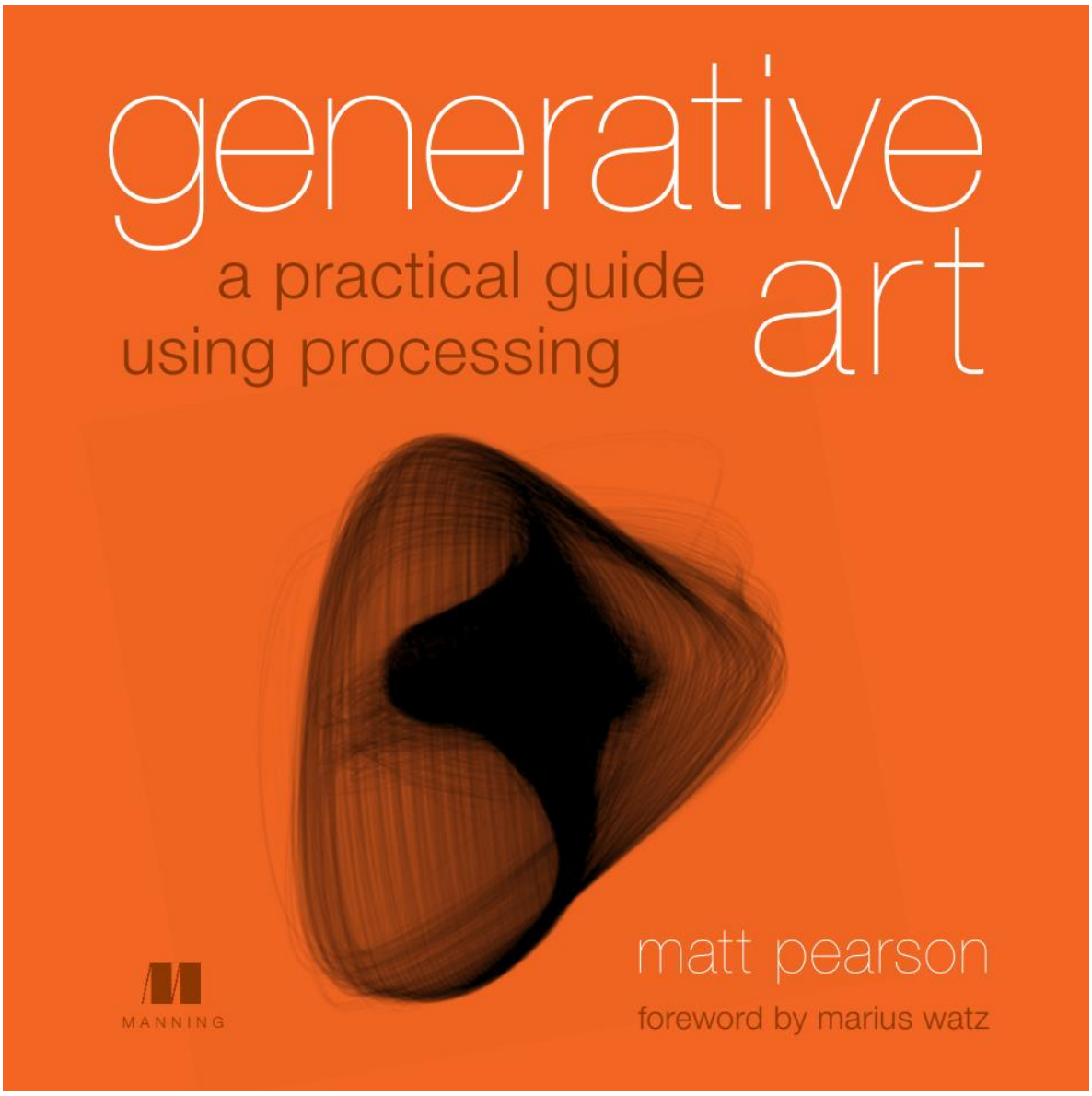
Practica 5, Interpretar código



Esta práctica consiste en interpretar de el libro Generative Art de Matt Pearson alguno de los códigos que encontraras en las siguientes páginas:

- Código 1 (Dificultad moderada). **Listing 5.6** Constructing a cube of three-dimensional noise. Páginas 97 y 98

Listing 5.6 **Constructing a cube of three-dimensional noise**

```
float xstart, ystart, zstart;
float xnoise, ynoise, znoise;

int sideLength = 200;
int spacing = 5;

void setup() {
 size(500, 300, P3D);
 background(0);
 noStroke();

 xstart = random(10);
 ystart = random(10);
 zstart = random(10);
}

void draw () {
 background(0);

 xstart += 0.01;
 ystart += 0.01;
 zstart += 0.01;
```

Changes renderer to P3D

(continued on next page)

(Listing 5.6 continued)

Reposition to get a better perspective

Three loops now

```
xnoise = xstart;
ynoise = ystart;
znoise = zstart;

translate(150, 20, -150);
rotateZ(frameCount * 0.1);
rotateY(frameCount * 0.1);

for (int z = 0; z <= sideLength; z+=spacing) {
 znoise += 0.1;
 ynoise = ystart;
 for (int y = 0; y <= sideLength; y+=spacing) {
  ynoise += 0.1;
  xnoise = xstart;
  for (int x = 0; x <= sideLength; x+=spacing) {
   xnoise += 0.1;
   drawPoint(x, y, z, noise(xnoise, ynoise, znoise));
  }
 }
}
}

void drawPoint(float x, float y, float z, float noiseFactor) {
 pushMatrix();
 translate(x, y, z);
 float grey = noiseFactor * 255;
 fill(grey, 10);
 box(spacing, spacing, spacing);
 popMatrix();
}
```

- Código 2 (Dificultad media). **Listing 6.4** Object-oriented circle-drawing code with movement. Páginas 117-119

Listing 6.4 **Object-oriented circle-drawing code with movement**

```
int _num = 10;
Circle[] _circleArr = {};                                   Define array of circles
void setup() {
 size(500,300);
 background(255);
 smooth();
 strokeWeight(1);
 fill(150, 50);
 drawCircles();
}

void draw() {
 background(255);
 for (int i=0; i<_circleArr.length; i++) {
  Circle thisCirc = _circleArr[i];
  thisCirc.updateMe();
 }
}

void mouseReleased() {
 drawCircles();
}

void drawCircles() {
 for (int i=0; i<_num; i++) {
  Circle thisCirc = new Circle();
  thisCirc.drawMe();
  _circleArr = (Circle[])append(_circleArr, thisCirc);      Add object to array
 }
```

(continued on next page)

```
}                                                          (Listing 6.4 continued)


//================================== objects

class Circle {

 float x, y;
 float radius;
 color linecol, fillcol;
 float alph;
 float xmove, ymove;

 Circle () {
  x = random(width);
  y = random(height);
  radius = random(100) + 10;
  linecol = color(random(255), random(255), random(255));
  fillcol = color(random(255), random(255), random(255));
  alph = random(255);
  xmove = random(10) - 5;
  ymove = random(10) - 5;
 }

 void drawMe() {
  noStroke();
  fill(fillcol, alph);
  ellipse(x, y, radius*2, radius*2);
  stroke(linecol, 150);
  noFill();
  ellipse(x, y, 10, 10);
 }

 void updateMe() {
  x += xmove;
  y += ymove;
  if (x > (width+radius)) { x = 0 - radius; }
  if (x < (0-radius)) { x = width+radius; }
  if (y > (height+radius)) { y = 0 - radius; }
  if (y < (0-radius)) { y = height+radius; }
```

Steps to move every frame

Random step

Move every frame

Wrap position at stage edges


```
  drawMe();
 }
}
```

Draw it

- Código 3 (Dificultad alta). **Listing 8.3** Final listing for the cog fractals. Páginas 168-170.

The complete code for this final version is in the following listing.

Listing 8.3 Final listing for the cog fractals

```
int _numChildren = 7;
int _maxLevels = 7;

Branch _trunk;

void setup() {
 size(750,500);
 background(255);
 noFill();
 smooth();
 newTree();
}

void newTree() {
 _trunk = new Branch(1, 0, width/2, height/2);
 _trunk.drawMe();
}

void draw() {
 background(255);
 _trunk.updateMe(width/2, height/2);
 _trunk.drawMe();
}

class Branch {
 float level, index;
 float x, y;
 float endx, endy;

 float strokeW, alph;
 float len, lenChange;
 float rot, rotChange;
```


```
  Branch[] children = new Branch[0];

  Branch(float lev, float ind, float ex, float why) {
   level = lev;
   index = ind;

   strokeW = (1/level) * 10;
   alph = 255 / level;
   len = (1/level) * random(500);
   rot = random(360);
   lenChange = random(10) - 5;
   rotChange = random(10) - 5;

   updateMe(ex, why);

   if (level < _maxLevels) {
    children = new Branch[_numChildren];
    for (int x=0; x<_numChildren; x++) {
     children[x] = new Branch(level+1, x, endx, endy);
    }
   }
  }

 void updateMe(float ex, float why) {
   x = ex;
   y = why;

   rot += rotChange;
   if (rot > 360) { rot = 0; }
   else if (rot < 0) { rot = 360; }

   len -= lenChange;
   if (len < 0) { lenChange *= -1; }
   else if (len > 500) { lenChange *= -1; }

   float radian = radians(rot);
   endx = x + (len * cos(radian));
   endy = y + (len * sin(radian));
```

(Continued on next page)

(Listing 8.3 continued)

```
  for (int i=0; i<children.length; i++) {
   children[i].updateMe(endx, endy);
  }
 }

 void drawMe() {
```

Don't draw trunk

```
  if (level > 1) {
   strokeWeight(strokeW);
   stroke(0, alph);
   fill(255, alph);
   line(x, y, endx, endy);
   ellipse(endx, endy, len/12, len/12);
  }
  for (int i=0; i<children.length; i++) {
    children[i].drawMe();
  }
 }

}
```

El objetivo de esta práctica es ver como podemos adaptar códigos realizados en un lenguaje de programación (processing), al lenguaje que estamos utilizando actualmente p5js.

Este ejercicio es importante porque de este libro podemos tomar para adaptar muchos recursos interesantes para nuestros propios proyectos.

Instrucciones: selecciona uno de los códigos; crea un proyecto de p5js, adapta el código; has algunas modificaciones personales (aprópiatelo) y has un video de menos de un minuto donde muestres el resultado.

La fecha límite para publicarlo es el martes 24 por la noche antes de las 22:59 h.

Nota. Si alguien tiene problemas con la IDE de Brackets usar openProcessing.