Documentation of the design and development

Project 1 - Simple particles

Phase 1

- In the first phase of making this project, I decided to get as much preparation done to help me understand how to execute the project from a coding/technical standpoint. I'm going to be discussing the videos, code, art work and more, that helped me build the foundation of this particular project.

- I wanted my first project to be based on the particle system mentioned in class. So to essentially make a project that has objects moving around the drawing scene in some form. To prepare for this I watched the recorded lesson going over particle systems and looked at the code produced from this lesson.

- The particle system program made in class by Will was a program which had particles set at different random sizes and colours moving across the screen and when a particle connected to another they would make a 'bouncing' notion by repositioning the particles velocity by -1. I found this code to be useful and I liked the fact that this example had the particles interact with one another and definitely wanted to implement that element into my own project. But as a whole I think for a 'simple' particle system it is a bit lengthly so I looked at other examples to see other forms of particle systems.

1.1 - Wills simple particle example

```
class Particle {
  constructor(x, y) {
    this.pos = createVector(x, y)
    this.vel = createVector(random(-1, 1), random(-1, 1))
    this.diam = random(15, 30)
    this.colour = random(0, 200)
  }
```

```
  update() {
    // move ourselves
    this.pos.add(this.vel)
    let rad = this.diam / 2


    if (this.pos.x <= rad || this.pos.x
>= width - rad) {
      this.vel.x = -this.vel.x
      // same as
      // this.velocity.x *= -1
    }
    if (this.pos.y <= rad || this.pos.y >= height - rad) {
      this.vel.y *= -1

    }
  }

  bounce(particles) {
    this.over = false
    for (let otherParticle of particles) {
      if (otherParticle != this) {
        let d = this.pos.dist(otherParticle.pos)
        if (d < (this.diam + otherParticle.diam) / 2) {
          let temp = otherParticle.vel.copy()
          otherParticle.vel = this.vel
          this.vel = temp
        }
      }
```

```
    }
  }

  draw() {
    noStroke()
    fill(this.colour)
    ellipse(this.pos.x, this.pos.y, this.diam, this.diam)
  }
}

let particles = []

function setup() {
  createCanvas(400, 400)

  for (let i = 0; i < 15; i++) {
    let particle = new Particle(random(30, width - 30), random
(30, height - 30))
    particles.push(particle)
  }
}

function draw() {
  background(255)

  for (let particle of particles) {
    particle.bounce(particles)
    particle.update()
    particle.draw()
  }
```

```
}
```

- The next example of code I looked at came from the ‘The Coding Train’ video which went over how to make a simple particle system program in p5.

- This piece of code produced a simple program that created a for loop for a large set of particles in one space that would be spread and pushed up the y axis by increasing the velocity as it moved up then once they reached a certain position, disappearing. Which would essentially create a sparkler looking effect on the particles. I also found this piece of code quite helpful and interesting since It mimicked a real object which could be helpful when placing the code into a creative sketch. I also thought the layout was a lot more easier to comprehend and the code was quite compressed. I took inspiration from this through wanting my particles to reflect a real life situation/object to make for a more interesting sketch.

1.2 - Code made from The Coding Train Video - Simple Particle System.

```
particles = [];

function setup() {
  createCanvas(600, 400);
}

function draw() {
  background(0);
  for (let i = 0; i < 5; i++) {
    let p = new Particle();
    particles.push(p);
  }
  for (let i = particles.length - 1; i >= 0; i--) {
    particles[i].update();
```

```
    particles[i].show();
    if (particles[i].finished()) {
      // remove this particle
      particles.splice(i, 1);
    }
  }
}

class Particle {
  constructor() {
    this.x = 300;
    this.y = 380;
    this.vx = random(-1, 1);
    this.vy = random(-5, -1);
    this.alpha = 255;
  }

  finished() {
    return this.alpha < 0;
  }

  update() {
    this.x += this.vx;
    this.y += this.vy;
    this.alpha -= 5;
  }

  show() {
    noStroke();
    //stroke(255);
```

```
    fill(255, this.alpha);
    ellipse(this.x, this.y, 16);
  }
}
```

https://youtu.be/UcdigVaIYAk

1.3 - The Coding Train Video - Simple Particle System

- Lastly, I then also looked at the particle system example given on the p5.js website to expand on my coding skills and further build my understanding on how I could program this mini project.

- This code is a program that creates a particle class, that displays a group of particles on the top the scene then adds acceleration and velocity to again like the last example create a sparkler looking effect on the particles. Great example to look at but essentially a bit unnecessarily lengthy for the same results as the piece code given by 'The Coding Train'.

1.4 - p5.js.org - Simple particle example

```
let system;

function setup() {
  createCanvas(720, 400);
  system = new ParticleSystem(createVec
tor(width / 2, 50));
}

function draw() {
  background(51);
  system.addParticle();
  system.run();
```

```
}

// A simple Particle class
let Particle = function(position) {
  this.acceleration = createVector(0, 0.05);
  this.velocity = createVector(random(-1, 1), random(-1, 0));
  this.position = position.copy();
  this.lifespan = 255;
};

Particle.prototype.run = function() {
  this.update();
  this.display();
};

// Method to update position
Particle.prototype.update = function(){
  this.velocity.add(this.acceleration);
  this.position.add(this.velocity);
  this.lifespan -= 2;
};

// Method to display
Particle.prototype.display = function() {
  stroke(200, this.lifespan);
  strokeWeight(2);
  fill(127, this.lifespan);
  ellipse(this.position.x, this.position.y, 12, 12);
};
```

```
// Is the particle still useful?
Particle.prototype.isDead = function(){
  return this.lifespan < 0;
};

let ParticleSystem = function(position) {
  this.origin = position.copy();
  this.particles = [];
};

ParticleSystem.prototype.addParticle = function() {
  this.particles.push(new Particle(this.origin));
};

ParticleSystem.prototype.run = function() {
      this.particles.splice(i, 1);
    }
  }
};
```

- To conclude, looking at the examples and the recorded lesson greatly improved my technical understanding on particle systems. e.g. building programs with objects and classes and probably using constructors and understanding how to manipulate the acceleration and velocity to have the objects more in a certain way . What I've taking from this stage of development is that I would like this project to have the object (particles) interact in some way, I want my code to be easy to read and compressed and to program the particles in a way that displays a real life object or scenario in some way to creative a more interesting drawing scene.

Phase 2

- In this section, I talk a little about the creative aspect of this project. As mention before I wanted to make a drawing scene that displayed a real life scenario, so i thought of some ways I could do this. There were three examples I thought about using one being a snowing/winter scenario where the particles would be the snow, a rainy day so the particle could be the rain and lastly I thought of a bee hive scene where I could make the small particles the bees.

Snowing example

Raining example

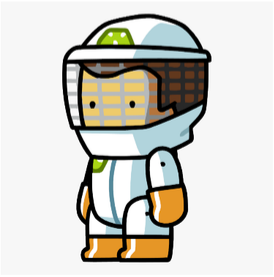
bee hive example

- After a little thought, I choose to do a code drawing based on a bee keeper/ bee hive example, mostly because I thought that choosing the snow or rain option would result in me doing a simple loop that would make the particles trickle down which might be too boring/easy. So doing the bee hive option would allow for a slight challenge for creating a randomised hectic moments for the bee particles.

Phase 3

- In this section, will just be quickly going over the plan on how to execute the sketch and what i want it to look like.

- Below is a simple sketchpad plan on what items i want to be included in my p5 project. obviously the p5 sketch will not be this detailed since it will be done with simple shapes. But essentially i just want my sketch to have clouds, a tree with bees buzzing around it, a beekeeper and lastly a bee hive box.



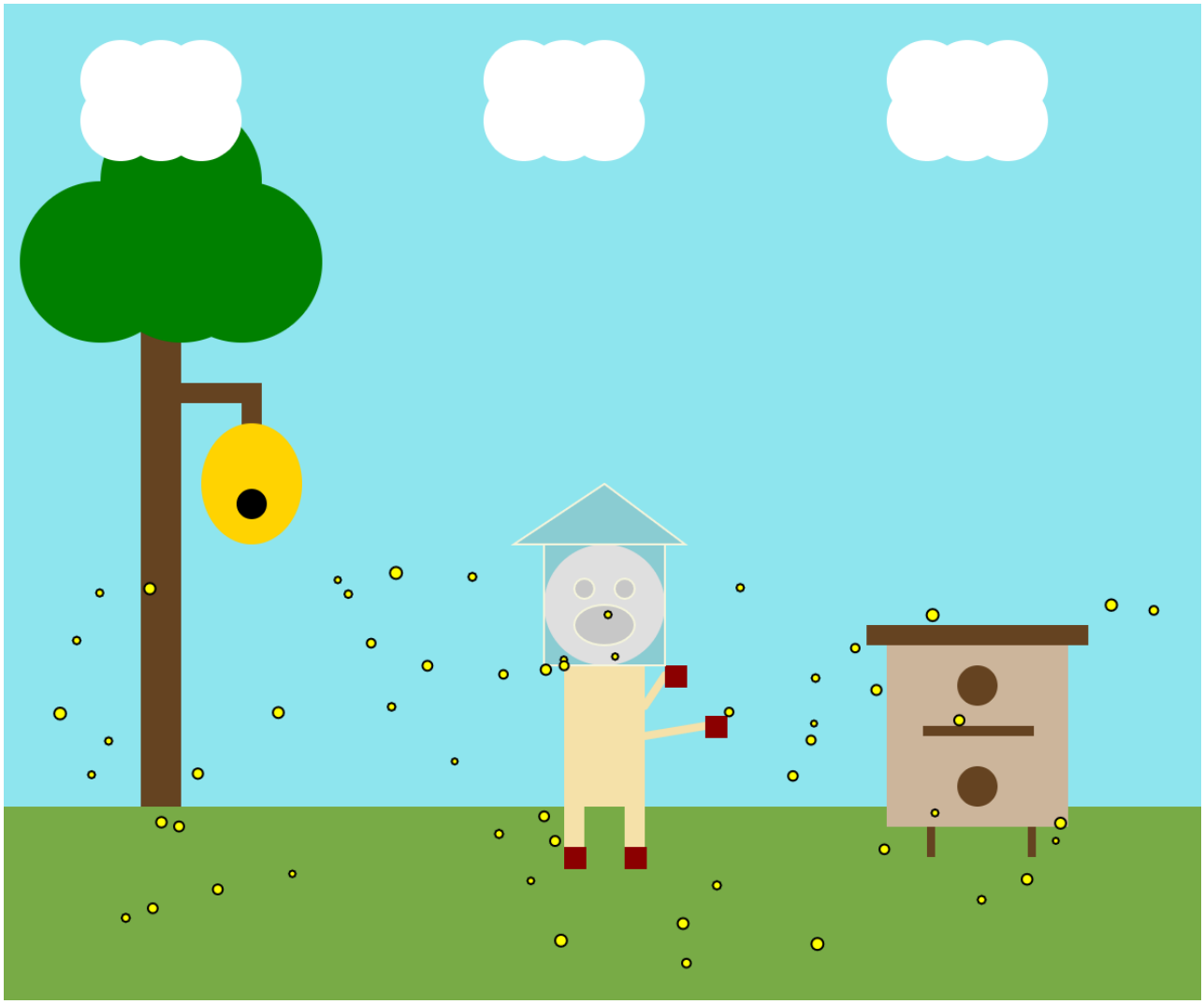
The code requirements:

- For the simple particle to generate I need for loops to create randomised bees across the screen and to have the bees displayed.
- A function to control the moment and the starting area of the bees so the function could possibly include the x, y position of the bees starting area, setting the size of the bees, setting how fast I want the bees to move and setting a radius to have the bees spread out.
- of course in the drawing area coding the visual aspects of the sketch e.g. tree, beekeeper and clouds.
- I could also include some bee sounds to add to the sketch.

Phase 4

- This last phase, is me going over the final project executed, explaining the code and if any changes to the original plan was made.

Final sketch
https://editor.p5js.org/shardaidawk/sketches/ij44ruAe9



Started off the code by adding arrays to hold the bees and clouds. As well as a variable that would hold the bee sounds.

```
let bees = []; // Array for the bees
let clouds = [{}, {}, {},]; // Array for the clouds
let beesound;
```

Then in the preload function I created a variable to hold the mp3 file ‘Beehivesound’, that I wanted to be played when the sketch started.

In the setup function I set the canvas size, set a colour and then proceeded to make sure the bee sounds that I put in the preload function, would loop when I started the sketch.

```
function preload() {
 beesound = loadSound('Beehivesound.mp3');
}

function setup() {
  createCanvas(600, 500);
  fill(240);
  noStroke();
  beesound.loop();

}
```

Now in the drawing function through the given p5 shapes, I made the sky and the green grass/floor, a beekeeper, tree and nest and a bee box. There were no changes from the original plan for the background objects I wanted to make and I managed to keep them the same colours and get them all fully completed.

```
function draw() {
  //Background settings to make sky & floor
  background('#8EE5EE');
  noStroke();
  fill('#78AB46');
  rect(0,400,600,100);
```

```
// Beekeeper
// Head
fill('white');
ellipse(300,300,60,60);
// Body
fill('#f5e1a9');
rect(280,330,40,70);
// Legs
rect(280,350,10,80);
rect(310,350,10,80);
// Arms
strokeWeight(4);
stroke('#f5e1a9');
line(330,335,320,350);
line(320,365,350,360);
strokeWeight(1);
// Shoes
noStroke();
fill('#8b0000');
rect(310,420,11,11);
rect(280,420,11,11);
// Hands
rect(350,355,11,11);
rect(330,330,11,11);
// Helmet
stroke("beige");
fill('rgba(128,128,128, 0.25)');
rect(270,270,60,60);
triangle(255, 270, 340, 270, 300, 240);
// Face
```

```
ellipse(290,292,10,10);
ellipse(310,292,10,10);
ellipse(300,310,30,20);


// Tree and nest
noStroke();
fill("#654321");
rect(70,130,20,270);
rect(70,190,60,10);
rect(120,200,10,20);
fill("#FFD300");
ellipse(125,240,50,60);
fill("black");
ellipse(125,250,15,15);

fill("green");
ellipse(50,130,80,80);
ellipse(90,130,80,80);
ellipse(120,130,80,80);
ellipse(90,90,80,80);

//Bee box
fill("#ccb59b");
rect(440,320,90,90);
fill("#654321");
rect(430,310,110,10);
ellipse(485,340,20,20);
rect(458,360,55,5);
ellipse(485,390,20,20);
```

```
rect(460,410,4,15);
rect(510,410,4,15);
```

As well as the previous background objects i mentioned, I also made clouds using a for loop so instead of making a bunch of ellipses to replicate the look of clouds in different areas, I did it once in a for loop to have the same bit of code duplicated across the sketch. In the for loop I made six ellipses that together look like a cloud and had that times by the length.

```
// For loop for the clouds

for (i = 0; i < clouds.length ; i++) {
  noStroke();
  fill("white");
  ellipse(100+200*i,60,40,40)
  ellipse(100+200*i,40,40,40)
  ellipse(80+200*i,40,40,40)
  ellipse(60+200*i,40,40,40)
  ellipse(60+200*i,60,40,40)
  ellipse(80+200*i,60,40,40)
  }
```

To end the drawing function, I started to make the code for the simple particle system. I started by setting the colours of the bees outer black and filling it yellow. Then setting the speed of the bees movement in the variable f. Then creating a loop to generate a random amount of the bees in each frame ranging from 0 to 5. Lastly I then created another for loop to have the bee object i just created move at the speed I set in the variable f and then had the bees displayed.

```
// Colour for the bees
  stroke("black");
  fill('yellow');

// Code to generate particle moments to mimick the movement of
bees
  let f = frameCount / 50; // Setting the speed of the bees mo
vement

// For loop to create a random amount of bees in each frame
  for (var i = 0; i < random(5); i++) {
    bees.push(new bee()); // Adding the bee objects
  }

// For loop to get the bees to be displayed
  for (let hive of bees) {
    hive.update(f); // update bee position
    hive.display(); // draw bee
  }
}
```

I then created a function called bee which included which set the x and y position of where the bees particles would come out from, the starting angle which ranges from 0 to 4*PI and the randomised size ranging from 3 to 6. And also the radius to allow the bee particles to be spread out using the square root and randomised function.

```
function bee() {
  // Setting the starting positions for the bees in which i ma
de it the bee hive opening. As well as setting the angle and s
ize of them.
  this.positionX = 125;
```

```
  this.positionY = 280;
  this.startingangle = random(0, 4 * PI);
  this.beesize = random(3, 6);

  // Written so the bees are spread out.
  this.radius = sqrt(random(pow(width / 2, 2.4)));
```

Then continuing in the bee function, I then inputted a section to that set an updated speed and x position for the bees, so they can move around the screen with these updated attributes. Then I also updated the y axises so that the bees would fall differently. Then to end this bit off I added some code to make sure that if the bees stoped displaying if they go past the screen.

```
this.update = function(time) {
    let s = 0.8; //speed
    let angle = s * time + this.startingangle;
    this.positionX = width / 2 + this.radius * cos(angle);

    // Written so bees fall differently on the y axis.
    this.positionY += pow(this.beesize, 0.8);

// Making sure the bees are stop if they go past the screen.
    if (this.positionY > height) {
      let index = bees.indexOf(this);
      bees.splice(index, 3);
    }
  };
```

Lastly, to gather the code altogether I made a display variable to start the bee function and made an ellipse with all the objects listed before to display the bees

and that was the end of the bee function. To end the entire sketch I made a mousePressed function so stop the bees sounds if the mouse is pressed in case it was annoying.

```
this.display = function() {
    ellipse(this.positionX, this.positionY, this.beesize);
  };

function mousePressed() {
  //Beesounds
  if (mouseX, mouseY) {
    beesound.play();
  }

}
}
```

To give some evaluation on the project, I did everything that I planned to do in terms of the objects I wanted to create and stuck fairly close to original plan, I got to make a simple particle system that resembled a real life scenario but I failed to find a logical way to have the particles interact with one another or another object. What I possibly could have done is have the bees move in a certain direction when interacting with the beekeeper object. There were no major challenges for this project and I believe I got to execute the exact project wanted technically and creatively.

Project 2 - Y2K Inspired Animation

Phase 1

- I wanted to make this project based around the geometric wave created in class by will. So the overall result of this project is to create an interesting animation using the the built in functions in angles modes in p5.js (sin, cos, PI, DEGREES, RADIANS).

- Just like before I started with looking how to execute the project from a technical aspect which I first looked for the 'silky wave' created in the online class. Looking at this example was very useful for understanding for the basics on how to create a geometric wave using these angle based functions. I now understood how to use the beginShape() function. And this example also taught me that using for loops with angle based functions could create amazing moving lines. Though this example is a great stepping stone to learning something a bit more complicated, I knew I wanted my project to be a little bit more conceptually interesting than just a simple wave.

```
function wave(q) {
  let amplitude = sin(q * 0.034) * 50;
  let frequency = 1;
  let phase = q * 0.05;

  beginShape();
  for (let x = 0; x < width; x += 2) {
    let t = x / width; // between 0 and
1
    let y = sin(2 * PI * frequency * t + phase) * amplitude;
    vertex(x, y);
  }
  endShape();
}

function setup() {
  createCanvas(400, 400);
  noFill();
```

```
}

function draw() {
  background(250);
  translate(0, height / 2);

  for (let i = 10; i > 0; i--) {
    stroke(i * 25);
    wave(frameCount - i * 3);
  }

}
```

- The next example I looked at was the youtube video ‘Coding Challenge #55: Mathematical Rose Patterns’ by The Coding train. The video went through how to create an angle based rose using sliders which changed way the rose looked. Through these example I understood more about the TWO_PI functions and the ways that can be used but also how to a slightly more interesting angle based project. I also liked the use of sliders for the users to to interact with the sketch.

```
// Daniel Shiffman
// Mathematical Roses
// Video: https://youtu.be/f5QBExMNB1I
// Based on: https://en.wikipedia.org/w
iki/Rose_(mathematics)
// https://thecodingtrain.com/CodingCha
llenges/055-roses.html

var d = 8;
var n = 5;
var sliderD;
var sliderN;
```

```
function setup() {
  createCanvas(400, 400);
  sliderD = createSlider(1, 20, 10, 1);
  sliderN = createSlider(1, 20, 10, 1);
  sliderD.input(draw);
  sliderN.input(draw);
}

function draw() {
  d = sliderD.value();
  n = sliderN.value();
  var k = n / d;
  background(51);
  push();
  translate(width / 2, height / 2);

  beginShape();
  stroke(255);
  noFill();
  strokeWeight(1);
  for (var a = 0; a < TWO_PI * reduceDenominator(n, d); a +=
0.02) {
    var r = 200 * cos(k * a);
    var x = r * cos(a);
    var y = r * sin(a);
    vertex(x, y);
  }
  endShape(CLOSE);
  pop();
```

```
  noLoop();
}

function reduceDenominator(numerator, denominator) {
    function rec(a, b) {
        return b ? rec(b, a % b) : a;
    }
    return denominator / rec(numerator, denominator);
}
```

https://youtu.be/f5QBExMNB1I

- To sum up the technical learning for this project, I found the examples that I wanted to be base my project on to be extremely useful in terms of learning new functions and how to incorporate them into these types of projects. I gathered that I wanted my project to be a bit more conceptually interesting and possibly have the users of the sketch influence the physically appearance of the project like rose pattern example.

Phase 2

- For the creative aspect of this project, I decided I wanted to be Y2K influenced when I came across a tumblr page 'Y2K aesthetic institute' (https://y2kaestheticinstitute.tumblr.com/). The page goes through all of the videos, adverts, graphic designs created in the Y2K area of the early 2000's, and I thought it would be an interesting concept to base my project off on.

- The Y2K Aesthetic is a specific design concept that existed primarily from 1995-2003. Designs included objects that were blobby, shiny, translucent and iridescent. It was the future preparing for the turn of the millennium. From fashion and architecture to music and art, the optimism and techno-utopianism of the early 2000s found its way into every element of design.

- Some elements included, blobby objects, features of avatars and robots, futuristic/techno fonts, translucency, shiny materials, space themes and Tokyo inspired themes.


V
y2k
I
C
E


Support the Institute for Y2K Aesthetics


STEVE JOHNSON &
MATT TANGENT
JOEY BELTRAM
DJ MAGIC
ANDY HUGHES
GUIDO
JOHN KELLEY
ELLIS DEE
Saturday, August 2, '97

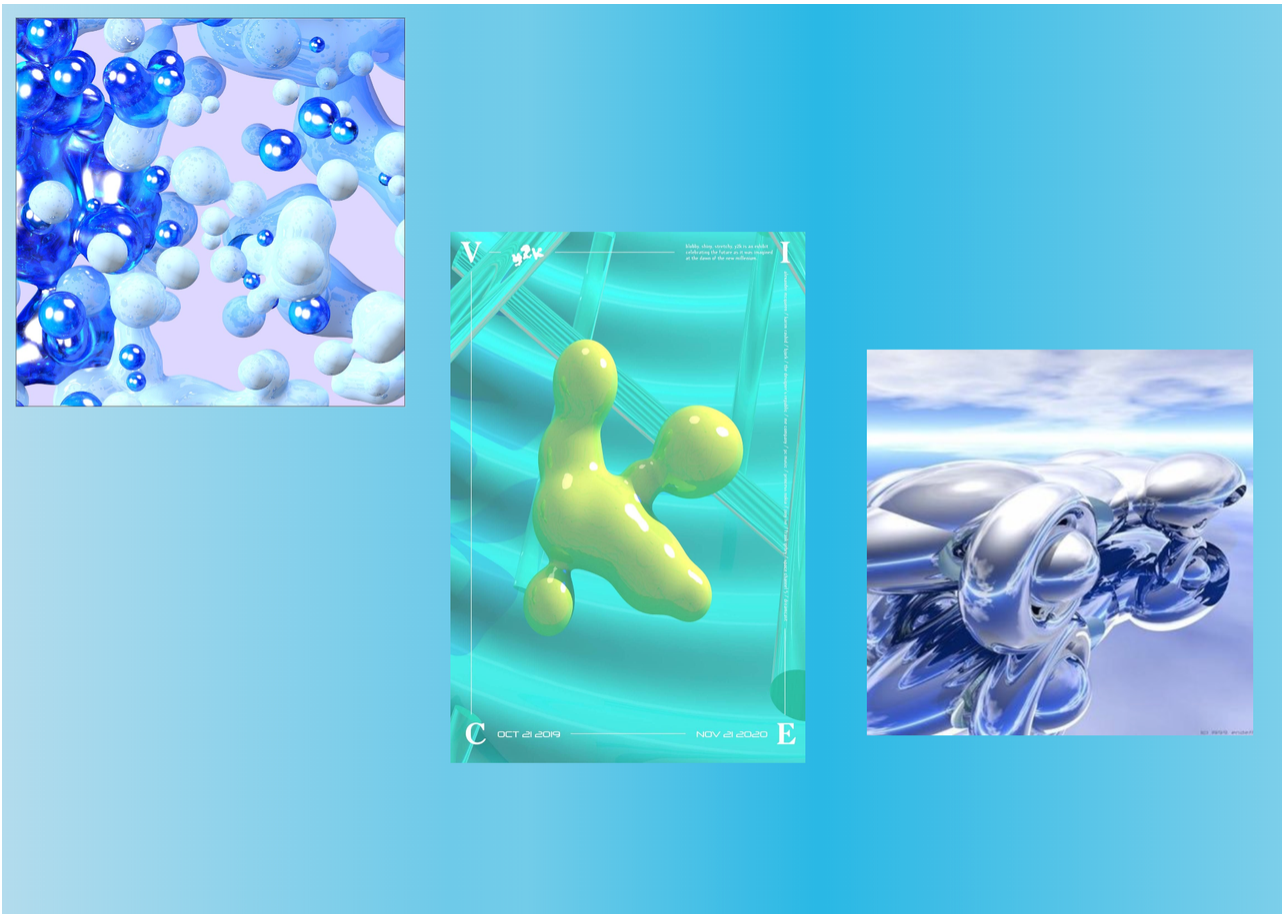
- A specific group of artists I looked into was The Designers Republic. The Designers Republic are a graphic design team based in Sheffield, England whose's work contributed to a huge part of the Y2K era. They are best known for electronic music logos, album artwork, and anti-establishment aesthetics, some example being album cover for Aphex Twin's Syro packing. They also made the infamous wipeout font used for many video game packeting and more.

- From all the research done, I believe I want my project to have more of a blobby theme used in the y2k aesthetics, maybe incorporating some music that has to do with the era and including the prime colours used in design such as bright futuristic pastels.

Phase 3

- The initial plan for this project is to have Y2k looking blobs across the screen moving using the angle mode functions in p5.

- Below is a quick sketch on what colours I want use and some examples of the type of blobs that I wanted to be incorporated. I also want the colours to be on the Y2K theme so a baby blue background would be needed.



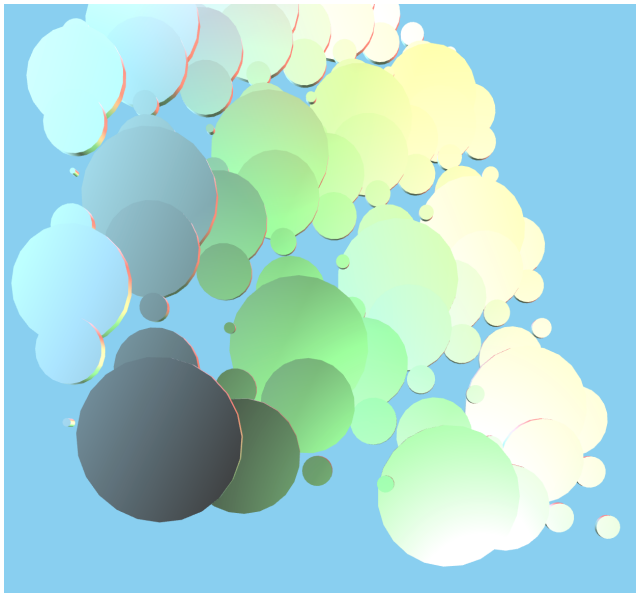
code requirements:

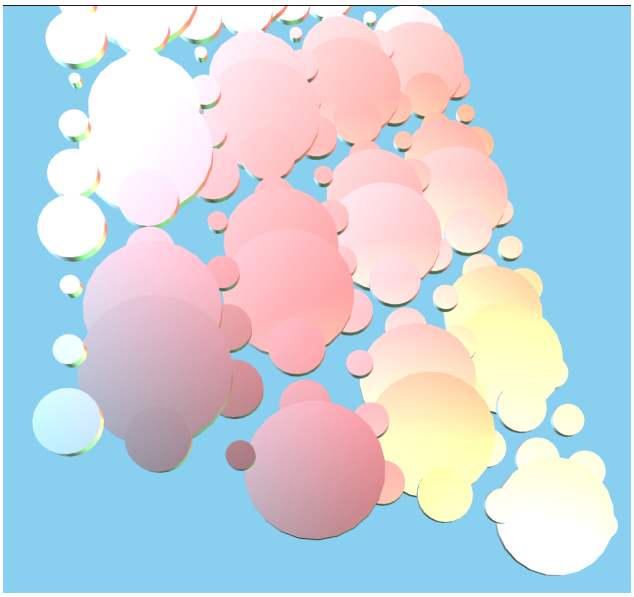
- Interesting animation using the the built in functions in angles modes.
- Users interaction to make the animation change in some way.
- Y2K blobs and colours.
- Y2K musics playing in the background

Phase 4

- This last phase, is me going over the final project executed, explaining the code and if any changes to the original plan was made.

Final sketch
https://editor.p5js.org/shardaidawk/sketches/_GE3q1MQQ





Started off the code with creating Aphex variable and in the preload function inserting the 'AphexTwin' file which one it, is the Aphex twin song 'Xtal' which I thought would be a good song to play in the background as Aphex twin's music has create connections to Y2K era with his futurist techno music as well as him working with the designer republic connecting his music further.

```
let Aphex;


function preload() {
 Aphex = loadSound('AphexTwin.mp3');


}
```

In the set up function, I created a canvas with and length size and included WEBGL which is an API that allows me to include 3D graphics to my project. This was

necessary as I wanted to create blob like objects as well as 3D design connecting to the Y2K era. Then I had the Aphex sound loop.

```
function setup() {
  createCanvas(700, 1000, WEBGL);
  Aphex.loop();
}
```

In the draw function I started by including the background colour which is a sky blue to stay on theme, inserted the nostroke function to avoid outer-lines on the blobs. Then started with the rotate function to have the objects rotate I set the angle to PI/4 and since I'm using WEBGL for the 3D graphic I set the axis. Then included the ambientLight function to give a glowing effect. Lastly I inputed two directionalLight functions to be able to create a directional light change with the mouse in Y2K colours.

```
function draw() {

  background("#89cff0");
  noStroke()

  rotate(PI/4, [1,1,0]);
  ambientLight(150);
  directionalLight(137, 207, 240, 0.5, 0.25, 0);
  directionalLight(233, 50, 50, -0.5, 0.5, 0);
```

Since the WEBGL API uses space differently I used the translate function to have the objects placed in the middle of the p5 editor screen and the specular material function to create a glowing light affect and lastly the point light function so when the mouse is moved so the light direction is moved with it. So this part ticks the box for wanting the user to interact with the animation.

```
translate(-width/2.5, -height/1);
  specularMaterial(100);
```

```
  pointLight(100, 200, 100, mouseX - mouseY/2, mouseY - mouse
X/2, 50);
```

Lastly I inputted a nested loop to have the blobs multiply across and down the sketch with the translate function again, two rotation functions to get the blobs to animate and rotate and lastly using the cylinder function to create the blobs. I also used the push and pop methods to save the translation state.

```
 for(var i = 0; i < width; i += 80){
    for(var t = 0; t < height; t += 80){
      push()

      translate(i,t)
      rotateZ(cos(i * t / 1000 + frameCount / 1000) /10 + mous
eY / 400)
      rotateX( mouseY / 400)
      cylinder(0.1 + sin (i / 50 + frameCount / 20 + mouseY /
100 ) * 50 + cos (t / 50 + frameCount / 10 ) * 30, 10)

      pop()

    }
  }
}
```

To conclude I was able to create an angle based/ wave inspired animation quite well, I felt as if I stuck to the Y2K theme and again no major changes to the original plan. I got to use Y2K themes such as blobby objects, bright futuristic colours and the music to match.

Project 3 - King Krule Promo Filter video

Phase 1

- For this project I was looking a the built in filtering in p5.js and the filtering project made in class. (https://editor.p5js.org/whg/sketches/pbSqC2XPR). I decided that I wanted to make a simply filter base project.

- Since this was going to be a more simple project, I only looked at the example from class for my reference which was just a simple filter sketch that put a visor like filter onto a picture with a frog. I thought I could make a video with a filter instead and add some extra elements to make it more interesting.

- Studying the example, made creating the project pretty straight forward, it showed me how to use the pixel elements in p5 to create filters and for loops to expand on the filters.

```
let img, originalPixels

function preload() {
  img = loadImage('frog.jpg')
}

function setup() {
  createCanvas(400, 400);
  img.loadPixels();
  originalPixels = img.pixels.slice()

  background(220);

  img.loadPixels()

  for (let y = 0; y < height; y++) {
    for (let x = 0; x < width; x++) {
```

```
      let index = (y * width + x) * 4
      let [r, g, b] = originalPixels.slice(index, index + 3)

      let d = (255 - dist(x, y, 100, 180) * 1) / 255
      img.pixels[index] = r * d ;
      img.pixels[index +1] = g * d;
      img.pixels[index +2] = d * 170;
    }
  }

  img.updatePixels()



  image(img, 0, 0)
  //filter(GRAY)
  //save('frog-vignette.jpg')
}
```

Phase 2

- There isn't much creative aspects put into this project but recently, my favourite artist King Krule released a new album called Man Alive! so i thought I could do a mini promo video using the filter methods for the album.

King Krule: Man Alive!

King Krule

Phase 3

- With this being a video based project, I couldn't make an actual sketch, but my thoughts were to do a video mask filter which essentially is the video playing inside of the text. Below is an example of how i wanted it to look.

https://youtu.be/uVN-Zx6U6SM

code requirements:

- Play King Krule music video inside text.

- using filter pixel methods.
- stop/play function.

Phase 4

final sketch https://editor.p5js.org/shardaidawk/sketches/aG6GjpbDh

Started the code with setting the variables needed for the whole sketch. In the preload function I loaded a font that downloaded called FilmNoir. Then I then loaded the Man Alive ! logo to have it in the corner of sketch.

```
let firstLine = "MANALIVE !";
let secondLine = "King Krule";
let mainVideo;
let font;
let mimg;
let playing = false;
let button;
```

```
function preload() {
    font = loadFont('FilmNoir.ttf');

    mimg = loadImage('Manalive.png');
}
```

In the setup of the function I created a button to start the video and when the mouse is pressed it pauses the video. Then for the main video variable I inserted the King Krule ‘Dum Surfer’ music video, had it loop and hide it and loaded the pixels as well as setting the canvas size to the video.

```
function setup() {

  button = createButton('PLAY!');

  button.mousePressed(PlaypauseVid);

  mainVideo = createVideo('Kingkrule.mp4', () => {
    mainVideo.loop();
    mainVideo.hide();
    mainVideo.loadPixels();
    createCanvas(mainVideo.width,mainVideo.height);
    });


  }

function draw() {
```

```
    background(0);
    mainfilter(mainVideo);
    image(mimg, 0, 0);

}
```

In the main filter function, This where I started to code the video mask filter. I stated by filling the background, and inputting the text with a bunch of attributes like font, alignment and size. Then I created a new variable to get a region of pixels which for this project was the whole canvas.

```
function mainfilter(theCanvas) {
    fill(255);
    textSize(100);
    textFont(font);
    textAlign(CENTER);
    text(firstLine,width/2,height/7 + height/4);
    text(secondLine, width/2, height/2 + height/4);

    let img = get(0, 0, width, height);
    image(theCanvas,0,0);
    imagePosition(img, theCanvas);
        img.resize(width, height);
    image(img, 0, 0);
}
```

In the image position function I loaded the pixels of the img and imgPos variable previously created. Then added a for loop, an if statement and for loop nested into that so that video pixels would only show through the text. Then updated the pixels on the imgPos to have the code executed.

```
function imagePosition(imgPos, img) {
```

```
    img.loadPixels();
    imgPos.loadPixels();

    for(let i = 0; i < imgPos.pixels.length; i += 4) {
        if(imgPos.pixels[i] != 0) {
            for(let t = i; t < i + 4; t++) {
                imgPos.pixels[t] = img.pixels[t];
            }
        }
    }

    imgPos.updatePixels();
}
```

Lastly like mentioned before i wanted to have a play and pause button so I added a simple function which included a if statement so when the button is pressed is plays and pressed again it pauses.

```
function PlaypauseVid() {
  if (playing) {
    mainVideo.pause();

    button.html('PLAY!');

  } else {
    mainVideo.loop();
    button.html('PAUSE!');
  }
  playing = !playing;
}
```

The project was pretty straight forward and I was able to getting everything done on time and execute all the code requirements.

Project 4 - My Neighbour Totoro

Phase 1

- Looking at 'The Coding Train' videos on objects and images, I thought I would make a animated image based project.

- Previously I went over the ways I learn't to use objects to create p5 animation so there wasn't much technical knowledge needed this one, I simply wanted to expand on my object skills and create a different type of animated project.

https://youtu.be/i2C1hrJMwz0

https://youtu.be/7BoJBYh16CQ

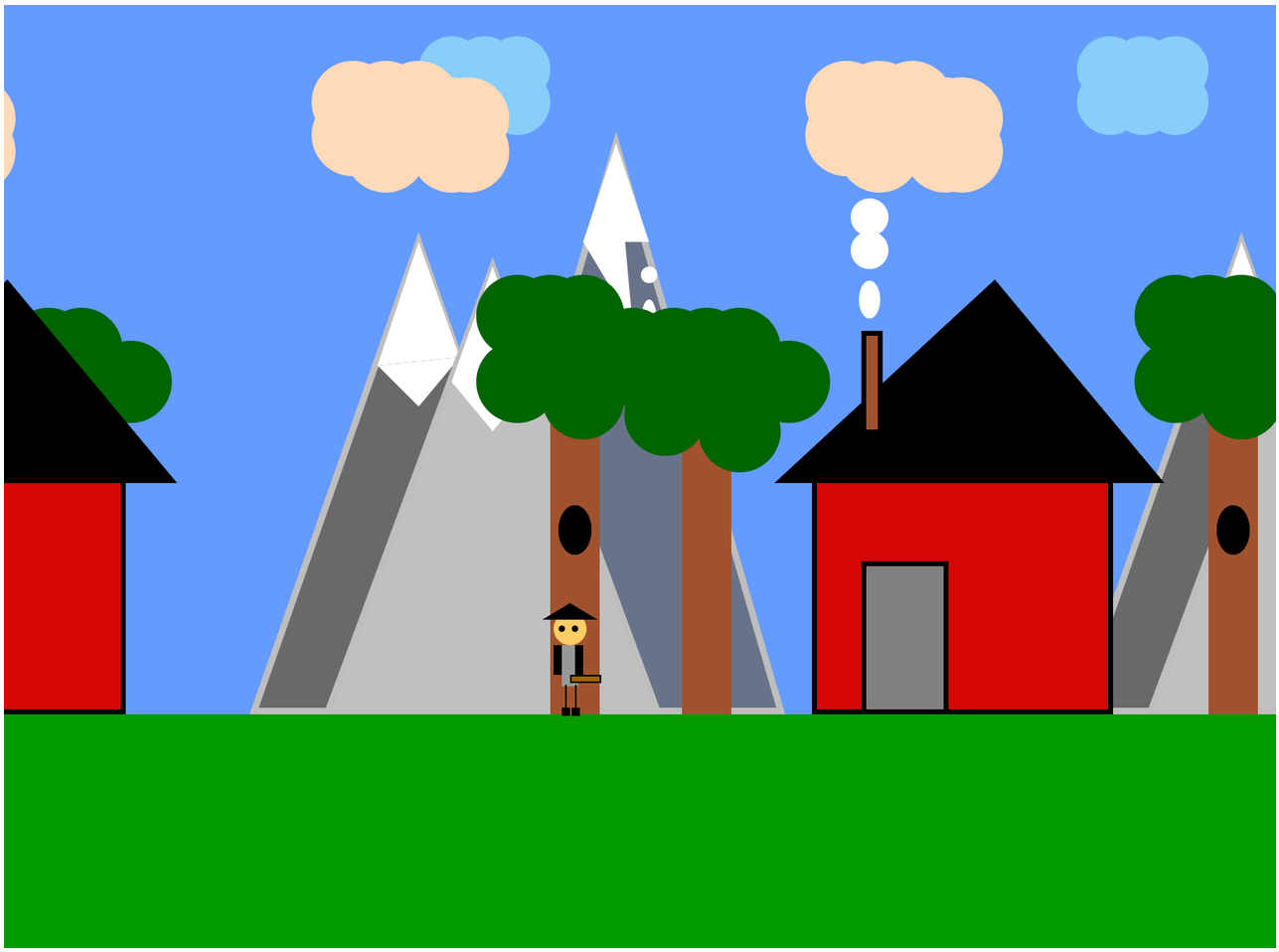
Phase 2

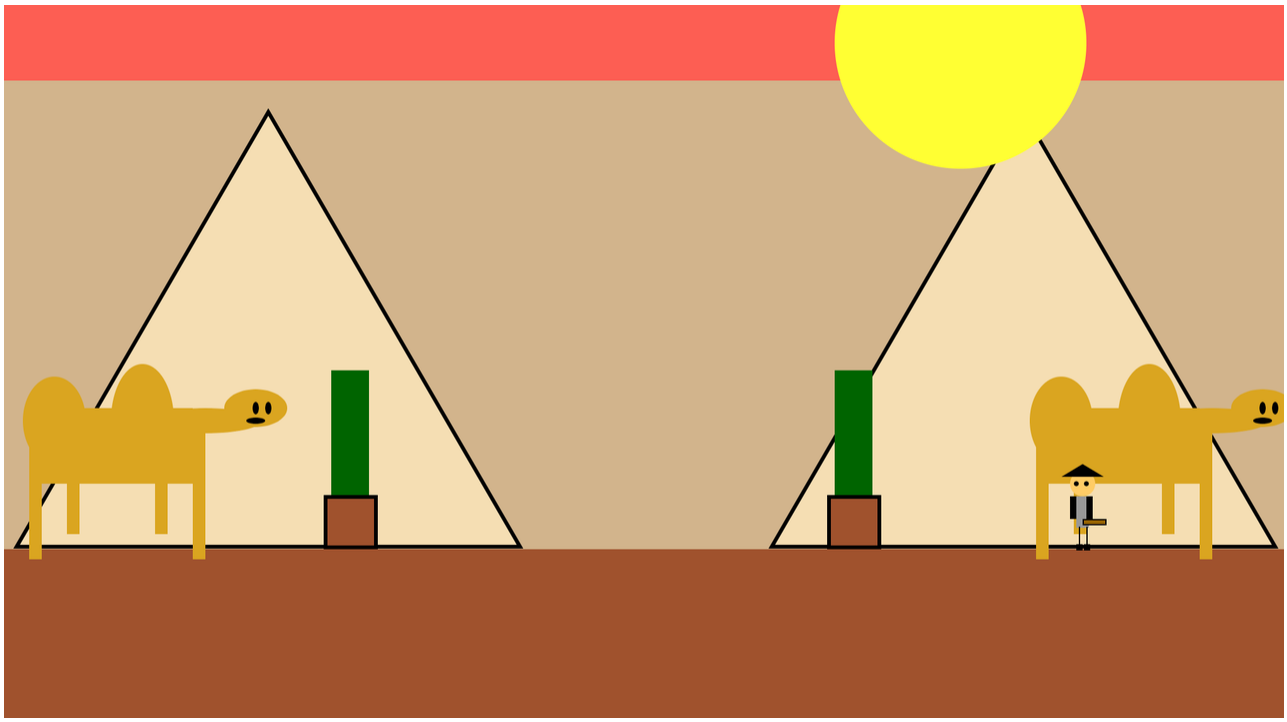
- When watching the movie 'My Neighbour Totoro' by Hayao Miyazaki, I was looking at the infamous rain bus stop scene and thought that could be a cool animation since the woodland creature Totoro appears out of nowhere, I could possibly make an animation where when the sun goes down Totoro shows on the canvas.

https://youtu.be/8aWPsKvpcco

- I also thought since there is rain in the scene, I could also animate that in an interesting way.

- I've done p5 work in the past where I used if statements to change the look of the canvas so that was also a creative inspiration and I knew exactly how i wanted to creatively execute the project. But with my previous project I had the canvas changed passed on the scroll, with this project I wanted to create an object that represent the sun so when the sun hit a certain x,y position the canvas would change.





Phase 3

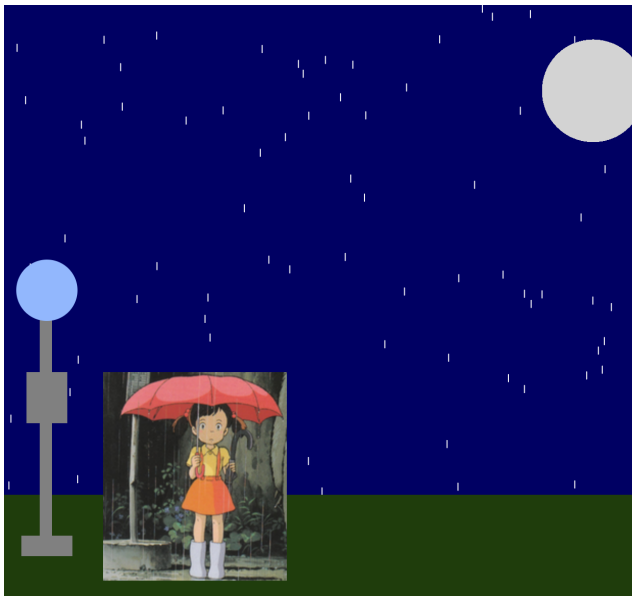
- My plan for the project was to create with p5 shapes the bus stop, the girl and her sister under the umbrella and Totoro, then have the rain falling with the moon/sun being able to move up and down. The sketch below is a very rough idea on how I want the sketch to look.

code requirements:

- Animated sun/moon when it moves down the Totoro character appears.
- Animated rain droplets.

Phase 4

final sketch: https://editor.p5js.org/shardaidawk/sketches/6PdT1uyx9



I started the code with setting variables and in preload loading the image of the little girl, my original idea was to make the little girl with p5 shapes but essentially just settled for a picture because it was to complicated to make and it wasn't going to be able to come out the way I wanted. I also loaded some rain sounds to add to the scenery.

```
let drop = new Array(100);
var x;
var y;
var speed;
let rainsound;

let img;

function preload() {
  img = loadImage('indarain.png');

   rainsound = loadSound('Rain.mp3');
}
```

In set up I added a for loop for the rain drops and looped the rain sound.

```
function setup(){
  createCanvas(620,580);

   for(var i=0;i<drop.length;i++){
    drop[i]=new Drop();
   }

  rainsound.loop();

}
```

In the draw function I created a map function to be able to move the mouseY down and change the scenery. I also added the variables day and night that had colours attached to them, and then a gradient so when mouseY moved down the background colour would gradually change from day to night. I added the rainSunY and SunY variables so that the sun/moon ellipses could be attached the map i've created. I also added a green floor.

```
function draw(){



  //Day/night cycle background colours
  var cycle = map(mouseY, 0, height, 0, 1);
  var night = color(27,30,35);
  var day = color(0,0,100);
  var gradient = lerpColor(day, night, cycle);

  background(gradient);
```

```
for(var i=0;i<drop.length;i++){
  drop[i].rain();


//lighttimerain
var rainSunY = map(mouseY, 0, height, 80, height+75);
noStroke();
fill("lightgrey");
ellipse(580, rainSunY, 100);

//nightimetrain
 var SunY = map(mouseY, 0, height, height+75, 60);
fill('grey');
ellipse(580, SunY, 100);

//green background
fill('#1F3D0C');
rect(0,480,800,100);

}
```

I was able to create the other objects with the p5 objects so believe is code i used to create the bus stop and the Totoro character.

```
 image(img,100, 360);

 //bus-stop
 fill("grey")
```

```
 rect(20,520,50,20);
 rect(38,300,12,220);
 rect(25,360,40,50);
 fill("#92B7FE");
 ellipse(45,280,60,60);

if(mouseY >= 200){

  fill('grey');
 ellipse(400,470,180,220);

  //ears
 ellipse(360,350,30,60);
ellipse(440,350,30,60);

  //arms

  ellipse(320,480,40,110);
  ellipse(480,480,40,110);

  //leaf

  fill("green");
  ellipse(400,350,50,20);
  ellipse(400,330,2,20);


 fill(225,198,153);
 ellipse(400,510,130,120);
 ellipse(400,510,130,120);
```

```
  fill('white');

   ellipse(360,400,30,30);
   ellipse(440,400,30,30);

  fill('black');
   ellipse(440,400,10,10);
  ellipse(360,400,10,10);

  ellipse(400,405,30,10);
  stroke('#222222');
  line(350, 440, 280, 410);
  line(350, 450, 280, 450);
  line(350, 460, 280, 490);


  line(510, 420, 440, 440);
  line(510, 450, 440, 450);
  line(510, 470, 440, 460);
  }
}
```

Then to make for the rain droplets in the background I used a constructor function to set the x, y and speed of the objects and set the objects as a line to give it a rain like effect.

```
class Drop{
constructor() {
```

```
                this.x=random(0,width);
                this.y=random(-100,0);
                this.speed=random(1,10);
        }
  rain(){
    stroke(255,255,255);

    line(this.x,this.y,this.x,this.y+7);
    this.y=this.y+this.speed;
    if(this.y > height){
      this.y = random(-100,0);
    }
  }
}
```

To conclude this project expanded on my animation skills and I did made some changes to the original plan like not being able to create the little girl with the p5 shapes but for the most part I completed the plan quite well.

Project 5 - Fire

Phase 1

- For this last project I just decided to combined multiple technical p5 skills learn't in class such as constructors, animations, sounds, colour modes, simple particles/ angles modes and more.

- So this project was based off a particular skill therefore I just wen't over all the examples made in the online lessons for reference and looked and the project i just made to refresh my memory.

Phase 2

- For this project, there were no particular artist or inspiration in my mind when thinking about what I wanted to produce, I just simply thought of a scenario in which all the technical elements I wanted to incorporate could be included.

- I choose to do a scenery animation with fire because I knew I could produce a variety of different types of code to execute the look.

Phase 3

- Didn't have the time to made a sketch or plan but simply as mentioned before wanted to blend a bunch of the skills together for one project and the theme being fire based.

Phase 4

Final sketch - https://editor.p5js.org/shardaidawk/sketches/S9-9oR76O

As always I start the code off by setting the variables and again I decided to add sound to add affect to the sketch.

```
var dots = [];
var sparks = [];
let a = 100;
let angle = 0
let firesound;
```

```
//cloud variables
var cloudX = [];
var cloudY = [];
var cloudSpeed = [];

function preload() {


   firesound = loadSound('fire.mp3');
}
```

I then added the colorMode function for the fire I wanted to include and a for loop for the dots/sparks to add to the fire animation. I also added a for loop for the clouds I wanted to add at the top of the sketch.

```
function setup() {
  createCanvas(700, 600);

   firesound.loop();

        colorMode(HSB, 360, 100, 100, 100);

        for (let x = 0; x < 10; x++) {
                dots[x] = new Dot();
                sparks[0] = new Spark();
        }


   //Initialise cloud variables
  for(var i = 0; i < 5; i ++){
```

```
    cloudX[i] = random(width);
    cloudY[i] = 75+i*50;
  }

}
```

Below is the fire truck and building object I made using the p5 shapes.

```
function draw() {
  background("#87CEEB");

  fill("lightgrey");
  rect(5,320,80,262);

  rect(85,272,110,310);
  rect(195,320,80,262);
  triangle(5, 272, 280, 272, 140, 90);



  fill("#8b0000");
  rect(400,460,220,100);
  rect(350,430,90,130);
  fill("Black")
  ellipse(430,550,80,80);
  ellipse(570,550,80,80);
  fill("grey");
  ellipse(430,550,40,40);
  ellipse(570,550,40,40);
  fill("#87CEEB");
  rect(370,440,50,50);
```

```
  fill("orange");
  rect(380,410,20,20);
```

This part of the code was for the rotating sparks and dots to add to the for effect, I used the push and pop method, for loops, splice and included angle degrees.

```
//sparkling fire

  noStroke();
  dots.push(new Dot());
        if (frameCount % 5 == 0) {
                sparks.push(new Spark());
        }

        for (let x = 0; x < dots.length; x++) {
                if (dots[x].size <= 0) {
                        dots.splice(x, 1);
                }
        }

        for (let x = 0; x < sparks.lenght; x++) {
                if (sparks[x].y <= 0) {
                        sparks.splice(x, 1);
                }
        }

        for (let x = 0; x < dots.length; x++) {
                dots[x].display();
                dots[x].move();
        }
```

```
        for (let x = 0; x < sparks.length; x++) {
                sparks[x].display();
                sparks[x].move();
        }

  angleMode(DEGREES);
  a = 0.04
```

```
  push();
  translate(random(-100,100),random(-100,100));
  fill(random(0,50),random(0,100),100)
  circle(150,150,5);
  circle(190,190,5);
  pop();


  push();
  stroke(50,random(0,100),100)
  translate(160,160);
  rotate(angle);
  ellipse(0,0,random(50,100),1);
  angle = angle+1000
pop();



    //clouds
  fill(255);
```

```
  //drawing clouds
  for(var i = 0; i < 5; i ++){
    clouds(cloudX[i], cloudY[i]);
    cloudX[i] = cloudX[i] + 0.5;
                //clouds have jitter movement in y-axis
    cloudY[i] = cloudY[i] + random(-0.5, 0.5);

  //when clouds reach beyond edge of screen, clouds reset to o
riginal side
    if (cloudX[i] > width+50) {
      cloudX[i] = -50;
    }
  }

function clouds(x, y){
  ellipse(x, y, 30);
  ellipse(x+10, y, 25);
  ellipse(x+10, y-10, 30);
  ellipse(x+20, y, 25);
  ellipse(x+30, y, 25);

}}
```

```
class Dot {

        constructor() {
                this.start();
        }
```

```
        start() {
                this.x = random(width / 2 - 400, width / 2 + 1
20);
                this.y = height;
                this.size = 100;
                this.velY = random(3, 7);
                this.h = random(0, 50);
                this.a = 50;
        }

        display() {
                fill(this.h, 100, 100, this.a);
                ellipse(this.x, this.y, this.size, this.size);
        }

        move() {
                this.y -= this.velY;
                this.x += cos(this.y) * 2;
                this.size -= 2;
                this.a -= 1.5;
        }
}
```

```
class Spark {

        constructor() {
                this.start();
        }

        start() {
```

```
                this.x = random(width / 2 - 400, width / 2 + 2
5);
                this.y = height + 50;
                this.size = 2;
                this.velY = random(5, 10);
        }

        display() {
                fill(0, 100, 100, 100, 10);
                ellipse(this.x, this.y, this.size, this.size);
        }

        move() {
                this.y -= this.velY;
                this.x += cos(this.y) * 1;
        }
}
```

This project was probably my most complicated and longest and there were definitely some technical issues along the way, but were quickly solved with online help. I felt as if i completed my wishes for this project which was to try to incorporate as much as the technical skills I learn't into one final project. I didn't do much code explaining as I'm sort of repeating myself since these are bits of code that I've talked about doing the previously project.