CAB302 Software Development

Assignment 2

Callum woodland n10067906

Joshua want n9723099

# Statement of Completeness

Below is the statement of completeness, features marked with an asterisk most likely have small bugs. These will be expanded upon below.

|  |  |
| --- | --- |
| **Feature** | **Completion** |
| Plot | Y |
| Line | Y |
| Rectangle | Y |
| Ellipse | Y |
| Polygon | N |
| Save | Y\* |
| Load | Y\* |
| Outline Colour | Y |
| Fill Colour | Y |
| Undo | Y |
| Clear | Y |
| Javadoc | Y |
| Junit 5 | Y |
| Git | Y |
| Additional Functionality | Y |

Save: bug related to disabling the fill colour of an object. It also does not save polygons as they are not implemented.

Load: cannot load polygons as they are not implemented.

# Development Process

During the development of this task Agile software development practices were used to ensure efficient and smooth development of features. An iterative design-code-test method was used, we focused on implementing features one at a time, constantly testing and adapting the code to allow for the best implementation of the feature. To maintain a reliable and even stream of work, a Trello board was used to pace and allocate work that was appropriate to the current stage of development. This board was fitted with deadlines for every major stage.

Features were implemented in the following stages:

1. GUI and drawing canvas
2. Basic draw functionality
3. Extend draw functionality to include ovals, colours
4. Colour selection for fill and outline
5. Save feature
6. Load feature
7. Undo/Clear

# Statement of Contribution

|  |  |
| --- | --- |
| **Person** | **Contribution** |
| Callum Woodland | Code writing, report, tests |
| Joshua Want | Code writing, report, tests |

# Software Architecture

Main – This class is used to iterate the PaintFrame Class. It is also used to generate Command Line Arguments.

PaintFrame – The paint frame is used to construct the main JFrame of the project as well as to define the starting parameters of this frame such as resizing, starting size and close operations. Finally, the Class iterates the PaintCanvas class.

PaintCanvas – This class contains a large portion of the program’s overall functionality. First, it is responsible for the implementation of the major UI elements such as buttons, dialogue boxes, and the canvas. Multiple event listeners such as action and mouse event listeners are present within this class to dictate functions of buttons and mouse clicks. It is also responsible for the collection of Shape data (Coordinates, colours, etc.). Finally, the PaintCanvas class contains an override of the PaintComponent class which allows the drawing of shapes stored within the ‘entries’ array list.

Shapes – This class is used as an abstract class to achieve polymorphism through the iteration of child ‘shapes’ The Shapes class itself is formed of multiple blank coordinates and colours. This information is then extended by shape child classes (Plot, Line, Rectangle, Oval, Polygon) to create the information relevant to a shape as seen by the program. This includes coordinate data for the placement and size of the shape, as well as pen and fill colours for the external and internal colouring respectively.

Load – The load class is responsible for the creation of a file choosing dialogue box which users can use to select a .vec file for importation into the program.

GetHex – this class relies on encapsulation to create, set and get a hexadecimal code based on the colour chosen by the user.

# Implementation of Object Orientated Programming

## Encapsulation

Encapsulation is present within the GetHex class and its methods. This class is used to populate a series of variables with information and return them to another class. This is accomplished with a pair of getter and setter methods.

## Inheritance

Inheritance is present within the many shape classes (Plot, Line, Square, etc.). These classes inherit information from their superior class and then finally displayed this information as coordinates and other relevant forms within the ‘entries’ array list.

## Polymorphism

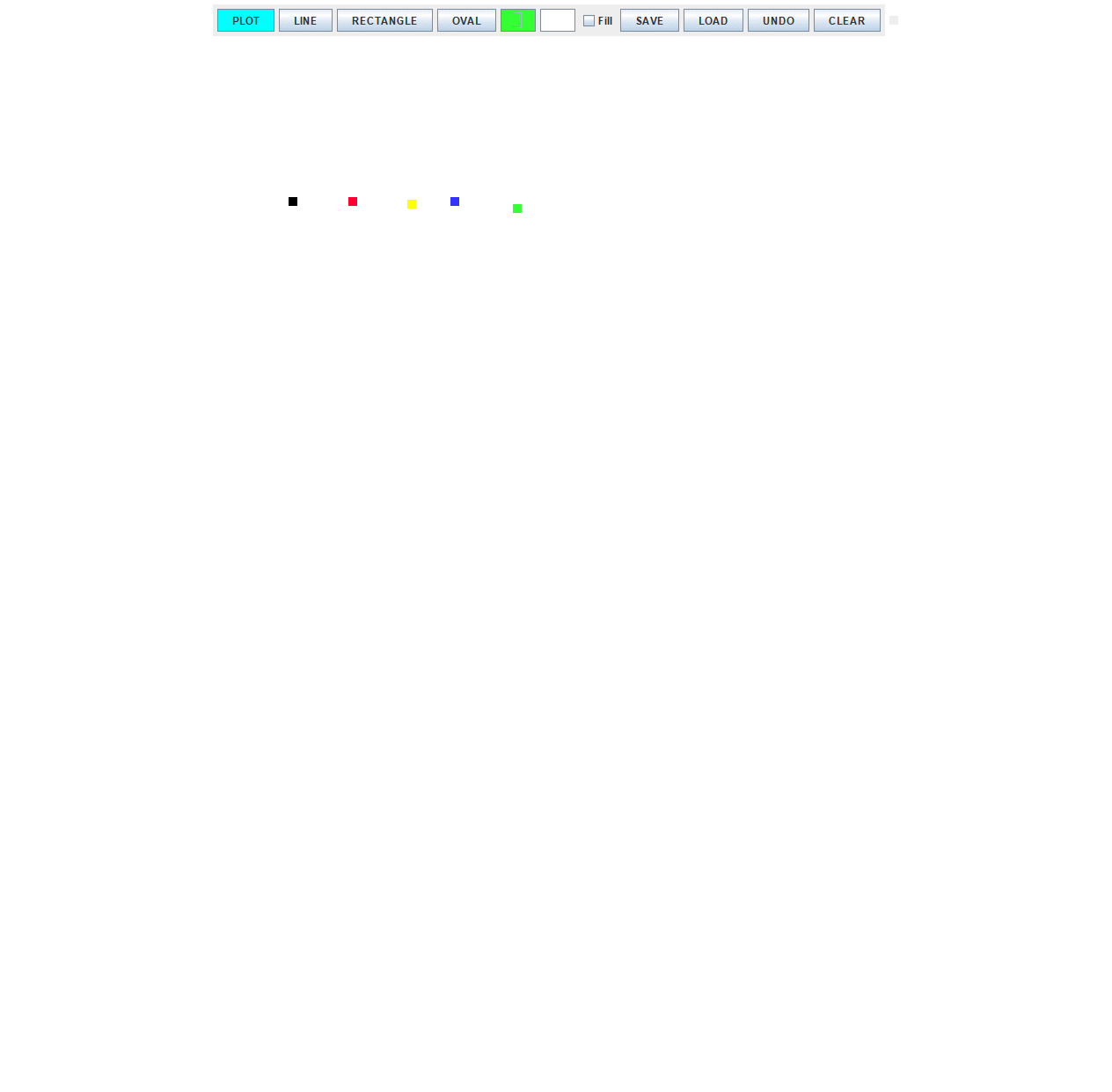
Polymorphism is the method on which the majority of the program is based. The Shapes class relies upon polymorphism to create an abstract class full of undefined variables that can be filled in any way by the child classes mentioned above.

# Usage Documentation

1. Open project in IntelliJ
2. Set entry point to main class
3. Run
4. All drawing functions are used by using a click-hold-release method 
5. Line and fill colour can be set with the 2 colour boxes in the centre of the GUI bar 
6. Save will save the current drawing to the root directory of the project 
7. Load will allow you to load .vec files from anywhere 
8. Undo/Clear will remove the most recent drawing or entire frame respectively 

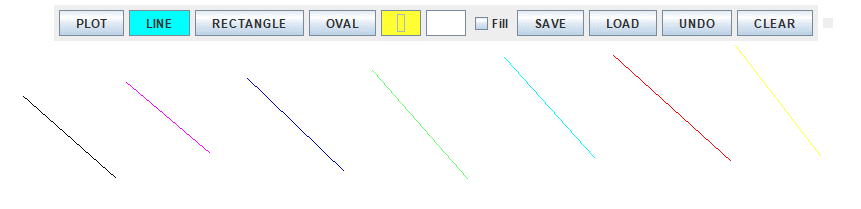
If you are unable to run the project you may need to mark the src folder as the source folder

## Functionality showcase

**Plot with multiple colours**Click the PLOT button. Click where you desire to leave a plot.

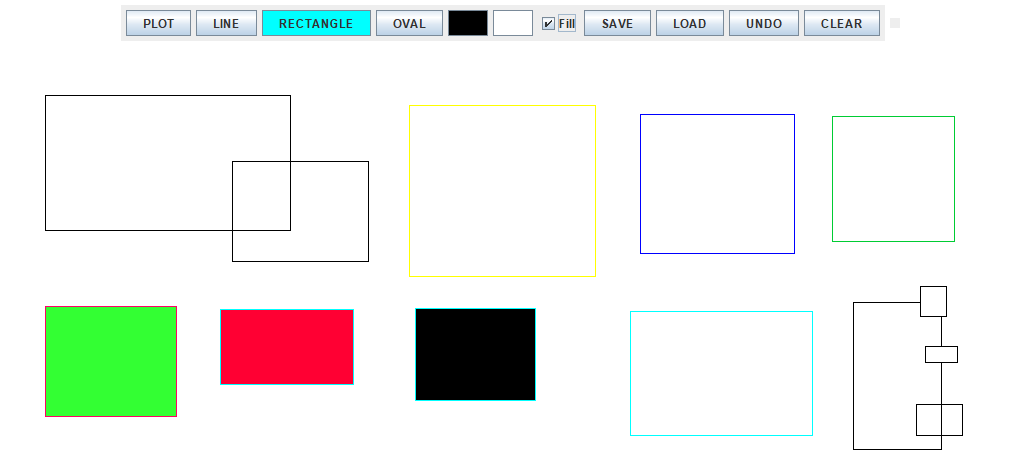
**Line with multiple colours**

Click the LINE button, click and drag the mouse on the canvas. Change colours as desired.



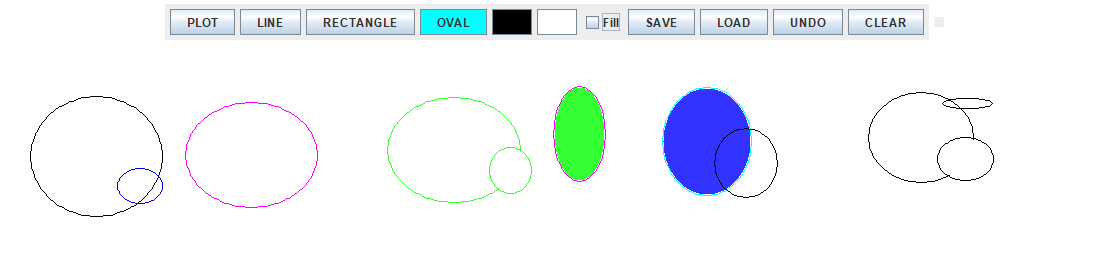
**Rectangle with various fill and line colours**

Click the RECTANGLE button, click and drag the mouse on the canvas. Change colours as desired.



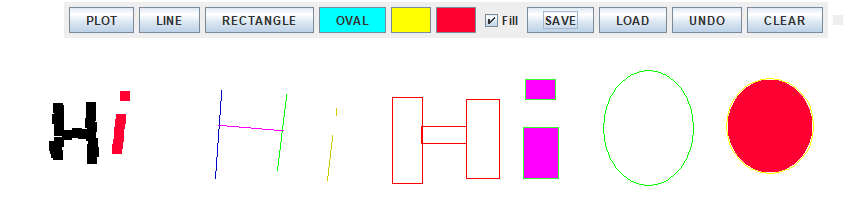
**Ellipse with various fill and line colours**

Click the OVAL button, click and drag the mouse on the canvas. Change colours as desired.

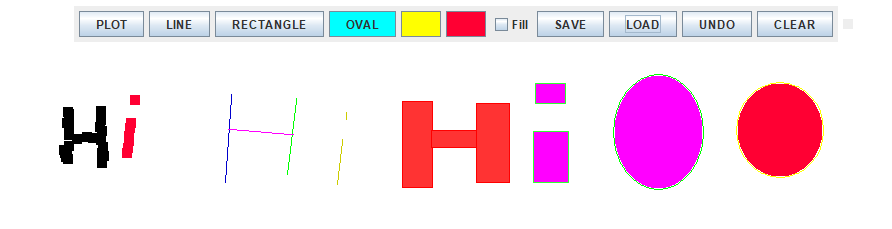


**Saved the following drawing**

Draw any desired picture, click the SAVE button. The save can be found in the project folder.

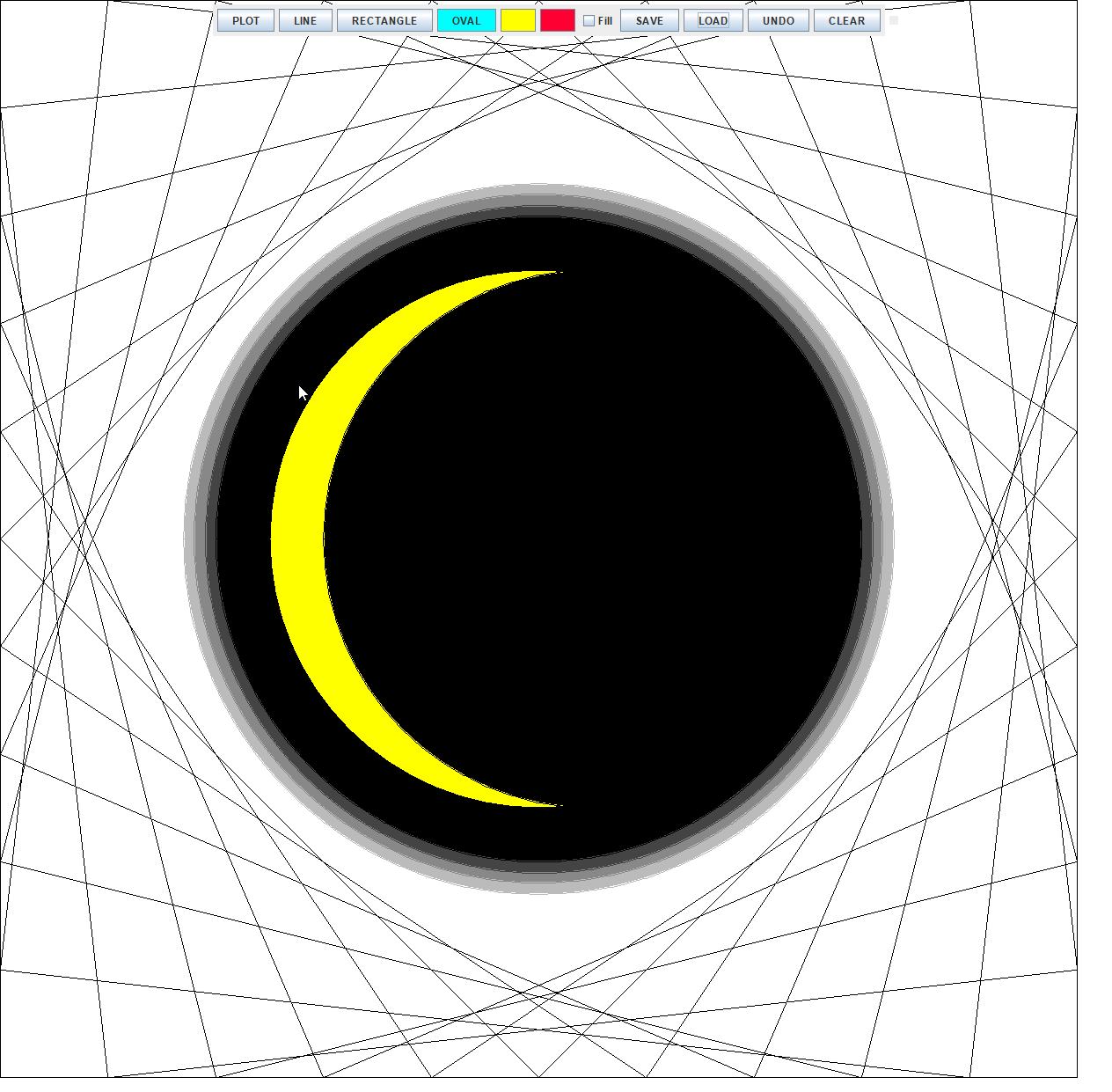


Loaded the generated VEC. It looks a little different cause the save functionality has a bug with fill colours.



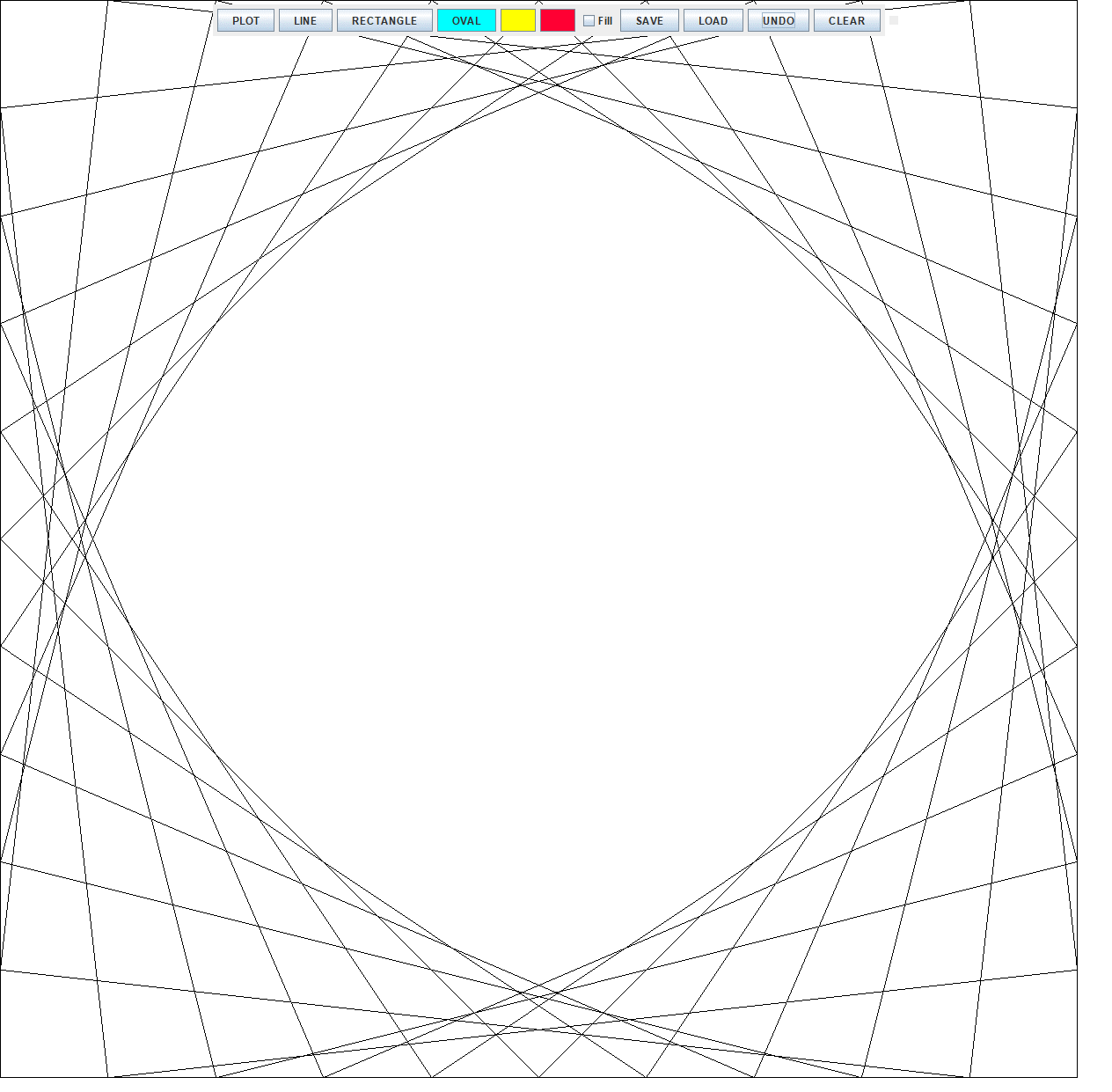
**Load example1.vec**

Click the LOAD button on the UI



**Undo:**

Click undo a desired number of times on the loaded example1.vec or any drawn picture.



**Grid view**:

Click the GRID button, the grid will appear after the first shape is drawn. This shape will snap to a coordinate rounded to 10 pixels (Usually with a left bias).

