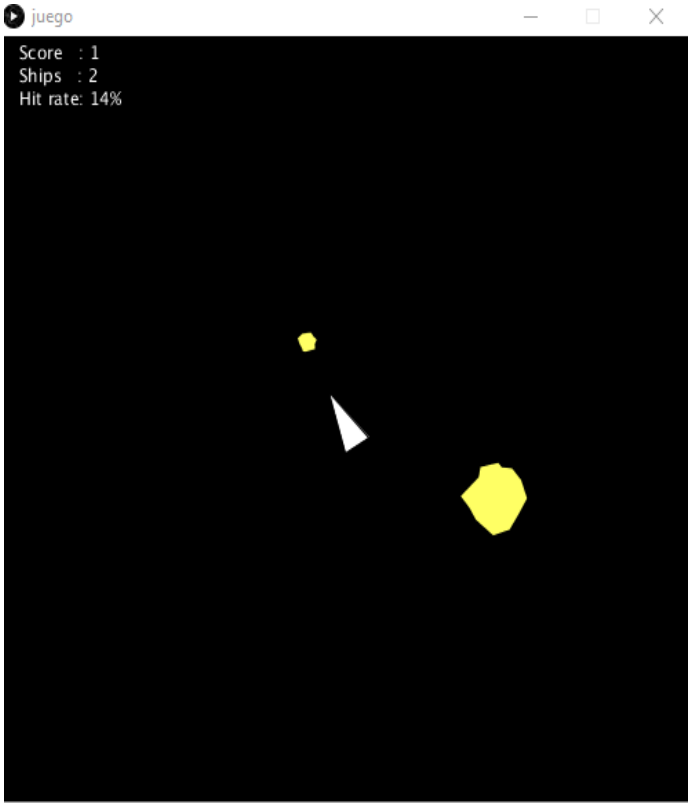

JUEGO DEL ASTEROIDE

PROCESSING

Ma. De Lourdes Rodríguez
Víctor Echeverría Morales
| GRAFICACIÓN | 28 de mayo 2018

PROCESSING

Processing es un lenguaje de programación y entorno de desarrollo integrado de código abierto basado en Java, de fácil utilización, y que sirve como medio para la enseñanza y producción de proyectos multimedia e interactivos de diseño digital.

- Uno de los objetivos declarados de Processing es el de actuar como herramienta para que artistas, diseñadores visuales y miembros de otras comunidades ajenos al lenguaje de la programación

JUEGO DEL ASTEROIDE CODIGO

En este programa que investigamos tiene líneas 305 de código, donde varias de estas funciones ya habíamos aprendido en clase.

Mencionaremos cual son las que aprendimos y describiremos su función de cada una de ellas.

```
 5
 6 int astroid_rate = 2 * 60;
 7 int astroid_count = 0;
 8 float ast_size = 10;
 9 int ast_id = 1;
10 int score = 0;
11 float hitRate = 0;
12 int numShots = 0;
13 int ships = 3;
14
15 int pause = 0;
16
```

Declaración de variables donde es asignado el tiempo de cuántos segundos hay entre cada nuevo astroide (3 segundos = 3 * 60)

Tambien se ve el tamaño en pixel del astroid nominal

```
19
20 void setup () {
21   frameRate(60);
22   size(500, 500);
23   stroke(0);
24   fill(255);
25 }
26
```

Define tamaño de pantalla

Define color

Unas de las funciones que aprendimos en void setup () { contienen declaraciones, o instrucciones, donde cierran bloques. En donde setup() es la parte encargada de recoger la configuración y loop() es la que contiene el programa que se ejecutará cíclicamente (de ahí el término loop –bucle-).

```
29 {
30   int i;
31  float angle = atan2(mouseY - 250, mouseX - 250);
32  if (pause==0) {
33
34
```

La función mouse se invoca después de presionar un botón del mouse y luego liberarlo
Los eventos de mouse encuentra el ángulo de x = 250, y = 250 al mouse

```
34
35  if (astroid_count--==0) {
36    astroids.add(new astroid(random(0, TWO_PI), random(0.1, 2.5), random(0.5, 4), random(-0.1, 0.1),
37      random(-150, 150), random(-150, 150), ast_id++));
38    astroid_count = astroid_rate--;
39  }
40
```

Indica que habrá 1 nuevo astroide cada 5 segundos (60 fps * 4 seg)

La función random () , devuelve un valor inesperado dentro del rango especificado.

```
42
43    background(0);
44    for (i = 0; i<astroids.size(); i++) {
45      astroid a = astroids.get(i);
46      if (a.update()) {
47        astroids.remove(i);
48      }
49
50      if (a.coll(250, 250, 6, -1) ||
51        a.coll(13*cos(angle-PI)+250, 13*sin(angle-PI)+250, 9, -1) ||
52        a.coll(10*cos(angle)+250, 10*sin(angle)+250, 4, -1) ||
53        a.coll(18*cos(angle)+250, 18*sin(angle)+250, 1, -1)) {
54        ships--;
55        pause=3*60;
56      }
57    }
```

Pantalla oscura

Remove los asteroides tocados

Establece elipses con su Angulo

Establece un for para Revisar todos los astroides (si las hay) y actualiza su posición.

```
63
64
65      pushMatrix();
66      translate(250, 250);
67      rotate(angle);
68      fill(255);
69      triangle(20, 0, -20, -10, -20, 10
70      popMatrix();
71    } else {
72      background(0, 10);
73
```

La función pushMatrix () guarda el sistema de coordenadas actual en la pila y popMatrix () restaura el sistema de coordenadas anterior también Si se llama a translate () dentro de draw () , la transformación se restablece cuando el ciclo comienza de nuevo. Esta función puede controlarse aún más utilizando pushMatrix () y popMatrix () .

```
126
127 void mousePressed() {
128   if (pause==0) {
129     if (mouseButton == LEFT) {
130       float angle = atan2(mouseY - 250, mouseX - 250);
131       shots.add(new shot(angle, 4));
132       numShots++;
133     }
134     if (mouseButton == RIGHT) {
135       astroids.add(new astroid(random(0, TWO_PI), random(0.1, 2.0), random(0.5, 4), random(-0.1, 0
136         random(-80, 80), random(-80, 80), ast_id++));
137     }
```

Se agrega disparos cuando estás en acción

Cuando se presiona el botón izquierdo del mouse, crea una nueva toma

void mousePressed ()

```
253
254     pushMatrix();
255     translate(x, y);
256     rotate(rotation);
257     scale(size);
258     shape(s, 0, 0);
259     popMatrix();
260
261     if (x<-300 || x>800 || y<-300 || y>800) {
262       return true;
263     } else {
264       return false;
265     }
```

En estas funciones como

pushMatrix // Establecer posición como el nuevo 0,0

translate // Gira la pantalla "ángulo" que el jugador decida

CONCLUCIONES

En mi punto de vista trabajar en esta plataforma de processing me agrado mucho porque va mucho de la mano con el entorno de desarrollo y lenguaje de programación en JAVA ya que es muy fácil de aprender y solo necesitamos imaginación para dibujar algún objeto .

Una ventaja que teníamos era crear clases de una forma muy sencilla.

Lourdes Rodriguez Villegas