ใบงานที่ 1

การทดสอบและอัพโหลดโปรแกรม

จุดประสงค์การเรียนรู้

1.ศึกษาการทดสอบบอร์ดทดลอง ESP32

2.ศึกษาการใช้งาน Serial Monitor ดูข้อมูลที่ส่งออกจากบอร์ดทดลอง ESP32

3.ศึกษาการใช้งาน Serial ส่งข้อมูลจากแป้นพิมพ์ไปยังบอร์ต

4.ศึกษาการใช้งาน Serial Plotter

เครื่องมือและอุปกรณ์การทดลอง

1.เครื่องไมโครคอมพิวเตอร์

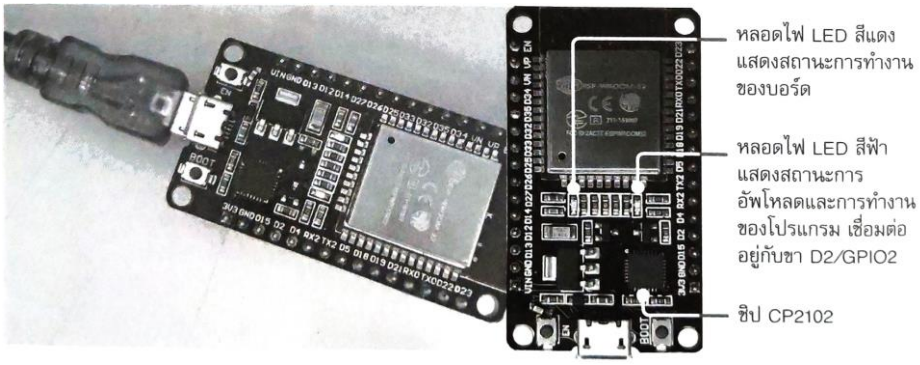
2.บอรด์ NodeMCU ESP32

3.โปรแกรมการทดลอง

4.อุปกรณ์อิเล็กทอรนิกส์สำหรับทดลอง

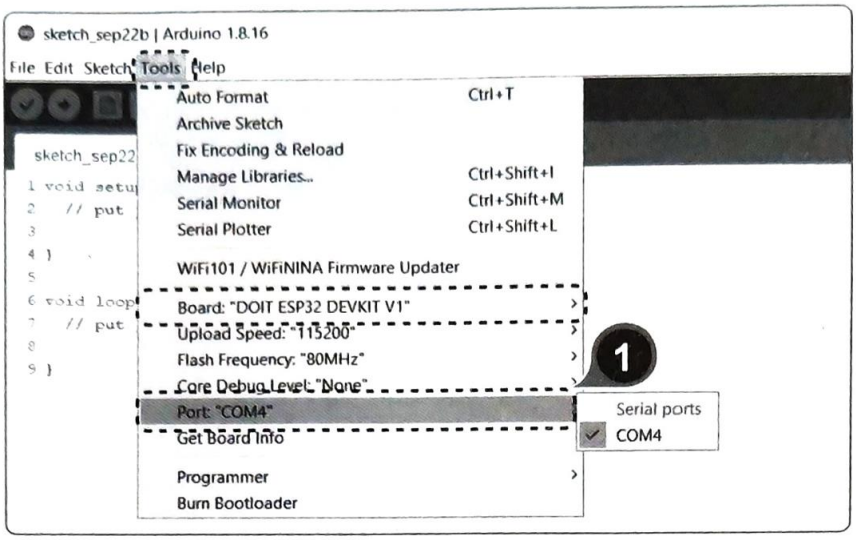
การทดลองที่ 1 ทดสอบบอร์ดด้วยไฟกะพริบด้วย Blink

บอร์ด NodeMCU ESP32 จะมีหลอดไฟ LED อยู่ 2 ตำแหน่ง ตำแหน่งแรกจะเป็นหลอดไฟ LED สีแดง (Power LED) ใช้บอกสถานะการทำงานของบอร์ด เมื่อจ่ายไฟให้กับบอร์ดหลอดไฟ LED สีแดงนี้จะติด ตำแหน่งที่ สองจะเป็นหลอดไฟ LED สีฟ้า (On Board LED) ใช้บอกสถานะการทำงานของโปรแกรม หลอดไฟนี้จะเชื่อมต่อ อยู่กับขา D2 หรือ GPIO2 เพราะฉะนั้นถ้าจะเทสบอร์ดโดยให้หลอดไฟ LED สีฟ้าที่อยู่บนบอร์ดกระพริบ ต้องกำหนดให้ขา D2 หรือ GPIO2 มีสถานะเป็น OUTPUT แล้วจึงค่อยกำหนดให้มีสถานะเป็น HIGH หรือ LOW เพื่อให้หลอดไฟ LED ติด/ดับ ตามต้องการ

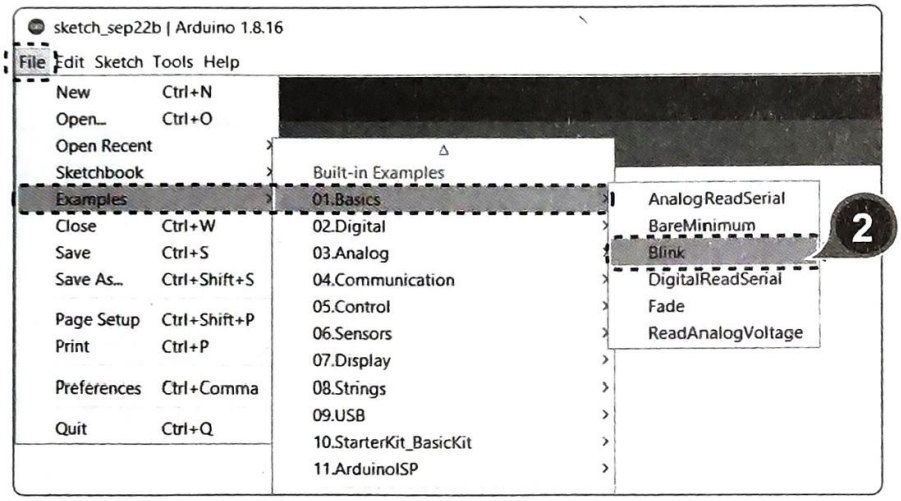


เริ่มต้นการเขียนโปรแกรมเพื่อทดสอบการทำงานของบอร์ด ก่อนอื่นให้เราเชื่อมต่อบอร์ดเข้ากับคอมพิวเตอร์ด้วยสาย USB (หลอดไฟ LED สีแดงบนบอร์ดจะติด แสดงว่ามีไฟเลี้ยงจ่ายให้กับบอร์ด พร้อมทำงาน) แล้วเปิด โปรแกรม Arduino IDE จากนั้นดำเนินการตามขั้นตอนดังนี้

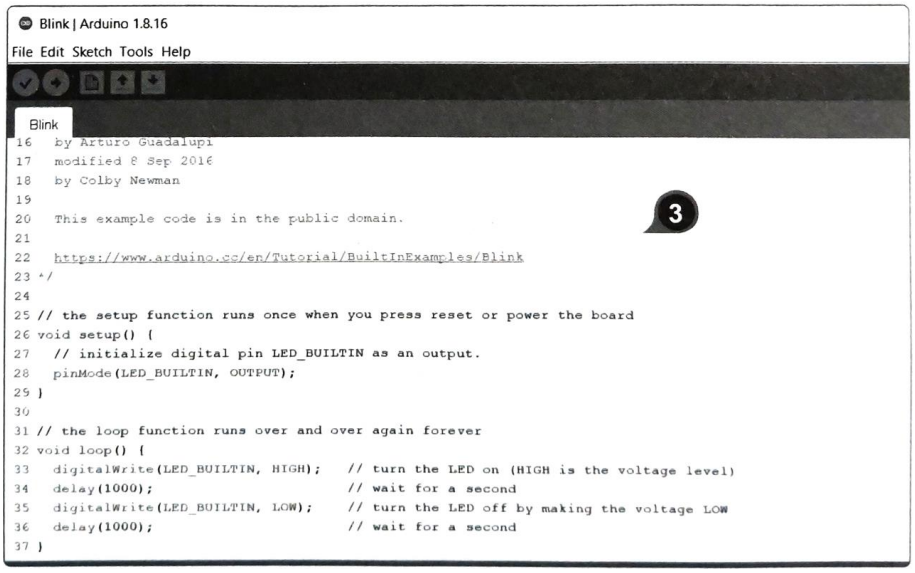
1. ไปที่เมนู Tools เลือก บอร์ด (Board) และ พอร์ต (Port) ให้ตรงกับที่ใช้งาน



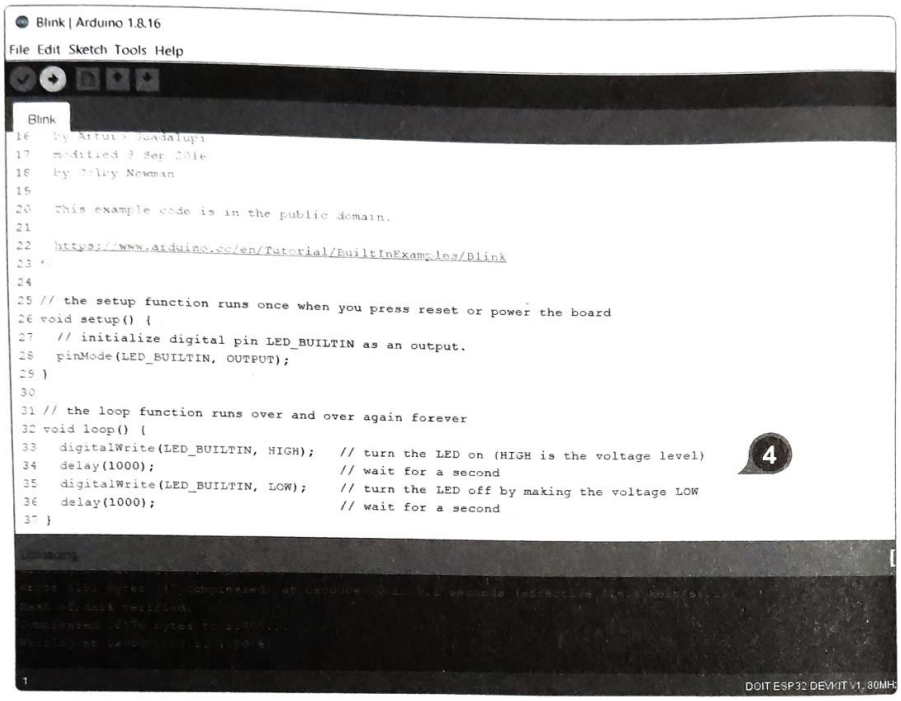
2. ไปที่เมนู File >> Examples >> Basics man Blink



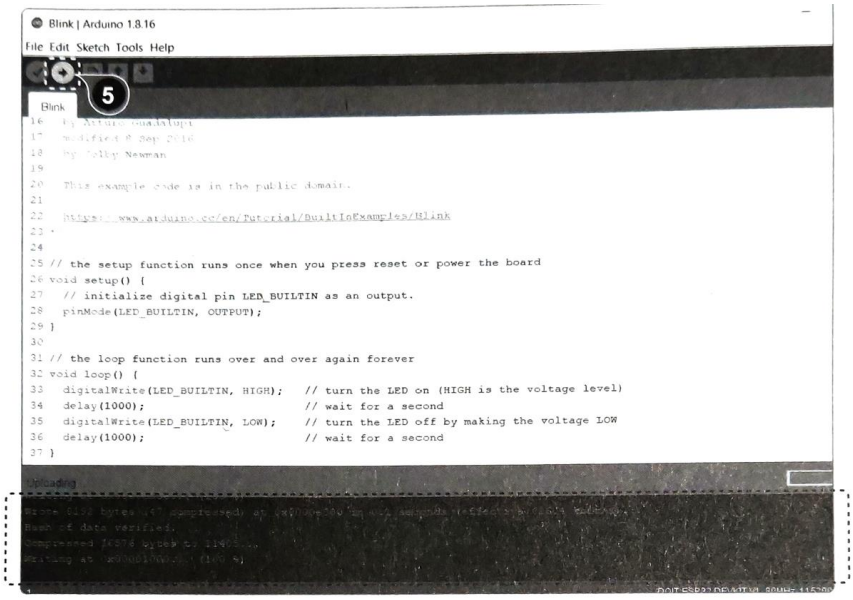
3. ตัวอย่างโค้ดโปรแกรม Blink จะถูกเปิดขึ้นมา ดังรูป



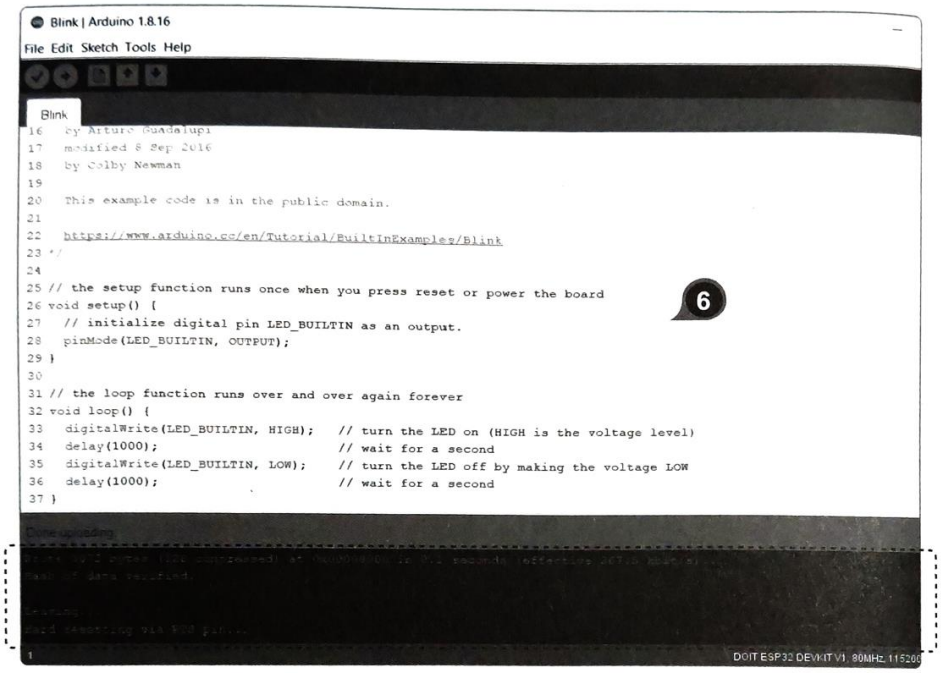
4. ในที่นี้จากโค้ดตรงตำแหน่งขาเราจะกำหนดเป็น LED BUILTIN หรือ 2 (GPIO2) ก็ได้ เพราะ LED BUILTIN เป็นขาที่ถูกกำหนดให้เชื่อมต่ออยู่กับหลอดไฟ LED แบบ built-in ที่ติดมากับบอร์ดอยู่แล้ว (แต่ถ้าจะเปลี่ยนเป็น 2 ก็ต้องเปลี่ยนทั้ง 3 จุด)



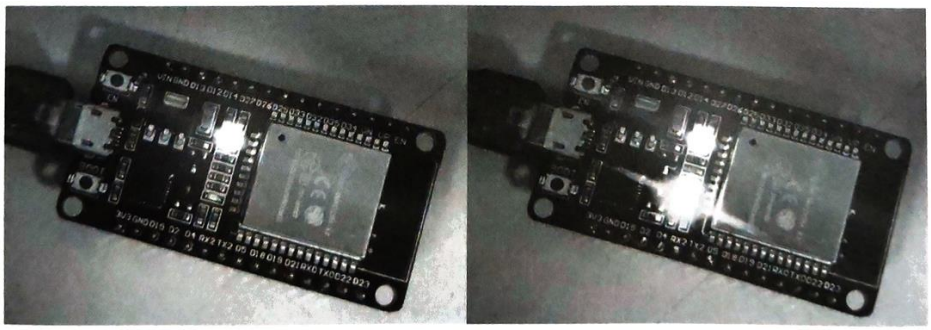
5. คลิกปุ่ม Upload รอสักครู่ ระหว่างนี้โค้ดโปรแกรมจะถูกตรวจสอบ (Verify) และอัพโหลด (Upload) ไปยังบอร์ด ซึ่งเราสามารถที่จะมองเห็นข้อความแสดงสถานะการทำงาน หรือข้อผิดพลาด (Error) ที่เกิดขึ้นได้จากกรอบแสดงผลด้านล่าง ดังรูปหน้าถัดไป



6. เมื่อกระบวนการอัพโหลดเสร็จสมบูรณ์ครบ 100% หลอดไฟ LED สีฟ้าที่อยู่บนบอร์ดจะกะพริบ (ติด...ดับ.. ติด...ดับ) สลับกันไปอย่างต่อเนื่องทุกๆ 1 วินาที (คำสั่ง delay กำหนดไว้ 1000 ms หรือ 1s) ดังรูป

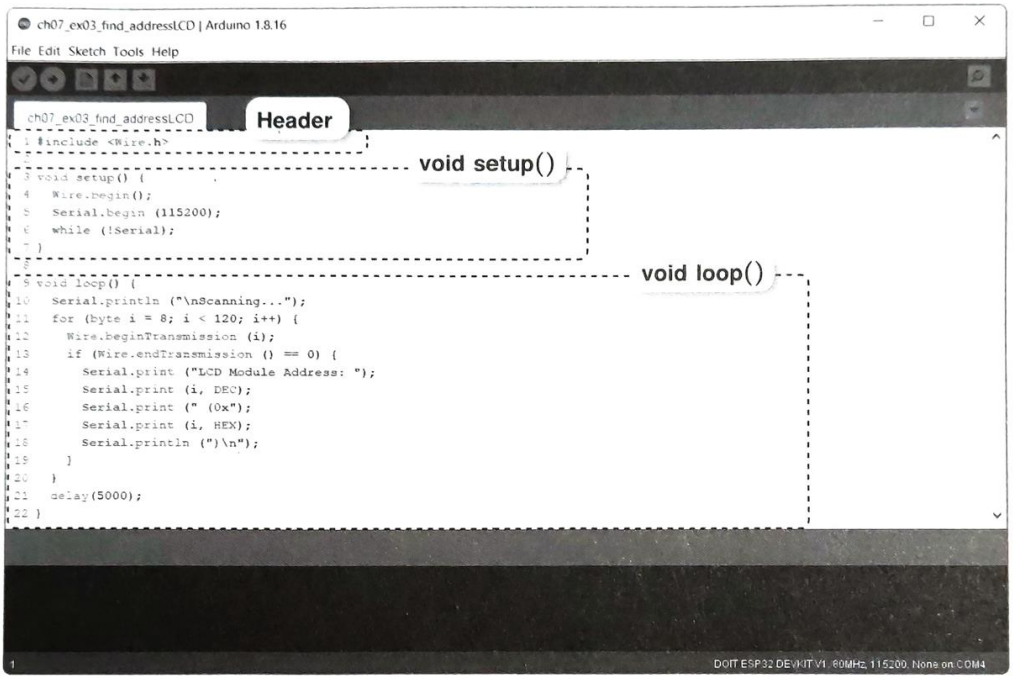


หลังจากนี้หากเราถอดสาย USB บอร์ดก็จะหยุดทำงาน เพราะไม่มีไฟเลี้ยงจ่ายให้กับบอร์ด แต่เราสามารถ นำเอาหัว USB Type A ฝั่งที่เสียบกับคอมพิวเตอร์ ไปเสียบกับแหล่งจ่ายไฟอื่นๆ อย่าง Power Bank หรือ Power Adapter เพื่อจ่ายไฟเลี้ยงให้กับบอร์ดได้ โดยบอร์ดจะเริ่มทำงานใหม่อีกครั้งวนไปเรื่อยๆ ตามโค้ดโปรแกรมที่เรา เขียนและอัพโหลดเข้าไปไว้ในชิปไมโครคอนโทรลเลอร์ที่อยู่บนบอร์ด



โครงสร้างภาษา C สำหรับการเขียนโค้ดโปรแกรม

การเขียนโค้ดโปรแกรมด้วยภาษา C สำหรับ Arduino เราจะแบ่งโครงสร้างหลักๆ ออกเป็น 3 ส่วน ดังนี้



Header

เป็นส่วนหัวของโค้ดโปรแกรมที่รวบรวมคำสั่งเพื่อการเรียกใช้ฟังก์ชั่นหรือตัวแปรจากไฟล์ Libraries, การ ประกาศตัวแปร และค่าคงที่ต่างๆ เพื่อให้ระบบเข้าใจถึงตัวแปรและค่าคงที่เหล่านั้น ซึ่งโดยปกติแล้วในส่วน ของ Header จะมีหรือไม่มีก็ได้ (ไม่บังคับ) เนื่องจากการประกาศตัวแปร และค่าคงที่ต่างๆ ตลอดจนการกำหนด สถานะ Input/Output ให้กับขาต่างๆ เราสามารถที่จะเขียนไว้ในส่วนของ void setup() ก็ได้ ส่วนข้อความใดๆ ที่อยู่หลังเครื่องหมาย // จะเป็นคอมเมนท์หรือส่วนอธิบายที่เราสามารถที่จะเขียนบันทึกอะไรลงไปก็ได้ โดยที่มัน จะไม่ถูกกันไปพร้อมๆ กับโปรแกรมด้วย

void setup()

เป็นฟังก์ชั่นบังคับที่จะต้องมีอยู่ในทุกๆ โค้ดโปรแกรม โดยจะเป็นส่วนที่ใช้ใส่คำสั่งที่ต้องการให้โปรแกรม รันการทำงานเพียงรอบเดียวนับตั้งแต่บอร์ดเริ่มทำงาน (เริ่มจ่ายไฟเลี้ยงหรือรีเซ็ตบอร์ด) เช่น คำสั่งที่เกี่ยวกับ การเซ็ตอัพหรือตั้งค่าการทำงานให้กับอุปกรณ์ต่างๆ ที่เชื่อมต่อ และการใช้คำสั่ง pinMode เพื่อกำหนดสถานะ Input/Output ให้กับขาต่างๆ เป็นต้น

void loop()

เป็นอีกหนึ่งฟังก์ชั่นบังคับที่จะต้องมีอยู่ในทุกๆ โค้ดโปรแกรมเช่นกัน โดยหลังจากรันคำสั่งที่ void setup() เพียงรอบเดียวเสร็จแล้ว ก็จะมาที่ void loop() นี้ ซึ่งจะเป็นส่วนที่ใช้ใส่คำสั่งและเงื่อนไขต่างๆ ที่ต้องการให้ โปรแกรมรันการทำงานแบบวนซ้ำไปเรื่อยๆ จนกว่าจะมีคำสั่งให้ออกจาก loop หรือจนกว่าบอร์ดจะหยุดทำงาน มาดูตัวอย่างการทำงานในแต่ละบรรทัดคำสั่งของโค้ดโปรแกรม Blink หรือไฟกะพริบกัน

บรรทัดคำสั่ง	การทำงาน
void setup()	เป็นฟังก์ชั่นที่ใช้กำหนดค่าเริ่มต้นต่างๆ ซึ่งจะรันเพียงครั้งเดียว
pinMode(LED_BUILTIN, OUTPUT);	กำหนดให้ขา D2 ที่ต่ออยู่กับหลอดไฟ LED บนบอร์ด มีสถานะเป็น Output (ส่งสัญญาณออก)
void loop()	เป็นฟังก์ชั่นที่จะมีคำสั่งและเงื่อนไขต่างๆ ซึ่งจะรันแบบวนซ้ำไปเรื่อยๆ จนกว่าจะมีคำสั่งให้ออกหรือบอร์ดหยุดทำงาน
digitalWrite(LED_BUILTIN, HIGH);	สั่งให้ขา D2 มีสถานะเป็น HIGH หรือมีค่าแรงดันไฟเป็น 3.3V ซึ่งในที่นี้จะทำให้หลอดไฟ LED บนบอร์ดที่ต่ออยู่กับขา D2 ติด
delay(1000);	สั่งให้หน่วงเวลารอ 1000 ms หรือ 1 วินาที (สถานะที่ขา D2 ยังคงเป็น HIGH หลอดไฟ LED ยังคงติดค้างอยู่เป็นเวลา 1 วินาที)

การใช้ Serial Monitor

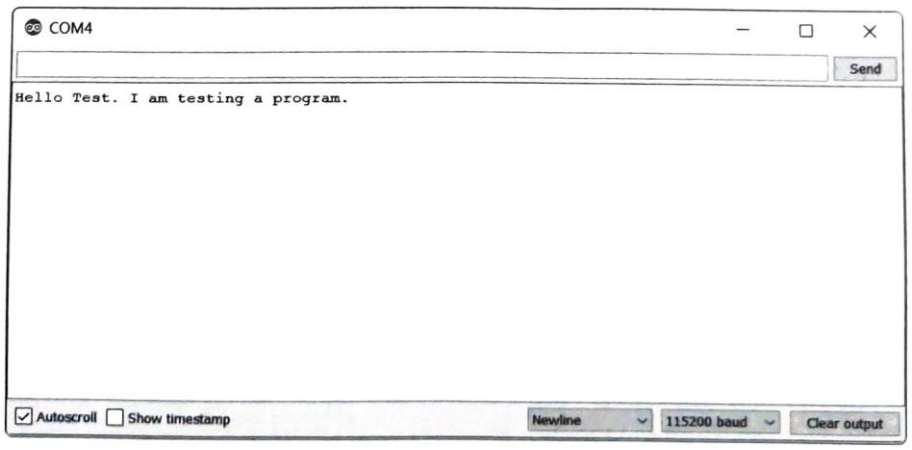
Serial Monitor เป็นเครื่องมือหรือหน้าต่างที่เอาไว้แสดงข้อมูลหรือผลลัพธ์ที่ได้จากการตรวจสอบการทำงานของโปรแกรม โดยเราสามารถตรวจดูข้อมูลที่ใช้รับส่งกันระหว่างคอมพิวเตอร์กับบอร์ดได้ เช่น การดูค่าที่อ่านได้จากเซ็นเซอร์, การตรวจดูเอาต์พุตที่จุดต่างๆ, การตรวจดูค่าตัวแปรในโปรแกรม ฯลฯ

การรับส่งข้อมูลระหว่างคอมพิวเตอร์หรือหน้าต่าง Serial Monitor กับบอร์ด จะเป็นการรับส่งข้อมูลแบบ อนุกรม ดังนั้นจึงต้องมีการกำหนดค่าความเร็วของ Baud Rate หรือ Upload Speed ที่มีหน่วยเป็นจำนวนบิต ต่อวินาที (bps) ให้ตรงกันด้วย ซึ่งปกติแล้วค่าความเร็วมาตรฐานจะอยู่ที่ 9600 แต่ก็สามารถเลือกใช้ค่าความเร็ว ที่สูงกว่านี้ได้ เช่น 115200 ซึ่งเป็นค่าความเร็วสูงสุดที่แนะนำ หากเกินกว่านี้การรับส่งข้อมูลอาจเกิดความ ผิดพลาดได้ง่าย แต่เงื่อนไขที่สำคัญคือ ต้องกำหนดค่าความเร็วทั้งจากตัวโปรแกรมที่ระบุไว้ในโค้ดตรงบรรทัด คำสั่ง Serial.begin() และในหน้าต่าง Serial Monitor ให้ตรงกัน ไม่อย่างนั้น Serial Monitor จะไม่ทำงาน

รูปแบบการใช้คำสั่งเพื่อดูข้อมูลที่ Serial Monitor

```
void setup() {
  Serial.begin(115200);                  //เริ่มการรับส่งข้อมูลด้วยความเร็ว Baud Rate 115200
  delay(100);
  Serial.print("Hello Test. ");          //พิมพ์ข้อความใน " " เว้นวรรค และไม่ต้องขึ้นบรรทัดใหม่
  Serial.print("I am testing a program."); //พิมพ์ข้อความใน " " แล้วไม่ต้องขึ้นบรรทัดใหม่
}
void loop() {

}
```

ผลลัพธ์บนหน้าจอ Serial Monitor



รูปแบบการใช้คำสั่งเพื่อดูข้อมูลที่ Serial Monitor (เพิ่มเติม)

```
void setup() {
  Serial.begin(115200);              //เริ่มการรับส่งข้อมูลด้วยความเร็ว Baud Rate 115200
  delay(100);
  Serial.print("Hello Test. ");     //พิมพ์ข้อความใน " " เว้นวรรค และไม่ต้องขึ้นบรรทัดใหม่
  Serial.println("I am testing a program.");    //พิมพ์ข้อความใน " " แล้วขึ้นบรรทัดใหม่
  int a, b, c;                                  //ประกาศตัวแปร a, b และ c เป็นตัวเลขจำนวนเต็ม
  a = 10;                                       //ให้ตัวแปร a มีค่าเท่ากับ 10
  b = 20;                                       //ให้ตัวแปร b มีค่าเท่ากับ 20
  c = a+b;                                      //ให้ตัวแปร c มีค่าเท่ากับ a บวก b
  Serial.print("Value a = ");                   //พิมพ์ข้อความใน " " เว้นวรรค และไม่ต้องขึ้นบรรทัดใหม่
  Serial.println(a);                            //พิมพ์ค่าตัวแปร a แล้วขึ้นบรรทัดใหม
  Serial.print("Value b = ");                   //พิมพ์ข้อความใน " " เว้นวรรค และไม่ต้องขึ้นบรรทัดใหม่
  Serial.println(b);                            //พิมพ์ค่าตัวแปร b แล้วขึ้นบรรทัดใหม่
  Serial.print("Value c = ");                   //พิมพ์ข้อความใน " " เว้นวรรค และไม่ต้องขึ้นบรรทัดใหม่
  Serial.println(c);                            //พิมพ์ค่าตัวแปร c แล้วขึ้นบรรทัดใหม่
}

void loop() {

}
```

บันทึกผลการทดลอง

.....

.....

.....

.....

การทดลองที่ 2 การใช้ Serial Monitor ดูข้อมูลที่ส่งออกจากบอร์ด

1. หลังจากเชื่อมต่อบอร์ดเข้ากับคอมพิวเตอร์ เปิด Arduino IDE แล้วไปที่เมนู Tools เลือก บอร์ด (Board), ความเร็วอัพโหลด (Upload Speed) และ พอร์ต (Port) ให้ตรงกับที่ใช้งาน

2. เปิดตัวอย่างโค้ดโปรแกรม Blink ขึ้นมา แล้วแก้ไขเพิ่มเติมคำสั่งดังนี้

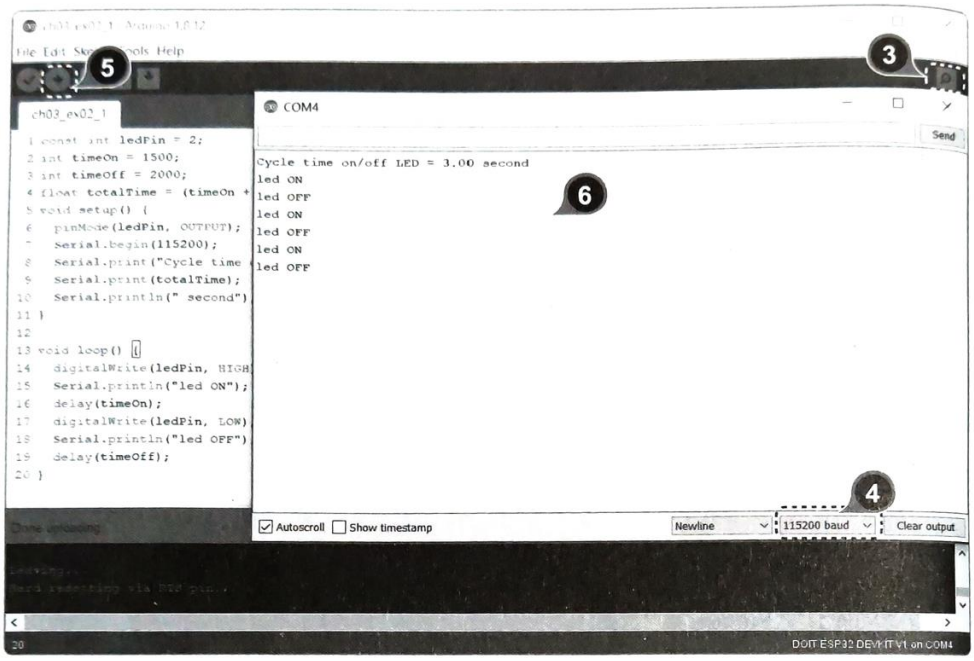
```
const int ledPin = 2;              //ประกาศตัวแปรขา D2/GPIO2 ที่ต่อกับหลอดไฟ LED เป็นค่าคงที่
int timeOn = 1500;                 //ประกาศตัวแปรและกำหนดระยะเวลาที่หลอดไฟ LED ติด
int timeOff = 2000;                //ประกาศตัวแปรและกำหนดระยะเวลาที่หลอดไฟ LED ดับ
float totalTime = (timeOn + timeOff)/1000;    //ประกาศตัวแปรของวงรอบระยะเวลาที่หลอดไฟ
                                              LED ติดและดับ เป็นค่าตัวเลขที่มีจุดทศนิยมแสดงเป็นวินาที

void setup() {
  pinMode(ledPin, OUTPUT);         //กำหนดให้ขา ledPin เป็น Output
  Serial.begin(115200);            //เริ่มการรับส่งข้อมูลด้วยความเร็ว Baud Rate 115200
  delay(100);
  Serial.print("Cycle time on/off LED = ");  //พิมพ์ข้อความใน " " เว้นวรรค และไม่ต้อง
                                             ขึ้นบรรทัดใหม่
  Serial.print(totalTime);         //พิมพ์ค่าตัวแปร totalTime และไม่ต้องขึ้นบรรทัดใหม่
  Serial.println(" second");       //พิมพ์ข้อความใน " " แล้วขึ้นบรรทัดใหม่
}

void loop() {
  digitalWrite(ledPin, HIGH);      //กำหนดให้ Output ที่ขา ledPin เป็น HIGH หลอดไฟติด
  Serial.println("led ON");        //พิมพ์ข้อความใน " " แสดงสถานะหลอดไฟติด
  delay(timeOn);                   //หน่วงรอเป็นระยะเวลาเท่ากับ timeOn
  digitalWrite(ledPin, LOW);       //กำหนดให้ Output ที่ขา ledPin เป็น LOW หลอดไฟดับ
  Serial.println("led OFF");       //พิมพ์ข้อความใน " " แสดงสถานะหลอดไฟดับ
  delay(timeOff);                  //หน่วงรอเป็นระยะเวลาเท่ากับ timeOff
}
```

3. คลิกปุ่ม + หรือคลิกที่เมนู Tools » Serial Monitor หรือกดคีย์ลัด Ctrl + Shift + A เพื่อเปิดหน้าจอSerial Monitor

4. บนหน้าจอ Serial Monitor คลิกเลือกค่าความเร็ว Baud Rate ซึ่งต้องเลือกให้ตรงค่าในโค้ดโปรแกรม ตรงบรรทัด Serial.begin() ในที่นี้คือ 115200

5. คลิกปุ่ม Upload รอสักครู่

6. หากไม่มีข้อผิดพลาดเมื่ออัพโหลดเสร็จ จะแสดงผลลัพธ์บนหน้าจอ Serial Monitor ดังรูปถัดไป



บันทึกผลการทดลอง

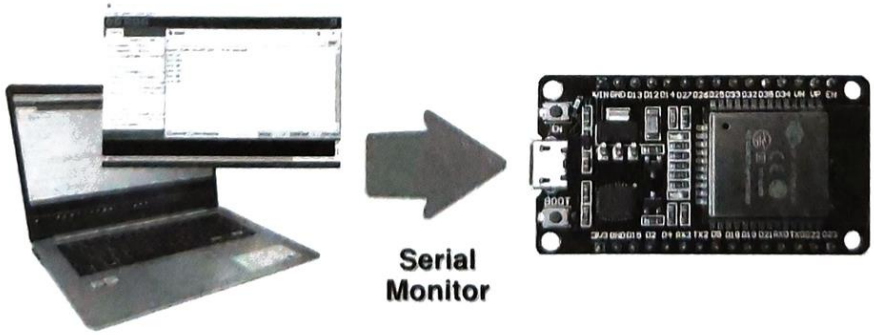
.....

.....

.....

.....

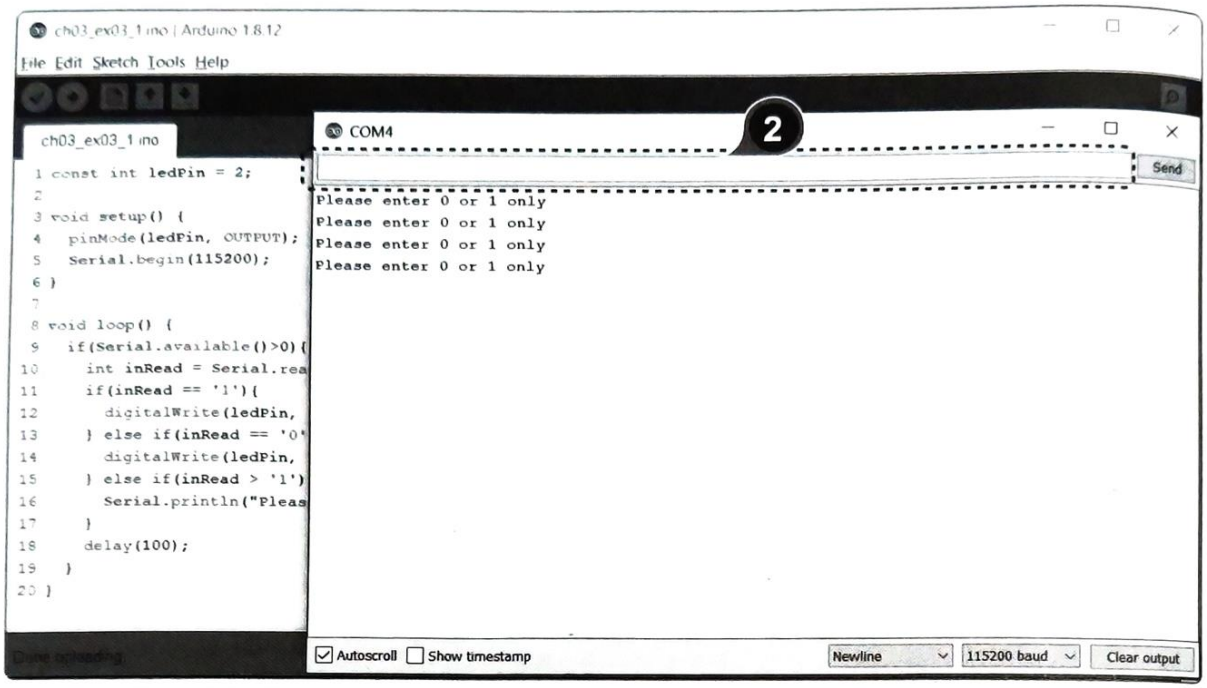
การทดลองที่ 3 การใช้ Serial Monitor ส่งข้อมูลจากแป้นพิมพ์ไปยังบอร์ด



1. นำโค้ดโปรแกรมจากตัวอย่างก่อนหน้านี้มาแก้ไขเพิ่มเติม

```
const int ledPin = 2;              //ประกาศตัวแปรขา D2/GPIO2 ที่ต่อกับหลอดไฟ LED เป็นค่าคงที่

void setup() {
  pinMode(ledPin, OUTPUT);         //กำหนดให้ขา ledPin เป็น Output
  Serial.begin(115200);            //เริ่มการรับส่งข้อมูลด้วยความเร็ว Baud Rate 115200
}

void loop() {
  if(Serial.available()>0){        //ตรวจสอบว่ามีข้อมูลส่งมาจาก Serial Monitor หรือไม่
    int inRead = Serial.read();    //อ่านค่าจาก Serial Monitor เก็บไว้ในตัวแปร
    if(inRead == '1'){             //ตรวจสอบเงื่อนไข ถ้าข้อมูลที่ส่งมาหรือค่าที่อ่านได้เป็น 1
     digitalWrite(ledPin, HIGH);   //ให้ Output ที่ขา ledPin มีสถานะเป็น HIGH หลอดไฟติด
    } else if(inRead == '0'){      //แต่ถ้าข้อมูลที่ส่งมาหรือค่าที่อ่านได้เป็น 0
      digitalWrite(ledPin, LOW);   //ให้ Output ที่ขา ledPin มีสถานะเป็น LOW หลอดไฟดับ
    } else if(inRead > '1'){       //แต่ถ้าข้อมูลที่ส่งมาหรือค่าที่อ่านได้มากกว่า 1
      Serial.println("Please enter 0 or 1 only");    //ให้พิมพ์ข้อความใน " "
                                                     แล้วขึ้นบรรทัดใหม่
    }
    delay(100);                    //หน่วงเวลารอก่อนที่จะวนคำสั่งซ้ำ
  }
}
```

2. เมื่ออัพโหลดเสร็จ ที่หน้าต่าง Serial Monitor ให้เราทดลองส่งข้อมูลไปยังบอร์ดด้วยการกดคีย์ตัวเลข 0 หรือ 1 บนแป้นพิมพ์ แล้วกดคีย์ Enter สังเกตที่หลอดไฟ LED บนบอร์ด เมื่อกดคีย์ตัวเลข 0 หลอดไฟจะดับ แต่ถ้าหากกดคีย์ตัวเลข 1 หลอดไฟจะติด ส่วนการกดคีย์ตัวเลขอื่นๆ จะแสดงข้อความบอกให้ใส่ค่าเฉพาะตัวเลข 0 หรือ 1 เท่านั้น ดังรูป (หน้าถัดไป) ซึ่งทั้งหมดก็เป็นไปตามเงื่อนไขที่กำหนด



บันทึกผลการทดลอง

.....

.....

.....

.....

การทดลองที่ 4 การใช้งาน Serial Plotter

1. นำโค้ดโปรแกรมจากตัวอย่าง การทดลองที่ 2 มาแก้ไขเพิ่มเติม

```
const int ledPin = 2;              //ประกาศตัวแปรขา D2/GPIO2 ที่ต่อกับหลอดไฟ LED เป็นค่าคงที่
int timeOn = 1000;                 //ประกาศตัวแปรและกำหนดระยะเวลาที่หลอดไฟ LED ติด
int timeOff = 2000;                //ประกาศตัวแปรและกำหนดระยะเวลาที่หลอดไฟ LED ดับ

void setup() {
  pinMode(ledPin, OUTPUT);         //กำหนดให้ขา ledPin เป็น Output
  Serial.begin(115200);            //เริ่มการรับส่งข้อมูลด้วยความเร็ว Baud Rate 115200
}

void loop() {
  digitalWrite(ledPin, HIGH);      //กำหนดให้ Output ที่ขา ledPin เป็น HIGH หลอดไฟติด
  Serial.println(1);               //พิมพ์ค่า 1 นำไปพล็อตกราฟแสดงสถานะหลอดไฟติด
  delay(timeOn);                   //หน่วงรอเป็นระยะเวลาเท่ากับ timeOn
  digitalWrite(ledPin, LOW);       //กำหนดให้ Output ที่ขา ledPin เป็น LOW หลอดไฟดับ
  Serial.println(0);               //พิมพ์ค่า 0 นำไปพล็อตกราฟแสดงสถานะหลอดไฟดับ
  delay(timeOff);                  //หน่วงรอเป็นระยะเวลาเท่ากับ timeOff
}
```

2. เปิดหน้าต่าง Serial Plotter โดยคลิกที่เมนู Tools » Serial Plotter หรือกดคีย์ Ctrl + Shift + (หากเปิดหน้าต่าง Serial Monitor ไว้ จำเป็นต้องปิดก่อน)

3. บนหน้าจอ Serial Plotter คลิกเลือกค่าความเร็ว Baud Rate ซึ่งต้องเลือกให้ตรงค่าในโค้ดโปรแกรม ตรงบรรทัด Serial.begin() ในที่นี้คือ 115200

4. คลิกปุ่ม Upload รอสักครู่

5. เมื่ออัพโหลดเสร็จ จะแสดงผลลัพธ์เป็นกราฟเส้นแสดงสถานะการเปิด/ปิดของหลอดไฟ LED บนหน้าจอ Serial Plotter ดังรูป

บันทึกผลการทดลอง

.....

.....

.....

.....

สรุปผลการทดลอง

.....

.....

.....

.....