INFORME PARCIAL INFORMÁTICA 2

Participantes: David Alejandro Morón Acacio y Luisa María Bohórquez Ardila

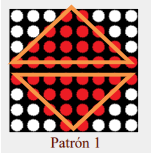
Primeros aportes (14/09/2023)

1. ***Análisis del problema y consideraciones para la alternativa de solución propuesta***

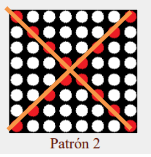
Se tiene como punto de partida el cómo representar una matriz en Arduino mediante el lenguaje de programación que maneja (el uso de arreglos, punteros, la memoria dinámica y la memoria estática para guardar y reproducir el comportamiento de los LEDs). Después de esa familiarización con el tipo de estructuras de control y de datos y de tener una fluidez con su manejo y sus definiciones al momento de usarlas (nótese que nos referimos a la parte de la codificación), resta el tener en consideración los componentes electrónicosa utilizar y concatenar (nótese que nos referimos a las características del hardware disponible).

1. ***Esquema donde describa las tareas que usted definió en el desarrollo de los algoritmos.***

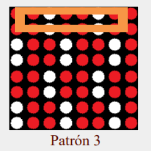
**Patrón 1:** Haremos un triángulo teniendo en cuenta las filas y luego replicaremos ese código de manera decreciente.



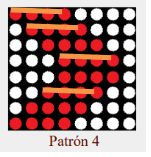
**Patrón 2:** Analizamos la figura por los extremos, a medida que va iterando la función vamos desplazando la posición de los leds encendidos unos a la izquierda(restando) y otros a la derecha(sumando).



**Patrón 3:** Haremos que cada dos renglones de la matriz se impriman, dado que son el mismo patrón, y a cada renglón se le encienden dos leds y se apaga uno.



**Patrón 4:** Encendemos 4 leds en el primer extremo y lo vamos desplazando a la derecha hasta la cuarta iteración, repetimos el patrón tal cual en la quinta y de la sexta a la octava desplazamos los leds encendidos a la izquierda



**Para encender toda la matriz,** vamos a recorrer con un arreglo cada variable asociada a un led se enciende.

1. ***Algoritmos implementados***

**Código para encender toda la matriz:**

| char high='+', low='-';  char matriz[8][8];  for(int i=0; i<8; i++){  for(int j=0;j<8;j++){  matriz[i][j]= high;  }  }  //Imprimir la matriz  for(int i=0; i<8; i++){  for(int j=0;j<8;j++){  char posicion= matriz[i][j];  char espacio= ' ';  cout<< espacio<< posicion;  }  cout<<endl;  } |
| --- |

**Código para el primer patrón:**

| char high='+', low='-';  char matriz[8][8];  int cotainferior=3, cotasuperior=5;  //primera parte de la figura  for(int i=0; i<5; i++){  for(int j=0;j<8;j++){  if(j>=cotainferior and j<cotasuperior){  for(int control2=j; control2<cotasuperior;control2++){  matriz[i][j]=high;}  }else{  matriz[i][j]=low;  }  }  if(i<4){  cotainferior-=1;  cotasuperior+=1;  }    }  //segunda parte de la figura  cotainferior=1; cotasuperior=7;  for(int i=5; i<8; i++){  for(int j=0;j<8;j++){  if(j>=cotainferior and j<cotasuperior){  for(int control2=j; control2<cotasuperior;control2++){  matriz[i][j]=high;}  }else{  matriz[i][j]=low;  }  }  cotainferior+=1;  cotasuperior-=1;  }    //Imprimir la matriz  for(int i=0; i<8; i++){  for(int j=0;j<8;j++){  char posicion= matriz[i][j];  char espacio= ' ';  cout<< espacio<< posicion;  }  cout<<endl;  } |
| --- |

**Código para el segundo patrón:**

| char high='+', low='-';  char matriz[8][8];  int extremoIzquierdo=0, extremoDerecho=7;  //Figura  for(int i=0; i<8; i++){  for(int j=0;j<8;j++){  if(j==extremoIzquierdo){  matriz[i][j]=high;  }else if(j==extremoDerecho){  matriz[i][j]=high;  }else{  matriz[i][j]=low;  }  }  extremoIzquierdo+=1;  extremoDerecho-=1;  }  //Imprimir la matriz  for(int i=0; i<8; i++){  for(int j=0;j<8;j++){  char posicion= matriz[i][j];  char espacio= ' ';  cout<< espacio<< posicion;  }  cout<<endl;  } |
| --- |

**Código para el tercer patrón:**

| int main()  {  char high = '+', low='-';  char matriz[8][8];  int i = 0;  int sep1 = 0,sep2 = 2, sep3 = 3, sep4 = 5, sep5 = 6; //señalamos los leds apagados  //introducimos el ciclo for que se encarga de llenar la matriz  while(i >=0 && i<8){  //hacemos los ciclos que van a crear las filas  for(int a=0; a<2; a++){  for(int j=0; j<8; j++){  if(j==sep2 || j==sep4){  matriz[i][j]=low;  }else{  matriz[i][j]=high;  }  }  i += 1;  }  for(int a=0;a<2;a++){  for(int j=0; j<8; j++){  if(j==sep1 || j==sep3 || j==sep5){  matriz[i][j]=low;  }else{  matriz[i][j]=high;  }  }  i += 1;  }  }  //imprimimos la matriz  for(int i=0; i<8; i++){  for(int j=0;j<8;j++){  char posicion= matriz[i][j];  char espacio= ' ';  cout<< espacio<< posicion;  }  cout<<endl;  }  } |
| --- |

**Código para el cuarto patrón:**

| char high='+', low='-';  char matriz[8][8];  int cotainferior=0, cotasuperior=4;  //primera parte de la figura  for(int i=0; i<5; i++){  for(int j=0;j<8;j++){  if(j>=cotainferior and j<cotasuperior){  for(int control2=j; control2<cotasuperior;control2++){  matriz[i][j]=high;}  }else{  matriz[i][j]=low;  }  }  if(i<3){  cotainferior+=1;  cotasuperior+=1;  }  }  //segunda parte de la figura  cotainferior=2; cotasuperior=6;  for(int i=5; i<8; i++){  for(int j=0;j<8;j++){  if(j>=cotainferior and j<cotasuperior){  for(int control2=j; control2<cotasuperior;control2++){  matriz[i][j]=high;}  }else{  matriz[i][j]=low;  }  }  cotainferior-=1;  cotasuperior-=1;  }  //Imprimir la matriz  for(int i=0; i<8; i++){  for(int j=0;j<8;j++){  char posicion= matriz[i][j];  char espacio= ' ';  cout<< espacio<< posicion;  }  cout<<endl;  } |
| --- |

1. ***Problemas de desarrollo que afrontó***

La primera piedra con la que tropezamos durante el desarrollo del trabajo fue el aprender a manejar adecuadamente el sistema de Arduino y del integrado 74HC595. El tener en cuenta el funcionamiento de las entradas seriales y salidas en paralelo, el cómo recibir y tratar esa información para lograr el resultado deseado y el cómo extrapolar lo que ya sabíamos a esta aplicación, y los demás aspectos electrónicos involucrados (como programar los voltajes que soportan los leds).

Preguntas formuladas en el proceso:

1. ¿Qué ventajas tiene la memoria dinámica ante el stack?
2. ¿Para qué su usa el Git-ignore?
3. ¿Cuál es el formato de entrada de los datos para los LED’S?
4. ¿Cómo usar correctamente el integrado 74HC595?

Acuerdos:

1. Trabajar de 08:00 -09:00 am y de 08:00 pm-12:00 am en el archivo todos los días
2. Subir el segundo commit a las 11:50 pm todos los días
3. Realizaremos el vídeo el viernes 22 de septiembre.

1. ***Evolución de la solución y consideraciones a tener en cuenta en la implementación.***

Vamos a utilizar un Arduino y un integrado electrónico conocido como 74HC595, que nos va a permitir generar datos en paralelo a partir de datos en serie. De esta forma, podemos generar ocho salidas con sólo tres pines digitales del Arduino. No obstante, dado el tamaño de la matriz, se debe recurrir a una definición electrónica de la información y también a la concatenación de otro 74HC595 al anterior (cosa que es posible por un lujo del mismo 74HC595). De esta manera, podríamos manejar prácticamente de forma simultánea a cada led en la matriz, facilitando los problemas de impresión consecutiva y de imagen a mostrar.

Entonces, mediante el serial se ingresan los primeros ocho bits que van a representar las columnas de la matriz (estos ocho bits saldrán por el segundo 74HC595), y sus salidas estarán conectadas a los ánodos de los leds en la matriz, y los siguientes ocho bits ingresados por el serial serán los que representan las filas de la matriz (estos ocho bits saldrán por el primer 74HC595) y sus salidas estarán conectadas a los cátodos de la matriz, de manera que si un valor es alto, toda la fila/columna se enciende, y si ambos valores son altos, toda la fila/columna se apaga. Por decirlo de alguna forma, se van a apagar todos los componentes que tengan ambos valores altos, y se encenderán todos los componentes que tengan sólo un valor alto.

Enlace al proyecto en tinkercard: <https://www.tinkercad.com/things/81HGCE5FQIT?sharecode=mrxWwzQoLmXpCdjmZE6az5yd21304jRjNfOSm302SNQ>

Cosas por hacer:

1. Creación de la matriz de LEDS
2. Creación de la función verificación que encienda y apague los 64 LEDS cada cierto tiempo
3. Creación de la función imagen que recibe ordenes e imprime una figura indicada
4. Imprimir los patrones

**Análisis en papel**

**Ejercicios 1 y 4**

