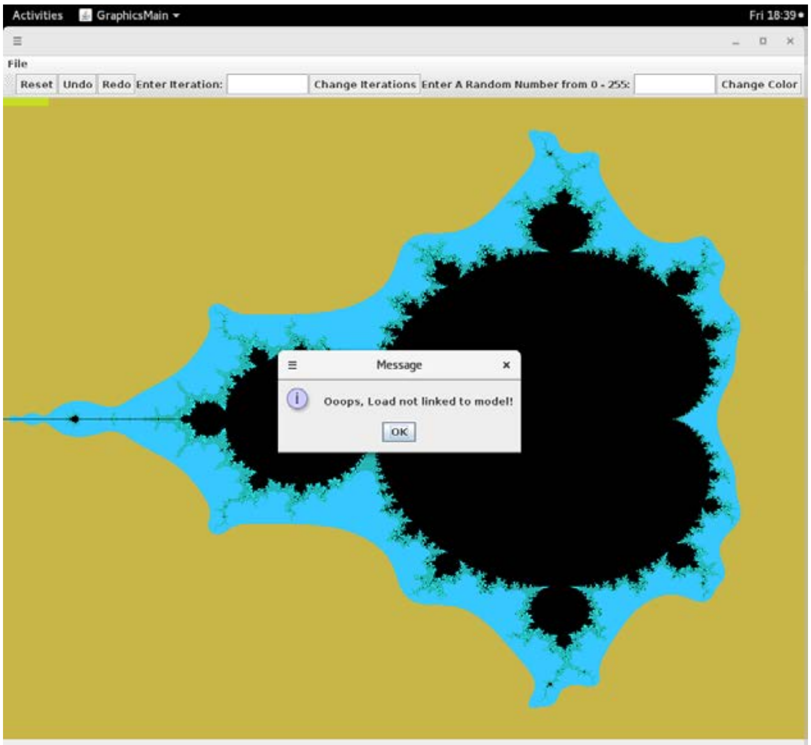
Figure 4 shows a model with zooming three times, five times and eight times respectively. When the time of zooming is up to about eight times, the picture is not very available anymore.



Your image has been stored under /cs/home/yl90/CS5001-p4-graphics/src

Figure 5 Save the current Model Picture

Figure 5 is a screenshot of using “Save” function. Figure 6 and Figure 7 show the rough design of “Load” function. Because time reason, this part is not available. Bugs exist.



Ooops, Load not linked to model!

Figure 6 Use “Load” function

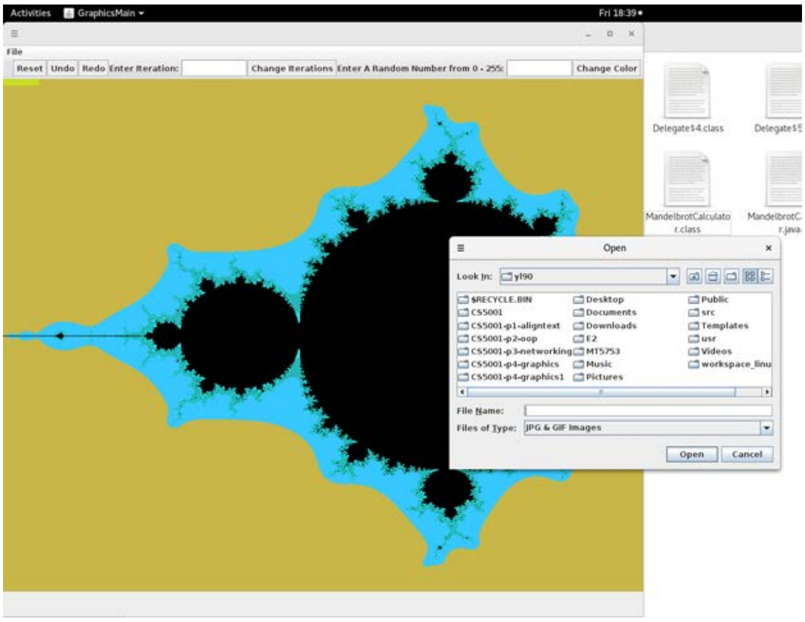


Figure 7 Use “Load” Function

Conclusion

In conclusion, I finished the basic part and a little of enhancement. This program can work but still not ideal. Details should be more precise and well-considered. For example, the colour of one model should be more friendly and can be controlled by users. In addition, stack should be used to undo and redo. It was not considered till the work is almost finished. This implies that I did not make enough preparations at beginning. In terms of testing, there is no code test for the program. I only tested it by click buttons and input different parameters manually. This part is not well-achieved. However, comparing to practical I have done before, I have been more familiar with OO programming. Comparing to C, it is more flexible. I realised a different programming thinking style should be applied when I use Java.