“MANEJO DE ESTRUCTURAS Y SUBRUTINAS EN ARDUINO”

Trabajo Preparatorio N°4
Laboratorio de Sistemas Embebidos

Melanny Cecibel Dávila Pazmiño
Ingeniería en Telecomunicaciones
Facultad de Eléctrica y Electrónica
Quito, Ecuador
melanny.davila@epn.edu.ec

***Abstract*—En el presente trabajo preparatorio se tratará temas acerca de las estructuras y subrutinas que pueden ser utilizadas en Arduino con el fin de comprender su uso e implementarlas. De esta manera se reducirá en gran manera las líneas de código utilizadas, mejorando así la eficiencia del código.**

***Index Terms*—Arduino, estructura, bucle, subrutina.**

I. Introducción

Al momento de programar se puede llegar a repetir mucho una estructura de código o en algunos casos se podría necesitar la misma estructura de código para otra aplicación, por esta razón se ha creado la opción de tener subrutinas dentro del mismo código. Además, para poder ejecutar muchas aplicaciones es necesario comparar o repetir fragmentos de código N veces por lo que se han creado también estructuras iterativas y comparativas.

II. Objetivos

- Relacionar al estudiante con el uso y manejo de las diferentes estructuras y subrutinas de control.
- Diseñar e implementar segmentos de código que permitan al estudiante explorar esquemas de automatización

III. Cuestionario

A. Definir que son las y estructuras de control repetitivas o iterativas en Arduino. Para cada una de estas estructuras consultar sus comandos, colocar los parámetros y objetos necesarios para su uso.

- **Estructuras de control condicionales** Son estructuras usadas para realizar comparaciones entre dos valores de cualquier tipo, de manera que si se cumple la condición que se encuentra dentro de su argumento se ejecute el código dentro de esta estructura o, en el caso de que no se cumpla la condición, no se ejecute el fragmento de código.
  - **if:** Comprueba si hay una condición y ejecuta la instrucción o conjunto de instrucciones en curso si la condición es ’verdadera’.

```
if(condición para ser evaluada)
{
// Instrucciones
}
```

Fig. 1. if

  - **else:** Si la condición presentada en if, no se cumple se utiliza este comando con el fin de ejecutar otro conjunto de instrucciones; este paso puede ser opcional.

```
else
{
        // Instrucciones
}
```

Fig. 2. else

  - **switch:** Sirve para evaluar múltiples opciones y dependiendo del valor de entrada se ejecutará una instrucción u otra. El uso de esta estructura de control condicional implica menor retardo en comparación al uso del anidamiento de condicionales if y else.

```
switch (expresión) {
          case valor1:
          //instrucciones
                    [break]
          case valor2:
          // instrucciones
                    [break]
.
.
.
        default:
                //instrucciones por defecto
}
```

Fig. 3. switch

– **if-else:** Se puede anidar ambos comandos con el fin de declarar condiciones if dentro de otros condicionales if.

```
if(condición para ser evaluada)
  {
  // Instrucciones
    if(condición para ser evaluada)
     {
        // Instrucciones
     }
    else {
                // Instrucciones
     }
  }
else {
        // Instrucciones
   if(condición para ser evaluada)
    {
       // Instrucciones
    }
   else
    {
        // Instrucciones
    }
}
```

Fig. 4. if-else

- **Estructuras de control repetitivas** Son estructuras usadas para repetir el fragmento de código que se encuentra de ellas un número de veces dado por el propio lazo que puede ser un número determinado o hasta que se deje de cumplir una condición dada dentro del argumento de estas estructuras.
  – **while:** agrupa un conjunto de instrucciones que solo se ejecutan si se cumple la condición inicial.

```
while (condición) {
     // instrucciones
}
```

Fig. 5. while

  – **do while:** Su funcionamiento es similar a while, su principal diferencia radica en que la comparación es realizada al final, es decir, por lo menos las intrucciones se ejecutan una vez.

```
do {
     // Instrucciones
} while (condición)
```

Fig. 6. do while

  – **for:** es una estructura de control iterativa mediante el uso de una variable auxiliar. Se evalúa una condición y en base a esto la variable incrementa o decrementa su valor.

```
for( inicialización de la variable; condición a ser evaluada; incremento/decremento)
{
        // Instrucciones que se ejecutaran en caso que cumpla la condición.
}
```

Fig. 7. for

B. Definir en sus propias palabras que son las Funciones y Subrutinas, su declaración y uso dentro de un sketch de Arduino.

- **Funciones:** Son comandos asociados a las funciones eléctricas de la placa capaces de cambiar la forma en la que se recibe o envía información o a la vez agregar retardos a las señales que se manipulan. Se declaran escribiendo su nombre y en el caso de ser necesario escribiendo los argumentos que necesitan estas funciones para poder trabajar y su uso más común puede ser el agregar retrasos entre instrucciones o leer pines o enviar un tipo de señal a través de ellos.

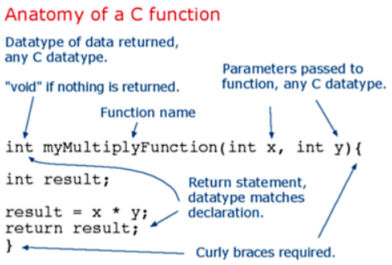

Fig. 8. Estructura de una función

- **Subrutinas:** Son pequeños códigos que generalmente se usan mas de una vez dentro de uno o varios programas de manera que se definieron para tener un código reutilizable y fácil de llamar en el caso de que se necesite. Estas pueden tener varias salidas y varias entradas, todo depende de lo que el usuario necesite, su código se coloca debajo del principal y se llaman como si fueran funciones.

C. En una tienda de abarrotes en la cual se tienen 6 parqueaderos disponibles se ha implementado un circuito de control cuyo funcionamiento se detalla a continuación:

- Al ingresar al parqueadero se presenta un display (BCD 7 segmentos) en el cual se muestra el número de plazas de parqueo disponibles.
- En cada plaza de parqueo existe una fotocelda la cual al ocuparse ese espacio (se interrumpe el paso de luz), emite una señal que permite reducir el número de lugares disponibles en el display colocado al ingreso del parqueadero.

A continuación se presenta el circuito diseñado.

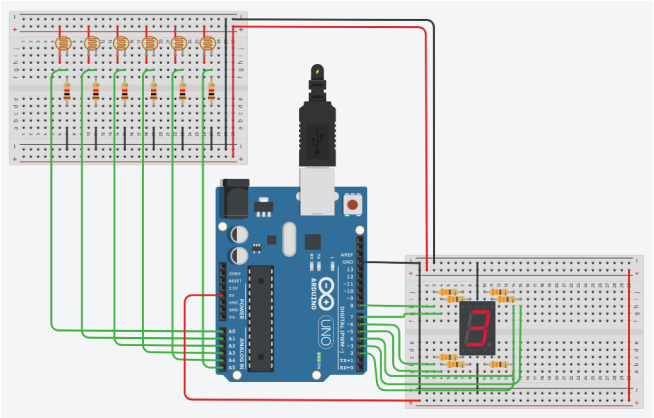

Fig. 9. Esquemático implementado

```
/* Trabajo Preparatorio Nro. 4
Funcionamiento de un parqueadero
Variables para determinar el estado de los parqueaderos
1 desocupado, 0 ocupado */

//Declaracion de las variables
int P1;
int P2;
int P3;
int P4;
int P5;
int P6;
// Lectura del valor del pin analogico
float D1;
float D2;
float D3;
float D4;
float D5;
float D6;
int tiempo=1000; //Variable de tiempo

int NTotal;// numero de parqueaderos disponibles

void setup(){
  // pines de salida para display
  DDRD=B11111110;
  pinMode(8,OUTPUT);
  // inicio del puerto serial
  Serial.begin(9600);

 }

// Funcion para mostrar el numero en el display
void display(int valor){
  //puerto 0 y 1 libre por activacion del puerto serial
  //puerto 1 al 7 display
  // vector de numeros para el display
  byte numero[7]={B11111101,B00011000,B01101110,B00111110,B10011010,B10110110,B11110110};
  PORTD=numero[valor];
  if (valor==0){
    digitalWrite(8,LOW);
  }else if (valor==1){
    digitalWrite(8,LOW);
  }else if (valor==2){
    digitalWrite(8,HIGH);
  }else if (valor==3){
    digitalWrite(8,HIGH);
  }else if (valor==4){
    digitalWrite(8,HIGH);
  }else if (valor==5){
    digitalWrite(8,HIGH);
  }else if (valor==6){
    digitalWrite(8,HIGH);
  }
}
// Funcion para determinar la ocupacion del lugar dependiendo de la fotoresistencia
int ocupacion (float valor){
 int estado; // 1 desocupado 0 ocupado
   if (valor>350){
  estado=1;
  }else{
  estado=0;
 }
return estado;
}


 void loop()  {

   // Lectura del valor analogico de la fotocelda
   D1=analogRead(A0);
   D2=analogRead(A1);
   D3=analogRead(A2);
   D4=analogRead(A3);
   D5=analogRead(A4);
   D6=analogRead(A5);
   //Lectura de los estados de la fotocelda
   P1=ocupacion(D1);
   P2=ocupacion(D2);
   P3=ocupacion(D3);
   P4=ocupacion(D4);
   P5=ocupacion(D5);
   P6=ocupacion(D6);
   // contar lugares disponibles
   NTotal=P1+P2+P3+P4+P5+P6;
   //Mostra los lugares disponibles
   display(NTotal);


   delay(tiempo); // mostrar resultados cada segundo
 }
```

REFERENCES

[1] "Aprendiendo Arduino Funciones", Descubrearduino.com, feb. 20, 2019. https://descubrearduino.com/funciones/ (accedido dic. 17, 2020).

[2] E. Espinosa, E. Tatayo, "MANEJO DE ESTRUCTURAS Y SUBRUTINAS EN ARDUINO". C.P. SISTEMAS EMBEBIDOS, Accedido: dic. 15, 2020. [En línea].

[3] "Estructuras de control en Arduino", Creatividad Codificada, sep. 05, 2019. https://creatividadcodificada.com/arduino/estructuras-de-control-en-arduino/ (accedido dic. 15, 2020).

[4] "Funciones definidas por usuario", Aprendiendo Arduino, nov. 16, 2016. https://aprendiendoarduino.wordpress.com/2016/11/16/funciones-definidas-por-usuario-2/ (accedido dic. 17, 2020).