Trabajo Final programación 2

Alumno: Kevin Raúl Meza Castro

Materia: Programación 2

Fecha de Presentación: 06/12/2022

Presentación E instrucciones Del Juego

introducción

El juego es un clon del Ping-Pong hecho en C++ bajo la biblioteca Allegro.h

Está vinculado a bibliotecas personalizadas las cuales almacenan clases, funciones, prototipos, bitmaps y samples de audio.

No tiene utilización de bibliotecas de terceros. Sino la de allegro 4.2.1

- Dimensiones: 1000x500 bits.
- Píxel color: 24.

Inicializaciones (Timer)

INICIALIZACIONES El mecanismo del juego se centra en un TIMER el cual esta instalado (install_timer) e inicializado bajo el comando BPS_TO_TIMER (ticks por segundo) con parámetro 70. Esto controla la velocidad del juego. El tiempo de ejecución es de 70 ticks por segundo.

El timer esta controlado bajo la función incremento el cual lo incrementa en 1.

A su vez tiene instalado el teclado (install_keyboard) y el sonido (install_sound). Cuenta con 7 tipos de sonidos .WAV distintos y 11 bitmaps. 1 buffer central, printeado en la screen con dimensiones (1000x500 bits) y 10 bitmaps personalizados cargados desde imágenes .BMP

Portada

Mediante esta imagen de la portada del juego de pingpong podemos ver las instrucciones del juego

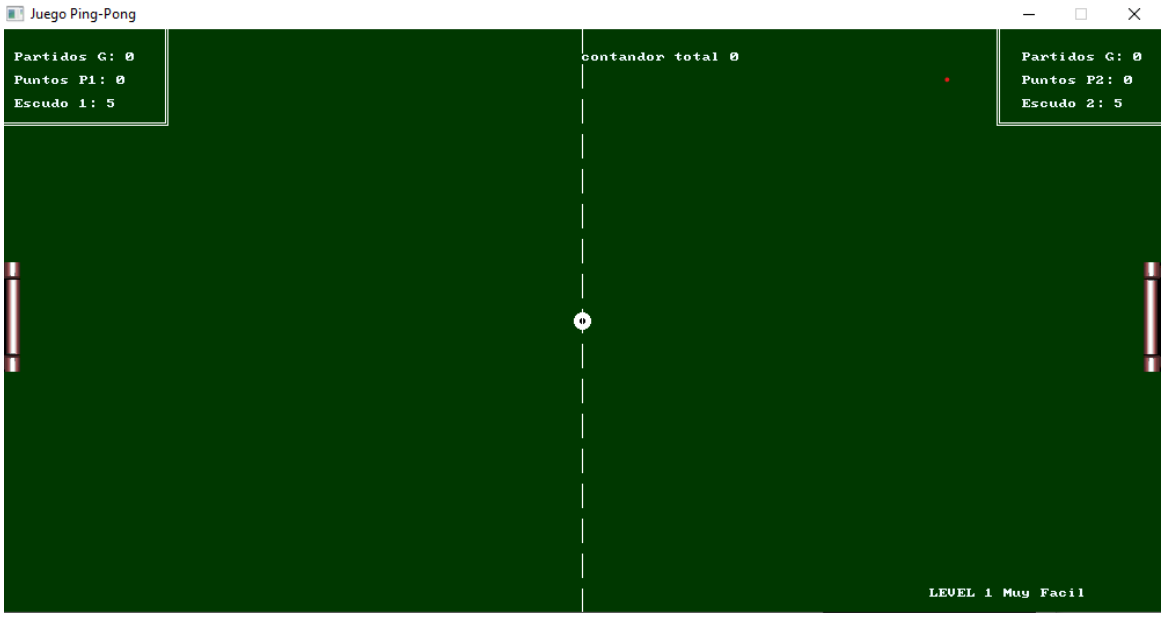
El Jugador 1 se puede mover mediante las teclas w,s

El jugador 2 se puede mover mediante las teclas fleca de arriba y abajo

Para poder pasar de la portada al primer nivel tienes que presionar enter

Para poder hacer el saque es con la barra espaciadora y empezar el juego

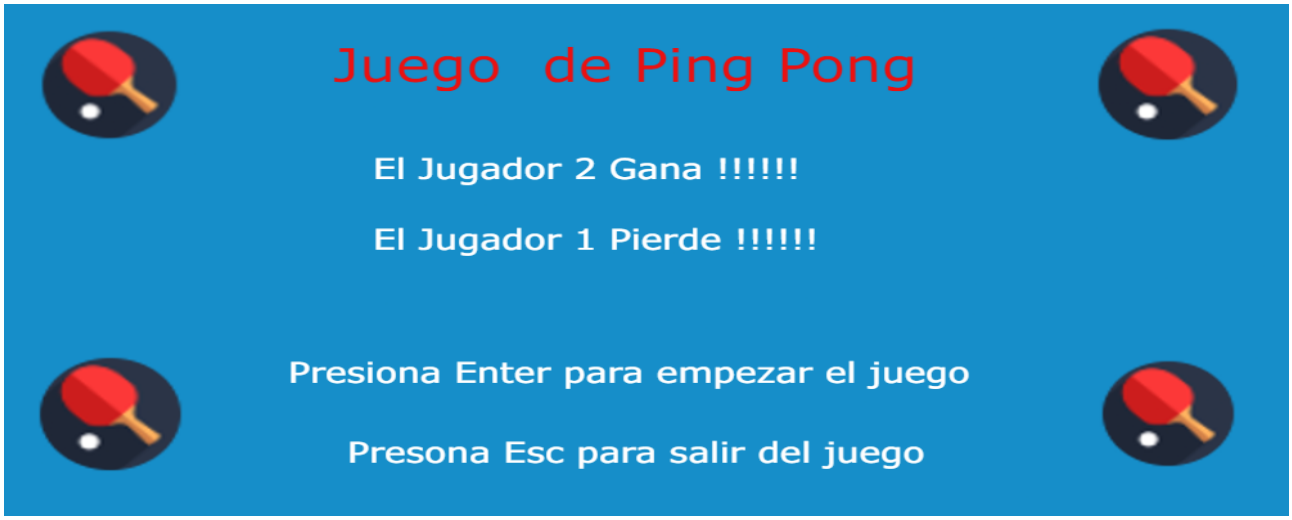
Nivel 1 Juego Pingpong



En todos los niveles tienen un contador de puntos contador (ESCORE)

Tiene un puntaje que contabiliza los puntos anotados

Tiene un contador de escudos esto quiere decir mientras tenga escudos no te podrán anotar puntos hasta que estos lleguen a 0



Cuando el juego finaliza ya sea que gane el jugador 1 o 2 te salda una portada como en la si presionas entre volverás a jugar el juego o si presionas esc saldrás del juego dando así por terminado el programa.

Codigo Del Juego De Ping Pong

```
#include <ctime>
#include <cstdlib>
#include <iostream>
#include <allegro.h>
#define ANCHO 1000
#define ALTO 500
using namespace std;
//DECLARACION DE ESTRUCTURAS
typedef struct
{
   BITMAP *mapa_de_bits;
} bitmaps;
typedef struct
{
   SAMPLE *muestra_de_audio;
} samples;
void incremento( );
void cerrar_ventana( );
void PrimeraVentana( );
template <class type1>///type1 = int
class Jugador
{
   type1 puntaje1;
   type1 puntaje2;
   type1 escudo1;
   type1 escudo2;
public:
   Jugador()
   {
      puntaje1 = 0 ;
      puntaje2 = 0 ;
      escudo1  = 0 ;
```

```
    escudo2  = 0 ;
}
~Jugador()
{

}
void setPuntaje1( type1 Puntaje1 )
{
    this->puntaje1 = Puntaje1;
};
void setPuntaje2( type1 Puntaje2 )
{
    this->puntaje2 = Puntaje2;
};
void setEscudo1( type1 escudo1 )
{
    this->escudo1 = escudo1;
}
void setEscudo2( type1 escudo2 )
{
    this->escudo2 = escudo2;
}

type1 getEscudo1()
{
    return escudo1;
}
type1 getEscudo2()
{
    return escudo2;
}
type1 getPuntaje1()
{
    return puntaje1;
```

```
    }
    type1 getPuntaje2()
    {
        return puntaje2;
    }
};

template <class type1,class type2>///type1 int && type2 bool
class Funcionalidad
{
    type1 resultado;
    type1 cont;
    type1 nivel;
    type1 tiempo;
    type1 mensaje;
    type1 escore;
    type1 ganar1 ;
    type1 ganar2 ;
    type1 contadorP ;
    type2 ventana;
    type2 Portada;
    type2 nuevo;
public:
    Funcionalidad ()
    {
        resultado = 0;
        nuevo = false;
        nivel = 1;
        escore = 0 ;
        ganar1 = 0 ;
        ganar2 = 0 ;
        cont=0;
        contadorP = 5;
        ventana = false;
```

```
    Portada = false;
    tiempo++;
}

~Funcionalidad ()
{

}
void setGanar1(type1 ganar1)
{
    this->ganar1 = ganar1 ;
}
void setGanar2(type1 ganar2)
{
    this->ganar2 = ganar2 ;
}
void setNivel ( type1 nivel )
{
    this->nivel = nivel;
}

void setEscore ( type1 escore )
{
    this->escore = escore;
}

void setResultado ( type1 resultado )
{
    this->resultado = resultado;
}

void setNuevo ( type2 nuevo )
{
    this->nuevo = nuevo;
```

```
}

void setVentana ( type2 ventana )
{
   this->ventana = ventana;
}

void setPortada ( type2 Portada )
{
   this->Portada = Portada;
}

void setTiempo ( type1 Tiempo )
{
   this->tiempo = Tiempo;
}
void setContadorP(type1 contadorP)
{
   this->contadorP=contadorP ;
}
void setMensaje ( type1 Mensaje )
{
   this->mensaje = Mensaje;
}
void setCont(type1 cont)
{
   this->cont ;
}
type1 getCont()
{
   return cont ;
}
type1 getGanar1()
{
```

```
    return ganar1;
}
type1 getGanar2()
{
    return ganar2;
}
type1 getMensaje ()
{
    return mensaje;
}
type1 getResultado ()
{
    return resultado;
}
type1 getTiempo ()
{
    return tiempo;
}

type1 getNivel ( )
{
    return nivel;
}
type1 getEscore()
{
    return escore ;
}

type2 getVentana ()
{
    return ventana;
}

type1 getContadorP()
```

```
    {
        return contadorP ;
    }
    type2 getNuevo ()
    {
        return nuevo;
    }
    type2 getPortada ()
    {
        return Portada;
    }
};
template <class type1>///type1 = int
class Paleta
{
    type1 X;
    type1 Y;
    type1 alto;
    type1 ancho;

public:
    Paleta ()
    {
        X = 0;
        Y = 0;
        alto = 10;
        ancho = 2;
    }
    ~Paleta ()
    {

    }

    void setTamano( type1 ancho, type1 alto )
```

```
{
   this->alto = alto;
   this->ancho = ancho;
}

void setX( type1 x )
{
   this->X = x;
}

void setY( type1 y )
{
   this->Y = y;
}

type1 getX()
{
   return X;
}

type1 getY()
{
   return Y;
}

type1 getAlto()
{
   return alto;
}

type1 getAncho()
{
   return ancho;
}
```

```
};
template <class type1,class type2>///type1 = int - type2 = bool
class Pelota
{
   type1 X;
   type1 Y;
   type1 alto;
   type1 ancho;
   type1 identidad ;
   type1 dirX;
   type1 dirY;
   type2 velocidadX;
   type2 velocidadY;
   type1 velocidad;

public:
   Pelota()
   {
      X=0;
      Y=0;
      dirX=0;
      dirY=0;
      identidad=0;
   }
   ~Pelota ()
   {

   }
   void setVel( type1 velocidad )
   {
      this->velocidad = velocidad;
   }
   void setVelX ( type2 velocidadX )
   {
```

```
    this->velocidadX = velocidadX;
}
void setVelY ( type2 velocidadY )
{
    this->velocidadY = velocidadY;
}
void setX( type1 x)
{
    this->X = x;
}
void setY ( type1 y)
{
    this->Y = y;
}
void setAlto ( type1 alto)
{
    this->alto = alto;
}
void setAncho ( type1 ancho )
{
    this->ancho = ancho;
}

void setDirX ( type1 dirX)
{
    this->dirX = dirX;
}

void setDirY ( type1 dirY )
{
    this->dirY = dirY;
}
void setIdentidad(type1 identidad)
{
```

```
    this->identidad = identidad ;
}
type1 getIdentidad()
{
    return identidad ;
}
type1 getVel ()
{
    return velocidad;
}

type2 getVelX ()
{
    return velocidadX;
}

type2 getVelY ()
{
    return velocidadY;
}

type1 getX ()
{
    return X;
}

type1 getY ()
{
    return Y;
}

type1 getAncho ()
{
    return ancho;
```

```
}

type1 getAlto ()
{
   return alto;
}

type1 getDirX ()
{
   return dirX;
}

type1 getDirY ()
{
   return dirY;
};
void direccion_Pelota(Pelota<int,float> &pelota, Funcionalidad<int,bool> &funciones)
{
   int direccionX,direccionY;
   ///DIRECCION DE LA PELOTA
   srand( time( 0 ) );
   direccionX = ( rand() % 2) + 1; ///DIRECCION X RAND 0/1
   direccionY = ( rand() % 2) + 1; ///DIRECCION Y RAND 0/1
   direccionX = ( direccionX == 1 ) ? - 1 : 1;///1 = -1 , 2 = 1
   direccionY = ( direccionY == 1 ) ? - 1 : 1;///1 = -1 , 2 = 1
   pelota.setDirX( direccionX   );///SETTER
   pelota.setDirY( direccionY   );///SETTER
   funciones.setResultado( 1 );///MARCA DE INICIO
}
void resetPelotaPaleta( Pelota<int,float> &pelota,Pelota<int,float> &pelota1,Pelota<int,float>
&pelota2,Pelota<int,float> &pelota3,Pelota<int,float> &pelota4,Pelota<int,float> &pelota5, Paleta<int>
paleta, int a, Jugador<int> &jugadores, Funcionalidad<int,bool> &funciones, BITMAP *buffer, BITMAP
*jugador1, BITMAP *jugador2, SAMPLE *level_up, SAMPLE *winner)
{
   if ( a == 1 )
```

```
{
   ///GANA PALETA2    //PELOTA SETTEADA EN PALETA1
   if(pelota.getIdentidad()== pelota1.getIdentidad() )
   {
      pelota1.setX( paleta.getX() + paleta.getAncho() + pelota.getAncho() - 3);
      pelota1.setY( paleta.getY() + paleta.getAlto() / 2);
      pelota1.setDirX( 1 );
   }
   if(pelota.getIdentidad()== pelota2.getIdentidad())
   {
      pelota2.setY( (paleta.getY() + paleta.getAlto() / 2) - 20);
      pelota2.setX( paleta.getX() + paleta.getAncho() + pelota.getAncho() - 3);
      pelota2.setDirX( 1 );
   }
   if(pelota.getIdentidad()== pelota3.getIdentidad())
   {
      pelota3.setX( paleta.getX() + paleta.getAncho() + pelota.getAncho() - 3);
      pelota3.setY( (paleta.getY() + paleta.getAlto() / 2)+ 20);
      pelota3.setDirX( 1 );
   }
   if(pelota.getIdentidad()== pelota4.getIdentidad())
   {
      pelota4.setX( paleta.getX() + paleta.getAncho() + pelota.getAncho() - 3);
      pelota4.setY( (paleta.getY() + paleta.getAlto() / 2)+ 40);
      pelota4.setDirX( 1 );
   }
   if(pelota.getIdentidad()== pelota5.getIdentidad())
   {
      pelota5.setX( paleta.getX() + paleta.getAncho() + pelota.getAncho() - 3);
      pelota5.setY( (paleta.getY() + paleta.getAlto() / 2) - 40);
      pelota5.setDirX( 1 );
   }
   jugadores.setPuntaje2( jugadores.getPuntaje2() + 1 );
   funciones.setEscore(funciones.getEscore()+10) ;
```

```
}
else
{
   ///GANA PALETA1  //PELOTA SETTEADA EN PALETA2
   if(pelota.getIdentidad()== pelota1.getIdentidad() )
   {
      pelota1.setX( paleta.getX() - paleta.getAncho() / 2 - pelota.getAncho() + 7 );
      pelota1.setY( paleta.getY() + paleta.getAlto() / 2);
      pelota1.setDirX( -1 );
   }
   if(pelota.getIdentidad()== pelota2.getIdentidad() )
   {
      pelota2.setX( paleta.getX() - paleta.getAncho() / 2 - pelota.getAncho() + 7 );
      pelota2.setY( (paleta.getY() + paleta.getAlto() / 2) - 20);
      pelota2.setDirX( -1 );
   }
   if(pelota.getIdentidad()== pelota3.getIdentidad() )
   {
      pelota3.setX( paleta.getX() - paleta.getAncho() / 2 - pelota.getAncho() + 7 );
      pelota3.setY( (paleta.getY() + paleta.getAlto() / 2)+ 20);
      pelota3.setDirX( -1 );
   }
   if(pelota.getIdentidad()== pelota4.getIdentidad() )
   {
      pelota4.setX( paleta.getX() - paleta.getAncho() / 2 - pelota.getAncho() + 7 );
      pelota4.setY( (paleta.getY() + paleta.getAlto() / 2)+ 40);
      pelota4.setDirX( -1 );
   }
   if(pelota.getIdentidad()== pelota5.getIdentidad() )
   {
      pelota5.setX( paleta.getX() - paleta.getAncho() / 2 - pelota.getAncho() + 7 );
      pelota5.setY( (paleta.getY() + paleta.getAlto() / 2) - 40);
      pelota5.setDirX( -1 );
   }
```

```
   jugadores.setPuntaje1( jugadores.getPuntaje1() + 1 );
   funciones.setEscore(funciones.getEscore()+10) ;
}
if(jugadores.getPuntaje1() == 5 )
{
   funciones.setGanar1(funciones.getGanar1() + 1 );
}
else if (jugadores.getPuntaje2()== 5)
{
   funciones.setGanar2(funciones.getGanar2() + 1 );
}
if((jugadores.getPuntaje1()== 5 || jugadores.getPuntaje2()==5) && funciones.getNivel()== 1)
{
   play_sample( level_up, 200, 150,  1000,  0 );
   funciones.setNivel(2);
   jugadores.setPuntaje1(0);
   jugadores.setPuntaje2(0);
}
else if((jugadores.getPuntaje1()== 5 || jugadores.getPuntaje2()==5) && funciones.getNivel()== 2 )
{
   /// Setea los puntajes en  0
   play_sample( level_up, 200, 150,  1000,  0 );
   funciones.setNivel(3);
   jugadores.setPuntaje1(0);
   jugadores.setPuntaje2(0);
}
else if((jugadores.getPuntaje1()== 5 || jugadores.getPuntaje2()==5) && funciones.getNivel()== 3)
{
   /// Setea los puntajes en  0
   play_sample( level_up, 200, 150,  1000,  0 );
   pelota.setVel(8);
   funciones.setNivel(4);
   jugadores.setPuntaje1(0);
   jugadores.setPuntaje2(0);
```

```
}
else if((jugadores.getPuntaje1()== 5 || jugadores.getPuntaje2()==5)&& funciones.getNivel()== 4 )
{
   /// Setea los puntajes en 0
   play_sample( level_up, 200, 150, 1000, 0 );
   funciones.setNivel(5);
   jugadores.setPuntaje1(0);
   jugadores.setPuntaje2(0);
}
if( funciones.getGanar1() >= 3 )///Gana El Jugador 1
{
   play_sample( winner, 200, 150, 1000, 0 );
   funciones.setContadorP(funciones.getContadorP() +1 );
   while( ! key[KEY_ENTER] )
   {
      ///Setea los valores para la nueva partida
      blit( jugador1, screen, 0, 0, 0, 0, ANCHO, ALTO );///Portada Gano el jugado 1
      jugadores.setPuntaje1( 0 );
      jugadores.setPuntaje2( 0 );
      jugadores.setEscudo1( 5 );
      jugadores.setEscudo2( 5 );
      funciones.setNuevo( true );
      funciones.setNivel(1);
      funciones.setGanar1(0);
      funciones.setGanar2(0);
      funciones.setEscore(0);
      if( key[KEY_ESC] )
      {
         /// Sales del Juego
         allegro_exit();
      }
   }
}
else if( funciones.getGanar2() >= 3 )
```

```
{
   ///Gana el Jugador 2
   play_sample( winner, 200, 150,  1000,  0 );
   funciones.setContadorP(funciones.getContadorP() +1 );
   while(!key[KEY_ENTER])
   {
      /// Seteo los Valores para el nuevo juego
      blit( jugador2, screen, 0, 0, 0, 0, ANCHO, ALTO );///Portada Gano el jugado 2
      jugadores.setPuntaje1( 0 );
      jugadores.setPuntaje2( 0 );
      jugadores.setEscudo1( 5 );
      jugadores.setEscudo2( 5 );
      funciones.setGanar1(0);
      funciones.setGanar2(0);
      funciones.setNivel(1);
      funciones.setNuevo( true );
      funciones.setEscore(0);
      if( key[KEY_ESC] )
      {
         /// Salir Del Juego
         allegro_exit();
      }
   }
}
if( funciones.getNuevo() == true )
{
   ///CASE ENTER: NUEVO = FALSE
   funciones.setNuevo( false );
}
if(funciones.getContadorP()== 6 )
{
   ///SETEAMOS EL CONTADOR DE  PELOTA SEN 1
   funciones.setContadorP(1)  ;
}
```

```
    ///SETTER DE VELOCIDAD LUEGO DEL RESTART, CONTADOR DE TURNOS, MARCA DE RESULTADO/INICIALIZACION
    pelota.setVelX( pelota.getVel() );
    pelota.setVelY( pelota.getVel() );
    funciones.setResultado( 1 );
  }
  void moverPelota( Pelota<int,float> &pelota,Pelota<int,float> &pelota1,Pelota<int,float> &pelota2,Pelota<int,float> &pelota3,Pelota<int,float> &pelota4,Pelota<int,float> &pelota5, Paleta<int> &paleta1, Paleta<int> &paleta2, Jugador<int> &jugadores,Funcionalidad<int,bool> &funciones, BITMAP *buffer, BITMAP *jugador1, BITMAP *jugador2,SAMPLE *hit,SAMPLE *punto,SAMPLE *level_up, SAMPLE *winner, SAMPLE *wall,SAMPLE *escudo)
  {
    if(funciones.getNivel() == 1)
    {
      pelota.setVel(5) ;
    }
    else if(funciones.getNivel() == 2 )
    {
      pelota.setVel(6) ;
    }
    else if (funciones.getNivel() == 3)
    {
      pelota.setVel(7) ;
    }
    else if (funciones.getNivel() == 4)
    {
      pelota.setVel(8) ;
    }
    else if (funciones.getNivel() == 5)
    {
      pelota.setVel(9) ;
    }
    ///DECLARACION DE COORDENADAS Y DIRECCIONES
    int dirx = pelota.getDirX( );
    int diry = pelota.getDirY( );
```

```
int px = pelota.getX( );
int py = pelota.getY( );
int y1 = paleta1.getY( );
int y2 = paleta2.getY( );
if ( px <= paleta1.getAncho() + pelota.getAncho() / 2 )  ///GOLPE BORDE INTERIOR DE PALETA1
{
   if ( ( py + pelota.getAncho() / 2 ) >= y1 && ( py - pelota.getAncho() / 2 ) <= y1 + paleta1.getAlto())
      ///GOLPE LIMITES LARGO DE PALETA
   {
      if( py + pelota.getAncho() / 2 >= y1 && py <= y1 + paleta1.getAlto() - 63 )
         ///GOLPE ESQUINA SUPERIOR DE PALETA1. EJE Y>X
      {
         if( diry == 1 )
         {
            diry = -1;
         }
         play_sample( hit, 200, 150, 1000, 0 );
         pelota.setVelX( pelota.getVel() - 1 );
         pelota.setVelY( pelota.getVel() + 1 );
      }
      else if( py >= y1 + 62 && ( py - pelota.getAncho() / 2 ) <= y1 + paleta1.getAlto() )
         ///GOLPE ESQUINA INFERIOR DE PALETA1. EJE Y>X
      {
         if( diry == -1 )
         {
            diry = 1;
         }
         play_sample( hit, 200, 150, 1000, 0 );
         pelota.setVelX( pelota.getVel() - 1 );
         pelota.setVelY( pelota.getVel() + 1 );
      }
      else if( py >= y1 + 31 && py <= y1 + paleta1.getAlto() - 32 )
         ///GOLPE EN EL MEDIO DE LA PALETA. Y=X
      {
```

```
            play_sample( hit, 200, 150, 1000, 0 );
            pelota.setVelX( pelota.getVel() );
            pelota.setVelY( pelota.getVel() );
          }
          dirx *= -1;
        }
        else///PALETA1 PIERDE
        {
          if( ( py > y1 + paleta1.getAlto() ) && ( px > paleta1.getAncho() + pelota.getAncho() / 2 ) && ( diry == -1 ) && ( px > 0 ) )
            ///PALETA1 REBOTE
          {
            play_sample( hit, 200, 150, 1000, 0 );
            diry = -1;
          }
          else if( ( py < y1 ) && ( px > paleta1.getAncho() + pelota.getAncho() / 2 ) && ( diry == 1 ) && ( px > 0 ) )
          {
            play_sample( hit, 200, 150, 1000, 0 );
            diry = 1;
          }

          if( px >= 0 &&jugadores.getEscudo1() > 0)
          {
            if( diry == -1 )
            {
               diry = -1;
            }
            dirx =  1 ;
            play_sample(escudo, 200, 150, 1000, 0);
            jugadores.setEscudo1( jugadores.getEscudo1() - 1 );//masked_blit( escudo1, buffer, 0, 0, px -1 , py , 10, 70);
          }
          if( px <= 0 )
          {
```

```
            funciones.setMensaje( 1 );
            play_sample( punto, 200, 150, 1000, 0 );
            jugadores.setEscudo2(jugadores.getEscudo2() + 1 );
            resetPelotaPaleta(pelota,pelota1,pelota2,pelota3,pelota4,pelota5, paleta1, 1, jugadores,
funciones, buffer, jugador1, jugador2, level_up, winner);
            return;
          }
        }
      }
      else if ( px >= ANCHO - paleta2.getAncho() - pelota.getAncho() / 2 )//GOLPE BORDE INTERIOR DE
PALETA2
      {
        if (( py + pelota.getAncho() / 2 ) >= y2 && ( py - pelota.getAncho() / 2 )<= y2 + paleta2.getAlto())
          ///GOLPE LIMITES LARGO DE PALETA
        {
          if( ( py + pelota.getAncho() / 2 ) >= y2 && py <= y2 + paleta2.getAlto() - 63 )
            ///GOLPE ESQUINA SUPERIOR. VELOCIDAD DE EJE Y>X
          {
            if( diry == 1 )
            {
              diry = -1;
            }
            play_sample( hit, 200, 150, 1000, 0 );
            pelota.setVelX( pelota.getVel() - 1 );
            pelota.setVelY( pelota.getVel() + 1 );
          }
          else if( py >= y2 + 62 && ( py - pelota.getAncho() / 2 ) <= y2 + paleta2.getAlto())
            ///GOLPE ESQUINA INFERIOR. VELOCIDAD DE EJE Y>X
          {
            if( diry == -1 )
            {
              diry = 1;
            }
            play_sample( hit, 200, 150, 1000, 0 );
```

```
            pelota.setVelX( pelota.getVel() - 1 );
            pelota.setVelY( pelota.getVel() + 1 );
         }
         else if( py >= y2 + 31  && py <= y2 + paleta1.getAlto() - 32 )///GOLPE MEDIO DE LA PALETA
         {
            play_sample( hit, 200, 150, 1000, 0 );
            pelota.setVelX( pelota.getVel() );
            pelota.setVelY( pelota.getVel() );
         }
         dirx *= -1;
      }
      else///PIERDE PALETA2
      {
         if(( ( py - pelota.getAncho() / 2 ) > y2 + paleta2.getAlto() ) && ( px > ANCHO - paleta2.getAncho() -
pelota.getAncho() / 2 ) && ( diry == -1 ) && ( px < ANCHO ) )
            ///PALETA2 REBOTE
         {
            play_sample( hit, 200, 150, 1000, 0 );
            diry = 1;
         }
         else if( ( ( py + pelota.getAncho() / 2 ) < y2 ) && ( px > ANCHO - paleta2.getAncho() -
pelota.getAncho() / 2 ) && ( diry == 1 ) && ( px < ANCHO ) )
         {


            play_sample( hit, 200, 150, 1000, 0 );
            diry = -1;
         }
         if( px < ANCHO && jugadores.getEscudo2() > 0)
         {
            if (diry ==  1)
            {
               diry =  -1 ;
            }
            else if(diry == -1)
```

```
            {
               diry =  1 ;
            }
            dirx = -1 ;
            play_sample(escudo, 200, 150, 1000, 0);
            jugadores.setEscudo2( jugadores.getEscudo2() - 1 );
         }

         if( px >= ANCHO )
         {
            ///PUNTO
            funciones.setMensaje( 1 );
            play_sample( punto, 200, 150, 1000, 0 );
            jugadores.setEscudo1(jugadores.getEscudo1() + 1 );
            resetPelotaPaleta( pelota,pelota1,pelota2,pelota3,pelota4,pelota5, paleta2, 2, jugadores,
funciones, buffer, jugador1, jugador2, level_up, winner);
            return;
         }
      }
   }
   else if (((diry < 0 && py <= 0 )) || (diry > 0  &&  py >= (ALTO-pelota.getAlto())) )//REBOTE PARED
   {
      play_sample( wall, 200, 150, 1000, 0 );
      diry *= -1;
   }
   ///MOVIMIENTOS CONSTANTES DE PELOTA
   pelota.setX( pelota.getX() + pelota.getVelX() * dirx );
   pelota.setDirX( dirx );
   pelota.setDirY( diry );
   pelota.setY( pelota.getY() + pelota.getVelY() * diry );
   pelota.setDirX( dirx );
   pelota.setDirY( diry );
  }
};
```

```
class Dibujar
{
public:
   void dibujar_mesa( BITMAP* buffer,Funcionalidad<int,bool> &funciones )
   ///DIBUJA LA LINEA DEL MEDIO
   {
      int i = 0;
      int j = 20;
      while( j <= ALTO )
      {
         rectfill( buffer, ANCHO/2, i, ANCHO/2, j, 0xFFFFFF );
         i = j + 10 ;
         j = i + 20 ;
      }
      ///rect(buffer,900,50,80,450,0xFFFFFF);
   }
   void tablero( BITMAP *Buffer, Jugador<int> &jugadores, Funcionalidad<int,bool> &funciones )//DIBUJAR
TABLERO
   {
      ///cajita para el score
      line( Buffer, ANCHO - 140, 0, ANCHO - 140, 80, 0xFFFFFF );
      line( Buffer, ANCHO - 142, 0, ANCHO - 142, 82, 0xFFFFFF );
      line( Buffer, ANCHO - 142, 82, ANCHO, 82, 0xFFFFFF );
      line( Buffer, ANCHO - 140, 80, ANCHO, 80, 0xFFFFFF );
      ///cajita para el score
      line( Buffer, 140, 0, 140, 80, 0xFFFFFF );
      line( Buffer, 142, 0, 142, 82, 0xFFFFFF );
      line( Buffer, 142, 82, 0, 82, 0xFFFFFF );
      line( Buffer, 140, 80, 0, 80, 0xFFFFFF );
      ///Puntaje para el Score
      textprintf_ex( Buffer, font, 10, 20, 0x00F0FBFF, - 1, "Partidos G: %d", funciones.getGanar1() );
      textprintf_ex( Buffer, font, 880, 20, 0x00F0FBFF, - 1, "Partidos G: %d", funciones.getGanar2() );
      textprintf_ex( Buffer, font, 10, 40, 0x00F0FBFF, - 1, "Puntos P1: %d", jugadores.getPuntaje1() );
```

```
textprintf_ex( Buffer, font, 880, 40, 0x00F0FBFF, - 1, "Puntos P2: %d", jugadores.getPuntaje2() );
textprintf_ex( Buffer, font, 500, 20, 0x00F0FBFF, - 1, "contandor total %d", funciones.getEscore() );

///Puntaje de escudos
if( jugadores.getEscudo1() > 0)
{
   textprintf_ex( Buffer, font, 10, 60, 0x00F0FBFF, - 1, "Escudo 1: %d", jugadores.getEscudo1() );
}
else
{
   textprintf_ex( Buffer, font, 10, 60, 0x00F0FBFF, - 1, "Escudo 1: 0" );
}
if( jugadores.getEscudo2() > 0)
{
   textprintf_ex( Buffer, font, 880, 60, 0x00F0FBFF, - 1, "Escudo 2: %d", jugadores.getEscudo2());
}
else
{
   textprintf_ex( Buffer, font, 880, 60, 0x00F0FBFF, - 1, "Escudo 2: 0" );
}
///Cambio de Nivel
switch(funciones.getNivel())
{
case 1 :
   textprintf_ex( Buffer, font, 800, 480,0x00F0FBFF, - 1, "LEVEL %d Muy Facil", funciones.getNivel());
   break ;
case 2 :
   textprintf_ex( Buffer, font, 800, 480, 0x00F0FBFF, - 1, "LEVEL %d Facil ", funciones.getNivel() );
   break ;
case 3 :
   textprintf_ex( Buffer, font, 800, 480, 0x00F0FBFF, - 1, "LEVEL %d Normal ", funciones.getNivel());
   break ;
case 4 :
   textprintf_ex( Buffer, font, 800, 480, 0x00F0FBFF, - 1, "LEVEL %d Dificil ", funciones.getNivel() );
```

```
        break;
      case 5 :
        textprintf_ex(Buffer, font, 800, 480, 0x00F0FBFF, -1, "LEVEL %d Muy Dificil ", funciones.getNivel());
        break;
      }
  }
  /// Dibujo de las diferentes pelotas
  void Dibujar_pelota(BITMAP *Buffer, Pelota<int,float>& pelota) //DIBUJAR PELOTA
  {
    circlefill ( Buffer, pelota.getX(), pelota.getY(), pelota.getAncho() / 2, 0xFFFFFF );
    circlefill ( Buffer, pelota.getX(), pelota.getY(), pelota.getAncho() / 6, 0x000000 );   /// me da el color
negro
  }
};
///INVOCACION DE LA FUNCIONALIDAD
Funcionalidad<int,bool> funcion;
class juego: public Dibujar, public Pelota <int,float>,public Funcionalidad<int,bool>, public Jugador<int>,
public Paleta<int>
{
///INVOCACION DE CLASES
  Paleta<int> paletaV1;
  Paleta<int> paletaV2;
  Pelota<int,float> pelota1;
  Pelota<int,float> pelota2;
  Pelota<int,float> pelota3;
  Pelota<int,float> pelota4;
  Pelota<int,float> pelota5;
  Jugador<int> jugadores;
  bitmaps *bits;
  samples *audio;
public :
  void IniciarYJugar( )
  {
    ///INICIALIZACIONES Y SETTERS DEL JUEGO
```

```
	allegro_init( );
	set_window_title( "Juego Ping-Pong" );
	install_keyboard( );
	install_timer( );
	install_sound( DIGI_AUTODETECT, MIDI_AUTODETECT, NULL);
	set_volume( 230, 200 );
	set_color_depth( 24 );
	set_gfx_mode( GFX_AUTODETECT_WINDOWED, ANCHO, ALTO, 0, 0 );
	install_int_ex( incremento, BPS_TO_TIMER( 70 ) );
	set_close_button_callback( cerrar_ventana );

	///INVOCACION DE ESTRUCTURAS
	bitmaps *bits;
	samples *audio;
	///DECLARACION DE SAMPLES DE AUDIO
	SAMPLE *audio1 = load_wav(
"C:/Users/kevin/OneDrive/Escritorio/Final_Programacion_2/PingPong/audio/Intro.wav" );
	SAMPLE *audio2 = load_wav(
"C:/Users/kevin/OneDrive/Escritorio/Final_Programacion_2/PingPong/audio/Hit.wav" );
	SAMPLE *audio3 = load_wav(
"C:/Users/kevin/OneDrive/Escritorio/Final_Programacion_2/PingPong/audio/Punto.wav" );
	SAMPLE *audio4 = load_wav(
"C:/Users/kevin/OneDrive/Escritorio/Final_Programacion_2/PingPong/audio/Saque.wav" );
	SAMPLE *audio5 = load_wav(
"C:/Users/kevin/OneDrive/Escritorio/Final_Programacion_2/PingPong/audio/Level-Up.wav" );
	SAMPLE *audio6 = load_wav(
"C:/Users/kevin/OneDrive/Escritorio/Final_Programacion_2/PingPong/audio/Winner.wav" );
	SAMPLE *audio7 = load_wav(
"C:/Users/kevin/OneDrive/Escritorio/Final_Programacion_2/PingPong/audio/WallHit.wav" );
	SAMPLE *audio8 = load_wav(
"C:/Users/kevin/OneDrive/Escritorio/Final_Programacion_2/PingPong/audio/escudo.wav");
	///DECLARACION DE BITMAPS
	BITMAP * buffer = create_bitmap( ANCHO, ALTO );
	BITMAP *buffer1 = create_bitmap( ANCHO, ALTO );
	BITMAP *buffer2 = create_bitmap( ANCHO, ALTO );
	BITMAP *buffer3 = create_bitmap( ANCHO, ALTO );
```

```
BITMAP *buffer4 = create_bitmap( ANCHO, ALTO );

BITMAP *buffer5 = create_bitmap( ANCHO, ALTO );


BITMAP *paleta_izq =
load_bitmap("C:/Users/kevin/OneDrive/Escritorio/Final_Programacion_2/PingPong/Imagenes/paleta1.bmp
", NULL);

BITMAP *paleta_der =
load_bitmap("C:/Users/kevin/OneDrive/Escritorio/Final_Programacion_2/PingPong/Imagenes/paleta2.bmp
", NULL);

BITMAP * inicio    =
load_bitmap("C:/Users/kevin/OneDrive/Escritorio/Final_Programacion_2/PingPong/Imagenes/portada.bm
p", NULL);

BITMAP * jugador1w =
load_bitmap("C:/Users/kevin/OneDrive/Escritorio/Final_Programacion_2/PingPong/Imagenes/Jugador_1_
Gana.bmp", NULL);

BITMAP * jugador2w =
load_bitmap("C:/Users/kevin/OneDrive/Escritorio/Final_Programacion_2/PingPong/Imagenes/Jugador_2_
Gana.bmp", NULL);


///ALMACENADO DE BITMAPS EN MEMORIA DINAMICA

bits = ( bitmaps * ) malloc( sizeof ( bitmaps ) * 13 );


bits[0].mapa_de_bits = buffer;

bits[1].mapa_de_bits = buffer1;

bits[2].mapa_de_bits = buffer2;

bits[3].mapa_de_bits = buffer3;

bits[4].mapa_de_bits = buffer4;

bits[5].mapa_de_bits = buffer5;

bits[6].mapa_de_bits = paleta_izq;

bits[7].mapa_de_bits = paleta_der;

bits[8].mapa_de_bits = inicio;

bits[9].mapa_de_bits = jugador1w;

bits[10].mapa_de_bits = jugador2w;


///ALMACENADO DE SAMPLES DE AUDIO EN MEMORIA DINAMICA

audio = ( samples * ) malloc( sizeof ( samples ) * 8 );

audio[0].muestra_de_audio = audio1;/// intro
```

```
audio[1].muestra_de_audio = audio2;/// hit
audio[2].muestra_de_audio = audio3;/// punto
audio[3].muestra_de_audio = audio4;/// saque
audio[4].muestra_de_audio = audio5;/// level-up
audio[5].muestra_de_audio = audio6;/// winner
audio[6].muestra_de_audio = audio7;/// wall-hit
audio[7].muestra_de_audio = audio8;/// escudo

///SETTEO E INICIALIZACION DE CLASES Y FUNCIONES
pelota1.setVelX( 5 );
pelota1.setVelY( 5 );
pelota1.setVel(5);
pelota1.setAlto( 15 );
pelota1.setAncho( 15 );
pelota1.setX(ANCHO / 2 );
pelota1.setY(ALTO / 2 );
pelota1.setIdentidad(1);
direccion_Pelota(pelota1,funcion );

pelota2.setVelX( 5);
pelota2.setVelY( 5);
pelota2.setVel( 4 );
pelota2.setAlto( 15 );
pelota2.setAncho( 15 );
pelota2.setX(ANCHO / 2 );
pelota2.setY((ALTO / 2) - 20  );
pelota2.setIdentidad(2);
direccion_Pelota(pelota2,funcion );

pelota3.setVelX( 4 );
pelota3.setVelY( 5);
pelota3.setVel( 5 );
pelota3.setAlto( 15 );
pelota3.setAncho( 15 );
```

```
pelota3.setX(ANCHO / 2);
pelota3.setY((ALTO / 2) + 20  );
pelota3.setIdentidad(3);
direccion_Pelota(pelota3,funcion);

pelota4.setVelX( 5);
pelota4.setVelY(4);
pelota4.setVel( 4);
pelota4.setAlto( 15);
pelota4.setAncho( 15);
pelota4.setX(ANCHO / 2);
pelota4.setY((ALTO / 2) + 40  );
pelota4.setIdentidad(4);
direccion_Pelota(pelota4,funcion);

pelota5.setVelX( 4);
pelota5.setVelY( 4);
pelota5.setVel( 5);
pelota5.setAlto( 15);
pelota5.setAncho( 15);
pelota5.setX(ANCHO / 2);
pelota5.setY((ALTO / 2) - 40  );
pelota5.setIdentidad(5);
direccion_Pelota(pelota5,funcion);

///Posicionamos las paletas y escudos
paletaV1.setX( 0);
paletaV1.setY( ALTO / 2 - 50 );
paletaV1.setTamano( 15, 94  );
paletaV2.setX( ANCHO - paletaV1.getAncho());
paletaV2.setY( ALTO / 2 - 50 );
paletaV2.setTamano( 15, 94  );
jugadores.setEscudo1( 5);
jugadores.setEscudo2( 5);
```

```
///FUNCION PORTADA
play_sample(audio[0].muestra_de_audio, 200, 150, 1000, 0 );
while( !funcion.getPortada() )
{
   if( key[KEY_ENTER] )
   {
      funcion.setPortada( true );
   }
   else if (key[KEY_ESC])
   {
      allegro_exit( );
   }
   blit( bits[8].mapa_de_bits, screen, 0, 0, 0, 0, ANCHO, ALTO );
}

///inicio del juego
///Funcion para salir de la ventana (ESC = SALIR)
while( ! funcion.getVentana() )
{
   if( key[KEY_ESC] )
      funcion.setVentana( true ); /// sirve para cerrar la ventana
   ///Color De los Bitmap De Fondo
   clear_to_color( bits[1].mapa_de_bits, 0x003800 );
   clear_to_color( bits[2].mapa_de_bits, 0x370000 );
   clear_to_color( bits[3].mapa_de_bits, 0x060739 );
   clear_to_color( bits[4].mapa_de_bits, 0x363300 );
   clear_to_color( bits[5].mapa_de_bits, 0x000000 );

   ///SET DE TIEMPO E INICIALIZACION DEL JUEGO
   while( funcion.getTiempo() > 0 )
   {
      if( funcion.getResultado() == 1 )
      {
```

```
   if( key[KEY_SPACE] )
   {
      play_sample( audio[3].muestra_de_audio, 200, 150, 1000, 0 );
      funcion.setResultado(0);
      funcion.setMensaje( 0);
   }
   funcion.setTiempo( 0);
   continue;
}
///DECLARACION DE VARIABLES, MARCA DE COORDENADAS Y POSICIONES
int posy1 = ALTO - paletaV1.getAlto();
int posy2 = ALTO - paletaV2.getAlto();
int y1 = paletaV1.getY();
int y2 = paletaV2.getY();
int movSpeed = 3;/// Velocidad de movimiento de las paletas
switch(funcion.getNivel())// velocidad de las paletas
{
case 1:
   movSpeed = 3 + funcion.getContadorP();
   break;
case 2:
   movSpeed = 4 + funcion.getContadorP();
   break;
case 3:
   movSpeed = 5 + funcion.getContadorP();
   break;
case 4:
   movSpeed = 6 + funcion.getContadorP();
   break;
case 5:
   movSpeed = 7 + funcion.getContadorP();
   break;
}
///MOVIMIENTOS
```

```
        if ( key[KEY_W] )
        {
          if( paletaV1.getY() >= 0 && y1 <= posy1 )
          {
            paletaV1.setY( paletaV1.getY() - movSpeed );
          }
          else
          {
            paletaV1.setY( ( paletaV1.getY() < 0 ) ?0 : posy1 );/// if ternario // lo anoto por que se me
olvida el nombre :D
          }
        }
        else if( key[KEY_S] )
        {
          if( paletaV1.getY() >= 0 && y1 <= posy1 )
          {
            paletaV1.setY( paletaV1.getY() + movSpeed );
          }
          else
          {
            paletaV1.setY( ( paletaV1.getY() < 0 ) ?0 : posy1 );
          }
        }
        if ( key[KEY_UP] )
        {
          if( paletaV2.getY() >= 0 && y2 <= posy2 )
          {
            paletaV2.setY( paletaV2.getY() - movSpeed );
          }
          else
          {
            paletaV2.setY( ( paletaV2.getY() < 0 ) ? 0 : posy2 );
          }
        }
```

```
        else if( key[KEY_DOWN] )
        {
          if( paletaV2.getY() >= 0 && y2 <= posy2 )
          {
             paletaV2.setY( paletaV2.getY() + movSpeed );
          }
          else
          {
             paletaV2.setY( ( paletaV2.getY() < 0 ) ?0 : posy2 );
          }
        }
        ///FUNCIONES VARIAS
        switch(funcion.getContadorP())
        {
        case 1 :
           moverPelota(pelota1, pelota1,pelota2,pelota3,pelota4,pelota5, paletaV1, paletaV2, jugadores,
funcion, bits[0].mapa_de_bits, bits[9].mapa_de_bits, bits[10].mapa_de_bits,
audio[1].muestra_de_audio,audio[2].muestra_de_audio,audio[4].muestra_de_audio,
audio[5].muestra_de_audio, audio[6].muestra_de_audio,audio[7].muestra_de_audio );
           break ;
        case 2:
           moverPelota( pelota1, pelota1,pelota2,pelota3,pelota4,pelota5,paletaV1, paletaV2, jugadores,
funcion, bits[0].mapa_de_bits, bits[9].mapa_de_bits, bits[10].mapa_de_bits,
audio[1].muestra_de_audio,audio[2].muestra_de_audio,audio[4].muestra_de_audio,
audio[5].muestra_de_audio, audio[6].muestra_de_audio,audio[7].muestra_de_audio );
           moverPelota( pelota2, pelota1,pelota2,pelota3,pelota4,pelota5, paletaV1, paletaV2, jugadores,
funcion, bits[0].mapa_de_bits, bits[9].mapa_de_bits, bits[10].mapa_de_bits,
audio[1].muestra_de_audio,audio[2].muestra_de_audio,audio[4].muestra_de_audio,
audio[5].muestra_de_audio, audio[6].muestra_de_audio,audio[7].muestra_de_audio );
           break;
        case 3:
           moverPelota( pelota1, pelota1,pelota2,pelota3,pelota4,pelota5,paletaV1, paletaV2, jugadores,
funcion, bits[0].mapa_de_bits, bits[9].mapa_de_bits, bits[10].mapa_de_bits,
audio[1].muestra_de_audio,audio[2].muestra_de_audio,audio[4].muestra_de_audio,
audio[5].muestra_de_audio, audio[6].muestra_de_audio,audio[7].muestra_de_audio );
           moverPelota( pelota2,pelota1,pelota2,pelota3,pelota4,pelota5, paletaV1, paletaV2, jugadores,
funcion, bits[0].mapa_de_bits, bits[9].mapa_de_bits, bits[10].mapa_de_bits,
audio[1].muestra_de_audio,audio[2].muestra_de_audio,audio[4].muestra_de_audio,
audio[5].muestra_de_audio, audio[6].muestra_de_audio,audio[7].muestra_de_audio );
```

```
            moverPelota( pelota3, pelota1,pelota2,pelota3,pelota4,pelota5,paletaV1, paletaV2, jugadores,
funcion, bits[0].mapa_de_bits, bits[9].mapa_de_bits, bits[10].mapa_de_bits,
audio[1].muestra_de_audio,audio[2].muestra_de_audio,audio[4].muestra_de_audio,
audio[5].muestra_de_audio, audio[6].muestra_de_audio,audio[7].muestra_de_audio );

            break;

        case 4:

            moverPelota( pelota1,pelota1,pelota2,pelota3,pelota4,pelota5, paletaV1, paletaV2, jugadores,
funcion, bits[0].mapa_de_bits, bits[9].mapa_de_bits, bits[10].mapa_de_bits,
audio[1].muestra_de_audio,audio[2].muestra_de_audio,audio[4].muestra_de_audio,
audio[5].muestra_de_audio, audio[6].muestra_de_audio,audio[7].muestra_de_audio );

            moverPelota( pelota2,pelota1,pelota2,pelota3,pelota4,pelota5, paletaV1, paletaV2, jugadores,
funcion, bits[0].mapa_de_bits, bits[9].mapa_de_bits, bits[10].mapa_de_bits,
audio[1].muestra_de_audio,audio[2].muestra_de_audio,audio[4].muestra_de_audio,
audio[5].muestra_de_audio, audio[6].muestra_de_audio,audio[7].muestra_de_audio );

            moverPelota( pelota3,pelota1,pelota2,pelota3,pelota4,pelota5, paletaV1, paletaV2, jugadores,
funcion, bits[0].mapa_de_bits, bits[9].mapa_de_bits, bits[10].mapa_de_bits,
audio[1].muestra_de_audio,audio[2].muestra_de_audio,audio[4].muestra_de_audio,
audio[5].muestra_de_audio, audio[6].muestra_de_audio,audio[7].muestra_de_audio );

            moverPelota( pelota4,pelota1,pelota2,pelota3,pelota4,pelota5, paletaV1, paletaV2, jugadores,
funcion, bits[0].mapa_de_bits, bits[9].mapa_de_bits, bits[10].mapa_de_bits,
audio[1].muestra_de_audio,audio[2].muestra_de_audio,audio[4].muestra_de_audio,
audio[5].muestra_de_audio, audio[6].muestra_de_audio,audio[7].muestra_de_audio );

            break;

        case 5:

            moverPelota( pelota1,pelota1,pelota2,pelota3,pelota4,pelota5, paletaV1, paletaV2, jugadores,
funcion, bits[0].mapa_de_bits, bits[9].mapa_de_bits, bits[10].mapa_de_bits,
audio[1].muestra_de_audio,audio[2].muestra_de_audio,audio[4].muestra_de_audio,
audio[5].muestra_de_audio, audio[6].muestra_de_audio,audio[7].muestra_de_audio );

            moverPelota( pelota2,pelota1,pelota2,pelota3,pelota4,pelota5, paletaV1, paletaV2, jugadores,
funcion, bits[0].mapa_de_bits, bits[9].mapa_de_bits, bits[10].mapa_de_bits,
audio[1].muestra_de_audio,audio[2].muestra_de_audio,audio[4].muestra_de_audio,
audio[5].muestra_de_audio, audio[6].muestra_de_audio,audio[7].muestra_de_audio );

            moverPelota( pelota3,pelota1,pelota2,pelota3,pelota4,pelota5, paletaV1, paletaV2, jugadores,
funcion, bits[0].mapa_de_bits, bits[9].mapa_de_bits, bits[10].mapa_de_bits,
audio[1].muestra_de_audio,audio[2].muestra_de_audio,audio[4].muestra_de_audio,
audio[5].muestra_de_audio, audio[6].muestra_de_audio,audio[7].muestra_de_audio );

            moverPelota( pelota4,pelota1,pelota2,pelota3,pelota4,pelota5, paletaV1, paletaV2, jugadores,
funcion, bits[0].mapa_de_bits, bits[9].mapa_de_bits, bits[10].mapa_de_bits,
audio[1].muestra_de_audio,audio[2].muestra_de_audio,audio[4].muestra_de_audio,
audio[5].muestra_de_audio, audio[6].muestra_de_audio,audio[7].muestra_de_audio );

            moverPelota( pelota5,pelota1,pelota2,pelota3,pelota4,pelota5, paletaV1, paletaV2, jugadores,
funcion, bits[0].mapa_de_bits, bits[9].mapa_de_bits, bits[10].mapa_de_bits,
audio[1].muestra_de_audio,audio[2].muestra_de_audio,audio[4].muestra_de_audio,
audio[5].muestra_de_audio, audio[6].muestra_de_audio,audio[7].muestra_de_audio );
```

```
        break;
    }
    ///TIEMPO - 1
    funcion.setTiempo(funcion.getTiempo() - 1);
}

switch(funcion.getContadorP())
{
case 1 :
    Dibujar_pelota( bits[0].mapa_de_bits, pelota1);
    break ;
case 2 :
    Dibujar_pelota( bits[0].mapa_de_bits, pelota1);
    Dibujar_pelota( bits[0].mapa_de_bits, pelota2);
    break;
case 3:
    Dibujar_pelota( bits[0].mapa_de_bits, pelota1);
    Dibujar_pelota( bits[0].mapa_de_bits, pelota2);
    Dibujar_pelota( bits[0].mapa_de_bits, pelota3);
    break;
case 4 :
    Dibujar_pelota( bits[0].mapa_de_bits, pelota1);
    Dibujar_pelota( bits[0].mapa_de_bits, pelota2);
    Dibujar_pelota( bits[0].mapa_de_bits, pelota3);
    Dibujar_pelota( bits[0].mapa_de_bits, pelota4);
    break;
case 5 :
    Dibujar_pelota( bits[0].mapa_de_bits, pelota1);
    Dibujar_pelota( bits[0].mapa_de_bits, pelota2);
    Dibujar_pelota( bits[0].mapa_de_bits, pelota3);
    Dibujar_pelota( bits[0].mapa_de_bits, pelota4);
    Dibujar_pelota( bits[0].mapa_de_bits, pelota5);
    break ;
}
```

```
        ///FUNCIONES Y SETTERS DE PALETAS, PELOTAS, MESA
        dibujar_mesa( bits[0].mapa_de_bits, funcion );
        ///TABLERO
        tablero( bits[0].mapa_de_bits, jugadores, funcion);
        /// dibujar las paletas
        masked_blit( bits[6].mapa_de_bits, bits[0].mapa_de_bits, 0, 0, paletaV1.getX(), paletaV1.getY(), 15,
94 );
        ///dibujar las segunda paleta
        masked_blit( bits[7].mapa_de_bits, bits[0].mapa_de_bits, 0, 0, paletaV2.getX(), paletaV2.getY(), 15,
94 );
        ///Dibuja los diferentes niveles
        switch(funcion.getNivel())
        {
        case 1 :
           bits[0].mapa_de_bits = bits[1].mapa_de_bits;
           blit( bits[0].mapa_de_bits, screen, 0, 0, 0, 0, ANCHO, ALTO );
           break ;
        case 2 :
           bits[0].mapa_de_bits = bits[2].mapa_de_bits;
           blit( bits[0].mapa_de_bits, screen, 0, 0, 0, 0, ANCHO, ALTO );
           break ;
        case 3 :
           bits[0].mapa_de_bits = bits[3].mapa_de_bits;
           blit( bits[0].mapa_de_bits, screen, 0, 0, 0, 0, ANCHO, ALTO );
           break ;
        case 4 :
           bits[0].mapa_de_bits = bits[4].mapa_de_bits;
           blit( bits[0].mapa_de_bits, screen, 0, 0, 0, 0, ANCHO, ALTO );
           break ;
        case 5 :
           bits[0].mapa_de_bits = bits[5].mapa_de_bits;
           blit( bits[0].mapa_de_bits, screen, 0, 0, 0, 0, ANCHO, ALTO );
           break ;
        }
```

```
        }
        ///TIEMPO INTERLOCUCION DE CPU (20)
        rest( 25 );
        ///LIMPIEZA DE BITMAPS
        clear_bitmap( bits[0].mapa_de_bits );

        ///DESTRUCTORES DE BITMAPS
        destroy_bitmap( bits[0].mapa_de_bits );
        destroy_bitmap( bits[1].mapa_de_bits );
        destroy_bitmap( bits[2].mapa_de_bits );
        destroy_bitmap( bits[3].mapa_de_bits );
        destroy_bitmap( bits[4].mapa_de_bits );
        destroy_bitmap( bits[5].mapa_de_bits );
        destroy_bitmap( bits[6].mapa_de_bits );
        destroy_bitmap( bits[7].mapa_de_bits );
        destroy_bitmap( bits[8].mapa_de_bits );
        destroy_bitmap( bits[9].mapa_de_bits );
        destroy_bitmap( bits[10].mapa_de_bits );

        free(bits);
        free(audio);
    }
};

void cerrar_portada()
{
   funcion.setPortada( true );
}
void cerrar_ventana()
{
   funcion.setVentana( true );
}
void incremento()
{
```

```
   funcion.setTiempo( funcion.getTiempo() + 1 );
}

int main()
{
   juego init ;
   init.IniciarYJugar();
   allegro_exit( );
   return 0;
}
END_OF_MAIN( )
```