      Even Semester (2023)

**A picture containing text, screenshot, graphics, graphic design

Description automatically generated**

Assignment Cover Letter

(Individual Assignment)

|  |  |
| --- | --- |
| **Student Names:**  Kenneth Lay | **Student IDs:**  2602119272 |
| **Course Code:** COMP6699001 | **Course Name:**  Object Oriented Programming |
| **Class:** L2AC | **Name of Lecturer:**  Jude Joseph Lamug Martinez, MCS |
| **Major:** Computer Science |  |
| **Title of Assignment:**  Pixel Art Drawer |  |
| **Type of Assignment:** Final Project |  |
| **Due Date:**   16 - 06 - 2023 |  |

TABLE OF CONTENTS

1. **Specification**
2. **Background**

* There was actually a long process on deciding on what my final project will be. At first I wanted to make a diversity map simulation which can simulate the entities spawn within the frame or plane and map it out with color by coordinates than give out a statement on the bottom of the program that states the legend depending on the color, like let’s say green = high diversity index maybe a 65%, red is low diversity index which is like 10% and so on. But then there seemed to be a change of mind, since I deemed it too difficult and so I change to a banking system, since a few subject likes to make banking system as practice work. Like a more advanced banking system with UI, database and so on, but I had troubles making the UI and so I tried to go back to plan no.1 but it was just too difficult tried out other projects like platformer, tarot card reader, gacha system, and so on but to no avail as I deemed them incomplete and so I turned back to plan 1 and 2, but again I changed my mind ended up making a tower defense game, but there was too much bugs, and I had so little times left ,so out of desperation I stayed up all night and deconstructed a part of the tower defense game to create a simple pixel art painter. Version 1 could drag but grid didn’t work, version 2 had grids but when using drag option there was a slight displacement tried to adjust displacement didn’t work so I had to at the end completely remove it because I was too exhausted at that point in time. And so at the final version or should I say current version there is a pixel art painter with grids, there is color palette you can select colors from, in the current version I believe there is only 11 colors and you have to click the tiles one by one to color it.

1. **Description**

* The project I made for my final object-oriented programming project is a pixel art painter, the pixel art painter is a program that can be used to make pixel arts as stated by the name. User can pick a certain color from the color palette at the bottom of the frame. After picking a color they can press on the canvas grid tiles and color it, you can only color the tile by clicking only. They can save their finish pixel art drawings and load it later on, they can also resize the canvas size and how big the box is, for the canvas size it can only be a square so if the input is 64 it will make a 64 by 64 grid.

1. Class Diagram

A picture containing text, diagram, line, technical drawing

Description automatically generated

1. Module / Library

There is 4 modules or libraries were used for this program which is:

1. Java Swing

* Java swing is a graphical user interface toolkit that is a part of Java Foundation Classes, it allows developers to create an interactive and visually appealing desktop application in Java. It provides a lot of tools and components for building a GUI such as buttons check boxes and etc. These are platform independent components. It is useful to make desktop apps since it is lightweight and very customizable.

1. Java Awt

* Java Awt is a set of classes and APIs but it’s also a GUI toolkit and older than java swing. Java Awt contains lots of tools and components for making GUI, they are like java swing also platform independent. I use java awt to build the programs layout and other things. The benefits of using java awt is that it can be use for event driven programming, light weight and simple but unlike swing it’s limited in customization.

1. Java IO

* Java IO or shot Java Input Output, JavaIO is a module/library that contains a collection of classes and interfaces in Java SE that provides a excellent and flexible framework for doing input and output operations in java, Java IO allows developers to read and write to a variety of data sources.

1. Java ImageIO

* Java ImageIO is a module/library that contains a variety of classes and APIs provided by Java platform. It’s a framework that is used for reading, writing and editing images in various formats. I use this to load the image, save the image and etc.

1. **Implementation**
2. **Pixel Art App Class**



The pixel Art app is where the JFrame exist and where everything will be set up and loaded into the JFrame, this class also contains the main

a). Public PixelArtApp



- this is pretty much where everything is setup, so it’s more of a constructor class.

A screenshot of a computer program

Description automatically generated with medium confidence

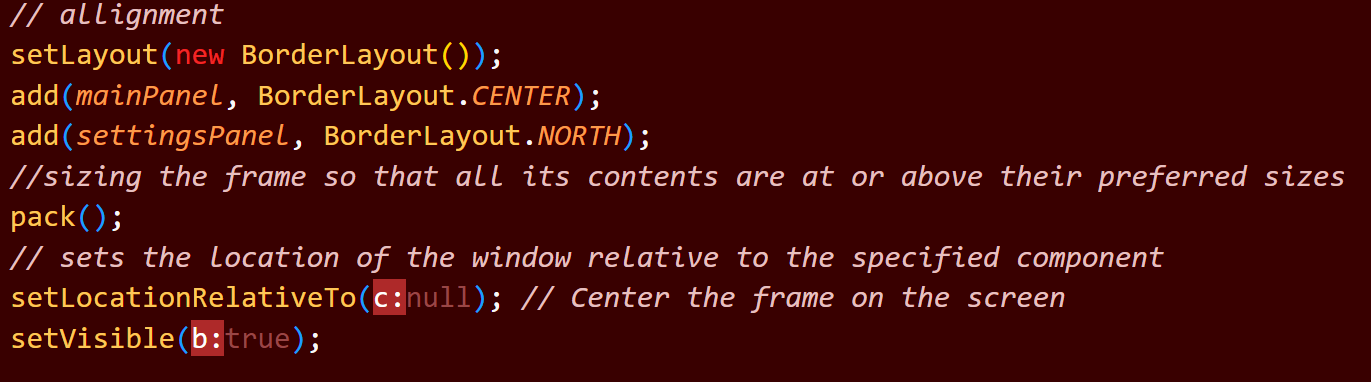
* The code above shows the initializing and setting up of the Frame
* And the panels too
* Pixel Canvas size is set

A screen shot of a computer program

Description automatically generated with low confidence

- The code above just creates the button and their functions

- The 3 buttons that is being created and loaded is the save, load and resize buttons



- Sizing the frame and aligning and mapping the panels.

b). Main

A screen shot of a computer code

Description automatically generated with low confidence

- the main class implements the method SwingUtilities.invokeLater which schedules the creation of the GUI on the event dispatch thread or it can be shorten to EDT. If I don’t use this there might be a problem with the GUI when I run it

- new Runnable is a inner class that implements Runnable interface, which is used to define the task that will be execute on the EDT. The task that will be executed is creating an instance of PixelArtApp.

- run() method code will execute when the task is executed on the EDT

- new PixelArtApp is self-explanatory, it created a new instance of PixelArtApp

1. Pixel Canvas Class

A screen shot of a computer code

Description automatically generated with low confidence

- Pixel Canvas Default and Attributes

a.) PixelCanvas



Constructor class

Where the canvas is created

b.) getCanvas

A picture containing text, font, screenshot, graphics

Description automatically generated

Getter class that returns the canvas

c.) setSelectedColor

A picture containing text, font, screenshot

Description automatically generated

Set the current color to the selected color from the color palette

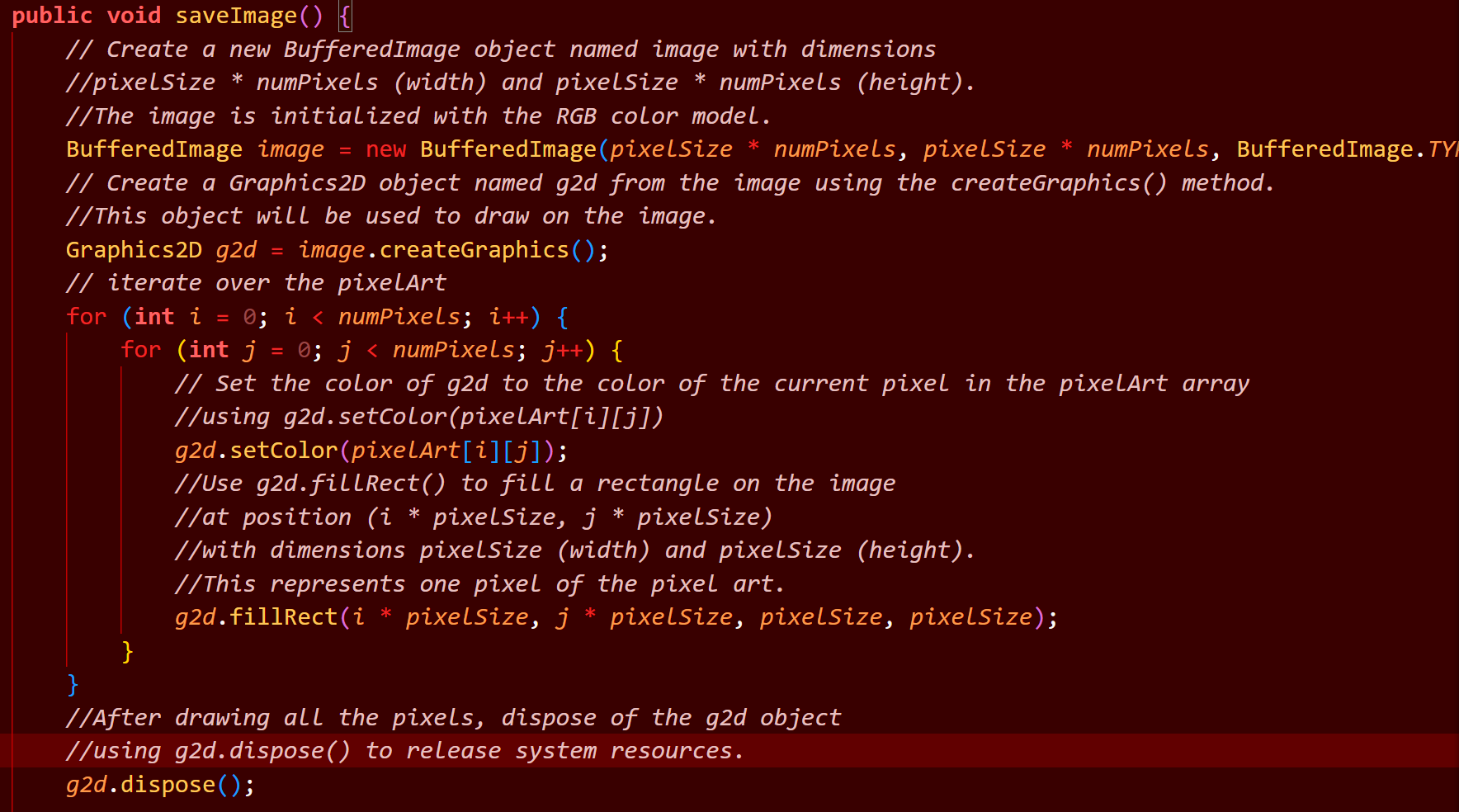
d.) resizeCanvas

A picture containing text, screenshot, font

Description automatically generated

* The method used to resize the canvas.
* This method will however reset your drawing if you do resize canvas when you have a drawing still there. As it pretty much just re create a new canvas with different box size and overall size.

e.) saveImage



A screenshot of a computer program

Description automatically generated with medium confidence

* The saveImage class contains the method to save image well it supposed to save as png but it doesn’t and if you try to save as png the file type will be png but it’s actually a jpeg

f.) loadImage

A screenshot of a computer program

Description automatically generated with medium confidence

* Using Jfilechooser to load the file and read the image after it reads the image it will try to draw the image into the canvas

g.) Setters

A screen shot of a computer code

Description automatically generated with low confidence

The rest are just some basic setters.

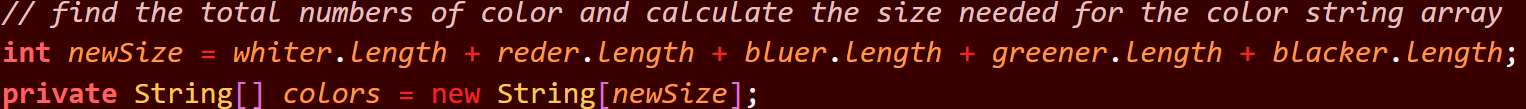
1. Color Palette Class



* This class is where the color chooser stuff I contained



* This part is initializing the colors for the color palette and adding the attributes



* The code above is used to figure out the new size of the array for the loop that is going to happen for the creation of each colored buttons in the color palette.

1. ColorPalette Constructor

A screen shot of a computer code

Description automatically generated with low confidence

* In this part we just loop through the colors and append them into the new string array
* After it is added to the new string array called colors
* The string array colors will go through a loop where the button for each color in the colors array is created its functionality and its entirety.

1. getPalette

A picture containing text, font, screenshot, graphics

Description automatically generated

* getter class to get the palette.

1. All Color class.

A picture containing text, font, screenshot, line

Description automatically generated

* All color class are made up of this abstract class.
* The colors classes contains an string array that contains the hex codes of the specified color that is going to be added to the palette

1. Evidence

A picture containing text, screenshot, plot, line

Description automatically generated

The picture above is of the working program

Github Link: <https://github.com/KELASU/OOPFInalsC/tree/main/PixelArtV2>

Video Link: <https://drive.google.com/drive/folders/11pf5lTfQSgeA9LqkCuwo-3-Cs0J9lJ2_?usp=sharing>