

1: encender y apagar el led 1.

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); Serial.println("Led 1 encendido"); delay(1000);
  MFS.writeLeds(LED_1, OFF); Serial.println("Led 1 apagado"); 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);
  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);
  int Led=1;
}
void loop() {
  MFS.writeLed(1,ON); delay(1000);
  MFS.writeLed(1,OFF); delay(1000);
}
```

2: encender y apagar el led 2.

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); Serial.println("Led 2 encendido"); delay(1000);
  MFS.writeLeds(LED_2, OFF); Serial.println("Led 2 apagado"); delay(1000);
}
```

B).-

```
#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);
}
```

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 = 2);
    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=2;
}
void loop() {
  MFS.writeLed(2,ON); delay(1000);
  MFS.writeLed(2,OFF); delay(1000);
}
```

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);
}
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);
}
```

B).-

```
#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);
}
```

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);
  int Led=3;
}
void loop() {
  MFS.writeLed(3,ON); delay(1000); MFS.writeLed(3,OFF); delay(1000); }
```

4: 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,OFF); delay(1000);
}

```

5: 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>
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);

```

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);
    --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_2, ON); delay(1000);
    MFS.writeLeds(LED_1, 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 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();
  }
}
```

6: 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( 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= 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();
  }
}

```

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>
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);
}

```

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= 1;
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_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);
}

```

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

```

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>
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);
}

```

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();
  ++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_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);
}
```

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

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);
}
#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);
}

```

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);
  --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_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);
}

```

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()){
        Led2.Apagar(); Led3.Apagar();
    }
}

```

10: 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();
    }
}

```

11: 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_2, 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();
  }
}

```

12: 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>
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);
    }

```

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);
  --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_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();
  }
}

```

13: 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();
}
}
}

```

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 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);
}
#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);
}

```

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= 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_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);
}
```

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()){
    Led2.Apagar(); Led4.Apagar();
  }
}
```

16: 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>
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);
}

```

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 = 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_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);
}

```

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();
}
}
}

```

17: 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>
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);
}

```

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 = 3;
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_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);
}

```

D).-

```

#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();
  }
}

```

18: 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>
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);
}

```

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 = 3;
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_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);
}
```

D).-

```
#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();
  }
}
```

19.- 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);
}

```

20.- 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);
}
21.- 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); }

```


22.- 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);
}
#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);
}

```

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_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); }
25.- parpadeo rápido de los cuatro leds

```

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.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); }

```

26.- semaforo

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);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);
}
```

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>
#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); }
}

```

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,OFF); delay(1000);
MFS.writeLed(2,ON); delay(1000);
MFS.writeLed(2,OFF); delay(1000);
MFS.writeLed(3,ON); delay(1000);
MFS.writeLed(3,OFF); delay(1000);
}
}

```

27.- 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 <pooLcdV2.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,OFF); delay(1000);
MFS.writeLed(4,ON); delay(1000);
MFS.writeLed(4,OFF); delay(1000);
MFS.writeLed(2,ON); delay(1000);
MFS.writeLed(2,OFF); delay(1000);
MFS.writeLed(3,ON); delay(1000);
MFS.writeLed(3,OFF); delay(1000);
}

```

28.- 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);
}
#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);
}

```

29.- 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);
}
void loop() {
MFS.writeLeds(LED_2, ON); MFS.beep(); delay(1000);
MFS.writeLeds(LED_2, OFF); delay(1000);
}

```

```

}
#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);
}
30.- 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);
}
void loop() {
  MFS.writeLeds(LED_3, ON); MFS.beep(); delay(1000);
  MFS.writeLeds(LED_3, OFF); delay(1000);
}
#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);
}

```

```

31.- 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);
}
#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);
}

```

32.- encender leds con el potenciómetro

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);
}
}
}

```

33.- decrementar leds con el potenciómetro 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); }

```

34.- incrementar Leds con el potenciometro del 1 al 4

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, 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);
}
}
}
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;
}
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 = PrenderLed(Led = 4); estado = PrenderLed(Led = 2);
estado = PrenderLed(Led = 3); delay(1000); }

```

35.- mensaje en el display mientras decrementa el potenciometro 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");
}
}
}

```

```

#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); }

```

36.-mensaje en el display mientras incrementa el potenciometro 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");
}
}

```

37.- 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) {

}
}

```

38.- 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);
    }
}
}
}

```

39.- prender, apagar y reiniciar Leds

Boton 1 reinicia

boton 2 incrementa leds

Boton 3 decrementa leds

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 == 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 );
}

```

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 == 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);
    }
}
}
}

```

40.- 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");
    }
}
}
}

```

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 == 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) {

}
}
```