Informe - Parcial I
Informática II

José Alejandro Moreno Mesa - 1001369765
Miguel Ángel Restrepo Rueda - 1001017183

Despartamento de Ingeniería Electrónica y Telecomunicaciones
Universidad de Antioquia
Medellín
Abrl de 2021

Índice

1. Introducción

Se presentará por medio de distintas divisiones la propuesta de solución al parcial 1, mediante el uso de TINKERCAD y el entorno de arduino, haciendo uso de todos los conceptos aprendidos a través de las sesiones teóricas y de laboratorio, tales como memoria dinámica, punteros, arreglos, funciones, uso de Arduino, entre muchos otros. Por lo tanto, se mostrará detenidamente el proceso para el resultado hallado, de este modo evidenciando el desarrollo que tuvo tanto el diseño del circuito como el código hecho. Finalmente, se presentarán los problemas encontrados a través del camino y sus respectivas soluciones, además del modelo de trabajo con las tareas planteadas y el plan de trabajo usado.

2. Análisis del problema

Primero se lee detenidamente los enunciados del parcial, con sus requisitos y especificaciones, haciendo ciertas anotaciones para tener en cuenta en un futuro.

En primera instancia se debe diseñar el circuito con el circuito integrado 74HC595 y los 64 LEDS de manera que se pueda trabajar con el arduino. En un inicio se pensó en utilizar sólo 2 integrados, uno que permitiera controlar las filas y otro las columnas, pero esto implicaba el uso de transistores, lo cual no estaba permitido por lo que se busco otra alternativa, la cual es utilizar 8 circuitos integrados, para que cada integrado controle 8 LEDS de manera independiente y así con los 8 integrados se puedan controlar los 64 LEDS de manera independiente, facilitando así la programación de cada LED al poder trabajar con cada uno de manera individual. Se quiso multiplexar las tierras de cada LED(multiplexar los cátodos) para reducir la cantidad de cables ya que estos solo irán a tierra en la protoboard, también, se define un color para cada cable según donde va conectado para tener un mayor orden y claridad a la hora de hacer conexiones. Una vez montado el circuito se comparten los puertos seriales entre todos los integrados para poder mandar las mismas instrucciones a través de todas las filas de LEDS a su vez se comparten los relojes RCLK y SRCLK para mandar controlarlos de igual manera para todas las filas de la matriz, además compartiendo todos estos elementos hace que se usen menos puertos digitales del Arduino, utilizando solo 3 puertos de los 7 que estaban permitidos.

Una vez montado el circuito, conectando debidamente todos los circuitos integrados con el arduino y con los LEDS, se calcula la resistencia en la cual los LEDS alumbran a una intensidad pertinente y no se queman los circuitos integrados, cabe resaltar que cada LED tiene que tener su debida resistencia, por lo que en total habrán 64 resistencias, todas del mismo valor. Ahora, viendo que todo esté conectado debidamente, se sigue con la parte de codificación, primero definiendo un menú claro para el usuario, en el que pueda acceder fácilmente a todas las funciones, presentando 4 opciones:

0. Detener la función que está siendo ejecutada.
1.Función verificación(Que se prendan todos los LEDS).
2.Función imagen(Permitirle al usuario ingresar un patrón para mostrar).
3.Función Publik(Permitirle al usuario ingresar la cantidad de patrones que desee con el tiempo de visualización de cada patrón que desee).

También dentro de cada función se habilita la posibilidad de volver al menú para ejecutar otra opción.
Luego, para la función verificación es necesario prender toda la matriz de LEDS por lo que se manda un 1 a cada LED para que vayan prendiendo de manera secuencial pero guardando los datos de los LEDS anteriores, para esto es necesario recorrer toda la matriz de LEDS mediante un ciclo saltando entre filas

mientras se prenden todos los LEDS. Una vez se termina de recorrer toda la matriz y se prenden los LEDS, estos se dejan prendidos hasta que el usuario ejecute la opción 0 del menú.

Después, para la función de imagen es necesario pedirle al usuario que vaya ingresando en orden descendente(De la primera fila hasta la última) que LEDS quiere prender y cuáles no, para prender un LED pone un 1 y para apagarlo un 0, a su vez tiene que meter toda la fila especificando todos los LEDS que quiere prender o apagar, luego se salta a la siguiente fila y se hace el mismo proceso, para esto también se usa un ciclo que vaya recorriendo la matriz pero que mande la información que ingresa el usuario, por lo que si se ingresa un 1 entre a una función que prenda el debido LED y si se ingresa un 0 entra a una función que deja el LED apagado, los punteros, arreglos y funciones facilitan este código. Toda la información se almacena y una vez acabada de ingresar todos los valores de cada fila se prenden los LEDS indicados.

Por último para la función Publik se puede reutilizar parte del código para la función imagen, haciendo algunos cambios, para que el usuario pueda meter la cantidad de patrones que desee(Ya que en la función imagen solo se ingresaba un patrón a mostrar) y que indique la cantidad de tiempo que desea entre cada patrón, para esto se agrega otro ciclo que haga lo mismo que la función imagen pero la cantidad de veces definida por el usuario, mostrando todos los patrones que el usuario desee y con el tiempo de retardo que el usuario ingrese se toma el delay que a tener en cuenta. Cabe aclarar que la simulación en Tinkercad no ocurre del todo en tiempo real por lo que el usuario se tiene que fijar en el reloj ubicado en la esquina superior derecha para evidenciar cada cuanto se cambia el patrón ingresado. Aquí es necesario el uso de la memoria dinámica, ya que no se sabe cuantos patrones desea el usuario por lo que no se puede reservar una cantidad fija de memoria. Y así como para la función imagen se usan punteros y arreglos para facilitar el trabajo.

3. Esquema de tareas

Para abordar este proyecto, se definieron las siguientes etapas y tareas que se consideraron necesarias:

1.Pruebas del integrado 74HC595:
Se estudió el funcionamiento de este integrado, sus componentes y pines, sus relojes y sus características en general.

2.Diseño inicial del circuito:
Se diseñó un posible circuito donde se implementaron ocho integrados para controlar una matriz de 64 LED (uno para cada fila). En una primera etapa se implementaron ocho transistores para tener un circuito donde solo se requerían dos integrados 74HC595, pero luego se cambió por el modelo ya explicado.

3.Implementación en TinkerCad:
Se realizó el montaje del circuito en la plataforma Tinkercad, donde posteriormente se realizaron pruebas básicas para comprobar el correcto funcionamiento de las conexiones hechas en el circuito.

4.Diseño de la función Verificación:
Se pensó cuál de las posibles ideas que se tenían para la función de verificación, para posteriormente elegir una que implementa la conversión de números decimales a binario para el control de los led. Luego, se realizaron la codificación y pruebas del código que implementa la función.

5. Estudio de los punteros dobles y triples:
De acuerdo con las ideas que se eligieron para las funciones requeridas, se vio la necesidad de estudiar, con la ayuda de múltiples foros y explicaciones adicionales de los profesores, la sintaxis y funcionalidad de los punteros dobles y triples, ya que se éstos se acomodan perfectamente a la ideación realizada para las funciones.

6.Diseño de la función Imagen:
Trabajando sobre una copia del circuito implementado en Tinkercad, se realizó la misma lluvia de ideas y selección que se hizo en la tarea anterior para desarrollar la función Imagen. Sin embargo, en este paso se vio la necesidad de crear múltiples sub funciones que permitieran el correcto funcionamiento de la función de manera independiente. Finalmente, se realizaron múltiples pruebas de la función.

7. Implementación de funciones adicionales (detener y regresar):
Al comienzo de la fusión de las funciones ya desarrolladas, se precisó el uso de dos funcionalidades adicionales que harían la experiencia de usuario más agradable. La primera se nombró “detener” y le permite al usuario apagar la matriz para que sea más cómodo para éste percibir la transición entre patrones. Asi-

mismo, se implementó la funcionalidad de “regresar” que le da al usuario la capacidad de decidir cuándo quiere dejar el trabajo que hizo en alguna de las funciones para volver al menú principal.

8. Fusión de las funciones en el código final:
Se decidió usar el IDE de Arduino para realizar la fusión de las fusiones debido a su útil depurador y cómoda visualización del código. Se hace la copia de todas las fusiones en el programa, se definen todos los diferentes pines que necesitará cada una y se comienza con el desarrollo del Loop. Primero, se crea el menú para que sea impreso en el serial y le informe al usuario de las posibilidades que tiene al usar el circuito. Posteriormente, se hace la creación de casos para que el programa sepa cuáles funciones ejecutar dependiendo de las opciones del menú que el usuario escoja.

9. Creación del manual:
Se diseñó un manual claro y sencillo donde se explica el funcionamiento del circuito y se explican las diferentes acciones que se pueden hacer con éste. Posteriormente se presentó el manual a diferentes personas que dieron su opinión para corregir las parte que quizá no quedaron completamente claras desde un principio.

4. Algoritmo implementado

Algoritmo implementado en C++ y adecuado para Arduino. Se emplean punteros simples, dobles y triples, además de memoria dinámica.

define SER 2 //PUERTO SERIAL
define RCLK 3 //RELOJ DE REGISTRO DESPLAZAMIENTO
define SRCLK 4 //RELOJ DE REGISTRO DE SALIDA
//int menuIn = 0;

void setup()

//Inclizamos el serial para poder recibir datos por consola.
Serial.begin(9600);
//configuracion de puertos digitales
pinMode(SER, OUTPUT);
pinMode(RCLK, OUTPUT);
pinMode(SRCLK, OUTPUT);

//Se inicializa el serial y los relojes en LOW.
digitalWrite(SER, 0);
digitalWrite(RCLK, 0);
digitalWrite(SRCLK, 0);

//COMIENZO A AGREGAR LAS FUNCIONES PROPIAS

//Funcion que crea el doble puntero para la funcion imagen
int **CrearM()
int **matrizCero;//Puntero doble que me recorre la matriz

matrizCero = new int*[8]; // Hice que el puntero tome un arreglo de 8 punteros(que serán las filas)
for (int f = 0; f ¡8; f++)
matrizCero[f] = new int[8]; // Los 8 elementos serán las filas
return matrizCero;

//Esta el la funcion que recibe los datos de un patron y arma el doble puntero para la funcion Imagen
int **Imagen (int **matrizCero)
//Le pido al usuario que me diga cuales led prender y cuales no

```
for (int fila = 0; fila¡8; fila++)


Serial.print("");
Serial.print("Fila: ");
Serial.print(8-fila,DEC);
Serial.print("");
Serial.print(Ingrese 1 si el led debe prenderse y 0 si debe estar apagado (respectivamente por fila): ");
while(!Serial.available()¿0);
long long int dato = Serial.parseInt();
for (int columna = 0; columna¡7; columna++)
int individual = datodato = dato/10;
matrizCero[fila][columna]=individual;

matrizCero[fila][7]=dato;


return matrizCero;



//Esta es la funcion que manda los datos del doble puntero hacia los integrados
void EjecutarImagen(int ** matrizCero)


for (int fila = 0; fila¡8; fila++)
for (int columna = 0; columna¡8; columna++)


int ingreso = *(*(matrizCero+fila)+columna) ;


digitalWrite(SER, ingreso);//Se le da el valor al serial, que es el que va pasando los numeros.


digitalWrite(RCLK, 0);
digitalWrite(RCLK, 1);
digitalWrite(RCLK, 0);

digitalWrite(SRCLK, 0);
digitalWrite(SRCLK, 1);
digitalWrite(SRCLK, 0);
```

//Funcion de verificacion
void Verificacion(int num, int serial ,int dezpla,int salida)//Num lo que hace es recorrer la matriz de LEDS, la segunda variable es para mandar los datos al serial y las otras 2 ajustan los relojes.

digitalWrite(serial, num);//Se le da el valor al serial, que es el que va pasando los numeros.

digitalWrite(salida, 0);
digitalWrite(salida, 1);
digitalWrite(salida, 0);

digitalWrite(dezpla, 0);
digitalWrite(dezpla, 1);
digitalWrite(dezpla, 0);

//Crea el triple puntero para la funcion Publik
int ***ArregloPublik(int patrones)
int ***ArregloP = new int **[patrones];

for(int p = 0; p¡patrones ; p++)
ArregloP[p] = new int*[8];
for(int f = 0; f¡8 ; f++)
ArregloP[p][f] = new int[8];

return ArregloP;

//Le pide al usuario los datos para armar el triple puntero de la funcion Publik
int ***MeterP (int ***ArregloP, int patrones)
//Le pido al usuario que me diga cuales led prender y cuales no
for(int patron = 0; patron¡patrones ; patron++)
for (int fila = 0; fila¡8; fila++)

Serial.print("");
Serial.print("Numero de patron:");
Serial.print(patron+1,DEC);
Serial.print("");
Serial.print("Fila: ");
Serial.print(8-fila,DEC);
Serial.print("");
Serial.print(Ïngrese 1 si el led debe prenderse y 0 si debe estar apagado (ocho numeros por fila): ");
while(!Serial.available()¿0);
long long int dato = Serial.parseInt();
for (int columna = 0; columna¡7; columna++)
int individual = datodato = dato/10;
ArregloP[patron][fila][columna]=individual;

ArregloP[patron][fila][7]=dato;

return ArregloP;

//Envia los datos del triple puntero a los integrados
void EjecutarPub(int *** MeterP, int patrones, int secpatron)

for(int patron = 0; patron¡patrones ; patron++)
for (int fila = 0; fila¡8; fila++)
for (int columna = 0; columna¡8; columna++)

int ingreso = *(*(*(MeterP+patron)+fila)+columna) ;

digitalWrite(SER, ingreso);//Se le da el valor al serial, que es el que va pasando los numeros.

digitalWrite(RCLK, 0);
digitalWrite(RCLK, 1);
digitalWrite(RCLK, 0);

digitalWrite(SRCLK, 0);
digitalWrite(SRCLK, 1);
digitalWrite(SRCLK, 0);

```
int time = secpatron*1000;
delay(time);



//Funcion principal Publik
void Publik()


Serial.print("");
Serial.print("");
Serial.print(Ingrese la cantidad de patrones que desea imprimir: ");
while(!Serial.available()¿0);
int patrones = Serial.parseInt();
Serial.print("");
Serial.print(Ingrese el tiempo (en segundos) entre cada impresion de patrones:
");
while(!Serial.available()¿0);
int secpatron = Serial.parseInt();
int ***InicialP = ArregloPublik(patrones);
int ***MedioP = MeterP (InicialP,patrones);
EjecutarPub(MedioP,patrones,secpatron);



void loop()



//Para que no se note el menu anterior en la consola
for(int n=0;n¡10;n++) Serial.print("");
//Se imprime el menú para escoger ls fuciones
Serial.print("BIENVENIDO");
Serial.print("");
Serial.print(.Escoja una de las opciones del menu");
Serial.print("");
Serial.print("0. Detener lo que se esta ejecutando");
Serial.print("");
Serial.print("1. Secuencia de VERIFICACION del estado de los LED");
Serial.print("");
Serial.print("2. Ingrese y visualice su propia IMAGEN");
Serial.print("");
```

```
Serial.print("3. Ingrese varios patrones a visualizar");
Serial.print("");
delay(3000);


if (Serial.available()¿0)
Serial.flush();
int caso = Serial.parseInt();

switch(caso)

    case 0 :
Serial.print("");
Serial.print("DETENIENDO LO QUE SE ESTA EJECUTANDO");
Serial.print("");
for(int i = 1; i¡=65; i++)
Verificacion(0,SER,RCLK,SRCLK);
Serial.print("");
Serial.print("Para ejecutar otra opcion ingrese cualquier letra");
Serial.print("");
while(!Serial.available()¿0);
break;


case 1 :

//Para ejecutar la funcion de verificacion
Serial.print("");
Serial.print(.Ejecutando funcion de Verificacion");
for(int i = 1; i¡=65; i++)
Verificacion(i,SER,RCLK,SRCLK);//Este for es el que permite recorrer los 64
LEDS.
// delay(8000);
Serial.print("");
Serial.print("Para ejecutar otra opcion ingrese cualquier letra");
Serial.print("");
while(!Serial.available()¿0);
break;


case 2 :

//Para ejecutar la funcion de imagen
int **inicial = CrearM();
int **final = Imagen(inicial);
EjecutarImagen(final);
```

```
Serial.print("");
Serial.print("Para ejecutar otra opcion ingrese cualquier letra");
Serial.print("");
while(!Serial.available()¿0);
break;



case 3 :
Serial.print("");
Serial.print("");
Publik();
Serial.print("");
Serial.print("Para ejecutar otra opcion ingrese cualquier letra");
Serial.print("");
while(!Serial.available()¿0);
break;


//Del switch
//Del if
//Del loop
```

5. Problemas de desarrollo

Se presentaron múltiples problemas durante el desarrollo tanto del circuito como del código, mediante los que tuvimos la necesidad de investigar conceptos que no teníamos claros, buscar exhaustivamente errores de sintaxis, entre otros. Los problemas que hubo fueron:

Cuando se desarrolló el primer modelo de circuito, se vio la necesidad de usar algunos switches electrónicos que permitieran regular mediante pulsos las conexiones a tierra de los LED. Para este problema se investigó en algunos foros de electrónica y con los profesores, llegando a la implementación de ocho transistores que cumplían la función. Sin embargo, este modelo fue descartado debido a que este no cumplía los parámetros establecidos. Para el desarrollo del segundo modelo, fue necesario reducir la cantidad de pines que habían conectados al Arduino. Para solucionar este problema, se buscó la manera de emplear secuencialmente el pin de salida alterna QH', ya que este permitía conectar en serie las entradas de voltaje de cada integrado, reduciendo así la cantidad de pines de siete a tres, sin contar el de potencia y el de tierra. Durante el desarrollo de la función Publik, la implementación de un triple puntero fue un verdadero reto, puesto que a pesar de hacer el código muy corto y ordenado, hizo necesaria una investigación más completa y una dedicación de tiempo considerable durante su depuración. Finalmente, cuando se estaban realizando las pruebas del circuito con el código final, fue evidente la falta de algunas funciones que permitieran usar las funciones ya creadas sin que una interfiriera con la otra. Para solucionarlo, se creó una función que detuviera alguna función que se estuviera ejecutando y otra que le permitiera al usuario volver al menú principal sin tener que reiniciar la ejecución que estaba ejecutándose. Se presentaron varios problemas con las actualizaciones del repositorio desde otras cuentas de Github diferentes a la principal. Se llegó a implementar la actualización por ramas y por medio de Pull-requests. Además, se crearon secuencias de comandos que hicieron más fácil cada actualización. Para algunas funciones, se decidió replantear el código mientras se mantenía el mismo concepto, ya que se propuso no emplear la función shiftOut() y crear el código desde cero.

6. evolución del algoritmo

Primera implementación de la función verificación

```
define SER 2 //ENTRADA SERIAL
define RCLK 3 //RELOJ REGISTRO DESPLAZAMIENTO
define SRCLK 4 //REGISTRO DE SALIDA


//funcion que recibe 4 variables tipo int, la primera el valor que se le dara
a cada led y las siguientes los puertos analogos del arduino
void Verificacion(int num, int serial,int despla,int salida);


void setup()

//configuracion de puertos digitales de forma: OUTPUT
pinMode(SER, OUTPUT);
pinMode(RCLK, OUTPUT);
pinMode(SRCLK, OUTPUT);


//Inicializar la entrada serial y los relojes en bajo
digitalWrite(SER, 0);
digitalWrite(RCLK, 0);
digitalWrite(SRCLK, 0);

for(int i = 1; i¡=65; i++)

Verificacion(i,SER,RCLK,SRCLK);




void loop()




void Verificacion(int num, int serial ,int dezpla,int salida)

digitalWrite(serial, num);//se le da el valor para al serial el cual pasara el numero

digitalWrite(salida, 0);
digitalWrite(salida, 1);
```

digitalWrite(salida, 0);

digitalWrite(dezpla, 0);
digitalWrite(dezpla, 1);
digitalWrite(dezpla, 0);

Primera implementación de la función imagen
define SER 2 //PUERTO SERIAL
define RCLK 3 //RELOJ DE REGISTRO DESPLAZAMIENTO
define SRCLK 4 //RELOJ DE REGISTRO DE SALIDA

int *pmatriz, filas[8], columnas[8] = 0,0,0,0,0,0,0,0;

void Matrizcero()

pmatriz = filas;
for(int i = 0 ; i¡8 ; i++)
pmatriz[i] = columnas;

void Mostrar ()
for(int f = 0 ; f¡8 ; f++)
for(int c = 0 ; c¡8 ; c++)
Serial.print(*((*(pmatriz+f))+c));
Serial.print();

Serial.print("");

void setup()

//Inclizamos el serial para poder recibir datos por consola.
Serial.begin(9600);
//configuracion de puertos digitales
pinMode(SER, OUTPUT);
pinMode(RCLK, OUTPUT);

pinMode(SRCLK, OUTPUT);

//Se inicializa el serial y los relojes en LOW.
digitalWrite(SER, 0);
digitalWrite(RCLK, 0);
digitalWrite(SRCLK, 0);

void loop()

Matrizcero();
Mostrar();

VERSION 2 funcion imagen

define SER 2 //PUERTO SERIAL
define RCLK 3 //RELOJ DE REGISTRO DESPLAZAMIENTO
define SRCLK 4 //RELOJ DE REGISTRO DE SALIDA

int **matrizCero;//Puntero doble que me recorrela matriz

matrizCero = new int*[8]; // Hice que el puntero tome un arreglo de 8 punteros(que serán las filas)
for (int f = 0; f ¡8; f++)
matrizCero[f] = new int[8]; // Los 8 elementos serán las filas

void Imagen ()
//Le pido al usuario que me diga cuales led prender y cuales no
for (int fila = 0; fila¡8; fila++)
for (int columna = 0; columna¡8; columna++)
Serial.print("");
Serial.print("Fila: ");
Serial.print(fila-'0');
Serial.print("");
Serial.print(Çolumna: ");
Serial.print(columna-'0');
Serial.print(Ïngrese 1 si el led debe prenderse y 0 si no: ");
int dato = Serial.parseInt();

```
delay(3000);
matrizCero[fila][columa]=dato;



void EjecutarImagen(matrizCero);


for (int fila = 0; fila¡8; fila++)
for (int columna = 0; columna¡8; columna++)


ingreso = *(*(matrizCero+fila)+columna) ;


digitalWrite(serial, ingreso);//Se le da el valor al serial, que es el que va pasando los numeros.


digitalWrite(salida, 0);
digitalWrite(salida, 1);
digitalWrite(salida, 0);

digitalWrite(dezpla, 0);
digitalWrite(dezpla, 1);
digitalWrite(dezpla, 0);



void setup()

//Inclizamos el serial para poder recibir datos por consola.
Serial.begin(9600);
//configuracion de puertos digitales
pinMode(SER, OUTPUT);
pinMode(RCLK, OUTPUT);
pinMode(SRCLK, OUTPUT);


//Se inicializa el serial y los relojes en LOW.
digitalWrite(SER, 0);
digitalWrite(RCLK, 0);
digitalWrite(SRCLK, 0);
```

```
int main()
Imagen();


void loop()
```

FUNCIÓN PUBLIK LISTA

```
define SER 2 //PUERTO SERIAL
define RCLK 3 //RELOJ DE REGISTRO DESPLAZAMIENTO
define SRCLK 4 //RELOJ DE REGISTRO DE SALIDA

void setup()
//Inclizamos el serial para poder recibir datos por consola.
Serial.begin(9600);
//configuracion de puertos digitales
pinMode(SER, OUTPUT);
pinMode(RCLK, OUTPUT);
pinMode(SRCLK, OUTPUT);


//Se inicializa el serial y los relojes en LOW.
digitalWrite(SER, 0);
digitalWrite(RCLK, 0);
digitalWrite(SRCLK, 0);



int ***ArregloPublik( int patrones)
int ***ArregloP = new int **[patrones];


for(int p = 0; p¡patrones ; p++)
ArregloP[p] = new int*[8];
for(int f = 0; f¡8 ; f++)
ArregloP[p][f] = new int[8];
```

return ArregloP;

int ***MeterP (int ***ArregloP, int patrones)
//Le pido al usuario que me diga cuales led prender y cuales no
for(int patron = 0; patron¡patrones ; patron++)
for (int fila = 0; fila¡8; fila++)

Serial.print("");
Serial.print("Numero de patron:");
Serial.print(patron+1,DEC);
Serial.print("");
Serial.print("Fila: ");
Serial.print(8-fila,DEC);
Serial.print("");
Serial.print(Ïngrese 1 si el led debe prenderse y 0 si debe estar apagado (ocho numeros por fila): ");
while(!Serial.available()¿0);
long long int dato = Serial.parseInt();
for (int columna = 0; columna¡7; columna++)
int individual = dato
dato = dato/10;
ArregloP[patron][fila][columna]=individual;

ArregloP[patron][fila][7]=dato;

return ArregloP;

void EjecutarPub(int *** MeterP, int patrones, int secpatron)

for(int patron = 0; patron¡patrones ; patron++)
for (int fila = 0; fila¡8; fila++)
for (int columna = 0; columna¡8; columna++)

int ingreso = *(*(*(MeterP+patron)+fila)+columna) ;

digitalWrite(SER, ingreso);//Se le da el valor al serial, que es el que va pasando los numeros.

digitalWrite(RCLK, 0);

```
digitalWrite(RCLK, 1);
digitalWrite(RCLK, 0);

digitalWrite(SRCLK, 0);
digitalWrite(SRCLK, 1);
digitalWrite(SRCLK, 0);


int time = secpatron*1000;
delay(time);
```

```
    void Publik()

Serial.print(Ïngrese la cantidad de patrones que desea imprimir: ");
while(!Serial.available()¿0);
int patrones = Serial.parseInt();
Serial.print(Ïngrese el tiempo (en segundos) entre cada impresion de patrones:
");
while(!Serial.available()¿0);
int secpatron = Serial.parseInt();
int ***InicialP = ArregloPublik(patrones);
int ***MedioP = MeterP (InicialP,patrones);
EjecutarPub(MedioP,patrones,secpatron);
```

```
void loop()

Publik();
```

Luego de esta versión ya se obtuvo la versión final que se ve en numeral 4 del índice.

7. Manual de uso

El funcionamiento del circuito es muy intuitivo. Se dispone de un menú en la consola que indica las posibles opciones, cuyas funciones son:

0 - Esta opción se puede implementar cuando se desea apagar todos los LED de la matriz. Se recomienda escoger esta opción cuando se finalice el uso de alguna opción y se desee cambiar a una nueva.

1 - Esta opción se encarga de activar todos los LED de la matriz para verificar que funcionan bien y que no hay errores en su encendido.

2 - Esta opción le permite al usuario ingresar por filas un patrón. Se deben seguir los siguientes pasos para cada vez que se utilice esta función:
a. Enviar un 2 en el menú.
b. Ingresar de a un número de ocho (8) dígitos para cada fila (desde la última hasta la primera) del patrón que se desea ingresar. Es importante que el número solo tenga ocho dígitos y que se coloque uno para prender el LED o cero para apagarlo en la posición respectiva.
c. No enviar el número siguiente hasta que la consola lo indique.

3 - Esta opción permite implementar la opción 2 varias veces. Es importante ingresar un número entero de patrones y segundos en esta opción. Posteriormente, la consola le irá indicando en qué parte de la programación de los patrones va. Para cada nuevo patrón recuerde que debe ir llenando desde la última fila de la matriz hasta la primera siguiendo los pasos de la opción 2.