|  |
| --- |
| **UNIVERSIDAD NACIONAL DE GENERAL**  **SARMIENTO**  PROGRAMACION I (2° SEMESTRE DE 2020)  TRABAJO PRÁCTICO: BOSS RABBIT RABBER    **Ubicación:** Los Polvorines, ciudad del partido de Malvinas Argentinas.  **Fecha de entrega:** 1/12/2020  **Comisión:** COM-03 (Turno Noche)  **Docentes:** Waingarten, Leonardo y Niveyro, César  **Estudiantes:** Grupo 7:  Lucas Barrientos, legajo: 38630174/2018, e-mail: [lucasbarrientos7@hotmail.com](mailto:lucasbarrientos7@hotmail.com)  Diego Gomez, Legajo: 41803316/2017, e-mail: [diego.gomezs856@gmail.com](mailto:diego.gomezs856@gmail.com)  Sosa Garcia Matias, Legajo 41385966, e-mail: [mati.g.q.ms@gmail.com](mailto:mati.g.q.ms@gmail.com) |

**ÍNDICE**

# **Introducción al TP…….3**

# **Descripción de clases…….4**

### **Clase Juego…….4**

### **Clase Conejo…….7**

### **Clase Disparo…….8**

### **Clase Calle……...9**

### **Clase Auto……….9**

### **Clase Lobo……….11**

### **Clase Arbusto……….11**

### **Clase Nave……….12**

# **Implementación……..13**

# **Conclusión………..28**

**INTRODUCCIÓN**

**DEL TRABAJO PRÁCTICO**

El juego “*Boss Rabbit Rabber*” es un juego de supervivencia en el cual nuestro protagonista es un conejo extraterrestre que bajó al planeta tierra y necesita volver a su nave espacial a salvo, pero para eso, deberá cruzar los peligrosos obstáculos que se interpongan en su camino.

El juego comienza con el conejo en la parte inferior de la pantalla. Deberá cruzar las distintas calles que irán apareciendo, y esquivar los autos que aparecen por la parte derecha y la parte izquierda de la pantalla. También deberá tener cuidado con los lobos que aparecen entre los arbustos que a su vez, obstaculizan su trayecto. El conejo contará podrá eliminar los autos que se le interpongan utilizando su poder Kamehameha.

Al mismo tiempo, el conejo deberá estar siempre en constante movimiento, debido a que la parte inferior de la pantalla se irá desplazando hacia abajo. En el caso de que la pantalla toque al conejo, el jugador automáticamente perderá.



**DESCRIPCIÓN DE CLASES**

A partir de lo planteado grupalmente, decidimos construir 7 clases además de la clase Juego ya creada, para realizar a cabo el proyecto “*Boss Rabbit Rabber*”. Las cuales son:

* Clase Juego.
* Clase Conejo.
* Clase Disparo.
* Clase Calle.
* Clase Auto.
* Clase Lobo.
* Clase Arbusto.
* Clase Nave.

A continuación se describirán de manera general las clases, sus variables de instancia y sus métodos de clase.

**Clase Juego**

Variables de instancia:

* **int time:** Variable que contabiliza la cantidad de tiempo transcurrido jugando**.**
* **int cantSaltos:** Variable que contabiliza la cantidad de saltos hechos**.**
* **Conejo conejo:** Se define la variable conejo**;**
* **Calle calle:** Se define la variable calle**;**
* **Auto [] autos:** Se define el array autos**;**
* **Disparo disparo:** Se define la variable disparo**;**
* **Arbusto [] arbustos:** Se define el array arbustos**;**
* **Lobo lobo:** Se define la variable lobo**;**
* **Nave nave:** Se define la variable nave**;**
* **Image fondo:** Carga y guarda la imagen del fondo**;**
* **Image espacio:** Carga y guarda la imagen de victoria**;**
* **Image gameOver:** Carga y guarda la imagen de derrota**;**
* **Clip musica:** Almacena el archivo de música que se utiliza en el juego**.**
* **Clip salto:** Almacena el archivo de salto que se utiliza en el juego;
* **Clip derrota:** Almacena el archivo derrota que se utiliza en el juego;
* **Clip ladrido:** Almacena el archivo del ladrido que se utiliza en el juego;
* **Clip detonacion:** Almacena el archivo disparo que se utiliza en la ejecución del disparo;
* **Clip victoria:** Almacena el archivo de victoria al ganar el juego;

Métodos de clase:

* **Juego():** Inicializa el objeto entorno
* **public void tick():** Se encarga de ejecutar todo el bloque de código del proyecto.
* **void inicializarTodo():** inicializa todas las variables y todos los objetos.
* **void recorridoDelJuego():** ejecuta el juego mientras no haya ocasionado un True en colisionNave, colisionAutos, colisionLobo, colisionBordes.
* **boolean colisionNave(Nave nave):** Una condición en donde si el conejo se toca con la nave retorna true, o sino false
* **void reaparecerArbusto(Arbusto [] arbustos, Calle calle):**
* **void imprimirNave():** Imprime en pantalla la imagen de la nave.
* **void imprimirVictoria():** imprime por pantalla la imagen de victoria y la cantidad de saltos.
* **posicionArbustoX():** posición aleatoria que toma el arbusto sobre el eje x.
* **void level2():** El nivel 2 del juego en donde se aumenta la velocidad de los autos, se agrega los arbustos y aparece el lobo.
* **boolean colisionAutos(Auto[] autos):** Una condición en donde si el conejo choca con un auto retorna true, o sino false.
* **boolean colisionLobo(Lobo lobo):** Una condición en donde si el conejo choca con el lobo retorna true, o sino false.
* **void colisionArbusto():** Una condición si el conejo choca con arbusto se llama a chocar.
* **void imprimirAutos(Auto[] autos):** Imprime en pantalla la imagen de los autos.
* **void reaparecerAutos(Auto[] autos):** Reaparecen los autos cada vez que reaparece la calle.
* **void colisionDisparo():** Esta función muestra si el disparo choca con un auto el mismo desaparece.
* **boolean colisionBordes():** Una condición en donde si el conejo toca los bordes de la pantalla retorna true, o sino false.
* **void gameOver():** Se para la música, el disparo desaparece, también imprime por pantalla que el jugador perdió y que tiene que presionar enter para reiniciar el juego.
* **void restartGame():** reinicia el juego y llama a la función inicializarTodo().
* **void imprimirTiempo():** Muestra en la pantalla el tiempo transcurrido.
* **void imprimirSaltos():** Muestra en la pantalla la cantidad de saltos que se van realizando.
* **void sonidoSalto():** Cada vez que está presionada las teclas con la cual se mueve el conejo se ejecuta el sonido de salto.
* **void saltoConejo():** Ejecuta los movimientos del conejo, llamando a salto(), saltarIzquierda(), saltarDerecha() y caer().
* **void disparoDelConejo():** Presionando ESPACIO se ejecuta el disparo, y no se crea otro hasta que desaparezca el mismo. También transforma el disparo en null si es que supera la pantalla.

Anotaciones:

En un principio, al desarrollar el proyecto, nuestro grupo propuso como primer paso dibujar algo en pantalla pero no lo lograbamos realizarlo, gracias a la ayuda de los profesores que lo resolvieron en clase, pudimos realizarlo.

A partir de esto último, pudimos lograr imprimir todos las imágenes, salvo la imagen del disparo, el cual no se dibujaba de manera correcta, pero a través de una clase de consulta con los profesores nos dieron un indicio de cómo realizarlo de manera correcta.

Otra problemática que se resolvió gracias a la clase de consulta, fue el imprimir el arreglo de autos, ya que anteriormente se habían creado cuatro objetos Auto de manera separada y sabíamos que plantearlo de esa forma estaba erróneo.

Por último, decidimos modificar el bloque de código que se encuentra dentro del método tick() de manera que este quede mejor legible.

**Clase Conejo:**

Variables de instancia:

* **int x:** Valor horizontal.
* **int y:** Valor Vertical
* **Image conejo**: carga y guarda la imagen del conejo

Métodos de clase:

* **Constructor:** Construye el objeto Conejo tomando como parámetros los valores de x e y.
* **Setters:** Los valores de x e y están seteados.
* **Getters:** Los valores de x e y están getteados para obtenerlos fuera de la clase.
* **void salto():** Mueve al conejo hacia arriba, disminuyendo la y.
* **void saltarDerecha():** Mueve al conejo hacia la derecha, disminuyendo la x.
* **void saltarIzquierda():** Mueve al conejo hacia la izquierda, incrementando la x.
* **void caer():** Aumenta la posición del conejo en y constantemente para dar el efecto de decrecimiento.
* **void chocar():** Este método es llamado cuando el conejo choca contra un arbusto, aumenta su valor en y para dar el efecto de rebote
* **void imprimir(Entorno e):** Dibuja en pantalla la imagen del conejo.

**Clase Disparo:**

Variables de instancia:

* **int x:** Valor horizontal.
* **int y:** Valor Vertical.
* **Image disparo**: Carga y guarda la imagen del disparo.

Métodos de clase:

* **Constructor:** Construye el objeto Disparo tomando como parámetros los valores de x e y.
* **Setters:** Los valores de x e y están seteados.
* **Getters:** Los valores de x e y están getteados para obtenerlos fuera de la clase.
* **void moverse():** Este método es utilizado para que el disparo se desplace hacia el borde superior de la pantalla.
* **void disparar():** Imprime en pantalla la imagen del disparo y dentro de la misma también llama al método moverse().

**Clase Calle:**

Variables de instancia:

* **int x:** Valor horizontal.
* **int y:** Valor Vertical
* **Image calle**: Carga y guarda la imagen de la calle.

Métodos de clase:

* **Constructor:** Construye el objeto Calle tomando como parámetros los valores de x e y.
* **Setters:** Los valores de x e y están seteados.
* **Getters:** Los valores de x e y están getteados para obtenerlos fuera de la clase.
* **void bajarCalle():** Este método es utilizado para descender la imagen de la calle en pantalla.
* **void imprimir():** Imprime en pantalla la imagen de la calle y dentro de la misma también llama al metodo bajarCalle().
* **void aparecerCalle():** Este método se utiliza para cuando la imagen de la calle descienda completamente de la pantalla vuelva a aparecer en el borde superior de la pantalla.

**Clase Auto:**

Variables de instancia:

* **int x:** Valor horizontal.
* **int y:** Valor Vertical.
* **int carril:** Variable usada para diferenciar los carriles de la calle.
* **String nombreAuto:** Variable usada para trabajar de manera más sencilla los métodos de clase que eligen aleatoriamente la imagen de un auto.
* **Image disparo**: Carga y guarda la imagen del disparo.

Métodos de clase:

* **Constructor:** Construye el objeto Auto tomando como parámetros los valores de x, y, carril y nombreAuto.
* **Setters:** Los valores de x e y están seteados.
* **Getters:** Los valores de x e y están getteados para obtenerlos fuera de la clase.
* **void moverse():** Dependiendo qué carril tiene como parámetro el objeto Auto, desplaza la imagen del auto de izquierda a derecha o viceversa.
* **void moverse2():** Una vez que el usuario logre llegar al nivel 2, dependiendo qué carril tiene como parámetro el objeto Auto los desplaza a mayor velocidad hacia los bordes laterales.
* **void imprimir():** Imprime en pantalla la imagen de un auto y además este método llama al método moverse().
* **String cambiarAuto():** Dependiendo qué carril tiene como parámetro el objeto Auto, utilizando un arreglo de String con los nombres de las imágenes de los autos, obtenemos uno de manera aleatoria.
* **void reaparecerAuto():** Dependiendo qué carril tiene como parámetro el objeto Auto, intercambia la imagen del auto y hace reaparecer el auto por el borde derecho o el izquierdo dependiendo su carril.

Anotaciones:

Las variables nombreAuto y carril fueron creadas para manipularlos los elementos del juego de manera más sencilla. En nombreAuto decidimos guardar el nombre de la imagen de un auto dependiendo qué carril ocupaba, para que de esta manera los autos se imprimen de manera correcta mirando hacia el lado en la que se direccionan.

También nos favoreció el uso de la variable carril para modificar la velocidad con respecto al otro carril de la mano contraria.

**Clase Lobo:**

Variables de instancia:

* **int x:** Valor horizontal.
* **int y:** Valor Vertical
* **Image lobo**: Carga y guarda la imagen del lobo.

Métodos de clase:

* **Constructor:** Construye el objeto Disparo tomando como parámetros los valores de x e y.
* **Setters:** Los valores de x e y están seteados.
* **Getters:** Los valores de x e y están getteados para obtenerlos fuera de la clase.
* **void imprimir(Entorno e):** Imprime en pantalla la imagen de un lobo.
* **void moverse():** Mueve al lobo hacia la izquierda, decreciendo su valor en x.
* **void reaparecerLobo(Calle calle):** Reaparece el lobo por el borde derecho de la pantalla una vez que sale del borde izquierdo de la misma.

**Clase Arbusto**

Variables de instancia:

* **int x:** Valor horizontal.
* **int y:** Valor Vertical
* **Image arbusto**: Carga la imagen del arbusto.
* **Int fila:** Valor en donde aparece el arbusto.

Métodos de clase:

* **Constructor:** Construye el objeto Arbusto tomando como parámetros los valores de x, y, fila.
* **Setters:** Los valores de x, y, fila están seteados.
* **Getters:** Los valores de x, y, fila están getteados para obtenerlos fuera de la clase.
* **Void bajarArbusto():** Este método es utilizado para descender la imagen del arbusto en pantalla.
* **void imprimir():** Imprime en pantalla la imagen del arbusto y dentro de la misma también llama al método bajarArbusto().

Anotaciones:

Una vez realizado el arreglo de autos, implementamos un arreglo de Arbustos y pusimos en práctica lo aprendido de manipular un arreglo de objetos.

La variable fila, funciona a semejanza de lo que era carril para el Objeto Auto, solo que en este caso fue para lograr que la imágenes de los arbustos no se toquen y se dibujen en una distancia correcta en la que el usuario pueda pasar.

**Clase Nave:**

Variables de instancia:

* **int x:** Valor horizontal.
* **int y:** Valor Vertical
* **Image nave**: Carga y guarda la imagen de la nave.

Métodos de clase:

* **Constructor:** Construye el objeto Disparo tomando como parámetros los valores de x e y.
* **Setters:** Los valores de x e y están seteados.
* **Getters:** Los valores de x e y están getteados para obtenerlos fuera de la clase.
* **void bajarNave():** Aumenta la posición en y de la nave constantemente para dar el efecto de decrecimiento.
* **void imprimir(Entorno e):** Imprime en pantalla la imagen de una nave.

**IMPLEMENTACIÓN**

A continuación se presentará textualmente el código fuente tal cual está en el proyecto realizado y con comentarios generales:

**Clase Juego:**

**package** juego;

**import** java.awt.Color;

**import** java.awt.Image;

**import** java.util.Random;

**import** javax.sound.sampled.Clip;

**import** entorno.Entorno;

**import** entorno.Herramientas;

**import** entorno.InterfaceJuego;

**public** **class** Juego **extends** InterfaceJuego {

**private** Entorno entorno;

// Variables:

**private** **int** time;

**private** **int** score;

**private** **int** cantSaltos;

**private** Conejo conejo;

**private** Disparo disparo;

**private** Calle calle;

**private** Auto[] autos;

**private** Arbusto[] arbustos;

**private** Lobo lobo;

**private** Nave nave;

**private** Image fondo;

**private** Image espacio;

**private** Image gameOver;

**private** Clip musica;

**private** Clip detonacion;

**private** Clip ladrido;

**private** Clip salto;

**private** Clip victoria;

**private** Clip derrota;

Juego() {

// Inicializa el objeto entorno

**this**.entorno = **new** Entorno(**this**, "barrientos-gomez-sosa-tp-p1", 800, 600);

// Inicializar lo que haga falta para el juego

**this**.inicializarTodo();

// Inicia el juego!

**this**.entorno.iniciar();

}

**public** **void** tick() {

**if** (colisionBordes() || colisionLobo(lobo)|| colisionAutos(autos)) {

**this**.gameOver();

**this**.restartGame();

} **else** **if** (colisionNave(nave)) {

**this**.imprimirVictoria();

} **else** {

**this**.recorridoDelJuego();

}

}

// Metodos:

/\*\* Inicializa todas las variables \*/

**void** inicializarTodo() {

**this**.time = 0;

**this**.score = 0;

**this**.cantSaltos = 0;

**this**.musica = Herramientas.*cargarSonido*("musica.wav");

**this**.salto = Herramientas.*cargarSonido*("salto.wav");

**this**.ladrido = Herramientas.*cargarSonido*("ladrido.wav");

**this**.detonacion = Herramientas.*cargarSonido*("detonacion.wav");

**this**.victoria = Herramientas.*cargarSonido*("victoria.wav");

**this**.derrota = Herramientas.*cargarSonido*("derrota.wav");

**this**.fondo = Herramientas.*cargarImagen*("fondo.png");

**this**.espacio = Herramientas.*cargarImagen*("espacio.png");

**this**.gameOver = Herramientas.*cargarImagen*("gameOver.png");

**this**.conejo = **new** Conejo(400, 430);

**this**.calle = **new** Calle(400, -150);

**this**.lobo = **new** Lobo(900, -150);

**this**.nave = **new** Nave(400, **this**.calle.getY()-150);

**this**.autos = **new** Auto[4];

**for** (**int** i = 0; i < **this**.autos.length; i++) {

**if** (i % 2 != 0) {

**this**.autos[i] = **new** Auto(900, -75 \* (i + 1), i + 1, "autoIzq" + (i + 1) + ".png");

} **else** {

**this**.autos[i] = **new** Auto(-100, 75 \* (i + 1), i + 1, "autoDer" + (i + 1) + ".png");

}

}

**this**.arbustos = **new** Arbusto[4];

**for** (**int** i = 0; i < **this**.arbustos.length; i++) {

**this**.arbustos[i] = **new** Arbusto(**this**.posicionArbustoX(), **this**.calle.getY() - 600 - (i \* 150), i + 1);

}

}

/\*\* Recorrido comun del juego \*/

**void** recorridoDelJuego() {

**this**.time++;

**this**.derrota.stop();

**this**.musica.loop(1);

**this**.entorno.dibujarImagen(fondo, 450, 300, 0);

**this**.calle.imprimir(entorno);

**this**.calle.reaparecerCalle();

**this**.conejo.imprimir(entorno);

**this**.imprimirAutos(autos);

**this**.reaparecerAutos(autos);

**this**.disparoDelConejo();

**this**.saltoConejo();

**this**.sonidoSalto();

**this**.imprimirPuntaje();

**this**.imprimirSaltos();

**this**.level2();

**this**.imprimirNave();

}

/\*\* Imprime en pantalla la derrota del Usuario \*/

**void** gameOver() {

**this**.disparo = **null**;

**this**.derrota.start();

**this**.musica.close();

**this**.entorno.dibujarImagen(gameOver, 450, 300, 0);

**this**.entorno.cambiarFont("Aharoni", 50, Color.***WHITE***);

**this**.entorno.escribirTexto("GAME OVER", 250, 250);

**this**.entorno.cambiarFont("Aharoni", 30, Color.***RED***);

**this**.entorno.escribirTexto("PRESS ENTER FOR RESTART THE GAME", 140, 300);

}

/\*\* Reinicia el juego \*/

**void** restartGame() {

**if** (**this**.entorno.sePresiono(**this**.entorno.TECLA\_ENTER)) {

**this**.inicializarTodo();

}

}

/\*\* Ejecuta el nivel 2 del juego \*/

**void** level2() {

**if** (time > 1900 && time < 2200) {

**this**.entorno.cambiarFont("Showcard Gothic", 50, Color.***RED***);

**this**.entorno.escribirTexto("LEVEL 2", 300, 500);

}**if** (time>1800) {

**this**.imprimirAutos2(autos);

}

**if** (time > 2300) {

**this**.reaparecerArbusto(arbustos, calle);

}

**if** (time > 2300 && time < 5000) {

**this**.ladrido.loop(1);

**this**.lobo.imprimir(entorno);

**if** (time > 2400 && time < 2650) {

**this**.entorno.cambiarFont("Showcard Gothic", 45, Color.***RED***);

**this**.entorno.escribirTexto("BEWARE OF THE WOLF!", 130, 500);

}

**this**.lobo.reaparecerLobo(calle);

**this**.imprimirSaltos();

**this**.imprimirPuntaje();

}

}

/\*\* Imprime en pantalla la imagen de la Nave \*/

**void** imprimirNave() {

**if** (time >= 4900) {

**this**.ladrido.close();

**this**.nave.imprimir(entorno);

}

}

/\*\* Imprime la pantalla de victoria \*/

**void** imprimirVictoria() {

**this**.musica.close();

**this**.victoria.start();

**this**.entorno.dibujarImagen(espacio, 450, 300, 0);

**this**.entorno.cambiarFont("Showcard Gothic", 70, Color.***WHITE***);

**this**.entorno.escribirTexto("YOU WIN!", 230, 150);

**this**.entorno.cambiarFont("Showcard Gothic", 40, Color.***YELLOW***);

**this**.entorno.escribirTexto("JUMPS: " + cantSaltos + " " + "SCORE: " + score, 190, 200);

**this**.entorno.cambiarFont("Aharoni", 30, Color.***RED***);

**this**.entorno.escribirTexto("PRESS ENTER FOR RESTART THE GAME", 140, 520);

**this**.restartGame();

}

/\*\* Otorga una posicion aleatoria en x al objeto Arbusto \*/

**int** posicionArbustoX() {

Random r = **new** Random();

**int**[] aleatorio = { 140, 300, 500, 660 };

**int** posicion = r.nextInt(4);

**int** numero = aleatorio[posicion];

**return** numero;

}

/\*\* Imprime en pantalla el arreglo de autos \*/

**void** imprimirAutos(Auto[] autos) {

**for** (**int** i = 0; i < **this**.autos.length; i++) {

**if**(**this**.autos[i]!=**null**) {

**this**.autos[i].imprimir(entorno);

} **else** {

**if** (i % 2 != 0) {

**this**.autos[i] = **new** Auto(900, -75 \* (i + 1), i + 1, "autoIzq" + (i + 1) + ".png");

**this**.autos[i].imprimir(entorno);

} **else** {

**this**.autos[i] = **new** Auto(-100, 75 \* (i + 1), i + 1, "autoDer" + (i + 1) + ".png");

**this**.autos[i].imprimir(entorno);

}

}

}

}

/\*\* Imprime los autos en el nuivel 2 y eleva su velocidad \*/

**void** imprimirAutos2(Auto[] autos) {

**for** (**int** i = 0; i < **this**.autos.length; i++) {

**if**(**this**.autos[i]!=**null**) {

**this**.autos[i].imprimir(entorno);

**this**.autos[i].moverse2();

} **else** {

**if** (i % 2 != 0) {

**this**.autos[i] = **new** Auto(900, -75 \* (i + 1), i + 1, "autoIzq" + (i + 1) + ".png");

**this**.autos[i].imprimir(entorno);

**this**.autos[i].moverse2();

} **else** {

**this**.autos[i] = **new** Auto(-100, 75 \* (i + 1), i + 1, "autoDer" + (i + 1) + ".png");

**this**.autos[i].imprimir(entorno);

**this**.autos[i].moverse2();

}

}

}

}

/\*\* Verifica si el conejo colisiono con un Objeto Auto \*/

**boolean** colisionAutos(Auto[] autos) {

**for** (**int** i = 0; i < **this**.autos.length; i++) {

**if** (**this**.autos[i]!=**null**) {

**if** (**this**.conejo.getX() + 15 >= **this**.autos[i].getX() - 70

&& **this**.conejo.getX() - 15 <= **this**.autos[i].getX() + 70

&& **this**.conejo.getY() + 15 >= **this**.autos[i].getY() - 10

&& **this**.conejo.getY() - 15 <= **this**.autos[i].getY() + 22) {

**return** **true**;

}

}

}

**return** **false**;

}

/\*\* Verifica si el conejo ingresó a la nave \*/

**boolean** colisionNave(Nave nave) {

**if** (**this**.conejo.getX() + 30 >= **this**.nave.getX() - 800 && **this**.conejo.getX() - 30 <= **this**.nave.getX() + 800

&& **this**.conejo.getY() + 25 >= **this**.nave.getY() - 50

&& **this**.conejo.getY() - 35 <= **this**.nave.getY() + 50) {

**return** **true**;

}

**return** **false**;

}

/\*\* Verifica si el conejo colisiono con un Objeto Lobo \*/

**boolean** colisionLobo(Lobo lobo) {

**if** (**this**.conejo.getX() + 30 >= **this**.lobo.getX() - 70 && **this**.conejo.getX() - 30 <= **this**.lobo.getX() + 70

&& **this**.conejo.getY() + 25 >= **this**.lobo.getY() - 10

&& **this**.conejo.getY() - 35 <= **this**.lobo.getY() + 22) {

**return** **true**;

}

**return** **false**;

}

/\*\* Verifica si el conejo colisiono con un Objeto Arbusto \*/

**void** colisionArbusto() {

**for** (**int** i = 0; i < **this**.arbustos.length; i++) {

**if** (**this**.conejo.getX() + 30 >= **this**.arbustos[i].getX() - 120

&& **this**.conejo.getX() - 30 <= **this**.arbustos[i].getX() + 120

&& **this**.conejo.getY() + 25 >= **this**.arbustos[i].getY() - 10

&& **this**.conejo.getY() - 35 <= **this**.arbustos[i].getY() + 22) {

**this**.conejo.chocar();

}

}

}

/\*\* Verifica si el disparo colisiono con un Objeto Auto \*/

**void** colisionDisparo() {

**for** (**int** i = 0; i < **this**.autos.length; i++) {

**if**(**this**.disparo!=**null**) {

**if** (**this**.disparo.getX() + 15 >= **this**.autos[i].getX() - 55

&& **this**.disparo.getX() - 15 <= **this**.autos[i].getX() + 55

&& **this**.disparo.getY() + 50 >= **this**.autos[i].getY() - 40

&& **this**.disparo.getY() - 20 <= **this**.autos[i].getY() + 40) {

**this**.score +=5;

**this**.autos[i]=**null**;

**this**.disparo=**null**;

}

}

}

}

/\*\* Verifica si el conejo colisiono con los bordes de la pantalla \*/

**boolean** colisionBordes() {

**if** (**this**.conejo.getX() > 790 || **this**.conejo.getX() < 10 || **this**.conejo.getY() > 590

|| **this**.conejo.getY() < 10) {

**return** **true**;

}

**return** **false**;

}

/\*\* Reaparece los autos cuando salen por los bordes laterales \*/

**void** reaparecerAutos(Auto[] autos) {

**for** (**int** i = 0; i < **this**.autos.length; i++) {

**this**.autos[i].reaparecerAuto(calle);

}

}

/\*\* Reaparece el arreglo de Arbustos entre calle y calle \*/

**void** reaparecerArbusto(Arbusto[] arbustos, Calle calle) {

**for** (**int** i = 0; i < **this**.arbustos.length; i++) {

**this**.arbustos[i].imprimir(entorno);

}

**for** (**int** i = 0; i < **this**.arbustos.length; i++) {

**if** (**this**.arbustos[i].getY() > 810) {

**this**.arbustos[i].setX(posicionArbustoX());

**this**.arbustos[i].setY(**this**.calle.getY() - 180 \* (i + 1));

}

}

}

/\*\* Imprime en pantalla el puntaje del Usuario \*/

**void** imprimirPuntaje() {

**this**.entorno.cambiarFont("Showcard Gothic", 30, Color.***WHITE***);

**this**.entorno.escribirTexto("SCORE:" + **this**.score, 600, 50);

}

/\*\* Imprime en pantalla los saltos que realiza el conejo \*/

**void** imprimirSaltos() {

**this**.entorno.cambiarFont("Showcard Gothic", 30, Color.***yellow***);

**this**.entorno.escribirTexto("JUMPS: " + cantSaltos, 50, 50);

}

/\*\* Ejecuta el sonido que hace el conejo al saltar \*/

**void** sonidoSalto() {

**if** (**this**.entorno.estaPresionada(**this**.entorno.TECLA\_ARRIBA)

|| **this**.entorno.estaPresionada(**this**.entorno.TECLA\_DERECHA)

|| **this**.entorno.estaPresionada(**this**.entorno.TECLA\_IZQUIERDA)) {

**this**.salto.loop(1);

}

}

/\*\* Determina a que direccion salta el conejo utilizando las flechas \*/

**void** saltoConejo() {

**if** (**this**.entorno.sePresiono(**this**.entorno.TECLA\_ARRIBA)) {

**this**.conejo.salto();

cantSaltos++;

} **else** **if** (**this**.entorno.sePresiono(**this**.entorno.TECLA\_DERECHA)) {

**this**.conejo.saltarIzquierda();

} **else** **if** (**this**.entorno.sePresiono(**this**.entorno.TECLA\_IZQUIERDA)) {

**this**.conejo.saltarDerecha();

} **else** {

**this**.conejo.caer();

}

**this**.colisionArbusto();

}

/\*\* Ejecuta todo lo relacionado con el disparo \*/

**void** disparoDelConejo() {

**if** (**this**.entorno.sePresiono(**this**.entorno.TECLA\_ESPACIO)) {

**if** (**this**.disparo == **null**) {

**this**.detonacion.loop(1);

**this**.disparo = **new** Disparo(**this**.conejo.getX(), **this**.conejo.getY());

}

}

**if** (**this**.disparo != **null**) {

**this**.disparo.disparar(entorno);

**this**.colisionDisparo();

}

**if** (**this**.disparo != **null**) {

**if** (**this**.disparo.getY() < 0) {

**this**.disparo = **null**;

}

}

}

@SuppressWarnings("unused")

**public** **static** **void** main(String[] args) {

Juego juego = **new** Juego();

} // Cierre del Main

} // Cierre total del Programa

**Clase Conejo:**

**package** juego;

**import** java.awt.Image;

**import** entorno.Entorno;

**import** entorno.Herramientas;

**public** **class** Conejo {

// Variables de Instancia:

**private** **int** x;

**private** **int** y;

**private** Image conejo;

/\*\* Constructor del Objeto Conejo: \*/

Conejo(**int** x, **int** y) {

**this**.x = x;

**this**.y = y;

**this**.conejo = Herramientas.*cargarImagen*("conejoStop.png");

}

/\*\* Permite obtener el valor de x \*/

**public** **double** getX() {

**return** x;

}

/\*\* Permite obtener el valor de y \*/

**public** **double** getY() {

**return** y;

}

/\*\* Mueve al conejo hacia arriba, disminuyendo la y \*/

**void** salto() {

**this**.conejo = Herramientas.*cargarImagen*("conejoJump.png");

**this**.y -= 40;

}

/\*\* Mueve al conejo hacia la derecha, disminuyendo la x \*/

**void** saltarDerecha() {

**this**.conejo = Herramientas.*cargarImagen*("conejoJump.png");

**this**.x -= 40;

}

/\*\* Mueve al conejo hacia la izquierda, incrementando la x \*/

**void** saltarIzquierda() {

**this**.conejo = Herramientas.*cargarImagen*("conejoJump.png");

**this**.x += 40;

}

/\*\* Decrece la posición del conejo constantemente \*/

**void** caer() {

**this**.conejo = Herramientas.*cargarImagen*("conejoStop.png");

**this**.y += 1;

}

/\*\* Disminuye la posicion del conejo cuando choca con un arbusto \*/

**void** chocar() {

**this**.conejo = Herramientas.*cargarImagen*("conejoJump.png");

**this**.y += 10;

}

/\*\* Dibuja en Pantalla la imagen del conejo \*/

**void** imprimir(Entorno entorno) {

entorno.dibujarImagen(conejo, x, y, 0, 0.35);

}

} // Cierre total de la Clase Conejo

**Clase Disparo:**

**package** juego;

**import** java.awt.Image;

**import** entorno.Entorno;

**import** entorno.Herramientas;

**public** **class** Disparo {

// Variables de Instancia:

**private** **double** x;

**private** **double** y;

**private** Image disparo;

/\*\* Constructor del Objeto Alimento: \*/

Disparo(**double** x, **double** y) {

**this**.x = x;

**this**.y = y;

**this**.disparo = Herramientas.*cargarImagen*("kamehameha.png");

}

/\*\* Permite obtener el valor de x \*/

**public** **double** getX() {

**return** x;

}

/\*\* Permite modificar el valor de x \*/

**public** **void** setX(**double** x) {

**this**.x = x;

}

/\*\* Permite obtener el valor de y \*/

**public** **double** getY() {

**return** y;

}

/\*\* Permite modificar el valor de y \*/

**public** **void** setY(**double** y) {

**this**.y = y;

}

/\*\* Mueve el disparo hacia arriba, disminuyendo la y \*/

**private** **void** moverse() {

**this**.y -= 10;

}

/\*\* Imprime en pantalla la imagen de un disparo \*/

**void** disparar(Entorno entorno) {

entorno.dibujarImagen(disparo, x, y-60, 0, 0.06);

moverse();

}

} // Cierre total de la Clase Disparo

**Clase Calle:**

**package** juego;

**import** java.awt.Image;

**import** entorno.Entorno;

**public** **class** Calle {

// Variables de Instancia:

**private** **int** x;

**private** **int** y;

**private** Image calle;

/\*\* Constructor del Objeto Calle: \*/

**public** Calle(**int** x, **int** y) {

**this**.x = x;

**this**.y = y;

**this**.calle = entorno.Herramientas.*cargarImagen*("calle.png");

}

/\*\* Permite obtener el valor de x \*/

**public** **int** getX() {

**return** x;

}

/\*\* Permite modificar el valor de x \*/

**public** **void** setX(**int** x) {

**this**.x = x;

}

/\*\* Permite obtener el valor de y \*/

**public** **int** getY() {

**return** y;

}

/\*\* Permite modificar el valor de y \*/

**public** **void** setY(**int** y) {

**this**.y = y-1;

}

/\*\* Decrece la posición de la calle constantemente \*/

**private** **void** bajarCalle() {

**this**.y = y + 1;

}

/\*\* Dibuja en Pantalla la imagen de la calle \*/

**void** imprimir(Entorno entorno) {

entorno.dibujarImagen(calle, x + 5, y, 0, 1.0);

**this**.bajarCalle();

}

/\*\* Si la calle llega al final de la pantalla, ubica la calle en el principio \*/

**void** reaparecerCalle() {

**if** (**this**.y > 750) {

**this**.y=-150;

}

}

} // Cierre total de la Clase Calle

**Clase Auto:**

**package** juego;

**import** java.awt.Image;

**import** java.util.Random;

**import** entorno.Entorno;

**import** entorno.Herramientas;

**public** **class** Auto {

// Variables de Instancia:

**private** **int** x;

**private** **int** y;

**private** **int** carril;

**private** String nombreAuto;

**private** Image auto;

/\*\* Constructor del Objeto Auto: \*/

**public** Auto(**int** x, **int** y, **int** carril, String nombreAuto) {

**this**.x = x;

**this**.y = y;

**this**.carril = carril;

**this**.auto = Herramientas.*cargarImagen*(**this**.cambiarAuto());

}

/\*\* Permite obtener el valor de x \*/

**public** **int** getX() {

**return** x;

}

/\*\* Permite modificar el valor de x \*/

**public** **void** setX(**int** x) {

**this**.x = x;

}

/\*\* Permite obtener el valor de y \*/

**public** **int** getY() {

**return** y;

}

/\*\* Permite modificar el valor de y \*/

**public** **void** setY(**int** y) {

**this**.y = y;

}

/\*\* Desplaza la imagen del auto dependiendo su carril \*/

**void** moverse() {

**if**(**this**.carril==1 || **this**.carril==3) {

**this**.x -=3;

**this**.y +=1;

} **else** {

**this**.x += 2;

**this**.y += 1;

}

}

/\*\* En el nivel 2: desplaza la imagen del auto dependiendo su carril \*/

**void** moverse2() {

**if**(**this**.carril==1 || **this**.carril==3) {

**this**.y -= 1;

} **else** {

**this**.x -= 1;

**this**.y -= 1;

}

}

/\*\* Imprime en pantalla la imagen de un auto \*/

**void** imprimir(Entorno entorno) {

entorno.dibujarImagen(auto, **this**.x, **this**.y, 0, 0.8);

**this**.moverse();

}

/\*\* Elije el nombre de la imagen de un auto aleatoriamente \*/

String cambiarAuto() {

**if**(**this**.carril==1 || **this**.carril==3) {

Random random = **new** Random();

String[] traficoDerecha = { "autoIzq1.png", "autoIzq2.png", "autoIzq3.png", "autoIzq4.png" };

**int** autoAleatorio = random.nextInt(4);

**this**.nombreAuto = traficoDerecha[autoAleatorio];

**return** nombreAuto;

} **else** {

Random random = **new** Random();

String[] traficoDerecha = { "autoDer1.png", "autoDer2.png", "autoDer3.png", "autoDer4.png" };

**int** autoAleatorio = random.nextInt(4);

**this**.nombreAuto = traficoDerecha[autoAleatorio];

**return** nombreAuto;

}

}

/\*\* Dependiendo su carril, hace reaparecer al auto en el mismo carril \*/

**void** reaparecerAuto(Calle calle) {

**if**(**this**.carril==1) {

**if** (**this**.x < 0 || **this**.auto==**null**) {

**this**.auto = Herramientas.*cargarImagen*(**this**.cambiarAuto());

**this**.x = 1000;

**this**.y =calle.getY()-105;

}

} **else** **if** (**this**.carril==2){

**if** (**this**.x > 900 || **this**.auto==**null**) {

**this**.auto = Herramientas.*cargarImagen*(**this**.cambiarAuto());

**this**.x = 0;

**this**.y =calle.getY()-45;

}

} **else** **if** (**this**.carril==3){

**if** (**this**.x < 0 || **this**.auto==**null**) {

**this**.auto = Herramientas.*cargarImagen*(**this**.cambiarAuto());

**this**.x = 900;

**this**.y =calle.getY()+25;

}

} **else** **if** (**this**.carril==4){

**if** (**this**.x > 900 || **this**.auto==**null**) {

**this**.auto = Herramientas.*cargarImagen*(**this**.cambiarAuto());

**this**.x = -100;

**this**.y =calle.getY()+80;

}

}

}

} // Cierre total de la Clase Auto

**Clase Lobo:**

**package** juego;

**import** java.awt.Image;

**import** entorno.Entorno;

**import** entorno.Herramientas;

**public** **class** Lobo {

// Variables de Instancia:

**private** **int** x;

**private** **int** y;

**private** Image lobo;

/\*\* Constructor del Objeto Lobo: \*/

**public** Lobo(**int** x, **int** y) {

**this**.x = x;

**this**.y = y;

**this**.lobo = Herramientas.*cargarImagen*("lobo.gif");

}

/\*\* Permite obtener el valor de x \*/

**public** **int** getX() {

**return** x;

}

/\*\* Permite modificar el valor de x \*/

**public** **void** setX(**int** x) {

**this**.x = x;

}

/\*\* Permite obtener el valor de y \*/

**public** **int** getY() {

**return** y;

}

/\*\* Permite modificar el valor de y \*/

**public** **void** setY(**int** y) {

**this**.y = y;

}

/\*\* Mueve al lobo hacia la derecha, disminuyendo la x e incrementando la y \*/

**private** **void** moverse() {

**this**.x -= 0.3;

**this**.y += 1;

}

/\*\* Dibuja en Pantalla la imagen del lobo \*/

**void** imprimir(Entorno entorno) {

entorno.dibujarImagen(lobo, x, y, 0, 0.7);

**this**.moverse();

}

/\*\* Si el lobo llega al final de la pantalla, lo ubica en el principio \*/

**void** reaparecerLobo(Calle calle) {

**if** (**this**.x < 0 || **this**.y > 600) {

**this**.x = 900;

**this**.y = calle.getY() - 470;

}

}

} // Cierre total de la Clase Lobo

**Clase Arbusto:**

**package** juego;

**import** java.awt.Image;

**import** entorno.Entorno;

**import** entorno.Herramientas;

**public** **class** Arbusto {

// Variables de Instancia:

**private** **int** x;

**private** **int** y;

**private** Image arbusto;

**private** **int** fila;

/\*\* Constructor del Objeto Arbusto \*/

**public** Arbusto (**int** x, **int** y, **int** fila) {

**this**.x = x;

**this**.y = y;

**this**.fila=0;

**this**.arbusto=Herramientas.*cargarImagen*("arbustos.png");

}

/\*\* Permite obtener el valor de Fila \*/

**public** **int** getFila() {

**return** fila;

}

/\*\* Permite modificar el valor de Fila \*/

**public** **void** setFila(**int** fila) {

**this**.fila = fila;

}

/\*\* Permite obtener el valor de x \*/

**public** **int** getX() {

**return** x;

}

/\*\* Permite modificar el valor de x \*/

**public** **void** setX(**int** x) {

**this**.x = x;

}

/\*\* Permite obtener el valor de y \*/

**public** **int** getY() {

**return** y;

}

/\*\* Permite modificar el valor de y \*/

**public** **void** setY(**int** y) {

**this**.y = y;

}

/\*\* Decrece la imagen del arbusto \*/

**void** bajarArbusto() {

**this**.y = y + 1;

}

/\*\* Invoca al entorno y dibuja la imagen arbusto \*/

**void** imprimir(Entorno entorno) {

entorno.dibujarImagen(arbusto, x, y, 0, 0.35);

**this**.bajarArbusto();

}

} // Cierre total de la Clase Arbusto

**Clase Nave:**

**package** juego;

**import** java.awt.Image;

**import** entorno.Entorno;

**import** entorno.Herramientas;

**public** **class** Nave {

// Variables de Instancia:

**private** **int** x;

**private** **int** y;

**private** Image nave;

/\*\* Constructor del Objeto Nave: \*/

**public** Nave(**int** x, **int** y) {

**this**.x = x;

**this**.y = y;

**this**.nave = Herramientas.*cargarImagen*("nave.png");

}

/\*\* Permite obtener el valor de x \*/

**public** **int** getX() {

**return** x;

}

/\*\* Permite modificar el valor de x \*/

**public** **void** setX(**int** x) {

**this**.x = x;

}

/\*\* Permite obtener el valor de y \*/

**public** **int** getY() {

**return** y;

}

/\*\* Permite modificar el valor de y \*/

**public** **void** setY(**int** y) {

**this**.y = y;

}

/\*\* Decrece la posición de la nave constantemente \*/

**private** **void** bajarNave() {

**this**.y = y + 1;

}

/\*\* Imprime en pantalla la imagen de una nave \*/

**void** imprimir(Entorno entorno) {

entorno.dibujarImagen(nave, **this**.x, **this**.y, 0, 1);

**this**.bajarNave();

}

} // Cierre total de la Clase Nave

**CONCLUSIÓN**

En la implementación del programa se pusieron en práctica gran mayoría de los temas vistos en clase, como creación de objetos, arreglos de los mismos e implementación de métodos de clase, también se usaron muchas funciones realizadas en las prácticas y ejercicios resueltos en clase.

Al momento de pensar la organización de las clases y funciones, fue de mucha ayuda lo visto en clase y el contenido teórico visto en en el entregable de invariable de representación ya que se pudo implementar y poner en práctica.

En conclusión, creemos que este trabajo nos ayudó a reforzar todo lo visto durante la cursada y darle un cierre a temas que eran quizás un poco más difíciles de comprender gracias a la prueba y error.

Para finalizar, al momento de ejecutar el juego y comprobar que cuenta con un buen funcionamiento, se puede decir que los temas vistos en esta cursada se aplicaron correctamente.