Universidad Nacional de General Sarmiento
Instituto de Industria

UNGS
Estudiantes de Sistemas

Trabajo Práctico

Attack on titans

Final Season

Integrantes: Maximiliano Massa
Martín Gamboggi
Julián Kidjekouchian

E-mails: martin_gamboggi@hotmail.com
maxi_massa@hotmail.com
juliankidje@hotmail.com

Introducción:

El presente trabajo, consta de un videojuego creado en lenguaje Java y el IDE utilizado para dicho fin fue Eclipse V.2022-03.

El juego se basa en un personaje llamado “Mikasa” que lucha en un pueblo contra los “Kyojines” , titanes que intentan destruir la ciudad y matar a Mikasa.

La Heroína de dicho juego debe evitar que los Kyojines la alcanzan o se les descontará una vida de las 3 que posee, para combatir a estos titanes, ella se vale de un misil que podrá disparar contra ellos. Ella sólo puede disparar un misil por vez, si llegara a fallar, no podrá volver a disparar uno nuevo hasta que el actual salga de pantalla. Además se vale de un suero especial que la transforma en una “Kyojina” y al tocar a un titán lo eliminará automáticamente.

La protagonista deberá esquivar los diferentes obstáculos de la pantalla tratando de sobrevivir a los titanes. Cada determinado tiempo, aleatoriamente aparecerá un “Life Up” que le otorgará una vida más a Mikasa en caso de que ya haya perdido alguna de las 3 con las que comenzó.

El juego finalizará y ganará el jugador cuando haya matado a 10 titanes o hayan transcurrido 60 segundos y Mikasa siga con vida. En caso contrario el jugador perderá la partida.

Descripcion:

En el presente juego se implementaron las siguientes clases:

- Fondo{
  Variables de instancias:

  ```
  private double x;
  private double y;
  private Image imgFondo;
  double anguloFondo; }
  ```

  Dichas Variables asignan una posición en el eje X y en el eje Y, carga la imagen que se utilizará y el angulo de dicha imagen.
- Mikasa {
  Variables de instancias:

  ```
  double x;
  double y;
  double angulo;
  Image img1;
  Image img2;
  Image img3;
  Image img4;
  boolean giro;
  boolean transformada; }
  ```

  Las variables asignarán las imágenes para Mikasa, le darán una posición en X y en Y y un ángulo, también habrá un boolean que servirá de bandera para saber si tomó la poción o no.
- Kyojin{
  Variables de instancias:
  public boolean choco = true;

  ```
  private double x;
  private double y;
  private double velocidad;
  private double angulo;
  private double ancho;
  private double alto;
  private int radio;
  ```

  cada Kyojin tendrá una posición en el entorno con un X y un Y específico, tendrán una velocidad, y un ángulo de inclinación. Además de un ancho, alto y radio de la imagen.

Descripcion:

En el presente juego se implementaron las siguientes clases:

- Fondo{
  Variables de instancias:

  ```
  private double x;
  private double y;
  private Image imgFondo;
  double anguloFondo; }
  ```

  Dichas Variables asignan una posición en el eje X y en el eje Y, carga la imagen que se utilizará y el angulo de dicha imagen.
- Mikasa {
  Variables de instancias:

  ```
  double x;
  double y;
  double angulo;
  Image img1;
  Image img2;
  Image img3;
  Image img4;
  boolean giro;
  boolean transformada; }
  ```

  Las variables asignarán las imágenes para Mikasa, le darán una posición en X y en Y y un ángulo, también habrá un boolean que servirá de bandera para saber si tomó la poción o no.
- Kyojin{
  Variables de instancias:
  public boolean choco = true;

  ```
  private double x;
  private double y;
  private double velocidad;
  private double angulo;
  private double ancho;
  private double alto;
  private int radio;
  ```

  cada Kyojin tendrá una posición en el entorno con un X y un Y específico, tendrán una velocidad, y un ángulo de inclinación. Además de un ancho, alto y radio de la imagen.

- public class Misil {
    //Variables de instancias

    private Mikasa mikasa;
    double x;
    double y;
    double angulo;
    boolean giro;
    Image imgMisil;
    Image imgMisil2;
    String sonidoMisil;
    Clip clip
    private int radio;

  La clase misil, crea un objeto del tipo misil, que toma las coordenadas y el ángulo de Mikasa, así también tiene 2 imágenes que se cargan de acuerdo al sector hacia donde se disparó.

- public class Obstaculos {
    double x;
    double y;
    double ancho;
    double alto;
    Image imgCasas;
    Image imgCasaFuego;
    boolean fuego;
    Clip clip;
    Clip clip2;
    String explosion;
    String incendio;

  Los obstáculos son casas que poseen una posición fija en el juego y si son alcanzadas por un misil debido a un error de Mikasa ejecutarán un sonido de explosión y un sonido de incendio.

- public class Pocion {
    double x;
    double y;
    Image img;
    String sonidoPocima;
    Clip clip;

  El objeto poción se creará aleatoriamente en el entorno recibiendo una posición y cuando sea interceptada por Mikasa ejecutará un sonido.

- public class Vida {
    double x;
    double y;
    Image img;
    String sonidoVida;

    Clip clip;

Vida es un objeto que existirá durante un tiempo aleatorio en una coordenada aleatoria, y su función es ser interceptada por Mikasa y ejecutar un sonido.

- public class Gameover {

    private double x;
    private double y;
    private Image imgGameover;
    double anguloGameover;
    String sonidoGameover;
    Clip clip;

  GameOver es una clase que se ejecutará sólo cuando el participante pierda y cargará una imagen a pantalla completa con la puntuación del jugador y un sonido de finalización de juego.

- public class Ganaste {
    private double x;
    private double y;
    private Image imgFondo;
    double anguloGanaste;
    String sonidoGanaste;
    Clip clip;

Ganastes es una clase que aparecerá al final del juego y su finalidad es cargar una imagen y un sonido cuando el jugador logre ganar el juego.

Observación: El mayor problema encontrado fue lograr que los objetos que eran controlados aleatoriamente como los “Kyojines” no se salieran del entorno, esquivaran los obstáculos y persiguieran a Mikasa, para solucionar dicho problema estuvimos trabajando mucho con las funciones matemáticas de ofrece Java y realizando operaciones con los ángulos que dichas operaciones entregaban.

Implementación del código:

```java
package juego;

import java.awt.Image;

import entorno.Entorno;
import entorno.Herramientas;

public class Fondo {

    private double x;
    private double y;
    private Image imgFondo;
    double anguloFondo;

    public Fondo(double x, double y) {
        this.x = x;
        this.y = y;
        imgFondo = Herramientas.cargarImagen("fondo.jpg");
        anguloFondo=0;
    }
    public void dibujarse(Entorno entorno)
    {
        // El numero es el tamaño de la imagen
        entorno.dibujarImagen(imgFondo, this.x, this.y, this.anguloFondo, 0.8);
    }



    public double getX() {
        return x;
    }
    public double getY() {
        return y;
    }


}
```

```java
package juego;

import java.awt.Image;

import javax.sound.sampled.AudioSystem;
import javax.sound.sampled.Clip;
import javax.sound.sampled.LineUnavailableException;

import entorno.Entorno;
import entorno.Herramientas;

public class Gameover {

	private double x;
	private double y;
	private Image imgGameover;
	double anguloGameover;
	String sonidoGameover;
	Clip clip;

	public Gameover(double x, double y) {
	    this.x = x;
	    this.y = y;
	    imgGameover = Herramientas.cargarImagen("gameover.png");
	    anguloGameover=0;
	    sonidoGameover=".//gameover.wav";
	    clip= Herramientas.cargarSonido(sonidoGameover);
	}
	public void dibujarse(Entorno entorno)
	{
	    // El numero es el tamaño de la imagen
	    entorno.dibujarImagen(imgGameover, this.x, this.y, this.anguloGameover,
	    0.8);
	}


	public double getX() {
	    return x;
	}
	public double getY() {
	    return y;
	}
	public void cargarSonido(String sonidoMisil)  {
	    try {
	        this.clip= AudioSystem.getClip();
	    } catch (LineUnavailableException e) {
	        // TODO Bloque catch generado automáticamente
	        e.printStackTrace();
	    }

	}

	public void play() {
	    clip.start();
	}


}

```

```java
package juego;

import java.awt.Image;

import javax.sound.sampled.AudioSystem;
import javax.sound.sampled.Clip;
import javax.sound.sampled.LineUnavailableException;

import entorno.Entorno;
import entorno.Herramientas;

public class Ganaste {

    private double x;
    private double y;
    private Image imgFondo;
    double anguloGanaste;
    String sonidoGanaste;
    Clip clip;
    public Ganaste(double x, double y) {
        this.x = x;
        this.y = y;
        imgFondo = Herramientas.cargarImagen("ganaste.png");
        anguloGanaste=0;
        sonidoGanaste=".//ganaste.wav";
        clip= Herramientas.cargarSonido(sonidoGanaste);
    }
    public void dibujarse(Entorno entorno)
    {
        // El numero es el tamaño de la imagen
        entorno.dibujarImagen(imgFondo, this.x, this.y, this.anguloGanaste, 0.8);
    }



    public double getX() {
        return x;
    }
    public double getY() {
        return y;
    }
    public void cargarSonido(String sonidoMisil)  {
        try {
            this.clip= AudioSystem.getClip();
        } catch (LineUnavailableException e) {
            // TODO Bloque catch generado automáticamente
            e.printStackTrace();
        }

    }

    public void play() {
        clip.start();
    }
}
```

```java
package juego;

import java.awt.Color;
import java.awt.Image;
import java.util.Random;

import java.io.File;
import javax.sound.sampled.AudioFileFormat;
import javax.sound.sampled.AudioSystem;
import javax.sound.sampled.Clip;


import javax.management.timer.Timer;
import entorno.Board;
import entorno.Entorno;
import entorno.Herramientas;
import entorno.InterfaceJuego;

@SuppressWarnings("unused")
public class Juego extends InterfaceJuego
{
    // El objeto Entorno que controla el tiempo y otros

    private Entorno entorno;
    private Mikasa mikasa;
    Fondo fondo;
    Gameover gameover;
    Ganaste ganaste;
    Kyojin kyojin;
    Kyojin clip3;
    Kyojin[] kyojines;
    Misil misil;
    //Misil clip1;
    Obstaculos [] casas;
    //Obstaculos clip2;
    //Obstaculos clip4;
    Pocion pocion ;
    Vida vida;
    //Pocion clip;
    int tiempo = 3570;
    int segundos = 0;
    int cantSegundos=0;
    int puntaje = 0;
    int cantKyojines=0;
    int cantPociones = 0;
    //int tiempoColision = 0;
    int total = 0;
    int cantVidas = 3;


    // Variables y métodos propios de cada grupo
    // ...

    Juego()
    {
        // Inicializa el objeto entorno

        this.entorno = new Entorno(this, "Attack on Titan, Final Season - Grupo ... -
        v1", 800, 600);

        // Inicializar lo que haga falta para el juego

        // Crea mikasa

        mikasa = new Mikasa(400, 300, 0, false);

        // Crea el fondo

        fondo = new Fondo(400, 300);
        // Crea el gameover

```

```
71          gameover = new Gameover(400, 300);
72
73          ganaste = new Ganaste(400, 300);
74
75          // Crea el array de kyojines sin superponerse con los obstaculos y con Mikasa
76
77          kyojines= new Kyojin[6];
78          for(int i = 0 ; i <kyojines.length; i++ ) {
79              kyojines[i]= new Kyojin(Math.random()*entorno.ancho(),Math.random()*
                entorno.alto(),1,0,50);
80
81              while((kyojines[i].getX() > 150 && kyojines[i].getX() < 650) || (kyojines[
                i].getY() > 200 && kyojines[i].getY() < 400)) {
82
83                  kyojines[i]= new Kyojin(Math.random()*entorno.ancho(),Math.random()*
                    entorno.alto(),1,0,50);
84
85              }
86
87          }
88
89          // Crea la vida extra
90
91          vida = new Vida(800,600);
92
93          //Crea las pociones
94
95          pocion = new Pocion(550,350);
96
97
98
99
100         //Crea el array de casas
101
102         casas= new Obstaculos[6];
103         casas[0]=new Obstaculos(entorno.ancho()/4, 150);
104         casas[1]= new Obstaculos (entorno.ancho()/2, 150);
105         casas[2]= new Obstaculos(entorno.ancho()/1.3,150);
106         casas[3]= new Obstaculos(entorno.ancho()/4, 400);
107         casas[4]=new Obstaculos(entorno.ancho()/2,400);
108         casas[5]= new Obstaculos(entorno.ancho()/1.3,400);
109
110         // Inicia el juego!
111
112         this.entorno.iniciar();
113     }
114
115     // Detecta colisiones
116
117     public boolean colision(double x1, double y1, double x2, double y2, double dist) {
118         return (x1 - x2) * (x1 - x2) + (y1 - y2) * (y1 - y2) < dist * dist;
119     }
120
121
122     /**
123      * Durante el juego, el método tick() será ejecutado en cada instante y
124      * por lo tanto es el método más importante de esta clase. Aquí se debe
125      * actualizar el estado interno del juego para simular el paso del tiempo
126      * (ver el enunciado del TP para mayor detalle).
127      */
128     public void tick()
129     {
130         if ((cantVidas > 0 && cantVidas < 4) && cantKyojines < 10 && (tiempo < 3571 &&
              tiempo >0))
131         {
132         // Carga las imagenes en pantalla
133
134
135         // Inicializa el tiempo
136
137         tiempo --;
```

```
138
139         // Calcula los segundos
140
141         if(tiempo % 60 == 0) {
142             segundos ++;
143         }
144
145         // Dibuja el fondo
146
147         fondo.dibujarse(entorno);
148
149         // Dibuja los textos
150
151         fondo.dibujarse(entorno);
152         entorno.cambiarFont("gang of three",32, Color.black);
153         entorno.escribirTexto("Kyojines eliminados:"+cantKyojines, 5, 580);
154         entorno.escribirTexto("Puntaje:" + puntaje, 50, 50);
155         entorno.escribirTexto("Cantidad de vidas: " + cantVidas, 250, 50);
156         entorno.escribirTexto("Tiempo " + segundos , 620, 50);
157
158         //Dibuja el misil y si salio de pantalla lo elimina
159
160         if(misil != null){
161             misil.dibujarse(entorno);
162             misil.mover();
163             if(misil.chocasteCon(entorno)) {
164                 misil=null;
165             }
166         }
167
168         // Si mikasa colisionó con la pocion cambia el tamaño y elimina la pocion
169
170         if ((pocion != null) && colision(mikasa.getX(), mikasa.getY(), pocion.x ,
            pocion.y ,42)) {
171             pocion.play();
172             pocion = null;
173             mikasa.seTransformo = true;
174             cantSegundos = segundos;
175             cantPociones ++;
176         }
177
178         // Si no colisiono contra ningun kyojin a los 5 segundos le saca el poder de
            la pocion
179
180         if (mikasa.seTransformo == true && (segundos - cantSegundos) == 5) {
181             mikasa.seTransformo = false;
182         }
183
184         // Dibuja las pociones y verifica que no se superponga con un obstaculo
185
186
187         if (pocion !=null && cantPociones < 3) {
188             while(pocion.x > 150 && pocion.y < 650) {
189                 pocion = new Pocion(Math.random()*550,Math.random()*350);
190             }
191             pocion.dibujarse(entorno);
192
193         }
194         else {
195             if(segundos % 10 == 0 && cantPociones < 3) {
196
197                 pocion = new Pocion(Math.random()*550,Math.random()*350);
198                 while(pocion.x > 150 && pocion.y < 650) {
199                     pocion = new Pocion(Math.random()*550,Math.random()*350);
200                 }
201                 pocion.dibujarse(entorno);
202
203             }
204         }
205
206         // Dibuja la vida extra y verifica que no se superponga con un obstaculo
```

```
207
208
209             if (vida !=null && segundos > 15 && cantVidas < 3) {
210                 while(vida.x > 150 && vida.y < 650) {
211                     vida = new Vida(Math.random()*550,Math.random()*350);
212                 }
213                 vida.dibujarse(entorno);
214
215             }
216
217             // Si mikasa colisionó con la vida extra aumenta las vidas
218
219             if ((vida != null) && colision(mikasa.getX(), mikasa.getY(), vida.x , vida.y ,
                42)) {
220                 vida.play();
221                 vida = null;
222                 cantVidas ++;
223             }
224
225             // Dibuja a mikasa
226
227             if (mikasa != null) {
228                 mikasa.dibujar(entorno);
229             }
230
231             // Dibuja los kyojines
232
233             for(int i = 0 ; i <kyojines.length; i++ ) {
234
235                 if(kyojines[i] != null) {
236                     kyojines[i].mover();
237                     kyojines[i].cambiarAngulo(mikasa.getX(),mikasa.getY());
238                     kyojines[i].dibujar(entorno);
239
240                     if(kyojines[i] != null && kyojines[i].chocasteCon(entorno)) {
241                         kyojines[i].cambiarTrayectoria();
242                   }
243
244                 }
245             }
246
247
248             // Si los kyojines colisionan con el misil los elimina y reproduce sonido
249              for (int i= 0 ; i < kyojines.length ; i ++) {
250                 if(misil != null && kyojines[i] !=null) {
251                     if (colision(kyojines[i].getX(), kyojines[i].getY(), misil.getX(),
                        misil.getY(),32)) {
252                         kyojines[i].play();
253                         kyojines[i] = null;
254                         misil = null;
255                         cantKyojines++;
256                         puntaje = puntaje + 100;
257
258                         if (cantKyojines < 5 )
259                             for (int j =0; j< kyojines.length; j++) {
260                                 if(kyojines[j]==null)
261                                     kyojines[j]= new Kyojin(Math.random()*entorno.ancho(),
                                    Math.random()*entorno.alto(),1,0,50);
262                                     kyojines[j].cambiarAngulo(mikasa.getX(), mikasa.getY
                                    ());
263                                     kyojines[j].dibujar(entorno);
264                                     if(kyojines[j] != null && kyojines[j].chocasteCon(
                                    entorno)) {
265                                         kyojines[j].cambiarTrayectoria();
266                                   }
267                             }
268                     }
269                 }
270              }
271
272
```

```
273
274         // Si los kyojines colisionan contra mikasa transformada los elimina
275         // reproduce sonido y asigna puntaje
276         // Si esta sin transformar saca una vida
277
278         for (int i = 0 ; i < kyojines.length ; i++) {
279             if (kyojines[i] !=null && mikasa != null) {
280                 if((colision(kyojines[i].getX(), kyojines[i].getY(), mikasa.getX(),
                    mikasa.getY(),32))) {
281                     if(mikasa.seTransformo == false) {
282                         cantVidas = cantVidas - 1;
283                         kyojines[i] = null;
284                         cantKyojines++;
285
286                     }
287                     if(mikasa.seTransformo == true) {
288                         mikasa.seTransformo = false;
289                         kyojines[i].play();
290                         kyojines[i] = null;
291                         puntaje = puntaje + 50;
292                         cantKyojines++;
293                     }
294                     if (cantKyojines < 5)
295                         for (int j =0; j< kyojines.length; j++) {
296                             if(kyojines[j]==null)
297                                 kyojines[j]= new Kyojin(Math.random()*entorno.ancho(),
                                    Math.random()*entorno.alto(),1,0,50);
298                                 kyojines[j].cambiarAngulo(mikasa.getX(), mikasa.getY
                                    ());
299                                 kyojines[j].dibujar(entorno);
300                                 if(kyojines[j] != null && kyojines[j].chocasteCon(
                                    entorno)) {
301                                     kyojines[j].cambiarTrayectoria();
302                               }
303                         }
304                     }
305
306                 }
307             }
308
309
310             // Hace rebotar a los kyojines contra las casas
311                 for (int i =0 ; i < kyojines.length ; i++) {
312                     for(int j = 0; j < casas.length;j++) {
313                         if(kyojines[i]!=null && kyojines[j] != null) {
314                             if (colision(kyojines[i].getX(), kyojines[i].getY(),casas[
                                j].getX(),casas[j].getY(),62))
315                                 kyojines[i].cambiarTrayectoria();
316                             }
317                         }
318                     }
319
320                 //Colision entre kyojines
321
322                 for (int j =0 ; j< kyojines.length ; j++) {
323                   for(int i =0; i< kyojines.length; i++) {
324                       if(kyojines[j]!=null) {
325                         kyojines[j].colisionDeKyojines(kyojines,i);
326                         if(kyojines[j] != null && kyojines[j].chocasteCon(entorno)) {
327                             kyojines[j].cambiarTrayectoria();
328
329                       }
330
331                       }
332                     }
333                 }
334
335
336
337         // Si el misil le pegó a una casa la prende fuego y reroduce sonido
338
```

```
339             for(int i = 0 ; i < casas.length ; i++) {
340                 if(casas[i] !=null){
341                     casas[i].dibujarse(entorno);
342                     if(misil != null ) {
343                         if (colision(casas[i].getX(), casas[i].getY(), misil.getX(),
                            misil.getY(),72)) {
344                             casas[i].fuego=true;
345                             misil = null;
346                             casas[i].playExplosion();
347                             casas[i].playIncendio();
348                             puntaje = puntaje - 30;
349                         }
350                     }
351                 }
352             }
353 
354         // Mueve a mikasa de acuerdo a la tecla presionada
355 
356 
357         if (entorno.estaPresionada(entorno.TECLA_DERECHA))
358             mikasa.girar(Herramientas.radianes(1));
359             /*if(mikasa.getAngulo()<3.8  && mikasa.getAngulo()>2.3)
360 
361                 mikasa.setAngulo(0);*/
362 
363         if (entorno.estaPresionada(entorno.TECLA_IZQUIERDA))
364             mikasa.girar(Herramientas.radianes(-1));
365             /*if(mikasa.getAngulo()<3.8  && mikasa.getAngulo()>2.3)
366                 mikasa.setAngulo(0);*/
367 
368         if (entorno.estaPresionada(entorno.TECLA_ARRIBA)) {
369             mikasa.giro=false;
370 
371             //No deja que se pase de la pantalla con la tecla arriba presionada
372 
373             if (mikasa.x > (entorno.ancho()-20)) {
374                 if (mikasa.angulo < 1.5*Math.PI && mikasa.angulo > 0.5*Math.PI) {
375                     mikasa.moverAdelante();
376                 }
377             }
378 
379             else if (mikasa.x < 20) {
380                 if ((mikasa.angulo < 0.5*Math.PI && mikasa.angulo > 0*Math.PI)||(
                    mikasa.angulo < 2*Math.PI && mikasa.angulo > 1.5*Math.PI)) {
381                     mikasa.moverAdelante();
382                 }
383             }
384             else if (mikasa.y < 20) {
385                 if (mikasa.angulo < Math.PI && mikasa.angulo > 0*Math.PI) {
386                     mikasa.moverAdelante();
387                 }
388             }
389 
390             else if (mikasa.y > (entorno.alto()-20)) {
391 
392                 if ((mikasa.angulo < 1.5*Math.PI && mikasa.angulo > 1*Math.PI)||(
                    mikasa.angulo >1.5*Math.PI && mikasa.angulo > 0.5*Math.PI ))  {
393                     mikasa.moverAdelante();
394                 }
395             }
396             else
397                 mikasa.moverAdelante();
398         }
399 
400         if (entorno.estaPresionada(entorno.TECLA_ABAJO)) {
401             mikasa.giro=true;
402 
403             //No deja que se pase de la pantalla con la tecla abajo
                presionada
404 
405             if (mikasa.x > (entorno.ancho()-20)) {
```

```
406                 if ((mikasa.angulo < 0.5*Math.PI && mikasa.angulo > 0*Math.PI)||(
                    mikasa.angulo < 2*Math.PI && mikasa.angulo > 1.5*Math.PI)) {
407                     mikasa.moverAtras();
408                 }
409             }
410             else if (mikasa.x < 20) {
411                 if (mikasa.angulo < 1.5*Math.PI && mikasa.angulo > 0.5*Math.PI) {
412                     mikasa.moverAtras();
413                 }
414             }
415 
416             else if (mikasa.y < 20) {
417                 if ((mikasa.angulo < 1.5*Math.PI && mikasa.angulo > 1*Math.PI)||(
                    mikasa.angulo < 2*Math.PI && mikasa.angulo > 1.5* Math.PI)) {
418                     mikasa.moverAtras();
419                 }
420             }
421 
422             else if (mikasa.y > (entorno.alto()-20)) {
423 
424                 if (mikasa.angulo < 1*Math.PI && mikasa.angulo > 0*Math.PI)  {
425                     mikasa.moverAtras();
426                 }
427             }
428 
429             else
430                 mikasa.moverAtras();
431         }
432 
433         // Calcula colision de Mikasa con obstaculos
434 
435             for(int i = 0 ; i < casas.length ; i++) {
436                 if (mikasa.giro==false && colision(casas[i].getX(), casas[i].getY(),
                    mikasa.getX(), mikasa.getY(),  62))
437                     mikasa.moverAtras();
438                 if (mikasa.giro==true && colision(casas[i].getX(), casas[i].getY(),
                    mikasa.getX(), mikasa.getY(),  62 ))
439                     mikasa.moverAdelante();
440             }
441 
442         // Dispara misil y reproduce sonido
443 
444         if(entorno.sePresiono(entorno.TECLA_ESPACIO)) {
445             if (misil == null){
446                 misil= new Misil(mikasa.getX(), mikasa.getY(), mikasa.getAngulo());
447                 misil.giro=mikasa.giro;
448                 misil.disparar(mikasa.getAngulo());
449                 misil.play();           }
450 
451     }
452 }
453         else {
454             if (cantVidas == 0 || tiempo == 0) {
455             gameover.play();
456             gameover.dibujarse(entorno);
457             entorno.cambiarFont("gang of three",62, Color.white);
458             entorno.escribirTexto("GAME OVER" , 210, 150);
459             entorno.cambiarFont("gang of three",30, Color.white);
460             entorno.escribirTexto("Kyojines eliminados: "+cantKyojines, 240, 270);
461             entorno.escribirTexto("Puntaje: " + puntaje, 330, 370);
462             entorno.escribirTexto("Presione DELETE para salir" , 220, 470);
463             entorno.escribirTexto("Presione ENTER para volver a jugar", 150,550);
464 
465             }
466 
467             if(cantKyojines == 10) {
468             ganaste.play();
469             ganaste.dibujarse(entorno);
470             entorno.cambiarFont("gang of three",62, Color.white);
471             entorno.escribirTexto("¡¡Felicitaciones!!" , 50, 150);
472             entorno.cambiarFont("gang of three",30, Color.white);
```

```
473                 entorno.escribirTexto("Exterminaste a "+cantKyojines + " kyojines", 87,
                    270);
474                 entorno.escribirTexto("Hiciste: " + puntaje + " puntos", 87, 370);
475                 entorno.escribirTexto("Presione DELETE salir" , 87, 470);
476                 entorno.escribirTexto("Presione ENTER para volver a jugar", 150,550);
477                 }
478
479             }
480             if(entorno.estaPresionada(entorno.TECLA_DELETE)) {
481                 System.exit(0);
482
483
484             }
485
486             if(entorno.estaPresionada(entorno.TECLA_ENTER)) {
487                 System.exit(0);
488             }
489
490         }
491         @SuppressWarnings("unused")
492         public static void main(String[] args)
493         {
494             Juego juego = new Juego();
495
496
497
498         }
499
500     }
501
502
```

```java
package juego;
import java.util.Random;
import java.util.random.*;

import javax.sound.sampled.AudioSystem;
import javax.sound.sampled.Clip;
import javax.sound.sampled.LineUnavailableException;

import java.awt.Color;
import java.awt.Image;

import entorno.Entorno;
import entorno.Herramientas;
@SuppressWarnings("unused")

public class Kyojin {
    public boolean choco = true;
    private double x;
    private double y;
    private double velocidad;
    private double angulo;
    private double ancho;
    private double alto;
    private int radio;
    //public double mikasax;
    //public double mikasay;

    Clip clip;
    String explosion;
    public int kyojinTiempo = 0;
    Image img2;

    public Kyojin(double x, double y, double velocidad, double angulo, int radio) {
        this.x = x;
        this.y = y;
        this.velocidad = velocidad;
        this.angulo = angulo;
        this.radio = radio;
        ancho=22;
        alto=40;
        explosion=".//explosion.wav";
        clip= Herramientas.cargarSonido(explosion);
        img2 = Herramientas.cargarImagen("kyojin.png");
    }




    public void dibujar(Entorno entorno) {
        //entorno.dibujarRectangulo(x, y, 32, 82, 0, Color.RED);
        entorno.dibujarImagen(img2, x, y, 0, 0.10);
    }

    public void cargarSonido(String explosion)  {
        try {
            this.clip= AudioSystem.getClip();
        } catch (LineUnavailableException e) {
            // TODO Bloque catch generado automáticamente
            e.printStackTrace();
        }

    }

    public void play() {
        clip.start();
    }
    public void cambiarAngulo(double x2, double y2){
        //this.angulo = Math.atan2(y2 - this.y, x2 - this.x);
        this.angulo=Math.PI/2 *Math.atan2(y2-this.y, x2-this.x);

```

```
72
73      }
74
75      public void mover() {
76          //x += velocidad * Math.sin(angulo);
77          //y += velocidad * Math.cos(angulo);
78
79
80          x += Math.cos(angulo)*1.3;
81          y += Math.sin(angulo)*1.3;
82
83      }
84
85
86      //FIXME
87      public boolean chocasteCon(Entorno entorno) {
88          return this.x > 580 || this.x <10 ||  this.y <10 || this.y>380 ;
89
90
91      }
92
93
94
95      public void cambiarTrayectoria() {
96          angulo += Herramientas.radianes(270);
97
98      }
99
100     public void colisionDeKyojines(Kyojin[] kyojines,int indice) {
101          for (int i = 0; i < kyojines.length; i++) {
102           if (kyojines[i]!=null && indice != i && this.x < (kyojines[i].getX() +
              kyojines[i].getAncho() / 2 + ancho)
103             && this.x > (kyojines[i].getX() - kyojines[i].getAncho() / 2 - ancho)
104             && this.y < (kyojines[i].getY() + kyojines[i].getAlto() / 2 + alto)
105             && this.y > (kyojines[i].getY() - kyojines[i].getAlto() / 2 - alto)) {
106            angulo += Herramientas.radianes(270)*Math.PI;
107            choco = false;
108
109           }
110          }
111         }
112
113     public void acelerar() {
114
115         x += Math.cos(angulo)*0.8;
116         y += Math.sin(angulo)*0.8;
117     }
118
119
120     public double getX() {
121         return x;
122     }
123
124
125     public void setX(double x) {
126         this.x = x;
127     }
128
129
130     public double getY() {
131         return y;
132     }
133
134
135     public void setY(double y) {
136         this.y = y;
137     }
138
139
140     public double getVelocidad() {
141         return velocidad;
```

```
142         }
143
144
145         public void setVelocidad(double velocidad) {
146             this.velocidad = velocidad;
147         }
148
149
150         public double getAngulo() {
151             return angulo;
152         }
153
154
155         public void setAngulo(double angulo) {
156             this.angulo = angulo;
157         }
158
159
160         public int getRadio() {
161             return radio;
162         }
163
164
165         public void setRadio(int radio) {
166             this.radio = radio;
167         }
168         public double getAncho() {
169             return ancho;
170         }
171
172
173         public double getAlto() {
174             return alto;
175         }
176
177
178     }
```

```java
package juego;
import java.awt.Color;
import java.awt.Image;
import entorno.Entorno;
import entorno.Herramientas;

@SuppressWarnings("unused")
public class Mikasa {

    // Variables de instancia
    double x;
    double y;
    double angulo;
    Image img1;
    Image img2;
    Image img3;
    Image img4;
    boolean giro;
    boolean transformada;
    // private int radio;
    boolean seTransformo = false;

    public Mikasa(double x, double y,double orientacion, boolean transformada)
    {
        this.x = x;
        this.y = y;
        this.angulo=orientacion;
        this.transformada= transformada;
        giro = false;
        img1 = Herramientas.cargarImagen("mikasa.png");
        img2 = Herramientas.cargarImagen("m2.png");
        img3 = Herramientas.cargarImagen("kyojin2.png");
        img4 = Herramientas.cargarImagen("kyojin2.1.png");
    }

    public void dibujar(Entorno entorno){

        // el numero es el tamaño de la imagen

            if(giro==true) {
                if(seTransformo == false) {

                    entorno.dibujarImagen(img2, x, y, angulo, 0.10);

                }
                else{

                    entorno.dibujarImagen(img3, x, y, angulo, 0.35);

                }
            }
            if(giro==false) {
                if(seTransformo == false) {
                    entorno.dibujarImagen(img1, x, y, angulo, 0.10);
                }
                else {
                    entorno.dibujarImagen(img4, x, y, angulo, 0.35);
                }
            }
    }

    public void girar(double modificador)
    {
        this.angulo = angulo + modificador;
        if(angulo < 0) {
            angulo +=2*Math.PI;
        }
        if(angulo >2* Math.PI) {
            angulo -= 2*Math.PI;
        }
    }
```

```
72
73
74
75      public void moverAdelante() {
76
77          // el numero es la velocidad de movimiento
78
79              this.x += Math.cos(this.angulo)*5;
80              this.y += Math.sin(this.angulo)*5;
81              if(this.x > 900) {
82                  this.x=-100;
83              }
84              if(this.x < -100) {
85                  this.x=900;
86              }
87              if(this.y > 650) {
88                  this.y=-50;
89              }
90              if(this.y < -50) {
91                  this.y=650;
92              }
93      }
94
95      public void moverAtras() {
96
97              // el numero es la velocidad de movimiento
98
99              this.x += -Math.cos(this.angulo)*5;
100             this.y += -Math.sin(this.angulo)*5;
101
102             if(this.x > 900) {
103                 this.x=-100;
104             }
105             if(this.x < -100) {
106                 this.x=900;
107             }
108             if(this.y > 650) {
109                 this.y=-50;
110             }
111             if(this.y < -50) {
112                 this.y=650;
113             }
114         }
115
116
117
118     public double getX() {
119         return x;
120     }
121
122     public void setX(double x) {
123         this.x = x;
124     }
125
126     public double getY() {
127         return y;
128     }
129
130     public void setY(double y) {
131         this.y = y;
132     }
133
134     public double getAngulo() {
135         return angulo;
136     }
137
138     public void setAngulo(double angulo) {
139         this.angulo = angulo;
140     }
141
142
```

```
143
144
145
146    }
147
148
149
```

```java
package juego;

import java.awt.Color;
import java.awt.Image;

import javax.sound.sampled.AudioSystem;
import javax.sound.sampled.Clip;
import javax.sound.sampled.LineUnavailableException;

import entorno.Entorno;
import entorno.Herramientas;
import juego.Mikasa;

@SuppressWarnings("unused")
public class Misil {
    //Variables de instancias
    private Mikasa mikasa;
    double x;
    double y;
    double angulo;
    boolean giro;
    Image imgMisil;
    Image imgMisil2;
    String sonidoMisil;
    Clip clip;
    private int radio;
    public Misil(double x, double y, double angulo) {
        this.x= x;
        this.y= y;
        this.angulo= angulo;
        giro= false;
        sonidoMisil=".//misil.wav";
        imgMisil2= Herramientas.cargarImagen("misil.png");
        imgMisil= Herramientas.cargarImagen("bomba.png");
        clip= Herramientas.cargarSonido(sonidoMisil);
    }

    public void dibujarse(Entorno entorno) {

        //entorno.dibujarRectangulo(x, y, 50, 50, 0, Color.red);
            if(giro)
                entorno.dibujarImagen(imgMisil,x, y,angulo, 0.1);
            else
                entorno.dibujarImagen(imgMisil2, x, y, angulo, 0.1);

    }
    public void cargarSonido(String sonidoMisil)  {
        try {
            this.clip= AudioSystem.getClip();
        } catch (LineUnavailableException e) {
            // TODO Bloque catch generado automáticamente
            e.printStackTrace();
        }

    }

    public void play() {
        clip.start();
    }

    public void disparar(double angulo) {

        this.x += Math.cos(this.angulo)*50 ;
        this.y += Math.sin(this.angulo)*50;

    }

    public void mover() {

        if(giro==false) {

```

```
72              this.x += Math.cos(this.angulo)*5;
73              this.y += Math.sin(this.angulo)*5;
74              }
75
76          if(giro==true) {
77              this.x -= Math.cos(this.angulo)*3;
78              this.y -= Math.sin(this.angulo)*3;
79          }
80
81
82      }
83
84
85          public double getX() {
86          return x;
87      }
88          public boolean chocasteCon(Entorno entorno) {
89              return x <= radio || y <= radio || x >= entorno.ancho() - radio || y >=
                entorno.alto() - radio;
90          }
91      public double setX(double x) {
92          return this.x = x;
93      }
94
95      public double getY() {
96          return y;
97      }
98
99      public double setY(double y) {
100         return this.y = y;
101     }
102
103     public double getAngulo() {
104         return angulo;
105     }
106
107     public double setAngulo(double angulo) {
108         return this.angulo = angulo;
109     }
110
111 }
112
113
114
115
116
```

```java
package juego;

import java.awt.Color;
import java.awt.Image;

import javax.sound.sampled.AudioSystem;
import javax.sound.sampled.Clip;
import javax.sound.sampled.LineUnavailableException;

import entorno.Entorno;
import entorno.Herramientas;

@SuppressWarnings("unused")
public class Obstaculos {
    double x;
    double y;
    double ancho;
    double alto;
    Image imgCasas;
    Image imgCasaFuego;
    boolean fuego;
    Clip clip;
    Clip clip2;
    String explosion;
    String incendio;

    public Obstaculos( double x, double y) {
        this.x = x;
        this.y = y;
        ancho=120;
        alto=120;
        imgCasas= Herramientas.cargarImagen("Casa.png");
        imgCasaFuego= Herramientas.cargarImagen("casa-fuego.png");
        fuego=false;
        explosion=".//explosion.wav";
        clip= Herramientas.cargarSonido(explosion);
        incendio= ".//fuego.wav";
        clip2=Herramientas.cargarSonido(incendio);
    }

    public void dibujarse(Entorno entorno) {

        if (fuego==false)
            entorno.dibujarImagen(imgCasas, x, y, 0, 0.20);
        if(fuego==true)
            entorno.dibujarImagen(imgCasaFuego, x, y, 0, 0.20);

    }

    public void cargarSonido(String explosion)  {
        try {
            this.clip= AudioSystem.getClip();
        } catch (LineUnavailableException e) {
            // TODO Bloque catch generado automáticamente
            e.printStackTrace();
        }

    }

    public void playExplosion() {
        clip.start();
    }

    public void cargarSonido2(String incendio)  {
        try {
            this.clip2= AudioSystem.getClip();
        } catch (LineUnavailableException e) {
            // TODO Bloque catch generado automáticamente
            e.printStackTrace();
        }

```

```
72        }
73
74        public void playIncendio() {
75            clip2.start();
76        }
77
78        public double getX() {
79            return x;
80        }
81
82        public void setX(double x) {
83            this.x = x;
84        }
85
86        public double getY() {
87            return y;
88        }
89
90        public void setY(double y) {
91            this.y = y;
92        }
93
94    }
95
```

```java
package juego;

import java.awt.Image;

import javax.sound.sampled.AudioSystem;
import javax.sound.sampled.Clip;
import javax.sound.sampled.LineUnavailableException;

import entorno.Entorno;
import entorno.Herramientas;

public class Pocion {
    double x;
    double y;
    Image img;
    String sonidoPocima;
    Clip clip;

    public Pocion(double x, double y) {
        this.x= x;
        this.y= y;
        img= Herramientas.cargarImagen("pocion.png");
        sonidoPocima = ".//pocima.wav";
        clip= Herramientas.cargarSonido(sonidoPocima);

    }

    public void dibujarse(Entorno entorno) {
            entorno.dibujarImagen(img, x, y, 0, 0.1);
    }

    public void cargarSonido(String sonidoPocima)  {
        try {
            this.clip= AudioSystem.getClip();
        } catch (LineUnavailableException e) {
            // TODO Bloque catch generado automáticamente
            e.printStackTrace();
        }

    }

    public void play() {
        clip.start();
    }
}

```

```java
package juego;

import java.awt.Image;

import javax.sound.sampled.AudioSystem;
import javax.sound.sampled.Clip;
import javax.sound.sampled.LineUnavailableException;

import entorno.Entorno;
import entorno.Herramientas;

public class Vida {
    double x;
    double y;
    Image img;
    String sonidoVida;
    Clip clip;

    public Vida(double x, double y) {
        this.x= x;
        this.y= y;
        img= Herramientas.cargarImagen("mikasa.png");
        sonidoVida = ".//vida.wav";
        clip= Herramientas.cargarSonido(sonidoVida);

    }

    public void dibujarse(Entorno entorno) {
            entorno.dibujarImagen(img, x, y, 0, 0.05);
    }

    public void cargarSonido(String sonidoVida)  {
        try {
            this.clip= AudioSystem.getClip();
        } catch (LineUnavailableException e) {
            // TODO Bloque catch generado automáticamente
            e.printStackTrace();
        }

    }

    public void play() {
        clip.start();
    }
}
```

Conclusión: El presente trabajo nos representó un gran desafío para todos los integrantes del grupo, estuvimos largas noches escribiendo código y consultando librería de java, tutoriales de YouTube e información de las clases cursadas, más de una vez nos frustramos con algo que no nos salía como queríamos y mas allá de la nota que pueda llegar a tener el presente trabajo, hoy gracias a esto tenemos nuevos conocimientos en cuanto al lenguaje Java como así también a la implementación del Pseudocodigo y el manejo de Objetos y como estos se comportan.