DTI2301 การเขียนโปรแกรมควบคุมหุ่นยนต์เพื่อการศึกษา

SSRU Professionals

Control Robotic Programing for Education

บทที่ 3 ไมโครคอนโทลเลอร์ และการเขียนโปรแกรมควบคุม Arduino ด้วย Tinkercad

ผู้ช่วยศาสตราจารย์ ดร.ณัฐภัทร แก้วรัตนภัทร์
Asst.Prof. Dr.Nutthapat Kaewrattanapat

email nutthapat.ke@ssru.ac.th

รายวิชา DTC2301 การโปรแกรมควบคุมหุ่นยนต์เพื่อการศึกษา

3(2-2-5) หน่วยกิต

Control Robotic Programing for Education

คำอธิบายรายวิชา

แนวคิด ทฤษฎี ที่เกี่ยวข้องการเขียนโปรแกรมและการควบคุมหุ่นยนต์ หลักการส่วนประกอบ โครงสร้างและหน้าที่ของหุ่นยนต์ คุณสมบัติของโปรแกรมภาษาชนิดต่าง ๆ หลักการเบื้องต้น เกี่ยวกับองค์ประกอบ ลักษณะคำสั่ง การเขียนโปรแกรม ขั้นตอนวิธี การวิเคราะห์ การออกแบบการเขียนโปรแกรมควบคุมหุ่นยนต์เพื่อการศึกษา

Course Description

Principles theories associated with control robotic programing for education. Principles, component, structural of robotic. Computer language, Elements of computer language, Syntax, computer programing, Algorithms, Analysis and design control robotic programing for education.

ขอบเขตเนื้อหา

บทที่ 1 วิทยาการหุ่นยนต์เพื่อการศึกษา (ตอนที่ 1 และ ตอนที่ 2)

บทที่ 2 สะตีมศึกษากับการพัฒนานวัตกรหุ่นยนต์ (STEAM4INNOVATOR Certificate)

บทที่ 3 ไมโครคอนโทลเลอร์ และการเขียนโปรแกรมควบคุม อาดูอิโน่ Arduino ด้วย Tinkercad

บทที่ 4 การโปรแกรมควบคุมอาดูอิโน่ Arduino ร่วมกับจอ LCD

บทที่ 5 การโปรแกรมควบคุมอาดูอิโน่ Arduino ร่วมกับ มอเตอร์เซอร์โว

บทที่ 6 การโปรแกรมควบคุมอาดูอิโน่ Arduino ร่วมกับ เซนเซอร์ตรวจวัดระยะห่างด้วยคลื่นอัลตร้าโซนิค

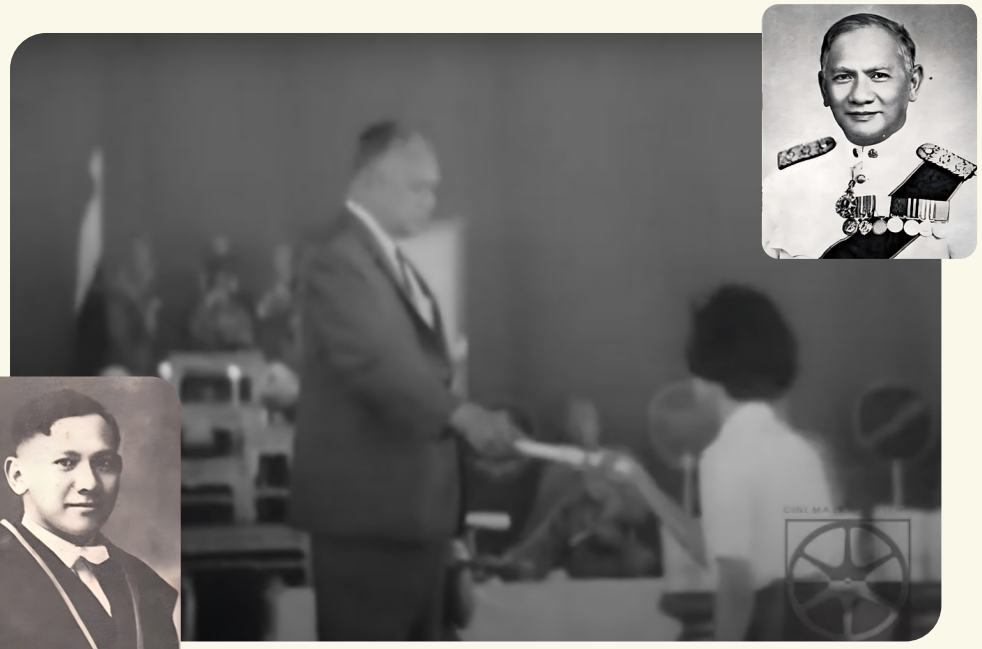
บทที่ 7 การโปรแกรมควบคุมอาดูอิโน่ Arduino ร่วมกับเซนเซอร์ตรวจวัดความชื้น

บทที่ 8 การพัฒนาโครงงานด้านวิทยาการหุ่นยนต์เพื่อการศึกษา

การวัดและการประเมินผลการเรียนรู้

การมีส่วนร่วมในชั้นเรียน	**10%**
การมอบหมายงาน	**30%**
สอบกลางภาค	**20%**
สอบปลายภาค	**20%**
โครงงาน	**20%**

เกรด	ช่วงคะแนน	ผลการประเมิน	ค่าระดับคะแนน
A	86 - 100	ดียอดเยี่ยม	4.00
A-	82 - 85	ดีเยี่ยม	3.75
B+	78 - 81	ดีมาก	3.50
B	74 - 77	ดี	3.00
B-	70 - 73	ค่อนข้างดี	2.75
C+	66 - 69	ปานกลางค่อนข้างดี	2.50
C	62 - 65	ปานกลาง	2.00
C-	58 - 61	ปานกลางค่อนข้างอ่อน	1.75
D+	54 - 57	ค่อนข้างอ่อน	1.50
D	50 - 53	อ่อน	1.00
D-	46 - 49	อ่อนมาก	0.75
F	0 - 45	ตก	0
I	รอการตัดเกรด		
W	ยกเลิกรายวิชา		



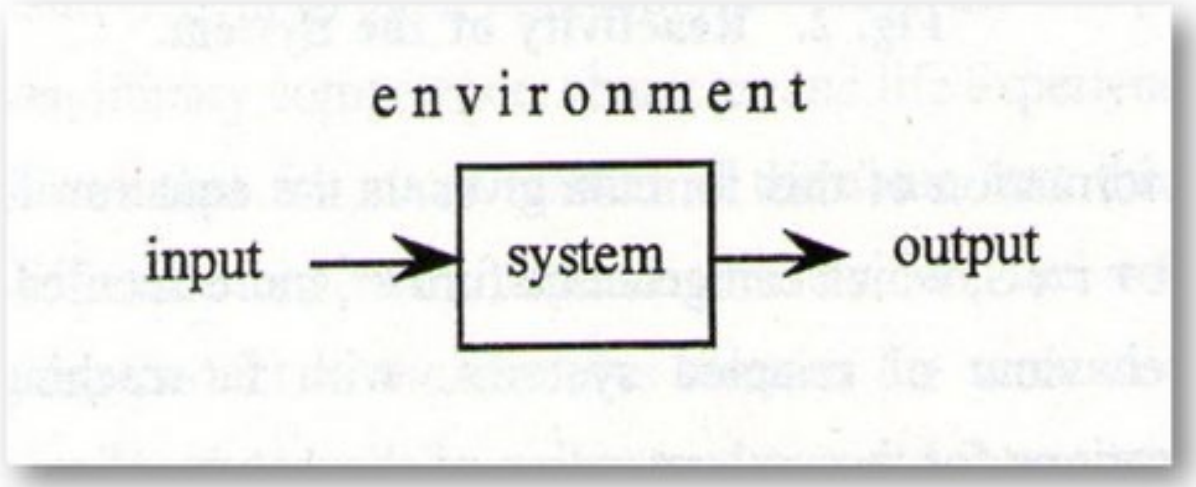
ม.ล.ปิ่น มาลากุล รัฐมนตรีว่าการกระทรวงศึกษาธิการ
ได้ไปเป็นประธานในพิธีแจกประกาศนียบัตรแก่นักเรียนและ
นักศึกษาวิทยาลัยครูสวนสุนันทา
ที่สำเร็จการศึกษาประจำปีการศึกษา ๒๕๐๖
ป่ายวันที่ ๑๓ มีนาคม ๒๕๐๗

ไมโครคอนโทลเลอร์ และการเขียนโปรแกรมควบคุม Arduino ด้วย Tinkercad

- ทราบและเข้าใจทฤษฎีระบบ (System Theory) และระบบสมองกลฝังตัว (Embedded System)
- ทราบและเข้าใจเกี่ยวกับไมโครคอนโทรเลอร์ (Microcontroller Unit: MCU)
- ทราบและเข้าใจการโปรแกรมเพื่อควบคุมการทำงานของไมโครคอนโทรเลอร์ (Control Programming)
- ทราบและเข้าใจเกี่ยวกับการแสดงผล (Output) ผ่านอุปกรณ์ เช่น หลอด LED แผง LED Matrix
- ทราบและเข้าใจเกี่ยวกับการนำเข้าข้อมูล (Input) ผ่านระบบเซนเซอร์ (Sensors)

ทฤษฎีระบบ
Systems Theory

Systems Theory



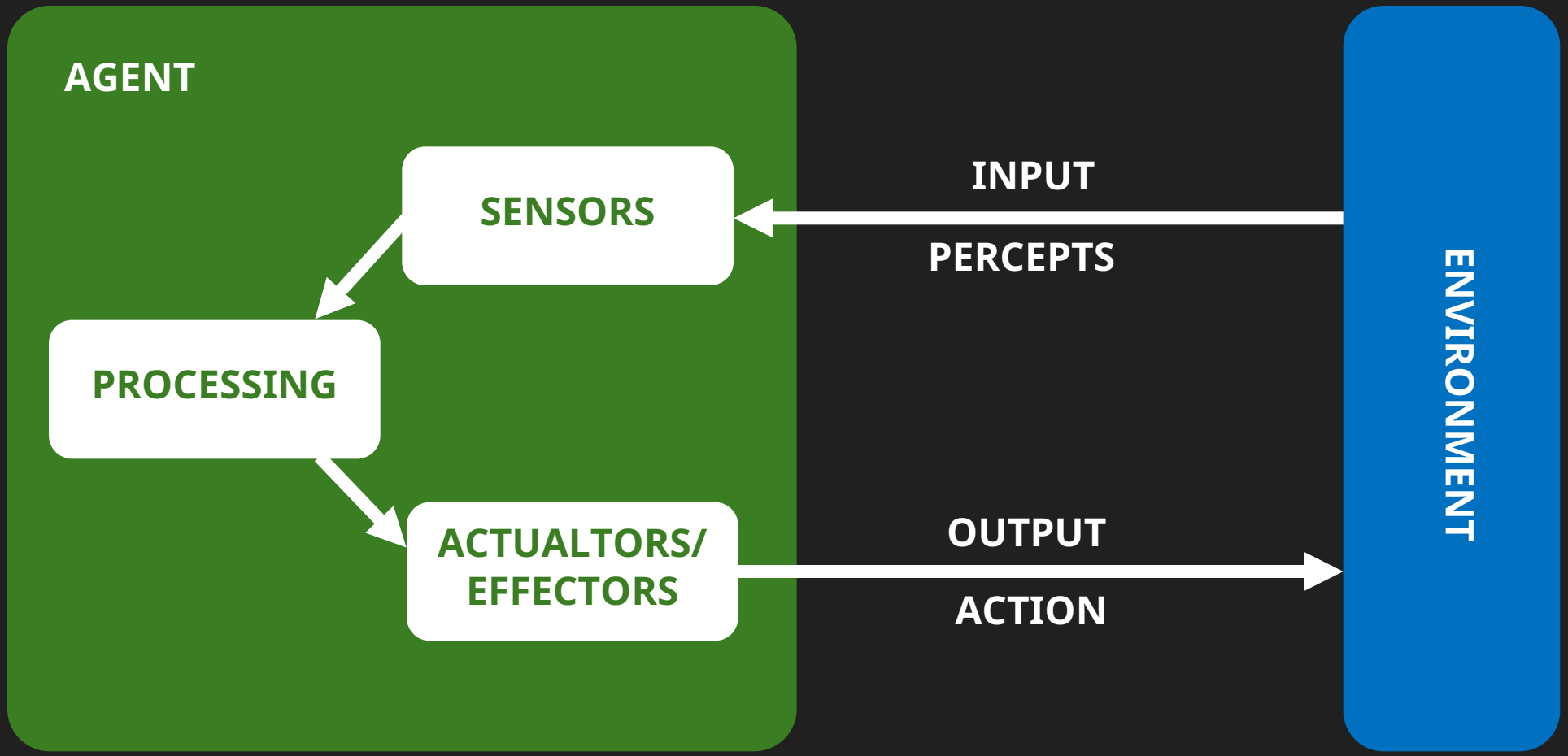
A system and its environment (Sadowski 1999, p. 19)

Sadowski, P (1999), *Systems Theory as an Approach to the Study of Literature: Origins and Functions of Literature*, E. Mellen Press, Lewiston, N.Y.

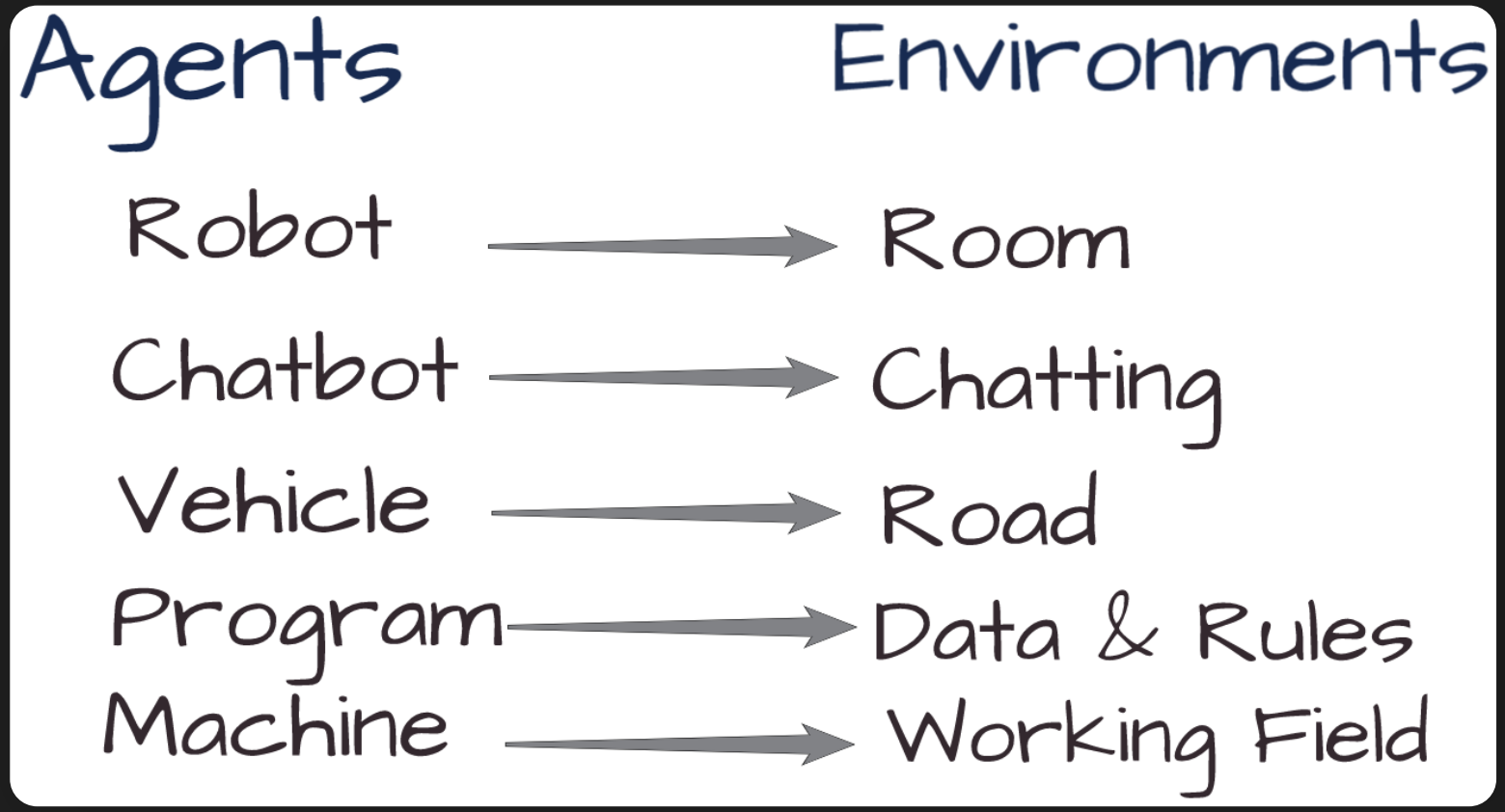
ทฤษฎีระบบ
Systems Theory

ตัวแทน
Agent

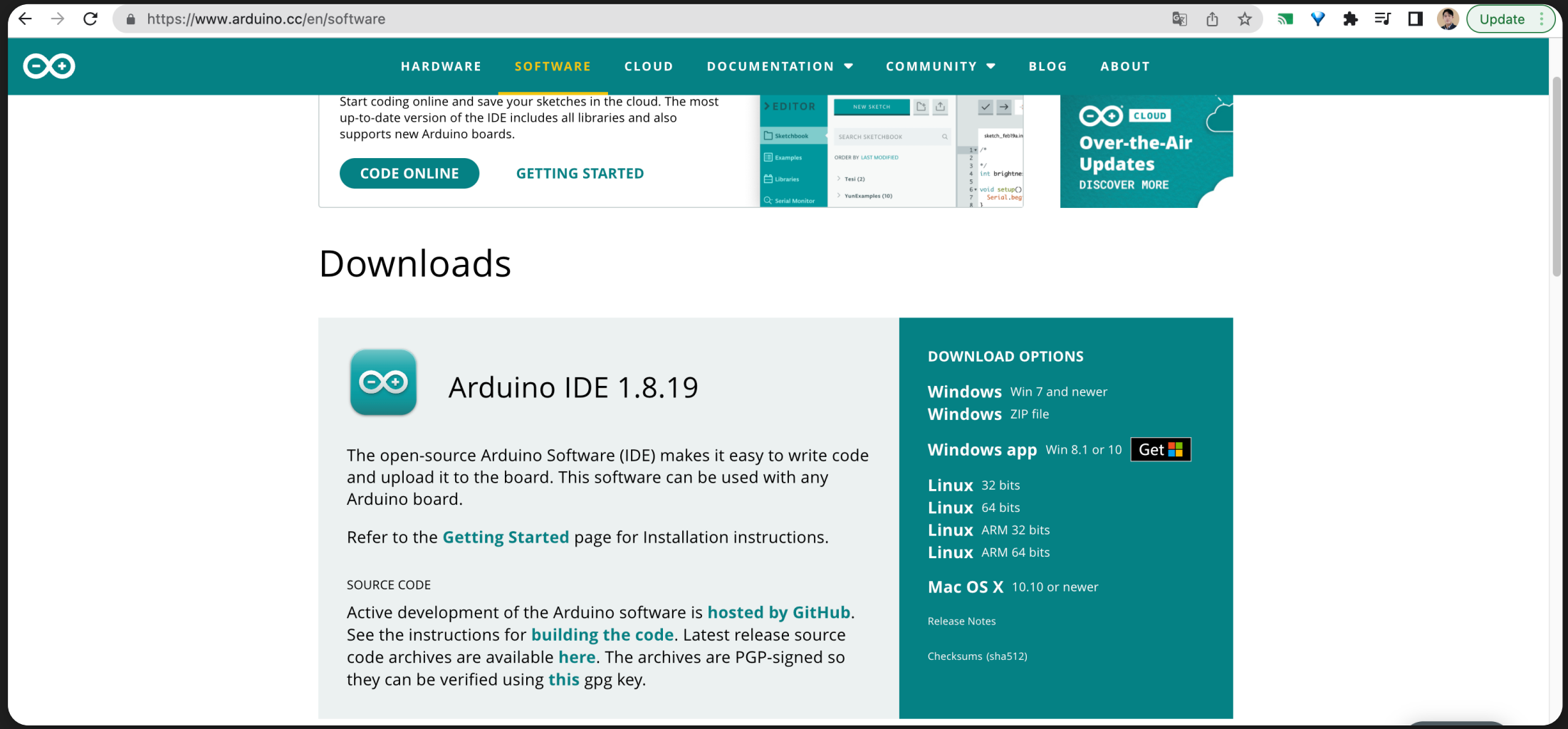
Simulation คลิกที่นี่



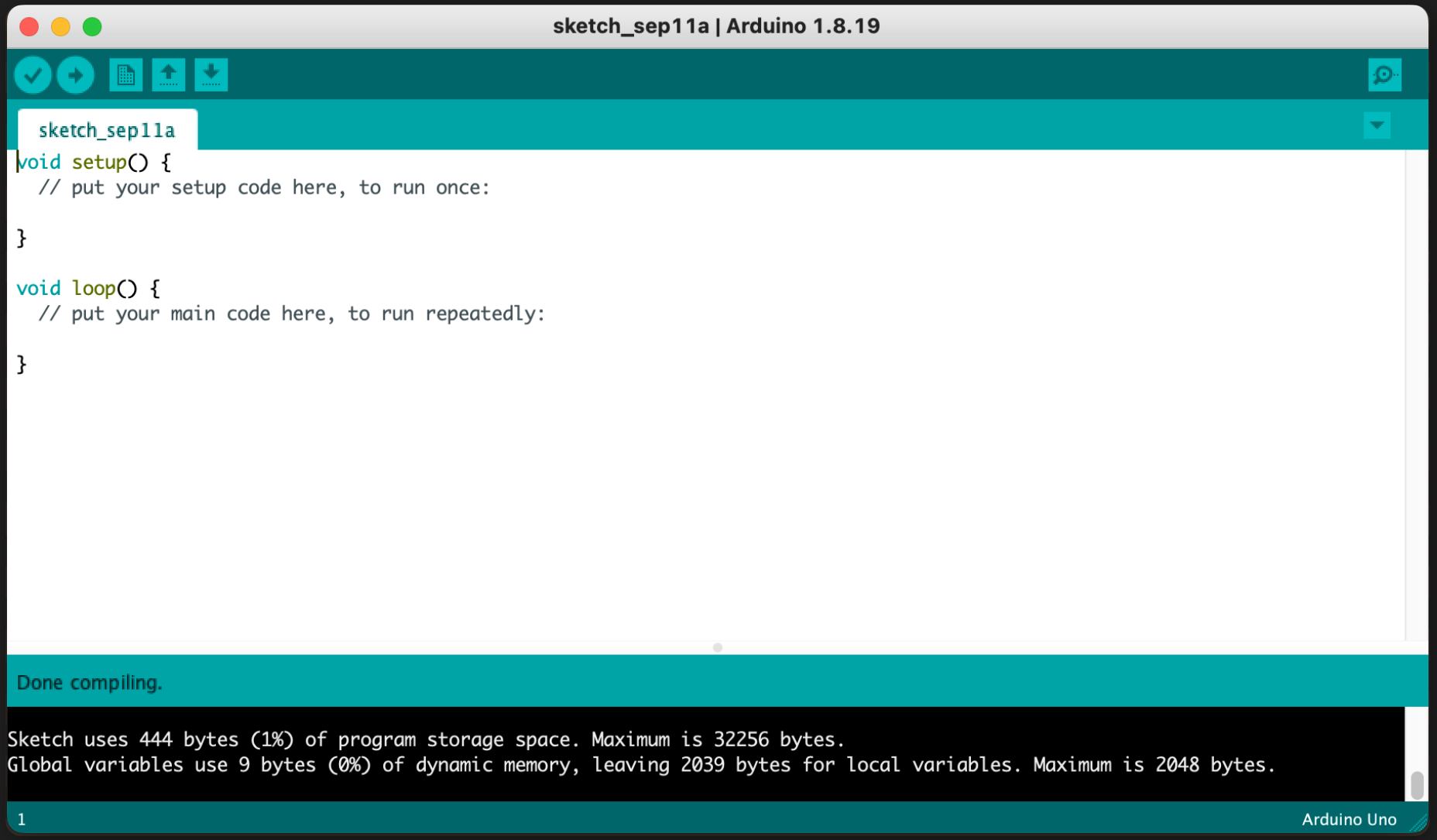
ตัวแทน กับ สภาพแวดล้อม
Agent and Environments



ติดตั้ง Arduino IDE

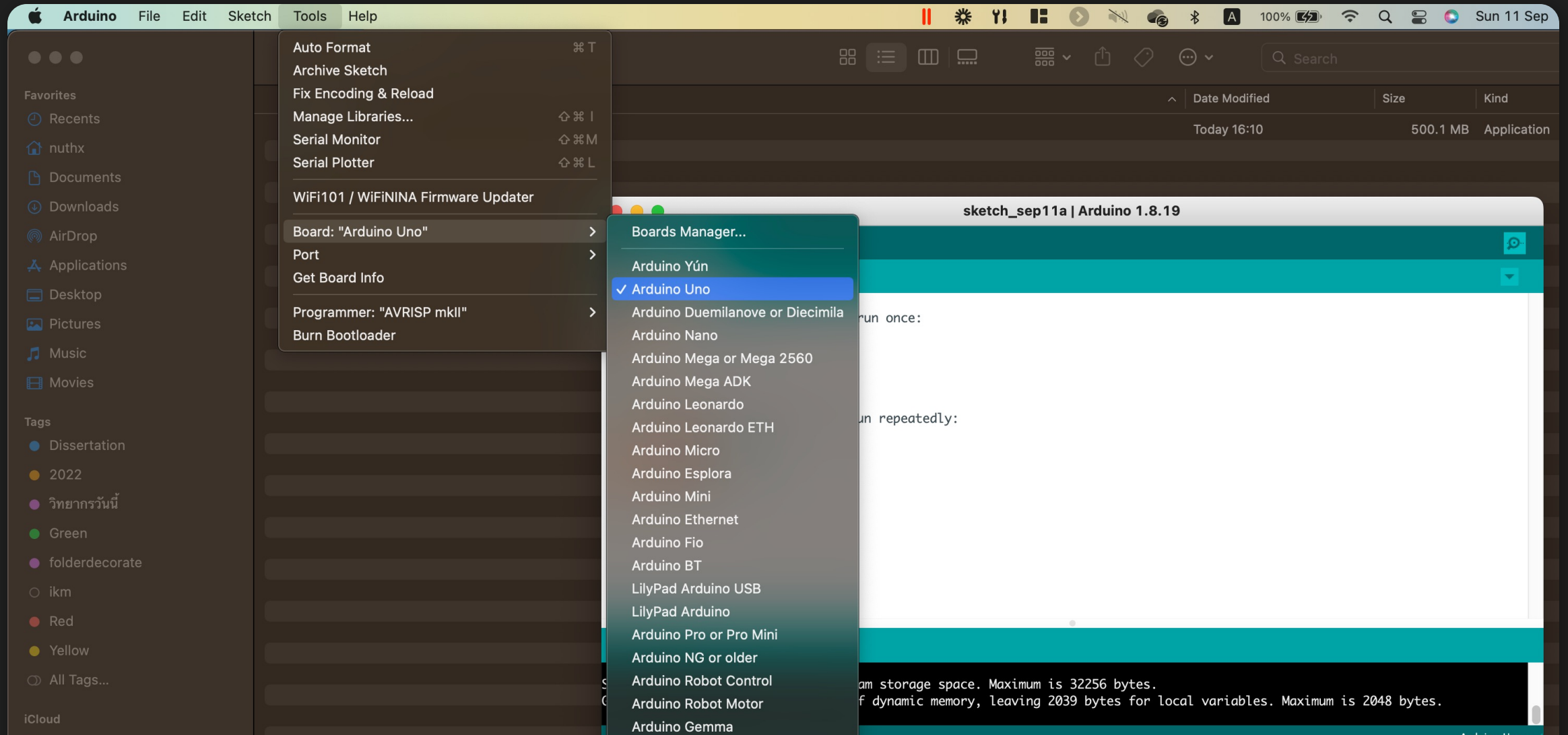


ติดตั้ง Arduino IDE

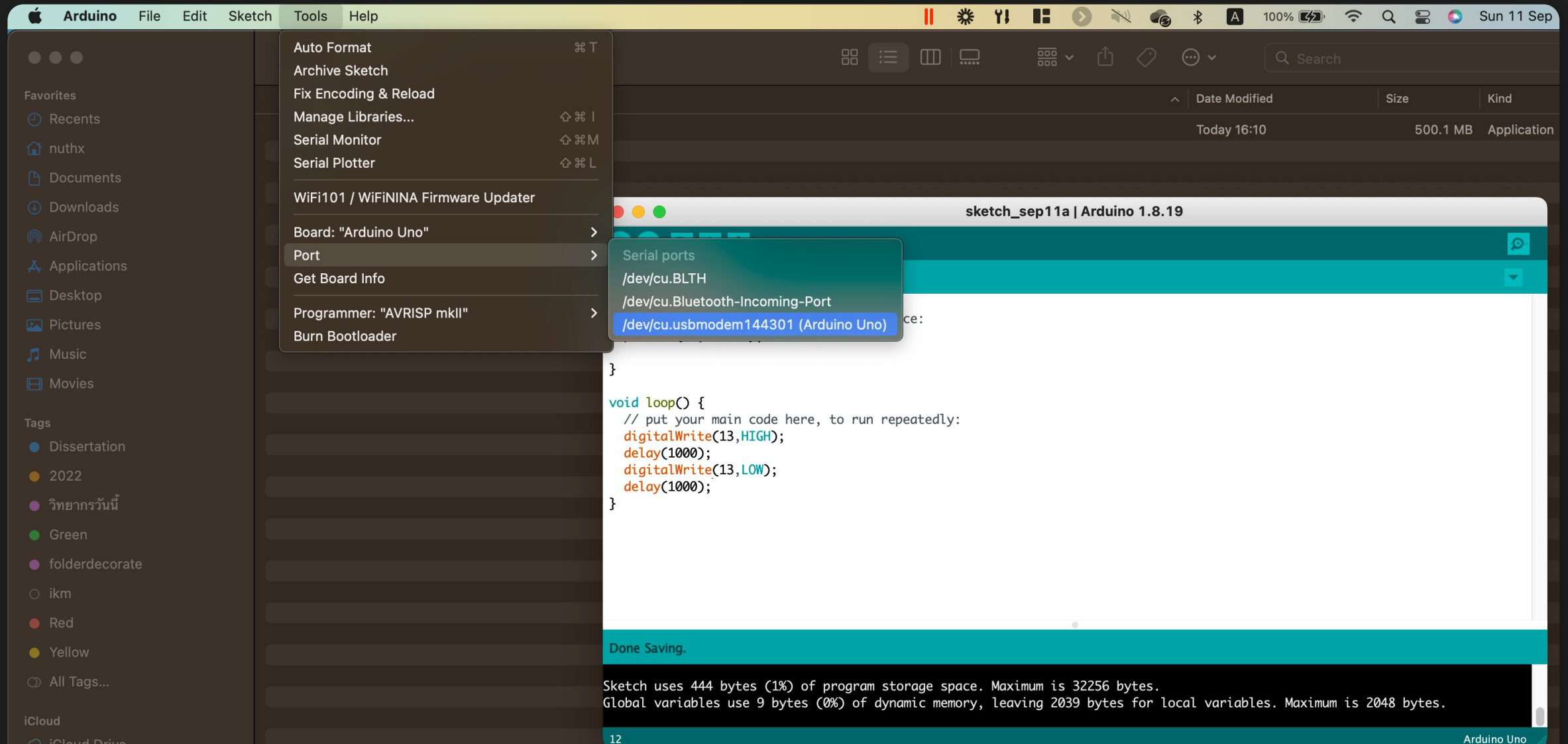


เชื่อมต่อ Arduino กับสาย USB

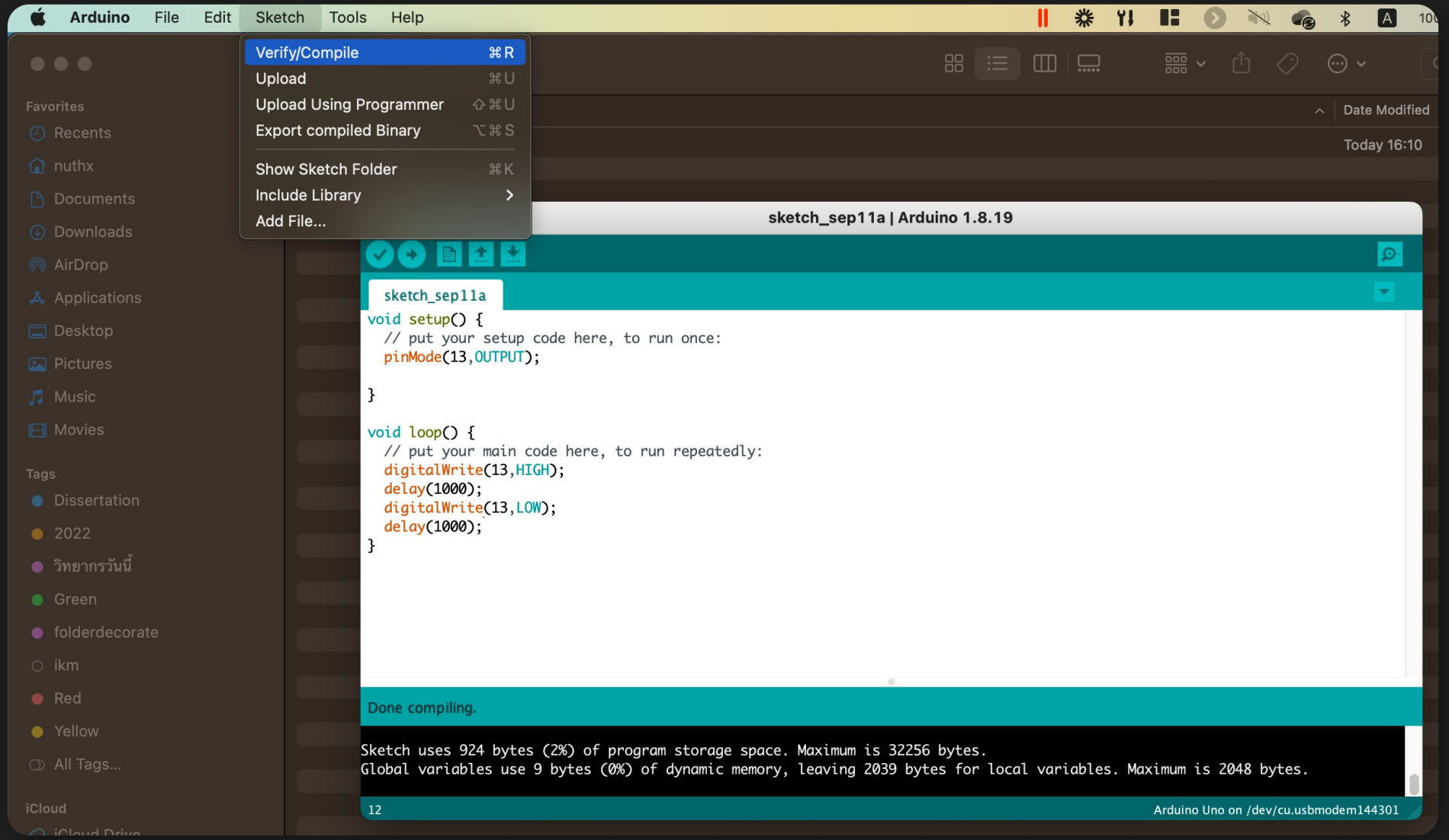
เชื่อมต่อ Arduino กับสาย USB



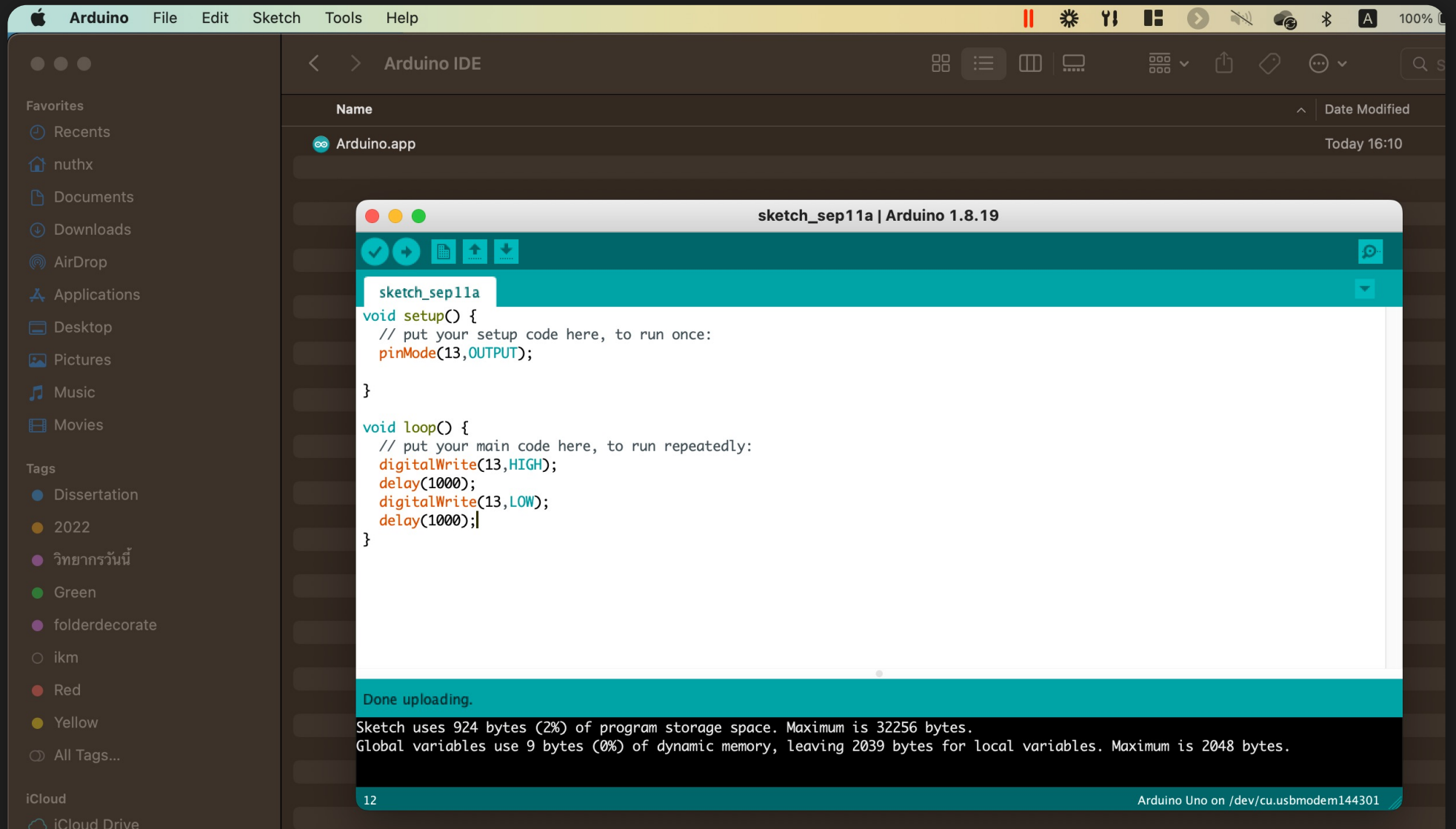
เชื่อมต่อ Arduino กับสาย USB



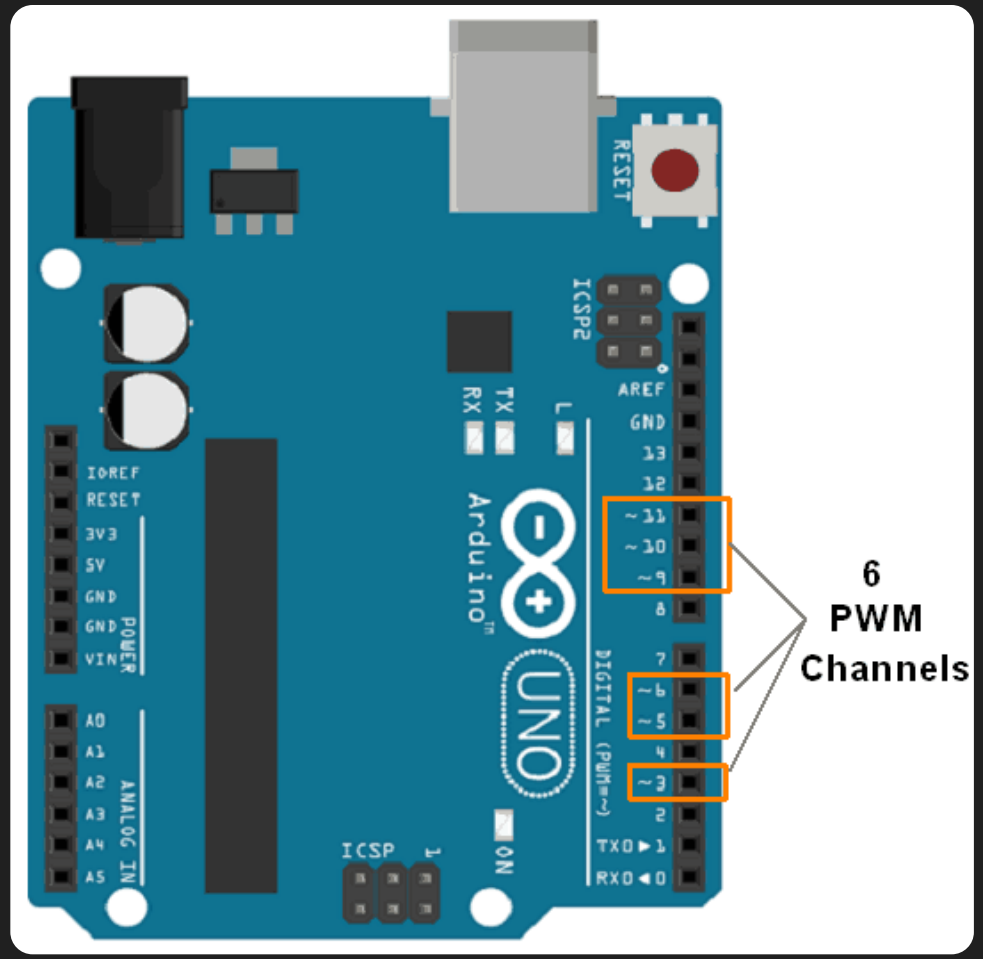
เชื่อมต่อ Arduino กับสาย USB

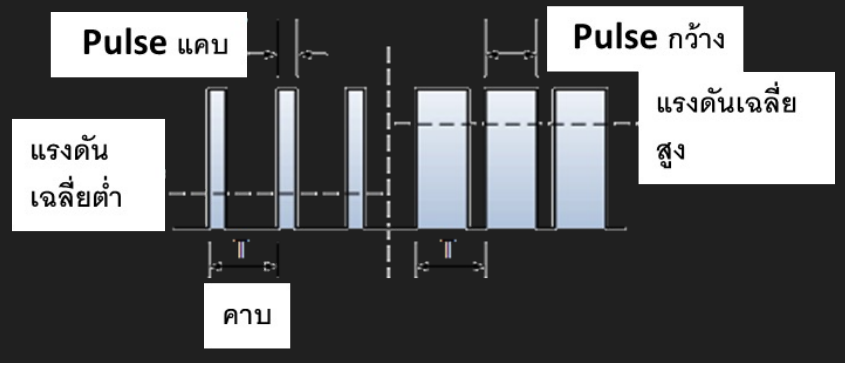


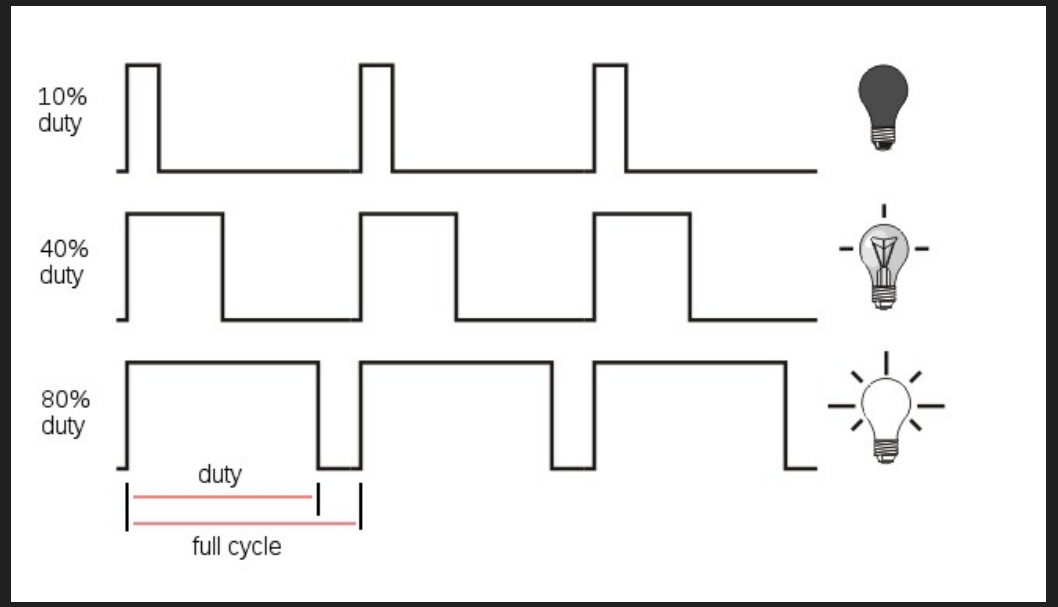
เชื่อมต่อ Arduino กับสาย USB



Pulse Width Modulation: PWM







Pulse Width Modulation: PWM

- PWM หรือ Pulse Width Modulation คือ เทคนิคการควบคุมแรงดันไฟฟ้าเฉลี่ยที่ส่งไปยังอุปกรณ์ไฟฟ้า โดยการปรับความกว้างของพัลส์สัญญาณดิจิทัล

หลักการทำงาน

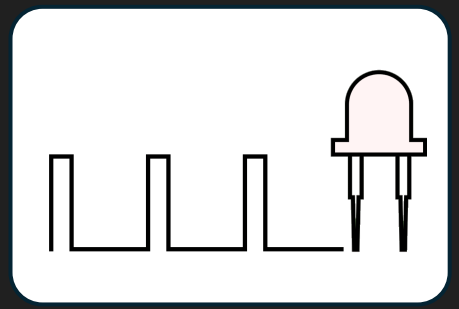
- สร้างสัญญาณพัลส์ที่มีความถี่คงที่
- ปรับเปลี่ยนช่วงเวลาที่สัญญาณเป็น ON (ความกว้างของพัลส์)
- ค่าเฉลี่ยของแรงดันไฟฟ้าจะขึ้นอยู่กับอัตราส่วนระหว่างช่วง ON และ OFF

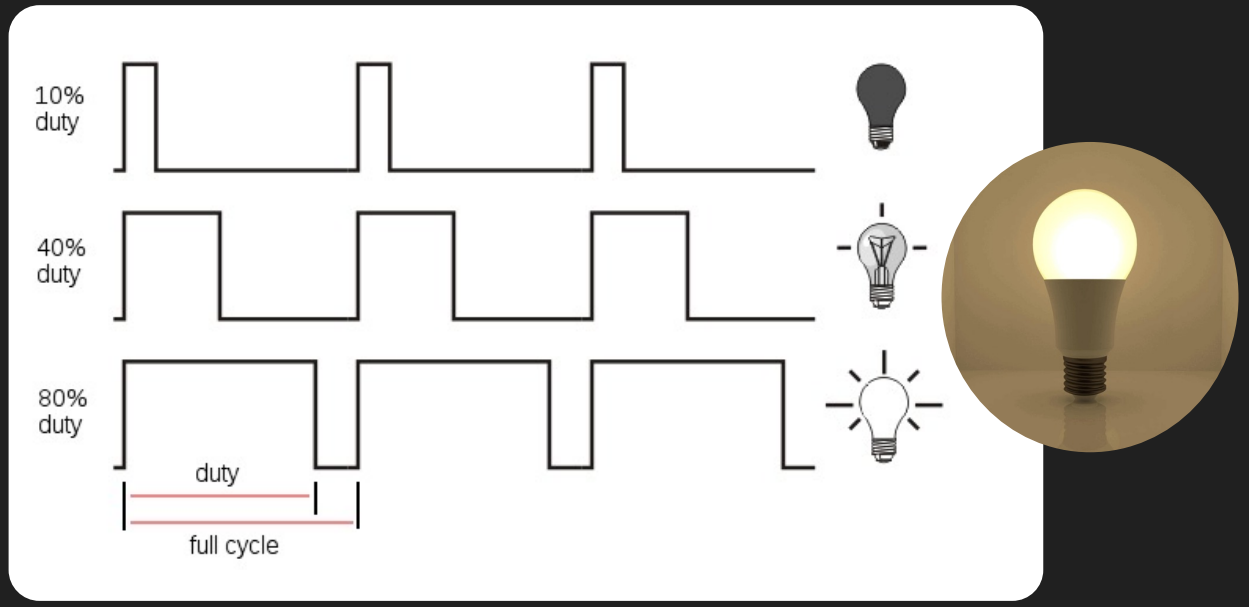
Pulse Width Modulation: PWM

- PWM หรือ Pulse Width Modulation คือ เทคนิคการควบคุมแรงดันไฟฟ้าเฉลี่ยที่ส่งไปยังอุปกรณ์ไฟฟ้า โดยการปรับความกว้างของพัลส์สัญญาณดิจิทัล

ประโยชน์

- ควบคุมความเร็วมอเตอร์
- ปรับความสว่างของ LED
- ควบคุมอุณหภูมิในระบบทำความร้อน
- ใช้ในวงจรแปลงไฟ DC เป็น AC





Pulse Width Modulation: PWM

Simulation คลิกที่นี่

- PWM หรือ Pulse Width Modulation คือ เทคนิคการควบคุมแรงดันไฟฟ้าเฉลี่ยที่ส่งไปยังอุปกรณ์ไฟฟ้า โดยการปรับความกว้างของพัลส์สัญญาณดิจิทัล

ภาพนี้แสดงตัวอย่างของสัญญาณ Pulse Width Modulation (PWM) ที่ระดับ Duty Cycle แตกต่างกัน 3 ระดับ พร้อมแสดงผลลัพธ์ที่มีต่อความสว่างของหลอดไฟ LED

10% Duty Cycle

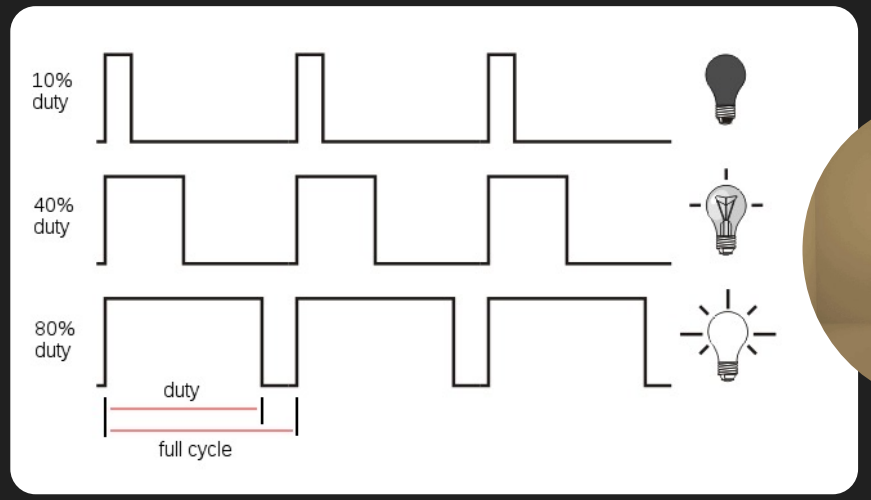
สัญญาณ ON เพียงช่วงสั้นๆ และ OFF เป็นส่วนใหญ่ ส่งผลให้หลอดไฟ LED สว่างน้อยที่สุด (แทบจะดับ)

40% Duty Cycle

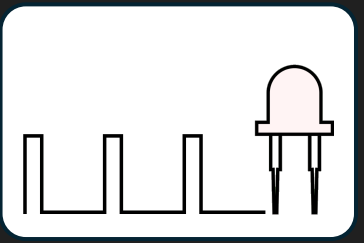
สัญญาณ ON นานขึ้น แต่ยังคง OFF มากกว่า ON หลอดไฟ LED สว่างขึ้นในระดับปานกลาง

80% Duty Cycle

สัญญาณ ON เป็นส่วนใหญ่ และ OFF เพียงช่วงสั้นๆ หลอดไฟ LED สว่างมากที่สุดในตัวอย่างนี้



duty คือ ช่วงเวลาที่สัญญาณเป็น ON
full cycle คือ หนึ่งรอบสัญญาณ ประกอบด้วยช่วง ON และ OFF



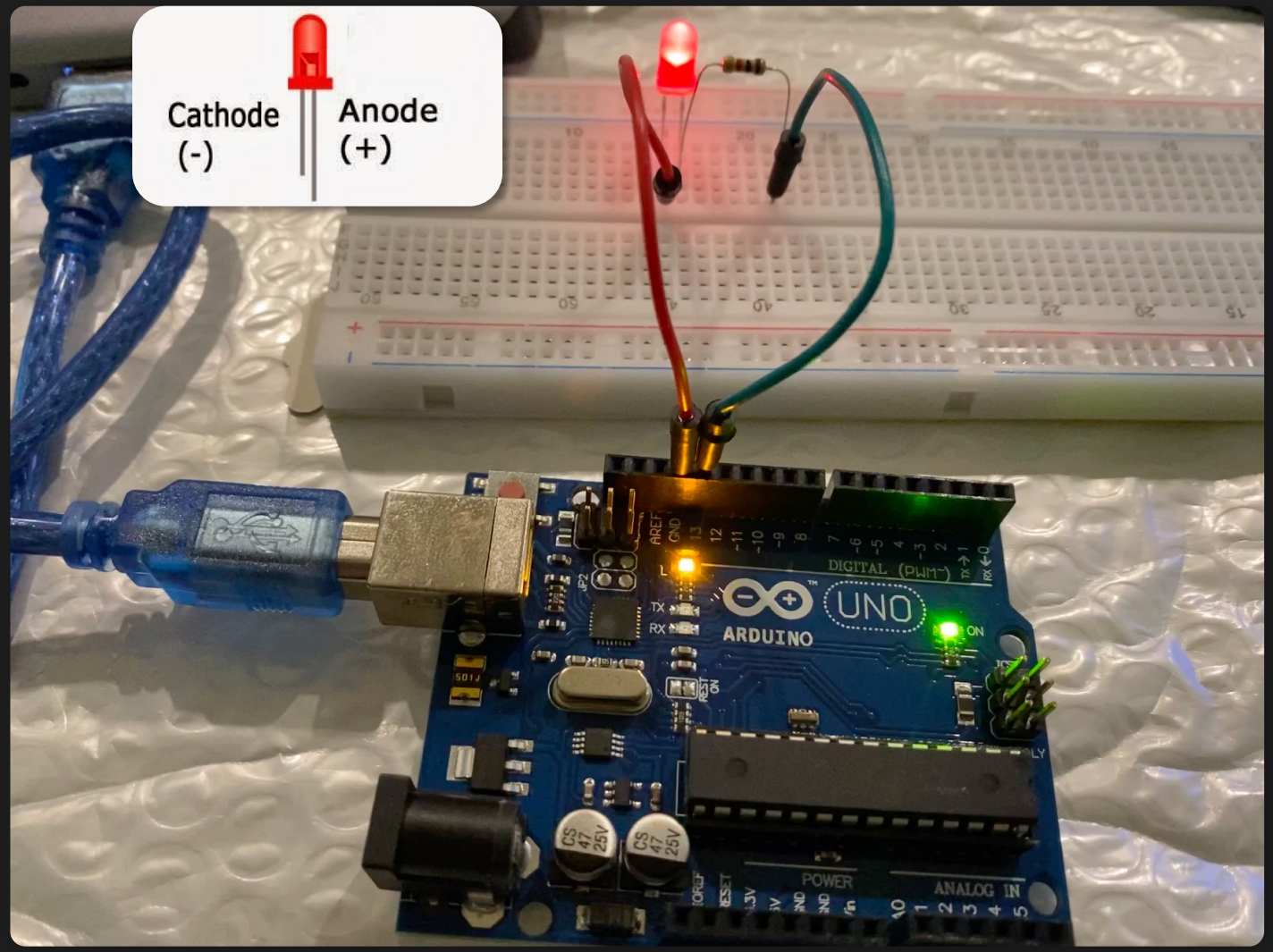
Pulse Width Modulation: PWM

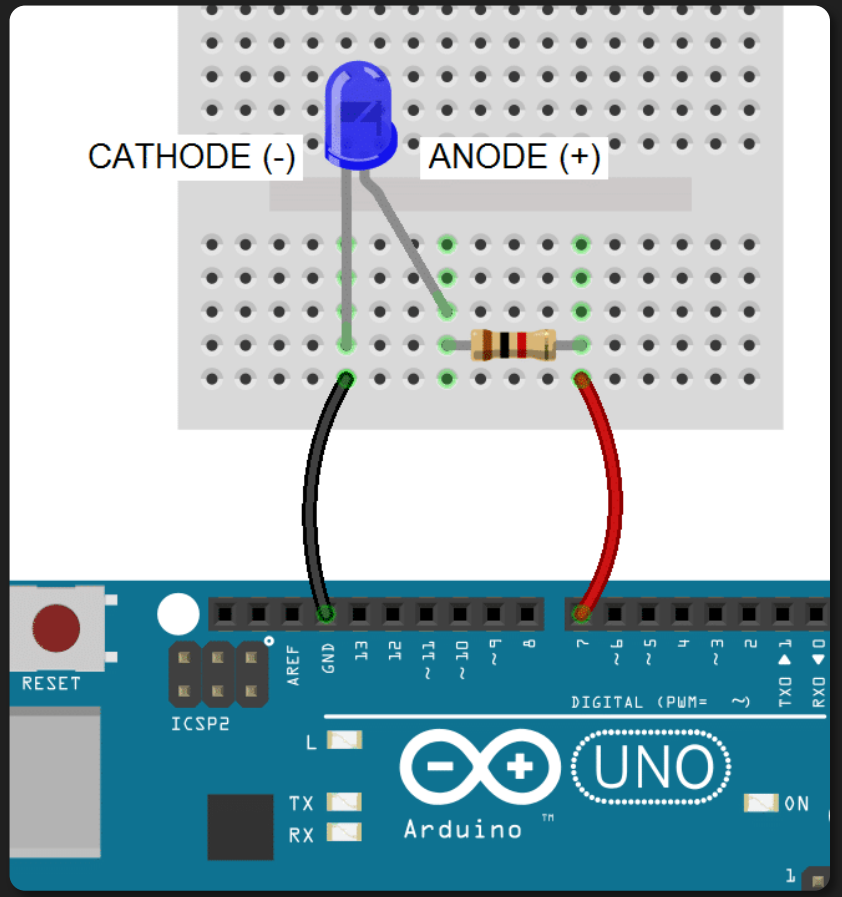
- PWM หรือ Pulse Width Modulation คือ เทคนิคการควบคุมแรงดันไฟฟ้าเฉลี่ยที่ส่งไปยังอุปกรณ์ไฟฟ้า โดยการปรับความกว้างของพัลส์สัญญาณดิจิทัล

ข้อดี
- ประสิทธิภาพสูง เนื่องจากการสูญเสียพลังงานต่ำ
- ควบคุมได้แม่นยำ
- ใช้งานง่ายกับระบบดิจิทัลและไมโครคอนโทรลเลอร์

การใช้งานทั่วไป
- อุปกรณ์อิเล็กทรอนิกส์
- ระบบควบคุมอัตโนมัติ
- ยานยนต์
- อุปกรณ์ไฟฟ้าในบ้าน

การโปรแกรมเพื่อควบคุมหลอด LED

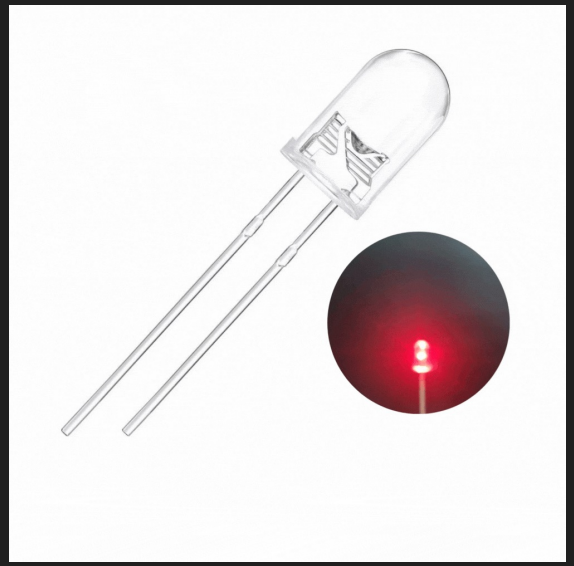




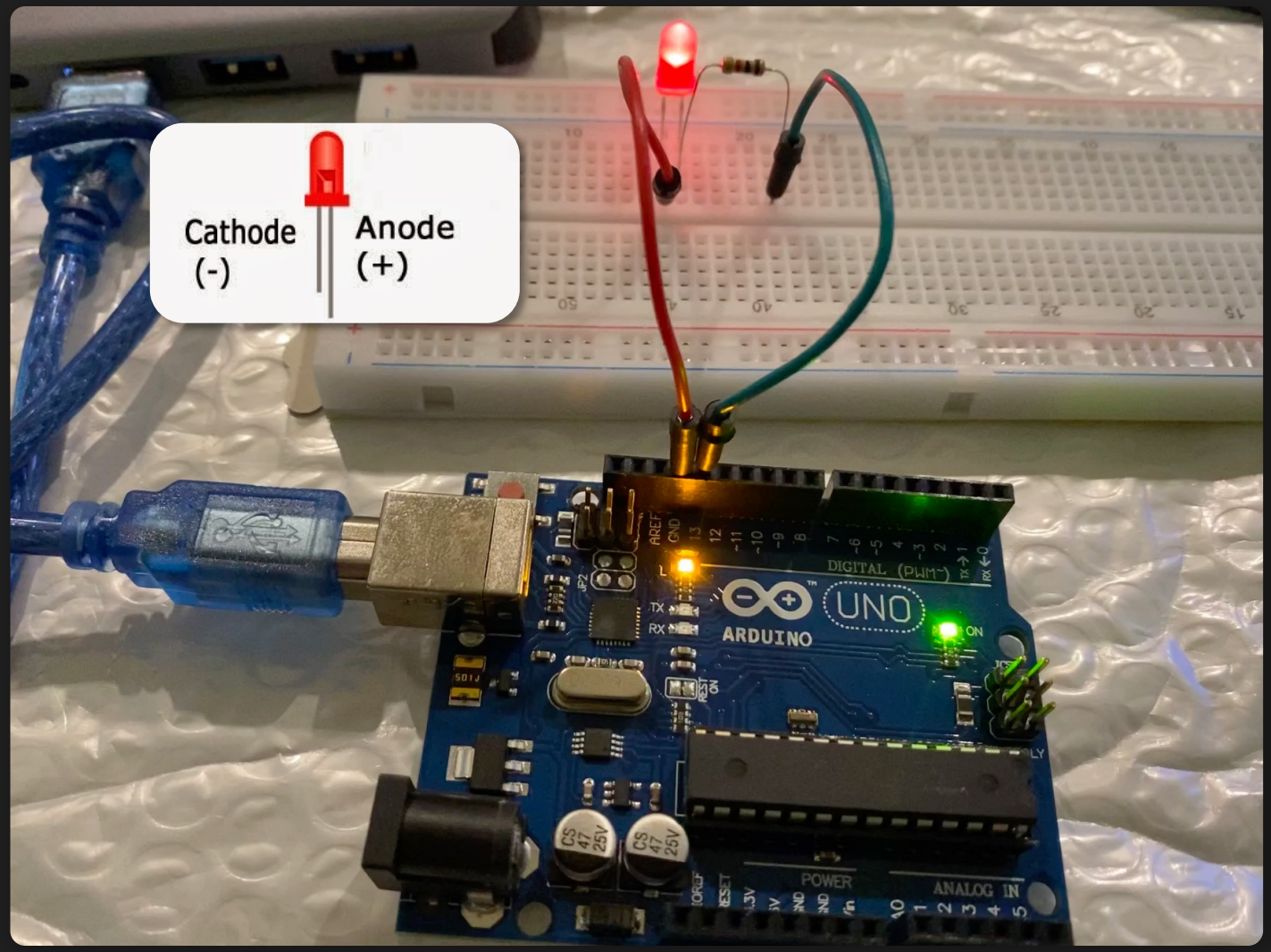
https://www.circuitbasics.com/arduino-7-segment-display-tutorial/

LED หรือ Light Emitting Diode

- คือ อุปกรณ์สารกึ่งตัวนำที่สามารถเปล่งแสงได้เมื่อมีกระแสไฟฟ้าไหลผ่าน ประหยัดพลังงาน มีอายุการใช้งานยาวนาน ไม่เปราะแตกง่าย ให้แสงสว่างทันทีเมื่อเปิดใช้งาน
- **หลักการทำงาน** เมื่อกระแสไฟฟ้าไหลผ่านสารกึ่งตัวนำ จะเกิดการปลดปล่อยพลังงานในรูปของแสง และ สีของแสงขึ้นอยู่กับชนิดของสารกึ่งตัวนำที่ใช้
- **คุณสมบัติสำคัญ**
  - ประสิทธิภาพสูงในการเปลี่ยนพลังงานไฟฟ้าเป็นแสง
  - มีอายุการใช้งานยาวนาน (มากกว่า 50,000 ชั่วโมงในบางรุ่น)
  - ขนาดเล็ก น้ำหนักเบา
  - ไม่มีสารปรอทหรือสารพิษอื่นๆ
  - สามารถควบคุมความสว่างได้ง่ายด้วยเทคนิค PWM



หลอด LED



Arduino1_Nutthapat

Verify

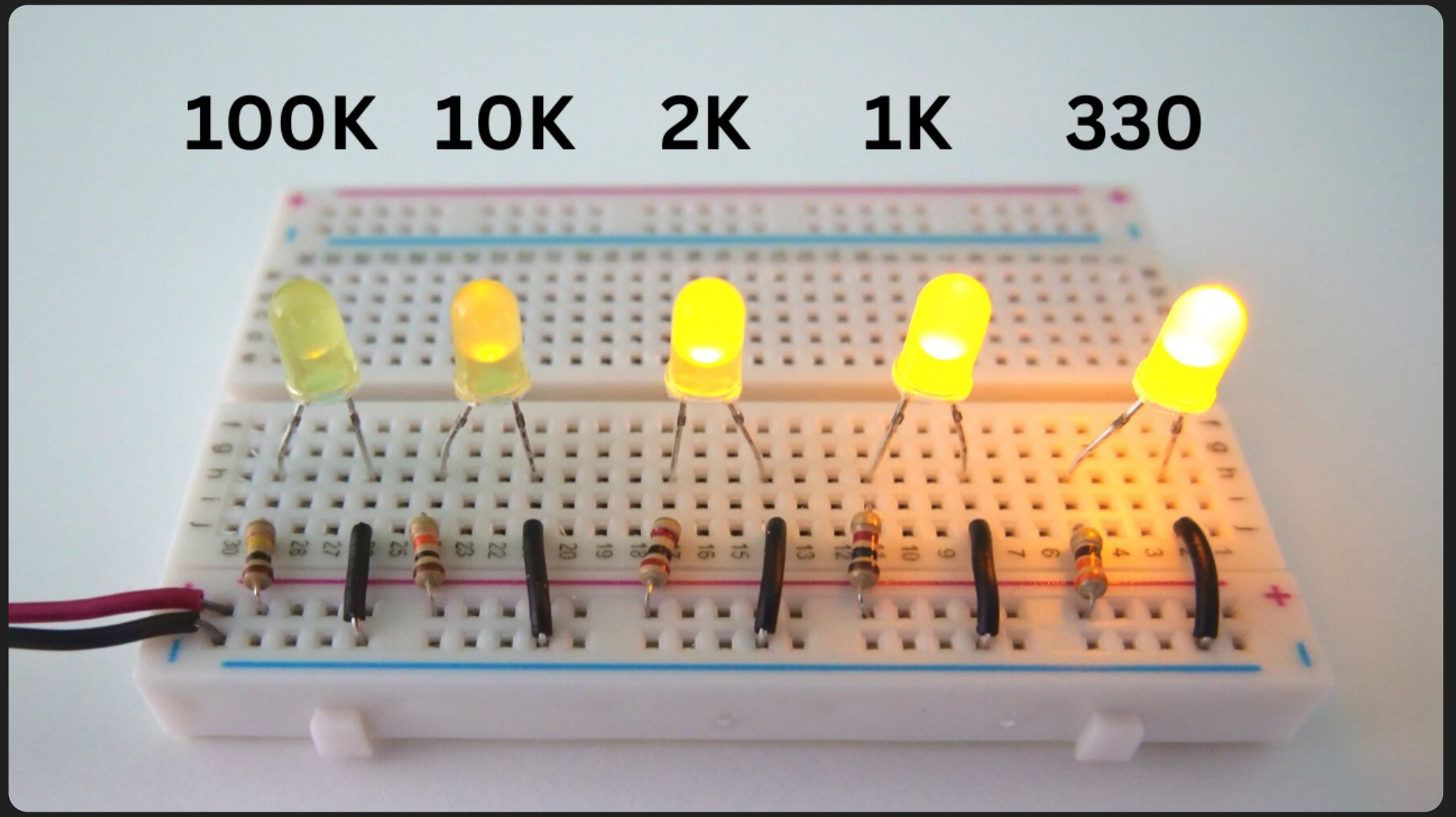
Arduino1_Nutthapat §

```
void setup() {
  // put your setup code here, to run once:
  pinMode(13,OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(13,HIGH);
  delay(1000);
  digitalWrite(13,LOW);
  delay(1000);
}
```

Done uploading.

```
Sketch uses 924 bytes (2%) of program storage space. Maximum is 322
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 b
```

ตัวต้านทาน (Resistor)



การคำนวณค่าตัวต้านทาน

R = (Vs - Vf) / (If * N)

โดยที่

R = ค่าความต้านทานที่ต้องการ (โอห์ม)

Vs = แรงดันแหล่งจ่าย (โวลต์)

Vf = แรงดันตกคร่อม LED (โวลต์)

If = กระแสที่ต้องการให้ไหลผ่าน LED แต่ละหลอด (แอมแปร์)

N = จำนวน LED (ในที่นี้คือ 10)

ตัวอย่างการคำนวณ

Vs = 5V (Arduino)
Vf = 2V
If = 20mA
N = 10 หลอด

R = (5 - 2) / (0.02 * 10) = 15 โอห์ม

ในทางปฏิบัติ ควรเลือกค่าตัวต้านทานที่ใกล้เคียงและมากกว่าค่าที่คำนวณได้เล็กน้อย เช่น 18 หรือ 20 โอห์ม

การจ่ายไฟฟ้าของ Arduino

Arduino จ่ายไฟออกมาได้หลายระดับ ขึ้นอยู่รุ่นของ **Arduino**

1.พินจ่ายไฟหลัก

- 5V จ่ายไฟกระแสตรง 5 โวลต์
- 3.3V จ่ายไฟกระแสตรง 3.3 โวลต์ (มีในบางรุ่น)

2.พินดิจิตอล (Digital pins)

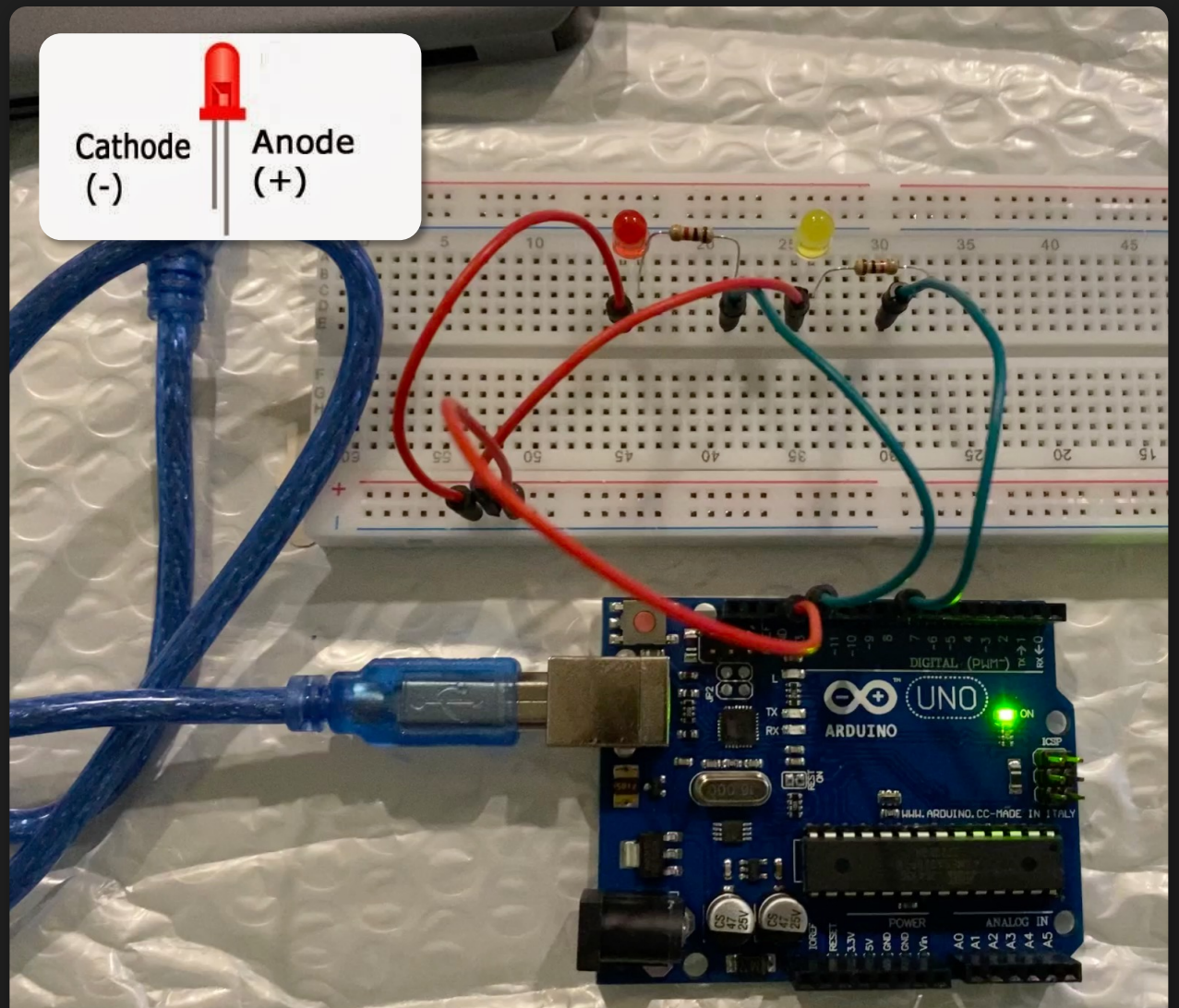
- เมื่อตั้งค่าเป็น OUTPUT และกำหนดให้เป็น HIGH จะจ่ายไฟ 5 โวลต์
- บางรุ่น เช่น Arduino Due จ่ายไฟที่ 3.3 โวลต์

3.พิน VIN

- ในกรณีที่ Arduino ได้รับไฟเลี้ยงผ่านแจ็คไฟ พิน VIN จะจ่ายไฟเท่ากับแรงดันที่ป้อนเข้ามา (ก่อนการลดแรงดัน)

สำหรับการต่อ LED โดยตรง มักจะใช้พินดิจิตอลที่จ่ายไฟ 5 โวลต์

หลอด LED



Arduino1_Nutthapat

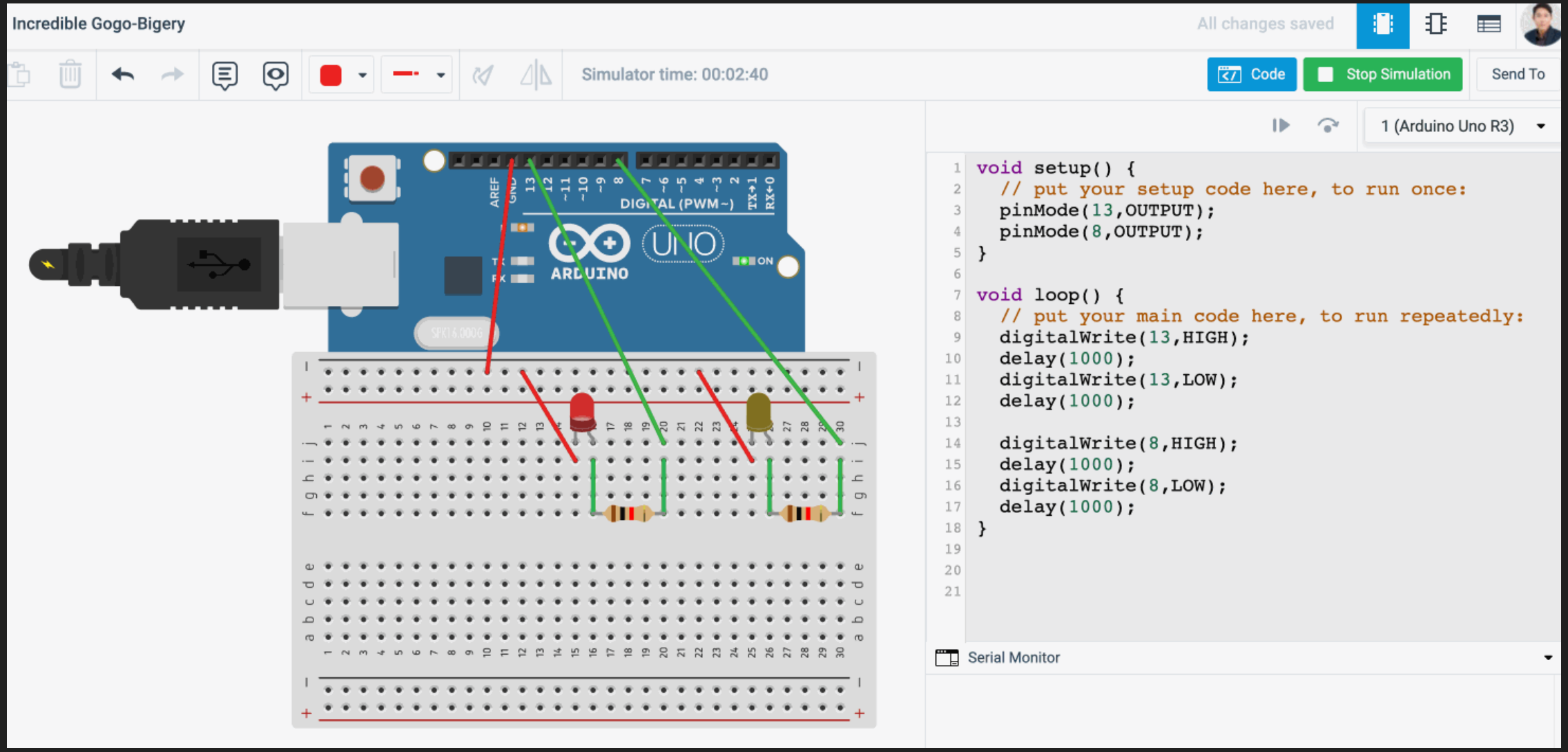
Arduino1_Nutthapat

```
void setup() {
  // put your setup code here, to run once:
  pinMode(13,OUTPUT);
  pinMode(8,OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(13,HIGH);
  delay(1000);
  digitalWrite(13,LOW);
  delay(1000);

  digitalWrite(8,HIGH);
  delay(1000);
  digitalWrite(8,LOW);
  delay(1000);
}
```

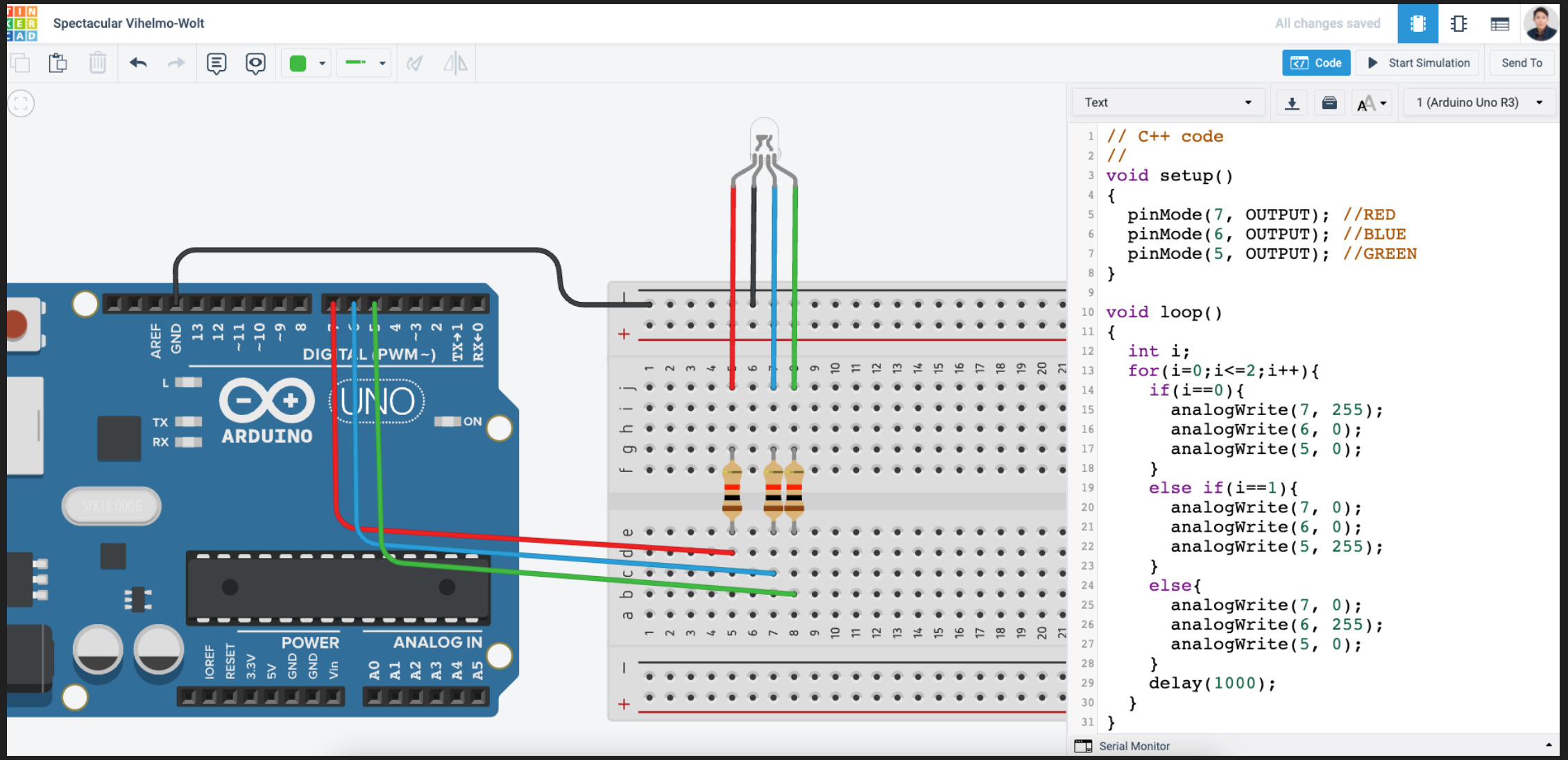
Done uploading.

Sketch uses 980 bytes (3%) of program storage space. Maximum is 322
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 b

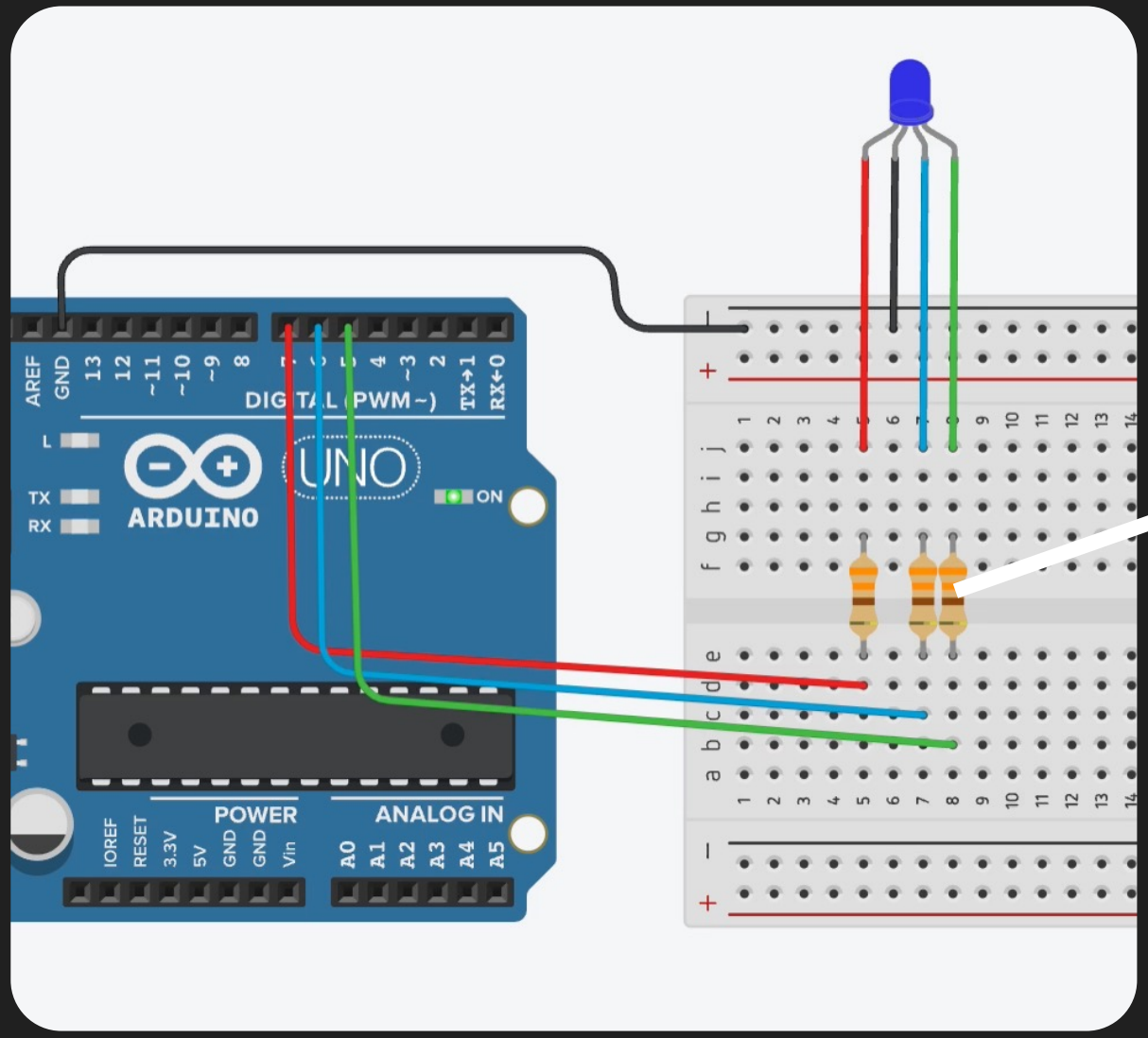
การโปรแกรมเพื่อควบคุมหลอด LED



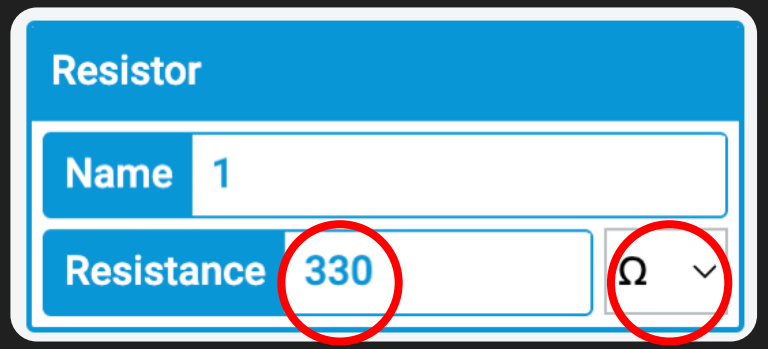
การโปรแกรมเพื่อควบคุมหลอด LED



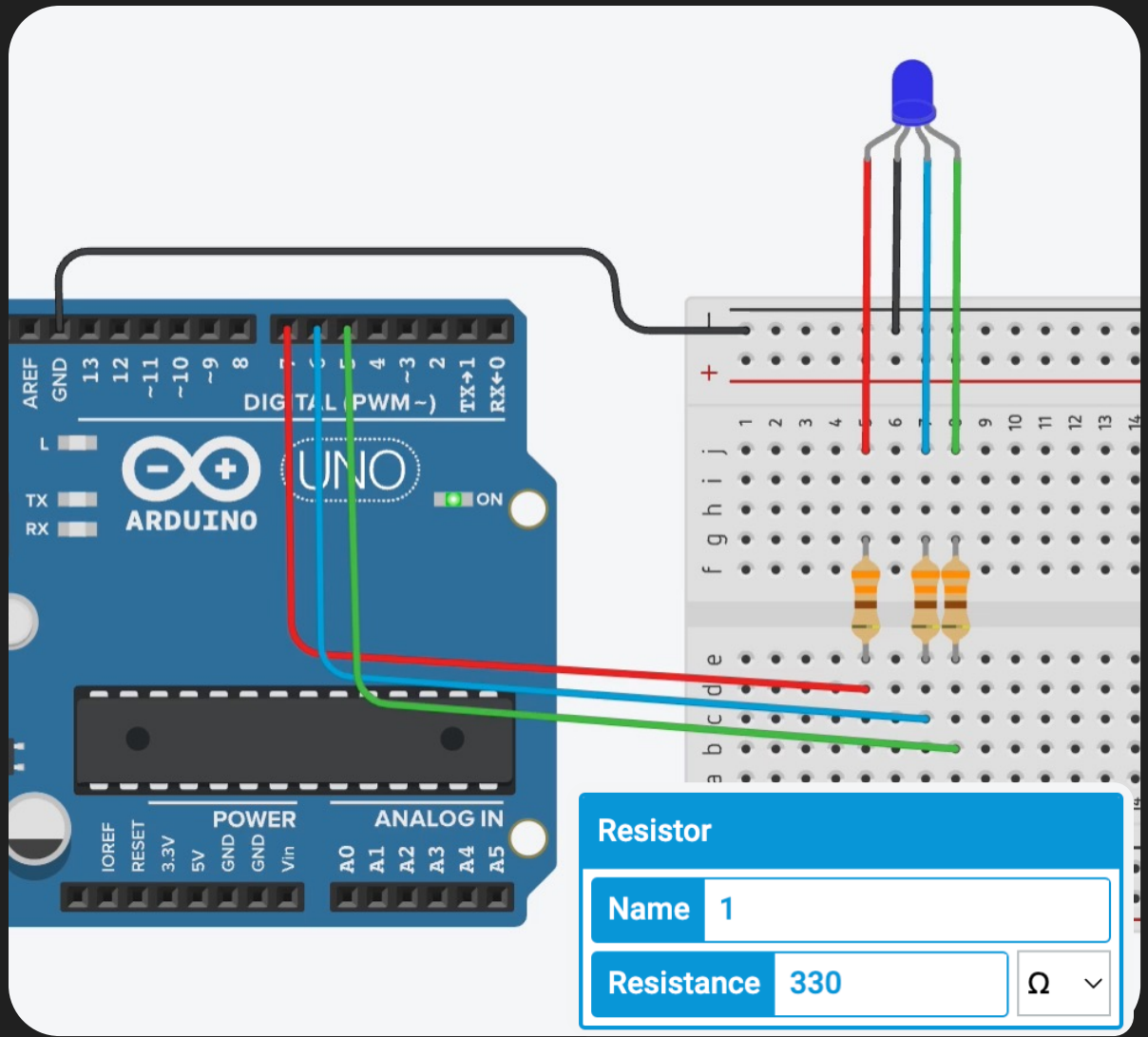
การโปรแกรมเพื่อควบคุมหลอด LED

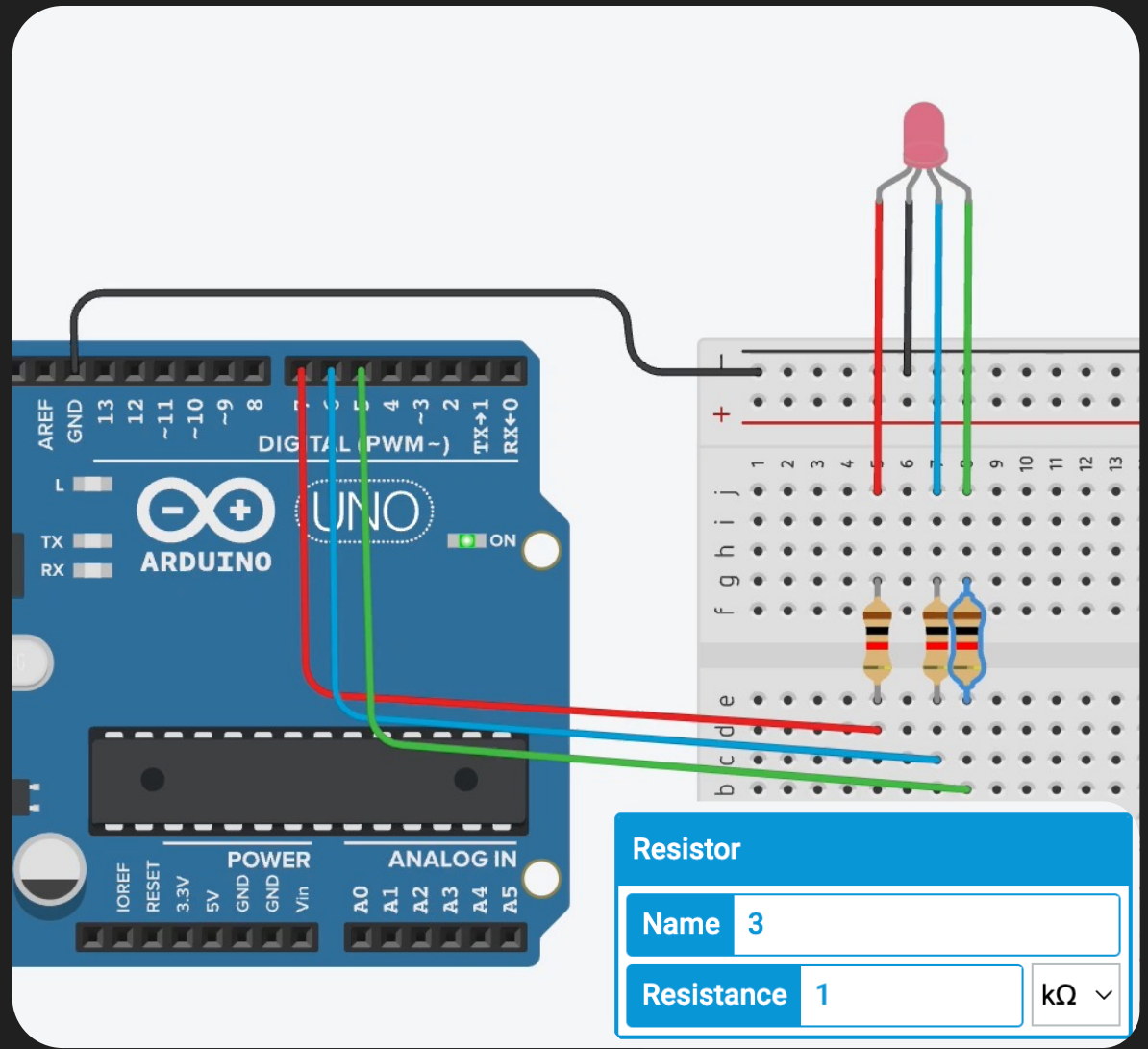


ตัวต้านทาน

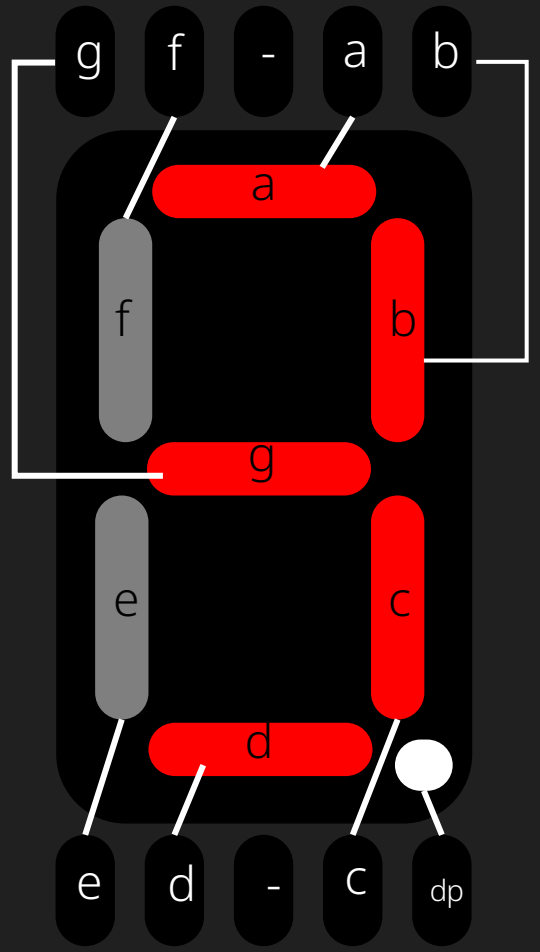


การโปรแกรมเพื่อควบคุมหลอด LED

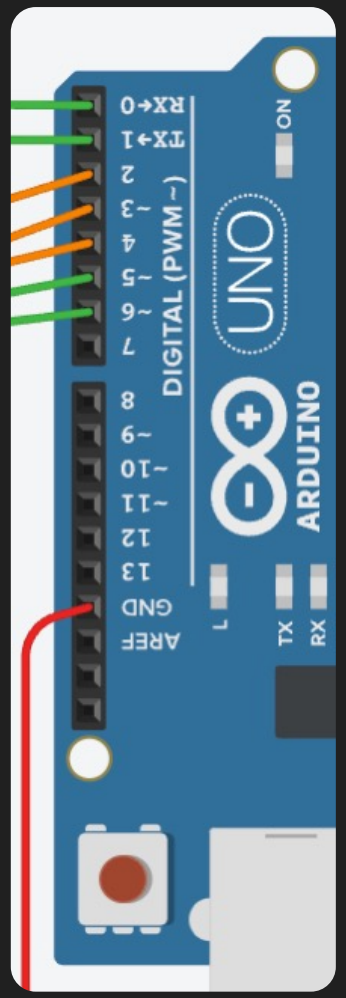




Seven Segment กับ Arduino UNO



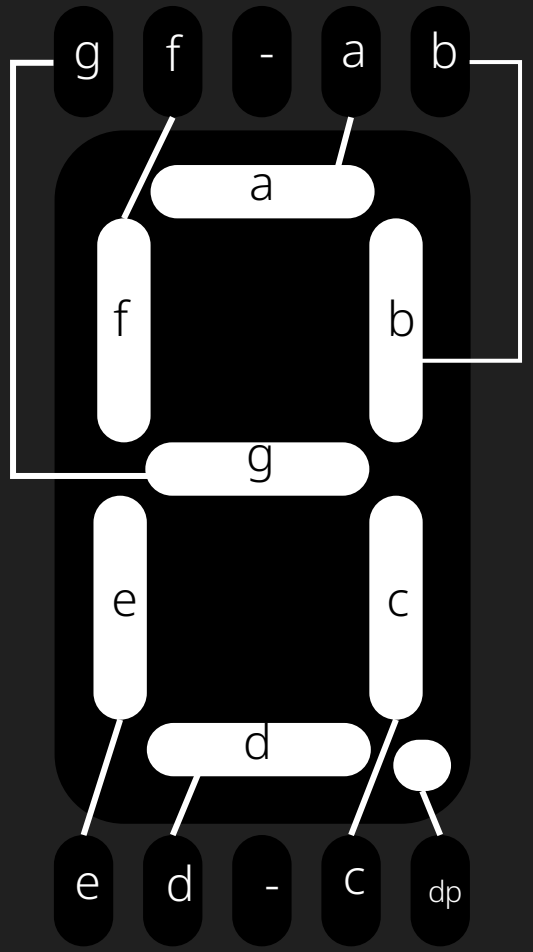
กำหนดให้	
Segment Pin	Arduino Pin
a	0
b	1
c	2
d	3
e	4
f	5
g	6



SevenSegment_Three

```
void setup() {
  // put your setup code here, to run once:
  for(int x=0;x<=6;x++){
    pinMode(x,OUTPUT);
  }
}

void show_three() {
  digitalWrite(0,HIGH);
  digitalWrite(1,HIGH);
  digitalWrite(2,HIGH);
  digitalWrite(3,HIGH);
  digitalWrite(4,LOW);
  digitalWrite(5,LOW);
  digitalWrite(6,HIGH);
}

void loop() {
  // put your main code here, to run repeatedly:
  show_three();
}
```

Seven Segment กับ Arduino UNO

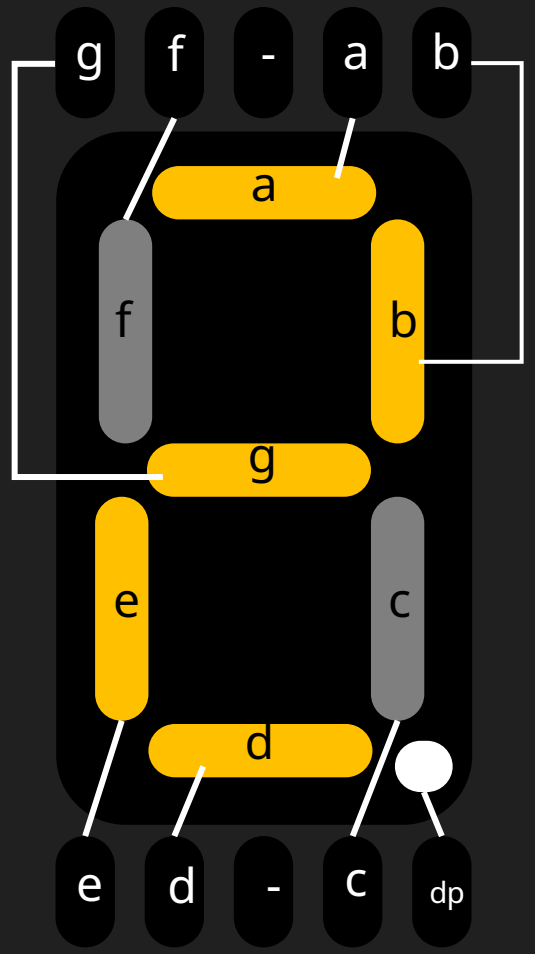


SevenSegment

SevenSegment §

```
void setup() {
  // put your setup code here, to run once:
  for(int x=0;x<=6;x++){
    pinMode(x,OUTPUT);
  }
}

void show_two() {
  digitalWrite(0,HIGH);
  digitalWrite(1,HIGH);
  digitalWrite(2,LOW);
  digitalWrite(3,HIGH);
  digitalWrite(4,HIGH);
  digitalWrite(5,LOW);
  digitalWrite(6,HIGH);
}

void led_off() {
  for(int x=0;x<=6;x++){
    digitalWrite(x,LOW);
  }
}

void loop() {
  // put your main code here, to run repeatedly:
  show_two();
  delay(1000);
  led_off();
  delay(1000);
}
```

Seven Segment กับ Arduino UNO



กำหนดให้	
Segment Pin	Arduino Pin
a	0
b	1
c	2
d	3
e	4
f	5
g	6

SevenSegment

SevenSegment §

```
void setup() {
  // put your setup code here, to run once:
  for(int x=0;x<=6;x++){
    pinMode(x,OUTPUT);
  }
}

void show_two() {
  digitalWrite(0,HIGH);
  digitalWrite(1,HIGH);
  digitalWrite(2,LOW);
  digitalWrite(3,HIGH);
  digitalWrite(4,HIGH);
  digitalWrite(5,LOW);
  digitalWrite(6,HIGH);
}

void led_off() {
  for(int x=0;x<=6;x++){
    digitalWrite(x,LOW);
  }
}

void loop() {
  // put your main code here, to run repeatedly:
  show_two();
  delay(1000);
  led_off();
  delay(1000);
}
```

Seven Segment กับ Arduino UNO

SAMPLE