	แผนการจัดการเรียนรู้	หน่วยที่1
	รหัสวิชา 21901-2007 วิชา เทคโนโลยีระบบสมองกลฝังตัวและไอโอที	สอนครั้งที่.....
	ชื่อหน่วยการเรียนรู้ พื้นฐาน ESP32 และการเขียนโปรแกรมเบื้องต้น	ทฤษฎี 2..ชม. ปฏิบัติ 6..ชม.
ชื่อเรื่อง/งาน โปรแกรมควบคุม LED 7 หลอด ทำงานตามเงื่อนไข		

### 1. ผลลัพธ์การเรียนรู้ระดับหน่วยการเรียนรู้

ผู้เรียนสามารถเขียนโปรแกรมบน ESP32 เพื่อควบคุม LED 7 หลอดให้ทำงานตามเงื่อนไขที่กำหนดได้อย่างถูกต้อง และอธิบายหลักการทำงานของคำสั่งที่ใช้ได้

### 2. อ้างอิงมาตรฐาน/เชื่อมโยงกลุ่มอาชีพ

-

### 3. สมรรถนะประจำหน่วย

3.1 โปรแกรมควบคุม LED 7 หลอด ทำงานตามเงื่อนไข

### 4. จุดประสงค์เชิงพฤติกรรม

- 4.1 อธิบายโครงสร้างพื้นฐานของบอร์ด ESP32 และหมายเลขพินสำหรับควบคุม LED ได้อย่างถูกต้อง
- 4.2 เขียนโปรแกรมควบคุม LED 7 หลอดให้ทำงานตามเงื่อนไข (เช่น กะพริบสลับ, วิ่งไล่, แสดงผลตามรูปแบบ) ได้อย่างถูกต้องและมีประสิทธิภาพ
- 4.3 มีกิจนิสัยการทำงานเป็นระบบ ตรงต่อเวลา และรับผิดชอบต่องานที่ได้รับมอบหมาย
- 4.4 ปฏิบัติงานด้านการเขียนโปรแกรมและการต่อวงจรตามมาตรฐานความปลอดภัยได้อย่างถูกต้อง

### 5. สารการเรียนรู้

5.1 ความรู้พื้นฐานเกี่ยวกับ ESP32 โครงสร้างและส่วนประกอบของบอร์ด ESP32 หมายเลขพิน GPIO และการกำหนดพินสำหรับ Input/Output

5.2 การควบคุม LED ด้วย ESP32 หลักการทำงานของ LED และการต่อวงจรพื้นฐาน คำสั่ง pinMode() สำหรับกำหนดโหมดพิน คำสั่ง digitalWrite() สำหรับส่งสัญญาณ HIGH/LOW

5.3 โครงสร้างคำสั่งพื้นฐานในภาษา C/C++ คำสั่งควบคุมแบบมีเงื่อนไข (if-else, switch-case) คำสั่งวนซ้ำ (for loop, while loop) ฟังก์ชัน delay() สำหรับหน่วงเวลา

5.4 การเขียนโปรแกรมควบคุม LED 7 หลอด การควบคุม LED แบบเดี่ยวและแบบหลายหลอด การสร้างรูปแบบการทำงาน (Pattern)

### 6. กิจกรรมการเรียนรู้

ชั้นสอน (ทฤษฎี 2 ชม.)

โครงสร้างและส่วนประกอบของ ESP32 พร้อมยกตัวอย่างการใช้งานพิน GPIO อธิบายหลักการ  
ทำงานของคำสั่ง pinMode(), digitalWrite() พร้อมตัวอย่างโค้ด อธิบายโครงสร้างคำสั่งควบคุม (if-  
else, switch-case) และลูป (for, while)

ขั้นปฏิบัติ (6 ชม.)

กิจกรรมที่ 1: การต่อวงจรและทดสอบ LED

- ผู้เรียนต่อวงจร **LED** 1 หลอดกับ **ESP32** ตามแผนผังที่กำหนด
- เขียนโปรแกรมควบคุม **LED** ให้กะพริบเปิด-ปิด
- ทดสอบและแก้ไขข้อผิดพลาด

กิจกรรมที่ 2: การควบคุม LED 7 หลอดแบบพื้นฐาน

- ผู้เรียนต่อวงจร **LED** 7 หลอดกับบอร์ด **ESP32**
- ทดสอบและปรับปรุงโปรแกรม

กิจกรรมที่ 3: การสร้างรูปแบบการทำงานตามเงื่อนไข

- ผู้เรียนออกแบบและเขียนโปรแกรมสร้างรูปแบบการทำงาน
- ใช้ **array** และ **loop** ในการควบคุม **LED** หลายหลอด
- ทดสอบและนำเสนอผลงาน

## 7. สื่อและแหล่งการเรียนรู้

เอกสารประกอบการสอน เรื่อง "พื้นฐาน ESP32 และการควบคุม LED"

สไลด์นำเสนอ

Video สาธิตการต่อวงจรและการเขียนโปรแกรม

ใบงาน/ใบความรู้

## 8. หลักฐานการเรียนรู้

### 8.1 หลักฐานความรู้

ใบงานทฤษฎี เรื่อง โครงสร้าง ESP32 และคำสั่งพื้นฐาน

แบบทดสอบก่อนเรียน-หลังเรียน

สมุดบันทึกการเรียนรู้ (Learning Log)

ใบความรู้ที่มีการจดบันทึกและสรุปประเด็นสำคัญ

### 8.2 หลักฐานการปฏิบัติงาน

ไฟล์โค้ดโปรแกรม (.ino) ที่สามารถทำงานได้ถูกต้อง

Video/ภาพถ่ายการทำงานของ LED 7 หลอดตามเงื่อนไขที่กำหนด

## 9. การวัดและประเมินผล

### 9.1 เกณฑ์การปฏิบัติงาน

ด้านความรู้ (40%)

อธิบายหลักการทำงานของ ESP32 และคำสั่งควบคุม LED ได้ถูกต้อง ครบถ้วน

ด้านทักษะ (40%)

ต่อวงจร LED 7 หลอดได้ถูกต้องตามหลักการ ไม่เกิดไฟช็อต

เขียนโปรแกรมควบคุม LED ได้อย่างถูกต้อง มีโครงสร้างเป็นระบบ  
ด้านเจตคติและกิจนิสัย (20%)

มีความรับผิดชอบต่อการใช้อุปกรณ์และส่งงานตรงเวลา

## 9.2 วิธีการประเมิน

สังเกตพฤติกรรมการทำงานระหว่างเรียน

ประเมินจากชิ้นงานและไฟล์โค้ดโปรแกรม

ประเมินจากรายงานผลการปฏิบัติงาน

## 9.3 เครื่องมือประเมิน

แบบประเมินทักษะการปฏิบัติงาน

Check List การส่งงาน

แบบประเมินการนำเสนอผลงาน (Presentation Rubric)

# 10. บันทึกผลหลังการจัดการเรียนรู้

## 10.1 ผลการจัดการเรียนรู้ที่เกิดขึ้นกับผู้เรียน

ผู้เรียนส่วนใหญ่เข้าใจหลักการทำงานของคำสั่งพื้นฐานและสามารถประยุกต์ใช้ได้

ผู้เรียนแสดงความสนใจและกระตือรือร้นในการทดลองสร้างรูปแบบการทำงานใหม่ๆ

## 10.2 ปัญหา อุปสรรคที่พบ

ผู้เรียนบางคนมีพื้นฐานด้านการเขียนโปรแกรมน้อย ทำให้เข้าใจช้ากว่าเพื่อน

เวลาปฏิบัติไม่เพียงพอสำหรับผู้เรียนที่ต้องการทดลองเพิ่มเติม

ผู้เรียนบางคนไม่ส่งรายงานตรงเวลา

## 10.3 การแก้ไขปัญหา

### 1) ผลการแก้ไขปัญหาที่ส่งผลดีต่อผู้เรียน

จัดเตรียม Video Tutorial ให้ผู้เรียนสามารถทบทวนได้ด้วยตนเอง


เพิ่มเวลาห้องปฏิบัติการเปิดเพื่อให้ผู้เรียนมาฝึกฝนเพิ่มเติมนอกเวลาเรียน

เน้นย้ำการตรวจสอบวงจรก่อนเปิดไฟทุกครั้ง

### 2) แนวทางแก้ปัญหาในครั้งต่อไป

พัฒนาสื่อการเรียนรู้ออนไลน์เพิ่มเติม เช่น Interactive Simulation

กำหนดใหม่ไลน์การส่งงานที่ชัดเจนและติดตามอย่างใกล้ชิด

	ใบความรู้ ที่ 1	หน่วยที่ 1
	รหัสวิชา 21901-2007 วิชา เทคโนโลยีระบบสมองกลฝังตัวและไอโอที	สอนครั้งที่ 1-2
	ชื่อหน่วยการเรียนรู้ พื้นฐาน ESP32 และการเขียนโปรแกรมเบื้องต้น	ทฤษฎี 2.ชม. ปฏิบัติ 6.ชม.
ชื่อเรื่อง โปรแกรมควบคุม LED 7 หลอด ทำงานตามเงื่อนไข		

### 1. ผลลัพธ์การเรียนรู้ระดับหน่วยการเรียนรู้

ผู้เรียนสามารถเขียนโปรแกรมบน ESP32 เพื่อควบคุม LED 7 หลอดให้ทำงานตามเงื่อนไขที่กำหนดได้อย่างถูกต้อง และอธิบายหลักการทำงานของคำสั่งที่ใช้ได้

### 2. อ้างอิงมาตรฐาน/เชื่อมโยงกลุ่มอาชีพ

:-

### 3. สมรรถนะประจำหน่วย

3.1 โปรแกรมควบคุม LED 7 หลอด ทำงานตามเงื่อนไข

### 4. จุดประสงค์เชิงพฤติกรรม

4.1 อธิบายโครงสร้างโปรแกรมพื้นฐานใน Arduino IDE ได้

4.2 เขียนโปรแกรมควบคุม LED 7 หลอดให้ทำงานตามเงื่อนไขได้

4.3 ใช้ตัวแปรและโครงสร้างควบคุม (if, for, while) ในการเขียนโปรแกรมได้

4.4 ทดสอบและปรับแก้โค้ดให้ทำงานได้ถูกต้อง

### 5. เนื้อหาสาระ

ชั้นปฏิบัติ (16 ชม.)

#### 1. ESP 32 คืออะไร

ESP32 เป็นไมโครคอนโทรลเลอร์ที่พัฒนาโดย Espressif Systems ซึ่งมีฟีเจอร์ที่เหนือกว่าตัว Arduino ทั่วไปในหลายๆ ด้าน เช่น การเชื่อมต่อ Wi-Fi และ Bluetooth ในตัว, ความสามารถในการประมวลผลที่สูงขึ้น และขนาดเล็กกว่า

#### 2. ประเภทขาใช้งาน

- ขาที่เป็นอินพุตอย่างเดียว

ขา GPIO34, GPIO35, GPIO36, GPIO39 เป็นอินพุตอย่างเดียว ไม่มีวงจร Pull-up, Pull-down รองรับคำสั่ง analogRead(), digitalRead() เท่านั้น คำสั่งอื่น เช่น digitalWrite(), PWM ไม่รองรับ

- Strapping Pins

Strapping Pins เป็นขาเกี่ยวข้องกับการทำงานของ ESP32 ในระหว่างการบูตโปรแกรม แบ่งเป็นขาที่ใช้งานแล้วอาจทำให้ ESP32 ไม่สามารถทำงานได้ กับขาที่ใช้งานได้ปกติแต่อาจมีสัญญาณรบกวนออกมาระหว่างบูตโปรแกรม ขาที่ไม่ควรนำมาใช้งานเลย ประกอบด้วยขา GPIO0, GPIO2 ซึ่งเกี่ยวข้องกับการบูตโปรแกรม ขาที่ใช้งานได้ปกติแต่อาจมีสัญญาณรบกวนออกมาระหว่างบูตโปรแกรม ประกอบด้วยขา GPIO12, GPIO15, GPIO5 ไม่ควรนำไปใช้ต่อกับอุปกรณ์ I2C เพราะอาจทำให้เกิดการทำงานผิดปกติของอุปกรณ์ I2C ได้

- ขาที่ต่อชิป Flash และ PSRAM

ขาที่ต่อชิป Flash เป็นขาที่ห้ามนำมาใช้เด็ดขาดเพราะกระทบกับการอ่านโปรแกรมที่เคียวโฮลด์ไว้ หรือทำให้ฮาร์ดแวร์โปรแกรมไม่เข้า ประกอบด้วยขา GPIO9, GPIO10, GPIO11, GPIO6, GPIO7, GPIO8 ขาที่ต่อชิป PSRAM เป็นขาที่เมื่อนำมาใช้จะไม่สามารถใช้งาน PSRAM ได้ ซึ่ง PSRAM จะมีในบอร์ดที่ระบุอย่างชัดเจนว่ามี PSRAM หากบอร์ดไม่มี PSRAM อยู่แล้วก็ไม่จำเป็นต้องสนใจ แต่หากบอร์ดมี PSRAM ต้องไม่นำขา GPIO16, GPIO17 มาใช้งาน หากนำมาใช้ต้องแน่ใจว่าโปรแกรมที่เขียนไม่เรียกใช้ PSRAM

- ขาฮาร์ดแวร์โปรแกรม

นอกจากขา Strapping Pins ที่เกี่ยวข้องกับการฮาร์ดแวร์โปรแกรมและการบูตโปรแกรม ยังมีขา RX, TX ที่ใช้ในการฮาร์ดแวร์โปรแกรม หากนำขาดังกล่าวไปใช้ จะทำให้ฮาร์ดแวร์โปรแกรมไม่ได้ (แต่โปรแกรมที่เคียวโฮลด์ไว้ยังทำงานได้ปกติ)

- ขา Digital Input

ขาที่ใช้อ่านค่าดิจิทัลได้ (ใช้คำสั่ง digitalRead() ได้) คือทุกขาที่ไม่ใช่ Strapping Pins, ขาที่ต่อชิป Flash และ PSRAM, ขาฮาร์ดแวร์โปรแกรม โดยสรุปคือขา GPIO0, GPIO2, GPIO9, GPIO10, GPIO11, GPIO6, GPIO7, GPIO8, RX, TX ใช้ไม่ได้ ที่เหลือใช้ได้หมด

- ขา Digital Output

ขาที่เขียนค่าดิจิทัลได้ (ใช้คำสั่ง digitalWrite() ได้) คือทุกขาที่ไม่ใช่ ขาที่เป็นอินพุตอย่างเดียว, Strapping Pins, ขาที่ต่อชิป Flash และ PSRAM, ขาฮาร์ดแวร์โปรแกรม โดยสรุปคือขา GPIO0, GPIO2, GPIO9, GPIO10, GPIO11, GPIO6, GPIO7, GPIO8, RX, TX, GPIO34, GPIO35, GPIO36, GPIO39 ใช้ไม่ได้ ที่เหลือใช้ได้หมด

- ขา Analog Input

ขาที่อ่านค่าแอนะล็อกได้ (ใช้คำสั่ง analogRead() ได้) อ่านค่าได้ 0 ถึง 3.3V แบ่งขาออกเป็น 2 ชุด คือ ADC1 และ ADC2 โดย ADC1 สามารถใช้งานได้เลย ไม่มีเงื่อนไขอะไร ส่วน ADC2 จะใช้งานได้เมื่อปิดใช้ WiFi เท่านั้น ADC1 ใช้งานได้เลย ประกอบด้วยขา GPIO36, GPIO39, GPIO34, GPIO35, GPIO32, GPIO33 ADC2 ต้องปิด WiFi จึงจะใช้งานได้ ประกอบด้วยขา GPIO25, GPIO26, GPIO27, GPIO14, GPIO12, GPIO13, GPIO15, GPIO4

- ขา PWM

ขาที่อ่านค่าแอนะล็อกได้ (ใช้คำสั่ง `analogRead()` ได้) อ่านค่าได้ 0 ถึง 3.3V แบ่งขาออกเป็น 2 ชุด คือ ADC1 และ ADC2 โดย ADC1 สามารถใช้งานได้เลย ไม่มีเงื่อนไขอะไร ส่วน ADC2 จะใช้งานได้เมื่อปิดใช้ WiFi เท่านั้น ADC1 ใช้งานได้เลย ประกอบด้วยขา GPIO36, GPIO39, GPIO34, GPIO35, GPIO32, GPIO33 ADC2 ต้องปิด WiFi จึงจะใช้งานได้ ประกอบด้วยขา GPIO25, GPIO26, GPIO27, GPIO14, GPIO12, GPIO13, GPIO15, GPIO4

- ขา I2C

I2C แบ่งเป็น 2 ชุด คือ I2C0 และ I2C1 โดยปกติใช้เฉพาะ I2C0 ซึ่ง I2C0 ค่าเริ่มต้นกำหนดให้อยู่ที่ ขา SDA = GPIO21, SCL = GPIO22 สามารถเปลี่ยนเป็นขาอื่นได้โดยกำหนดในคำสั่ง `Wire.begin()` ตัวอย่างการย้ายขา SCL ไปที่ GPIO4 และ SDA ไป GPIO5

- ขา SPI

SPI แบ่งเป็น 2 ชุด คือ HSPI และ VSPI โดยปกติใช้เฉพาะ VSPI ซึ่ง VSPI ค่าเริ่มต้นกำหนดให้อยู่ที่ ขา CS = GPIO5, SCK = GPIO18, MISO = GPIO19, MOSI = GPIO23 สามารถเปลี่ยนเป็นขาอื่นได้ โดยกำหนดในคำสั่ง `SPI.begin()` ตัวอย่างการย้ายขา VSPI ไปที่ SCK = GPIO25, MISO = GPIO26, MOSI = GPIO27 และ CS = GPIO15

- ขา UART

UART แบ่งเป็น 3 ชุด คือ Serial0, Serial1 และ Serial2 โดย Serial0 ใช้ฮาร์ดแวร์โปรแกรมและสื่อสารกับคอมพิวเตอร์ อยู่ที่ขา TX, RX ไม่ควรนำมาใช้ต่ออุปกรณ์อื่น ส่วน Serial1 และ Serial2 ใช้งานได้อิสระ

Serial1 ค่าเริ่มต้นกำหนดให้อยู่ที่ขา RX = GPIO26, TX = GPIO27 เปลี่ยนขาได้โดยใช้คำสั่ง `Serial1.setPins()`

Serial2 ค่าเริ่มต้นกำหนดให้อยู่ที่ขา RX = GPIO4, TX = GPIO25 เปลี่ยนขาได้โดยใช้คำสั่ง `Serial2.setPins()`

ขา TX, RX ที่ย้ายไปใช้ได้ คือทุกขาที่ไม่ใช่ ขาที่เป็นอินพุตอย่างเดียว, Strapping Pins, ขาที่ต่อชิป Flash และ PSRAM, ขาฮาร์ดแวร์โปรแกรม โดยสรุปคือขา GPIO0, GPIO2, GPIO9, GPIO10, GPIO11, GPIO6, GPIO7, GPIO8, RX, TX, GPIO34, GPIO35, GPIO36, GPIO39 ใช้ไม่ได้ ที่เหลือใช้ได้หมด

- ขา CAN

CAN bus ขา TX, RX ไม่ได้ต่อกับขาใด ๆ เป็นค่าเริ่มต้น ก่อนเริ่มใช้งาน CAN bus ต้องกำหนดขา TX, RX ลงในโค้ดโปรแกรมก่อน โดยขา TX, RX ที่ใช้ได้ คือทุกขาที่ไม่ใช่ ขาที่เป็นอินพุตอย่างเดียว, Strapping Pins, ขาที่ต่อชิป Flash และ PSRAM, ขาฮาร์ดแวร์โปรแกรม โดยสรุปคือขา GPIO0, GPIO2, GPIO9, GPIO10, GPIO11, GPIO6, GPIO7, GPIO8, RX, TX, GPIO34, GPIO35, GPIO36, GPIO39 ใช้ไม่ได้ ที่เหลือใช้ได้หมด

- ขา I2S

I2S แบ่งเป็น 2 ชุด คือ I2S0 และ I2S1 โดยขา DIN, DOUT, BCLK, WS ไม่ได้ต่อกับขาใด ๆ เป็นค่าเริ่มต้น ก่อนเริ่มใช้งาน I2S ต้องกำหนดขาทั้งหมดลงในโค้ดโปรแกรมก่อน โดยขาที่ใช้ได้ คือทุกขาที่ไม่ใช่ ขาที่เป็นอินพุตอย่างเดียว, Strapping Pins, ขาที่ต่อชิป Flash และ PSRAM, ขาอัพโหลดโปรแกรม โดยสรุปคือขา GPIO0, GPIO2, GPIO9, GPIO10, GPIO11, GPIO6, GPIO7, GPIO8, RX, TX, GPIO34, GPIO35, GPIO36, GPIO39 ใช้ไม่ได้ ที่เหลือใช้ได้หมด

GPIO	Digital Input	Digital Output	Analog Input	PWM	I2C / SPI / UART / CAN / I2S
IO36	✓	✗	✓	✗	✗
IO39	✓	✗	✓	✗	✗
IO34	✓	✗	✓	✗	✗
IO35	✓	✗	✓	✗	✗
IO32	✓	✓	✓	✓	✓
IO33	✓	✓	✓	✓	✓
IO25	✓	✓	—	✓	✓
IO26	✓	✓	—	✓	✓
IO27	✓	✓	—	✓	✓
IO14	✓	✓	—	✓	✓
IO12	✓	✓	—	✓	✓
IO13	✓	✓	—	✓	✓
IO9	✗	✗	✗	✗	✗
IO10	✗	✗	✗	✗	✗
IO11	✗	✗	✗	✗	✗
IO6	✗	✗	✗	✗	✗
IO7	✗	✗	✗	✗	✗
IO8	✗	✗	✗	✗	✗
IO15	✓	✓	—	✓	✓
IO2	—	—	✗	✗	✗
IO0	—	—	✗	✗	✗
IO4	✓	✓	—	✓	✓
IO16	✓	✓	✗	✓	✓

GPIO	Digital Input	Digital Output	Analog Input	PWM	I2C / SPI / UART / CAN / I2S
IO17					
IO5					
IO18					
IO19					
IO21					
RX					
TX					
IO22					
IO23					

### 3. ติดตั้ง Arduino IDE

- ดาวน์โหลด Arduino IDE จาก Arduino website
- ติดตั้งโปรแกรมโดยทำตามขั้นตอนใน install

### 4. ติดตั้ง ESP32 Board

- เปิด Arduino IDE -> ไปที่ "File" -> "Preferences"
- เพิ่ม URL: [https://dl.espressif.com/dl/package\\_esp32\\_index.json](https://dl.espressif.com/dl/package_esp32_index.json) ใน "Additional Boards Manager URLs"

### Manager URLs

- ไปที่ "Tools" -> "Board" -> "Board Manager"
- ค้นหา "ESP32" และติดตั้งแพ็คเกจ

### 5. สิ่งที่ต้องรู้ก่อนเริ่ม

#### 5.1 โครงสร้างโปรแกรมพื้นฐาน

```
// ฟังก์ชันเริ่มต้น ทำงานครั้งเดียวตอนเปิดบอร์ดหรือกด Reset
void setup() {
  // ใช้สำหรับกำหนดค่าเริ่มต้น เช่น ตั้งค่า pin, เริ่ม Serial
}
// ฟังก์ชันวนซ้ำ ทำงานไปเรื่อย ๆ
void loop() {
  // ใช้สำหรับคำสั่งที่ทำซ้ำ เช่น อ่านค่าเซนเซอร์ ควบคุม LED
}
```

#### 5.2 ตัวแปร (Variables)

#### 5.3 ตัวแปรแบบ Global (ใช้งานได้ทั้งโปรแกรม)



#### 5.4 ตัวแปรแบบ Local (ใช้งานเฉพาะในฟังก์ชัน)

```
void loop() {  
  int localVar = 5; // ใช้ได้เฉพาะใน loop()  
  Serial.println(localVar);  
}
```

#### 5.5 คำสั่งควบคุมขา GPIO

```
pinMode(pin, mode); // กำหนดโหมดของขา  
digitalWrite(pin, value); // เขียนค่า (HIGH / LOW)  
digitalRead(pin); // อ่านค่าดิจิทัล (0 / 1)  
analogRead(pin); // อ่านค่าอนาล็อก (0–4095 บน ESP32)  
analogWrite(pin, value); // เขียนค่า PWM (0–255)
```

#### 5.6 ฟังก์ชัน (Functions)

```
// สร้างฟังก์ชัน  
void blinkLED(int pin, int timeDelay) {  
  digitalWrite(pin, HIGH);  
  delay(timeDelay);  
  digitalWrite(pin, LOW);  
  delay(timeDelay);  
}  
  
void loop() {  
  blinkLED(led, 500); // เรียกใช้ฟังก์ชัน  
}
```

## พื้นฐานการเรียนรู้เขียนโปรแกรมควบคุมหลอด LED

Week1.ino

```
1  #define LED1 32
2  #define LED2 33
3  #define LED3 25
4  #define LED4 26
5  #define LED5 27
6  #define LED6 12
7  #define LED7 13
8  void setup() {
9      Serial.begin(115200);
10     pinMode(LED1, OUTPUT);
11     pinMode(LED2, OUTPUT);
12     pinMode(LED3, OUTPUT);
13     pinMode(LED4, OUTPUT);
14     pinMode(LED5, OUTPUT);
15     pinMode(LED6, OUTPUT);
16     pinMode(LED7, OUTPUT);
17 }
18
19 void loop() {
20     digitalWrite(LED1, 1);
21     digitalWrite(LED2, 1);
22     digitalWrite(LED3, 1);
23     digitalWrite(LED4, 1);
24     digitalWrite(LED5, 1);
25     digitalWrite(LED6, 1);
26     digitalWrite(LED7, 1);
27     delay(1000);
28     digitalWrite(LED1, 0);
29     digitalWrite(LED2, 0);
30     digitalWrite(LED3, 0);
31     digitalWrite(LED4, 0);
32     digitalWrite(LED5, 0);
33     digitalWrite(LED6, 0);
34     digitalWrite(LED7, 0);
35     delay(1000);
36
37 }
38
```

พื้นฐานการเรียนรู้เขียนโปรแกรมควบคุมหลอด LED โดยใช้งาน For loop

```
Week1-2.ino
1  #define NUM_LEDS 7
2  int leds[NUM_LEDS] = {32, 33, 25, 26, 27, 12, 13};
3
4  void setup() {
5      Serial.begin(115200);
6      // ตั้งค่า pinMode ทุกตัวใน array
7      for (int i = 0; i < NUM_LEDS; i++) {
8          pinMode(leds[i], OUTPUT);
9      }
10 }
11
12 void loop() {
13     // เปิด LED ทั้งหมด
14     for (int i = 0; i < NUM_LEDS; i++) {
15         digitalWrite(leds[i], 1);
16     }
17     delay(1000);
18
19     // ปิด LED ทั้งหมด
20     for (int i = 0; i < NUM_LEDS; i++) {
21         digitalWrite(leds[i], 0);
22     }
23     delay(1000);
24 }
25
```

พื้นฐานการเรียนรู้เขียนโปรแกรมควบคุมหลอด LED โดยใช้งาน For loop สร้างรูปแบบการทำงาน

```
Week2.ino
1  #define NUM_LEDS 7
2  int leds[NUM_LEDS] = {32, 33, 25, 26, 27, 12, 13};
3  void setup() {
4      Serial.begin(115200);
5      // ตั้งค่า pinMode ทุกตัวใน array
6      for (int i = 0; i < NUM_LEDS; i++) {
7          pinMode(leds[i], OUTPUT);
8      }
9  }
10
11 void loop() {
12     // เปิด LED ทั้งหมด
13     for (int i = 0; i < NUM_LEDS; i++) {
14         digitalWrite(leds[i], 1);
15     }
16     delay(1000);
17
18     // ปิด LED ทั้งหมด
19     for (int i = 0; i < NUM_LEDS; i++) {
20         digitalWrite(leds[i], 0);
21     }
22     delay(1000);
23
24     // ----- ลูกเล่นไฟวิ่ง -----
25     for (int i = 0; i < NUM_LEDS; i++) {
26         digitalWrite(leds[i], 1); // เปิดทีละดวง
27         delay(300);
28         digitalWrite(leds[i], 0); // ดับทีละดวง
29     }
30
31 }
32
```

6. แบบฝึกหัด/แบบทดสอบ


ใบงาน ที่ 1

7. เอกสารอ้างอิง (ชั้นหน้าใหม่)

-

8. ภาคผนวก (เฉลยแบบฝึกหัด เฉลยแบบทดสอบ ฯ)

-

	ใบงาน ที่ 1	หน่วยที่ 1
	รหัสวิชา 21901-2007 วิชา เทคโนโลยีระบบสมองกลฝังตัวและไอโอที	สอนครั้งที่ 1-2
	ชื่อหน่วยการเรียนรู้ พื้นฐาน ESP32 และการเขียนโปรแกรมเบื้องต้น	ทฤษฎี 2.ชม. ปฏิบัติ 6.ชม.
ชื่อเรื่อง โปรแกรมควบคุม LED 7 หลอด ทำงานตามเงื่อนไข		

# 1. ผลลัพธ์การเรียนรู้ระดับหน่วยการเรียนรู้

ผู้เรียนสามารถเขียนโปรแกรมบน ESP32 เพื่อควบคุม LED 7 หลอดให้ทำงานตามเงื่อนไขที่กำหนดได้อย่างถูกต้อง และอธิบายหลักการทำงานของคำสั่งที่ใช้ได้

# 2. อ้างอิงมาตรฐาน/เชื่อมโยงกลุ่มอาชีพ

-

# 3. สมรรถนะประจำหน่วย

3.1 โปรแกรมควบคุม LED 7 หลอด ทำงานตามเงื่อนไข

# 4. จุดประสงค์เชิงพฤติกรรม

4.1 อธิบายโครงสร้างโปรแกรมพื้นฐานใน Arduino IDE ได้

4.2 เขียนโปรแกรมควบคุม LED 7 หลอดให้ทำงานตามเงื่อนไขได้

4.3 ใช้ตัวแปรและโครงสร้างควบคุม (if, for, while) ในการเขียนโปรแกรมได้

4.4 ทดสอบและปรับแก้โค้ดให้ทำงานได้ถูกต้อง

# 5. เครื่องมือ วัสดุ และอุปกรณ์

5.1 Arduino IDE 2..3.6

5.2 ชุดฝึกปฏิบัติ

# 6. คำแนะนำ/ข้อควรระวัง

-

# 7. ขั้นตอนการปฏิบัติงาน

7.1 ต่อบอร์ด LED 7 หลอดเข้ากับ ESP32

7.2 เขียนโปรแกรมให้ LED ทำงานตามลำดับที่กำหนด

7.3 ทดสอบและปรับแก้โปรแกรม

# 8. สรุปและวิจารณ์ผล

-

## 9. การประเมินผล

### 1. แบบประเมินสมรรถนะงานภาคปฏิบัติ (6 คะแนน)

ความถูกต้องของการต่อวงจร

ความถูกต้องของโปรแกรม

ความสมบูรณ์ของผลลัพธ์

### 2. แบบสังเกตพฤติกรรมลักษณะนิสัยการทำงาน (4 คะแนน)

ตรงต่อเวลา

ความร่วมมือและวินัย

## 10. เอกสารอ้างอิง /เอกสารค้นคว้าเพิ่มเติม

-