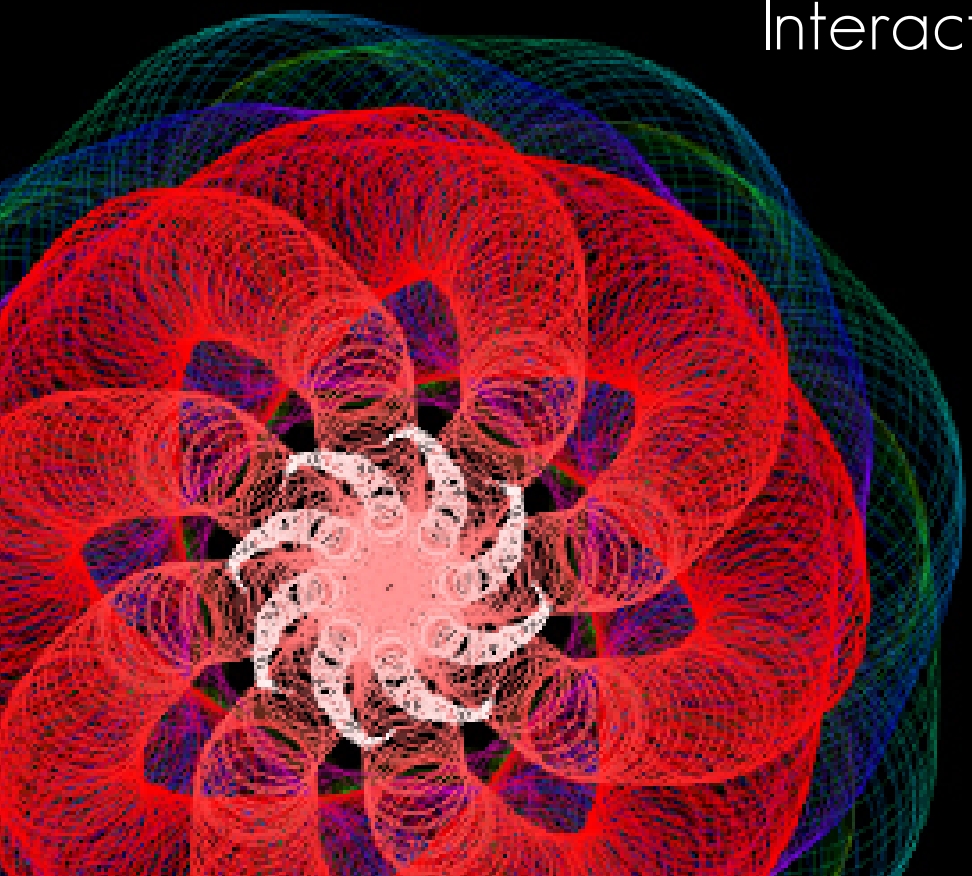
Computational Prototyping
Interactive Drawing tool

Mitchell Hunter
s34899309



Design intention

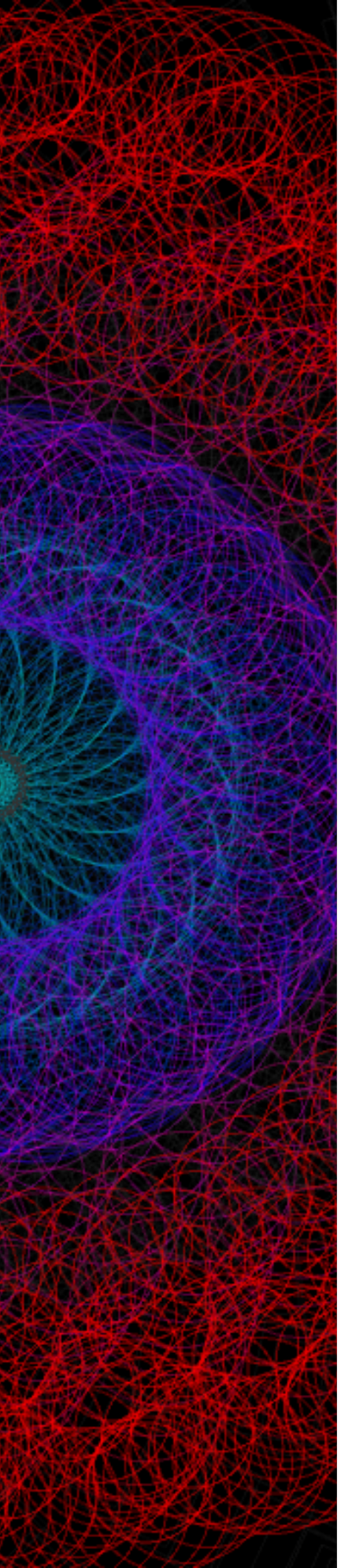
For this interactive drawing tool, I tried to recreate the repetitious geometric visuals of a kaleidoscope. The drawing tool has 3 separate drawing functions;

A classic spirograph inspired drawing sketch,

A sketch that creates enlarging circles that change neon colours,

A sketch which creates rotating growing squares,

The drawing tool allows for real-time interaction using the mouse and keyboard buttons. The user can control variables, save screenshots of the drawing and clear the sketch and change the sketch.



Main Sketch

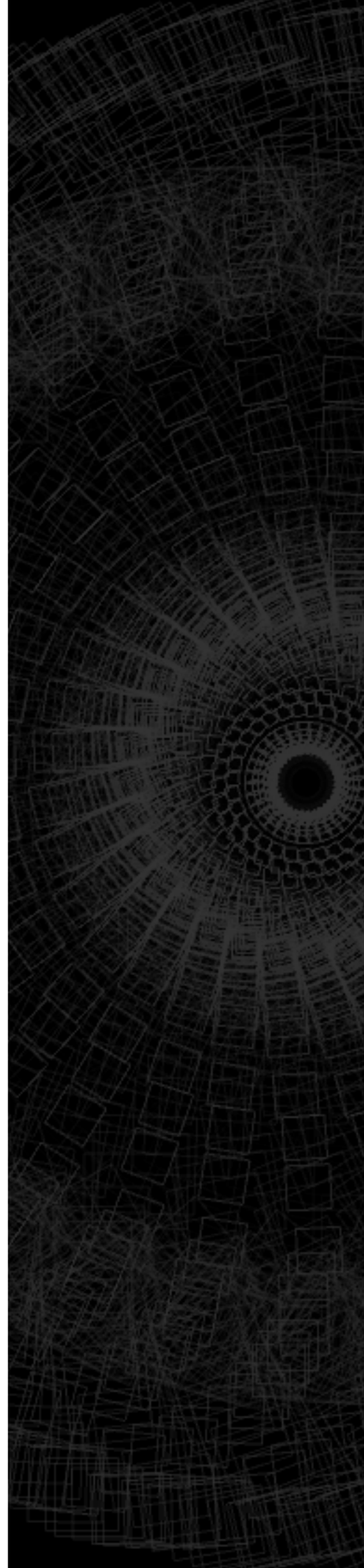
```
//-----here all the classes are defined-----
SplashScreen ss;
Button b1;
Button b2;
Button b3;
Button bs;
Button bc;
Button bp;
Button bm;

int slices = 8; //this variable controls how many time the drawing is repeated
int pattern = 2; //this variavble controls which drawing tool is being used

//-----here all the arraylists for all the sketches are created-----
ArrayList<Circle> circles = new ArrayList<Circle>();
ArrayList <StaticCircle> curves = new ArrayList<StaticCircle>();
ArrayList<SpinningSquare> square = new ArrayList<SpinningSquare>();
ArrayList<CustomBrush> stars = new ArrayList<CustomBrush>();

boolean start = false; //a boolean is defined to start the sketch

//-----here the custom fonts are defined-----
PFont font;
PFont font1;
PFont font2;

PImage img; //here a custom image is defined

void setup() {
  fullScreen(P3D); //setting the sketch resolution to fullscreen
  colorMode(HSB, 100, 100, 100); //changes the colour to Hue, Saturation and Brightness
  background (0); //setting the background as black


  img = loadImage("Star.png"); //here the custom image is loaded from the data folder

  font = createFont("GeosansLight", 22); //here three instances of a custom font are created
  font1 = createFont("GeosansLight", 15);
  font2 = createFont("GeosansLight", 10);

  ss = new SplashScreen(); //the Welcome screen is setup so it can be called
  b1 = new Button(30, height-195, #151515, '1'); //here all the buttons are defined
  b2 = new Button(30, height-170, #151515, '2'); //the variables are the starting point in the x axis,
  b3 = new Button(30, height-145, #151515, '3'); //followed by the startingn point in the y axis,
  bs = new Button(30, height-120, #151515, 'S'); // then the colour of the button,
  bc = new Button(30, height-95, #151515, 'C');  // then finally a letter identifier
  bp = new Button(30, height-70, #151515, 'P');
  bm = new Button(30, height-45, #151515, 'M');
}

void draw() {

  if (start == false) { // here the if statement checks if the sketch has been
    ss.splash();          // initialised, as it hasnt been the splash screen is loaded
  } else if (start == true) { //once ENTER has been pressed
    background (0);       // the  start main sketch is started and the buttons are drawn
    b1.buttonDraw();
    b2.buttonDraw();
    b3.buttonDraw();
    bs.buttonDraw();
    bc.buttonDraw();
    bp.buttonDraw();
    bm.buttonDraw();

    // check if the mouse is hovering over buttons----------
    b1.checkClick();
    b2.checkClick();
    b3.checkClick();
    bs.checkClick();
    bc.checkClick();
    bp.checkClick();
    bm.checkClick();

    pattern1(); //each of the drawing tools are called
    pattern2();
    pattern3();
    pattern4();
  }
}

void mousePressed() {

  if (b1.selected == true) { //clicking on the button changes to pattern 1
    pattern = 1;
  }
  if (b2.selected == true) { //clicking on the button changes to pattern 2
    pattern = 2;
  }
  if (b3.selected == true) { //clicking on the button changes to pattern 3
    pattern = 3;
  }
  if (bs.selected == true) {  //clicking on the save button saves an image
    saveFrame("line-######.png");
  }
  if (bc.selected == true) { //clears all the data currently on screen
    circles.clear();
    background(0);
  }
  if (bp.selected == true) { //increases the number of slices
    slices++;
  }
  if (bm.selected == true) { //decreases the number of slices
    slices--;
  }
}

void mouseDragged() {
  if (pattern == 1) { //if the mouse is dragged and pattern one is selected a new curve is added to
the array
    curves.add(new StaticCircle(mouseX, mouseY));
  }
  if (pattern == 2) { //if the mouse is dragged and pattern two is selected a new circle is added to
the array
    circles.add(new Circle(mouseX, mouseY));
```

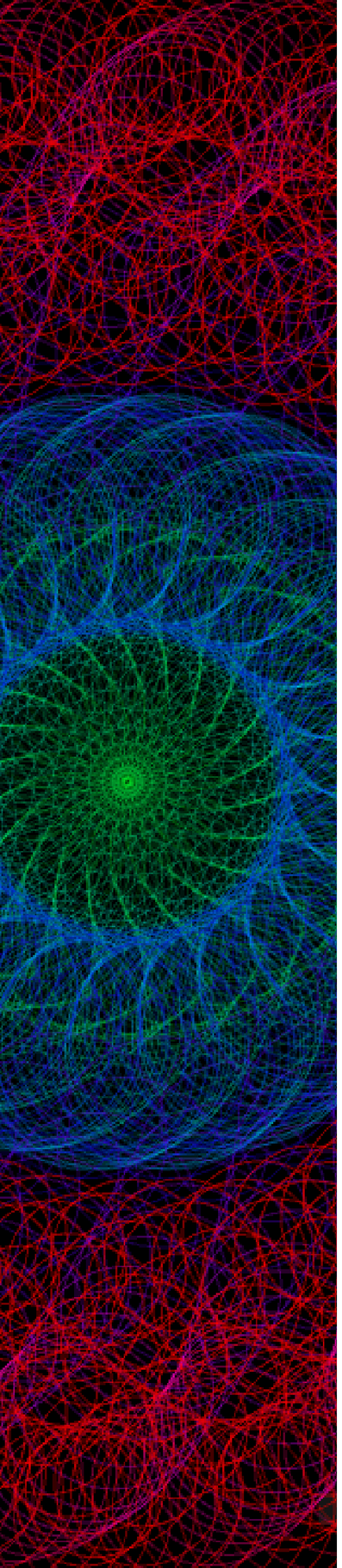
Main Sketch

```
  }
  if (pattern == 3) { //if the mouse is dragged and pattern one is selected a new square is added
to the array
    square.add(new SpinningSquare(mouseX, mouseY));
  }
}
void keyPressed() {
  if (key == ENTER) {  //transitions from splash screen to main drawing
       start = true;
  }
  if (key=='s'||key=='S') { // Saves a screenshot of current screen
       saveFrame("line-######.png");
  }
  if (key=='c'||key=='C') { //Clears all the data from the arrays on screen
   circles.clear();
    curves.clear();
    square.clear();
    stars.clear();
  }
  if (key=='1') { // Changes pattern to 1
    pattern = 1;
  }
  if (key=='2') { // Changes pattern to 2
    pattern = 2;
  }
  if (key=='3') { // Changes pattern to 3
    pattern = 3;
  }
  if (key=='4') { // Changes pattern to 4
    pattern = 4;
  }
  if (key=='-') { // Increases the number of slices in the drawing
       slices--;
  }
  if (key=='+') { // Decreases the number of slices in the drawing
    slices++;
  }
}

void pattern1() {
  for (int j = 0; j < curves.size(); j++) {  //here the array is defined
    StaticCircle staticcircle = curves.get(j);
    println(staticcircle.x);
    for (int i = 0; i<slices; i++) {
      pushMatrix(); //here the sketch plane is released
      translate(width/2, height/2); //and moved so that 0,0 is now in the center of the screen
      rotate(radians(i*360/slices)); //the for loop then divides a full rotation in to the amount of and
copys the curve around for each segment

      staticcircle.draw(); //here the sketch is drawn
      popMatrix(); //and the sketch planeis put back into place ready for the next step of the loop
    }
  }
}


void pattern2() { //here the same thing is completed but for the 2nd pattern
```

Main Sketch

```
  for (int j = 0; j < circles.size(); j++) {
    Circle circle = circles.get(j);
    circle.update();

    for (int i = 0; i<slices; i++) {
      pushMatrix();
      translate(width/2, height/2);
      rotate(radians(i*360/slices));

      circle.draw();
      popMatrix();
    }
  }
}

void pattern3() { //here the same thing is completed but for the 3rd pattern
  for (int j = 0; j < square.size(); j++) {
    SpinningSquare spinningsquare = square.get(j);
    spinningsquare.update();

    for (int i = 0; i<slices; i++) {
      pushMatrix();
      translate(width/2, height/2);
      rotate(radians(i*360/slices));

      spinningsquare.draw();
      popMatrix();
    }
  }
}

void pattern4() { //here the same thing is completed but for the 4th sketch
  for (int j = 0; j < stars.size(); j++) {
    CustomBrush custombrush = stars.get(j);
    for (int i = 0; i<100; i++) {

      custombrush.draw();
    }
  }
}
```





Button Class

```
class Button {
  int x, y;   //variables are created
  int buttonW = 15;
  int buttonH = 15;
  color colour;
  boolean selected = false;
  char button;

  //create a constructor for button positioning, colour and identifier//
  Button(int x, int _y, color colour, char button)
  {
    x = x;
    y = _y;
    colour = colour;
    button = button;
  }

  void checkClick() {
    //Here its checking if the mouse is inside the button
    if (mouseX >= x && mouseX < x+buttonW &&
      mouseY >= y && mouseY < y+buttonH) {  //if the mouse is within these parameters
      stroke(colour); //the square is highlighted
      selected = true; //then the boolean is true
    } else {
      selected = false; //otherwise the boolean is false
    }
  }

  void buttonDraw() {
    //Customisating the buttons
    fill(colour); //selecting the button colour
    if (selected) {
      stroke(#c3c3c3); //creating a special outline if the mouse is on the button
      fill(#c3c3c3); //creating the colour for the text
      text(button, x+20, y+12); //creating the location for the text
      fill(#000000); //to keep the inside of the box dark
    } else {
      stroke(colour); //if the button isnt moused over keep he outline the same color as the box
    }
    rect(x, y, buttonW, buttonH); //create the button
  }
  boolean overButton(int x, int y, int buttonW, int buttonH) { //here a boolean is created in a similar way
    if (mouseX >= x && mouseX <= x+buttonW &&                  //to the mouse over function
      mouseY >= y && mouseY <= y+buttonH) {                    //so that the button can create a desired
action
      return true;
    } else {
      return false;
    }
  }
}
```

SplashScreen Class

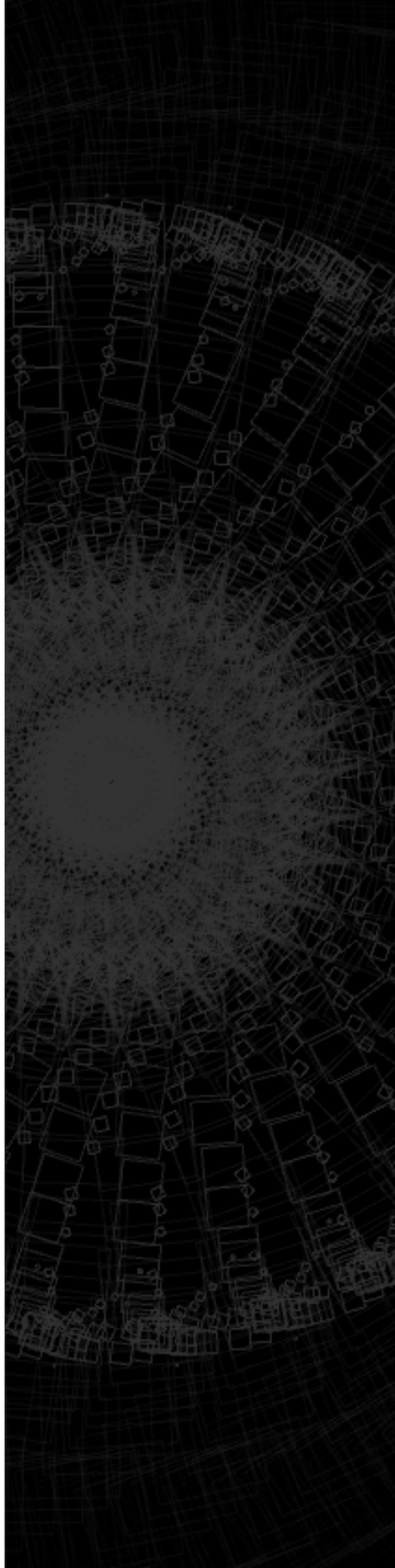
```
class SplashScreen //class is created
{

  int t = 50; //a vaiable is created to we the text should start on the x axis

  void splash() {
    background(0); //background is seet to black
    fill(#6f6f6f); // the colour of the text
    textFont(font); //the custom font is chosen
    textAlign(CENTER); //and aligned to the center for the header
    text("press ENTER to begin", width/2, height/2); //text is input followed by its x and y position
    textAlign(LEFT); //text alignment is changed to left
    textFont(font1); //and the second fond is chosen
    text("click or press 1 for simple curves", t, height-183); //as before the text is chosen
    text("click or press 2 for explading curves", t, height-157); //and so is its x and y position
    text("click or press 3 for _", t, height-133);
    text("click or press S to save a screenshot", t, height-107);
    text("click or press C to clear the canvas", t, height-83);
    text("click or press + or P to make more segments", t, height-57);
    text("click or press - or M to make fewer sections", t, height-33);
    b1.buttonDraw();
    b2.buttonDraw();  //buttons are drawn for illustrative purposes
    b3.buttonDraw();
    bs.buttonDraw();
    bc.buttonDraw();
    bp.buttonDraw();
    bm.buttonDraw();

  }
}
```

SpinningSquare Class

```
class SpinningSquare{
  int x;  //vairables for x and y position are creater
  int y;
  int radius;  //variable to radius is created
  int maxsize = 75; //max radius of the square is created and set
  int angle;  //variable for angle is also created

  // Constructor
  SpinningSquare(int x, int _y){
    x = x-width/2;  //mouse x position is adjuested for the sketch plane to be moved
    y = _y-height/2;  //as is the mouse y position
    radius = 0;  //initial radius is set to 0
    angle = 0;  //initial angle is set to 0
  }

  void update(){
    radius++;  //ever frame 1 is added to the radius
    angle++;   //and the angle
  }

  void draw(){
    noFill(); //fill colour is removed
    int alpha = (int)map(radius%maxsize,0,maxsize,255,0); //the radius of the square is mapped from
0-150 to 0-255
    stroke(alpha+50,alpha-255, 20, alpha); //then used to create changing colour for the stroke
    rectMode(CENTER); //rectangle is set to the center of the mouse position
    rotate((TAU/360)*radius); //the radius of the circle is then divided by 360 degrees then multiplied by
current radius
    rect(x,y,radius%maxsize,radius%maxsize); // the rectangle is drawn

  }
}
```

StaticCircle Class

```
class StaticCircle { //class is created

  int x; //variables for the x position, y position, height and width of the ellipse
  int y; //are created
  int h;
  int w;


  StaticCircle(int x, int _y) { //a constructor is created for the static circles
    x = x-width/2; //mouse x position adjusted for the moving of the sketch plane in the
    y = _y-height/2;  //main sketch
  }

  void draw() {
    stroke(#FFFFFF); //the stroke is set to white
    fill(#FFFFFF);  //and so is the fill
    ellipse(x, y, 2, 2);  //the draw function then draws the ellipse when called
  }
}
```

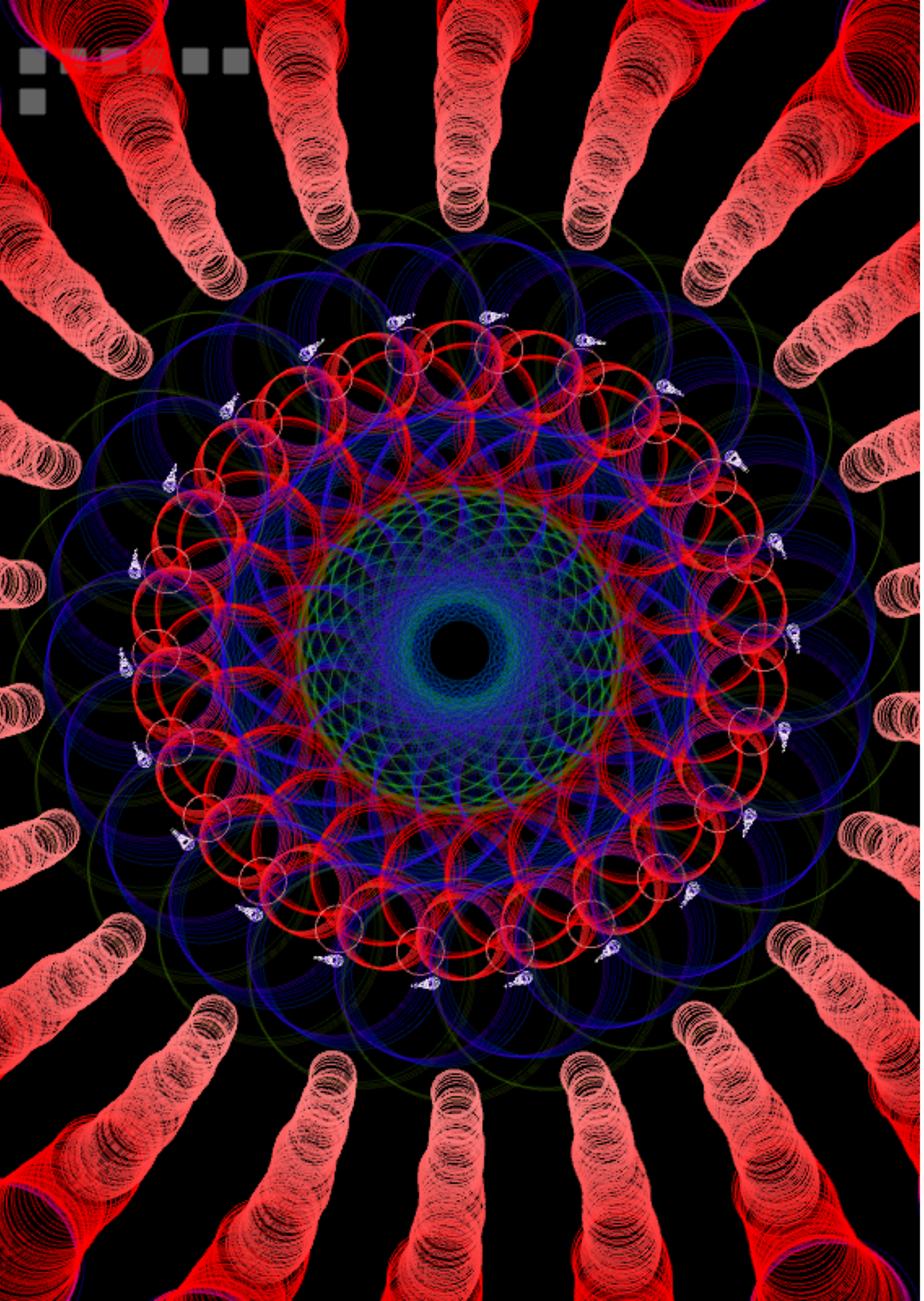


CustomBrush Class

```
class CustomBrush {  //custom brush class created

  int x;  //variables for x and y positions
  int y;
  int h;  // height and width are created
  int w;


  CustomBrush(int x, int _y) { //a constructor is created

    x = x; //x mouse position is set
    y = height-300; //the brush is automatically moved to the bottom edge
  }
  void draw() {
    image(img, x, height-50, 50, 50); //custom .png brush is drawn
  }
}
```

Circle Class

```
// Class created by Pierre Proske
// from the RMIT-Industrial-Design/IntroToProcessingTutorials repository

class Circle{ //class is created
  int x; //variables for x and y position
  int y;
  int radius; //variables for the radius
  int maxsize = 150; //maximum size of radius set

  // Constructor is created
  Circle(int x, int _y){
    x = x-width/2; //mouse x and y position are adjusted to compensate
    y = _y-height/2; // for the sketch plane being translated
    radius = 0;
  }

  void update(){
    radius++; //every frame the radius grows by 1
  }

  void draw(){
    noFill();
    int alpha = (int)map(radius%maxsize,0,maxsize,255,0); //the radius of the square is mapped from
0-150 to 0-255
    stroke(alpha,255-alpha, 255, alpha);  //then used to create changing colour for the str
    ellipse(x,y,radius%maxsize,radius%maxsize); //the ellipse is then drawn
  }
}
```



Reflection

As this was the first time I have ever had to write large slabs of code I found this task quite difficult. however, after looking back at what this task has taught me I feel that. I feel that I now understand the basics of processing and I am now able to read code and be able to understand how things work.

Although it might not be the most elegantly written code out there, I feel like I achieved what I set out to and some of the visuals that can be created with the script are fun and interesting.

This task has solidified the fact that I want to get better at coding because it is such a valuable asset to any designer.