Parcial 1 Informática II

Informa2 S.A.S

Johan David Rojas
Luis Fernando Torres
Mateo Alejandro Bravo

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

Índice

1. Análisis del problema

Para solucionar este desafio, se plantea un algoritmos que esta dividido en 2 etapas.

- **Etapa de Software** En esta etapa se recopila toda la información que el usuario ingresa por el monitor serial para ser procesada en la siguiente etapa. Aquí se debe ingresar el patrón de caracteres que el usuario desea ver en la matriz de LEDs y se realiza el proceso de llevar cada carecter a un sistema binario que la electronica del montaje entienda, es decir se codifica la solicitud del usuario.
- **Etapa de Hardware** En esta etapa se cuenta con un arreglo asignado para cada carácter. En el montaje del circuito se tiene ocho 74HC595 (cada uno controla una fila de la matriz) conectados de tal forma que llevándole información por el puerto SER mostrará un patrón correspondiente al carácter ingresado por el monitor serial. Cada posicion del arreglo nos dará información del dato a llevar al circuito integrado. Por cada circuito integrado habrá ocho salidas que en total serán 64 salidas que determinarán el estado de los LED, a cada LED se le conecta un resistor para evitar daños.

Una vez explicadas las etapas que se van a tener en cuenta en la solución de este desafío, se comienza a analizar la manera en la que se debe realizar el procedimiento para mostrar un solo carácter en el arreglo de LEDs, es decir como se va a implementar la función Imagen(). Para ello se realiza una recopilación de ideas, con posibles pasos que debe llevar cada etapa y unas observaciones para tener muy claro que se hace, tal como se muestra en la figura(**1**) :

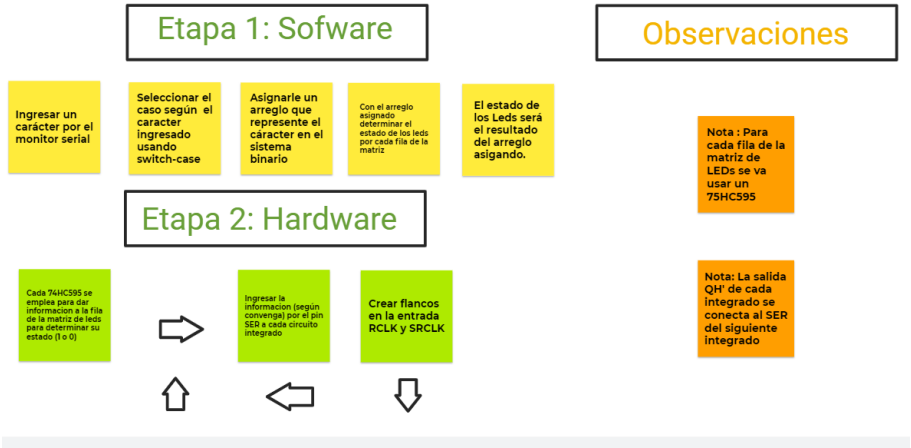

Figura 1: Procedimiento para mostrar un solo carácter

2. Proceso inicial de desarrollo

Para llevar a cabo la solución de este reto, lo que se busca es que el carácter que la persona ingrese por el monitor serial, sea reflejado en una matriz de LEDs y que se pueda observar de manera clara. Para ello se debe utilizar el sistema binario, ya que este indica el estado de un LED, donde cero (0) representa apagado y el uno (1) encendido.

Teniendo en cuenta lo dicho anteriormente, lo que se va a hacer es mostrar el carácter ingresado, en un arreglo de números en sistema binario, indicando que LEDs deben encenderse para formar dicho patrón ingresado en la matriz. tal como se muestre en la figura(**2**):

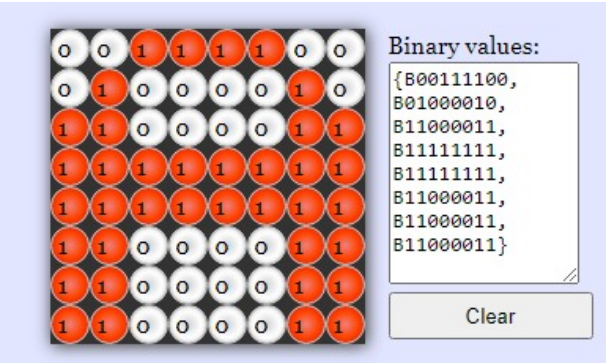

Figura 2: Letra A codificada para representar en la matriz

En la figura(**2**), se puede observar el ejemplo con la letra A. En la matriz de LEDs que está ubicada en la parte izquierda, los números uno que están en color naranja, indican como se va a ver esta letra en la matriz real de LEDs. Por otra parte derecha muestra el arreglo que representa esta letra en sistema binario, el cual es el que se va a procesar por medio del integrado 74HC595. Para realizar este proceso de conversion se hizo uso de una herramineta online de mucha ayuda [1], la cual tiene como fin generar el arreglo en binario para ahorrar tiempo.

Después de obtener el arreglo en binario de toda la matriz, lo que se hace es separar cada fila, la cual está hecha con 8 LEDs, razón por la cual se debe formar subgrupos 8 bits que indican el estado de cada fila. Al realizar esta acción, el arreglo queda con un tamaño de 8 posiciones donde cada espacio representa el estado de una fila de LEDs de la matriz.

Después de tener el arreglo bien organizado, se procede a realizar la conversión de binario a decimal de cada posición. Es decir que se va a obtener un arreglo de 8 elementos, donde cada espacio tiene un número decimal. tal como lo muestra la imagen(**3**)

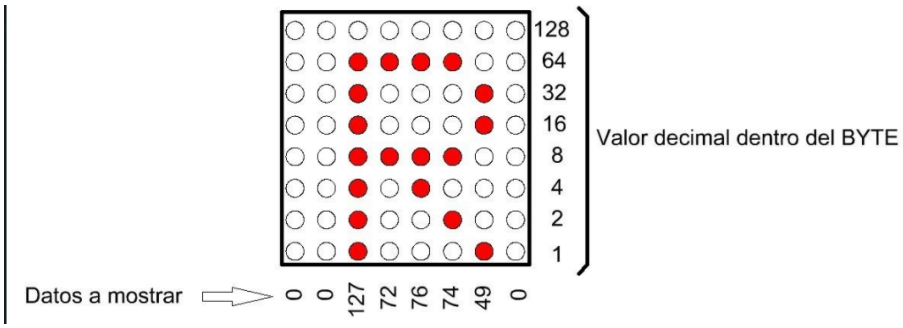

Figura 3: Paso de binario a decimal

Con el vector expresado de esta forma, es más sencillo de llevar el proceso, ya que los números decimales se puede identificar mucho más rápido los errores a la hora de imprimir.

3. Proceso de montaje

Como se comentó en la sección **1**, para el montaje del hardware se van a utilizar 8 circuitos integrados conectados entre sí. Donde el mecanismo usado para interconectar los, es que el pin de entrada SER del segundo integrado, se conecta con el pin de salida QH' del anterior integrado, formando así una cominucación en serie entre los integrados. Tal como se muestra en la siguiente figura (**4**).

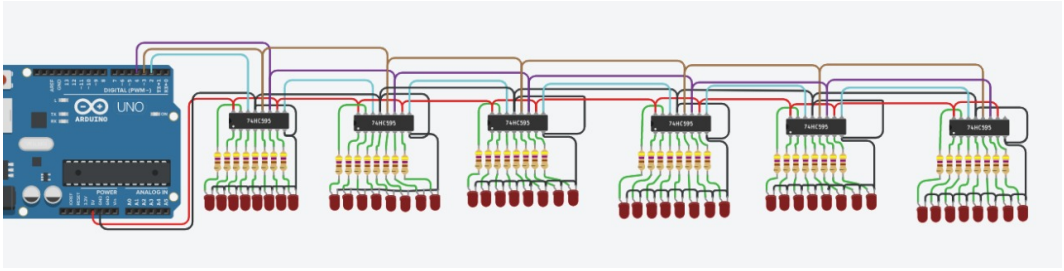

Figura 4: Interconexión de integrados

Se ha conectado 8 leds por cada circuito integrado (74HC595) donde cada configuración por CI representa una fila de nuestra matriz 8X8 de LEDs. Lo que se hace es conectar el terminal positivo (ánodo) de un LED en serie con una resistencia (470 ohm) por cada salida Qi de los CI (desde Qa hasta Qh), y la salida Qh' se interconecta con la entrada SER del siguiente circuito integrado.Por otra parte, los terminales negativos que quedan libres en cada LED se interconectan y son enviados a GND (tierra común), este proceso se repite por

cada CI hasta tener una configuración de 8 circuitos integrados y se acomoda cada fila una debajo de la otra para formar una matriz 8x8 de Leds tal como se muestra en la figura (**5**).

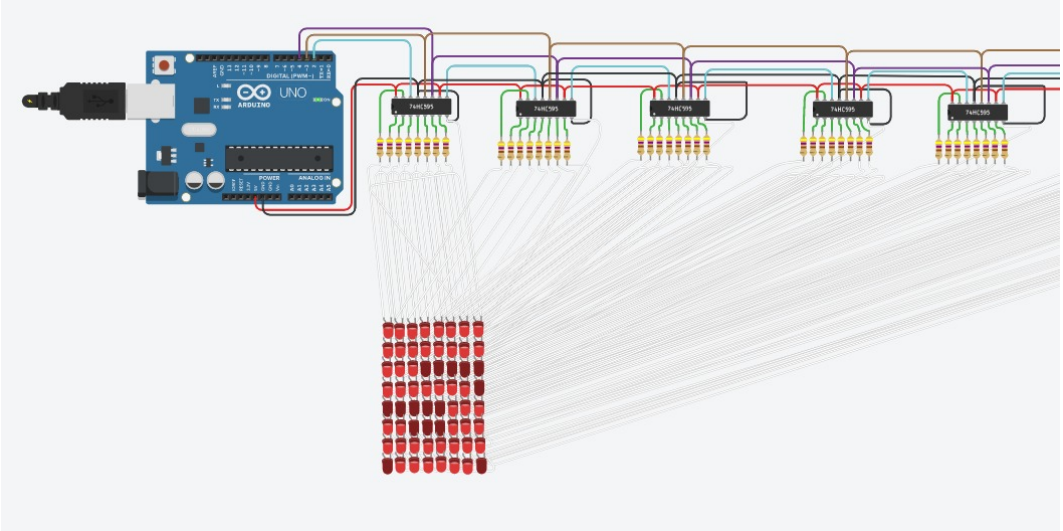

Figura 5: Matriz de LEDs funcionando

Cabe resaltar que para este primer prototipo se realiza el montaje y la interconexión de los componentes en el aire por facilidad y también para obtener los primeros resultados de prueba. En los siguientes prototipos se mejorará el montaje para organizar más el circuito.

Para poder mostrar el número 5 en la matriz de LEDs , se debe programar el Arduino.A continuación, se presenta el código, que permite realizar la accion mostrada en la figura(**5**).

```
cconst int SER = 2;
const int RCLK = 3;
const int SRCLK = 4;


char TECLA;

void ledon(int);

void prueba();

void setup()
{
```

```
  Serial.begin(9600);

  pinMode(SER , OUTPUT);
  pinMode(RCLK , OUTPUT);
  pinMode(SRCLK , OUTPUT);

  digitalWrite(SER , 0);
  digitalWrite(RCLK , 0);
  digitalWrite(SRCLK , 0);


  prueba1(); // Muestra el patron 5

}


void loop(){

}


void ledOn(int n){

        /*

    Funcion que enciende una fila del arreglo de LEDS
    segun n

    El parametro de entrada (n) es un entero que se convierte
    a binario para generar datos a la entrada SER


    */
    int b;
        for (int i = 0 ; i < 8 ; i++){
    b = n%2;
    n = n/2;

     //[paso 1]
    digitalWrite(SER ,b);

    // [Paso 2]
    digitalWrite(SRCLK , 0);
    digitalWrite(SRCLK , 1);
```

```
    digitalWrite(SRCLK , 0);

    // [Paso 3]
    digitalWrite(RCLK , 0);
    digitalWrite(RCLK , 1);
    digitalWrite(RCLK , 0);
  }
}


void prueba1(){

    /* Funcion que muestra el numero 5 en un patron de LEDS*/

  int p1[8] = {126,255,199,7,254,224,255,255};

  for (int k = 0; k < 8; k++){
    ledOn(*(p1+k));
  }
}


void verificacion(){

  /* Funcion que verifica que todos
  los LEDS esten encendidos*/

  int p2[8] = {255,255,255,255,255,255,255,255};

  for (int k = 0; k < 8; k++){
    ledOn(*(p2+k));
  }
}
```

Donde se puede apreciar que tiene 2 funciones principales.

- **void ledOn()**: Función encargada de encender una fila del arreglo de LEDs , a partir del parámetro de entrada **n** el cual es un número entero que es convertido a binario para generar el arreglo que será procesada por la entrada SER.
- **void verificacion()**:Función la cual le permitirá al usuario verificar que todos los LEDs que están conformando la matriz 8x8 estén funcionando correctamente y así no haya algún tipo de inconveniente al momento de generar el patrón que el usuario desee formar con los LEDs. Para esto básicamente se debe encender los 64 leds y así cerciorarce de que cada uno está trabajando.

```
void verificacion(){

  /* Funcion que verifica que todos
  los LEDS esten encendidos*/

  int p2[8] = {255,255,255,255,255,255,255,255};

  for (int k = 0; k < 8; k++){
    ledOn(*(p2+k));
  }
}
```

Por ejemplo para el código 3, se crea un arreglo tipo entero de 8 posiciones donde cada posición tiene almacenado un número decimal el cual representa una de las 8 filas de la matriz de leds, en este caso tenemos el número decimal 255 en las 8 posiciones del arreglo p2[8], ya que este número al convertirlo a binario con la función ledOn representa el número binario 11111111(lo cual significa que se tiene encendidos los 8 LEDs de esa fila) y así sucesivamente se realiza este proceso con el ciclo For 8 veces, al terminar este ciclo se tiene encendidos los 8 leds de las 8 filas, el cual representa la matriz 8x8 de LEDs encendidos todos.

Este proceso es de vital importancia dado que es la base de como se va a imprimir todas las letras y el patrón que ingrese el usuario.

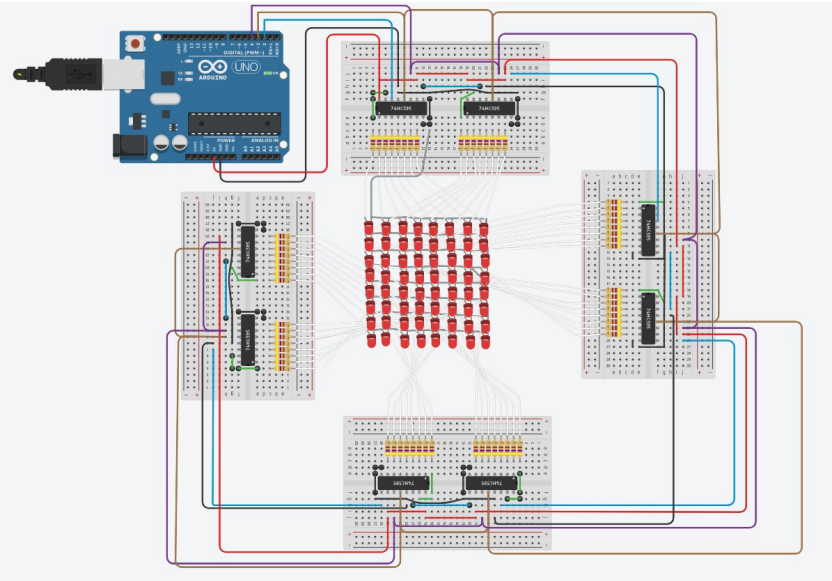

Figura 6: Montaje final hardware

En la figura (**6**) se muestra como se modificó el montaje de Tinkercad, ya que anteriormente se tenía el montaje en el aire, esto porque es más fácil visualizar cualquier error de falla de conexión. No obtasnte, ya pusimos a prueba el sistema tanto en programación como en hardware. Entonces ya se realizó el montaje en tablas de prueba o protoboard, donde cada una tiene 2 integrados para mayor orden.

Se puso en funcionalidad las lineas de código que hacen interactuar al usuario con el Monitor Serial. En la figura (**7**) se muestra que se va a ingresar el patrón HEY, con un delay de dos segundos. En la figura (**8**) los leds muestran una configuración mostrando el patrón Y, que es el ultimo patrón de la secuencia.

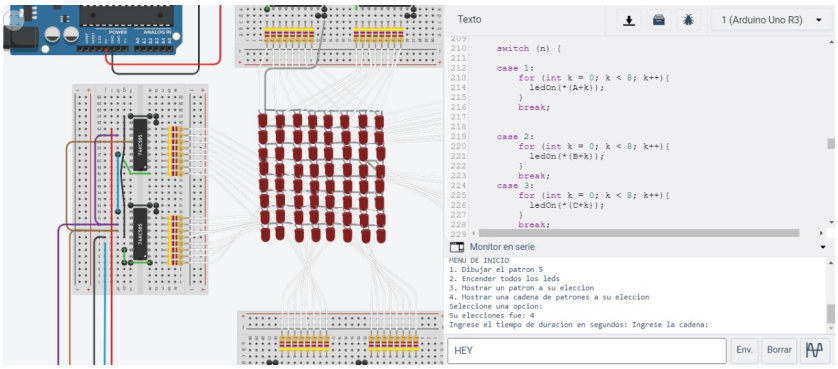

Figura 7: Solicitud de información por Monitor en Serie

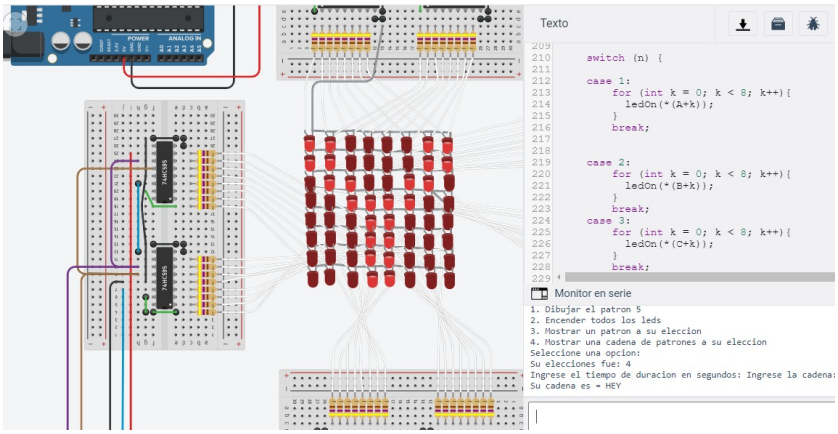

Figura 8: Finalización de ejecucion con la entrada HEY

El siguiente código muestra el usuario va a interactuar con la consola, se observa que la varaible *ENTRADA* es la que decide a que parte del programa

desea ir, entre las opciones está, **1.** Verficiar el *patron 5*, **2.** Verificar si todos los Leds están enceidos, **3.** Ingresar un carácter por teclado para mostrar en el arreglo de leds, **4.** Ingresar una cadena de cáracteres para mostrar en el arreglo de leds, cada condicional, según la entrada, lleva a otras instrucciones que son quienes ejecutan las procesos sobre el funcionamiento del arreglo

```
void verificacion(){
    while(Serial.available()==0){}

  int ENTRADA = Serial.parseInt(); // lee un entero
  Serial.print("Su elecciones fue: ");
  Serial.println(ENTRADA);



  if (ENTRADA == 1){
    // Ingresa a verificar el patron 5
    prueba1();
  }

  else if(ENTRADA == 2){
    // Ingresa a realizar la verifiacion
    // que todos los LEDS esten encnendidos

    verificacion();
  }

  else if(ENTRADA == 3){

    Serial.println("");

    // pedir al usuario un caracter para ser
    // visualizado

    Serial.println("Ingrese un caracter para mostrar el patron: ");
    while(Serial.available()==0){}

        int ENTRADA3 = Serial.read(); // lee ascii
        //Serial.println("Su caracter ingresado fue: ");
        //Serial.println(ENTRADA3);

    imagen(ENTRADA3); // A -> 65  caso 1 = A 65-64 -> 1


  }
```

```
  if(ENTRADA == 4){
    // Pide al usuario que le ingrese una palabra para ser
    // mostrada secuencialmente, el tiempo entre patron y
    // patron lo define el usuario y la variable se llama
    // TIEMPO
        Serial.println("");
    Serial.print("Ingrese el tiempo de duracion en segundos: ");
    while(Serial.available()==0){}
    int SEGU = Serial.parseInt();

        publik(SEGU);

  }
}
  }
}
```

En última instancia sé realizaron unos cambios para utilizar memoria dinámica en cada carácter, con el fin de que una vez usado un arreglo, se borre del programa y así se obtenga mejor rendimiento y más eficiencia en el uso de la memoria,tal como se observa en el código 3

```
else if (n == 56) {
    //numero 8
    p_a = new int[8];
    p_a[0] = 60; p_a[1] = 66; p_a[2] = 66;
    p_a[3] = 60; p_a[4] = 60; p_a[5] = 66;
    p_a[6] = 66; p_a[7] = 60;
    for (int i=0; i<8; i++){
        ledOn(*(p_a+i));
    }
    delete[] p_a;
}
```

Este código se implementa con cada uno de los caracteres que tiene el problema, y se realza la selección mediante un condicional para poder hace que solo se cree el areglo que se requiere y no todos los posibles arreglos de el sistema alfa-numérico como toca hacerlo con un switch-case

4. Problemas obtenidos

En esta sección se habla sobre los errores y problemas que se tuvo a la hora de realizar el desafío, con el fin de ponerlos al conocimiento del estudiantado para aprender a trabajar y solucionar en momentos de crisis, o bajo presión.

En primer lugar, el problema más notable que se obtuvo está relacionado con los repositorios, donde al momento de crear un repositorio compartido el día domingo 18 de abril del 2021,se subieron unos documentos de prueba para verificar el uso de esta herramienta. Tiempo después se intentó realizar commits, pero siempre se encontraban problemas con la sincronización de repositorios, arrojando errores respecto realizar un git pull,ver figura (**9**).

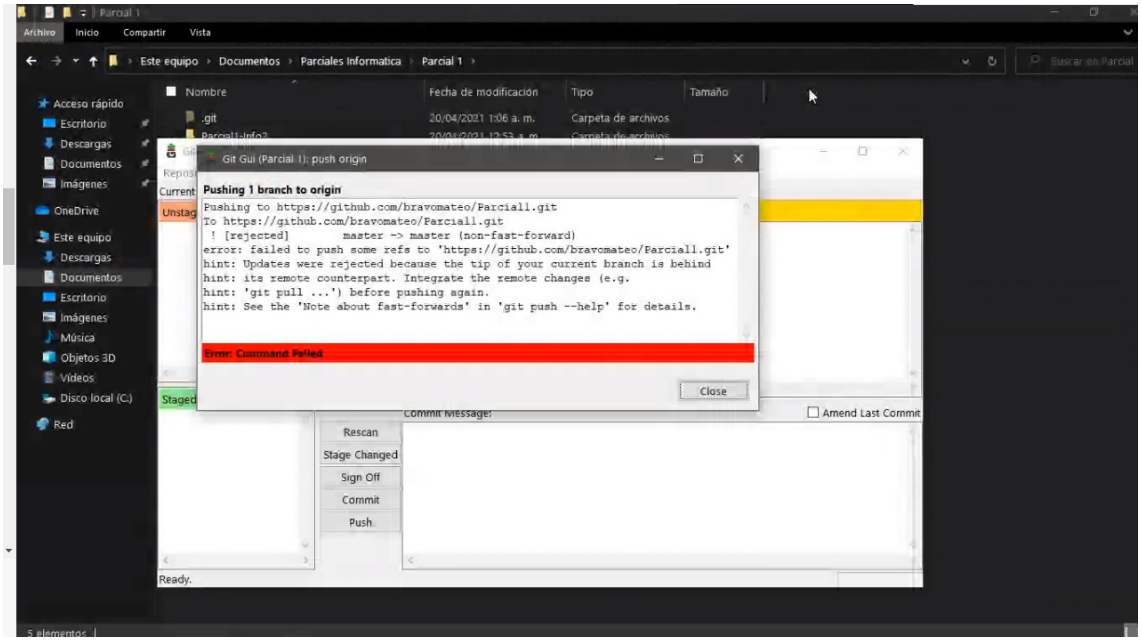

Figura 9: Error de repositorios

Se intentó arreglar este error buscando en diversas fuentes, distintas soluciones, utilizando comandos tales como, git pull remote, git fetch, git merge, incluso se optó por borrar a los contribuidores con el fin de que no intervinieran más, pero la respuesta siguió siendo negativa,razón por la que se dejó de lado el repositorio
`https://github.com/bravomateo/Parcial1.git` . Dado esta circuntancia , se tomó la iniciativa de crear un nuevo repositorio el día lunes 19 de abril sin tener ningun colaborador , para evitar tener este problema y poder continuar con el desarrollo del reto.Este problema del repositorio tuvo un fuerte impacto porque tuvo un costo de mucho tiempo.

Otro problema que se obtuvo esta relacionado con confusiones en el código, dado que al tener varias versiones del mismo, se producían errores distintos en cada uno y cuando se descargaban 2 versiones del código, no se sabía cuál era el indicado para realizar el siguiente commit.

Y por último, un problema bastante tedioso que se obtuvo, está relacionado

con la memoria dinámica, ya que al implementarla en Arduino se obtuvo muchos errores, ya que queríamos usarla con punteros previamente definidos, razón por la cual toco especificar que valor se deseaba tener en cada posición del arreglo y no se pudo realizar en un mismo paso a la hora de declararlo (ver código 3.

Referencias

[1] Learn on the fly. [Online]. Available: https://www.riyas.org/2013/12/online-led-matrix-font-generator-with.html