Trabajo Informática

Curso 2018/19- GRUPO A109

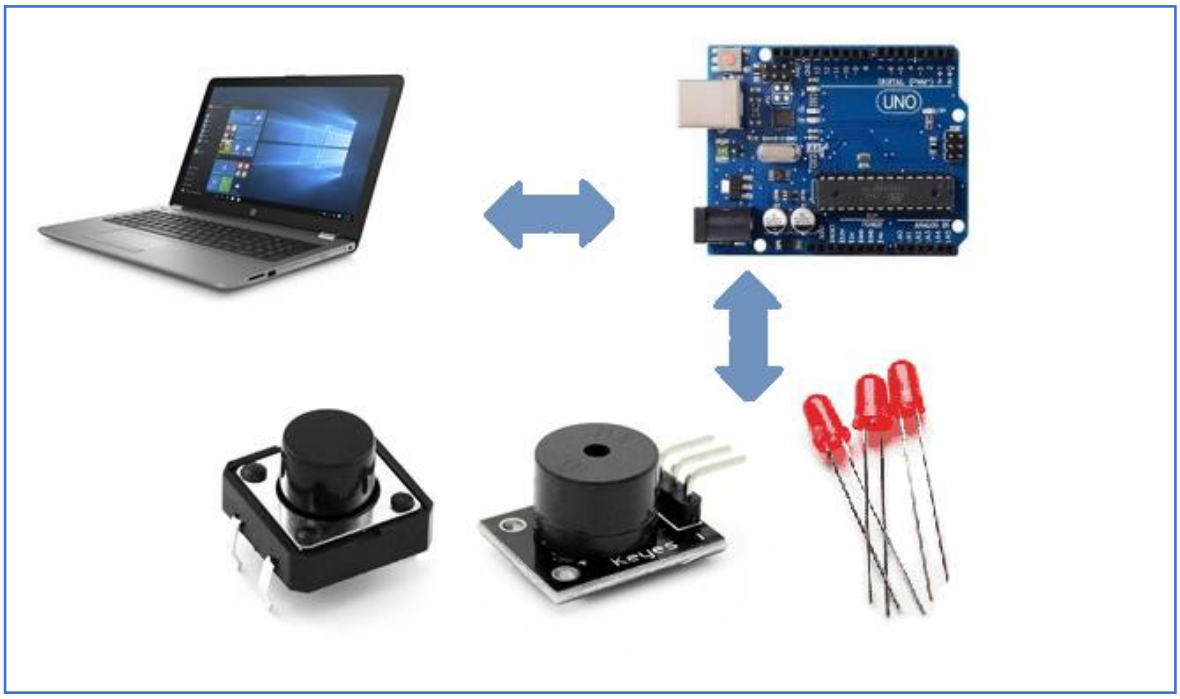
DATOS DE LOS MIEMBROS DEL GRUPO

NOMBRE	APELLIDOS	NºMAT	CORREO ELECTRÓNICO
Pablo Jose	Mendez Camino	54745	Pablo.mendez.camino@alumnos.upm.es
Victor	Rincón Calvo	54823	Victor.rincon.calvo@alumnos.upm.es
Rubén	Pleiter García	54808	r.pleiter@alumnos.upm.es

TITULO Y RESUMEN

JUEGO DE MEMORIA

Es un juego en el cual hay varios leds de diferentes colores con un pulsador cada uno y un sonido diferente. El juego consiste en que se enciende un led y tienes que pulsar el pulsador de ese led. Después se enciende ese led y a continuación otro aleatoriamente y debes pulsar el pulsador del primero y después del segundo. Y así sucesivamente. El juego acaba cuando te equivocas y pulsas un led que no se ha encendido. A continuación, aparece la secuencia de derrota y el usuario ve la puntuación obtenida en la pantalla. El juego debe comunicarse con un ordenador para el envío de la secuencia y para recibir la puntuación obtenida. Cuenta con 3 niveles de dificultad.



REQUISITOS FUNCIONALES

1º- La aplicación dispone de un menú en pantalla para elegir el modo de juego según su dificultad.

2º- Iluminación de LEDS y lectura de pulsadores.

3º- Reproducción de sonidos a través de un buzzer.

4º- La puntuación obtenida se compara con el record y aparece en el ordenador.

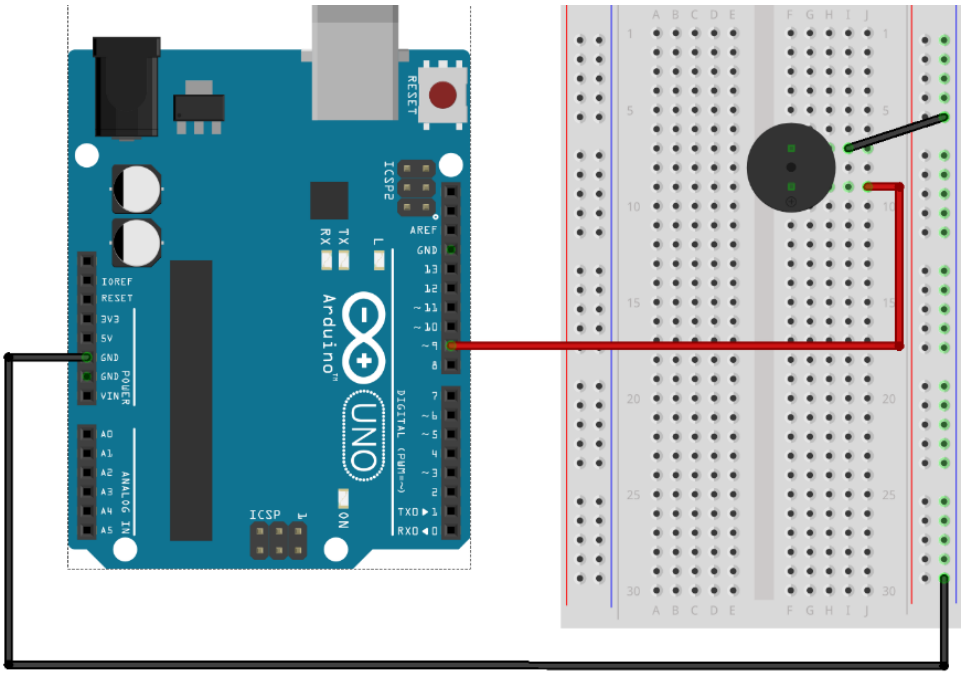
HARDWARE-FUNDAMENTOS TÉCNICOS

BUZZER PASIVO

Un buzzer pasivo es un dispositivo que permite convertir una señal eléctrica en una onda de sonido. Estos dispositivos no disponen de electrónica interna, por lo que tenemos que proporcionar una señal eléctrica para conseguir el sonido deseado. En oposición, los buzzer activos disponen de un oscilador interno, por lo que únicamente tenemos que alimentar el dispositivo para que se produzca el sonido.

Pese a tener la complejidad de proporcionar y controlar nosotros la señal eléctrica, los buzzer pasivos tienen la ventaja de que podemos variar el tono emitido modificando la señal que aplicamos al altavoz, lo que nos permite generar melodías.

Técnicamente los buzzer son transductores electroacústicos, es decir, dispositivos que convierten señales eléctricas en sonido. La diferencia entre ambos es el fenómeno en el que basan su funcionamiento. Los materiales piezoeléctricos tienen la propiedad especial de variar su volumen al ser atravesados por corrientes eléctricas. Este aprovecha este fenómeno para hacer vibrar una membrana al atravesar el material piezoeléctrico con una señal eléctrica.

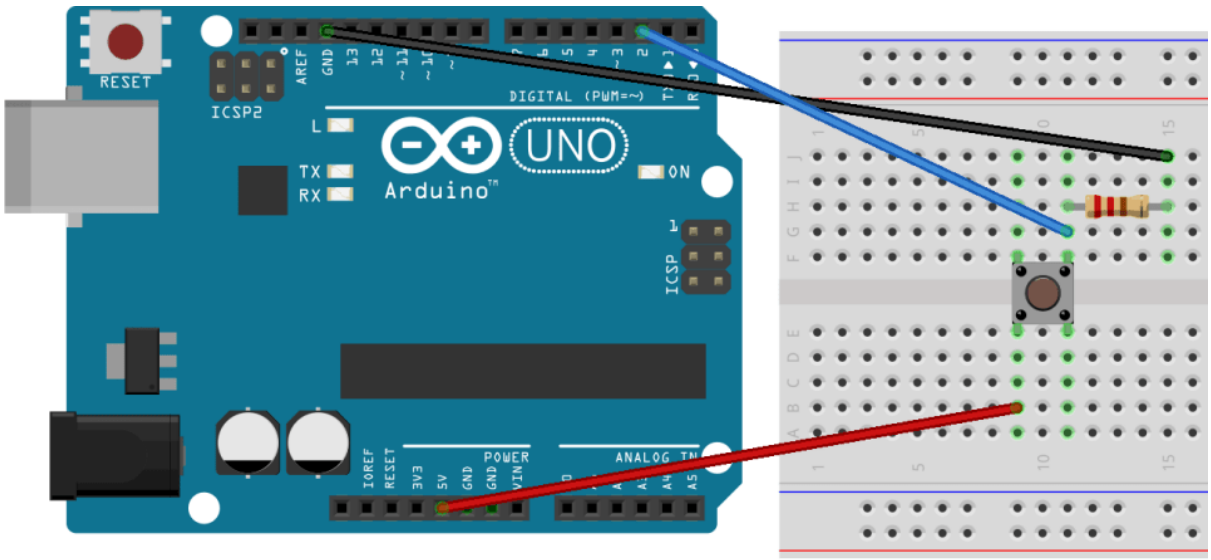


Fuente: https://www.luisllamas.es/reproducir-sonidos-arduino-buzzer-pasivo-altavoz/

PULSADOR

Un pulsador es un componente mecánico que permite cerrar establecer un cortocircuito entre dos terminales al accionar un pulsador. Funciona como un interruptor pero este se mantiene cerrado exclusivamente mientras se encuentra accionado, por lo que al soltarlo regresa de nuevo a su estado de reposo abriendo de nuevo el circuito.

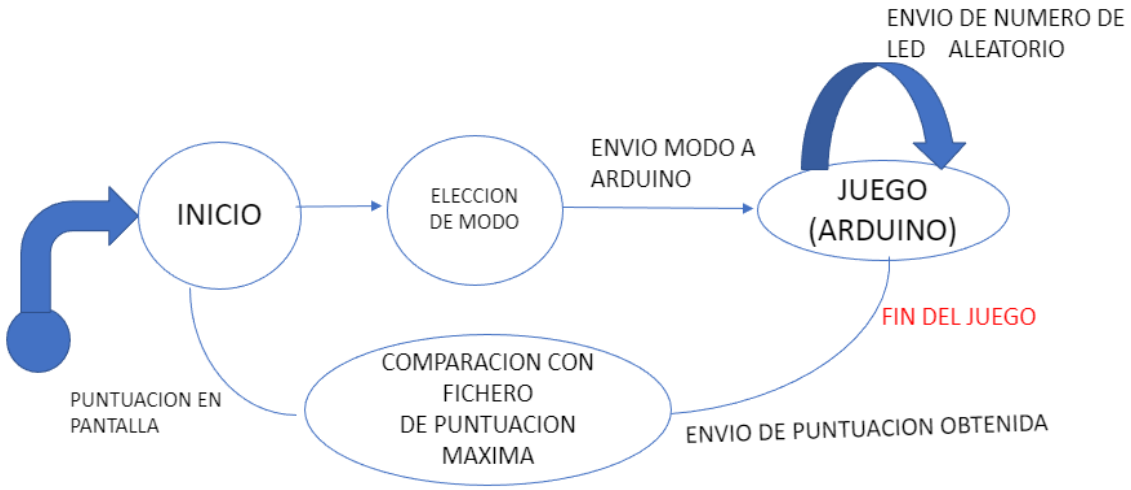
Para evitar que en uno de los dos estados, el pin de Arduino esté "flotando" eléctricamente, existen dos configuraciones de conexión en las que se incorporan resistencias de pull-up y puldown que por un lado limitan la corriente cuando el interruptor se cierra y por otros aseguran que el pin de Arduino cuando el interruptor se encuentre abierto, esté conectado a Vcc (resistencia de pull-up) o a GND (resistencia de pull-down). Cuando se utilizan la configuración pull-up, el pin de Arduino detecta un "1" lógico en reposo y un "0" lógico al ser pulsado. Si la configuración en es pull-down sucede lo contrario "0" en reposo, "1" pulsado.



Fuente: https://www.luisllamas.es/leer-un-pulsador-con-arduino/

Diseño del Software

Al tratarse de dos plataformas hardware intercomunicadas por un puerto serie existen dos aplicaciones a desarrollar. La primera es la aplicación de envío de datos, la elección de menú y se encarga de la puntuación máxima. Es una aplicación desarrollada en C y su funcionamiento se describe con la máquina de estados siguiente:




```
/El usuario elige entre los tres modos de juego y su eleccion es enviada a
Arduino y empieza a mandar numeros para establecer la secuencia.

do
{
opcion=menu();
switch (opcion)
{
      case '1':
      {
      eleccion=1;
      printf("Iniciando modo facil\n");
sendData = '1';
writeSerialPort(arduino,&sendData, sizeof(char));
////////////////////////////////////////
for(i=0;i<NIVEL_MAX;i++)
      {
      while(readResult==0)
{
       readResult = readSerialPort(arduino,incomingData, MAX_DATA_LENGTH);
}
      n=rand() % 4;
      if(n==0)
      {
      sendData='2';
      writeSerialPort(arduino,&sendData, sizeof(char));
      }
      if(n==1)
      {
```

```
        sendData='3';
        writeSerialPort(arduino,&sendData, sizeof(char));
        }
        if(n==2)
        {
        sendData='4';
        writeSerialPort(arduino,&sendData, sizeof(char));
        }
        if(n==3)
        {
        sendData='5';
        writeSerialPort(arduino,&sendData, sizeof(char));
        }
        readResult=0;

        }

////////////////////////////////////////
        }
              break;
        case '2':
              {
              printf("Iniciando modo medio\n");
                    eleccion=2;
                          sendData = '2';
                   writeSerialPort(arduino,&sendData, sizeof(char));
                  ////////////////////////////////////////////
for(i=0;i<NIVEL_MAX;i++)
        {
        while(readResult==0)
{
         readResult = readSerialPort(arduino,incomingData, MAX_DATA_LENGTH);
}
        n=rand() % 4;
        if(n==0)
        {
        sendData='2';
        writeSerialPort(arduino,&sendData, sizeof(char));
        }
        if(n==1)
        {
        sendData='3';
        writeSerialPort(arduino,&sendData, sizeof(char));
        }
        if(n==2)
        {
        sendData='4';
        writeSerialPort(arduino,&sendData, sizeof(char));
        }
        if(n==3)
        {
        sendData='5';
        writeSerialPort(arduino,&sendData, sizeof(char));
        }
        readResult=0;

        }
////////////////////////////////////////

              }
            break;
```

```
      case '3':
      {
      printf("Iniciando modo extremo\n");
            eleccion=3;
                  sendData = '3';
writeSerialPort(arduino,&sendData, sizeof(char));
//////////////////////////////////////////
for(i=0;i<NIVEL_MAX;i++)
      {
      while(readResult==0)
{
       readResult = readSerialPort(arduino,incomingData, MAX_DATA_LENGTH);
}
      n=rand() % 4;
      if(n==0)
      {
      sendData='2';
      writeSerialPort(arduino,&sendData, sizeof(char));
      }
      if(n==1)
      {
      sendData='3';
      writeSerialPort(arduino,&sendData, sizeof(char));
      }
      if(n==2)
      {
      sendData='4';
      writeSerialPort(arduino,&sendData, sizeof(char));
      }
      if(n==3)
      {
      sendData='5';
      writeSerialPort(arduino,&sendData, sizeof(char));
      }
      readResult=0;


      }

//////////////////////////////////////////


      }break;
      case '4':break;
    default: printf("Opcion incorrecta, elija de nuevo\n");
      }
      if(opcion=='3' || opcion=='2' || opcion=='1' )
      {
```

```
/Cada vez que haces una ronda le manda un dato (‘2’ si ha completado
correctamente la ronda y ‘8’ si ha fallado y por lo tanto acaba el juego).


printf("Comienza el juego\n");


            while(readResult==0 && contador==0  )
{
      readResult = readSerialPort(arduino,incomingData, MAX_DATA_LENGTH);
      if(incomingData[0]=='2')
```

```
        puntos++;
        if(incomingData[0]=='8')
        {
              contador++;
        }
        readResult=0;
}

printf("Enhorabuena has conseguido %d puntos\n",puntos);
comparar_datos_guardar(puntos,eleccion);
}
        }while(opcion!='4');



if (!isConnected(arduino))
 printf ("Se ha perdido la conexión con Arduino\n");
}
```

La función guardar_puntuacionm guarda la puntuacion obtenida y la guarda en un fichero. La funcion comparar_datos_guardar es la encargada de comparar la puntuación obtenida con las puntuaciones máximas a través de la comparación de los dos ficheros.

```
void guardar_puntuacionm(int n,int n1,int n2)

{

        int i;
        FILE *salida;
        salida=fopen("PuntuacionMaxima.txt","wt");
        if (salida==NULL)
        printf("NO SE HA PODIDO GUARDAR LOS DATOS\n");
        else
        {
        fprintf(salida,"%d %d %d",n,n1,n2);
        }
        fclose(salida);

        }



void comparar_datos_guardar(int PuntuacionM,int ab)
{
    int n,n1,n2;
      FILE *entrada;
      entrada=fopen("PuntuacionMaxima.txt","rt");
      if (entrada==NULL)
      {
      printf("No existen registros previos, creando archivo nuevo\n");
      guardar_puntuacionm(0,0,0);
    comparar_datos_guardar(PuntuacionM,ab);
      }
      else

      {
      fscanf(entrada,"%d",&n);
```

```
      fscanf(entrada,"%d",&n1);
      fscanf(entrada,"%d",&n2);
      switch(ab)
      {
            case 1:
      if(PuntuacionM>=n)
      {
      n=PuntuacionM;
    printf("Nueva puntuacion maxima en nivel FACIL\n");
      guardar_puntuacionm(n,n1,n2);
    }
    else
    printf("Casi superas el record, sigue intentandolo \n");
    break;
    case 2:if(PuntuacionM>=n1)
      {
      n1=PuntuacionM;
    printf("Nueva puntuacion maxima en nivel MEDIO\n");
      guardar_puntuacionm(n,n1,n2);
    }
    else
    printf("Casi superas el record, sigue intentandolo \n");
    break;
    case 3:if(PuntuacionM>=n2)
      {
      n2=PuntuacionM;
    printf("Nueva puntuacion maxima en nivel EXTREMO\n");
      guardar_puntuacionm(n,n1,n2);
    }
    else
    printf("Casi superas el record, sigue intentandolo \n");
    break;
      }
      fclose(entrada);

}
      }
```

Función de menu de opciones

```
char menu(void)
{
      char opcion;
printf("Elige una dificultad\n");
printf("1 - facil \n");
printf("2 - media \n");
printf("3 - dificil\n");
printf("4 - salir\n");
opcion=getch();
      return opcion;
}
```

ARDUINO

El juego está desarrollado con Arduino y su funcionamiento se describe con la máquina de estados siguiente:


```
//DEFINICION DE VARIABLES

#define PULSADOR1  A3
#define PULSADOR2  A2
#define PULSADOR3  A1
#define PULSADOR4  A0

#define NIVEL_MAX 100

#define LED1  2
#define LED2 3
#define LED3  4
#define LED4  5

#define BUZZER 7

void generaSecuencia(int,int []);
int secuenciaCorrecta(int,int);
void muestraSecuencia(int,int,int []);
int leeSecuencia(int *,int,int,int []);
void melodiaError(void);
void secuenciaError(void);
void juego(char,int []);
void juego1(int,int []);

void setup()
{
   Serial.begin(9600);
   pinMode(PULSADOR1, INPUT);
   pinMode(PULSADOR2, INPUT);
   pinMode(PULSADOR3, INPUT);
   pinMode(PULSADOR4, INPUT);
   pinMode(LED1, OUTPUT);
   pinMode(LED2, OUTPUT);
   pinMode(LED3, OUTPUT);
   pinMode(LED4, OUTPUT);
   pinMode(BUZZER, OUTPUT);
   digitalWrite(LED1, LOW);
```

```
   digitalWrite(LED2, LOW);
   digitalWrite(LED3, LOW);
   digitalWrite(LED4, LOW);
}
```

```
void loop()
{
  char dato;
  int v=99;
  int secuencia[NIVEL_MAX];
if (Serial.available() > 0)  //recibe del ordenador el char correspondiente
al modo
{

  dato=Serial.read();
  generaSecuencia(v,secuencia);
  juego(dato,secuencia);
}
}
```

```
void muestraSecuencia(int gameover,int nivelActual,int secuencia[]) //se
encienden leds con un sonido distintivo para cada uno a partir del vector
secuencia
{
   digitalWrite(LED1, LOW);
   digitalWrite(LED2, LOW);
   digitalWrite(LED3, LOW);
   digitalWrite(LED4, LOW);

   if(gameover==0)
    {
   for(int i = 0; i < nivelActual; i++)
   {
      if(secuencia[i] == LED1)
      {
         tone(BUZZER, 200);
         delay(200);
         noTone(BUZZER);
      }

      if(secuencia[i] == LED2)
      {
         tone(BUZZER, 300);
         delay(200);
         noTone(BUZZER);
      }

      if(secuencia[i] == LED3)
      {
         tone(BUZZER, 400);
         delay(200);
         noTone(BUZZER);
      }

     if(secuencia[i] == LED4)
     {
        tone(BUZZER, 500);
        delay(200);
        noTone(BUZZER);
     }
```

```
      digitalWrite(secuencia[i], HIGH);
      delay(250);
      digitalWrite(secuencia[i], LOW);
      delay(150);
   }
}
 }
```

```
int leeSecuencia(int *go,int nivelActual,int a,int secuencia[])     //
Compara el boton pulsado con el correcto y hace que se encienda el led del
boton correspondiente
{
   int flag;
   int secuenciaUsuario[NIVEL_MAX];
   int i,j;
   int  gameover;
    gameover=*go;
   if(gameover==0)
   {
   for(i = 0; i < nivelActual && gameover==0; i++)
   {
     flag = 0;
      while(flag == 0)
      {
       if(digitalRead(PULSADOR4) == LOW)
       {
            digitalWrite(LED4, HIGH);
            tone(BUZZER, 500);
            delay(300);
            noTone(BUZZER);
            secuenciaUsuario[i] = LED4;
            flag = 1;
            delay(100);
            if(secuenciaUsuario[i] != secuencia[i])
            {
              gameover=1;
             secuenciaError();

            }
            digitalWrite(LED4, LOW);
        }

        if(digitalRead(PULSADOR3) == LOW)
        {
           digitalWrite(LED3, HIGH);
           tone(BUZZER, 400);
           delay(300);
           noTone(BUZZER);
           secuenciaUsuario[i] = LED3;
           flag = 1;
           delay(100);
           if(secuenciaUsuario[i] != secuencia[i])
           {
             gameover=1;
             secuenciaError();
           }
           digitalWrite(LED3, LOW);
        }

        if(digitalRead(PULSADOR2) == LOW)
```

```
        {
           digitalWrite(LED2, HIGH);
           tone(BUZZER, 300);
           delay(300);
           noTone(BUZZER);
           secuenciaUsuario[i] = LED2;
           flag = 1;
           delay(100);
           if(secuenciaUsuario[i] != secuencia[i])
           {
             gameover=1;
             secuenciaError();

           }
           digitalWrite(LED2, LOW);
        }

        if(digitalRead(PULSADOR1) == LOW)
        {
           digitalWrite(LED1, HIGH);
           tone(BUZZER, 200);
           delay(300);
           noTone(BUZZER);
           secuenciaUsuario[i] = LED1;
           flag = 1;
           delay(100);
           if(secuenciaUsuario[i] != secuencia[i])
           {
             gameover=1;
              secuenciaError();

           }
           digitalWrite(LED1, LOW);
        }
      }
    }
    nivelActual=secuenciaCorrecta(nivelActual,a);
}
 *go=gameover;
return nivelActual;
}
```

```
void generaSecuencia(int a,int secuencia[]) // pide y recibe del la
secuencia
{
  int i,j,v,n;
  char dato;
 j=0;
 Serial.println('1');
for(i=0;i<NIVEL_MAX-a;i++)
{
  dato=0;
  while(dato==0)
  {
  dato=Serial.read();
  v=dato-48;
  if(v==2 || v==3 || v==4 || v==5)
  secuencia[i]=v;
  }
```

```
   Serial.println('1');
   delay(10);
}
}
```

```
void melodiaError(void)
{
  int duracionNotas[] = {4, 8, 8, 4, 4, 4, 4, 4};
  int melodia[ ] = {262, 196, 196, 220, 196, 196, 247, 262};
  int i,duracionNota,pausaEntreNotas;

   for(i = 0; i < 8; i++)
   {
      duracionNota = 1000/duracionNotas[i];
      tone(BUZZER,melodia[i],duracionNotas);
      pausaEntreNotas = duracionNota * 1.30;
      delay(pausaEntreNotas);
      noTone(BUZZER);

   }

}
```

```
void secuenciaError(void)
{
   digitalWrite(LED1, HIGH);
   digitalWrite(LED2, HIGH);
   digitalWrite(LED3, HIGH);
   digitalWrite(LED4, HIGH);
   delay(250);
   digitalWrite(LED1, LOW);
   digitalWrite(LED2, LOW);
   digitalWrite(LED3, LOW);
   digitalWrite(LED4, LOW);
   delay(250);
   melodiaError();
   Serial.println('8');
   Serial.println('8');
}
```

```
int secuenciaCorrecta(int nivelActual,int a)
{
   if(nivelActual < NIVEL_MAX)
   nivelActual=nivelActual+a;
   Serial.println('2');
   delay(150);
return nivelActual;
}
```

```
void juego(char a,int v[]) // modo
{
 int n=0;
  if(a=='1')
  {
    n=1;
    juego1(n,v);
  }

  if(a=='2')
  {
    n=2;
    juego1(n,v);
  }

  if(a=='3')
  {
    n=5;
   juego1(n,v);

}

}
```

```
void juego1(int a,int s[])
{
int gameover=0;
int nivelActual=1;
int v=0;
  while(gameover==0)
    {
   if(nivelActual == 1)
   {
    generaSecuencia(v,s);
    muestraSecuencia(gameover,nivelActual,s);
    nivelActual=leeSecuencia(&gameover,nivelActual,a,s);
   }

   if(nivelActual != 1)
   {
      muestraSecuencia(gameover,nivelActual,s);
      nivelActual=leeSecuencia(&gameover,nivelActual,a,s);
   }
}
}
```

FOTOS EN FUNCIONAMIENTO

```
Conectado con Arduino en el puerto \\.\COM4
Bienvenidos al fantastico Memory Game
Elige una dificultad
1 - facil
2 - media
3 - dificil
4 - salir
Iniciando modo facil
Comienza el juego
Enhorabuena has conseguido 4 puntos
No existen registros previos, creando archivo nuevo
Nueva puntuacion maxima en nivel FACIL
Elige una dificultad
1 - facil
2 - media
3 - dificil
4 - salir
```

Maximizar