

Instituto Tecnológico y de Estudios  
Superiores de Occidente – ITESO



**ITESO**

**Universidad Jesuita  
de Guadalajara**

**Materia:** Fundamentos de programación

**Profesor:** Luis Alberto Sánchez Gatica

**PROYECTO INTEGRADOR**

**Fecha:** 02 de diciembre del 2016

**Temas:** The Snake Game

**Autor(es):** Chávez Medina Mariana

Madriz Almanza Omar Antonio

## Introducción

Para iniciar con el desarrollo de este proyecto, diversos puntos se tomaron en discusión. Primeramente, tuvimos que tomar en cuenta las habilidades técnicas que cada proyecto requería, así como lo que queríamos desarrollar.

Después de un análisis, tanto de habilidades, como de conocimientos previos, decidimos desarrollar el juego de “The Snake”. Juego en el cual, una serpiente atraviesa un espacio comiendo ratones y objetos bonus, evitando chocar consigo misma o con humanos. Sin duda, sería un proyecto ambicioso y retador, el cual demandaría de nuestro tiempo y, sobre todo, análisis y trabajo.

Utilizando la librería de gráficos Allegro, en lenguaje C, se desarrollaría el proyecto atendiendo los siguientes puntos generales:

- Un menú de inicio con o Título de juego
- Botones: Play, Instuctions, Exit
- Al hacer clic en Play, aparece display con fondo negro y la víbora verde en medio.
- Un marcador de puntos en la esquina superior derecha.

Después de puntualizar las especificaciones generales anteriores, profundizamos con distintas características como el aumento de nivel, bonus y puntos menos para el usuario.

A pesar de ser completamente un nuevo concepto el manejo de gráficos para nosotros, el funcionamiento realmente dependería de conocimientos vistos previamente, ciclos y condiciones.

Desde un punto práctico, el juego tendría como fin, el entretenimiento y la estrategia. Por medio de pasos planeados, el usuario tendrá que mover a la serpiente para atrapar a los ratones y, además, evitar chocar consigo misma o con el humano. El juego parece simple en los primeros niveles, pero la dificultad aumentará al mismo tiempo que la serpiente aumente de tamaño.

## Código fuente

```
/*
 * Snake2-1.c
 *   * Created on: 02/12/2016
 *   Author: Chávez y Madriz
 */

#include <stdio.h>
#include <allegro5/allegro.h>
#include <stdlib.h>
#include <time.h>
#include <math.h>
#include <allegro5/allegro_font.h>
#include <allegro5/allegro_ttf.h>
#include "structs.h"

const float FPS = 60;
const int SCREEN_W = 962;
const int SCREEN_H = 541;
Point mouse;
Snake mySnake;
Point apple;
Point chick;
Point croc;
Point human;
Point body;
enum MYKEYS {
    KEY_UP, KEY_DOWN, KEY_LEFT, KEY_RIGHT, SPACEBAR
};

enum ESTADOS {
    MENU_INICIAL, CERRAR, INSTRUCCIONES, JUEGO, PAUSA, GAME_OVER
};

enum BONUS {
    APPLE, CHICKEN, HUMAN, CROC
};

int main(int argc, char **argv)
{
    ALLEGRO_DISPLAY *display = NULL;
    ALLEGRO_BITMAP *menu,*instruct, *end;
    ALLEGRO_EVENT_QUEUE *event_queue = NULL;
    ALLEGRO_TIMER *timer = NULL;
    ALLEGRO_TIMER *gameTimer = NULL;
    ALLEGRO_FONT *font=NULL;

    int Estados = MENU_INICIAL;

    srand(time(NULL));
    int count = 0;
```

```

//double countTimer;
//double showOff;
int bonus; //variable para hacer random el bonus*/
int score=0; //para ir contando puntaje

//Variables para click en mouse
float clickX;
float clickY;

//Iniciación del mouse_comida
mouse.size_x = 18;
mouse.size_y = 11;
mouse.x = rand()%(SCREEN_W-16);
mouse.y = rand()%(SCREEN_H-16);
mouse.image=NULL;
mouse.puntaje = 5;

//dimensiones del mouse
int xi_m[4], yi_m[4];

//Iniciación del apple
apple.size_x = 10;
apple.size_y = 14;
apple.x = rand()%(SCREEN_W-16);
apple.y = rand()%(SCREEN_H-16);
apple.image=NULL;
apple.puntaje = 7;

//Iniciación de la gallina
chick.size_x = 18;
chick.size_y = 20;
chick.x = rand()%(SCREEN_W-16);
chick.y = rand()%(SCREEN_H-16);
chick.image=NULL;
chick.puntaje = 10;

//Iniciación del cocodrilo
croc.size_x = 39;
croc.size_y = 18;
croc.x = rand()%(SCREEN_W-16);
croc.y = rand()%(SCREEN_H-16);
croc.image=NULL;
croc.puntaje = -5;

//Iniciación de humano
human.size_x = 14;
human.size_y = 27;
human.x = rand()%(SCREEN_W-16);
human.y = rand()%(SCREEN_H-16);
human.image=NULL;
human.puntaje = 0;

//dimensiones del humano
int xi_h[4], yi_h[4];

```

```

//Iniciación del snake
mySnake.points[0].size_x = 16;
mySnake.points[0].size_y = 16;
mySnake.points[1].size_x = 16;
mySnake.points[1].size_y = 16;
mySnake.points[2].size_x = 16;
mySnake.points[2].size_y = 16;
body.x= mySnake.points[1].x;
body.y= mySnake.points[1].y;

mySnake.dx = 0;
mySnake.dy = 0;
mySnake.length = 4;
mySnake.offset = 0;
mySnake.points[0].x = SCREEN_W / 2.0 -
(mySnake.points[0].size_x*mySnake.length) / 2.0;
mySnake.points[0].y = SCREEN_H / 2.0 -
(mySnake.points[0].size_y*mySnake.length) / 2.0;
fillSnake(&mySnake, SCREEN_W, SCREEN_H);

bool key[5] = { false, false, false, false, false };
bool redraw = true;
/*bool up = false;
bool down = false;
bool right = false;
bool left = false;*/

//Banderas para maquina de estado
bool x_Close = false;
bool b1_Instruct = false;
bool b2_Play = false;
bool b3_Exit = false;
bool b4_Menu = true;
bool pause = false;
bool gameOver= false;
bool tryAgain = false;

if(!al_init()) {
    fprintf(stderr, "failed to initialize allegro!\n");
    return -1;
}

if(!al_install_mouse()) {
    fprintf(stderr, "failed to initialize the mouse!\n");
    return -1;
}

if(!al_install_keyboard()) {
    fprintf(stderr, "failed to initialize the keyboard!\n");
    return -1;
}

```

```

timer = al_create_timer(1.0 / FPS);
if(!timer) {
    fprintf(stderr, "failed to create timer!\n");
    return -1;
}

gameTimer = al_create_timer(1.0);
if(!timer) {
    fprintf(stderr, "failed to create timer!\n");
    return -1;
}

display = al_create_display(SCREEN_W, SCREEN_H);
if(!display) {
    fprintf(stderr, "failed to create display!\n");
    al_destroy_timer(timer);
    al_destroy_timer(gameTimer);
    return -1;
}

al_set_window_title(display, "The Snake Game");

//Creación del snake

mySnake.points[0].image =
al_create_bitmap(mySnake.points[0].size_x, mySnake.points[0].size_y);
if(!mySnake.points[0].image) {
    fprintf(stderr, "failed to create snakeBody bitmap!\n");
    al_destroy_display(display);
    al_destroy_timer(timer);
    al_destroy_timer(gameTimer);
    return -1;
}

al_set_target_bitmap(mySnake.points[0].image); //indica que va a
pintar en el bouncer y no la pantalla

al_clear_to_color(al_map_rgb(80, 164, 30)); //Gracias a la linea
anterior esto pinto el bouncer

//Creación de villanos
al_init_image_addon();

ALLEGRO_PATH *path3 = al_get_standard_path(ALLEGRO_RESOURCES_PATH);
al_set_path_filename(path3, "Raton.png");
mouse.image = al_load_bitmap(al_path_cstr(path3, '/'));

if(!mouse.image) {
    fprintf(stderr, "failed to create mouse bitmap!\n");
    al_destroy_bitmap(mySnake.points[0].image);
    al_destroy_display(display);
    al_destroy_timer(timer);
    al_destroy_timer(gameTimer);
}

```

```

        return -1;
    }

    ALLEGRO_PATH *path4 = al_get_standard_path(ALLEGRO_RESOURCES_PATH);
    al_set_path_filename(path4, "Coco.png");
    croc.image = al_load_bitmap(al_path_cstr(path4, '/'));

    if(!croc.image) {
        fprintf(stderr, "failed to create mouse bitmap!\n");
        al_destroy_bitmap(mySnake.points[0].image);
        al_destroy_bitmap(mouse.image);
        al_destroy_display(display);
        al_destroy_timer(timer);
        al_destroy_timer(gameTimer);
        return -1;
    }

    ALLEGRO_PATH *path5 = al_get_standard_path(ALLEGRO_RESOURCES_PATH);
    al_set_path_filename(path5, "Gallina.png");
    chick.image = al_load_bitmap(al_path_cstr(path5, '/'));

    if(!chick.image) {
        fprintf(stderr, "failed to create mouse bitmap!\n");
        al_destroy_bitmap(mySnake.points[0].image);
        al_destroy_bitmap(mouse.image);
        al_destroy_bitmap(croc.image);
        al_destroy_display(display);
        al_destroy_timer(timer);
        al_destroy_timer(gameTimer);
        return -1;
    }

    ALLEGRO_PATH *path6 = al_get_standard_path(ALLEGRO_RESOURCES_PATH);
    al_set_path_filename(path6, "Manzana.png");
    apple.image = al_load_bitmap(al_path_cstr(path6, '/'));

    if(!apple.image) {
        fprintf(stderr, "failed to create mouse bitmap!\n");
        al_destroy_bitmap(mySnake.points[0].image);
        al_destroy_bitmap(mouse.image);
        al_destroy_bitmap(croc.image);
        al_destroy_bitmap(chick.image);
        al_destroy_display(display);
        al_destroy_timer(timer);
        al_destroy_timer(gameTimer);
        return -1;
    }

    ALLEGRO_PATH *path7 = al_get_standard_path(ALLEGRO_RESOURCES_PATH);
    al_set_path_filename(path7, "Humano.png");
    human.image = al_load_bitmap(al_path_cstr(path7, '/'));

    if(!human.image) {
        fprintf(stderr, "failed to create mouse bitmap!\n");
        al_destroy_bitmap(mySnake.points[0].image);

```

```

        al_destroy_bitmap(mouse.image);
        al_destroy_bitmap(croc.image);
        al_destroy_bitmap(chick.image);
        al_destroy_bitmap(apple.image);
        al_destroy_display(display);
        al_destroy_timer(timer);
        al_destroy_timer(gameTimer);
        return -1;
    }

```

`al_set_target_bitmap(al_get_backbuffer(display));` //vuelve a cambiar el target `al` display, el `backbuffer` muestra lo que se ha hecho en la pantalla.

```

//inicializar la cola de eventos
event_queue = al_create_event_queue();
if(!event_queue) {
    fprintf(stderr, "failed to create event_queue!\n");
    al_destroy_bitmap(mouse.image);
    al_destroy_bitmap(mySnake.points[0].image);
    al_destroy_bitmap(croc.image);
    al_destroy_bitmap(chick.image);
    al_destroy_bitmap(apple.image);
    al_destroy_display(display);
    al_destroy_timer(timer);
    al_destroy_timer(gameTimer);
    return -1;
}

```

`//Eventos para teclado`  
`al_register_event_source(event_queue,`  
`al_get_display_event_source(display));` //Registra los eventos de la cola.

`al_register_event_source(event_queue,`  
`al_get_timer_event_source(timer));`

`al_register_event_source(event_queue,`  
`al_get_keyboard_event_source());`

`al_register_event_source(event_queue, al_get_mouse_event_source());`

`al_init_primitives_addon();` //Mariana

`al_init_font_addon();`

`al_init_ttf_addon();`

`al_clear_to_color(al_map_rgb(0,0,0));`

`al_init_image_addon();` //Inicia la imagen (MARIANA)



```

// Imágenes a usarse en el menu de inicio

ALLEGRO_PATH *path = al_get_standard_path(ALLEGRO_RESOURCES_PATH);
al_set_path_filename(path, "menuDisplay.png");
menu = al_load_bitmap(al_path_cstr(path, '/'));
al_draw_bitmap(menu, 0, 0, 0);

ALLEGRO_PATH *path2 = al_get_standard_path(ALLEGRO_RESOURCES_PATH);
al_set_path_filename(path2, "instrDisplay.png");
instruct = al_load_bitmap(al_path_cstr(path2, '/'));

ALLEGRO_PATH *path9 = al_get_standard_path(ALLEGRO_RESOURCES_PATH);
al_set_path_filename(path9, "gameOverDisplay.png");
end = al_load_bitmap(al_path_cstr(path9, '/'));

//creación de path para la fuente de texto
ALLEGRO_PATH *path8 = al_get_standard_path(ALLEGRO_RESOURCES_PATH);
al_set_path_filename(path8, "unispace.ttf");
font = al_load_ttf_font(al_path_cstr(path8, '/'), 30, 0);

if (!font)
{
    fprintf(stderr, "failed to create font!\n");
    al_destroy_bitmap(mySnake.points[0].image);
    al_destroy_bitmap(mouse.image);
    al_destroy_bitmap(croc.image);
    al_destroy_bitmap(chick.image);
    al_destroy_bitmap(apple.image);
    al_destroy_display(display);
    al_destroy_timer(timer);
    al_destroy_timer(gameTimer);
    return -1;
}

al_flip_display();//Lo que se pinto en el buffer se va a la
pantalla principal

al_start_timer(timer); //Timer para teclado

al_start_timer(gameTimer); //Timer de juego

while(Estados!= CERRAR)
{
    ALLEGRO_EVENT ev;
    al_wait_for_event(event_queue, &ev);

    //Funcionalidad del mouse
    if(ev.type==ALLEGRO_EVENT_MOUSE_BUTTON_DOWN && (Estados==
MENU_INICIAL))
    {
        clickX = ev.mouse.x;
        clickY = ev.mouse.y;

        if ((clickX > 353 && clickX < 591) && (clickY > 478 &&
clickY < 542))

```

```

        {
            b3_Exit = true;
        }

        else if ((clickX > 345 && clickX < 603) && (clickY >
388 && clickY < 458))
        {
            al_draw_bitmap(instruct, 0, 0, 0);
            al_flip_display();
            b1_Instruct = true;
        }

        else if ((clickX > 353 && clickX < 591) && (clickY >
305 && clickY < 368))
        {//entra codigo de serpiente HOMER
            b2_Play = true;
            al_clear_to_color(al_map_rgb(0,0,0));
            al_flip_display();
        }
        else if(ev.type == ALLEGRO_EVENT_DISPLAY_CLOSE)
        {
            x_Close = true;
        }
    }

    if(ev.type==ALLEGRO_EVENT_MOUSE_BUTTON_DOWN && (Estados
==INSTRUCCIONES))
    {

        clickX = ev.mouse.x;
        clickY = ev.mouse.y;

        if((clickX > 28 && clickX < 262) && (clickY > 445 &&
clickY < 500))
        {
            al_draw_bitmap(menu,0,0,0);
            al_flip_display();
            b4_Menu = true;
        }
        else if(ev.type == ALLEGRO_EVENT_DISPLAY_CLOSE)
        {
            x_Close = true;
        }
    }

    if((Estados==JUEGO))
    {
        while(Estados !=CERRAR)
        {
            //movimiento de la serpiente

            al_wait_for_event(event_queue, &ev);

```

```

        if(ev.type == ALLEGRO_EVENT_TIMER &&
(Estados==JUEGO))
        {
            count++;

            if(count!=20)
            {
                if(key[KEY_UP] ){//&&
mySnake.points[0].y >= 4.0) {

                    mySnake.dy = -16.0;
                    mySnake.dx= 0;
                    /*up= true;
                    down= false;
                    left = false;
                    right = false;*/

                }

                if(key[KEY_DOWN] ){//&&
mySnake.points[0].y <= SCREEN_H -mySnake.points[0].size_y - 4.0) {
                    mySnake.dy = 16.0;
                    mySnake.dx= 0;
                    /*down = true;
                    up= false;
                    left= false;
                    right = false;*/

                }

                if(key[KEY_LEFT] ){//&&
mySnake.points[0].x >= 4.0) {

                    mySnake.dx = -16.0;
                    mySnake.dy= 0;
                    /*left = true;
                    down = false;
                    right =false;
                    up = false;*/

                }

                if(key[KEY_RIGHT] ){//&&
mySnake.points[0].x <= SCREEN_W - mySnake.points[0].size_x - 4.0) {
                    mySnake.dx = 16.0;
                    mySnake.dy= 0;
                    /*    right = true;
                    down = false;
                    left = false;
                    up = false;*/

                }
            }
        }
    else
    {
        count=0;

        int c;

        if(mySnake.dx!=0 || mySnake.dy!=0)

```

```

c>=1;c--)

mySnake.points[c-1].x;

mySnake.points[c-1].y;

for(c=mySnake.length-1;

{
    mySnake.points[c].x =

    mySnake.points[c].y =

}

mySnake.points[0].y += mySnake.dy;
mySnake.points[0].x += mySnake.dx;

if(mySnake.points[0].y < 4.0 ||
mySnake.points[0].y > SCREEN_H - mySnake.points[0].size_x - 4.0
||mySnake.points[0].x < 4.0 || mySnake.points[0].x > SCREEN_W -
/*mySnake.points[0].size_x -*/ 4.0)
{
    if(mySnake.points[0].y <
4.0)
        mySnake.points[0].y=SCREEN_H - mySnake.points[0].size_y;
    else if(mySnake.points[0].y
> SCREEN_H - mySnake.points[0].size_x - 4.0)
        mySnake.points[0].y=
mySnake.points[0].size_y;
    else if (mySnake.points[0].x
< 4.0)
        mySnake.points[0].x=
SCREEN_W - mySnake.points[0].size_x;
    else if(mySnake.points[0].x
> SCREEN_W - mySnake.points[0].size_x - 4.0)
        mySnake.points[0].x=
mySnake.points[0].size_x;
}

//Dimensión de la cabeza del snake
int x1= mySnake.points[0].x;
int x2= mySnake.points[0].x +

int y1= mySnake.points[0].y;
int y2= mySnake.points[0].y +

mySnake.ultimo_x =
mySnake.points[mySnake.length - 1].x;
mySnake.ultimo_y =
mySnake.points[mySnake.length - 1].y;

if(intersects(x1, x2, y1, y2,
{
    mouse.x= rand()%(SCREEN_W-
16);

```

```

16);

mouse.y= rand()%(SCREEN_H-

mySnake.points[mySnake.length].x = mySnake.ultimo_x;

mySnake.points[mySnake.length].y = mySnake.ultimo_y;
    mySnake.length ++;
    score+=5;
}

if(intersects(x1, x2, y1, y2,
human, xi_h, yi_h))
{
    gameOver= true;
}

//Dimensiones del cuerpo del snake
int xi_s[4];
int yi_s[4];

body.x= mySnake.points[1].x;
body.y= mySnake.points[1].y;

int z;
for(z=1; z< mySnake.length; z++){
    //if(intersects(x1, x2, y1,
y2, mySnake.points[z], xi_s, yi_s))
        if(x1==mySnake.points[z].x
&& y1==mySnake.points[z].y)
        {

            fprintf(stderr,"drama\n");

                                gameOver = true;
                                }
        }
    redraw = true;
}
}
else if(ev.type == ALLEGRO_EVENT_DISPLAY_CLOSE)
{
    x_Close= true;
}
else if(ev.type == ALLEGRO_EVENT_KEY_DOWN)
{
//Evento cuando la tecla se presiona.
switch(ev.keyboard.keycode) {
case ALLEGRO_KEY_UP:
    key[KEY_UP] = true;
    break;

case ALLEGRO_KEY_DOWN:
    key[KEY_DOWN] = true;
    break;

case ALLEGRO_KEY_LEFT:

```

```

        key[KEY_LEFT] = true;
        break;

    case ALLEGRO_KEY_RIGHT:
        key[KEY_RIGHT] = true;
        break;

    case ALLEGRO_KEY_SPACE:
        key[SPACEBAR] = true;
        pause = true;
        break;
    }
}
else if(ev.type == ALLEGRO_EVENT_KEY_UP) {
    switch(ev.keyboard.keycode) {
        case ALLEGRO_KEY_UP:
            key[KEY_UP] = false;
            break;

        case ALLEGRO_KEY_DOWN:
            key[KEY_DOWN] = false;
            break;

        case ALLEGRO_KEY_LEFT:
            key[KEY_LEFT] = false;
            break;

        case ALLEGRO_KEY_RIGHT:
            key[KEY_RIGHT] = false;
            break;

        case ALLEGRO_KEY_ESCAPE:
            b3_Exit = true;
            break;

        case ALLEGRO_KEY_SPACE:
            key[SPACEBAR] = false;
            break;
    }
}

if(redraw &&
al_is_event_queue_empty(event_queue)) {
    redraw = false;

    al_clear_to_color(al_map_rgb(0,0,0));

    drawBitmap(&mySnake,

mySnake.points[0].image);

    //    countTimer =
al_get_timer_count(gameTimer); //variable que contiene valor de timer del
juego

```

```

        al_draw_textf(font,
al_map_rgb(255,255,255), 890, 30, ALLEGRO_ALIGN_CENTRE, "%i", score);

        //showOff=countTimer;

        //switch para que aparezcan objetos bonus
de manera random en x tiempo random
        /*
        bonus = rand()%4;
        switch(bonus)
        {
        case APPLE:
            if (showOff == 20)
            {
                al_draw_bitmap(apple.image,
apple.x, apple.y, 0);
            }
            break;

        case CHICKEN:
            if(showOff > 70 && showOff<80)
            {
                al_draw_bitmap(chick.image,
chick.x, chick.y, 0);
            }
            break;

        case CROC:
            if(showOff > 33 && showOff<40)
            {
                al_draw_bitmap(croc.image,
croc.x, croc.y, 0);
            }
            break;

        case HUMAN:
            if(showOff > 83 && showOff<90)
            {
                al_draw_bitmap(human.image,
human.x, human.y, 0);
            }
            break;
        }*/

        al_draw_bitmap(mouse.image, mouse.x,
mouse.y, 0);

        al_draw_bitmap(human.image, human.x,
human.y, 0);

```

```

        al_flip_display();

    }

    if(Estados == PAUSA)
    {
        al_wait_for_event(event_queue, &ev);

        if(ev.type == ALLEGRO_EVENT_TIMER)
        {
            if(key[SPACEBAR])
            {
                b2_Play = true;
            }
            else if(ev.type ==
ALLEGRO_EVENT_DISPLAY_CLOSE) {
                x_Close= true;
            }
        }
    }

    if(Estados == GAME_OVER)
    {
        al_draw_bitmap(end, 0,0,0);
        al_flip_display();
        score-=5;
    }
    switch(Estados)
    {
    case MENU_INICIAL:
        if(x_Close)
            Estados= CERRAR;
        if(b1_Instruct)
            Estados = INSTRUCCIONES;
        if(b2_Play)
            Estados = JUEGO;
        if(b3_Exit)
            Estados = CERRAR;
        break;

    case INSTRUCCIONES:
        if(x_Close)
            Estados = CERRAR;
        if(b4_Menu)
            Estados = MENU_INICIAL;
        if(b3_Exit)
            Estados = CERRAR;
        break;

    case JUEGO:
        if(x_Close)
            Estados = CERRAR;
        if(pause)

```



```

        Estados = PAUSA;
        if(gameOver)
            Estados = GAME_OVER;
        if(b3_Exit)
            Estados = CERRAR;
        break;

    case PAUSA:
        if(x_Close)
            Estados = CERRAR;
        if(pause)
            Estados = JUEGO;
        if(b3_Exit)
            Estados = CERRAR;
        break;

    case GAME_OVER:
        if(x_Close)
            Estados = CERRAR;
        if(tryAgain)
            Estados = JUEGO;
        if(b3_Exit)
            Estados = CERRAR;
        break;
    }
    x_Close = false;
    b1_Instruct = false;
    b2_Play = false;
    b3_Exit = false;
    b4_Menu = false;
    pause = false;
    gameOver= false;
    tryAgain = false;
}

switch(Estados)
{
    case MENU_INICIAL:
        if(x_Close)
            Estados= CERRAR;
        if(b1_Instruct)
            Estados = INSTRUCCIONES;
        if(b2_Play)
            Estados = JUEGO;
        if(b3_Exit)
            Estados = CERRAR;
        break;

    case INSTRUCCIONES:
        if(x_Close)
            Estados = CERRAR;
        if(b4_Menu)
            Estados = MENU_INICIAL;
        if(b3_Exit)

```

```

        Estados = CERRAR;
        break;

    case JUEGO:
        if(x_Close)
            Estados = CERRAR;
        if(pause)
            Estados = PAUSA;
        if(gameOver)
            Estados = GAME_OVER;
        if(b3_Exit)
            Estados = CERRAR;
        break;

    case PAUSA:
        if(x_Close)
            Estados = CERRAR;
        if(pause)
            Estados = JUEGO;
        if(b3_Exit)
            Estados = CERRAR;
        break;

    case GAME_OVER:
        if(x_Close)
            Estados = CERRAR;
        if(tryAgain)
            Estados = JUEGO;
        if(b3_Exit)
            Estados = CERRAR;
        break;
}
x_Close = false;
b1_Instruct = false;
b2_Play = false;
b3_Exit = false;
b4_Menu = false;
pause = false;
gameOver= false;
tryAgain = false;
}

al_destroy_bitmap(mySnake.points[0].image);
al_destroy_bitmap(mouse.image);
al_destroy_bitmap(apple.image);
al_destroy_bitmap(chick.image);
al_destroy_bitmap(croc.image);
al_destroy_bitmap(human.image);
al_destroy_bitmap(instruct);
al_destroy_bitmap(menu);
al_destroy_timer(timer);
al_destroy_timer(gameTimer);
al_destroy_display(display);
al_destroy_event_queue(event_queue);

```

```

        return 0;
    }

```

## LIBRERÍA "STRUCTS"

```

#include "allegro5/allegro.h"
#define MAX_L 32527

typedef struct point
{
    unsigned int size_x, size_y;
    unsigned int x, y;
    ALLEGRO_BITMAP *image;
    int puntaje;
} Point;

typedef struct snake
{
    Point points[MAX_L];
    unsigned int length;
    int dx;
    int dy;
    unsigned int offset;
    int ultimo_x, ultimo_y;
}Snake;

void drawBitmap (Snake * mysnake, ALLEGRO_BITMAP * image)
{
    int i;
    for(i=mysnake->offset; i< (mysnake->length + mysnake->offset); i++)
        al_draw_bitmap(image, mysnake->points[i].x, mysnake->points[i].y,
0);
}

void fillSnake (Snake * mysnake, const int SCREEN_W, const int SCREEN_H)
{
    int i;
    for(i=1; i<MAX_L; i++){
        mysnake->points[i].size_x = 16;
        mysnake->points[i].size_y = 16;
        mysnake->points[i].x = (mysnake->points[i-1].x + mysnake->points[i-
1].size_x)%SCREEN_W;
        mysnake->points[i].y = (mysnake->points[i-1].y)%SCREEN_H;
    }
}

```

```
}
```

```
bool intersects(int x1, int x2, int y1, int y2, Point object, int xi[4], int yi[4])  
{
```

```
    int i, A, B;  
    for(i=0; i<4; i++)  
    {  
        A= i & 1;  
        B= (i>>1)&1;  
  
        xi[i] = object.x + A * object.size_x;  
        yi[i] = object.y + B * object.size_y;  
  
        if((xi[i]>= x1) && (xi[i]<= x2) && (yi[i] >= y1) && (yi[i] <= y2))  
        {  
            return true;  
        }  
    }  
    return false;  
}
```

```
}
```

```
bool bodyIntersect(int x1, int x2, int y1, int y2, Point object, int xi[4], int yi[4], Snake snake)  
{
```

```
    int i, A, B;  
    for(i=0; i<4; i++)  
    {  
        A= i & 1;  
        B= (i>>1)&1;  
  
        xi[i] = object.x + A * object.size_x * snake.length;  
        yi[i] = object.y + B * object.size_y * snake.length;  
  
        if((xi[i]>= x1) && (xi[i]<= x2) && (yi[i] >= y1) && (yi[i] <= y2))  
        {  
            return true;  
        }  
    }  
    return false;  
}
```

```
}
```

## Conclusiones

Al finalizar el desarrollo del juego, concluimos que fue un proyecto muy completo. Además de ampliar nuestro conocimiento, adquirimos diversas competencias, como el trabajo en equipo y organización de tiempos. Nos ayudó bastante el establecer horarios y hacer el máximo para cumplirlos, así como dividirnos las tareas y comunicarnos al finalizar cada una. A manera personal, las barreras se rompieron en nuestras mentes. Construimos de la nada un proyecto completo, buscando y procesando información nueva y unificándola con los conocimientos que teníamos previamente.

Sin duda, más que un proyecto escolar para alcanzar una calificación, el desarrollo de “The Snake” nos trajo una gran satisfacción personal. El saber que no hay nada que un poco de ingenio y ganas, se pueda lograr.