APRYL HOUSE CIM540 FINAL
CODE PLAN

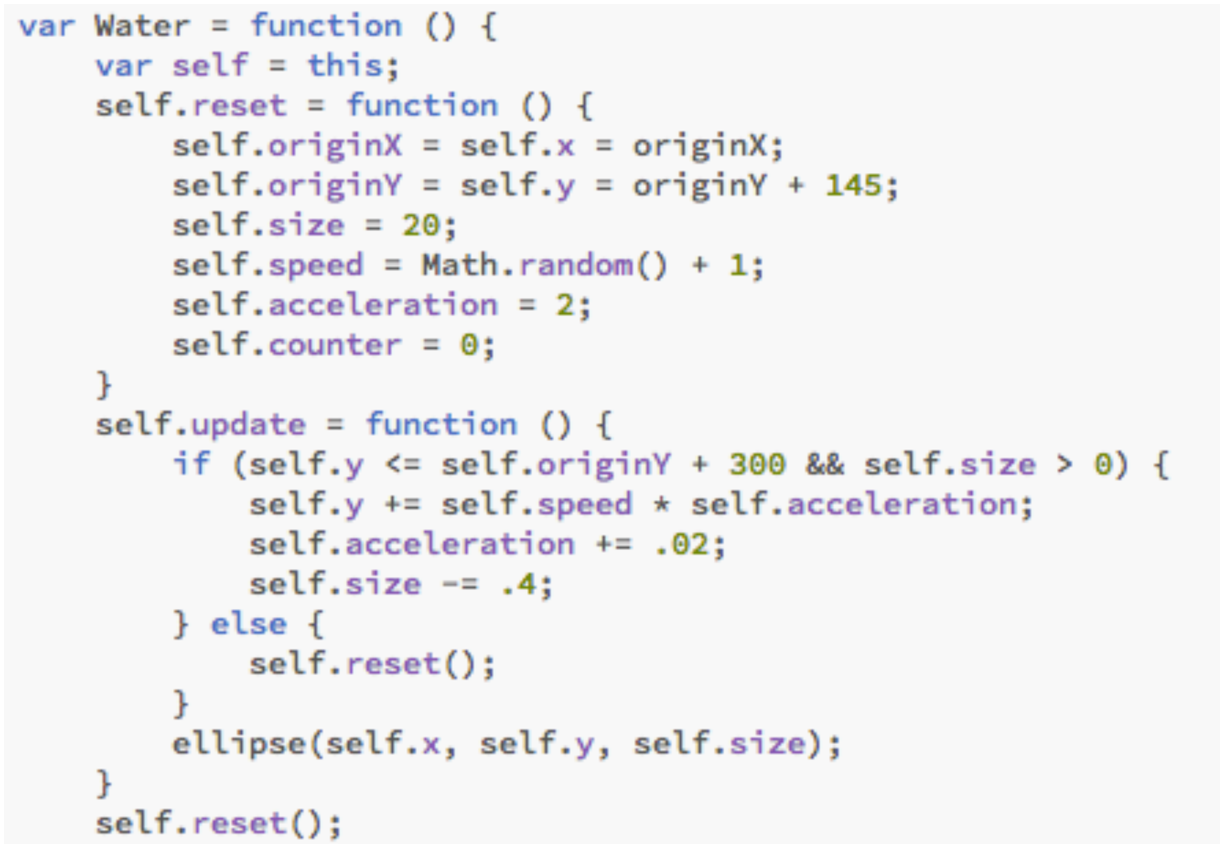
My final project is an interactive art piece. I imagine this piece to function as a screensaver or as an interactive visualization to music.

INPUT	OUTPUT	PSUEDOCODE
<u>object (watering can)</u>		
follows mouse left/right	image of flowers appear in specified portion	load image of watering can , follow mouse X and with offset variables, generate particle variables (water) that originate from image at specific origin and speed, load image of flowers and create Boolean variables to link their display with the position of mouse X and mouse Y
<u>sound</u>		
plays	image of sun scales up and down according to the amplitude	load sound, create variable to link sound source to amplitude, create variable to set scale of image to amplitude level
<u>animation</u>		
plays	sequence of images is pushed on to screen	load image sequence, create array that changes images according to a millis variable, reset millis to allow for looping
<u>object (butterfly)</u>		
screen is loaded	image moves across screen	load image of butterfly, create x and y variables, set x and/or y variables to move with plus or minus, create Boolean variables in order for image to reset (loop)

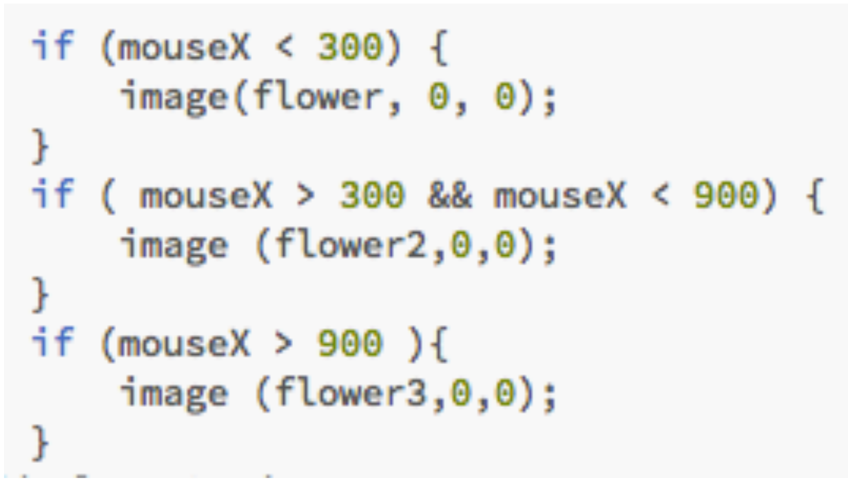
WRITE UP

Although my project has a rather simplistic purpose, the coding of it happens to be a little complex with various working parts. I built my code by establishing the individual working pieces and then arranging them to be a cohesive piece. The first working part I incorporated into my project was the “water” feature.

Following the basic of constructing a particle variable, I was able to manipulate the speed and origin of the particles through variables to achieve a water-like effect. Then I had to create the object the “water” would be coming out of, the watering can. I first began by setting up the speed and origin of the watering can to follow mouse X and mouse Y using a temporary ellipse primitive. After getting the desired speed and origin, I designed a watering can in Illustrator and loaded the png file into my code. Then it was just a matter of reworking the specific position of the water and watering can to make it seems as if the particles (water) was pouring out of the spout of the watering can.



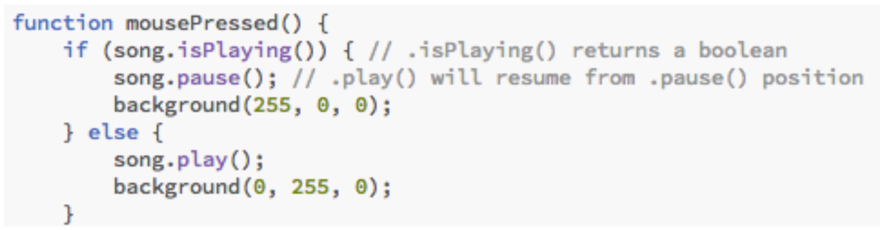
```
var Water = function () {
    var self = this;
    self.reset = function () {
        self.originX = self.x = originX;
        self.originY = self.y = originY + 145;
        self.size = 20;
        self.speed = Math.random() + 1;
        self.acceleration = 2;
        self.counter = 0;
    }
    self.update = function () {
        if (self.y <= self.originY + 300 && self.size > 0) {
            self.y += self.speed * self.acceleration;
            self.acceleration += .02;
            self.size -= .4;
        } else {
            self.reset();
        }
        ellipse(self.x, self.y, self.size);
    }
    self.reset();
```



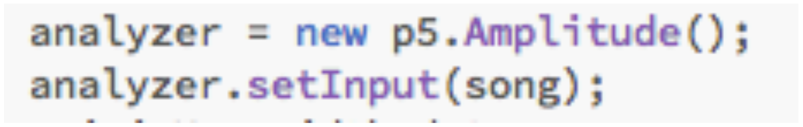
```
if (mouseX < 300) {
    image(flower, 0, 0);
}
if ( mouseX > 300 && mouseX < 900) {
    image (flower2,0,0);
}
if (mouseX > 900 ){
    image (flower3,0,0);
}
```

The next element I added to my code were the flowers. While I initially created a frame animation through a sequence of images I designed in Illustrator, I decided to rethink this idea. I ended up producing an effect to where when the watering can (mouse) rolls over a certain portion of the screen, flowers will appear in that portion of the screen. I achieved this by preloading images of my flowers and then calling them to display with Boolean variables. I divided the screen in to three sections and once my mouse X reach a specified x positions flowers would appear. I did this three times for each section of the screen.

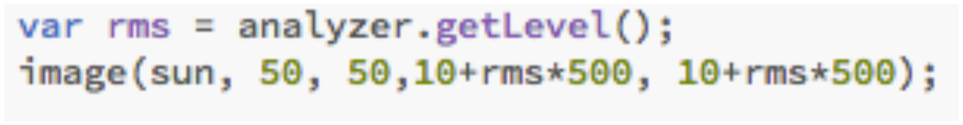
I then added sound visualizations to my code, to achieve this I developed two separate elements and then linked them through variables. I preloaded a sound file into my code and then set it to play automatically and loop. I loaded and drew and image onto the screen as the sun. I then built a variable (var analyzer) that would track the amplitude of my sound file and placed it in the setup. Taking this variable I created a second variable (var rms) to be able to place the information received from the var analyzer into an element that could be attached to my image. When the sound plays the amplitude is represented through the scaling of the sun image.



```
function mousePressed() {
    if (song.isPlaying()) { // .isPlaying() returns a boolean
        song.pause(); // .play() will resume from .pause() position
        background(255, 0, 0);
    } else {
        song.play();
        background(0, 255, 0);
    }
```

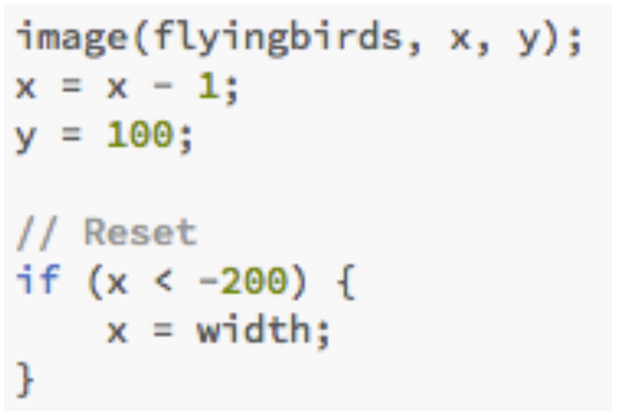


```
analyzer = new p5.Amplitude();
analyzer.setInput(song);
```



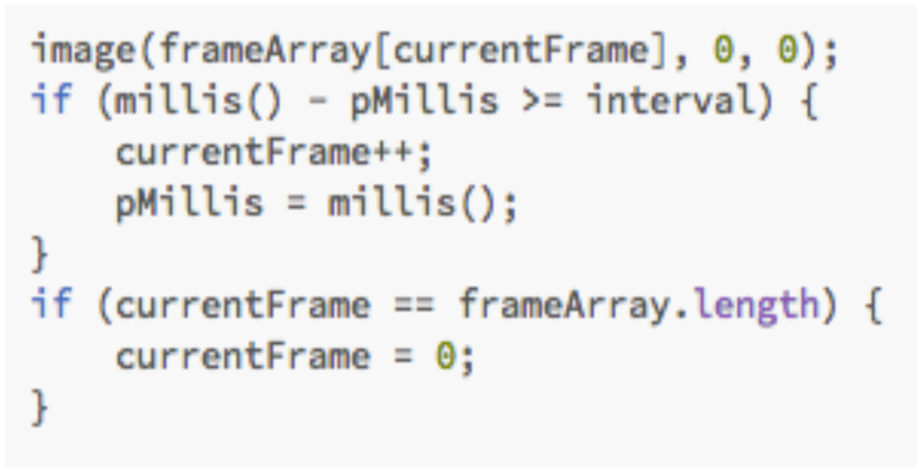
```
var rms = analyzer.getLevel();
image(sun, 50, 50,10+rms*500, 10+rms*500);
```

Once I had achieved the basic interaction and functionality of my code, I added in a few more elements to make the visualization more interesting. I set the background a sky blue and inserted a rectangle and colored it green to simulate grass. I also added a few moving elements. The first was a butterfly that gives the impression of flying across the screen. I built this by loading the image of my butterfly and creating x and y variables for my butterfly's position. I manipulated the x variable to move by adding a "minus one" to the x position of the variable. I then created a Boolean variable that allows the butterfly to loop.



```
image(flyingbirds, x, y);
x = x - 1;
y = 100;

// Reset
if (x < -200) {
    x = width;
}
```



```
image(frameArray[currentFrame], 0, 0);
if (millis() - pMillis >= interval) {
    currentFrame++;
    pMillis = millis();
}
if (currentFrame == frameArray.length) {
    currentFrame = 0;
}
```

The second moving element I added was a frame animation. This was achieved with an array that pushes a sequence of images I created in Illustrator onto the screen. A millis variable is used to set the timing and allow for looping. This animation shows a couple flying a kite.

Lastly, I added text to better explain the interactivity of my code and minor aesthetic elements such as no cursor and tweaked the position of all my variables to achieve my desired visualization.