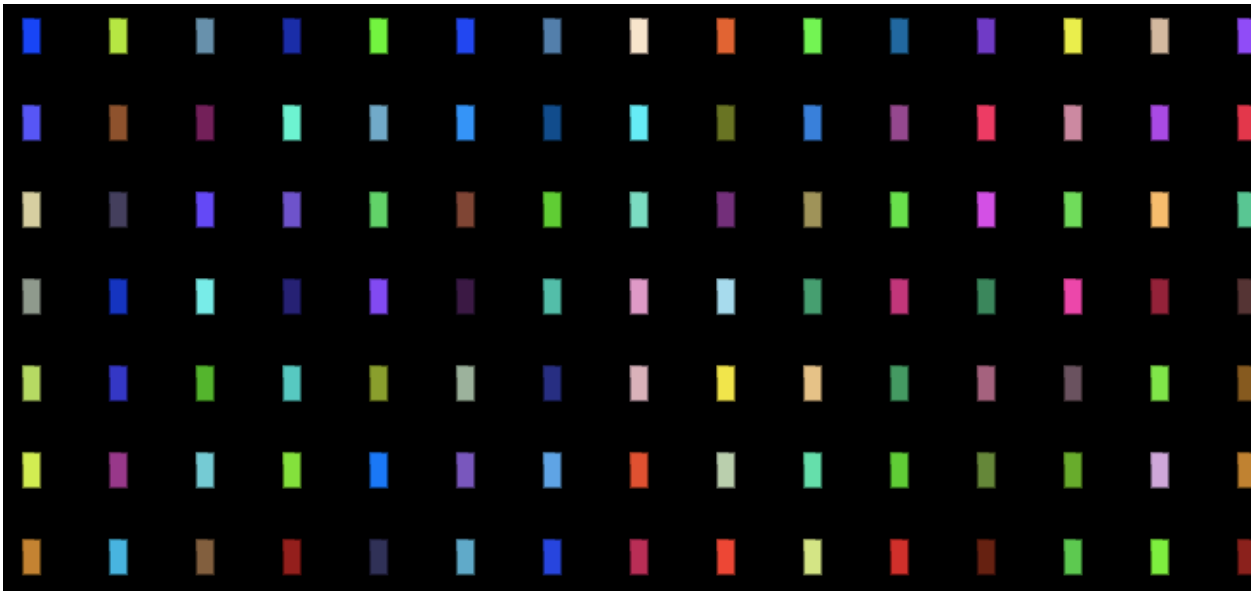
Midterm Reflection/Documentation:

For this project, I was initially inspired by the movie Bird Box. Solely because of how the movie speaks about mental health in such a subtle yet obvious way. The movie is revolved around a group of characters who all have to remain blinded folded as there is some sort creature who if they look at in the eye, they will immediately kill themselves and those around them. As the movie progresses you begin to realize that the only creature that exists, is the so called “demon” that lives in your head, which are your worst thoughts about yourself and your surroundings, it focuses on how you are your biggest critique and no one is going to be as harsh on you as you are on yourself, thus they show how the biggest monster in your head is you. That was basically the synopsis of the movie and it had me wondering about how much this is related to real life scenario of suicide.

When we think of suicide, you envision one person who is so deep into their mental illness that had to unfortunately take their own lives, yet we do not tend think about the mass of people that going through the exact same thing, instead, the majority think we’re alone in this world. According to WHO, in every forty seconds there is someone taking their own lives. This was the main fact I wanted to incorporate into my project, however as I thought about it, I realized I did not just want to show whom it affects or how many people it affects but the process for an individual to reach the point of having suicidal thoughts and committing suicide.



```
function createCell( xOff, yOff, x, y, speed, unit ){

  let cell = {

    xOff: xOff,
    yOff: yOff,
    x: x,
    y: y,
    speed: speed,
    unit: unit,
    xDir: 10,
    yDir: 10,
    lifeSpan: 400,
    update:update,
    draw:display

  }

  return cell



}

function update() {
    this.x = this.x + this.speed * this.xDir;
    if (this.x >= this.unit || this.x <= 1) {
      this.xDir *= -2;
      this.x = this.x + 1 * this.xDir;
      this.y = this.y + 1 * this.yDir;
    }
    if (this.y >= this.unit || this.y <= 0) {
      this.yDir *= -1;
      this.y = this.y + 1 * this.yDir;
    }
  }

function display() {
  console.log('draw')
  this.lifeSpan--
    fill(random(255),random(255),random(255),this.lifeSpan);
    // console.log(this.xOff)
    rect(this.xOff + this.x, this.yOff + this.y, 6, 6);
     rect(this.xOff + this.x, this.yOff + this.y, 10, 20);
}


let unit = 50;
```

```
let unit = 50;
let count;
let mods = [];
let highCount, wideCount

function setup() {
  createCanvas(720, 360);
  noStroke();
  wideCount = width / unit;
  highCount = height / unit;
  count = wideCount * highCount;

  let index = 0;
  for (let y = 0; y < highCount; y++) {

    let col = []
    for (let x = 0; x < wideCount; x++) {

      let mod = createCell(
        x * unit,
        y * unit,
        unit / 4,
        unit / 4,
        random(0.01, 0.02),
        unit
      )
      col.push(mod)
    }
    mods.push(col)
  }
}

function draw() {
  background(0);

  for (let y = 0; y < highCount; y++) {
    for (let x = 0; x < wideCount; x++) {
// console.log(mods[y][x])
      let mod = mods[y][x]
      let neighBoor = mods[y+1][x]
      if (mod){
        fill(random(255))
        // ellipse(mod.x+mod.xOff,mod.y+mod.yOff,10,10)
        // mod.update()
        mod.draw()
      } else {
        console.log(x,y)
```

Code is below:

```
    for (let x = 0; x < wideCount; x++) {

      let mod = createCell(
        x * unit,
        y * unit,
        unit / 4,
        unit / 4,
        random(0.01, 0.02),
        unit
      )
      col.push(mod)
    }
    mods.push(col)
  }
}

function draw() {
  background(0);

  for (let y = 0; y < highCount; y++) {
    for (let x = 0; x < wideCount; x++) {
// console.log(mods[y][x])
      let mod = mods[y][x]
      let neighBoor = mods[y+1][x]
      if (mod){
        fill(random(255))
        // ellipse(mod.x+mod.xOff,mod.y+mod.yOff,10,10)
        // mod.update()
        mod.draw()
      } else {
        console.log(x,y)

      }
      // console.log(mod)

      // fill(255,255,0)

      // ellipse(mod.x,mod.y,5,5)


    }
  }
  // for (let i = 0; i < count; i++) {
  //   mods[i].update();
  //   mods[i].draw();
  // }
}
// source: https://p5js.org/examples/arrays-objects.html
```

The image above shows my first sketch, which represents my first idea of having a building complex, and when the clock hits forty seconds, a light switches off. However,

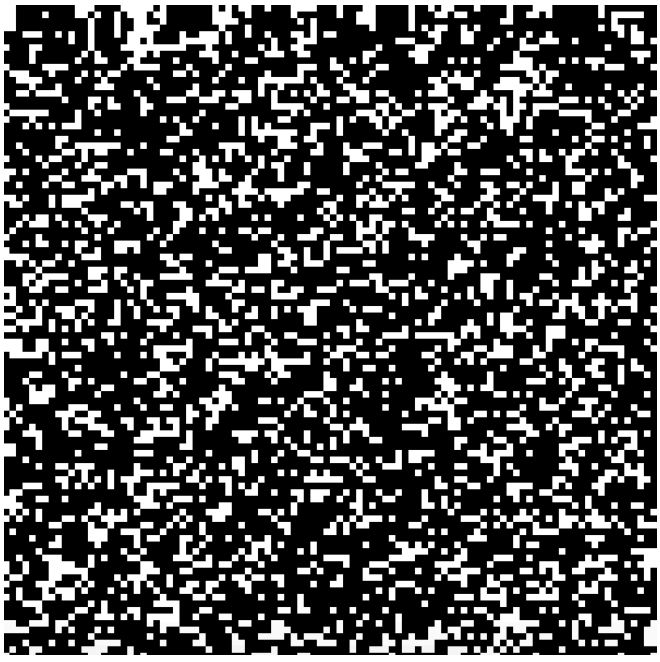
in this case, all the windows begin to fade into black which represents the process of being engulfed by a mental illness to the extent of suicide.

After this sketch, I was introduced to the Game of Life by John Horton Conway. Which basically is a zero-player game which have the rules of:

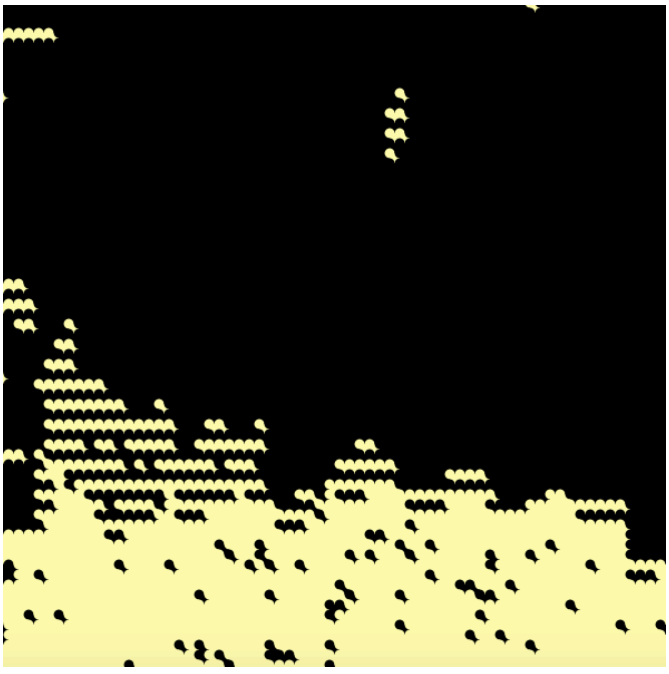
1. Any cell with fewer than two live neighbors dies, by loneliness.
2. Any cell with two or three live neighbors lives continues to the next generation.

However I altered the rules based on my scenario to be:

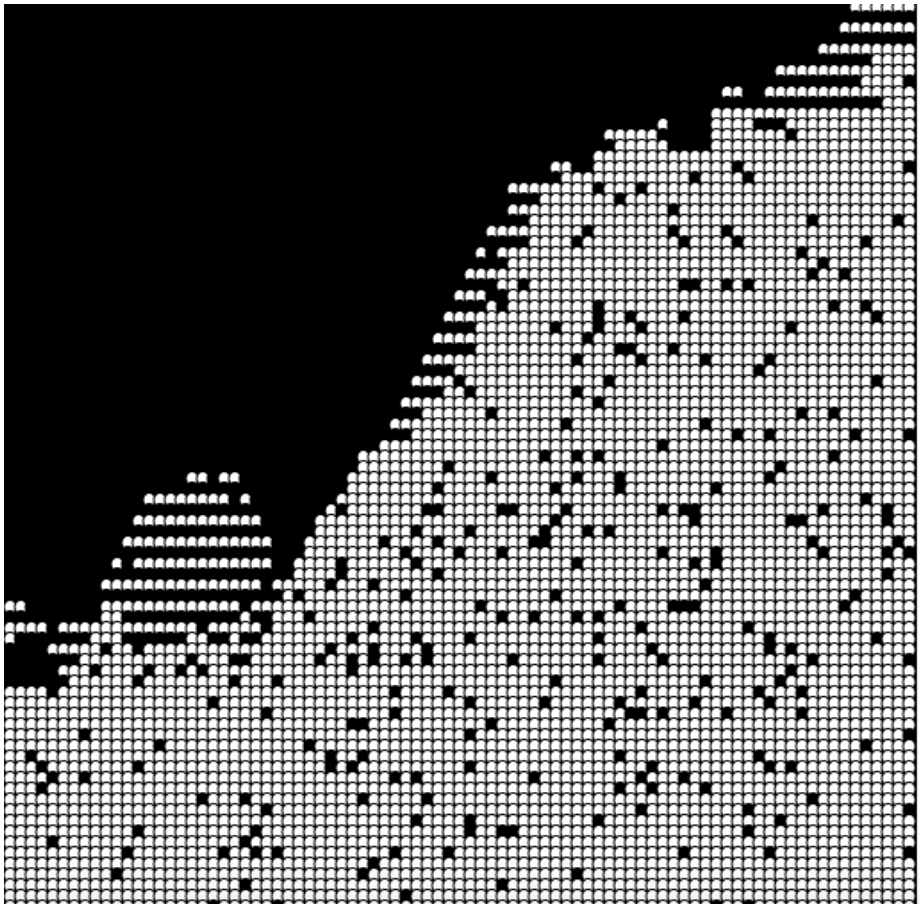
1. Any cell with fewer than one or two neighbors dies but from suicide, they will be engulfed with their depression which in the case of my sketches are the black cells, those who live will receive a new neighbor.
2. Black cells will eat up those who are live, none will survive as it is inevitable.
3. A wave of black cells will kill the remaining cells alive, those who survive are either lonely or have two to three neighbors to accompany them. This represents people who have overcome their mental illnesses.



Example of Rule 1



Example of Rule 2



Example of Rule 3

Code for Rule 3:

```
function make2DArray (cols, rows){

  let array = []

  for (var i = 0; i < cols; i++){

    let row = []

    for (var j = 0; j < rows; j++){

      let obj = {

        x:i,
        y:j,
        state: random(0.05)

      }

      row.push(obj)

    }

      array.push(row)
  }

  // console.log(array[4][3])

  // let i = 3
  // let j = 3


  // let cell = array[i][j]
  // n2 = array[i+1][j]
  //  n1 = array[i-][j]
  // let arr = new Array(cols);
  // for (let i =0; i < arr.length; i++) {
  //   arr [i] = new Array(rows);

  // }

  return array

}

let grid;
let cols;
let rows;
let resolution = 7;
```

```
let resolution = 7;


function setup(){
  createCanvas(600,600);
  cols = width / resolution;
  rows = height / resolution;
  grid = make2DArray(cols, rows);



}

function draw(){
  background(0)

  grid = eat()
  display()



}

function display(){

  for (var x = 0; x < grid.length; x++){

    let row = grid[x]

    for (var y = 0; y < row.length; y++){

      let cell = grid[x][y]

      fill(255)
      stroke(0)


      if (cell.state){
        fill(0)
      }

    ellipse(cell.x*resolution,cell.y*resolution,8,9)




    }
```

```
    }
  }
}

function eat(){

  let newGrid = []

  for (var i = 0; i < grid.length; i++){

    let newRow = []
    let row = grid[i]

    for (var j = 0; j < row.length; j++){

      let score = 0
      let cell = grid[i][j]

      let a =  constrain(i-1, 0, grid.length);
      let b =  constrain(i+1, 0, grid.length);
      let c =  constrain(j-1, 0, row.length);
      let d =  constrain(j+1, 0, row.length);



     // let n1 = grid[a][c]
      //let n2 = grid[a][j]
      let n3 = grid[a][d]
      // console.log(n1)

      let n4 = grid[i][c]
      let n6 = grid[i][d]

      let n7 = grid[a][j]
      let n8 = grid[a][c]
      // let n9 = grid[b][d]

      let neighbours = [n7,n8,n3,n4,n6]

      for (var r = 2; r < neighbours.length; r++){
        if (neighbours[r]){
          score += neighbours[r].state
        }

      }
```

```
      }
      // console.log(score,a,b,c,d)

      let obj = {

        x:i,
        y:j,

      }
      if (score > 1){
        obj.state = true
      } else if (random()<0.05) {
        obj.state = true

      } else {

        obj.state = false

      }


      newRow.push(obj)


    }
    newGrid.push(newRow)

  }

  return newGrid

}
// function draw(){
```

Code for Rule 1:

```
function make2DArray (cols, rows){

  let array = []

  for (var i = 0; i < cols; i++){

    let row = []

    for (var j = 0; j < rows; j++){

      let obj = {

        x:i,
        y:j,
        state: (random()>0.5)

      }

      row.push(obj)

    }

      array.push(row)
  }

  // console.log(array[4][3])

  // let i = 3
  // let j = 3


  // let cell = array[i][j]
  // n2 = array[i+1][j]
  //  n1 = array[i-][j]
  // let arr = new Array(cols);
  // for (let i =0; i < arr.length; i++) {
  //   arr [i] = new Array(rows);

  // }

  return array

}

let grid;
let cols;
let rows;
let resolution = 6;
```

```
let resolution = 6;


function setup(){
  createCanvas(600,600);
  cols = width / resolution;
  rows = height / resolution;
  grid = make2DArray(cols, rows);



}

function draw(){
  background(255)

  grid = eat()
  display()



}

function display(){

  for (var x = 0; x < grid.length; x++){

    let row = grid[x]

    for (var y = 0; y < row.length; y++){

      let cell = grid[x][y]

      fill(255)
      strokeWeight(0)


      if (cell.state){
        fill(0)
      }

    rect(cell.x*resolution,cell.y*resolution,resolution,resolution)



    }
```

```
    }

  }

}

function eat(){

  let newGrid = []

  for (var i = 0; i < grid.length; i++){

    let newRow = []
    let row = grid[i]

    for (var j = 0; j < row.length; j++){

      let score = 0
      let cell = grid[i][j]

      let a =  constrain(i-1, 0, grid.length);
      let b =  constrain(i+1, 1, grid.length);
      let c =  constrain(j-1, 2, row.length);
      let d =  constrain(j+1, 4, row.length);



      let n1 = grid[a][c]
      let n2 = grid[a][j]
      let n3 = grid[a][d]
      // console.log(n1)

      let n4 = grid[i][c]
      let n6 = grid[i][d]

      //let n7 = grid[a][j]
      let n8 = grid[a][c]
       //let n9 = grid[b][d]

      let neighbours = [n1,n2,n3,n4,n8]

      for (var r = 3; r < neighbours.length; r++){
        if (neighbours[r]){
          score += neighbours[r].state
        }

      }
```

Code for Rule 2:

```
function make2DArray (cols, rows){

  let array = []

  for (var i = 0; i < cols; i++){

    let row = []

    for (var j = 0; j < rows; j++){

      let obj = {

        x:i,
        y:j,
        state: (random()>0.5)

      }

      row.push(obj)

    }

      array.push(row)
  }

  // console.log(array[4][3])

  // let i = 3
  // let j = 3


  // let cell = array[i][j]
  // n2 = array[i+1][j]
  //  n1 = array[i-][j]
  // let arr = new Array(cols);
  // for (let i =0; i < arr.length; i++) {
  //   arr [i] = new Array(rows);

  // }

  return array

}

let grid;
let cols;
let rows;
let resolution = 9;

function setup(){
  createCanvas(600,600);
  cols = width / resolution;
  rows = height / resolution;
  grid = make2DArray(cols, rows);



}

function draw(){
  background(255)

  grid = eat()
  display()



}
function display(){

  for (var x = 0; x < grid.length; x++){

    let row = grid[x]

    for (var y = 0; y < row.length; y++){

      let cell = grid[x][y]

      fill(255,255,174)
      stroke(0)
      strokeWeight(0)


      if (cell.state){
        fill(0)
      }

    rect(cell.x*resolution,cell.y*resolution,resolution,resolution)
     ellipse(cell.x*resolution,cell.y*resolution,resolution,resolution)



    }

  }
```

```
142
143         let obj = {
144
145           x:i,
146           y:j,
147
148         }
149         if (score < 2){
150           obj.state = true
151         } else if (random()<0.5) {
152           obj.state = true
153
154         } else {
155
156           obj.state = false
157
158         }
159
160
161         newRow.push(obj)
162
163
164       }
165       newGrid.push(newRow)
166
167     }
168
169     return newGrid
170
171   }
172   // function draw(){
173   //   background(0);
174
```

References:

World Health Organization. (2019). Suicide across the world (2016). [online] Available at: https://www.who.int/mental_health/prevention/suicide/suicideprevent/en/ [Accessed 20 Oct. 2019].