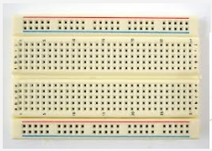
TECLADOS MATRICIALES

Arduino y los keypads de 4x4 ***(Versión 06-5-18)***

OBJETIVOS

- Familiarizarnos con un teclado matricial de 4×4.
- Usar otra librería externa: KeyPad.
- Volver a revisar los arrays, esta vez de dos dimensiones.

MATERIAL REQUERIDO.

	**Arduino Uno o similar**.Esta sesión acepta cualquier otro modelo de Arduino.
 	Una Protoboard más cables.
	Un **teclado matricial 4×4**

COMO FUNCIONA UN TECLADO MATRICIAL

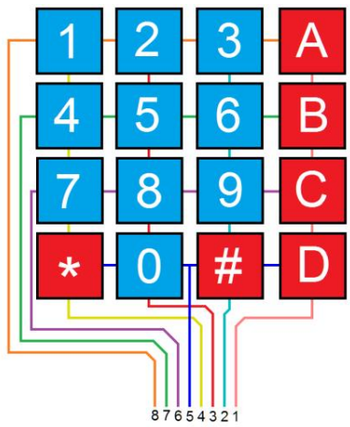
Un teclado no es más que una colección de botones, a cada uno de los cuales le asignamos un símbolo o una función determinada. Pero botones al fin y al cabo.

Leer botones es algo que ya no tiene secretos para nosotros, pero si conectáramos cada tecla a un pin digital de nuestro Arduino, pronto estaríamos en apuros.

El teclado de nuestro ordenador suele ser de alrededor de 106 teclas, así que el método de fuerza bruta va a entrar en apuros rápidamente. Necesitamos otra solución.

Y como el mundo está lleno de gente ingeniosa se les ocurrió una solución de lo más elegante, una **matriz de teclas**.

Vamos a ver un ejemplo con un pequeño **teclado numérico** de 16 teclas tipo los de los teléfonos móviles o los de los cajeros automáticos.



Para que nuestro Arduino pueda saber que tecla se pulsa, basta con poner tensión en las filas de forma secuencial y luego leer las columnas para ver cuál de ellas tiene HIGH.Los **teclados matriciales** usan una combinación de filas y columnas para conocer el estado de los botones. Cada tecla es un pulsador conectado a una fila y a una columna. Cuando se pulsa una de las teclas, se cierra una conexión **única** entre una fila y una columna.

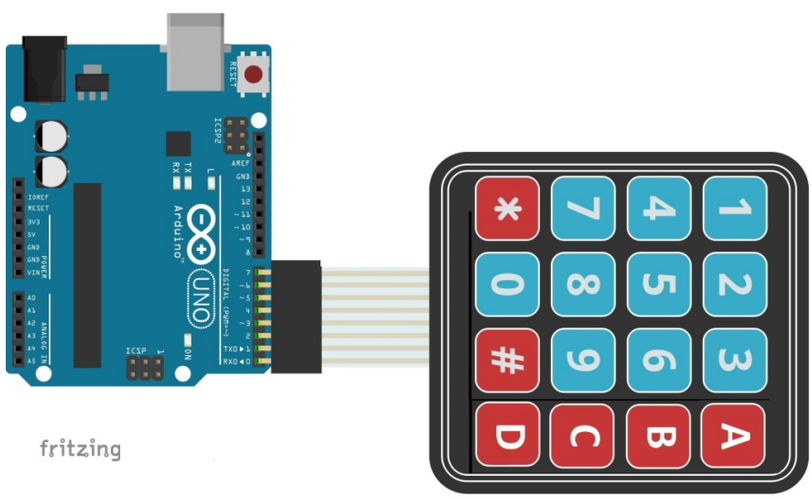
Por ejemplo, ponemos HIGH en la primera fila (hilo 8 en el diagrama de la derecha) y después leemos sucesivamente los hilos correspondientes a las columnas (hilos 4, 3, 2,1). Si ninguno está en HIGH es que no se ha pulsado ninguna tecla de la primera fila.

Pasamos a la segunda fila (hilo 7) y ponemos HIGH, si al leer los hilos 4, 3, 2,1 encontramos que el hilo 1 está en HIGH, es que se ha pulsado la tecla correspondiente a la “B”.

De este modo, para leer un teclado matricial de 4×4 necesitamos 8 hilos en lugar de 16, aunque nos dará un poco más de guerra a la hora de programar. Para un teclado de PC 106 teclas bastaría una matriz de 10×11 o sea 21 hilos en vez de 106.

DIAGRAMA DE CONEXIÓN

Lo más práctico es conectar el teclado a Arduino directamente, podemos usar cables de protoboard, o bien con un peine de pines para que no se suelte nada.



Esta es otra de esas raras ocasiones en que un esquema electrónico tendría poco sentido, así que pasaremos rápidamente a ver el tema del programa.

- *Un apunte rápido: Por comodidad en el diagrama, y porque en los ejemplos que vienen con Arduino utilizan estos mismos números de pines del 0 al 7, los hemos mantenido. Pero conviene saber que Arduino usa los pines 0 y 1 para comunicarse a través del USB, por lo que no es buena usarlos para otra cosa cuando vayamos a usar el puerto serie.*

EL PROGRAMA DE LECTURA

Como ya dijimos, el programa ha de consistir en ir dando tensión consecutivamente a los pines de las filas 8, 7, 6, 5 y después ir leyendo los valores de las columnas para detectar que teclas se han pulsado.

No es un programa complicado, pero antes de que el pánico se apodere de los que siguen la sesión con creciente temor (*y recuerden que tienen algo urgente e ineludible que hacer*) comentaremos que alguien ha tenido este problema antes que nosotros y amablemente ha puesto a nuestra disposición una librería llamada KeyPad ideal para leer este tipo de matrices sin complicaciones,.

- Lo que no quita para que podáis escribir un programa que lea directamente la matriz y reconozcáis las pulsaciones. Animo, no es difícil y el ejercicio siempre es bueno.

Lo primero es descargar la librería e instalarla. La tenéis aquí **Descargar**, o bien en la página de Arduino

http://playground.arduino.cc/Code/Keypad#Download

***RECORDAR:* En la seccion de bibiliotecas del Tutor de Arduino tiene una copia de la misma**

Una vez instalada para usarla basta con que le indiquéis que deseáis usarla. SI hacéis Programa\Importar Librería \ Keypad veréis que incluye la siguiente línea en vuestro programa:

```
#include <Keypad.h>
```

Vamos a ver como usamos esta librería. Lo primero es definir un par de contantes

```
const byte Filas = 4;           //KeyPad de 4 filas

const byte Cols = 4;            //y 4 columnas
```

Y ahora una par de arrays para indicarle a la librería que pines de Arduino corresponden a las filas y cuales a las columnas del keypad:

```
byte Pins_Filas[] = {7, 6, 5, 4};      //Pines Arduino para las
filas.

byte Pins_Cols[] = { 3, 2, 1, 0};      // Pines Arduino para las
columnas.
```

Recordad que un array es una colección de elementos que pasamos entre llaves y separados por comas. Es más cómodo que definir 8 variables.

Tendremos que definirle, además, que símbolos corresponden a cada posición de las teclas. Una nueva oportunidad para disfrutar de los arrays, pero esta vez de 2 dimensiones en vez de una.

```
char Teclas [ Filas ][ Cols ] =

    {

        {'1','2','3','A'},

        {'4','5','6','B'},

        {'7','8','9','C'},

        {'*','0','#','D'}

     };
```

- Definimos el array de 4×4 pasándole las dimensiones entre corchetes. Sus miembros van a ser 4 filas, cada una de las cuales es un array de 4 elementos, y después enumeramos los cuatro arrays de 4 miembros.
- Notemos que cada array va entre llaves. Los elementos de un array se separan por comas, pero al final de las llaves exteriores va un punto y coma porque toda el conjunto es una asignación.
- Los arrays siempre parecen complicados hasta que nos acostumbramos, pero no es para tanto y este es un buen ejemplo para entender la idea de un array de 2 dimensiones.

- Una de las curiosidades de leer una matriz de teclas y después asignar un carácter contenido en un array es que podemos redefinir el valor de las teclas sin más que cambiar el contenido del array.

- Si por lo que sea hubiéramos conectado mal los pines del keypad al Arduino, podríamos redefinir los valores del array para que coincidan con lo que pone en las teclas. Puede ser más fácil que mover de sitio los cables.
- De hecho es esa capacidad de mapear la representación de un carácter a una posición en una tabla, es lo que permite que podamos reconfigurar el mismo teclado a diferentes lenguas, sin más que cambiar los valores del array (aunque también ayudará mucho luego cambiar los símbolos rotulados en el teclado).

Por ultimo tendremos que crear una instancia de Keypad que llamaremos Teclado1

```
    Keypad Teclado1 = Keypad(makeKeymap(Teclas), Pins_Filas, Pins_Cols, Fils, Cols);
```

Que leído así asusta, pero que en realidad es una tontería. Traduciendo sería: Crea una instancia del tipo Keypad que llamaras Teclado1 y al que asignarás las teclas que tenemos en el array Teclas, y le decimos que hemos conectado las filas del keypad a los números de pines que te indico en el array Pins_Filas y las columnas al array Pins_Cols.

- MakeKeymap (Teclas) es una función disponible en la librería y que nos permite asignar un array de valores a una matriz. Se hace solo una vez al principio salvo que por lo que sea nos interese cambiarlo sobre la marcha.

Por ultimo para leer el keypad hacemos una llamada a otra función de la librería:

```
char pulsacion = Teclado1.getKey() ;
```

Por ultimo enviamos a la consola el carácter pulsado.

```
    #include <Keypad.h>            // Prog_19_1
```

```
const byte Filas = 4;          //Cuatro filas

const byte Cols = 4;           //Cuatro columnas



byte Pins_Filas[] = {7, 6, 5, 4};     //Pines Arduino para las filas

byte Pins_Cols[] = { 3, 2, 1, 0};     // Pines Arduinopara las columnas

char Teclas [ Filas ][ Cols ] =

     {

         {'1','2','3','A'},

         {'4','5','6','B'},

         {'7','8','9','C'},

         {'*','0','#','D'}

     };

Keypad Teclado1 = Keypad(makeKeymap(Teclas), Pins_Filas, Pins_Cols, Filas, Cols);



void setup()

   {             Serial.begin(9600) ; }



void loop()

   {       char pulsacion = Teclado1.getKey() ;

           if (pulsacion != 0)              // Si el valor es 0 es que no se

               Serial.println(pulsacion);   // se ha pulsado ninguna tecla

   }
```

Por cierto, en C++ y en otros lenguajes es muy frecuente escribir la instrucción:

```
if (pulsación != 0)
```

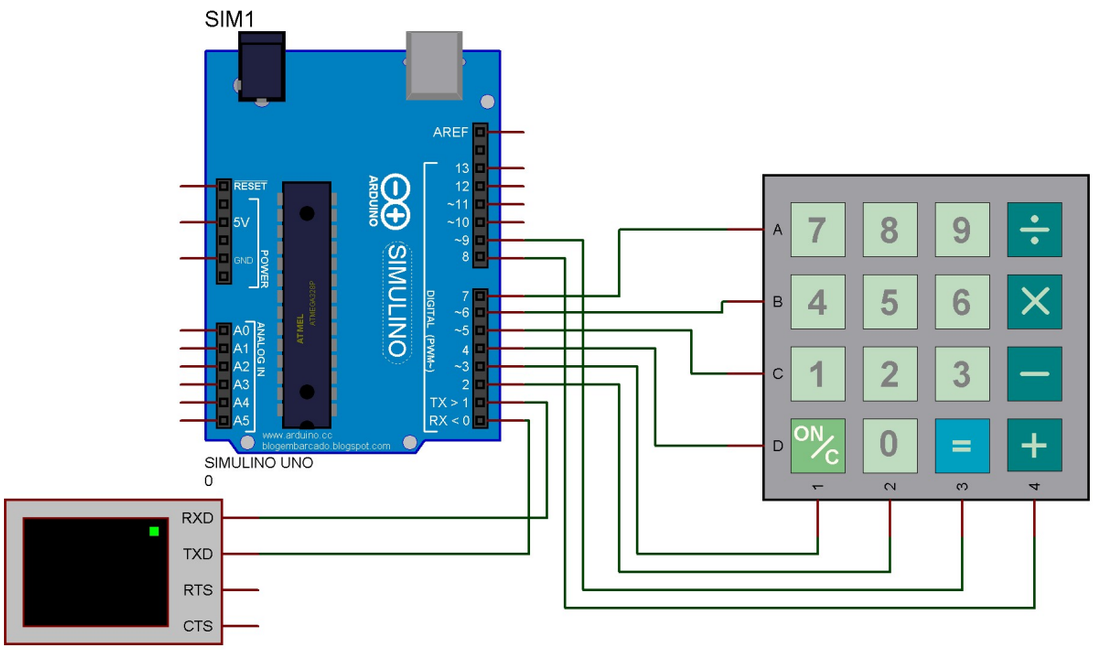
De esta otra manera (más elegante) *completamente equivalente*:

```
if (pulsación )
```

Porque la condición del if *debe evaluarse a* TRUE o FALSE y para los ordenadores un 0 es FALSE sin remisión, pero cualquier otro valor que no sea 0 es TRUE, incluyendo un carácter.

Ejemplo de manejo de Teclado matricial:

Programa Matricial1



Cada vez que se presiona una tecla, aparece cual se presiono en el Monitor Serie.

El programa:

```
//Programa Matricial Ejemplo 1

#include <Keypad.h>

    const byte Filas = 4;          //Cuatro filas

    const byte Cols = 4;           //Cuatro columnas

    byte Pins_Filas[] = {7, 6, 5, 4};     //Pines Arduino para las filas

    byte Pins_Cols[] = { 3, 2, 9, 8};     // Pines Arduino para las columnas

    char Teclas [ Filas ][ Cols ] =       //Define que símbolos corresponden a cada

                                  //posición de las teclas.

        {

            {'7','8','9','/'},

            {'4','5','6','X'},

            {'1','2','3','-'},

            {'C','0','=','+'}
```

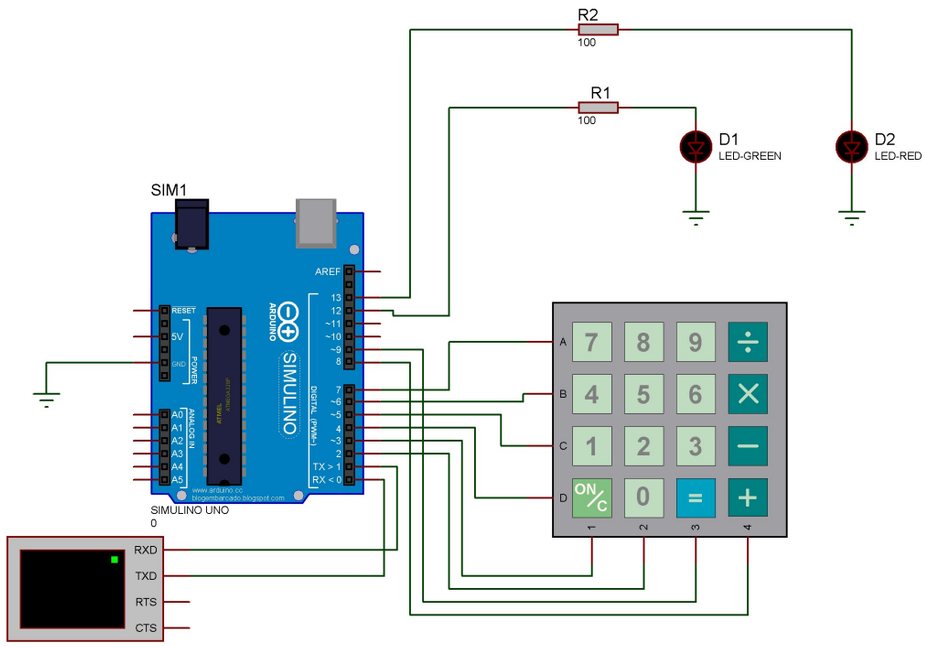
```
        };

    Keypad Teclado1 = Keypad(makeKeymap(Teclas), Pins_Filas, Pins_Cols, Filas, Cols);

     void setup()

       {

        Serial.begin(9600) ;

       }

void loop()

       {

             char pulsacion = Teclado1.getKey() ;

             if (pulsacion != 0)              // Si el valor es 0 es que no se

               {

                Serial.println(pulsacion);   // se ha pulsado ninguna tecla

               }

       }
```

Programa Matricial2

```
//Programa Matricial Ejemplo 2
//Presionando 1 enciende LED verde
//Presionando 2 enciende LED rojo
//Presionando 3 apaga ambos LEDs

 #include <Keypad.h>

    const byte Filas = 4;          //Cuatro filas
    const byte Cols = 4;           //Cuatro columnas

    byte Pins_Filas[] = {7, 6, 5, 4};     //Pines Arduino para las filas
    byte Pins_Cols[] = { 3, 2, 9, 8};     // Pines Arduino para las columnas
    char Teclas [ Filas ][ Cols ] =       //Define que símbolos corresponden a cada
                                  //posición de las teclas.
        {
           {'7','8','9','/'},
           {'4','5','6','X'},
           {'1','2','3','-'},
           {'C','0','=','+'}
        };
    Keypad Teclado1 = Keypad(makeKeymap(Teclas), Pins_Filas, Pins_Cols, Filas, Cols);

 void setup()
       {
        pinMode(13, OUTPUT); // configura 'pin' como salida- LED ROJO
```

```
        pinMode(12, OUTPUT); // configura 'pin' como salida- LED VERDE

        Serial.begin(9600) ;

         //digitalWrite(12,HIGH );//Enciende LED Rojo
        //digitalWrite(13,HIGH );//Enciende LED verde
       }


 void loop()
       {

            int pulsacion = Teclado1.getKey() ;
/* La instruccion  Teclado1.getKey() entrega el codigo ASCII de lo que
 *  viene del teclado matricial (o se dato tipo char).
 *  Por ejemplo cuado llega lo que se genera al presionar (1)
 *  entonces se envia 49.  Si (2) se envia 50. Si (3) se envia 51.
 *  Recuerde la tabla ASCII y vea el apunte EnvioCaracteresArduino.pdf
 */

           if (pulsacion != 0)


               {
                Serial.println(pulsacion);   // se ha pulsado ninguna tecla
               }


                 if(pulsacion == 49)
                        {

                        digitalWrite(12,HIGH );//Enciende LED Rojo
                        Serial.println(pulsacion);

                        }

              if(pulsacion== 50)
                        {

                         digitalWrite(13,HIGH );//Enciende LED Verde
                        Serial.println(pulsacion);

                        }

               if(pulsacion== 51)
                        {

                         digitalWrite(12,LOW );
                         digitalWrite(13,LOW );
                         Serial.println(pulsacion);
                        }
           }
```

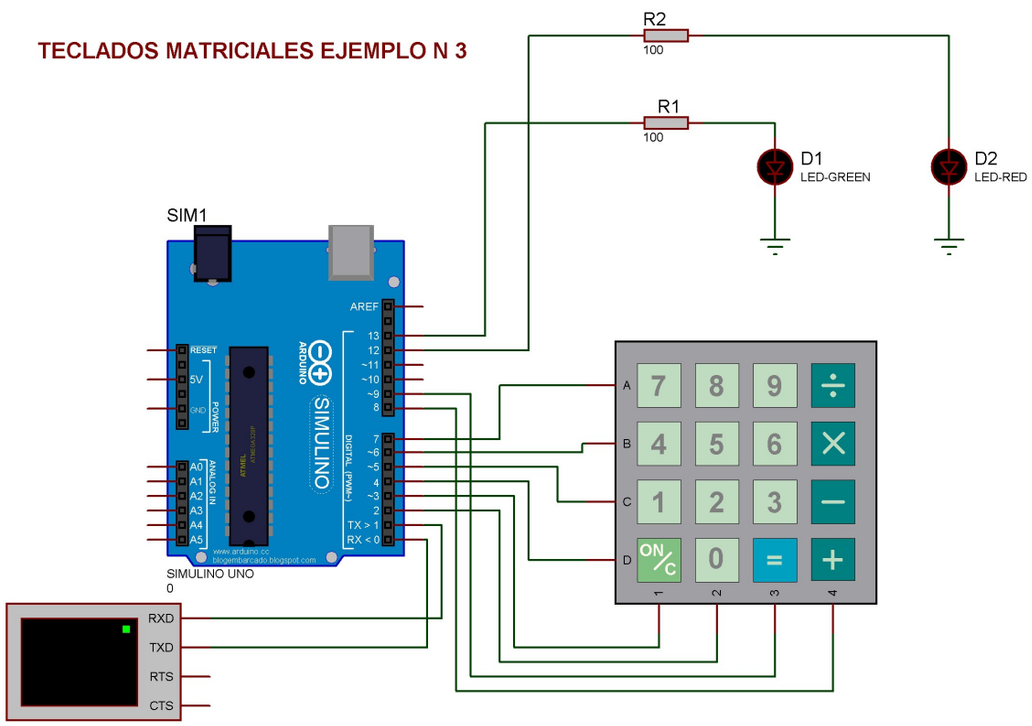
TECLADOS MATRICIALES EJEMPLO N 2

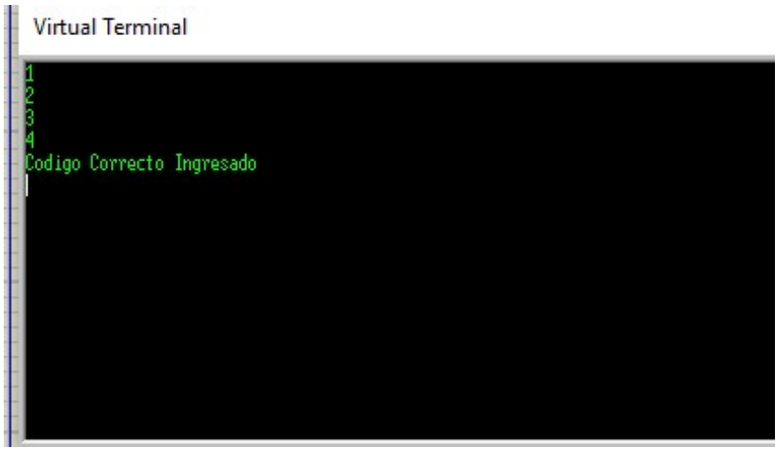


Programa Matricial3

```
//Programa Matricial3
//Se enciende un LED cuando se ha digitado el codigo correcto

#include <Keypad.h>


char contrasena[]="1234"; //aquí escribimos la contraseña de 4 dígitos

char codigo[4]; //Cadena donde se guardaran los caracteres de las teclas presionadas
int cont=0; //variable que se incrementara al presionar las teclas

const byte ROWS = 4; //Numero de filas del teclado que se esta usando
const byte COLS = 4; //Numero de columnas del teclado que se esta usando

char hexaKeys[ROWS][COLS] = //Aquí pondremos la disposición de los caracteres
                      //tal cual están en nuestro teclado
{
 {'7','8','9','/'},
 {'4','5','6','*'},
 {'1','2','3','-'},
 {'*','0','=','+'}
};

byte rowPins[ROWS] = {7, 6, 5, 4}; //Seleccionamos los pines en el arduino
                           //donde iran conectadas las filas
byte colPins[COLS] = {3, 2, 9, 8}; //Seleccionamos los pines en el arduino
                           //donde iran conectadas las columnas

Keypad customKeypad = Keypad(makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS); //inicializa
                                                    //el teclado

void setup()
{
 pinMode(13, OUTPUT); //Pin 13 como salida
 pinMode(12, OUTPUT); //Pin 12 como salida
 digitalWrite(12,HIGH);
```

```
 Serial.begin(9600); //inicializar puerto serie
}

void loop()
{
 char customKey = customKeypad.getKey(); //se guarda en la variable customKey
                                  //el caracter de la tecla presionada
 if (customKey != NO_KEY) //se evalúa si se presionó una tecla
 {
 codigo[cont]=customKey; //se guarda caracter por caracter en el arreglo codigo[]
 Serial.println(codigo[cont]); //se imprime en el puerto serie la tecla presionada
 cont=cont+1; //incrementamos la variable cont
 if(cont==4) //si ya fueron presionadas 4 teclas se evalúa si la contraseña es correcta
 {
 if(codigo[0]==contrasena[0] & codigo[1]==contrasena[1] & codigo[2]==contrasena[2] &
codigo[3]==contrasena[3])
 {
 digitalWrite(13,!digitalRead(13)); //si la contraseña fué correcta se enciende o se apaga el led del pin13
                            //Aqui usa el recurso de NEGAR estado de PIN13.
                            //Asi para encender el LED de PIN 13 se debe ingresar la
                            //contraseña, lo mismo para apagarlo. LED VERDE

  digitalWrite(12,!digitalRead(12));//Cambia estado LED rojo de espera.

  Serial.println("Codigo Correcto Ingresado");


 }
 cont=0; //resetear a 0 la variable cont
 }
 }
}
```





Notar que la instrucción que define la contraseña es:

```
char contrasena[ ]="1234"; //aquí escribimos la contraseña de 4 dígitos
```

RESUMEN DE LA SESIÓN

- Hemos usado un equipad o teclado matricial.
- Son prácticos y muy sencillos de usar.
- Están disponibles en múltiples formatos y tipos, Aquí hemos usado uno típico de 4×4, pero también los hay 4×3 y en diferentes materiales y acabados
- Un antiguo teléfono móvil roto se puede desmontar y usar su teclado además de otras cosas, y se puede utilizar de la forma que hemos descrito.
- Hemos usado la excusa del teclado para volver sobre los arrays. Vale la pena dedicarles un poco de tiempo pues nos ayudaran a resolver cómodamente más de un problemas de programación.
- Hemos conocido otra librería externa, KeyPad. Nos vendrá muy bien en nuestros proyectos. Más adelante podriamos hacer algún ejemplo más sofisticado con un keypad, como hacer que una cierta combinación abra una cerradura de servo por ejemplo.

Fuente:

https://www.prometec.net/teclados-matriciales/#

Adaptado: Prof. Bolaños DJ