CONTESTA EL SIGUENTE CUESTIONARIO:

1.- ¿Para qué sirve el Void setup()?

- A) Para mandar llamar las librerías necesarias para la utilización del programa.
- B) Para leer datos, llamar a funciones y seguir ejecutando el programa indefinidamente.
- C) Para inicializar las variables que se utilizarán durante la utilización del programa y asignarles un espacio en la memoria del Arduino.

2.- ¿Para qué sirve el Void loop()?

- A) Para mandar llamar las librerías necesarias para la utilización del programa.
- B) Es un bucle infinito que permite leer datos, llamar a funciones, modificar las variables, etc.
- C) Inicializa las variables del programa y les asigna un valor.

3.- ¿De qué se conforma una clase en Arduino?

- A) Se conforma de métodos, los cuales contienen atributos que puedes modificar conforme a tus necesidades.
- B) Se conforma de atributos, necesarios para establecer a un objeto
- C) Se conforma de atributos, con los cuales se definen los métodos, necesarios para la creación de objetos y la modificación de sus parámetros.

4.- ¿Qué es un atributo?

- A) Son parámetros que definen a una clase conforme a sus características.
- B) Son parámetros que definen a una clase conforme a sus métodos.
- C) Son parámetros que definen a una clase conforme a sus funcionalidades.

5.- ¿Qué es un método?

- A) Son parámetros que pueden ser utilizados por la clase, conforme a sus características.
- B) Son parámetros que se utilizan para definir el uso de las variables establecidas.
- C) Son parámetros que pueden ser utilizados por la clase, conforme a sus funcionalidades.

6.- ¿Cuál de los siguientes incisos contiene únicamente atributos?

- A) GetNombre(), SetNombre(), GetEdad()
- B) Nombre, Edad, Sexo
- C) Set.Edad, Get.Sexo, Set.Sexo

7.- ¿Cuál de los siguientes incisos contiene únicamente métodos?

- A) GetNombre(), GetSexo(), GetEdad()
- B) Set.Edad, Get.Sexo, Set.Sexo
- C) Setedad(), Getedad(), Setnombre()

8.- ¿Qué es el polimorfismo?

- A) Una interfaz única con un solo método.
- B) Varias interfaces con múltiples métodos.
- C) Una interfaz única con múltiples métodos.

9.- ¿Qué es la herencia?

- A) Proceso por el cual un objeto hereda las propiedades y métodos de otro, sin necesidad de volverlas a definir desde el principio.
- B) Proceso por el cual un objeto hereda las clases de otro, sin necesidad de volverlas a definir desde el principio.
- C) Proceso por el cual un objeto hereda las propiedades y métodos de otro, teniendo que volver a definirlas.

10.- ¿Cuál es la diferencia entre los modificadores de acceso Private y Public?

- A) Private permite la heredación entre clases y el acceso a todos sus datos, Public no.
- B) Private y Public permiten el acceso a todos sus datos.
- C) Public permite la heredación entre clases y el acceso a todos sus dato, Private no.

SECCIÓN 1
*1: encender y apagar el led 1.

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);}
int Led;
bool estado;
bool PrenderLed(int prendido) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;
void loop() {
Serial.println(estado);
estado=PrenderLed(Led = 1);  delay(1000); exit(0); }
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
pinMode(1, OUTPUT);
}
void loop() {
  digitalWrite(1, HIGH);
  Serial.println("Led encendido");
  delay (1000);
  digitalWrite(1, LOW);
  Serial.println("led apagado");
  delay(1000);
 delay(1000); exit(0);
}
```

C).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);}
int Led;
bool estado;
bool PrenderLed(int prendido) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;
void loop() {
Serial.println(estado);
estado=PrenderLed(Led = 1);  delay(1000); exit(0); }
```

D).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(Led_1,ON); delay(1000);
```

```
MFS.writeLeds(Led_1,0FF); delay(1000);
}
```

*2: prender el led tres y luego el dos, apagar el tres y apagar el dos

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
  Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
#define ON  1
#define OFF 0
class Led{
 private:
 bool estado;
 int pin;
 String etiqueta;
public:
  Led(int _pin = 13,bool _estado = OFF,String _etiqueta = "no definido");
  bool GetEstado();
  void SetEstado(bool _estado);
  int GetPin();
  void SetPin(int _pin);
  String GetEtiqueta();
  void SetEtiqueta(String _etiqueta);
  void Encender();
  void Apagar();

};
Led::Led(int _pin,bool _estado, String _etiqueta){
estado = _estado; pin = _pin; etiqueta= _etiqueta;
pinMode(pin, OUTPUT);
digitalWrite(pin, estado);
}
  bool Led::GetEstado()     {return estado;}
  void Led::SetEstado(bool _estado) {estado = _estado;}
  int Led::GetPin() { return pin;}
  void Led::SetPin(int _pin) {pin = _pin;}
  String Led::GetEtiqueta() {return etiqueta;}
  void Led::SetEtiqueta(String _etiqueta) {etiqueta = _etiqueta;}
  void Led::Encender() {digitalWrite(pin, ON); estado = ON;}
  void Led::Apagar(){digitalWrite(pin, OFF); estado = OFF;}

  Led Led3(3,OFF);
  Led Led2(2,OFF);
  bool bandera = true;
void loop() {
  Led3.Encender();
```

```
  Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(100);
Led2.Encender();
  Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(100);
  Led3.Apagar();
 Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(100);
Led2.Apagar();
  Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(100);
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
  Serial.begin(9600);
}
#define ON 1
#define OFF 0
   int     LedR = 2;
   int     LedZ= 3;
  void loop() {
    MFS.writeLeds();
  ++LedZ;
    delay(1000);
    ++LedR;
    delay(1000);
   --LedZ;
   delay(1000);
    --LedR;
  }
```

C).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_3, ON); delay(1000);
MFS.writeLeds(LED_2, ON); delay(1000);
MFS.writeLeds(LED_3, OFF);delay(1000);
MFS.writeLeds(LED_2, OFF);delay(1000);
}
```

D).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
int Led3=3, Led2=2;
}
void loop() {
if ((valor == 1) Encender() && else Apagar()){
 Led3.Encender(); Led2.Encender();
}
else if ((valor == 2) Encender() && else Apagar()){
Led3.Apagar(); Led2.Apagar();
}
}
```

*3: encender y apagar el led 3.

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
int Led=3;
}
void loop() {
MFS.writeLed(3,ON); delay(1000);  MFS.writeLed(3,0FF); delay(1000); }
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_3, ON); Serial.println("Led 3 encendido"); delay(1000);
MFS.writeLeds(LED_3, OFF); Serial.println("Led 3 apagado"); delay(1000);
}
```

C).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
```

```
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);}
int Led;
bool estado;
bool PrenderLed(int prendido) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;
void loop() {
Serial.println(estado);
estado=PrenderLed(Led = 3);  delay(1000); exit(0); }
```

D).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
pinMode(3, OUTPUT);
}
void loop() {
  digitalWrite(3, HIGH);
  Serial.println("Led encendido");
  delay (1000);
  digitalWrite(3, LOW);
  Serial.println("led apagado");
  delay(1000);
 delay(1000); exit(0);
}
```

*4: prender el led uno y luego el dos, apagar el uno y apagar el dos

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
  Serial.begin(9600);
}
#define ON 1
#define OFF 0
   int     LedR = 1;
   int     LedZ= 2;
  void loop() {
    MFS.writeLeds();
  ++LedR;
```

```
    delay(1000);
    ++LedZ;
    delay(1000);
   --LedR;
   delay(1000);
    --LedZ;
  }
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_1, ON); delay(1000);
MFS.writeLeds(LED_2, ON); delay(1000);
MFS.writeLeds(LED_1, OFF);delay(1000);
MFS.writeLeds(LED_2, OFF);delay(1000);
}
```

C).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
int Led1=1, Led2=2;
}
void loop() {
if ((valor == 1) Encender() && else Apagar()){
 Led1.Encender(); Led2.Encender();
}
else if ((valor == 2) Encender() && else Apagar()){
Led1.Apagar(); Led2.Apagar();
}
}
```

D).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
  Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
#define ON  1
#define OFF 0
```

```
class Led{
 private:
 bool estado;
 int pin;
 String etiqueta;
public:
  Led(int _pin = 13,bool _estado = OFF,String _etiqueta = "no definido");
  bool GetEstado();
  void SetEstado(bool _estado);
  int GetPin();
  void SetPin(int _pin);
  String GetEtiqueta();
  void SetEtiqueta(String _etiqueta);
  void Encender();
  void Apagar();

};
Led::Led(int _pin,bool _estado, String _etiqueta){
estado = _estado; pin = _pin; etiqueta= _etiqueta;
pinMode(pin, OUTPUT);
digitalWrite(pin, estado);
}
  bool Led::GetEstado()     {return estado;}
  void Led::SetEstado(bool _estado) {estado = _estado;}
  int Led::GetPin() { return pin;}
  void Led::SetPin(int _pin) {pin = _pin;}
  String Led::GetEtiqueta() {return etiqueta;}
  void Led::SetEtiqueta(String _etiqueta) {etiqueta = _etiqueta;}
  void Led::Encender() {digitalWrite(pin, ON); estado = ON;}
  void Led::Apagar(){digitalWrite(pin, OFF); estado = OFF;}

  Led Led1(1,OFF);
  Led Led2(2,OFF);
  bool bandera = true;
void loop() {
  Led1.Encender();
  Serial.print( Led1.GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(100);
Led2.Encender();
  Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(100);
  Led1.Apagar();
  Serial.print( Led1.GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(100);
  Led2.Apagar();
 Serial.print( Led2GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(100);
```

*5: prender el led uno y luego el dos, apagar el dos y apagar el uno

A).-

```
#include <TimerOne.h>
```

```
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
  Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
#define ON  1
#define OFF 0
class Led{
 private:
 bool estado;
 int pin;
 String etiqueta;
public:
  Led(int _pin = 13,bool _estado = OFF,String _etiqueta = "no definido");
  bool GetEstado();
  void SetEstado(bool _estado);
  int GetPin();
  void SetPin(int _pin);
  String GetEtiqueta();
  void SetEtiqueta(String _etiqueta);
  void Encender();
  void Apagar();

};
Led::Led(int _pin,bool _estado, String _etiqueta){
estado = _estado; pin = _pin; etiqueta= _etiqueta;
pinMode(pin, OUTPUT);
digitalWrite(pin, estado);
}
  bool Led::GetEstado()     {return estado;}
  void Led::SetEstado(bool _estado) {estado = _estado;}
  int Led::GetPin() { return pin;}
  void Led::SetPin(int _pin) {pin = _pin;}
  String Led::GetEtiqueta() {return etiqueta;}
  void Led::SetEtiqueta(String _etiqueta) {etiqueta = _etiqueta;}
  void Led::Encender() {digitalWrite(pin, ON); estado = ON;}
  void Led::Apagar(){digitalWrite(pin, OFF); estado = OFF;}

  Led Led1(1,OFF);
  Led Led2(2,OFF);
  bool bandera = true;
void loop() {
  Led1.Encender();
  Serial.print( Led1.GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(100);
Led2.Encender();
  Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(100);
  Led2.Apagar();
  Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
```

```
  delay(100);
  Led1.Apagar();
 Serial.print( Led1GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(100);
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
  Serial.begin(9600);
}
#define ON 1
#define OFF 0
   int     LedR = 1;
   int     LedZ= 2;
  void loop() {
    MFS.writeLeds();
  ++LedR;
    delay(1000);
    ++LedZ;
    delay(1000);
   --LedZ;
   delay(1000);
    --LedR;
  }
```

C).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_1, ON); delay(1000);
MFS.writeLeds(LED_2, ON); delay(1000);
MFS.writeLeds(LED_2, OFF);delay(1000);
MFS.writeLeds(LED_1, OFF);delay(1000);
}
```

D).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
int Led1=1, Led2=2;
}
```

```
void loop() {
if ((valor == 1) Encender() && else Apagar()){
 Led1.Encender(); Led2.Encender();
}
else if ((valor == 2) Encender() && else Apagar()){
Led2.Apagar(); Led1.Apagar();
}
}
```

*6: encender y apagar el led 4.

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_4, ON); Serial.println("Led 4 encendido"); delay(1000);
MFS.writeLeds(LED_4, OFF); Serial.println("Led 4 apagado"); delay(1000);
}
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
pinMode(4, OUTPUT);
}
void loop() {
  digitalWrite(4, HIGH);
  Serial.println("Led encendido");
  delay (1000);
  digitalWrite(4, LOW);
  Serial.println("led apagado");
  delay(1000);
 delay(1000); exit(0);
}
```

C).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
```

```
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);}
int Led;
bool estado;
bool PrenderLed(int prendido) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;
void loop() {
Serial.println(estado); estado=PrenderLed(Led = 4);  delay(1000); exit(0); }
```

D).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
int Led=4;
}
void loop() {
MFS.writeLed(4,ON); delay(1000);
MFS.writeLed(4,0FF); delay(1000);
}
```

*7: prender el led dos y luego el uno, apagar el dos y apagar el uno

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
int Led1=1, Led2=2;
}
void loop() {
if ((valor == 1) Encender() && else Apagar()){
 Led2.Encender(); Led1.Encender();
}
else if ((valor == 2) Encender() && else Apagar()){
Led2.Apagar(); Led1.Apagar();
}
}
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
```

```
#include <MultiFuncShield.h>
void setup() {
  Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
#define ON  1
#define OFF 0
class Led{
 private:
 bool estado;
 int pin;
 String etiqueta;
public:
  Led(int _pin = 13,bool _estado = OFF,String _etiqueta = "no definido");
  bool GetEstado();
  void SetEstado(bool _estado);
  int GetPin();
  void SetPin(int _pin);
  String GetEtiqueta();
  void SetEtiqueta(String _etiqueta);
  void Encender();
  void Apagar();

};
Led::Led(int _pin,bool _estado, String _etiqueta){
estado = _estado; pin = _pin; etiqueta= _etiqueta;
pinMode(pin, OUTPUT);
digitalWrite(pin, estado);
}
  bool Led::GetEstado()     {return estado;}
  void Led::SetEstado(bool _estado) {estado = _estado;}
  int Led::GetPin() { return pin;}
  void Led::SetPin(int _pin) {pin = _pin;}
  String Led::GetEtiqueta() {return etiqueta;}
  void Led::SetEtiqueta(String _etiqueta) {etiqueta = _etiqueta;}
  void Led::Encender() {digitalWrite(pin, ON); estado = ON;}
  void Led::Apagar(){digitalWrite(pin, OFF); estado = OFF;}

  Led Led1(1,OFF);
  Led Led2(2,OFF);
  bool bandera = true;
void loop() {
  Led2.Encender();
  Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(100);
Led1.Encender();
  Serial.print( Led1.GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(100);
  Led2.Apagar();
  Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(100);
```

```
  Led1.Apagar();
 Serial.print( Led1.GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(100);
```

C).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
  Serial.begin(9600);
}
#define ON 1
#define OFF 0
   int     LedR = 2;
   int     LedZ= 1;
  void loop() {
    MFS.writeLeds();
  ++LedR;
    delay(1000);
    ++LedZ;
   delay(1000);
   --LedR;
   delay(1000);
    --LedZ;
  }
```

D).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_2, ON); delay(1000);
MFS.writeLeds(LED_1, ON); delay(1000);
MFS.writeLeds(LED_2, OFF);delay(1000);
MFS.writeLeds(LED_1, OFF);delay(1000);
}
```

*8: prender el led dos y luego el tres, apagar el dos y apagar el tres

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
int Led3=3, Led2=2;
```

```
}
void loop() {
if ((valor == 1) Encender() && else Apagar()){
 Led2.Encender(); Led3.Encender();
}
else if ((valor == 2) Encender() && else Apagar()){
Led2.Apagar(); Led3.Apagar();
}
}
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_2, ON); delay(1000);
MFS.writeLeds(LED_3, ON); delay(1000);
MFS.writeLeds(LED_2, OFF);delay(1000);
MFS.writeLeds(LED_3, OFF);delay(1000);
}
```

C).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
  Serial.begin(9600);
}
#define ON 1
#define OFF 0
   int     LedR = 2;
   int     LedZ= 3;
  void loop() {
    MFS.writeLeds();
  ++LedR;
    delay(1000);
    ++LedZ;
    delay(1000);
   --LedR;
   delay(1000);
    --LedZ;
  }
```

D).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
```

```
  Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
#define ON  1
#define OFF 0
class Led{
 private:
 bool estado;
 int pin;
 String etiqueta;
public:
  Led(int _pin = 13,bool _estado = OFF,String _etiqueta = "no definido");
  bool GetEstado();
  void SetEstado(bool _estado);
  int GetPin();
  void SetPin(int _pin);
  String GetEtiqueta();
  void SetEtiqueta(String _etiqueta);
  void Encender();
  void Apagar();

};
Led::Led(int _pin,bool _estado, String _etiqueta){
estado = _estado; pin = _pin; etiqueta= _etiqueta;
pinMode(pin, OUTPUT);
digitalWrite(pin, estado);
}
  bool Led::GetEstado()     {return estado;}
  void Led::SetEstado(bool _estado) {estado = _estado;}
  int Led::GetPin() { return pin;}
  void Led::SetPin(int _pin) {pin = _pin;}
  String Led::GetEtiqueta() {return etiqueta;}
  void Led::SetEtiqueta(String _etiqueta) {etiqueta = _etiqueta;}
  void Led::Encender() {digitalWrite(pin, ON); estado = ON;}
  void Led::Apagar(){digitalWrite(pin, OFF); estado = OFF;}

  Led Led3(3,OFF);
  Led Led2(2,OFF);
  bool bandera = true;
void loop() {
  Led2.Encender();
  Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(100);
Led3.Encender();
  Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(100);
  Led2.Apagar();
  Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(100);
  Led3.Apagar();
 Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
```

```
  delay(100);
```

*9: prender el led tres y luego el dos, apagar el dos y apagar el tres

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_3, ON); delay(1000);
MFS.writeLeds(LED_2, ON); delay(1000);
MFS.writeLeds(LED_2, OFF);delay(1000);
MFS.writeLeds(LED_3, OFF);delay(1000);
}
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
  Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
#define ON  1
#define OFF 0
class Led{
 private:
 bool estado;
 int pin;
 String etiqueta;
public:
  Led(int _pin = 13,bool _estado = OFF,String _etiqueta = "no definido");
  bool GetEstado();
  void SetEstado(bool _estado);
  int GetPin();
  void SetPin(int _pin);
  String GetEtiqueta();
  void SetEtiqueta(String _etiqueta);
  void Encender();
  void Apagar();

};
Led::Led(int _pin,bool _estado, String _etiqueta){
estado = _estado; pin = _pin; etiqueta= _etiqueta;
pinMode(pin, OUTPUT);
digitalWrite(pin, estado);
}
```

```
  bool Led::GetEstado()     {return estado;}
  void Led::SetEstado(bool _estado) {estado = _estado;}
  int Led::GetPin() { return pin;}
  void Led::SetPin(int _pin) {pin = _pin;}
  String Led::GetEtiqueta() {return etiqueta;}
  void Led::SetEtiqueta(String _etiqueta) {etiqueta = _etiqueta;}
  void Led::Encender() {digitalWrite(pin, ON); estado = ON;}
  void Led::Apagar(){digitalWrite(pin, OFF); estado = OFF;}

  Led Led3(3,OFF);
  Led Led2(2,OFF);
  bool bandera = true;
void loop() {
  Led3.Encender();
  Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(100);
Led2.Encender();
  Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(100);
  Led2.Apagar();
  Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(100);
  Led3.Apagar();
 Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(100);

C).-
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
int Led3=3, Led2=2;
}
void loop() {
if ((valor == 1) Encender() && else Apagar()){
 Led3.Encender(); Led2.Encender();
}
else if ((valor == 2) Encender() && else Apagar()){
Led2.Apagar(); Led3.Apagar();
}
} D).-
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
  Serial.begin(9600);
}
#define ON 1
#define OFF 0
   int     LedR = 2;
```

```
  int     LedZ= 3;
 void loop() {
   MFS.writeLeds();
 ++LedZ;
   delay(1000);
   ++LedR;
   delay(1000);
  --LedR;
  delay(1000);
   --LedZ;
 }
```

*10: encender y apagar el led 2.

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);}
int Led;
bool estado;
bool PrenderLed(int prendido) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;
void loop() {
Serial.println(estado);
estado=PrenderLed(Led = 2);
delay(1000); exit(0); }
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
int Led=2;
}
void loop() {
MFS.writeLed(2,ON); delay(1000);
MFS.writeLed(2,0FF); delay(1000);
}
```

C).-

```
#include <TimerOne.h>
```

```
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_2, ON); Serial.println("Led 2 encendido"); delay(1000);
MFS.writeLeds(LED_2, OFF); Serial.println("Led 2 apagado"); delay(1000);
}
```

D).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
pinMode(2, OUTPUT);
}
void loop() {
  digitalWrite(2, HIGH);
  Serial.println("Led encendido");
  delay (1000);
  digitalWrite(2, LOW);
  Serial.println("led apagado");
  delay(1000);
delay(1000); exit(0);
}
```

*11: prender el led tres y luego el cuatro, apagar el tres y apagar el cuatro

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
int Led4=4, Led3=3;
}
void loop() {
if ((valor == 1) Encender() && else Apagar()){
 Led3.Encender(); Led4.Encender();
}
else if ((valor == 2) Encender() && else Apagar()){
Led3.Apagar(); Led4.Apagar();
}
}
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
  Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
#define ON  1
#define OFF 0
class Led{
 private:
 bool estado;
 int pin;
 String etiqueta;
public:
  Led(int _pin = 13,bool _estado = OFF,String _etiqueta = "no definido");
  bool GetEstado();
  void SetEstado(bool _estado);
  int GetPin();
  void SetPin(int _pin);
  String GetEtiqueta();
  void SetEtiqueta(String _etiqueta);
  void Encender();
  void Apagar();

};
Led::Led(int _pin,bool _estado, String _etiqueta){
estado = _estado; pin = _pin; etiqueta= _etiqueta;
pinMode(pin, OUTPUT);
digitalWrite(pin, estado);
}
  bool Led::GetEstado()     {return estado;}
  void Led::SetEstado(bool _estado) {estado = _estado;}
  int Led::GetPin() { return pin;}
  void Led::SetPin(int _pin) {pin = _pin;}
  String Led::GetEtiqueta() {return etiqueta;}
  void Led::SetEtiqueta(String _etiqueta) {etiqueta = _etiqueta;}
  void Led::Encender() {digitalWrite(pin, ON); estado = ON;}
  void Led::Apagar(){digitalWrite(pin, OFF); estado = OFF;}

  Led Led3(3,OFF);
  Led Led4(4,OFF);
  bool bandera = true;
void loop() {
Led3.Encender();
  Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(100);
Led4.Encender();
  Serial.print( Led4.GetEtiqueta() ); Serial.println(Led4.GetEstado() );
  delay(100);
Led3.Apagar();
```

```
 Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(100);
 Led4.Apagar();
 Serial.print( Led4.GetEtiqueta() ); Serial.println(Led4.GetEstado() );
  delay(100);
```

C).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
  Serial.begin(9600);
}
#define ON 1
#define OFF 0
   int     LedR = 3;
   int     LedZ= 4;
  void loop() {
    MFS.writeLeds();
  ++LedR;
    delay(1000);
    ++LedZ;
    delay(1000);
   --LedR;
   delay(1000);
    --LedZ;
  }
```

D).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_3, ON); delay(1000);
MFS.writeLeds(LED_4, ON); delay(1000);
MFS.writeLeds(LED_3, OFF);delay(1000);
MFS.writeLeds(LED_4, OFF);delay(1000);
}
```

*12: prender el led dos y luego el cuatro, apagar el dos y apagar el cuatro

A).-

```
#include <TimerOne.h>
#include <Wire.h>
```

```
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_2, ON); delay(1000);
MFS.writeLeds(LED_4, ON); delay(1000);
MFS.writeLeds(LED_2, OFF);delay(1000);
MFS.writeLeds(LED_4, OFF);delay(1000);
}
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
int Led4=4, Led2=2;
}
void loop() {
if ((valor == 1) Encender() && else Apagar()){
 Led2.Encender(); Led4.Encender();
}
else if ((valor == 2) Encender() && else Apagar()){
Led2.Apagar(); Led4.Apagar();
}
}
```

C).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
  Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
#define ON  1
#define OFF 0
class Led{
 private:
 bool estado;
 int pin;
 String etiqueta;
public:
  Led(int _pin = 13,bool _estado = OFF,String _etiqueta = "no definido");
  bool GetEstado();
  void SetEstado(bool _estado);
  int GetPin();
  void SetPin(int _pin);
```

```
  String GetEtiqueta();
  void SetEtiqueta(String _etiqueta);
  void Encender();
  void Apagar();

};
Led::Led(int _pin,bool _estado, String _etiqueta){
estado = _estado; pin = _pin; etiqueta= _etiqueta;
pinMode(pin, OUTPUT);
digitalWrite(pin, estado);
}
  bool Led::GetEstado()     {return estado;}
  void Led::SetEstado(bool _estado) {estado = _estado;}
  int Led::GetPin() { return pin;}
  void Led::SetPin(int _pin) {pin = _pin;}
  String Led::GetEtiqueta() {return etiqueta;}
  void Led::SetEtiqueta(String _etiqueta) {etiqueta = _etiqueta;}
  void Led::Encender() {digitalWrite(pin, ON); estado = ON;}
  void Led::Apagar(){digitalWrite(pin, OFF); estado = OFF;}

  Led Led2(2,OFF);
  Led Led4(4,OFF);
  bool bandera = true;
void loop() {
Led2.Encender();
  Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(100);
Led4.Encender();
  Serial.print( Led4.GetEtiqueta() ); Serial.println(Led4.GetEstado() );
  delay(100);
Led2.Apagar();
 Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(100);
 Led4.Apagar();
 Serial.print( Led4.GetEtiqueta() ); Serial.println(Led4.GetEstado() );
  delay(100);


D).-
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
  Serial.begin(9600);
}
#define ON 1
#define OFF 0
   int     LedR = 2;
   int     LedZ= 4;
  void loop() {
    MFS.writeLeds();
```

```
  ++LedR;
    delay(1000);
    ++LedZ;
    delay(1000);
   --LedR;
   delay(1000);
    --LedZ;
  }
```

*13: prender el led tres y luego el cuatro, apagar el cuatro y apagar el tres

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
  Serial.begin(9600);
}
#define ON 1
#define OFF 0
   int     LedR = 3;
   int     LedZ= 4;
  void loop() {
    MFS.writeLeds();
  ++LedR;
    delay(1000);
    ++LedZ;
    delay(1000);
   --LedZ;
   delay(1000);
    --LedR;
  }
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_3, ON); delay(1000);
MFS.writeLeds(LED_4, ON); delay(1000);
MFS.writeLeds(LED_4, OFF);delay(1000);
MFS.writeLeds(LED_3, OFF);delay(1000);
}
```

C).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
int Led4=4, Led3=3;
}
void loop() {
if ((valor == 1) Encender() && else Apagar()){
 Led3.Encender(); Led4.Encender();
}
else if ((valor == 2) Encender() && else Apagar()){
Led4.Apagar(); Led3.Apagar();
}
}
```

D).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
  Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
#define ON  1
#define OFF 0
class Led{
 private:
 bool estado;
 int pin;
 String etiqueta;
public:
  Led(int _pin = 13,bool _estado = OFF,String _etiqueta = "no definido");
  bool GetEstado();
  void SetEstado(bool _estado);
  int GetPin();
  void SetPin(int _pin);
  String GetEtiqueta();
  void SetEtiqueta(String _etiqueta);
  void Encender();
  void Apagar();

};
Led::Led(int _pin,bool _estado, String _etiqueta){
estado = _estado; pin = _pin; etiqueta= _etiqueta;
pinMode(pin, OUTPUT);
digitalWrite(pin, estado);
}
  bool Led::GetEstado()     {return estado;}
  void Led::SetEstado(bool _estado) {estado = _estado;}
  int Led::GetPin() { return pin;}
```

```
  void Led::SetPin(int _pin) {pin = _pin;}
  String Led::GetEtiqueta() {return etiqueta;}
  void Led::SetEtiqueta(String _etiqueta) {etiqueta = _etiqueta;}
  void Led::Encender() {digitalWrite(pin, ON); estado = ON;}
  void Led::Apagar(){digitalWrite(pin, OFF); estado = OFF;}

  Led Led3(3,OFF);
  Led Led4(4,OFF);
  bool bandera = true;
void loop() {
Led3.Encender();
  Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(100);
Led4.Encender();
  Serial.print( Led4.GetEtiqueta() ); Serial.println(Led4.GetEstado() );
  delay(100);
Led4.Apagar();
 Serial.print( Led4.GetEtiqueta() ); Serial.println(Led4.GetEstado() );
  delay(100);
Led3.Apagar();
 Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(100);
 Led4.Apagar();
 Serial.print( Led4.GetEtiqueta() ); Serial.println(Led4.GetEstado() );
  delay(100);
```

*14: prender el led uno y luego el cuatro, apagar el cuatro y apagar el uno

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
  Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
#define ON  1
#define OFF 0
class Led{
 private:
 bool estado;
 int pin;
 String etiqueta;
public:
  Led(int _pin = 13,bool _estado = OFF,String _etiqueta = "no definido");
  bool GetEstado();
  void SetEstado(bool _estado);
  int GetPin();
  void SetPin(int _pin);
  String GetEtiqueta();
  void SetEtiqueta(String _etiqueta);
```

```
  void Encender();
  void Apagar();

};
Led::Led(int _pin,bool _estado, String _etiqueta){
estado = _estado; pin = _pin; etiqueta= _etiqueta;
pinMode(pin, OUTPUT);
digitalWrite(pin, estado);
}
  bool Led::GetEstado()     {return estado;}
  void Led::SetEstado(bool _estado) {estado = _estado;}
  int Led::GetPin() { return pin;}
  void Led::SetPin(int _pin) {pin = _pin;}
  String Led::GetEtiqueta() {return etiqueta;}
  void Led::SetEtiqueta(String _etiqueta) {etiqueta = _etiqueta;}
  void Led::Encender() {digitalWrite(pin, ON); estado = ON;}
  void Led::Apagar(){digitalWrite(pin, OFF); estado = OFF;}

  Led Led4(4,OFF);
  Led Led1(1,OFF);
  bool bandera = true;
void loop() {
Led1.Encender();
  Serial.print( Led1.GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(100);
Led4.Encender();
  Serial.print( Led4.GetEtiqueta() ); Serial.println(Led4.GetEstado() );
  delay(100);
Led4.Apagar();
 Serial.print( Led4.GetEtiqueta() ); Serial.println(Led4.GetEstado() );
  delay(100);
Led1.Apagar();
 Serial.print( Led1.GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(100);

B).-
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
  Serial.begin(9600);
}
#define ON 1
#define OFF 0
   int     LedR = 1;
   int     LedZ= 4;
  void loop() {
    MFS.writeLeds();
  ++LedR;
    delay(1000);
    ++LedZ;
```

```
    delay(1000);
   --LedZ;
   delay(1000);
    --LedR;
  }
```

C).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_1, ON); delay(1000);
MFS.writeLeds(LED_4, ON); delay(1000);
MFS.writeLeds(LED_4, OFF);delay(1000);
MFS.writeLeds(LED_1, OFF);delay(1000);
}
```

D).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
int Led4=4, Led1=1;
}
void loop() {
if ((valor == 1) Encender() && else Apagar()){
 Led1.Encender(); Led4.Encender();
}
else if ((valor == 2) Encender() && else Apagar()){
Led4.Apagar(); Led1.Apagar();
}
}
```

*15.- prender el led 1, durar mucho tiempo, después medio, y finalmente muy rápido (3 veces rápido)

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
  Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
#define ON  1
#define OFF 0
```

```
class Led{
 private:
 bool estado;
 int pin;
 String etiqueta;
public:
  Led(int _pin = 13,bool _estado = OFF,String _etiqueta = "no definido");
  bool GetEstado();
  void SetEstado(bool _estado);
  int GetPin();
  void SetPin(int _pin);
  String GetEtiqueta();
  void SetEtiqueta(String _etiqueta);
  void Encender();
  void Apagar();

};
Led::Led(int _pin,bool _estado, String _etiqueta){
estado = _estado; pin = _pin; etiqueta= _etiqueta;
pinMode(pin, OUTPUT);
digitalWrite(pin, estado);
}
  bool Led::GetEstado()     {return estado;}
  void Led::SetEstado(bool _estado) {estado = _estado;}
  int Led::GetPin() { return pin;}
  void Led::SetPin(int _pin) {pin = _pin;}
  String Led::GetEtiqueta() {return etiqueta;}
  void Led::SetEtiqueta(String _etiqueta) {etiqueta = _etiqueta;}
  void Led::Encender() {digitalWrite(pin, ON); estado = ON;}
  void Led::Apagar(){digitalWrite(pin, OFF); estado = OFF;}

  Led Led1(1,OFF);
  bool bandera = true;
void loop() {
Led1.Encender();
  Serial.print( Led1.GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(1000);
Led1.Apagar();
 Serial.print( Led1.GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(1000);
Led1.Encender();
  Serial.print( Led1.GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(500);
Led1.Apagar();
 Serial.print( Led1.GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(500);
Led1.Encender();
  Serial.print( Led1.GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(100);
Led1.Apagar();
 Serial.print( Led1.GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(100);
```

```
Led1.Encender();
  Serial.print( Led1.GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(100);
Led1.Apagar();
 Serial.print( Led1.GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(100);
Led1.Encender();
  Serial.print( Led1.GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(100);
Led1.Apagar();
 Serial.print( Led1.GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(100);
}
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_1, ON); Serial.println("Led 1 encendido lento"); delay(1000);
MFS.writeLeds(LED_1, OFF); Serial.println("Led 1 apagado lento"); delay(1000);
MFS.writeLeds(LED_1, ON); Serial.println("Led 1 encendido medio"); delay(500);
MFS.writeLeds(LED_1, OFF); Serial.println("Led 1 apagado medio"); delay(500);
MFS.writeLeds(LED_1, ON); Serial.println("Led 1 encendido rápido"); delay(100);
MFS.writeLeds(LED_1, OFF); Serial.println("Led 1 apagado rápido"); delay(100);
MFS.writeLeds(LED_1, ON); Serial.println("Led 1 encendido rápido"); delay(100);
MFS.writeLeds(LED_1, OFF); Serial.println("Led 1 apagado rápido"); delay(100);
MFS.writeLeds(LED_1, ON); Serial.println("Led 1 encendido rápido"); delay(100);
MFS.writeLeds(LED_1, OFF); Serial.println("Led 1 apagado rápido"); delay(100);
}
```

*16: prender el led uno y luego el cuatro, apagar el uno y apagar el cuatro

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
  Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
#define ON  1
#define OFF 0
class Led{
 private:
 bool estado;
```

```
 int pin;
 String etiqueta;
public:
  Led(int _pin = 13,bool _estado = OFF,String _etiqueta = "no definido");
  bool GetEstado();
  void SetEstado(bool _estado);
  int GetPin();
  void SetPin(int _pin);
  String GetEtiqueta();
  void SetEtiqueta(String _etiqueta);
  void Encender();
  void Apagar();

};
Led::Led(int _pin,bool _estado, String _etiqueta){
estado = _estado; pin = _pin; etiqueta= _etiqueta;
pinMode(pin, OUTPUT);
digitalWrite(pin, estado);
}
  bool Led::GetEstado()     {return estado;}
  void Led::SetEstado(bool _estado) {estado = _estado;}
  int Led::GetPin() { return pin;}
  void Led::SetPin(int _pin) {pin = _pin;}
  String Led::GetEtiqueta() {return etiqueta;}
  void Led::SetEtiqueta(String _etiqueta) {etiqueta = _etiqueta;}
  void Led::Encender() {digitalWrite(pin, ON); estado = ON;}
  void Led::Apagar(){digitalWrite(pin, OFF); estado = OFF;}

  Led Led4(4,OFF);
  Led Led1(1,OFF);
  bool bandera = true;
void loop() {
Led1.Encender();
  Serial.print( Led1.GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(100);
Led4.Encender();
  Serial.print( Led4.GetEtiqueta() ); Serial.println(Led4.GetEstado() );
  delay(100);
Led1.Apagar();
 Serial.print( Led1.GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(100);
 Led4.Apagar();
 Serial.print( Led4.GetEtiqueta() ); Serial.println(Led4.GetEstado() );
  delay(100);


B).-
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
```

```
  Serial.begin(9600);
}
#define ON 1
#define OFF 0
   int     LedR = 1;
   int     LedZ= 4;
  void loop() {
    MFS.writeLeds();
  ++LedR;
    delay(1000);
    ++LedZ;
    delay(1000);
   --LedR;
   delay(1000);
    --LedZ;
  }
```

C).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_1, ON); delay(1000);
MFS.writeLeds(LED_4, ON); delay(1000);
MFS.writeLeds(LED_1, OFF);delay(1000);
MFS.writeLeds(LED_4, OFF);delay(1000);
}
```

D).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
int Led4=4, Led1=1;
}
void loop() {
if ((valor == 1) Encender() && else Apagar()){
 Led1.Encender(); Led4.Encender();
}
else if ((valor == 2) Encender() && else Apagar()){
Led1.Apagar(); Led4.Apagar();
}
}
```

*17.- prender el led 2, durar mucho tiempo, después medio, y finalmente muy rápido

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
  Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
#define ON  1
#define OFF 0
class Led{
 private:
 bool estado;
 int pin;
 String etiqueta;
public:
  Led(int _pin = 13,bool _estado = OFF,String _etiqueta = "no definido");
  bool GetEstado();
  void SetEstado(bool _estado);
  int GetPin();
  void SetPin(int _pin);
  String GetEtiqueta();
  void SetEtiqueta(String _etiqueta);
  void Encender();
  void Apagar();

};
Led::Led(int _pin,bool _estado, String _etiqueta){
estado = _estado; pin = _pin; etiqueta= _etiqueta;
pinMode(pin, OUTPUT);
digitalWrite(pin, estado);
}
  bool Led::GetEstado()     {return estado;}
  void Led::SetEstado(bool _estado) {estado = _estado;}
  int Led::GetPin() { return pin;}
  void Led::SetPin(int _pin) {pin = _pin;}
  String Led::GetEtiqueta() {return etiqueta;}
  void Led::SetEtiqueta(String _etiqueta) {etiqueta = _etiqueta;}
  void Led::Encender() {digitalWrite(pin, ON); estado = ON;}
  void Led::Apagar(){digitalWrite(pin, OFF); estado = OFF;}

  Led Led2(2,OFF);
  bool bandera = true;
void loop() {
Led2.Encender();
  Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(1000);
Led2.Apagar();
Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(1000);
```

```
Led2.Encender();
 Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(500);
Led2.Apagar();
Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(500);
Led2.Encender();
 Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(100);
Led2.Apagar();
Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(100);
Led2.Encender();
 Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(100);
Led2.Apagar();
Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(100);
Led2.Encender();
 Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(100);
Led2.Apagar();
Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );;
  delay(100);
}
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_2, ON); Serial.println("Led 2 encendido lento"); delay(1000);
MFS.writeLeds(LED_2, OFF); Serial.println("Led 2 apagado lento"); delay(1000);
MFS.writeLeds(LED_2, ON); Serial.println("Led 2 encendido medio"); delay(500);
MFS.writeLeds(LED_2, OFF); Serial.println("Led 2 apagado medio"); delay(500);
MFS.writeLeds(LED_2, ON); Serial.println("Led 2 encendido rápido"); delay(100);
MFS.writeLeds(LED_2, OFF); Serial.println("Led 2 apagado rápido"); delay(100);
MFS.writeLeds(LED_2, ON); Serial.println("Led 2 encendido rápido"); delay(100);
MFS.writeLeds(LED_2, OFF); Serial.println("Led 2 apagado rápido"); delay(100);
MFS.writeLeds(LED_2, ON); Serial.println("Led 2 encendido rápido"); delay(100);
MFS.writeLeds(LED_2, OFF); Serial.println("Led 2 apagado rápido"); delay(100);
}
```

*18: prender el led uno y luego el tres, apagar el uno y apagar el tres

A).-

```
#include <TimerOne.h>
```

```
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
  Serial.begin(9600);
}
#define ON 1
#define OFF 0
   int     LedR = 1;
   int     LedZ= 3;
  void loop() {
    MFS.writeLeds();
  ++LedR;
    delay(1000);
    ++LedZ;
    delay(1000);
   --LedR;
   delay(1000);
    --LedZ;
  }
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
  Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
#define ON  1
#define OFF 0
class Led{
 private:
 bool estado;
 int pin;
 String etiqueta;
public:
  Led(int _pin = 13,bool _estado = OFF,String _etiqueta = "no definido");
  bool GetEstado();
  void SetEstado(bool _estado);
  int GetPin();
  void SetPin(int _pin);
  String GetEtiqueta();
  void SetEtiqueta(String _etiqueta);
  void Encender();
  void Apagar();

};
Led::Led(int _pin,bool _estado, String _etiqueta){
estado = _estado; pin = _pin; etiqueta= _etiqueta;
pinMode(pin, OUTPUT);
```

```
digitalWrite(pin, estado);
}
  bool Led::GetEstado()     {return estado;}
  void Led::SetEstado(bool _estado) {estado = _estado;}
  int Led::GetPin() { return pin;}
  void Led::SetPin(int _pin) {pin = _pin;}
  String Led::GetEtiqueta() {return etiqueta;}
  void Led::SetEtiqueta(String _etiqueta) {etiqueta = _etiqueta;}
  void Led::Encender() {digitalWrite(pin, ON); estado = ON;}
  void Led::Apagar(){digitalWrite(pin, OFF); estado = OFF;}

  Led Led3(3,OFF);
  Led Led1(1,OFF);
  bool bandera = true;
void loop() {
Led1.Encender();
  Serial.print( Led1.GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(100);
Led3.Encender();
  Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(100);
Led1.Apagar();
 Serial.print( Led1.GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(100);
 Led3.Apagar();
 Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(100);
```

C).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_1, ON); delay(1000);
MFS.writeLeds(LED_2, ON); delay(1000);
MFS.writeLeds(LED_1, OFF);delay(1000);
MFS.writeLeds(LED_3, OFF);delay(1000);
}
```

D).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
```

```
void setup() {
int Led3=3, Led1=1;
}
void loop() {
if ((valor == 1) Encender() && else Apagar()){
 Led1.Encender(); Led2.Encender();
}
else if ((valor == 2) Encender() && else Apagar()){
Led1.Apagar(); Led3.Apagar();
}
}
```

*19: prender el led dos y luego el cuatro, apagar el cuatro y apagar el dos

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
  Serial.begin(9600);
}
#define ON 1
#define OFF 0
   int     LedR = 2;
   int     LedZ= 4;
  void loop() {
    MFS.writeLeds();
  ++LedR;
    delay(1000);
    ++LedZ;
    delay(1000);
   --LedZ;
   delay(1000);
    --LedR;
  }
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_2, ON); delay(1000);
MFS.writeLeds(LED_4, ON); delay(1000);
MFS.writeLeds(LED_4, OFF);delay(1000);
MFS.writeLeds(LED_2, OFF);delay(1000);
```

```
}

C).-
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
  Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
#define ON  1
#define OFF 0
class Led{
 private:
 bool estado;
 int pin;
 String etiqueta;
public:
  Led(int _pin = 13,bool _estado = OFF,String _etiqueta = "no definido");
  bool GetEstado();
  void SetEstado(bool _estado);
  int GetPin();
  void SetPin(int _pin);
  String GetEtiqueta();
  void SetEtiqueta(String _etiqueta);
  void Encender();
  void Apagar();

};
Led::Led(int _pin,bool _estado, String _etiqueta){
estado = _estado; pin = _pin; etiqueta= _etiqueta;
pinMode(pin, OUTPUT);
digitalWrite(pin, estado);
}
  bool Led::GetEstado()     {return estado;}
  void Led::SetEstado(bool _estado) {estado = _estado;}
  int Led::GetPin() { return pin;}
  void Led::SetPin(int _pin) {pin = _pin;}
  String Led::GetEtiqueta() {return etiqueta;}
  void Led::SetEtiqueta(String _etiqueta) {etiqueta = _etiqueta;}
  void Led::Encender() {digitalWrite(pin, ON); estado = ON;}
  void Led::Apagar(){digitalWrite(pin, OFF); estado = OFF;}

  Led Led2(2,OFF);
  Led Led4(4,OFF);
  bool bandera = true;
void loop() {
Led2.Encender();
  Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(100);
Led4.Encender();
```

```
  Serial.print( Led4.GetEtiqueta() ); Serial.println(Led4.GetEstado() );
  delay(100);
Led4.Apagar();
 Serial.print( Led4.GetEtiqueta() ); Serial.println(Led4.GetEstado() );
  delay(100);
Led2.Apagar();
 Serial.print( Led2.GetEtiqueta() ); Serial.println(Led2.GetEstado() );
  delay(100);
```

D).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
int Led4=4, Led2=2;
}
void loop() {
if ((valor == 1) Encender() && else Apagar()){
 Led2.Encender(); Led4.Encender();
}
else if ((valor == 2) Encender() && else Apagar()){
Led4.Apagar(); Led2.Apagar();
}
}
```

*20: prender el led uno y luego el tres, apagar el tres y apagar el uno

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
  Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
#define ON  1
#define OFF 0
class Led{
 private:
 bool estado;
 int pin;
 String etiqueta;
public:
  Led(int _pin = 13,bool _estado = OFF,String _etiqueta = "no definido");
  bool GetEstado();
  void SetEstado(bool _estado);
  int GetPin();
  void SetPin(int _pin);
  String GetEtiqueta();
  void SetEtiqueta(String _etiqueta);
```

```
  void Encender();
  void Apagar();

};
Led::Led(int _pin,bool _estado, String _etiqueta){
estado = _estado; pin = _pin; etiqueta= _etiqueta;
pinMode(pin, OUTPUT);
digitalWrite(pin, estado);
}
  bool Led::GetEstado()     {return estado;}
  void Led::SetEstado(bool _estado) {estado = _estado;}
  int Led::GetPin() { return pin;}
  void Led::SetPin(int _pin) {pin = _pin;}
  String Led::GetEtiqueta() {return etiqueta;}
  void Led::SetEtiqueta(String _etiqueta) {etiqueta = _etiqueta;}
  void Led::Encender() {digitalWrite(pin, ON); estado = ON;}
  void Led::Apagar(){digitalWrite(pin, OFF); estado = OFF;}

  Led Led3(3,OFF);
  Led Led1(1,OFF);
  bool bandera = true;
void loop() {
Led1.Encender();
  Serial.print( Led1.GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(100);
Led3.Encender();
  Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(100);
  Led3.Apagar();
 Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(100);
Led1.Apagar();
 Serial.print( Led1.GetEtiqueta() ); Serial.println(Led1.GetEstado() );
  delay(100);

B).-
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
  Serial.begin(9600);
}
#define ON 1
#define OFF 0
   int     LedR = 1;
   int     LedZ= 3;
  void loop() {
    MFS.writeLeds();
  ++LedR;
    delay(1000);
    ++LedZ;
```

```
    delay(1000);
   --LedZ;
   delay(1000);
    --LedR;
  }
```

C).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_1, ON); delay(1000);
MFS.writeLeds(LED_3, ON); delay(1000);
MFS.writeLeds(LED_3, OFF);delay(1000);
MFS.writeLeds(LED_1, OFF);delay(1000);
}
```

D).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
int Led3=3, Led1=1;
}
void loop() {
if ((valor == 1) Encender() && else Apagar()){
 Led1.Encender(); Led2.Encender();
}
else if ((valor == 2) Encender() && else Apagar()){
Led3.Apagar(); Led1.Apagar();
}
}
```

SECCIÓN 2

*1.- prender el led 3, durar mucho tiempo, después medio, y finalmente muy rápido

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
  Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
```

```
#define ON  1
#define OFF 0
class Led{
 private:
 bool estado;
 int pin;
 String etiqueta;
public:
  Led(int _pin = 13,bool _estado = OFF,String _etiqueta = "no definido");
  bool GetEstado();
  void SetEstado(bool _estado);
  int GetPin();
  void SetPin(int _pin);
  String GetEtiqueta();
  void SetEtiqueta(String _etiqueta);
  void Encender();
  void Apagar();

};
Led::Led(int _pin,bool _estado, String _etiqueta){
estado = _estado; pin = _pin; etiqueta= _etiqueta;
pinMode(pin, OUTPUT);
digitalWrite(pin, estado);
}
  bool Led::GetEstado()     {return estado;}
  void Led::SetEstado(bool _estado) {estado = _estado;}
  int Led::GetPin() { return pin;}
  void Led::SetPin(int _pin) {pin = _pin;}
  String Led::GetEtiqueta() {return etiqueta;}
  void Led::SetEtiqueta(String _etiqueta) {etiqueta = _etiqueta;}
  void Led::Encender() {digitalWrite(pin, ON); estado = ON;}
  void Led::Apagar(){digitalWrite(pin, OFF); estado = OFF;}

  Led Led3(3,OFF);
  bool bandera = true;
void loop() {
Led3.Encender();
  Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(1000);
Led3.Apagar();
Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(1000);
Led3.Encender();
Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(500);
Led3.Apagar();
Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(500);
Led3.Encender();
Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(100);
Led3.Apagar();
```

```
Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() ); delay(100);
Led3.Encender();
Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(100);
Led3.Apagar();
Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(100);
Led3.Encender();
 Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() ); delay(100);
Led3.Apagar();
Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(100);
}
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_3, ON); Serial.println("Led 3 encendido lento"); delay(1000);
MFS.writeLeds(LED_3, OFF); Serial.println("Led 3 apagado lento"); delay(1000);
MFS.writeLeds(LED_3, ON); Serial.println("Led 3 encendido medio"); delay(500);
MFS.writeLeds(LED_3, OFF); Serial.println("Led 3 apagado medio"); delay(500);
MFS.writeLeds(LED_3, ON); Serial.println("Led 3 encendido rápido"); delay(100);
MFS.writeLeds(LED_3, OFF); Serial.println("Led 3 apagado rápido"); delay(100);
MFS.writeLeds(LED_3, ON); Serial.println("Led 3 encendido rápido"); delay(100);
MFS.writeLeds(LED_3, OFF); Serial.println("Led 3 apagado rápido"); delay(100);
MFS.writeLeds(LED_3, ON); Serial.println("Led 3 encendido rápido"); delay(100);
MFS.writeLeds(LED_3, OFF); Serial.println("Led 3 apagado rápido"); delay(100); }
```

*2.- encender el led 1 y que suene un bucer, y después apagar

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_1, ON); MFS.beep(); delay(1000);
MFS.writeLeds(LED_1, OFF); delay(1000);
}
```

B).-

```
#include <TimerOne.h>
```

```
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
pinMode(1, OUTPUT);
}
void loop() {
  digitalWrite(1, HIGH); beep(5,2,3,6);
  delay (1000);
  digitalWrite(1, LOW);
  delay(1000);
}
```

*3.- prender el led 4, durar mucho tiempo, después medio, y finalmente muy rápido

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_4, ON); Serial.println("Led 4 encendido lento"); delay(1000);
MFS.writeLeds(LED_4, OFF); Serial.println("Led 4 apagado lento"); delay(1000);
MFS.writeLeds(LED_4, ON); Serial.println("Led 4 encendido medio"); delay(500);
MFS.writeLeds(LED_4, OFF); Serial.println("Led 4 apagado medio"); delay(500);
MFS.writeLeds(LED_4, ON); Serial.println("Led 4 encendido rápido"); delay(100);
MFS.writeLeds(LED_4, OFF); Serial.println("Led 4 apagado rápido"); delay(100);
MFS.writeLeds(LED_4, ON); Serial.println("Led 4 encendido rápido"); delay(100);
MFS.writeLeds(LED_4, OFF); Serial.println("Led 4 apagado rápido"); delay(100);
MFS.writeLeds(LED_4, ON); Serial.println("Led 4 encendido rápido"); delay(100);
MFS.writeLeds(LED_4, OFF); Serial.println("Led 4 apagado rápido"); delay(100); }
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
  Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
#define ON  1
#define OFF 0
class Led{
 private:
 bool estado;
```

```
 int pin;
 String etiqueta;
public:
  Led(int _pin = 13,bool _estado = OFF,String _etiqueta = "no definido");
  bool GetEstado();
  void SetEstado(bool _estado);
  int GetPin();
  void SetPin(int _pin);
  String GetEtiqueta();
  void SetEtiqueta(String _etiqueta);
  void Encender();
  void Apagar();

};
Led::Led(int _pin,bool _estado, String _etiqueta){
estado = _estado; pin = _pin; etiqueta= _etiqueta;
pinMode(pin, OUTPUT);
digitalWrite(pin, estado);
}
  bool Led::GetEstado()     {return estado;}
  void Led::SetEstado(bool _estado) {estado = _estado;}
  int Led::GetPin() { return pin;}
  void Led::SetPin(int _pin) {pin = _pin;}
  String Led::GetEtiqueta() {return etiqueta;}
  void Led::SetEtiqueta(String _etiqueta) {etiqueta = _etiqueta;}
  void Led::Encender() {digitalWrite(pin, ON); estado = ON;}
  void Led::Apagar(){digitalWrite(pin, OFF); estado = OFF;}

  Led Led4(4,OFF);
  bool bandera = true;
void loop() {
Led4.Encender();
  Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(1000);
Led4.Apagar();
Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(1000);
Led4.Encender();
Serial.print( Led3.GetEtiqueta() ); Serial.println(Led3.GetEstado() );
  delay(500);
Led4.Apagar();
Serial.print( Led4.GetEtiqueta() ); Serial.println(Led4.GetEstado() );
  delay(500);
Led4.Encender();
Serial.print( Led4.GetEtiqueta() ); Serial.println(Led4.GetEstado() );
  delay(100);
Led4.Apagar();
Serial.print( Led4.GetEtiqueta() ); Serial.println(Led4.GetEstado() ); delay(100);
Led4.Encender();
Serial.print( Led4.GetEtiqueta() ); Serial.println(Led4.GetEstado() );
  delay(100);
Led4.Apagar();
```

```
Serial.print( Led4.GetEtiqueta() ); Serial.println(Led4.GetEstado() );
  delay(100);
Led4.Encender();
 Serial.print( Led4.GetEtiqueta() ); Serial.println(Led4.GetEstado() ); delay(100);
Led4.Apagar();
Serial.print( Led4.GetEtiqueta() ); Serial.println(Led4.GetEstado() );
  delay(100);
}
```

*4.- encender el led 2 y que suene un bucer, y después apagar

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
pinMode(2, OUTPUT);
}
void loop() {
  digitalWrite(2, HIGH); beep(5,2,3,6);
  delay (1000);
  digitalWrite(2, LOW);
  delay(1000);
}
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_2, ON); MFS.beep(); delay(1000);
MFS.writeLeds(LED_2, OFF); delay(1000);
}
```

*5.- parpadeo rápido de los cuatro leds

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
```

```
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);}
int Led;
bool estado;
bool PrenderLed(int prendido) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;
void loop() {
Serial.println(estado);
estado=PrenderLed(Led = 1);  estado=PrenderLed(Led = 2);
estado=PrenderLed(Led = 3);  estado=PrenderLed(Led = 4);
delay(1000); exit(0); }
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_1, ON);MFS.writeLeds(LED_2, ON); MFS.writeLeds(LED_3, ON);
MFS.writeLeds(LED_4, ON);delay(100);
MFS.writeLeds(LED_1, OFF); MFS.writeLeds(LED_2, OFF); MFS.writeLeds(LED_3, OFF);
MFS.writeLeds(LED_4, OFF); delay(100); }
```

*6.- encender el led 3 y que suene un bucer, y después apagar

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
pinMode(3, OUTPUT);
}
void loop() {
  digitalWrite(3, HIGH); beep(5,2,3,6);
  delay (1000);
  digitalWrite(3, LOW);
  delay(1000);
```

```
}
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_3, ON); MFS.beep(); delay(1000);
MFS.writeLeds(LED_3, OFF); delay(1000); }
```

*7.- semaforo

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);}
int Led;
bool estado;
bool PrenderLed(int prendido) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;
void loop() {
Serial.println(estado);
estado=PrenderLed(Led = 1);  delay(1000);
estado=PrenderLed(Led = 2);  delay(1000);
estado=PrenderLed(Led = 3);  delay(1000); }
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
pinMode(1, OUTPUT); pinMode(2, OUTPUT); pinMode(3, OUTPUT);
}
void loop() {
  digitalWrite(1, HIGH);
```

```
  Serial.println(“VERDE PRENDIDO");
  delay (1000);
  digitalWrite(1, LOW);
  Serial.println(“VERDE APAGADO");
  delay(1000);
digitalWrite(2, HIGH);
  Serial.println(“AMARILLO PRENDIDO");
  delay (1000);
  digitalWrite(2, LOW);
  Serial.println(“AMARILLO APAGADO");
  delay(1000);
digitalWrite(3, HIGH);
  Serial.println(“ROJO PRENDIDO");
  delay (1000);
  digitalWrite(3, LOW);
  Serial.println(“ROJO APAGADO");
  delay(1000);
 delay(1000); exit(0);
}
```

C).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_1, ON);delay(2000);
MFS.writeLeds(LED_1, OFF);delay(10);
MFS.writeLeds(LED_2, ON);delay(1000);
MFS.writeLeds(LED_2, OFF);delay(10);
MFS.writeLeds(LED_2, ON);delay(100);
MFS.writeLeds(LED_2, OFF);delay(10);
MFS.writeLeds(LED_2, ON);delay(50);
MFS.writeLeds(LED_2, OFF);delay(100);
MFS.writeLeds(LED_2, ON);delay(10);
MFS.writeLeds(LED_2, OFF);delay(10);
MFS.writeLeds(LED_2, ON);delay(100);
MFS.writeLeds(LED_2, OFF);delay(100);
MFS.writeLeds(LED_3, ON);delay(2000);
MFS.writeLeds(LED_3, OFF);delay(100);
}
```

D).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
```

```
Timer1.initialize();
MFS.initialize(&Timer1);
int Led=1; int Led=2; int Led=3;
}
void loop() {
MFS.writeLed(1,ON); delay(1000);
MFS.writeLed(1,0FF); delay(1000);
MFS.writeLed(2,ON); delay(1000);
MFS.writeLed(2,0FF); delay(1000);
MFS.writeLed(3,ON); delay(1000);
MFS.writeLed(3,0FF); delay(1000);
}
```

*8.- encender el led 4 y que suene un bucer, y después apagar
A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_4, ON); MFS.beep(); delay(1000);
MFS.writeLeds(LED_4, OFF); delay(1000);
}
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
pinMode(1, OUTPUT);
}
void loop() {
  digitalWrite(4, HIGH); beep(5,2,3,6);
  delay (1000);
  digitalWrite(4, LOW);
  delay(1000);
}
```

*9.- secuencia de dos en dos
prender y apagar el 1-4-2-3 en ese orden

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooLedV2.h>
```

```
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);}
int Led;
bool estado;
bool PrenderLed(int prendido) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;
void loop() {
Serial.println(estado);
estado=PrenderLed(Led = 1);  delay(1000);
estado=PrenderLed(Led = 4);  delay(1000);
estado=PrenderLed(Led = 2);  delay(1000);
estado=PrenderLed(Led = 3);  delay(1000);}
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
void loop() {
MFS.writeLeds(LED_1, ON); delay(1000);
MFS.writeLeds(LED_4, ON); delay(1000);
MFS.writeLeds(LED_2, ON); delay(1000);
MFS.writeLeds(LED_3, ON); delay(1000);
MFS.writeLeds(LED_3, OFF);delay(1000);
MFS.writeLeds(LED_2, OFF);delay(1000);
MFS.writeLeds(LED_4, OFF);delay(1000);
MFS.writeLeds(LED_1, OFF);delay(1000);
}
```

C).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
int Led=1; int Led=2; int Led=3;
}
void loop() {
MFS.writeLed(1,ON); delay(1000);
MFS.writeLed(1,0FF); delay(1000);
```

```
MFS.writeLed(4,ON); delay(1000);
MFS.writeLed(4,0FF); delay(1000);
MFS.writeLed(2,ON); delay(1000);
MFS.writeLed(2,0FF); delay(1000);
MFS.writeLed(3,ON); delay(1000);
MFS.writeLed(3,0FF); delay(1000);
}
```

*10.- encender leds con el potenciometro
1 entre 0 y 125
2 entre 126 y 300
3 entre 301 y 700
4 entre 701 y 1000
prenderlos todos y luego apagarlos entre 1001 y 1023
A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <PooLed.V1.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
int Pot=0;
void loop() {
if(Led_1 && Pot <=125) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;
void loop() {
Serial.println(estado);
estado=PrenderLed(Led = 1);  estado=ApagarLed(Led = 2);  estado=ApagarLed(Led = 3);
estado=ApagarLed(Led = 4); delay(1000); }
 else if(Led_2 && Pot <=300) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;

Serial.println(estado);
estado=PrenderLed(Led = 2); estado=ApagarLed(Led = 1);  estado=ApagarLed(Led = 3);
estado=ApagarLed(Led = 4);  delay(1000); }
else if(Led_3 && Pot <=700) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;
```

```
Serial.println(estado);
estado=PrenderLed(Led = 3); estado=ApagarLed(Led = 1);  estado=ApagarLed(Led = 2);
estado=ApagarLed(Led = 4);  delay(1000); }

else if(Led_4 && Pot <=1000) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;

Serial.println(estado);
estado=PrenderLed(Led = 4); estado=ApagarLed(Led = 1);  estado=ApagarLed(Led = 2);
estado=ApagarLed(Led = 3);  delay(1000); }
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);

}
int valorPot=0;
void loop() {
valorPot = analogRead(POT_PIN);
Serial.print("Valor: ");
Serial.println(POT_PIN); brillo();
}
void brillo() {
   MFS.write(analogRead(POT_PIN));
  if(valorPot >= 5 && valorPot <=125) {
MFS.writeLeds(LED_1, ON);
MFS.writeLeds(LED_2, OFF);
MFS.writeLeds(LED_3, OFF);
MFS.writeLeds(LED_4, OFF);
}
else if(valorPot >=126 && valorPot <=300) {
MFS.writeLeds(LED_1, OFF);
MFS.writeLeds(LED_2, ON);
MFS.writeLeds(LED_3, OFF);
MFS.writeLeds(LED_4, OFF);
}
else if(valorPot >=301 && valorPot <=700) {
MFS.writeLeds(LED_1, OFF);
MFS.writeLeds(LED_2, OFF);
MFS.writeLeds(LED_3, ON);
MFS.writeLeds(LED_4, OFF);
}
```

```
else if(valorPot >=701 && valorPot <=1000) {
MFS.writeLeds(LED_1, OFF);
MFS.writeLeds(LED_2, OFF);
MFS.writeLeds(LED_3, OFF);
MFS.writeLeds(LED_4, ON);
}
else if(valorPot >=1001 && valorPot <=1009) {
MFS.writeLeds(LED_1, ON);
MFS.writeLeds(LED_2, ON);
MFS.writeLeds(LED_3, ON);
MFS.writeLeds(LED_4, ON);
}
else if(valorPot >=1010 && valorPot <=1023) {
MFS.writeLeds(LED_1, OFF);
MFS.writeLeds(LED_2, OFF);
MFS.writeLeds(LED_3, OFF);
MFS.writeLeds(LED_4, OFF);
}
}
```

11.- ¿por qué no aparece nada en el display?

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooClaseContadorV2.h>
void setup() {
Timer1.initialize();
MFS.initialize(&Timer1);
Serial.begin(9600);
}
Contador Boton1(0, 0, 10), Boton2(0, 0, 10), Boton3(1, 1, 10) ;
 ID;
int i=0;

void loop() {

byte btn = MFS.getButton();
 if (btn) {
byte buttonNumber = btn & B00111111;
    byte buttonAction  = btn & B11000000;
          if (buttonAction == BUTTON_PRESSED_IND){
      if(buttonNumber == 1){
        for(int i = 1; i <= ID; i++){
          Boton1.Incrementar();
        }
      }
      else if(buttonNumber == 2){
        for(int i = 1; i <= ID; i++){
          Boton2.Decrementar();
        }
      }
      else{
```

```
      ID = ID + 1;
      if(ID == 10){
        ID = 1;
      }
    }
  }
 }
```

A).- no imprime nada
B).- ls y li están mal
C).- está mal todo el programa
D).- las librerías están mal
E).- la posición de los displays está mal

*12.- ¿Qué le hace falta para que el código compile?

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooClaseContadorV2.h>

void setup() {
  Timer1.initialize();
  MFS.initialize(&Timer1);
  Serial.begin(9600);
  MFS.write("0000");
}

Contador D4(0,-1,10), D3(0,-1,10), D2(0,-1,10), D1(0,-1,10), Seleccion(1,1,5);
char Display[4] = {'0','0','0','0'};

void loop() {
  byte btn = MFS.getButton();

  if (btn) {
    byte buttonNumber = btn & B00111111;
    byte buttonAction = btn & B11000000;

    if ((buttonNumber == 1) && (buttonAction == BUTTON_SHORT_RELEASE_IND)) fase =
1;
    if ((buttonNumber == 2) && (buttonAction == BUTTON_SHORT_RELEASE_IND)) fase =
2;
    if ((buttonNumber == 3) && (buttonAction == BUTTON_LONG_RELEASE_IND))
continuar = !continuar;
    if ((buttonNumber == 3) && (buttonAction == BUTTON_SHORT_RELEASE_IND)) {
      Seleccion.Incrementar();
      if (Seleccion.GetN() == Seleccion.GetLimiteSuperior())
Seleccion.SetN( Seleccion.GetLimiteInferior() );
    }
  }
```

```
  if (continuar) {
  switch(fase) {
     case 1: // Fase 1
           switch(Seleccion.GetN()) {
              case 1: D4.Incrementar(); if (D4.GetN() == D4.GetLimiteSuperior()) D4.SetN( 0 );
break;
              case 2: D3.Incrementar(); if (D3.GetN() == D3.GetLimiteSuperior()) D3.SetN( 0 );
break;
              case 3: D2.Incrementar(); if (D2.GetN() == D2.GetLimiteSuperior()) D2.SetN( 0 );
break;
              case 4: D1.Incrementar(); if (D1.GetN() == D1.GetLimiteSuperior()) D1.SetN( 0 );
break;
           }
           break;
     case 2: // Fase 2
           switch(Seleccion.GetN()) {
              case 1: D4.Decrementar(); if (D4.GetN() == D4.GetLimiteInferior())
D4.SetN( D4.GetLimiteSuperior() - 1 ); break;
              case 2: D3.Decrementar(); if (D3.GetN() == D3.GetLimiteSuperior())
D3.SetN( D3.GetLimiteSuperior() - 1 ); break;
              case 3: D2.Decrementar(); if (D2.GetN() == D2.GetLimiteSuperior())
D2.SetN( D2.GetLimiteSuperior() - 1 ); break;
              case 4: D1.Decrementar(); if (D1.GetN() == D1.GetLimiteSuperior())
D1.SetN( D1.GetLimiteSuperior() - 1 ); break;
           }
           break;
    }
  }

    MFS.writeLeds(LED_ALL, OFF);
    switch(Seleccion.GetN()) {
      case 1: MFS.writeLeds(LED_1, ON); break;
      case 2: MFS.writeLeds(LED_2, ON); break;
      case 3: MFS.writeLeds(LED_3, ON); break;
      case 4: MFS.writeLeds(LED_4, ON); break;
    }

    if (D4.GetN() == D4.GetLimiteInferior()) D4.SetN( D4.GetLimiteSuperior() - 1 );
    if (D3.GetN() == D3.GetLimiteInferior()) D3.SetN( D3.GetLimiteSuperior() - 1 );
    if (D2.GetN() == D2.GetLimiteInferior()) D2.SetN( D2.GetLimiteSuperior() - 1 );
    if (D1.GetN() == D1.GetLimiteInferior()) D1.SetN( D1.GetLimiteSuperior() - 1 );

}

A).-
int false, true;
char continuar = !continuar;
MFS.writeLeds(LED_ALL, ON);
Display[0] = '0';
Display[1] = '0';
Display[2] = '0';
Display[3] = '0';
```

B).-

```
Contador C1(0, 0, 4);

    if (buttonAction == BUTTON_PRESSED_IND){
      if( buttonNumber == 1 ){
        C1.Incrementar(); Serial.println(C1.GetN());
        }
      else if( buttonNumber == 2 ){
        C1.Decrementar(); Serial.println(C1.GetN());
      }
      else if( buttonNumber == 3 ){
         C1.SetN(0); Serial.println(C1.GetN());
      }
```

C).-

```
bool PlayPause = true;
byte fase = 0;
bool continuar = true;
     Display[0] = '0' + D1.GetN();
     Display[1] = '0' + D2.GetN();
     Display[2] = '0' + D3.GetN();
     Display[3] = '0' + D4.GetN();

     MFS.write(Display);
     Serial.println(D4.GetN());
     Serial.println(D3.GetN());
     Serial.println(D2.GetN());
     Serial.println(D1.GetN());
     Serial.println("-----------");
     delay(200);
```

D).-

```
int ID;

byte buttonNumber = btn & B00111111;
    byte buttonAction  = btn & B11000000;
 if (btn) {
          if (buttonAction == BUTTON_PRESSED_IND){
      if(buttonNumber == 1){
        for(int i = 1; i <= ID; i++){
          Boton1.Incrementar();
        }
      }
      else if(buttonNumber == 2){
        for(int i = 1; i <= ID; i++){
          Boton2.Decrementar();
        }
      }
      else{
        ID = ID + 1;
```

```
      if(ID == 10){
        ID = 1;
      }
    }
```

*13.- prender led y que aparezca en la pantalla que led tienes encendido

A).-

```
#include <pooClaseContadorV2.h>
#include <TimerOne.h>
#include <MultiFuncShield.h>
#include <Wire.h>

void setup() {
  Timer1.initialize();
    MFS.initialize(&Timer1);
      Serial.begin(9600);
}

Contador C1(0, 0, 4);
void loop() {
  byte boton = MFS.getButton();
  if (boton){
    byte buttonNumber = boton & B00111111;
    byte buttonAction = boton & B11000000;

    if (buttonAction == BUTTON_PRESSED_IND){
      if( buttonNumber == 1 ){
        C1.Incrementar(); Serial.println(C1.GetN());
        }
      else if( buttonNumber == 2 ){
        C1.Decrementar(); Serial.println(C1.GetN());
      }
      else if( buttonNumber == 3 ){
         C1.SetN(0); Serial.println(C1.GetN());
      }
    }

    if( C1.GetN() == 0 ){
          MFS.writeLeds(LED_1, OFF); MFS.writeLeds(LED_2, OFF); MFS.writeLeds(LED_3,
OFF); MFS.writeLeds(LED_4, OFF); MFS.write("   ");
    }
    else if( C1.GetN() == 1 ){
          MFS.writeLeds(LED_1, ON); MFS.writeLeds(LED_2, OFF); MFS.writeLeds(LED_3,
OFF); MFS.writeLeds(LED_4, OFF); MFS.write("Led1");
    }
    else if ( C1.GetN() == 2){
          MFS.writeLeds(LED_1, OFF); MFS.writeLeds(LED_2, ON); MFS.writeLeds(LED_3,
OFF); MFS.writeLeds(LED_4, OFF); MFS.write("Led2");
    }
    else if ( C1.GetN() == 3){
          MFS.writeLeds(LED_1, OFF); MFS.writeLeds(LED_2, OFF); MFS.writeLeds(LED_3,
ON); MFS.writeLeds(LED_4, OFF); MFS.write("Led3");
```

```
    }
    else if ( C1.GetN() == 4){
         MFS.writeLeds(LED_1, OFF); MFS.writeLeds(LED_2, OFF); MFS.writeLeds(LED_3,
OFF); MFS.writeLeds(LED_4, ON); MFS.write("Led4");
    }
  }
}
```

B).-

```
#include <pooClaseContadorV2.h>
#include <TimerOne.h>
#include <MultiFuncShield.h>
#include <Wire.h>

void setup() {
  Timer1.initialize();
  MFS.initialize(&Timer1);
  Serial.begin(9600);
  MFS.write( "0000" );


}
Contador Contador1(0, -1, 10), Contador2(0, -1, 10), Contador3(0, -1, 10), Contador4(0,
-1, 10), Contador5(0, -1, 6);
char x[4] = {'0', '0', '0', '0'};
void loop() {

  byte btn = MFS.getButton();

  if (btn){

    byte buttonNumber = btn & B00111111;
    byte buttonAction = btn & B11000000;

    if (buttonAction == BUTTON_SHORT_RELEASE_IND){
      if(buttonNumber == 1){
        switch(Led_N.GetN() ){
      MFS.write(Led_N);
MFS.writeLeds(LED_1, ON);   MFS.writeLeds(LED_2, O);   MFS.writeLeds(LED_3, ON);
MFS.writeLeds(LED_4, ON);
}
      }
      else if(buttonNumber == 2){
        switch( Contador5.GetN() ){
         MFS.write(Led_N);
   MFS.writeLeds(LED_1, OFF); MFS.writeLeds(LED_2, OFF); MFS.writeLeds(LED_3, OFF);
MFS.writeLeds(LED_4, OFF);
        }
      }

      else if(buttonNumber == 3){
         Contador5.Incrementar();
```

```
        if(Contador5.GetN() == 9) Contador5.SetN(0);
      }
      else if(buttonNumber == 3) {

        }
      }
```

C).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooClaseContadorV2.h>
void setup() {
Timer1.initialize();
MFS.initialize(&Timer1);
Serial.begin(9600);
}
Contador Boton1(0, 0, 10), Boton2(0, 0, 10), Boton3(1, 1, 10) ;
int ID;

void loop() {}

byte btn = MFS.getButton();
byte buttonNumber = btn & B00111111;
byte buttonAction  = btn & B11000000;
 if (btn) {
          if (buttonAction == BUTTON_PRESSED_IND){
      if(buttonNumber == 1){
        for(int i = 1; i <= ID; i++){
          Boton1.Incrementar();
        }
      }
      else if(buttonNumber == 2){
        for(int i = 1; i <= ID; i++){
          Boton2.Decrementar();
        }
      }
      else{
        ID = ID + 1;
        if(ID == 10){
          ID = 1;
        }
      }
  }
}
```

*14.- prender, apagar y reiniciar Leds
Boton 1 incrementa leds
boton 2 decrementa leds
Boton 3 reinicia

A).-

```
#include <pooClaseContadorV2.h>
```

```
#include <TimerOne.h>
#include <MultiFuncShield.h>
#include <Wire.h>

void setup() {
  Timer1.initialize();
    MFS.initialize(&Timer1);
      Serial.begin(9600);
}

Contador C1(0, 0, 4);
void loop() {
  byte boton = MFS.getButton();
  if (boton){
    byte buttonNumber = boton & B00111111;
    byte buttonAction = boton & B11000000;

    if (buttonAction == BUTTON_PRESSED_IND){
      if( buttonNumber == 1 ){
        C1.Incrementar(); Serial.println(C1.GetN());
        }
      else if( buttonNumber == 2 ){
        C1.Decrementar(); Serial.println(C1.GetN());
      }
      else if( buttonNumber == 3 ){
         C1.SetN(0); Serial.println(C1.GetN());
      }
    }

    if( C1.GetN() == 0 ){
          MFS.writeLeds(LED_1, OFF); MFS.writeLeds(LED_2, OFF); MFS.writeLeds(LED_3,
OFF); MFS.writeLeds(LED_4, OFF);
    }
    else if( C1.GetN() == 1 ){
          MFS.writeLeds(LED_1, ON); MFS.writeLeds(LED_2, OFF); MFS.writeLeds(LED_3,
OFF); MFS.writeLeds(LED_4, OFF);
    }
    else if ( C1.GetN() == 2){
          MFS.writeLeds(LED_1, ON); MFS.writeLeds(LED_2, ON); MFS.writeLeds(LED_3,
OFF); MFS.writeLeds(LED_4, OFF);
    }
    else if ( C1.GetN() == 3){
          MFS.writeLeds(LED_1, ON); MFS.writeLeds(LED_2, ON); MFS.writeLeds(LED_3,
ON); MFS.writeLeds(LED_4, OFF);
    }
    else if ( C1.GetN() == 4){
          MFS.writeLeds(LED_1, ON); MFS.writeLeds(LED_2, ON); MFS.writeLeds(LED_3,
ON); MFS.writeLeds(LED_4, ON);
    }
  }
}
```

B).-

```
#include <pooClaseContadorV2.h>
#include <TimerOne.h>
#include <MultiFuncShield.h>
#include <Wire.h>

void setup() {
  Timer1.initialize();
  MFS.initialize(&Timer1);
  Serial.begin(9600);
  MFS.write( "0000" );


}
Contador Contador1(0, -1, 10), Contador2(0, -1, 10), Contador3(0, -1, 10), Contador4(0,
-1, 10), Contador5(0, -1, 6);
char x[4] = {'0', '0', '0', '0'};
void loop() {

  byte btn = MFS.getButton();

  if (btn){

    byte buttonNumber = btn & B00111111;
    byte buttonAction = btn & B11000000;

    if (buttonAction == BUTTON_SHORT_RELEASE_IND){
      if(buttonNumber == 1){
        switch( Contador5.GetN() ){
          case 0: Contador1.Incrementar(); if(Contador1.GetN() == 10) Contador1.SetN(0);
break;
          case 1: Contador2.Incrementar(); if(Contador2.GetN() == 10) Contador2.SetN(0);
break;
          case 2: Contador3.Incrementar(); if(Contador3.GetN() == 10) Contador3.SetN(0);
break;
          case 3: Contador4.Incrementar(); if(Contador4.GetN() == 10) Contador4.SetN(0);
break;
MFS.writeLeds(LED_1, ON);   MFS.writeLeds(LED_2, ON);   MFS.writeLeds(LED_3, ON);
MFS.writeLeds(LED_4, ON);
}
      }
      else if(buttonNumber == 2){
        switch( Contador5.GetN() ){
          case 0: Contador1.Decrementar(); if(Contador1.GetN() == -1) Contador1.SetN(9);
break;
          case 1: Contador2.Decrementar(); if(Contador2.GetN() == -1) Contador2.SetN(9);
break;
          case 2: Contador3.Decrementar(); if(Contador3.GetN() == -1) Contador3.SetN(9);
break;
          case 3: Contador4.Decrementar(); if(Contador4.GetN() == -1) Contador4.SetN(9);
break;
    MFS.writeLeds(LED_1, OFF); MFS.writeLeds(LED_2, OFF); MFS.writeLeds(LED_3, OFF);
MFS.writeLeds(LED_4, OFF);
```

```
      }
    }

    else if(buttonNumber == 3){
       Contador5.Incrementar();
       if(Contador5.GetN() == 9) Contador5.SetN(0);
    }
    else if(buttonNumber == 3) {

      }
    }
```

*15.- prender, apagar y reiniciar Leds
Boton 1 decrementa leds
boton 2 incrementa leds
Boton 3 reinicia

A).-

```
#include <pooClaseContadorV2.h>
#include <TimerOne.h>
#include <MultiFuncShield.h>
#include <Wire.h>

void setup() {
  Timer1.initialize();
  MFS.initialize(&Timer1);
  Serial.begin(9600);
  MFS.write( "0000" );


}
Contador Contador1(0, -1, 10), Contador2(0, -1, 10), Contador3(0, -1, 10), Contador4(0,
-1, 10), Contador5(0, -1, 6);
char x[4] = {'0', '0', '0', '0'};
void loop() {

  byte btn = MFS.getButton();

  if (btn){

    byte buttonNumber = btn & B00111111;
    byte buttonAction = btn & B11000000;

    if (buttonAction == BUTTON_SHORT_RELEASE_IND){
      if(buttonNumber == 2){
        switch( Contador5.GetN() ){
          case 0: Contador1.Incrementar(); if(Contador1.GetN() == 10) Contador1.SetN(0);
break;
          case 1: Contador2.Incrementar(); if(Contador2.GetN() == 10) Contador2.SetN(0);
break;
          case 2: Contador3.Incrementar(); if(Contador3.GetN() == 10) Contador3.SetN(0);
break;
```

```
        case 3: Contador4.Incrementar(); if(Contador4.GetN() == 10) Contador4.SetN(0);
break;
MFS.writeLeds(LED_1, ON);   MFS.writeLeds(LED_2, ON);   MFS.writeLeds(LED_3, ON);
MFS.writeLeds(LED_4, ON);
}
     }
     else if(buttonNumber == 1){
       switch( Contador5.GetN() ){
         case 0: Contador1.Decrementar(); if(Contador1.GetN() == -1) Contador1.SetN(9);
break;
         case 1: Contador2.Decrementar(); if(Contador2.GetN() == -1) Contador2.SetN(9);
break;
         case 2: Contador3.Decrementar(); if(Contador3.GetN() == -1) Contador3.SetN(9);
break;
         case 3: Contador4.Decrementar(); if(Contador4.GetN() == -1) Contador4.SetN(9);
break;
  MFS.writeLeds(LED_1, OFF); MFS.writeLeds(LED_2, OFF); MFS.writeLeds(LED_3, OFF);
MFS.writeLeds(LED_4, OFF);
       }
     }

     else if(buttonNumber == 3){
        Contador5.Incrementar();
        if(Contador5.GetN() == 9) Contador5.SetN(0);
     }
     else if(buttonNumber == 3) {

       }
     }

B).-
#include <pooClaseContadorV2.h>
#include <TimerOne.h>
#include <MultiFuncShield.h>
#include <Wire.h>

void setup() {
  Timer1.initialize();
    MFS.initialize(&Timer1);
     Serial.begin(9600);
}

Contador C1(0, 0, 4);
void loop() {
  byte boton = MFS.getButton();
  if (boton){
    byte buttonNumber = boton & B00111111;
    byte buttonAction = boton & B11000000;

    if (buttonAction == BUTTON_PRESSED_IND){
      if( buttonNumber == 2 ){
        C1.Incrementar(); Serial.println(C1.GetN());
```

```
      }
     else if( buttonNumber == 1 ){
       C1.Decrementar(); Serial.println(C1.GetN());
     }
     else if( buttonNumber == 3 ){
       C1.SetN(0); Serial.println(C1.GetN());
     }
   }

   if( C1.GetN() == 0 ){
        MFS.writeLeds(LED_1, OFF); MFS.writeLeds(LED_2, OFF); MFS.writeLeds(LED_3,
OFF); MFS.writeLeds(LED_4, OFF);
   }
   else if( C1.GetN() == 1 ){
        MFS.writeLeds(LED_1, ON); MFS.writeLeds(LED_2, OFF); MFS.writeLeds(LED_3,
OFF); MFS.writeLeds(LED_4, OFF);
   }
   else if ( C1.GetN() == 2){
        MFS.writeLeds(LED_1, ON); MFS.writeLeds(LED_2, ON); MFS.writeLeds(LED_3,
OFF); MFS.writeLeds(LED_4, OFF);
   }
   else if ( C1.GetN() == 3){
        MFS.writeLeds(LED_1, ON); MFS.writeLeds(LED_2, ON); MFS.writeLeds(LED_3,
ON); MFS.writeLeds(LED_4, OFF);
   }
   else if ( C1.GetN() == 4){
        MFS.writeLeds(LED_1, ON); MFS.writeLeds(LED_2, ON); MFS.writeLeds(LED_3,
ON); MFS.writeLeds(LED_4, ON);
   }
  }
}
```

*16.- prender, apagar y reiniciar Leds
Boton 1 reinicia
boton 2 incrementa leds
Boton 3 decrementa leds

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <pooClaseContadorV2.h>
void setup() {
Timer1.initialize();
MFS.initialize(&Timer1);
Serial.begin(9600);
}
Contador Boton1(0, 0, 10), Boton2(0, 0, 10), Boton3(1, 1, 10) ;
int ID;

void loop() {}
```

```
byte btn = MFS.getButton();
byte buttonNumber = btn & B00111111;
byte buttonAction  = btn & B11000000;
 if (btn) {
          if (buttonAction == BUTTON_PRESSED_IND){
      if(buttonNumber == 1){
        for(int i = 1; i <= ID; i++){
          Boton1.Incrementar();
        }
      }
      else if(buttonNumber == 2){
        for(int i = 1; i <= ID; i++){
          Boton2.Decrementar();
        }
      }
      else{
        ID = ID + 1;
        if(ID == 10){
          ID = 1;
        }
      }
  }
}
```

B).-

```
#include <pooClaseContadorV2.h>
#include <TimerOne.h>
#include <MultiFuncShield.h>
#include <Wire.h>

void setup() {
  Timer1.initialize();
  MFS.initialize(&Timer1);
  Serial.begin(9600);
  MFS.write( "0000" );


}
Contador Contador1(0, -1, 10), Contador2(0, -1, 10), Contador3(0, -1, 10), Contador4(0,
-1, 10), Contador5(0, -1, 6);
char x[4] = {'0', '0', '0', '0'};
void loop() {

  byte btn = MFS.getButton();

  if (btn){

    byte buttonNumber = btn & B00111111;
    byte buttonAction = btn & B11000000;

    if (buttonAction == BUTTON_SHORT_RELEASE_IND){
```

```
    if(buttonNumber == 2){
      switch( Contador5.GetN() ){
        case 0: Contador1.Incrementar(); if(Contador1.GetN() == 10) Contador1.SetN(0);
break;
        case 1: Contador2.Incrementar(); if(Contador2.GetN() == 10) Contador2.SetN(0);
break;
        case 2: Contador3.Incrementar(); if(Contador3.GetN() == 10) Contador3.SetN(0);
break;
        case 3: Contador4.Incrementar(); if(Contador4.GetN() == 10) Contador4.SetN(0);
break;
MFS.writeLeds(LED_1, ON);   MFS.writeLeds(LED_2, ON);   MFS.writeLeds(LED_3, ON);
MFS.writeLeds(LED_4, ON);
}
    }
    else if(buttonNumber == 3){
      switch( Contador5.GetN() ){
        case 0: Contador1.Decrementar(); if(Contador1.GetN() == -1) Contador1.SetN(9);
break;
        case 1: Contador2.Decrementar(); if(Contador2.GetN() == -1) Contador2.SetN(9);
break;
        case 2: Contador3.Decrementar(); if(Contador3.GetN() == -1) Contador3.SetN(9);
break;
        case 3: Contador4.Decrementar(); if(Contador4.GetN() == -1) Contador4.SetN(9);
break;
  MFS.writeLeds(LED_1, OFF); MFS.writeLeds(LED_2, OFF); MFS.writeLeds(LED_3, OFF);
MFS.writeLeds(LED_4, OFF);
      }
    }

    else if(buttonNumber == 1){
      Contador5.Incrementar();
      if(Contador5.GetN() == 9) Contador5.SetN(0);
    }
    else if(buttonNumber == 3) {

      }
    }

  MFS.write( x );
}
C).-
#include <pooClaseContadorV2.h>
#include <TimerOne.h>
#include <MultiFuncShield.h>
#include <Wire.h>

void setup() {
  Timer1.initialize();
    MFS.initialize(&Timer1);
      Serial.begin(9600);
}
```

```
Contador C1(0, 0, 4);
void loop() {
  byte boton = MFS.getButton();
  if (boton){
    byte buttonNumber = boton & B00111111;
    byte buttonAction = boton & B11000000;

    if (buttonAction == BUTTON_PRESSED_IND){
      if( buttonNumber == 2 ){
        C1.Incrementar(); Serial.println(C1.GetN());
        }
      else if( buttonNumber == 3 ){
        C1.Decrementar(); Serial.println(C1.GetN());
      }
      else if( buttonNumber == 1 ){
         C1.SetN(0); Serial.println(C1.GetN());
      }
    }

    if( C1.GetN() == 0 ){
          MFS.writeLeds(LED_1, OFF); MFS.writeLeds(LED_2, OFF); MFS.writeLeds(LED_3,
OFF); MFS.writeLeds(LED_4, OFF);
    }
    else if( C1.GetN() == 1 ){
          MFS.writeLeds(LED_1, ON); MFS.writeLeds(LED_2, OFF); MFS.writeLeds(LED_3,
OFF); MFS.writeLeds(LED_4, OFF);
    }
    else if ( C1.GetN() == 2){
          MFS.writeLeds(LED_1, ON); MFS.writeLeds(LED_2, ON); MFS.writeLeds(LED_3,
OFF); MFS.writeLeds(LED_4, OFF);
    }
    else if ( C1.GetN() == 3){
          MFS.writeLeds(LED_1, ON); MFS.writeLeds(LED_2, ON); MFS.writeLeds(LED_3,
ON); MFS.writeLeds(LED_4, OFF);
    }
    else if ( C1.GetN() == 4){
          MFS.writeLeds(LED_1, ON); MFS.writeLeds(LED_2, ON); MFS.writeLeds(LED_3,
ON); MFS.writeLeds(LED_4, ON);
    }
  }
}
```

*17.- decrementar leds con el potenciometro del 4 al 1
 A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
```

```
}
int valorPot=0;
void loop() {
valorPot = analogRead(POT_PIN);
Serial.print("Valor: ");
Serial.println(POT_PIN); brillo();
}
void brillo() {
  MFS.write(analogRead(POT_PIN));
  if(valorPot >= 5 && valorPot <=25) {
MFS.writeLeds(LED_1, OFF);
MFS.writeLeds(LED_2, OFF);
MFS.writeLeds(LED_3, OFF);
MFS.writeLeds(LED_4, OFF);
}
else if(valorPot >=26 && valorPot <=35) {
MFS.writeLeds(LED_1, OFF);
MFS.writeLeds(LED_2, OFF);
MFS.writeLeds(LED_3, OFF);
MFS.writeLeds(LED_4, ON);
}
else if(valorPot >=36 && valorPot <=45) {
MFS.writeLeds(LED_1, OFF);
MFS.writeLeds(LED_2, OFF);
MFS.writeLeds(LED_3, ON);
MFS.writeLeds(LED_4, ON);
}
else if(valorPot >=46 && valorPot <=55) {
MFS.writeLeds(LED_1, OFF);
MFS.writeLeds(LED_2, ON);
MFS.writeLeds(LED_3, ON);
MFS.writeLeds(LED_4, ON);
}
else if(valorPot >=56 && valorPot <=100) {
MFS.writeLeds(LED_1, ON);
MFS.writeLeds(LED_2, ON);
MFS.writeLeds(LED_3, ON);
MFS.writeLeds(LED_4, ON);
}
}
```

B).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <PooLed.V1.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
int Pot=0;
```

```
void loop() {
if(Led_4 && Pot <=125) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;
void loop() {
Serial.println(estado);
estado=PrenderLed(Led = 4);  estado=ApagarLed(Led = 2);  estado=ApagarLed(Led = 3);
estado=ApagarLed(Led = 1); delay(1000); }
 else if(Led_3 && Pot <=300) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;

Serial.println(estado);
estado=PrenderLed(Led = 3); estado=ApagarLed(Led = 1);  estado=ApagarLed(Led = 4);
estado=ApagarLed(Led = 2);  delay(1000); }
else if(Led_2 && Pot <=700) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;

Serial.println(estado);
estado=PrenderLed(Led = 2); estado=ApagarLed(Led = 1);  estado=ApagarLed(Led = 4);
estado=ApagarLed(Led = 3);  delay(1000); }

else if(Led_1 && Pot <=1000) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;

Serial.println(estado);
estado=PrenderLed(Led = 1); estado=ApagarLed(Led = 1);  estado=ApagarLed(Led = 2);
estado=ApagarLed(Led = 4);  delay(1000); }
```

*18.- incrementar Leds con el potenciomentro del 1 al 4

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <PooLed.V1.h>
void setup() {
Serial.begin(9600);
```

```
Timer1.initialize();
MFS.initialize(&Timer1);
}
int Pot=0;
void loop() {
if(Led_1 && Pot <=125) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;
void loop() {
Serial.println(estado);
estado=PrenderLed(Led = 1);  estado=ApagarLed(Led = 2);  estado=ApagarLed(Led = 3);
estado=ApagarLed(Led = 4); delay(1000); }
 else if(Led_2,Led_1 && Pot <=300) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;

Serial.println(estado);
estado=PrenderLed(Led = 2); estado=PrenderLed(Led = 1);  estado=ApagarLed(Led = 3);
estado=ApagarLed(Led = 4);  delay(1000); }
else if(Led_3,Led_2,Led_1 && Pot <=700) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;

Serial.println(estado);
estado=PrenderLed(Led = 3); estado=PrenderLed(Led = 1);  estado=PrenderLed(Led = 2);
estado=ApagarLed(Led = 4);  delay(1000); }

else if(Led_4, Led_3,Led_2,Led_1&& Pot <=1000) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;

Serial.println(estado);
estado=PrenderLed(Led = 1); estado=PrendeLed(Led = 4);  estado=PrendeLed(Led = 2);
estado=PrendeLed(Led = 3);  delay(1000); }

B).-
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
```

```
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);

}
int valorPot=0;
void loop() {
valorPot = analogRead(POT_PIN);
Serial.print("Valor: ");
Serial.println(POT_PIN); brillo();
}
void brillo() {
  MFS.write(analogRead(POT_PIN));
  if(valorPot >= 5 && valorPot <=25) {
MFS.writeLeds(LED_1, OFF);
MFS.writeLeds(LED_2, OFF);
MFS.writeLeds(LED_3, OFF);
MFS.writeLeds(LED_4, OFF);
}
else if(valorPot >=26 && valorPot <=35) {
MFS.writeLeds(LED_1, ON);
MFS.writeLeds(LED_2, OFF);
MFS.writeLeds(LED_3, OFF);
MFS.writeLeds(LED_4, OFF);
}
else if(valorPot >=36 && valorPot <=45) {
MFS.writeLeds(LED_1, ON);
MFS.writeLeds(LED_2, ON);
MFS.writeLeds(LED_3, OFF);
MFS.writeLeds(LED_4, OFF);
}
else if(valorPot >=46 && valorPot <=55) {
MFS.writeLeds(LED_1, ON);
MFS.writeLeds(LED_2, ON);
MFS.writeLeds(LED_3, ON);
MFS.writeLeds(LED_4, OFF);
}
else if(valorPot >=56 && valorPot <=100) {
MFS.writeLeds(LED_1, ON);
MFS.writeLeds(LED_2, ON);
MFS.writeLeds(LED_3, ON);
MFS.writeLeds(LED_4, ON);
}
}
```

*19.- mensaje en el display mientras decrementa el potenciomentro del 4 al 1
sehr gut= muy bien (alemán)

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
```

```
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);

}
int valorPot=0;
void loop() {
valorPot = analogRead(POT_PIN);
Serial.print("Valor: ");
Serial.println(POT_PIN); brillo();
}
void brillo() {
  if(valorPot >= 5 && valorPot <=25) {
MFS.writeLeds(LED_1, OFF);
MFS.writeLeds(LED_2, OFF);
MFS.writeLeds(LED_3, OFF);
MFS.writeLeds(LED_4, OFF);
MFS.write("s");
}
else if(valorPot >=26 && valorPot <=35) {
MFS.writeLeds(LED_1, OFF);
MFS.writeLeds(LED_2, OFF);
MFS.writeLeds(LED_3, OFF);
MFS.writeLeds(LED_4, ON);
MFS.write("se");
}
else if(valorPot >=36 && valorPot <=45) {
MFS.writeLeds(LED_1, OFF);
MFS.writeLeds(LED_2, OFF);
MFS.writeLeds(LED_3, ON);
MFS.writeLeds(LED_4, ON);
MFS.write("seh");
}
else if(valorPot >=46 && valorPot <=55) {
MFS.writeLeds(LED_1, OFF);
MFS.writeLeds(LED_2, ON);
MFS.writeLeds(LED_3, ON);
MFS.writeLeds(LED_4, ON);
MFS.write("sehr ");
}
else if(valorPot >=56 && valorPot <=100) {
MFS.writeLeds(LED_1, ON);
MFS.writeLeds(LED_2, ON);
MFS.writeLeds(LED_3, ON);
MFS.writeLeds(LED_4, ON);
MFS.write("fact");
}
}
```

B).-

```
#include <TimerOne.h>
```

```
#include <Wire.h>
#include <MultiFuncShield.h>
#include <PooLed.V1.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
int Pot=0;
void loop() {
if(Led_1 && Pot <=125) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;

Serial.println(estado);
estado=PrenderLed(Led = 1);  estado=ApagarLed(Led = 2);  estado=ApagarLed(Led = 3);
estado=ApagarLed(Led = 4);  Serial.println(“s”); delay(1000); }
 else if(Led_2,Led_1 && Pot <=300) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;

Serial.println(estado);
estado=PrenderLed(Led = 2); estado=PrenderLed(Led = 1);  estado=ApagarLed(Led = 3);
estado=ApagarLed(Led = 4);  Serial.println(“se”); delay(1000); }
else if(Led_3,Led_2,Led_1 && Pot <=700) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;
void loop() {
Serial.println(estado);
estado=PrenderLed(Led = 3); estado=PrenderLed(Led = 1);  estado=PrenderLed(Led = 2);
estado=ApagarLed(Led = 4);  Serial.println(“seh”); delay(1000); }

else if(Led_4, Led_3,Led_2,Led_1&& Pot <=1000) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;

Serial.println(estado);
estado=PrenderLed(Led = 4); estado=PrendeLed(Led = 1);  estado=PrendeLed(Led = 2);
estado=PrendeLed(Led = 3);  Serial.println(“sehr gut”); delay(1000); }
```

*20.-mensaje en el display mientras incrementa el potenciomentro del 1 al 4
sehr gut= muy bien (alemán)

A).-

```
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
#include <PooLed.V1.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);
}
int Pot=0;
void loop() {
if(Led_1 && Pot <=125) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;
void loop() {
Serial.println(estado);
estado=PrenderLed(Led = 1);  estado=ApagarLed(Led = 2);  estado=ApagarLed(Led = 3);
estado=ApagarLed(Led = 4); delay(1000); }
 else if(Led_2,Led_1 && Pot <=300) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;

Serial.println(estado);
estado=PrenderLed(Led = 2); estado=PrenderLed(Led = 1);  estado=ApagarLed(Led = 3);
estado=ApagarLed(Led = 4);  delay(1000); }
else if(Led_3,Led_2,Led_1 && Pot <=700) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;

Serial.println(estado);
estado=PrenderLed(Led = 3); estado=PrenderLed(Led = 1);  estado=PrenderLed(Led = 2);
estado=ApagarLed(Led = 4);  delay(1000); }

else if(Led_4, Led_3,Led_2,Led_1&& Pot <=1000) {
bool cadena_tmp;
switch (prendido) {
case 0: cadena_tmp = ‘ON’; break;
case 1: cadena_tmp = ‘OFF’; break;
default: cadena_tmp=“Parpadear”; break;
```

```
Serial.println(estado);
estado=PrenderLed(Led = 1); estado=PrendeLed(Led = 4);  estado=PrendeLed(Led = 2);
estado=PrendeLed(Led = 3);  delay(1000); }

B).-
#include <TimerOne.h>
#include <Wire.h>
#include <MultiFuncShield.h>
void setup() {
Serial.begin(9600);
Timer1.initialize();
MFS.initialize(&Timer1);

}
int valorPot=0;
void loop() {
valorPot = analogRead(POT_PIN);
Serial.print("Valor: ");
Serial.println(POT_PIN); brillo();
}
void brillo() {
  if(valorPot >= 5 && valorPot <=25) {
MFS.writeLeds(LED_1, OFF);
MFS.writeLeds(LED_2, OFF);
MFS.writeLeds(LED_3, OFF);
MFS.writeLeds(LED_4, OFF);
MFS.write("s");
}
else if(valorPot >=26 && valorPot <=35) {
MFS.writeLeds(LED_1, ON);
MFS.writeLeds(LED_2, OFF);
MFS.writeLeds(LED_3, OFF);
MFS.writeLeds(LED_4, OFF);
MFS.write("se");
}
else if(valorPot >=36 && valorPot <=45) {
MFS.writeLeds(LED_1, ON);
MFS.writeLeds(LED_2, ON);
MFS.writeLeds(LED_3, OFF);
MFS.writeLeds(LED_4, OFF);
MFS.write("seh ");
}
else if(valorPot >=46 && valorPot <=55) {
MFS.writeLeds(LED_1, ON);
MFS.writeLeds(LED_2, ON);
MFS.writeLeds(LED_3, ON);
MFS.writeLeds(LED_4, OFF);
MFS.write("sehr ");
}
else if(valorPot >=56 && valorPot <=100) {
MFS.writeLeds(LED_1, ON);
```

```
MFS.writeLeds(LED_2, ON);
MFS.writeLeds(LED_3, ON);
MFS.writeLeds(LED_4, ON);
MFS.write("fact");
}
}
```