Examen Final

**Nombre y Apellido**: Iván Marano

**Materia**: Programación II

**Temática**: Trabajo integrador

**Universidad**: UCES

**Docentes**: Santiago Fazzini, Roberto Guglielmino

**Fecha**: 13/12/2021

Presentación de proyecto

El trabajo realizado fue un videojuego llamado ESTRELLADOS. El mismo consiste en pilotear una nave que será afectada por la gravedad, la cual se puede mover hacia la izquierda, derecha o arriba, con el objetivo de atrapar una estrella dibujada en el mapa.

Este fue realizado con las siguientes bibliotecas:

#include <iostream>

#include <allegro.h> (principalmente)

#include <cmath>

#include <list>

El nombre surgió de un juego de palabras, en el cual se debe atrapar la estrella. De lo contrario, la nave se estrellará. De ahí, el nombre ESTRELLADOS.

Tanto la nave como la estrella cambiarán de posición a medida que pasan los niveles, y el jugador deberá superar una serie de obstáculos sin sobrepasar los limites de la pantalla en búsqueda de alcanzar la estrella.

El juego contiene 5 niveles los cuales irán aumentando su dificultad. Primero no habrá obstáculos, luego aparecerán algunos en forma de rectángulos, después en forma de triángulos y finalmente se combinarán para hacer más difícil cada nivel.

A su vez, existe un medidor de combustible el cual, si se agota, la nave no podrá pilotearse y chocará inmediatamente, aunque se puede recargar si se atraviesa un rectángulo rojo. Este va disminuyendo a medida que se presionan las teclas que mueven la nave (si no se presionan, este no bajará).

Por cada choque, el jugador tendrá la opción de reiniciar el nivel presionando la tecla R, y en caso de querer cerrar el juego, basta con apretar la tecla ESC.

Si el jugador logra pasar todos los niveles, finalmente se pedirán algunos datos del mismo para introducirlo en una lista de ganadores que se mostrará al finalizar el juego.

Código fuente del programa

#include <iostream>

#include <allegro.h>

#include <cmath>

#include <list>

using namespace std;

class Ganador

{

string nombre, pais; ///Datos del ganador

unsigned int edad;

public:

Ganador()

{

}

void setNombre(string nom)

{

this->nombre = nom;

}

string getNombre()

{

return this->nombre;

}

void setPais(string pa)

{

this->pais = pa;

}

string getPais()

{

return this->pais;

}

void setEdad(unsigned int ed)

{

this->edad = ed;

}

unsigned int getEdad()

{

return this->edad;

}

~Ganador()

{

}

};

class Juego

{

double cx = 100, cy = 100; ///Coordenadas x, y que tendrá la nave al iniciar el nivel 1

double vx = 0, vy = -1; ///Velocidades x, y que tendrá la nave al iniciar el nivel 1

int nivel = 1;

double combustible = 100; ///Combustible lleno para iniciar el nivel 1

bool gastar\_combustible = false; ///Dato booleano que cambiará más adelante para gastar el combustible

BITMAP \*buffer = create\_bitmap(740,500); ///Bitmap principal del proyecto

public:

int getNivel()

{

return this->nivel; ///Devuelve el nivel para el Main

}

void menu()

{

BITMAP \*fondo = load\_bitmap("Menu sin seleccionar.bmp", NULL); ///Bitmap que mostrará el menú

BITMAP \*comenzar = load\_bitmap("Menu con comenzar.bmp", NULL); ///Bitmap que mostrará el menú con la opción COMENZAR en amarillo

BITMAP \*salir = load\_bitmap("Menu con salir.bmp", NULL); ///Bitmap que mostrará la opcion SALIR en amarillo

BITMAP \*cursor = load\_bitmap("Cursor.bmp", NULL); ///Bitmap que mostrará el cursor

bool click = false; ///Dato booleano que luego se convertirá en true para avanzar

while(!click)

{

if(mouse\_x > 224 && mouse\_y > 313 && mouse\_x < 503 && mouse\_y < 370)

{

blit(comenzar, buffer, 0, 0, 0, 0, 740, 500);

if(mouse\_b & 1) ///Si se clickea el click izquierdo...

{

clear\_to\_color(buffer, 0x000000);

instrucciones(); ///Muestra las instrucciones al jugador

click = true; ///Devuelve true para salir del while

}

}

else if(mouse\_x > 228 && mouse\_y > 407 && mouse\_x < 374 && mouse\_y < 463)

{

blit(salir, buffer, 0, 0, 0, 0, 740, 500);

if(mouse\_b & 1)

{

click = true; ///Termina el while y se cierra el juego

}

}

else

{

blit(fondo, buffer, 0, 0, 0, 0, 740, 500);

}

masked\_blit(cursor, buffer, 0, 0, mouse\_x, mouse\_y, 13, 24); ///Imprime el cursor con transparencia en el fondo

blit(buffer, screen, 0, 0, 0, 0, 740, 500);

}

}

void instrucciones()

{

while(!key[KEY\_ENTER]) ///Muestra las instrucciones hasta que se presione ENTER.

{

textout\_centre\_ex(buffer, font, "BIENVENIDO/A A ESTRELLADOS", 370, 40, 0xFBFF00, 0x000000);

textout\_centre\_ex(buffer, font, "El objetivo del juego sera", 370, 80, 0xFFFFFF, 0x000000);

textout\_centre\_ex(buffer, font, "pilotear una nave a traves", 370, 100, 0xFFFFFF, 0x000000);

textout\_centre\_ex(buffer, font, "del mapa con el objetivo de", 370, 120, 0xFFFFFF, 0x000000);

textout\_centre\_ex(buffer, font, "atrapar la estrella perdida,", 370, 140, 0xFFFFFF, 0x000000);

textout\_centre\_ex(buffer, font, "evitando todos los obstaculos", 370, 160, 0xFFFFFF, 0x000000);

textout\_centre\_ex(buffer, font, "y sin superar el limite de la", 370, 180, 0xFFFFFF, 0x000000);

textout\_centre\_ex(buffer, font, "pantalla, de lo contrario,", 370, 200, 0xFFFFFF, 0x000000);

textout\_centre\_ex(buffer, font, "seras uno de los tantos", 370, 220, 0xFFFFFF, 0x000000);

textout\_centre\_ex(buffer, font, "ESTRELLADOS", 370, 240, 0xFBFF00, 0x000000);

textout\_centre\_ex(buffer, font, "Cuidado con el combustible, ya", 370, 280, 0xFFFFFF, 0x000000);

textout\_centre\_ex(buffer, font, "que si se agota, la nave no podra", 370, 300, 0xFFFFFF, 0x000000);

textout\_centre\_ex(buffer, font, "moverse y chocara enseguida.", 370, 320, 0xFFFFFF, 0x000000);

textout\_centre\_ex(buffer, font, "El mismo se podra recargar si", 370, 340, 0xFFFFFF, 0x000000);

textout\_centre\_ex(buffer, font, "se alcanza el rectangulo rojo =", 370, 360, 0xFFFFFF, 0x000000);

rectfill(buffer, 502, 358, 512, 368, 0xFF0000);

textout\_centre\_ex(buffer, font, "PRESIONA <ENTER> PARA JUGAR", 370, 400, 0xFBFF00, 0x000000);

blit(buffer, screen, 0, 0, 0, 0, 740, 500);

rest(20);

}

desarrollo();

}

void desarrollo()

{

while(!key[KEY\_ESC]) ///El juego se ejecuta mientras no se presione la tecla ESC

{

bool gastar\_combustible = false;

clear\_to\_color(buffer, 0x000000);

pintar\_nivel(nivel, buffer);

mover\_nave(cx, cy, vx, vy);

if((key[KEY\_UP] || key[KEY\_W]) && combustible > 0)

{

aceleracion(0, vx, vy);

pintar\_motores(cx, cy, buffer);

gastar\_combustible = true;

}

if((key[KEY\_RIGHT] || key[KEY\_D]) && combustible > 0)

{

aceleracion(-90, vx, vy);

pintar\_motores(cx, cy, buffer);

gastar\_combustible = true;

}

if((key[KEY\_LEFT] || key[KEY\_A]) && combustible > 0)

{

aceleracion(90, vx, vy);

pintar\_motores(cx, cy, buffer);

gastar\_combustible = true;

}

medidor\_combustible(gastar\_combustible, combustible, buffer, nivel, cx, cy);

pintar\_nave(cx, cy, buffer);

blit(buffer, screen, 0, 0, 0, 0, 740, 500);

GameOver(cx, cy, vx, vy, nivel, combustible, buffer);

pasar\_nivel(cx, cy, vx, vy, nivel, combustible, buffer);

rest(20);

}

}

void pintar\_estrella(double cx, double cy, BITMAP \*buffer) ///Pinta la estrella a través de un vector de puntos, donde luego se trazarán lineas

{

double estrella[] = {cx-15, cy , cx-5, cy , cx , cy-10, cx+5, cy , cx+15, cy, cx+5, cy+5,

cx+10, cy+15, cx , cy+10, cx-10, cy+15, cx-5, cy+5, cx-15, cy};

for(int i = 0; i <= 18; i+=2)

{

line(buffer, estrella[i], estrella[i+1], estrella[i+2], estrella[i+3], 0xeaeae6);

}

}

void pintar\_nave(double cx, double cy, BITMAP \*buffer) ///Pinta la nave a través de un vector de puntos, donde luego se trazarán lineas

{

double nav[] = {cx-20, cy+20, cx-20, cy+10, cx-10, cy, cx-10, cy-10, cx,

cy-15, cx+10, cy-10, cx+10, cy, cx+20, cy+10, cx+20, cy+20};

for(int i = 0; i <= 14; i+=2)

{

line(buffer, nav[i], nav[i+1], nav[i+2], nav[i+3], 0xeaeae6);

if(i == 14)

{

line(buffer, cx-10, cy, cx+10, cy, 0xeaeae6);

line(buffer, cx-10, cy, cx-10, cy+20, 0xeaeae6);

line(buffer, cx+10, cy, cx+10, cy+20, 0xeaeae6);

}

}

}

void mover\_nave(double &cx, double &cy, double &vx, double &vy) ///Se pasan las direcciones de las coordenadas para mover la nave

{

double ax = 0.0, ay = 0.05;

vx += ax;

vy += ay;

cx += vx;

cy += vy;

}

void rotar(double &x, double &y, double cx, double cy, double da) ///Hace rotar las partes de la nave en la explosión para darle un mejor efecto

{

double dx = x - cx;

double dy = y - cy;

double r = sqrt(dx\*dx + dy\*dy);

double ang = atan2(dy,dx);

double da\_rad = da/180 \* M\_PI;

ang -= da\_rad;

x = cx + r\*cos(ang);

y = cy + r\*sin(ang);

}

void aceleracion(double da, double &vx, double &vy)

{

double ax = 0;

double ay = -0.15; ///Tira la nave hacia abajo como si fuera un efecto de gravedad

rotar(ax, ay, 0, 0, da);

vx += ax;

vy += ay;

}

void pintar\_motores(double cx, double cy, BITMAP \*buffer) ///Pinta un motor central y dos motores laterales en la nave

{ ///Estos se pintan dependiendo que tecla se presione

double x\_izq, x\_der, x\_cen, y;

x\_izq = cx-15;

x\_der = cx+15;

x\_cen = cx;

y = cy;

y += 12;

double fuegoizq[] = {x\_izq-5, y+5, x\_izq-10, y+20, x\_izq-5, y+20, x\_izq, y+35, x\_izq+5, y+20, x\_izq+10, y+20, x\_izq+5, y+5};

double fuegoder[] = {x\_der-5, y+5, x\_der-10, y+20, x\_der-5, y+20, x\_der, y+35, x\_der+5, y+20, x\_der+10, y+20, x\_der+5, y+5};

double fuegocen[] = {x\_cen-5, y+5, x\_cen-10, y+20, x\_cen-5, y+20, x\_cen, y+35, x\_cen+5, y+20, x\_cen+10, y+20, x\_cen+5, y+5};

if(key[KEY\_RIGHT] || key[KEY\_D])

{

for(int i = 0; i <= 10; i+=2)

{

line(buffer, fuegoizq[i], fuegoizq[i+1], fuegoizq[i+2], fuegoizq[i+3], 0xfa9b00);

}

}

else if(key[KEY\_LEFT] || key[KEY\_A])

{

for(int i = 0; i <= 10; i+=2)

{

line(buffer, fuegoder[i], fuegoder[i+1], fuegoder[i+2], fuegoder[i+3], 0xfa9b00);

}

}

else

{

for(int i = 0; i <= 10; i+=2)

{

line(buffer, fuegocen[i], fuegocen[i+1], fuegocen[i+2], fuegocen[i+3], 0xfa9b00);

}

}

}

void medidor\_combustible(bool gastar\_combustible, double &combustible, BITMAP\* buffer, int nivel, double cx, double cy)

{

textout\_centre\_ex(buffer, font, "Combustible", 100, 30, 0x999999, 0x000000);

rectfill(buffer, 50, 50, 50+combustible, 55, 0xFF0000); ///Dibuja la barra de combustible que se irá agotando

recargar\_combustible(cx, cy, combustible, buffer, nivel);

if(gastar\_combustible == true)

{

combustible -= 0.25;

}

if(combustible < 0)

{

if(nivel == 4)

{

rectfill(buffer, 45, 45, 60, 55, 0x999999);

}

else

{

rectfill(buffer, 45, 45, 60, 55, 0x000000);

}

}

}

void pintar\_nivel(int nivel, BITMAP\* buffer)

{

if(nivel == 1)

{

textout\_centre\_ex(buffer, font, "Nivel 1", 700, 30, 0x999999, 0x000000);

pintar\_estrella(700, 450, buffer);

rectfill(buffer, 360, 240, 380, 260, 0xFF0000);

}

else if(nivel == 2)

{

textout\_centre\_ex(buffer, font, "Nivel 2", 700, 30, 0x999999, 0x000000);

rectfill(buffer, 100, 110, 250, 500, 0x999999);

rectfill(buffer, 450, 0, 640, 390, 0x999999);

pintar\_estrella(700, 100, buffer);

rectfill(buffer, 360, 440, 380, 460, 0xFF0000);

}

else if(nivel == 3)

{

textout\_centre\_ex(buffer, font, "Nivel 3", 700, 30, 0x999999, 0x000000);

triangle(buffer, 0, 80, 80, 0, 0, 0, 0x999999);

triangle(buffer, 100, 100, 470, 500, 100, 500, 0x999999);

triangle(buffer, 270, 0, 640, 400, 640, 0, 0x999999);

triangle(buffer, 660, 500, 740, 420, 740, 500, 0x999999);

pintar\_estrella(30, 450, buffer);

rectfill(buffer, 170, 50, 190, 70, 0xFF0000);

}

else if(nivel == 4)

{

textout\_centre\_ex(buffer, font, "Nivel 4", 700, 30, 0x999999, 0x000000);

rectfill(buffer, 0, 0, 142, 72, 0x999999);

rectfill(buffer, 110, 153, 204, 260, 0x999999);

rectfill(buffer, 204, 77, 305, 260, 0x999999);

rectfill(buffer, 305, 0, 492, 77, 0x999999);

rectfill(buffer, 492, 177, 583, 500, 0x999999);

rectfill(buffer, 583, 177, 661, 309, 0x999999);

triangle(buffer, 0, 270, 240, 500, 0, 500, 0x999999);

triangle(buffer, 150, 260, 305, 425, 305, 260, 0x999999);

triangle(buffer, 399, 177, 492, 500, 492, 177, 0x999999);

triangle(buffer, 583, 77, 661, 177, 583, 177, 0x999999);

pintar\_estrella(250, 30, buffer);

rectfill(buffer, 373, 403, 393, 423, 0xFF0000);

}

else if(nivel == 5)

{

textout\_centre\_ex(buffer, font, "Nivel 5", 700, 30, 0x999999, 0x000000);

rectfill(buffer, 100, 0, 186, 318, 0x999999);

rectfill(buffer, 186, 0, 540, 92, 0x999999);

rectfill(buffer, 186, 288, 552, 396, 0x999999);

rectfill(buffer, 435, 212, 640, 288, 0x999999);

triangle(buffer, 0, 318, 187, 500, 0, 500, 0x999999);

triangle(buffer, 186, 230, 272, 288, 187, 288, 0x999999);

triangle(buffer, 272, 150, 359, 212, 272, 212, 0x999999);

triangle(buffer, 350, 212, 435, 150, 435, 212, 0x999999);

triangle(buffer, 539, 0, 740, 212, 740, 0, 0x999999);

triangle(buffer, 552, 500, 740, 317, 740, 500, 0x999999);

pintar\_estrella(370, 250, buffer);

rectfill(buffer, 680, 240, 700, 260, 0xFF0000);

}

else

{

textout\_centre\_ex(buffer, font, "GANASTE", 370, 230, 0xFBFF00, 0x000000);

textout\_centre\_ex(buffer, font, "LOGRASTE SUPERAR TODOS LOS NIVELES", 370, 250, 0xFBFF00, 0x000000);

textout\_centre\_ex(buffer, font, "PRESIONA <ESC> PARA SALIR", 370, 270, 0x999999, 0x000000);

}

}

void recargar\_combustible(double &cx, double &cy, double &combustible, BITMAP \*buffer, int nivel)

{

if((nivel == 1 && choque\_rectangulo(cx, cy, 360, 240, 380, 260)) ||

(nivel == 2 && choque\_rectangulo(cx, cy, 360, 440, 380, 460)) ||

(nivel == 3 && choque\_rectangulo(cx, cy, 170, 50, 190, 70)) ||

(nivel == 4 && choque\_rectangulo(cx, cy, 373, 403, 393, 423)) ||

(nivel == 5 && choque\_rectangulo(cx, cy, 680, 240, 700, 260)))

{

combustible = 100;

rectfill(buffer, 50, 50, 50+combustible, 55, 0xFF0000);

}

}

void explosion(double cx, double cy, BITMAP\* buffer, int nivel) ///Crea vectores con distintas lineas las cuales serán disparadas por todo el mapa una vez que la nave choque

{

double x[] = {cx-10, cx+10, cx , cx , cx+15, cx-15, cx+5, cx-10, cx+10, cx-5 , cx-10, cx+10};

double y[] = {cy , cy , cy-15, cy+15, cy-15, cy+15, cy+5, cy-10, cy-10, cy+10, cy , cy };

double dx[] = {7 , 7 , 0 , -7 , -7 , 0 };

double dy[] = {0 , -7 , -7 , -7 , 0 , 7 };

clear(screen);

do

{

clear(buffer);

pintar\_nivel(nivel, buffer);

int j = 0;

for(int i = 0; i <= 10; i+=2)

{

line(buffer, x[i], y[i], x[i+1], y[i+1], 0xffffff);

rotar(x[i+1], y[i+1], x[i], y[i], 15);

x[i] += dx[j];

y[i] += dy[j];

x[i+1] += dx[j];

y[i+1] += dy[j];

j++;

}

textout\_centre\_ex(buffer, font, "PRESIONE <R> PARA REINICIAR", 370, 240, 0xFBFF00, 0x000000);

textout\_centre\_ex(buffer, font, "PRESIONE <ESC> PARA SALIR", 370, 250, 0xFBFF00, 0x000000);

blit(buffer, screen, 0, 0, 0, 0, 740, 500);

rest(20);

} while(!key[KEY\_ESC] && !key[KEY\_R]);

}

void GameOver(double &cx, double &cy, double &vx, double &vy, int nivel, double &combustible, BITMAP \*buffer) ///Función que se fija si la nave choca con los bordes o con algún obstáculo

{

if((cx-20 <= 0 || cx+20 >= 740 || cy-15 <= 0 || cy+20 >= 500) && nivel <= 5)

{

explosion(cx, cy, buffer, nivel);

reiniciar\_nivel(cx, cy, vx, vy, combustible);

}

else if(choque\_nave(nivel, cx, cy) == true)

{

explosion(cx, cy, buffer, nivel);

reiniciar\_nivel(cx, cy, vx, vy, combustible);

}

}

bool atrapar\_estrella(double cx, double cy, BITMAP \*buffer, int nivel) ///Se divide a la nave en pata izquierda, pata derecha y parte central para medir mejor los choques con la estrella

{

double izq1x = cx-20, izq1y = cy;

double izq2x = cx-10, izq2y = cy+20;

double der1x = cx+10, der1y = cy;

double der2x = cx+20, der2y = cy+20;

double cen1x = cx-10, cen1y = cy-15;

double cen2x = cx+10, cen2y = cy;

if(nivel == 1)

{

if((izq1x <= 715 && izq2x >= 685 && izq1y <= 465 && izq2y >= 450) ||

(choque\_triangulo(695, 450, 700, 440, izq1x, izq1y, izq2x, izq2y, "abajo")) ||

(choque\_triangulo(700, 440, 705, 450, izq1x, izq1y, izq2x, izq2y, "abajo")))

{

return true;

}

else if((der1x <= 715 && der2x >= 685 && der1y <= 465 && der2y >= 450) ||

(choque\_triangulo(695, 450, 700, 440, der1x, der1y, der2x, der2y, "abajo")) ||

(choque\_triangulo(700, 440, 705, 450, der1x, der1y, der2x, der2y, "abajo")))

{

return true;

}

else if((cen1x <= 715 && cen2x >= 685 && cen1y <= 465 && cen2y >= 450) ||

(choque\_triangulo(695, 450, 700, 440, cen1x, cen1y, cen2x, cen2y, "abajo")) ||

(choque\_triangulo(700, 440, 705, 450, cen1x, cen1y, cen2x, cen2y, "abajo")))

{

return true;

}

}

else if(nivel == 2)

{

if((izq1x <= 715 && izq2x >= 685 && izq1y <= 115 && izq2y >= 100))

{

return true;

}

else if((der1x <= 715 && der2x >= 685 && der1y <= 115 && der2y >= 100))

{

return true;

}

else if((cen1x <= 715 && cen2x >= 685 && cen1y <= 115 && cen2y >= 100))

{

return true;

}

}

else if(nivel == 3)

{

if((izq1x <= 45 && izq2x >= 15 && izq1y <= 465 && izq2y >= 450) ||

(choque\_triangulo(30, 440, 35, 450, izq1x, izq1y, izq2x, izq2y, "abajo")) ||

(choque\_triangulo(25, 450, 30, 440, izq1x, izq1y, izq2x, izq2y, "abajo")))

{

return true;

}

else if((der1x <= 45 && der2x >= 15 && der1y <= 465 && der2y >= 450) ||

(choque\_triangulo(30, 440, 35, 450, der1x, der1y, der2x, der2y, "abajo")) ||

(choque\_triangulo(25, 450, 30, 440, der1x, der1y, der2x, der2y, "abajo")))

{

return true;

}

else if((cen1x <= 45 && cen2x >= 15 && cen1y <= 465 && cen2y >= 450) ||

(choque\_triangulo(30, 440, 35, 450, cen1x, cen1y, cen2x, cen2y, "abajo")) ||

(choque\_triangulo(25, 450, 30, 440, cen1x, cen1y, cen2x, cen2y, "abajo")))

{

return true;

}

}

else if(nivel == 4)

{

if(der1x <= 265 && der2x >= 240 && der1y <= 45 && der2y >= 30)

{

return true;

}

}

else if(nivel == 5)

{

if(der1x <= 385 && der2x >= 360 && der1y <= 265 && der2y >= 250)

{

return true;

}

}

return false;

}

bool choque\_rectangulo(double &cx, double &cy, double p1, double p2, double p3, double p4) ///Calcula las posibles colisiones con rectángulos

{

if(cx-20 <= p3 && cy-15 <= p4 && cx+20 >= p1 && cy+20 >= p2)

{

return true;

}

return false;

}

bool choque\_triangulo(double x1, double y1, double x2, double y2, double p1x, double p1y, double p2x, double p2y, string tipo\_triangulo) ///Calcula las posibles colisiones con triángulos

{

double pendiente = (y2-y1)/(x2-x1);

double intersecciony = y1-pendiente\*x1;

if(tipo\_triangulo == "abajo")

{

if(pendiente > 0)

{

if(x1 <= p1x && x2 >= p1x)

{

if(p2y >= pendiente\*p1x + intersecciony)

{

return true;

}

}

else

{

if(p1x <= x1 && x1 <= p2x)

{

if(y1 <= p2y)

{

return true;

}

}

}

}

else if(pendiente < 0)

{

if(x1 <= p2x && x2 >= p2x)

{

if(p2y >= pendiente\*p2x + intersecciony)

{

return true;

}

}

else

{

if(p1x <= x2 && x2 <= p2x)

{

if(y2 <= p2y)

{

return true;

}

}

}

}

}

if(tipo\_triangulo == "arriba")

{

if(pendiente > 0)

{

if(x1 <= p2x && p2x <= x2)

{

if(p1y <= pendiente\*p2x + intersecciony)

{

return true;

}

}

else

{

if(p1x <= x2 && x2 <= p2x)

{

if(p1y <= y2)

{

return true;

}

}

}

return false;

}

else

{

if(x1 <= p1x && p1x <= x2)

{

if(p1y <= pendiente\*p1x + intersecciony)

{

return true;

}

}

else

{

if(p1x <= x1 && x1 <= p2x)

{

if(p1y <= y1)

{

return true;

}

}

}

return false;

}

}

return false;

}

bool choque\_nave(int nivel, double cx, double cy) ///Calcula los choques con los obstáculos de cada nivel

{

double izq1x = cx-20, izq1y = cy;

double izq2x = cx-10, izq2y = cy+20;

double der1x = cx+10, der1y = cy;

double der2x = cx+20, der2y = cy+20;

double cen1x = cx-10, cen1y = cy-15;

double cen2x = cx+10, cen2y = cy;

if(nivel == 2)

{

if (choque\_rectangulo(cx, cy, 100, 110, 250, 500))

{

return true;

}

else if (choque\_rectangulo(cx, cy, 450, 0, 640, 390))

{

return true;

}

}

else if(nivel == 3)

{

if (choque\_triangulo(0, 80, 80, 0, izq1x, izq1y, izq2x, izq2y, "arriba") ||

choque\_triangulo(0, 80, 80, 0, der1x, der1y, der2x, der2y, "arriba") ||

choque\_triangulo(0, 80, 80, 0, cen1x, cen1y, cen2x, cen2y, "arriba") )

{

return true;

}

else if (choque\_triangulo(100, 100, 470, 500, izq1x, izq1y, izq2x, izq2y, "abajo") ||

choque\_triangulo(100, 100, 470, 500, der1x, der1y, der2x, der2y, "abajo") ||

choque\_triangulo(100, 100, 470, 500, cen1x, cen1y, cen2x, cen2y, "abajo") )

{

return true;

}

else if (choque\_triangulo(270, 0, 640, 400, izq1x, izq1y, izq2x, izq2y, "arriba") ||

choque\_triangulo(270, 0, 640, 400, der1x, der1y, der2x, der2y, "arriba") ||

choque\_triangulo(270, 0, 640, 400, cen1x, cen1y, cen2x, cen2y, "arriba") )

{

return true;

}

else if (choque\_triangulo(660, 500, 740, 420, izq1x, izq1y, izq2x, izq2y, "abajo") ||

choque\_triangulo(660, 500, 740, 420, der1x, der1y, der2x, der2y, "abajo") ||

choque\_triangulo(660, 500, 740, 420, cen1x, cen1y, cen2x, cen2y, "abajo") )

{

return true;

}

}

else if(nivel == 4)

{

if (choque\_rectangulo(cx, cy, 0, 0, 142, 72))

{

return true;

}

else if (choque\_rectangulo(cx, cy, 110, 153, 204, 260))

{

return true;

}

else if (choque\_rectangulo(cx, cy, 204, 77, 305, 260))

{

return true;

}

else if (choque\_rectangulo(cx, cy, 305, 0, 492, 77))

{

return true;

}

else if (choque\_rectangulo(cx, cy, 492, 177, 583, 500))

{

return true;

}

else if (choque\_rectangulo(cx, cy, 583, 177, 661, 309))

{

return true;

}

else if (choque\_triangulo(0, 270, 240, 500, izq1x, izq1y, izq2x, izq2y, "abajo") ||

choque\_triangulo(0, 270, 240, 500, der1x, der1y, der2x, der2y, "abajo") ||

choque\_triangulo(0, 270, 240, 500, cen1x, cen1y, cen2x, cen2y, "abajo") )

{

return true;

}

else if (choque\_triangulo(150, 260, 305, 425, izq1x, izq1y, izq2x, izq2y, "arriba") ||

choque\_triangulo(150, 260, 305, 425, der1x, der1y, der2x, der2y, "arriba") ||

choque\_triangulo(150, 260, 305, 425, cen1x, cen1y, cen2x, cen2y, "arriba") )

{

if(choque\_rectangulo(cx, cy, 142, 72, 204, 143) || choque\_rectangulo(cx, cy, 142, 0, 305, 77))

{

return false;

}

else

{

return true;

}

}

else if (choque\_triangulo(400, 177, 492, 500, izq1x, izq1y, izq2x, izq2y, "arriba") ||

choque\_triangulo(400, 177, 492, 500, der1x, der1y, der2x, der2y, "arriba") ||

choque\_triangulo(400, 177, 492, 500, cen1x, cen1y, cen2x, cen2y, "arriba"))

{

if(choque\_rectangulo(cx, cy, 399, 177, 492, 178)) ///Si choca con la parte de arriba

{

return true;

}

else if(choque\_rectangulo(cx, cy, 305, 77, 492, 176))

{

return false;

}

else

{

return true;

}

}

else if (choque\_triangulo(583, 77, 661, 177, izq1x, izq1y, izq2x, izq2y, "abajo") ||

choque\_triangulo(583, 77, 661, 177, der1x, der1y, der2x, der2y, "abajo") ||

choque\_triangulo(583, 77, 661, 177, cen1x, cen1y, cen2x, cen2y, "abajo") )

{

if(choque\_rectangulo(cx, cy, 583, 309, 740, 500))

{

return false;

}

else

{

return true;

}

}

}

else if(nivel == 5)

{

if (choque\_rectangulo(cx, cy, 100, 0, 186, 318))

{

return true;

}

else if (choque\_rectangulo(cx, cy, 186, 0, 540, 92))

{

return true;

}

else if (choque\_rectangulo(cx, cy, 186, 288, 552, 396))

{

return true;

}

else if (choque\_rectangulo(cx, cy, 435, 212, 640, 288))

{

return true;

}

else if (choque\_rectangulo(cx, cy, 186, 396, 552, 500)) ///Rectangulo imaginario 1

{

return false;

}

else if (choque\_rectangulo(cx, cy, 272, 212, 435, 288)) ///Rectangulo imaginario 2

{

if(choque\_rectangulo(cx, cy, 272, 212, 359, 213)) ///Rectangulo imaginario para que choque abajo del triangulo

{

return true;

}

else

{

return false;

}

}

else if (choque\_triangulo(0, 318, 187, 500, izq1x, izq1y, izq2x, izq2y, "abajo") ||

choque\_triangulo(0, 318, 187, 500, der1x, der1y, der2x, der2y, "abajo") ||

choque\_triangulo(0, 318, 187, 500, cen1x, cen1y, cen2x, cen2y, "abajo") )

{

return true;

}

else if (choque\_triangulo(186, 230, 272, 288, izq1x, izq1y, izq2x, izq2y, "abajo") ||

choque\_triangulo(186, 230, 272, 288, der1x, der1y, der2x, der2y, "abajo") ||

choque\_triangulo(186, 230, 272, 288, cen1x, cen1y, cen2x, cen2y, "abajo") )

{

return true;

}

else if (choque\_triangulo(272, 150, 359, 212, izq1x, izq1y, izq2x, izq2y, "abajo") ||

choque\_triangulo(272, 150, 359, 212, der1x, der1y, der2x, der2y, "abajo") ||

choque\_triangulo(272, 150, 359, 212, cen1x, cen1y, cen2x, cen2y, "abajo") ||

choque\_rectangulo(cx, cy, 272, 212, 359, 213))

{

return true;

}

else if (choque\_triangulo(350, 212, 435, 150, izq1x, izq1y, izq2x, izq2y, "abajo") ||

choque\_triangulo(350, 212, 435, 150, der1x, der1y, der2x, der2y, "abajo") ||

choque\_triangulo(350, 212, 435, 150, cen1x, cen1y, cen2x, cen2y, "abajo") ||

choque\_rectangulo(cx, cy, 348, 212, 435, 213))

{

return true;

}

else if (choque\_triangulo(539, 0, 740, 212, izq1x, izq1y, izq2x, izq2y, "arriba") ||

choque\_triangulo(539, 0, 740, 212, der1x, der1y, der2x, der2y, "arriba") ||

choque\_triangulo(539, 0, 740, 212, cen1x, cen1y, cen2x, cen2y, "arriba") )

{

return true;

}

else if (choque\_triangulo(552, 500, 740, 317, izq1x, izq1y, izq2x, izq2y, "abajo") ||

choque\_triangulo(552, 500, 740, 317, der1x, der1y, der2x, der2y, "abajo") ||

choque\_triangulo(552, 500, 740, 317, cen1x, cen1y, cen2x, cen2y, "abajo") )

{

return true;

}

}

return false;

}

void pasar\_nivel(double &cx, double &cy, double &vx, double &vy, int &nivel, double &combustible, BITMAP \*buffer) ///Se calcula si hubo un paso de nivel y setea las coordenadas de la nave para el próximo

{

if (atrapar\_estrella(cx, cy, buffer, nivel))

{

if(nivel <= 5)

{

nivel++;

}

if(nivel == 2)

{

cx = 50, cy = 450;

vx = 0, vy = -1;

combustible = 100;

}

else if(nivel == 3)

{

cx = 700, cy = 100;

vx = 0, vy = -1;

combustible = 100;

}

else if(nivel == 4)

{

cx = 660, cy = 410;

vx = 0, vy = -1;

combustible = 100;

}

else if(nivel == 5)

{

cx = 50, cy = 100;

vx = 0, vy = -1;

combustible = 100;

}

else

{

cx = 370, cy = 50;

vx = 0, vy = -1;

combustible = 100;

}

while(!key[KEY\_ENTER])

{

textout\_centre\_ex(screen, font, "NIVEL SUPERADO", 370, 250, 0xFBFF00, 0x000000);

textout\_centre\_ex(screen, font, "PRESIONE <ENTER> PARA AVANZAR", 370, 265, 0xFBFF00, 0x000000);

rest(20);

}

}

}

void reiniciar\_nivel(double &cx, double &cy, double &vx, double &vy, double &combustible) ///Reinicia cada nivel con sus coordenadas correspondientes

{

if(nivel == 1)

{

cx = 100, cy = 100;

vx = 0, vy = -1;

combustible = 100;

}

else if(nivel == 2)

{

cx = 50, cy = 450;

vx = 0, vy = -1;

combustible = 100;

}

else if(nivel == 3)

{

cx = 700, cy = 100;

vx = 0, vy = -1;

combustible = 100;

}

else if(nivel == 4)

{

cx = 660, cy = 410;

vx = 0, vy = -1;

combustible = 100;

}

else if(nivel == 5)

{

cx = 50, cy = 100;

vx = 0, vy = -1;

combustible = 100;

}

}

};

int main()

{

allegro\_init();

install\_keyboard(); ///Habilita el uso del teclado

install\_mouse(); ///Habilita el uso del mouse

set\_color\_depth(32); ///Define los bits del juego

set\_gfx\_mode(GFX\_AUTODETECT\_WINDOWED, 740, 500, 0, 0); ///Define la pantalla del juego

Juego game;

game.menu();

allegro\_exit();

if(game.getNivel() > 5)

{

Ganador ganador1, ganador2, ganador\_actual;

list<Ganador> lista\_ganadores; ///Se crea una lista de ganadores

list<Ganador>::iterator lista\_iter;

ganador1.setNombre("Pedro");

ganador1.setPais("Colombia");

ganador1.setEdad(33);

ganador2.setNombre("Matias");

ganador2.setPais("Venezuela");

ganador2.setEdad(13);

string nombre\_ganador, pais\_ganador;

unsigned int edad\_ganador;

cout << "FELICIDADES POR GANAR EL JUEGO" << endl;

cout << "Inserte los siguientes datos para guardarlos en la lista de ganadores: " << endl;

cout << "Nombre: ";

cin >> nombre\_ganador;

cout << "Pais: ";

cin >> pais\_ganador;

cout << "Edad: ";

cin >> edad\_ganador;

ganador\_actual.setNombre(nombre\_ganador);

ganador\_actual.setPais(pais\_ganador);

ganador\_actual.setEdad(edad\_ganador);

lista\_ganadores.push\_back(ganador1);

lista\_ganadores.push\_back(ganador2);

lista\_ganadores.push\_back(ganador\_actual);

cout << endl<< "Excelente, ya formas parte de esta lista: " << endl << endl;

for(lista\_iter = lista\_ganadores.begin(); lista\_iter != lista\_ganadores.end(); lista\_iter++)

{

cout << "Nombre: " << lista\_iter->getNombre() << " - " << "Pais: " << lista\_iter->getPais() << " - " << "Edad: " << lista\_iter->getEdad() << endl;

}

lista\_ganadores.clear();

/\*\* El trabajo con la lista se realizó para guardar el nombre del ganador junto a otros inventados para figurar en ella.

Esta se recorre con un iterador y finalmente se procede a eliminarla\*/

cout << endl << "Gracias por jugar ESTRELLADOS" << endl;

cout << endl << "Creado por: Ivan Marano" << endl;

}

return 0;

}

END\_OF\_MAIN();