

DTI2301 การเขียนโปรแกรมควบคุมหุ่นยนต์เพื่อการศึกษา

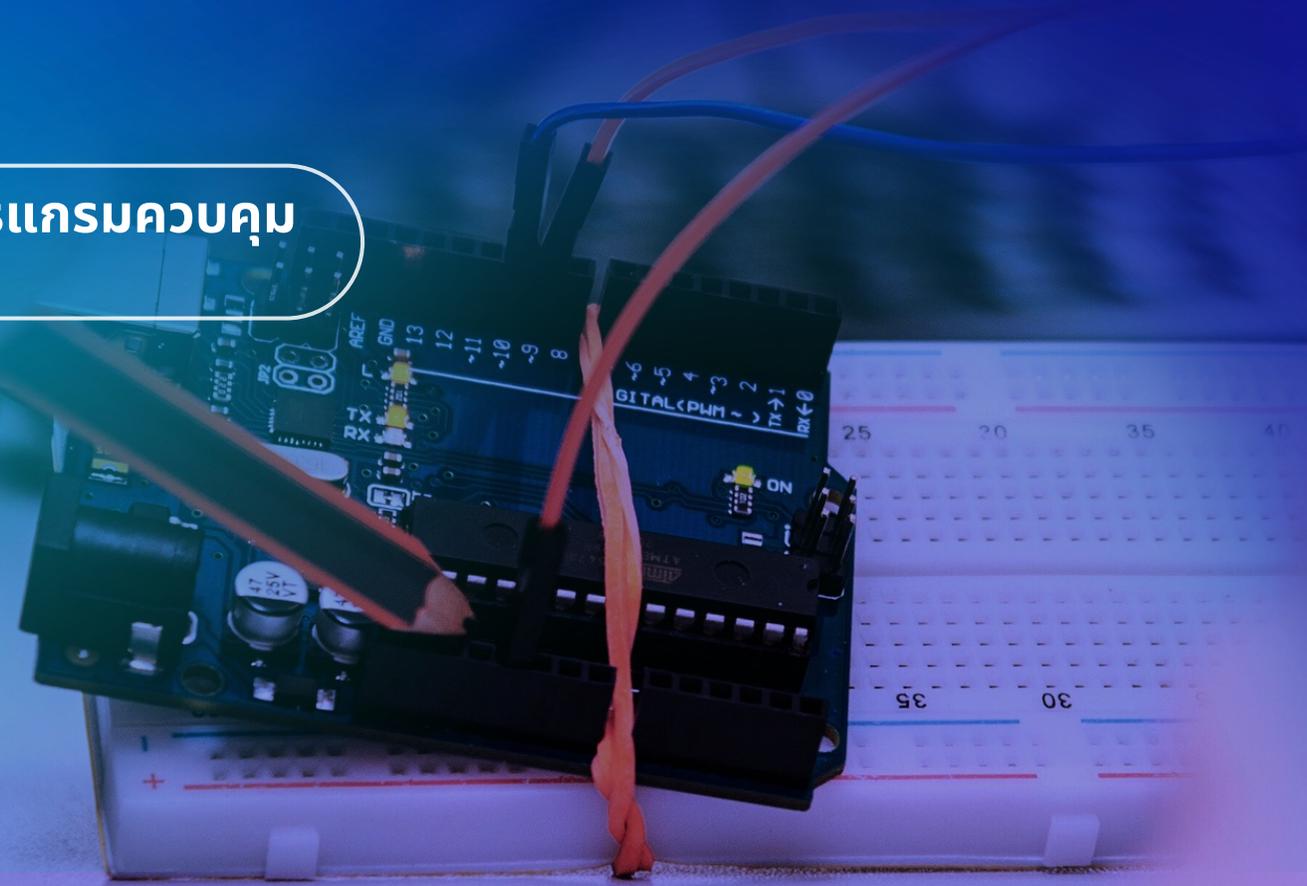
# Control Robotic Programming for Education

บทที่ 3 ไมโครคอนโทรลเลอร์ และการเขียนโปรแกรมควบคุม  
Arduino ด้วย Tinkercad



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

✉ email [nutthapat.ke@ssru.ac.th](mailto:nutthapat.ke@ssru.ac.th)



# รายวิชา DTC2301 การโปรแกรมควบคุมหุ่นยนต์เพื่อการศึกษา

## Control Robotic Programing for Education

3(2-2-5) หน่วยกิต

### คำอธิบายรายวิชา

แนวคิด ทฤษฎี ที่เกี่ยวข้องกับการเขียนโปรแกรมและการควบคุมหุ่นยนต์ หลักการ ส่วนประกอบ โครงสร้างและหน้าที่ของ หุ่นยนต์ คุณสมบัติของโปรแกรมภาษา ชนิดต่าง ๆ หลักการเบื้องต้น เกี่ยวกับ องค์ประกอบ ลักษณะคำสั่ง การเขียน โปรแกรม ขั้นตอนวิธี การวิเคราะห์ การ ออกแบบการเขียนโปรแกรมควบคุม หุ่นยนต์เพื่อการศึกษา

### 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** สะเต็มศึกษากับการพัฒนานวัตกรรมหุ่นยนต์ (STEAM4INNOVATOR Certificate)
- บทที่ 3** ไมโครคอนโทรลเลอร์ และการเขียนโปรแกรมควบคุม  
อาดูอิโน่ Arduino ด้วย Tinkercad
- บทที่ 4** การโปรแกรมควบคุมอาดูอิโน่ Arduino ร่วมกับจอ LCD
- บทที่ 5** การโปรแกรมควบคุมอาดูอิโน่ Arduino ร่วมกับ มอเตอร์เซอร์โว
- บทที่ 6** การโปรแกรมควบคุมอาดูอิโน่ Arduino ร่วมกับ เซนเซอร์ตรวจจับ  
ระยะห่างด้วยคลื่นอัลตราโซนิก
- บทที่ 7** การโปรแกรมควบคุมอาดูอิโน่ Arduino ร่วมกับเซนเซอร์ตรวจจับความชื้น
- บทที่ 8** การพัฒนาโครงงานด้านวิทยาการหุ่นยนต์เพื่อการศึกษา

# การวัดและการประเมินผลการเรียนรู้

<b>การมีส่วนร่วมในชั้นเรียน</b>	<b>10%</b>
<b>การมอบหมายงาน</b>	<b>30%</b>
<b>สอบกลางภาค</b>	<b>20%</b>
<b>สอบปลายภาค</b>	<b>20%</b>
<b>โครงการ</b>	<b>20%</b>

เกรด	ช่วงคะแนน	ผลการประเมิน	ค่าระดับคะแนน
A	86 - 100	ดีเยี่ยม	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	



นายปิ่น มาลากุล  
 ทักษิณ เวียงวั  
 ๑๑๑ ๑๑๑ ๑๑๑  
 ทักษิณ เวียงวั

นายปิ่น  
 ทักษิณ  
 ๑๑๑ ๑๑๑  
 ทักษิณ  
 ๑๑๑ ๑๑๑ ๑๑๑  
 ปิ่น มาลากุล

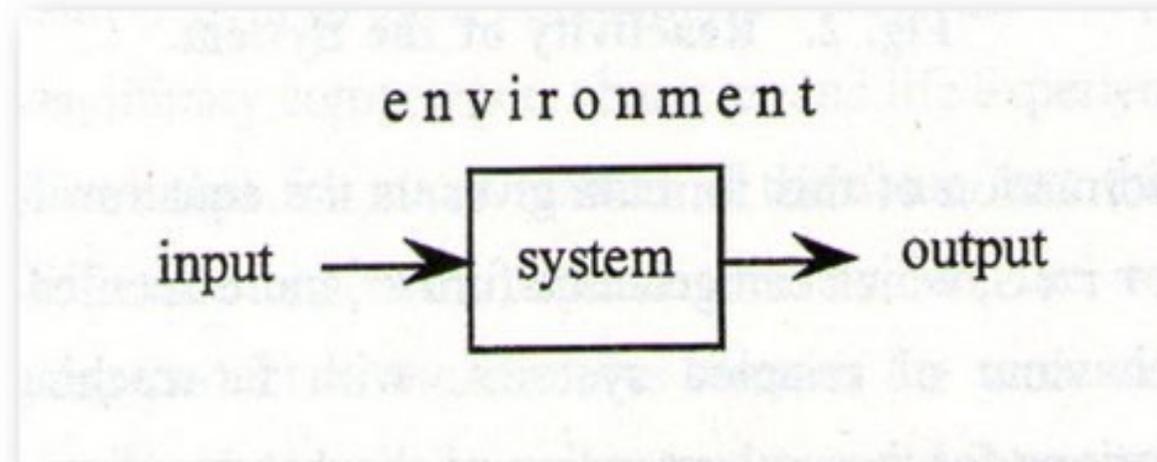
**ม.ล.ปิ่น มาลากุล** รัฐมนตรีว่าการกระทรวงศึกษาธิการ  
 ได้ไปเป็นประธานในพิธีแจกประกาศนียบัตรแก่นักเรียนและ  
 นักศึกษาวิทยาลัยครูสวนสุนันทา  
 ที่สำเร็จการศึกษาประจำปีการศึกษา ๒๕๐๖  
 บ่ายวันที่ ๑๓ มีนาคม ๒๕๐๗

# ไมโครคอนโทรลเลอร์ และการเขียนโปรแกรมควบคุม Arduino ด้วย Tinkercad

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

# ทฤษฎีระบบ Systems Theory

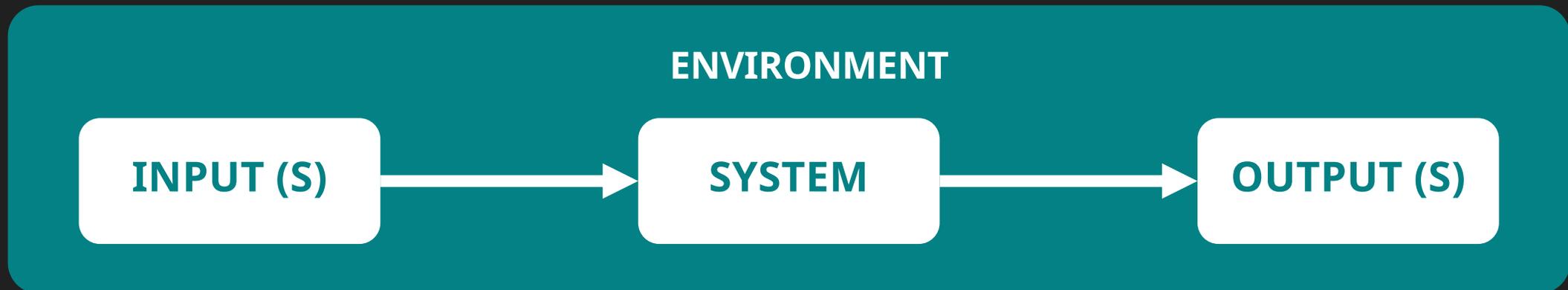
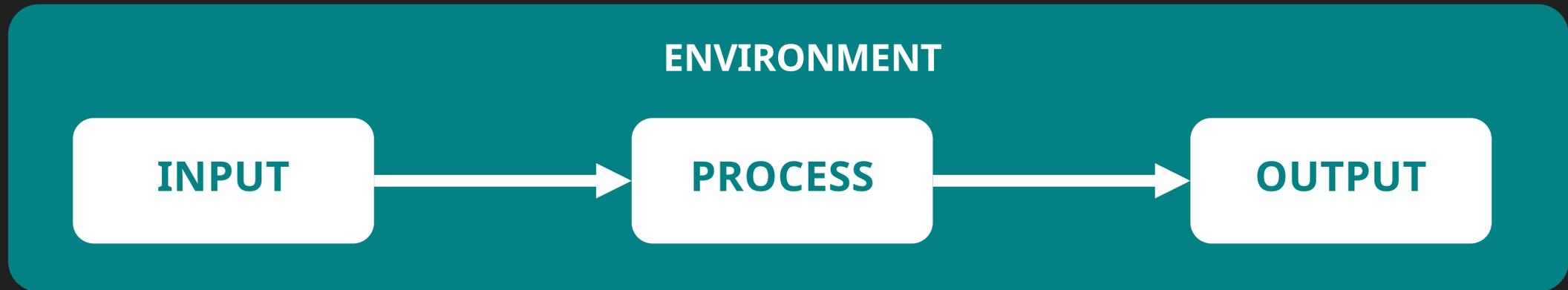
## Systems Theory



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

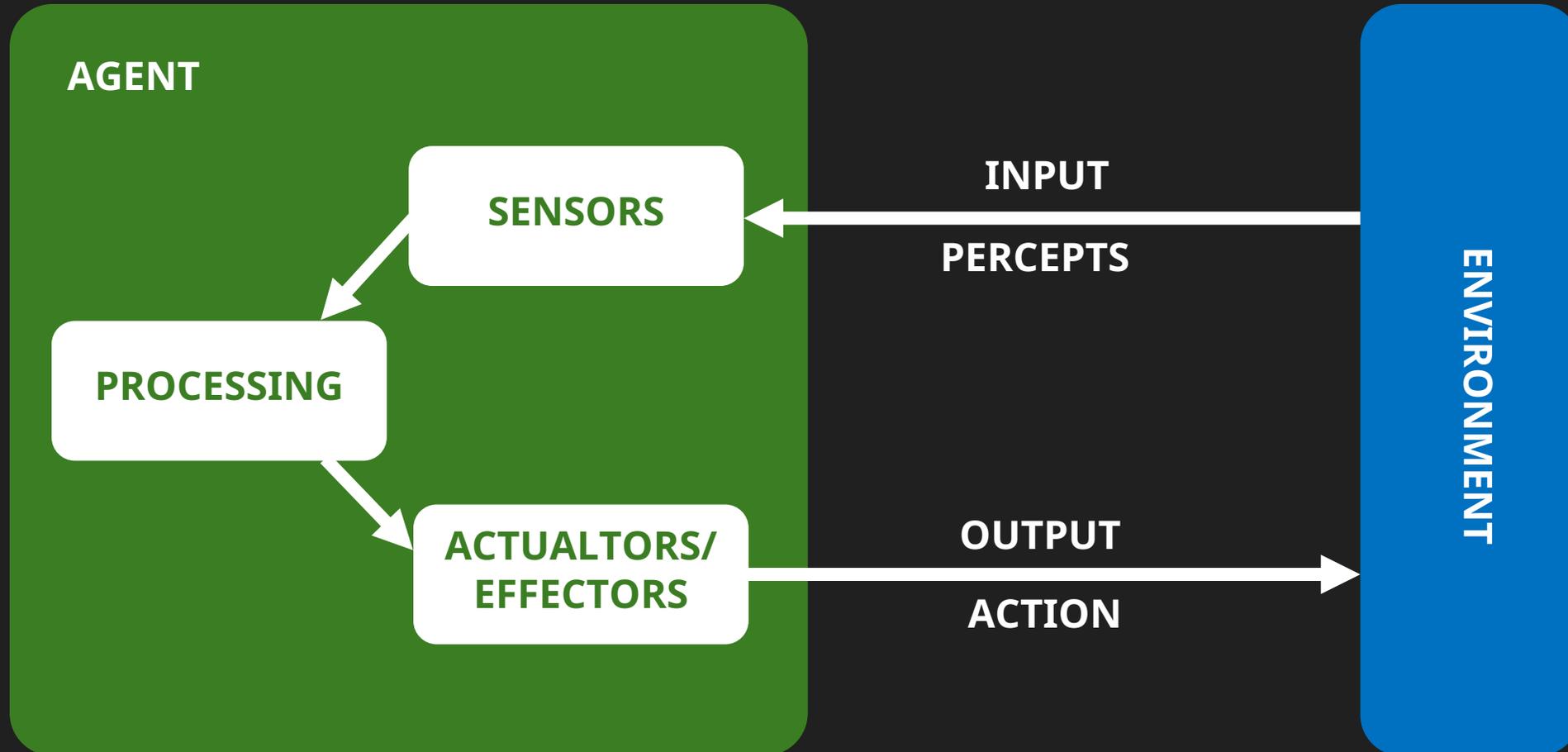
Sadowski, P (1999), *Systems Theory as an Approach to the Study of Literature: Origins and Functions of Literature*, E. Mellen Press, Lewiston, N.Y.

# ทฤษฎีระบบ Systems Theory

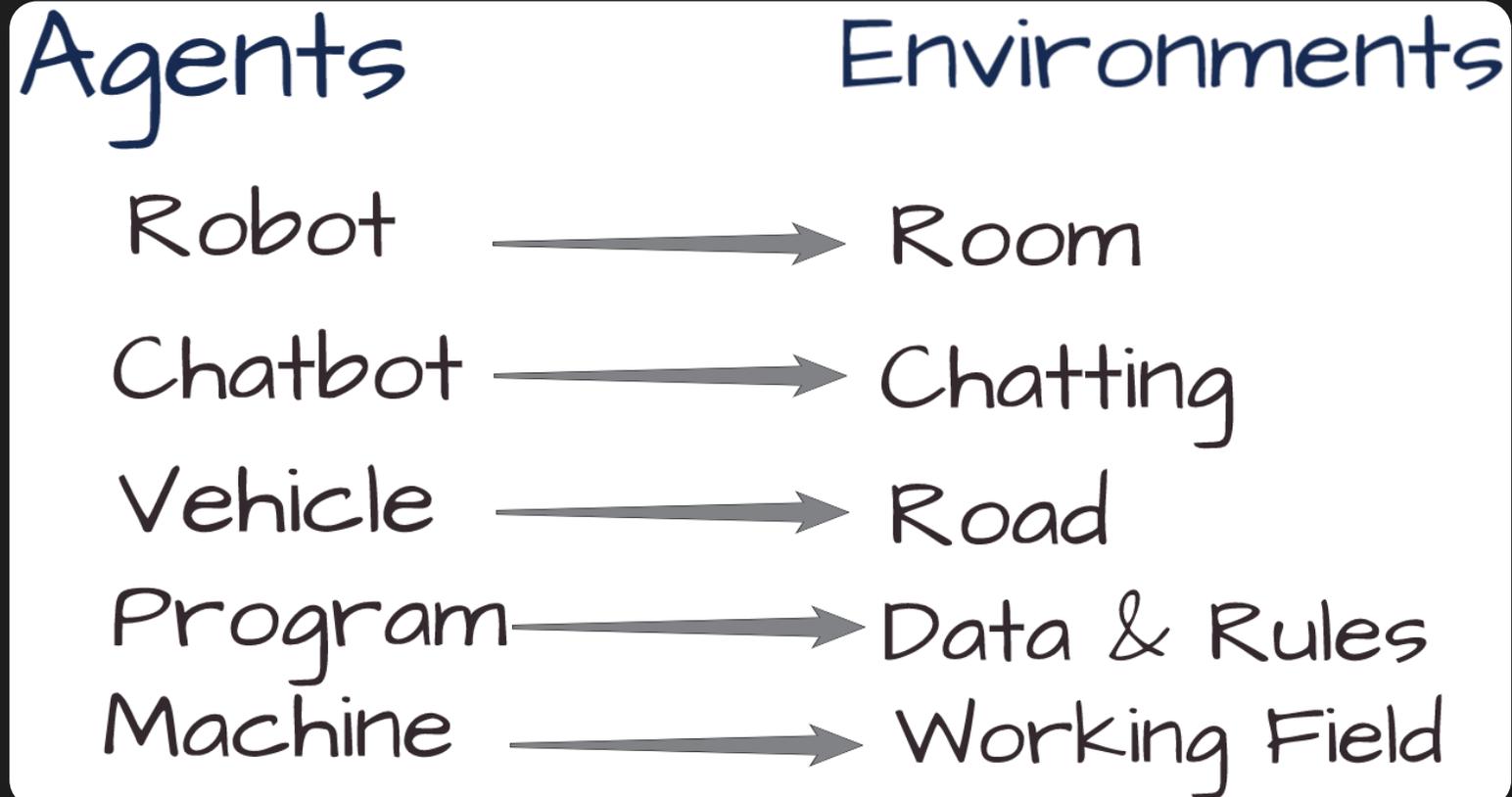


# ตัวแทน Agent

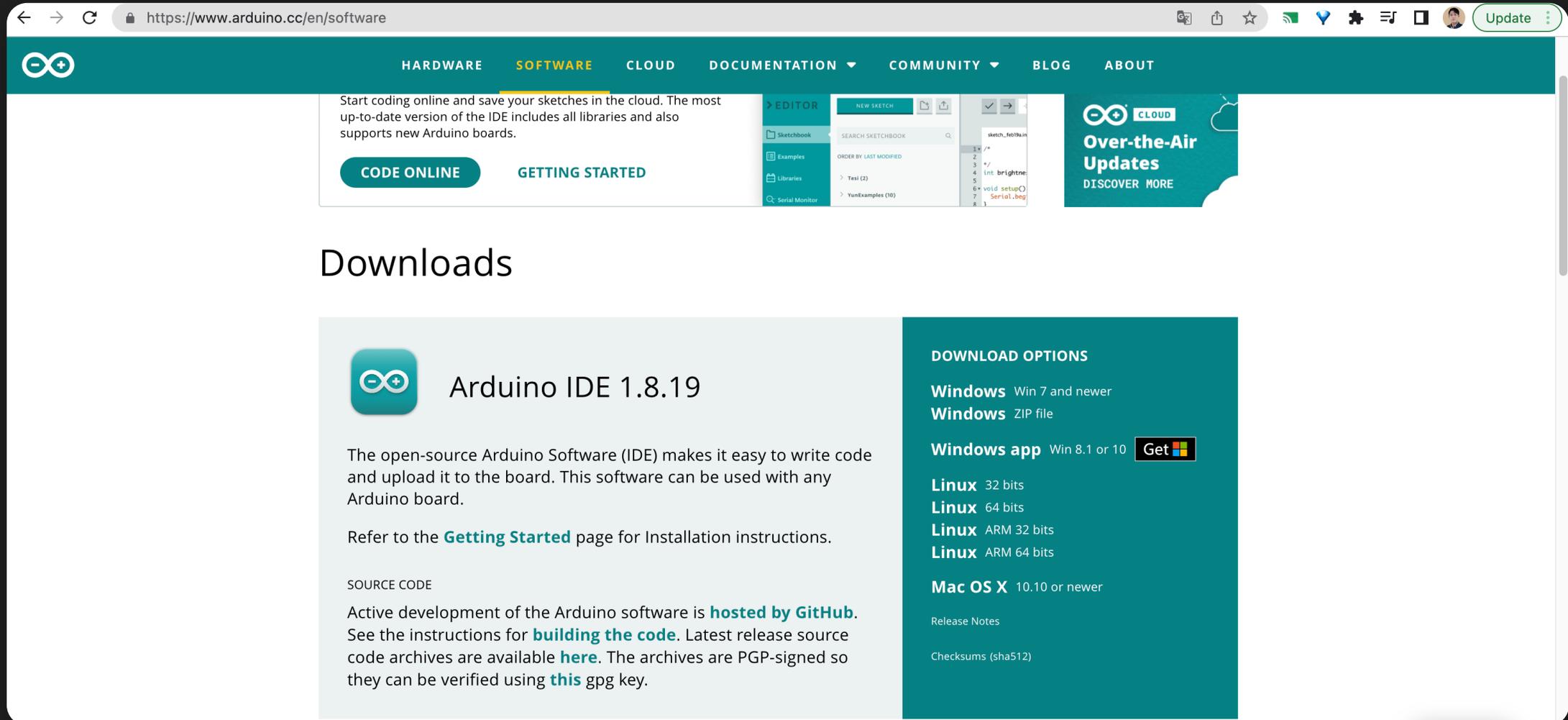
[Simulation คลิกที่นี่](#)



# ตัวแทน กับ สภาพแวดล้อม Agent and Environments



# ติดตั้ง Arduino IDE

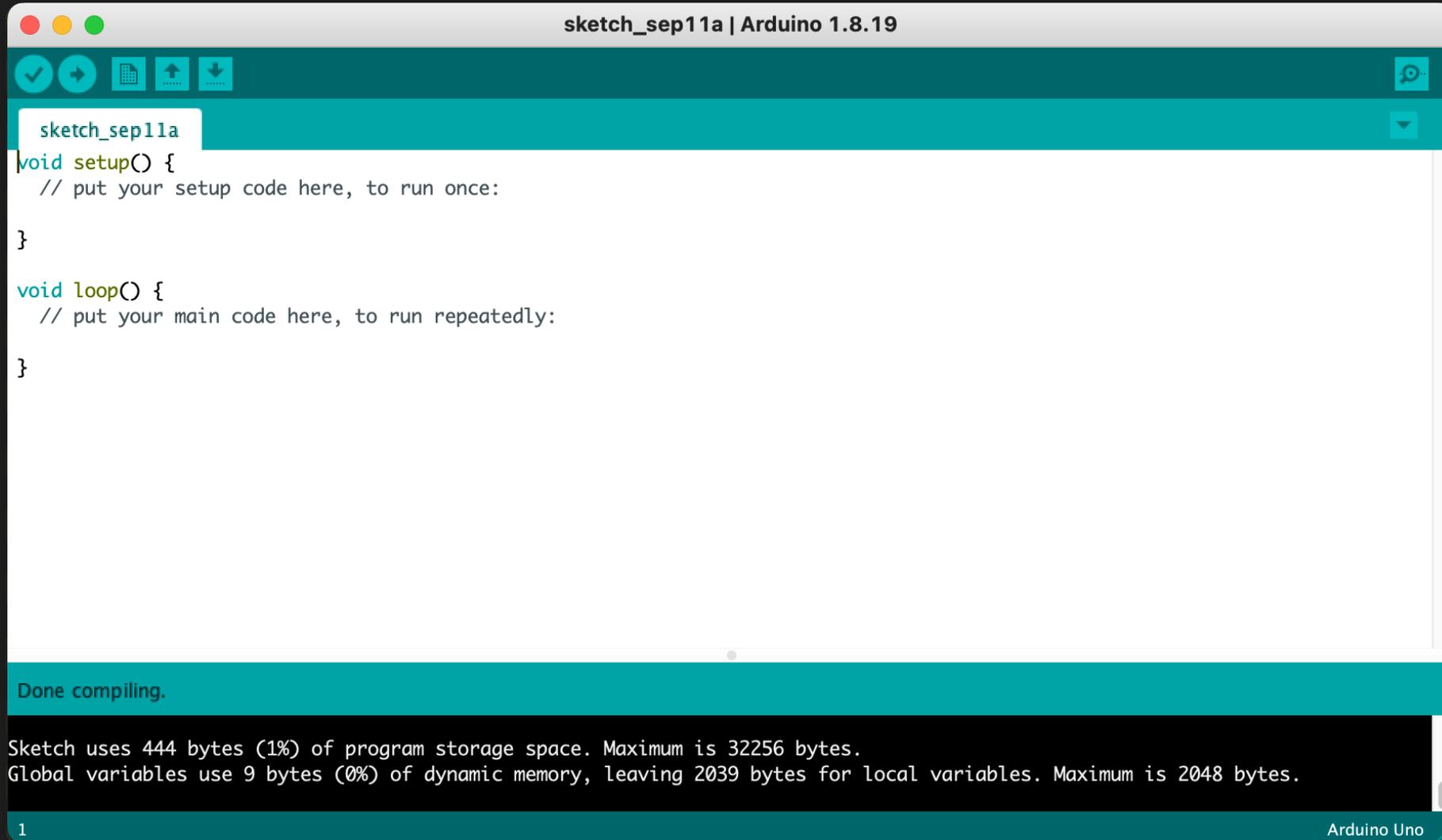


The screenshot shows the Arduino IDE website at <https://www.arduino.cc/en/software>. The navigation bar includes links for HARDWARE, SOFTWARE, CLOUD, DOCUMENTATION, COMMUNITY, BLOG, and ABOUT. The main content area features a 'Downloads' section for Arduino IDE 1.8.19. The download options are listed as follows:

Platform	Version	Download Link
Windows	Win 7 and newer	ZIP file
Windows app	Win 8.1 or 10	Get [Windows Store icon]
Linux	32 bits	[Download]
Linux	64 bits	[Download]
Linux	ARM 32 bits	[Download]
Linux	ARM 64 bits	[Download]
Mac OS X	10.10 or newer	[Download]

Additional links include Release Notes and Checksums (sha512).

# ติดตั้ง Arduino IDE



The screenshot shows the Arduino IDE interface. The title bar reads "sketch\_sep11a | Arduino 1.8.19". The main editor area contains the following code:

```
sketch_sep11a
void setup() {
  // put your setup code here, to run once:
}

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

Below the code editor, a status bar indicates "Done compiling." and provides memory usage statistics:

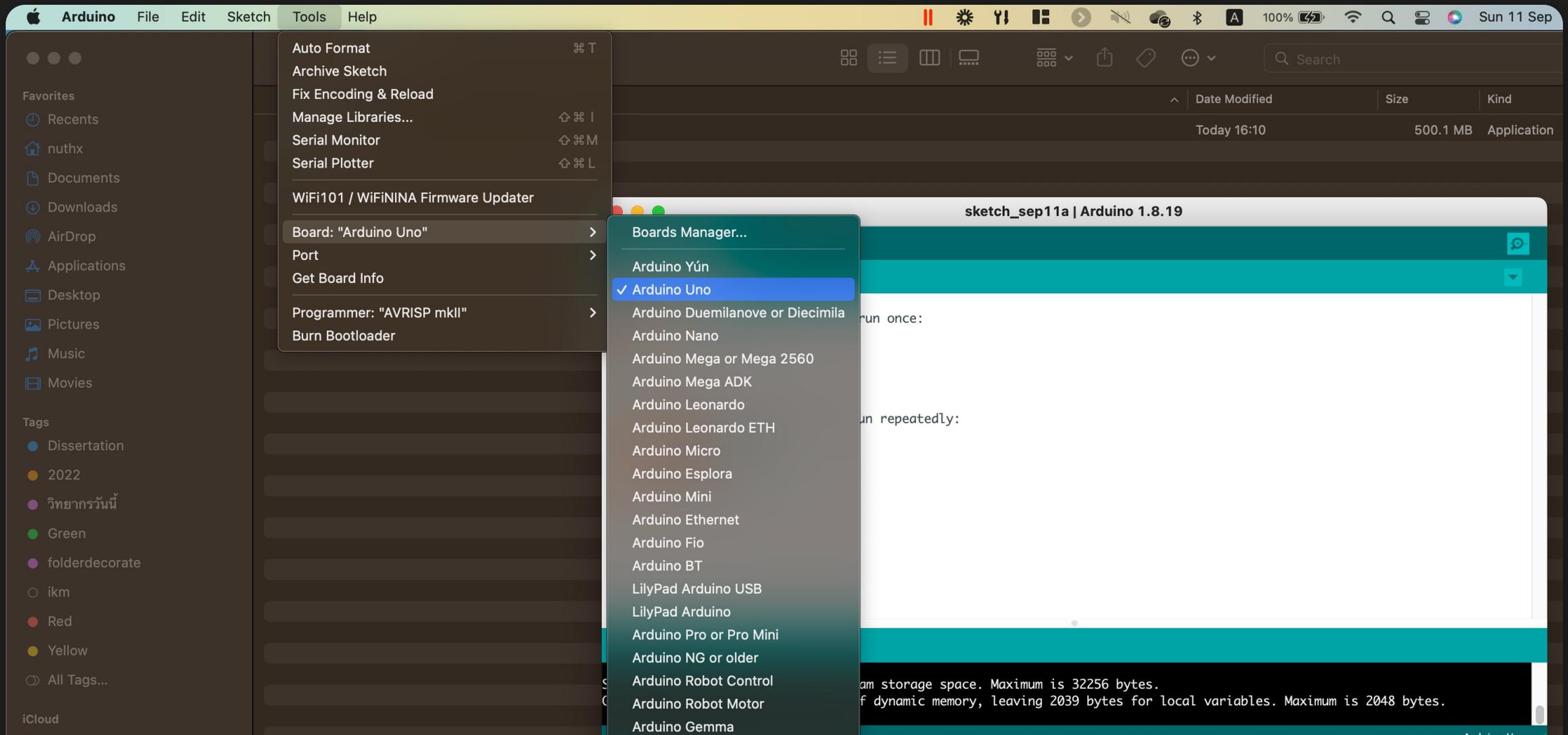
```
Sketch uses 444 bytes (1%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.
```

The bottom status bar shows "1" on the left and "Arduino Uno" on the right.

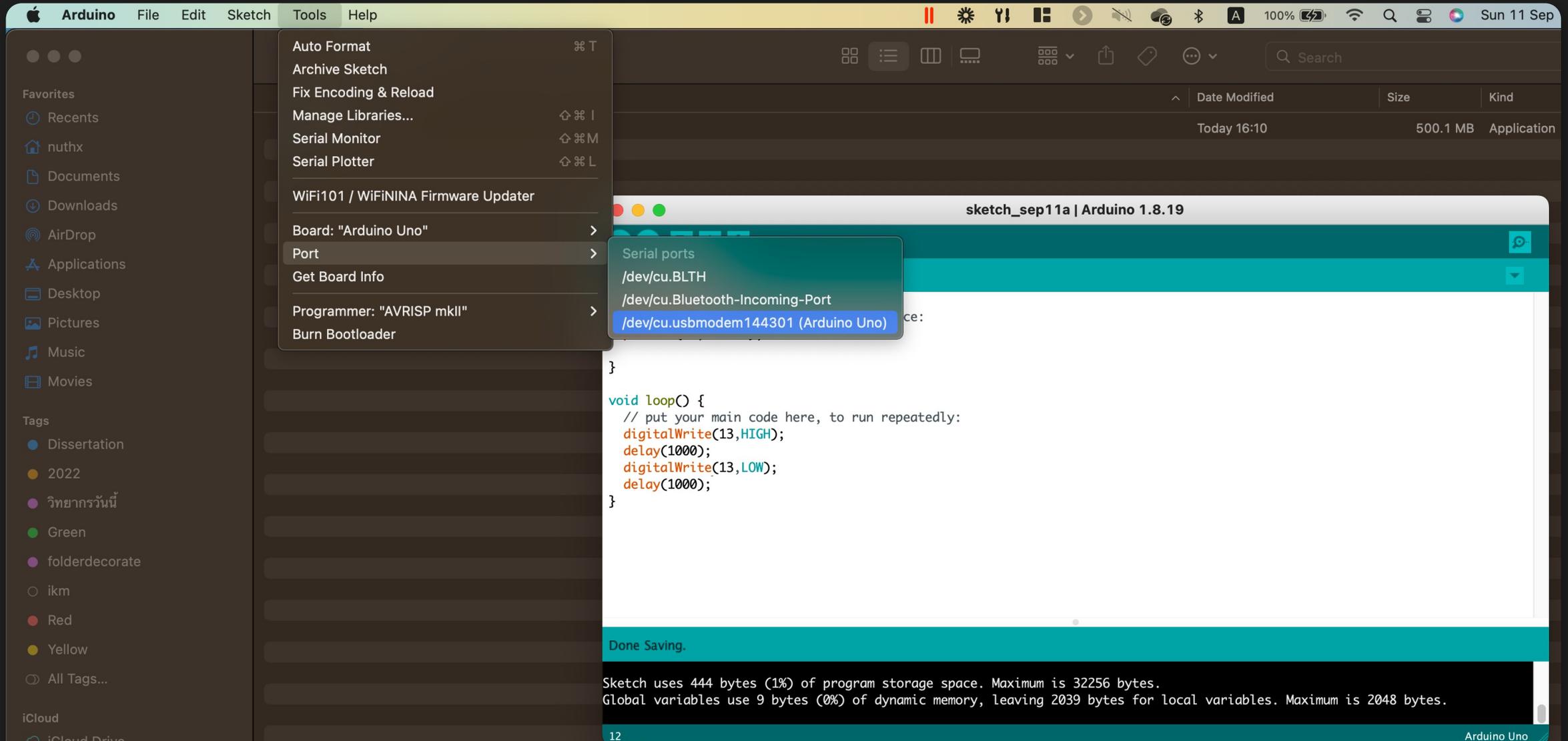
# เชื่อมต่อ Arduino กับสาย USB



# เชื่อมต่อ Arduino กับสาย USB



# เชื่อมต่อ Arduino กับสาย USB



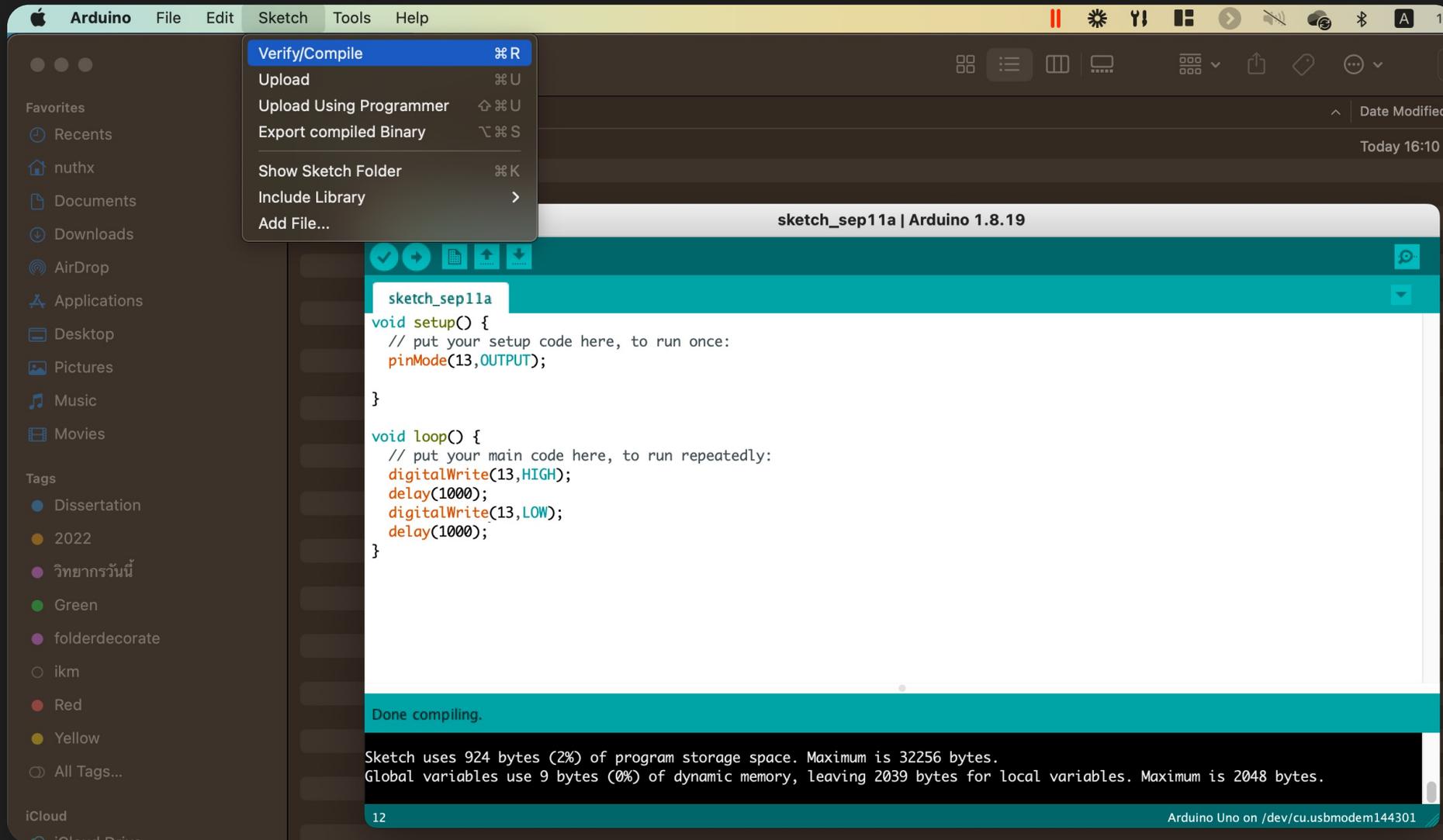
Arduino IDE Interface showing the 'Tools' menu with the 'Port' submenu open, highlighting the selected port: `/dev/cu.usbmodem144301 (Arduino Uno)`.

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

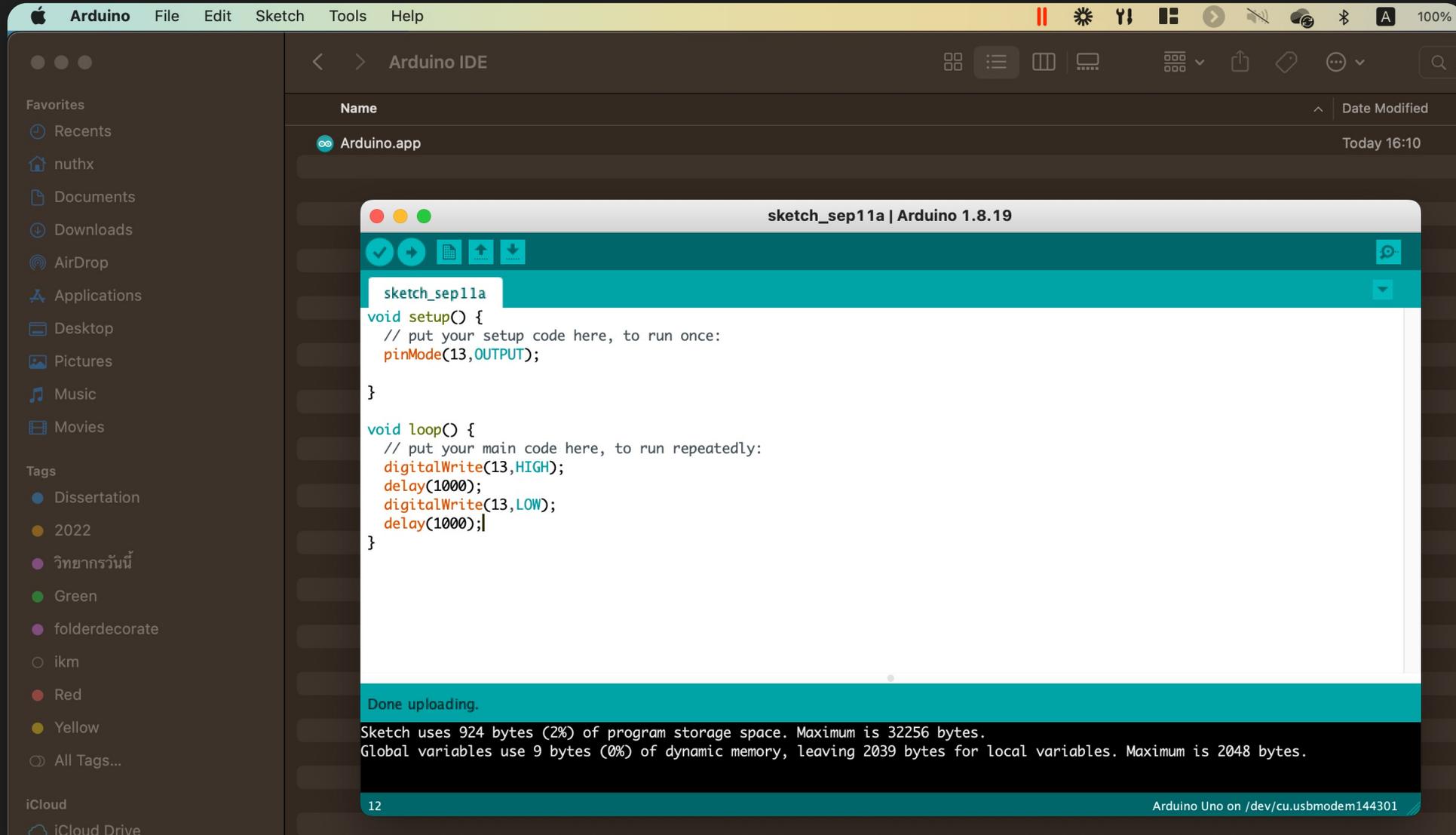
Done Saving.

Sketch uses 444 bytes (1%) of program storage space. Maximum is 32256 bytes.  
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

# เชื่อมต่อ Arduino กับสาย USB



# เชื่อมต่อ Arduino กับสาย USB



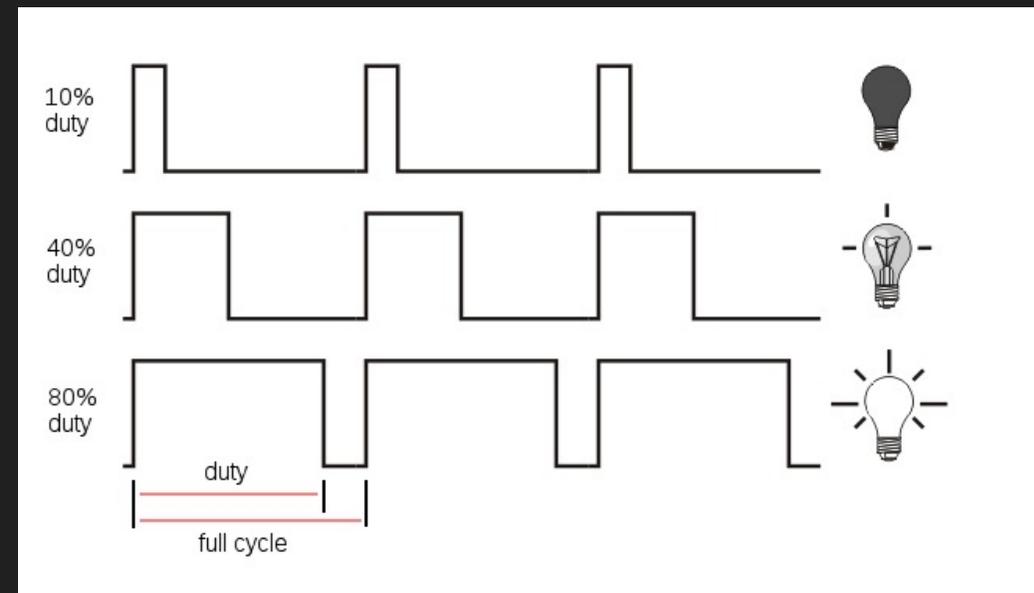
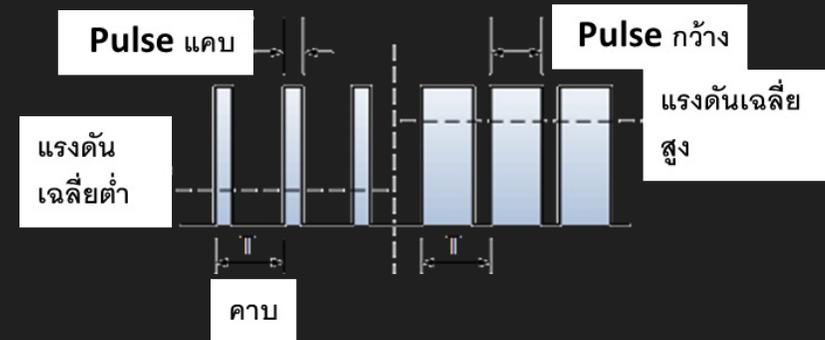
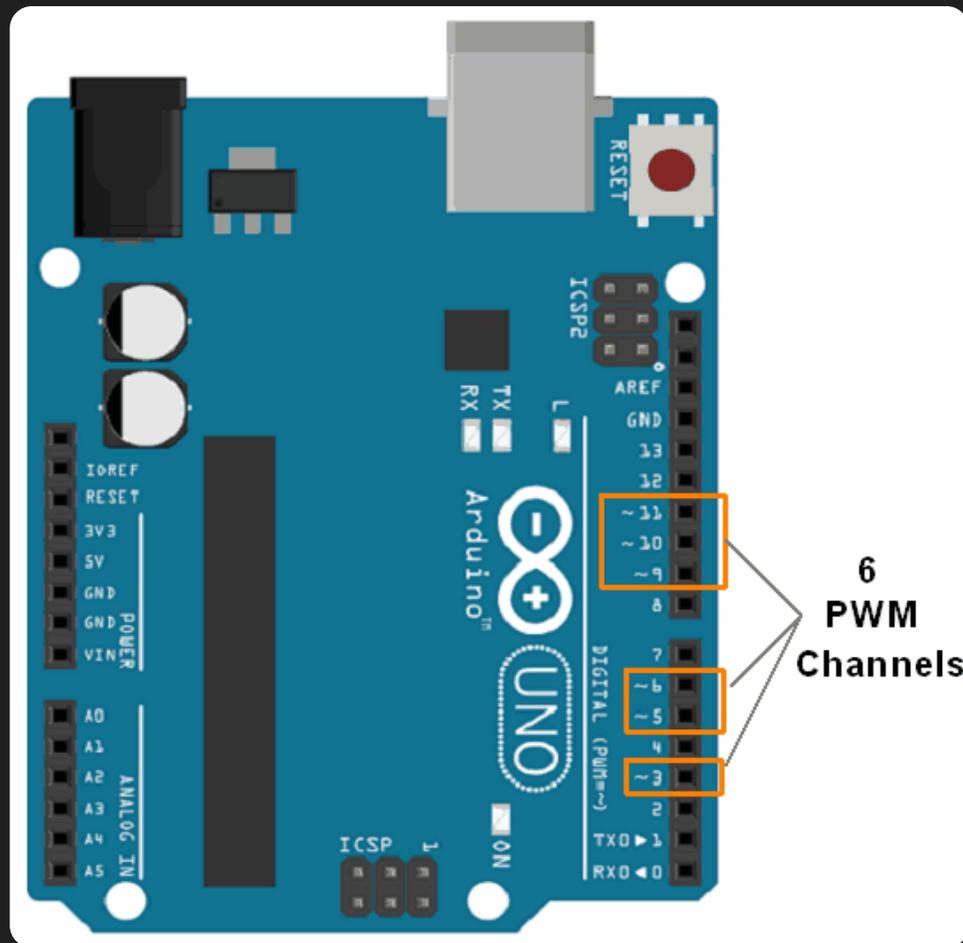
The screenshot displays the Arduino IDE environment. The main window shows a sketch named 'sketch\_sep11a' with the following code:

```
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);  
}
```

Below the code editor, a status bar indicates 'Done uploading.' and provides memory usage information: 'Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes. Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.'

The bottom status bar shows '12' and 'Arduino Uno on /dev/cu.usbmodem144301'.

# Pulse Width Modulation: PWM



# Pulse Width Modulation: PWM

- PWM หรือ Pulse Width Modulation คือ เทคนิคการควบคุมแรงดันไฟฟ้าเฉลี่ยที่ส่งไปยังอุปกรณ์ไฟฟ้า โดยการปรับความกว้างของพัลส์สัญญาณดิจิทัล

## หลักการทำงาน

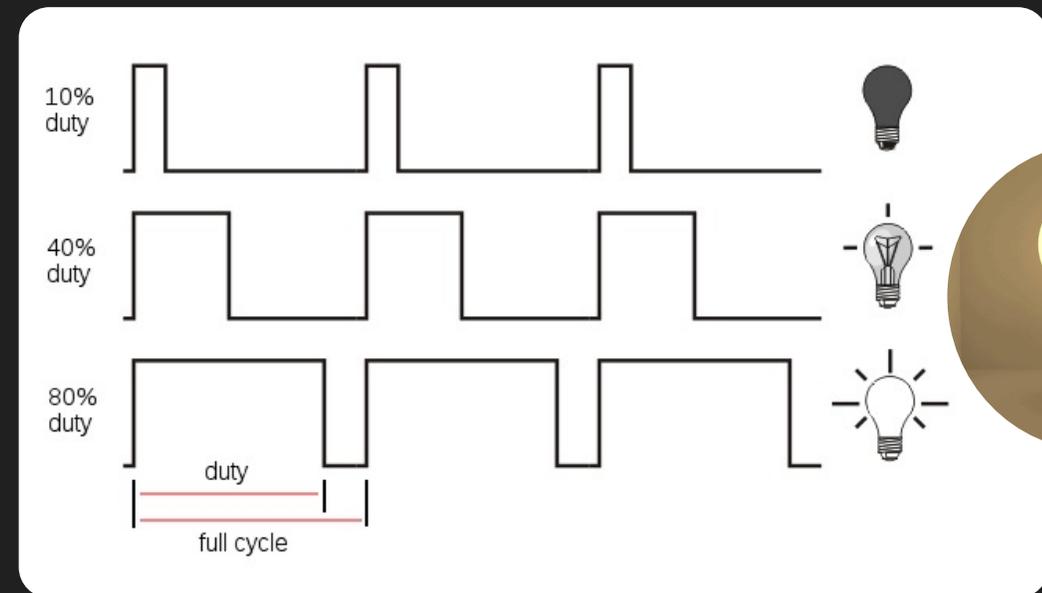
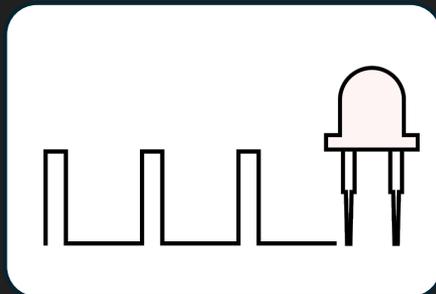
- สร้างสัญญาณพัลส์ที่มีความถี่คงที่
- ปรับเปลี่ยนช่วงเวลาสัญญาณเป็น ON (ความกว้างของพัลส์)
- ค่าเฉลี่ยของแรงดันไฟฟ้าจะขึ้นอยู่กับอัตราส่วนระหว่างช่วง ON และ OFF

# 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 คลิ๊กที่นี่](#)

- PWM หรือ Pulse Width Modulation คือ เทคนิคการควบคุมแรงดันไฟฟ้าเฉลี่ยที่ส่งไปยังอุปกรณ์ไฟฟ้า โดยการปรับความกว้างของพัลส์สัญญาณดิจิทัล

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

## 10% Duty Cycle

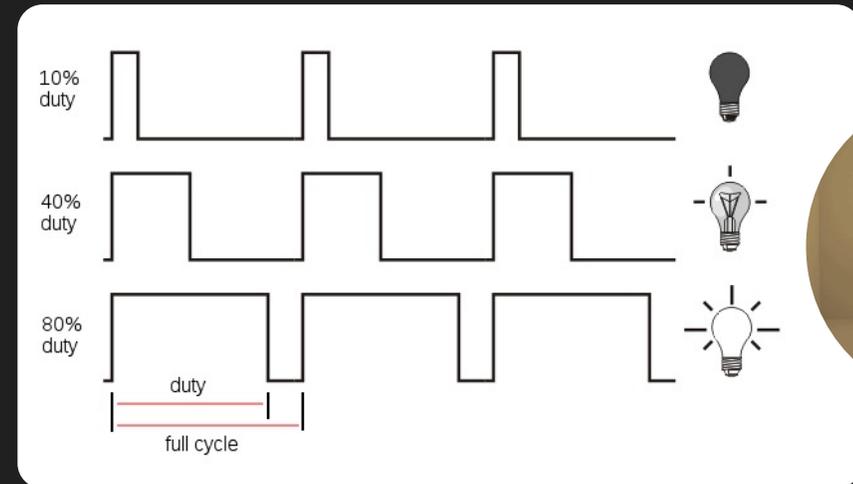
สัญญาณ ON เพียงช่วงสั้นๆ และ OFF เป็นส่วนใหญ่ ส่งผลให้หลอดไฟ LED สว่างน้อยที่สุด (แทบจะดับ)

## 40% Duty Cycle

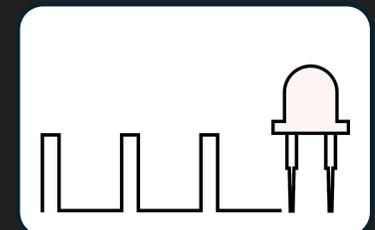
สัญญาณ ON นานขึ้น แต่ยังคง OFF มากกว่า ON หลอดไฟ LED สว่างขึ้นในระดับปานกลาง

## 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

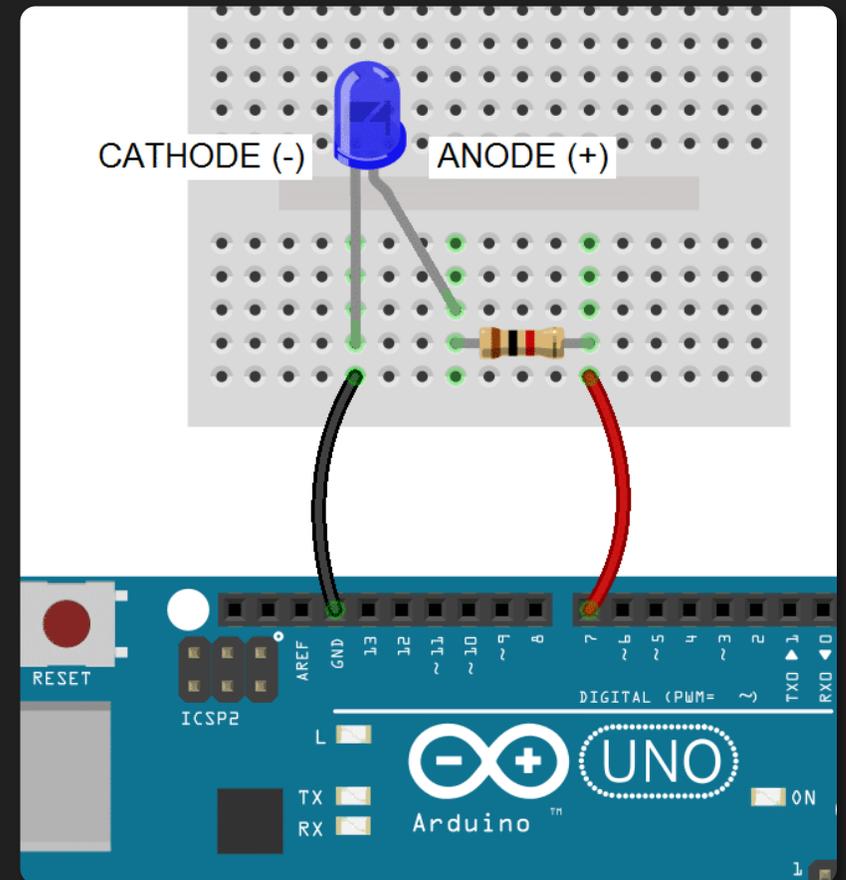
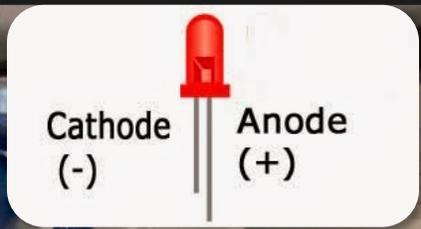
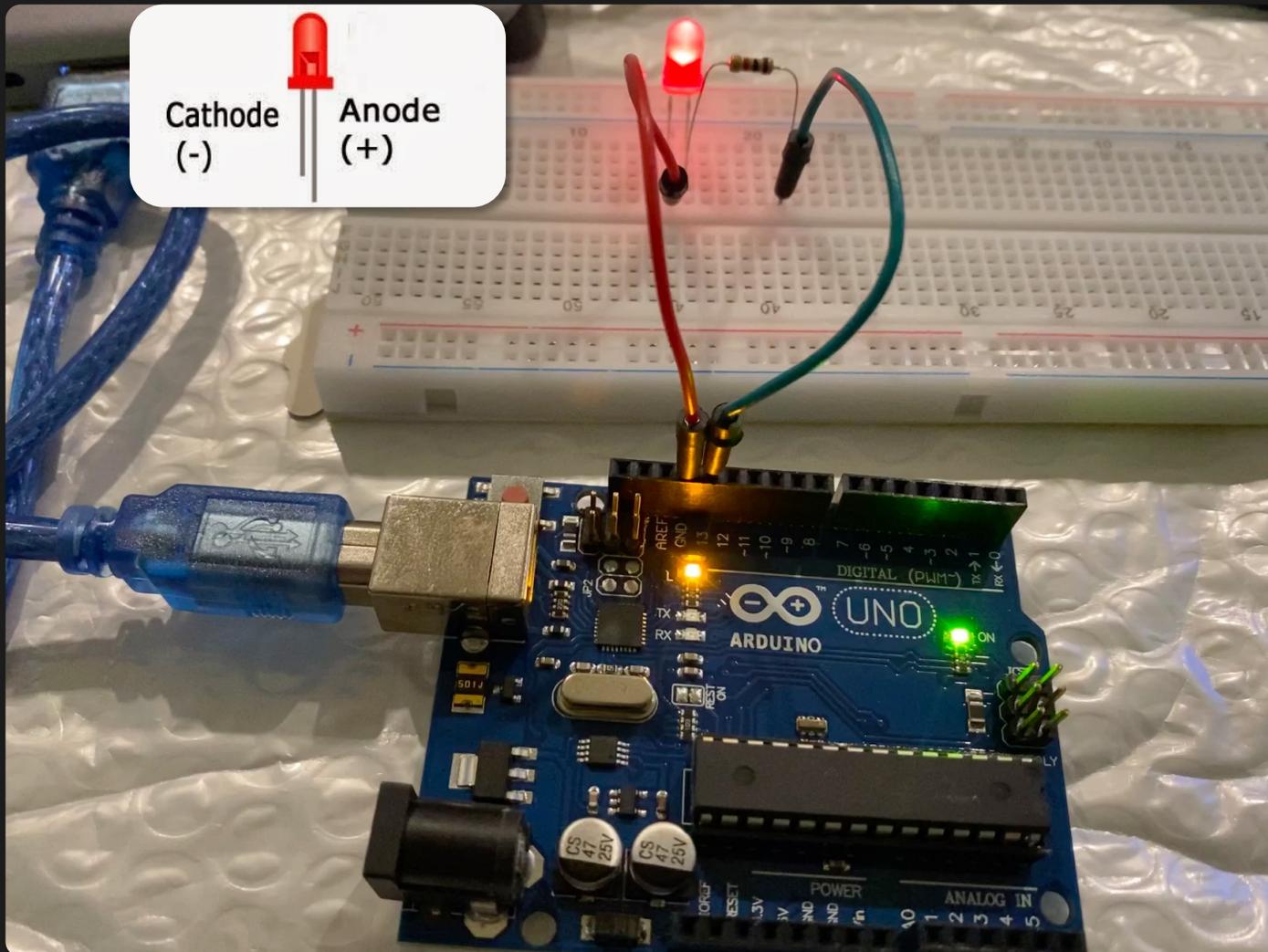
## ข้อดี

- ประสิทธิภาพสูง เนื่องจากการสูญเสียพลังงานต่ำ
- ควบคุมได้แม่นยำ
- ใช้งานง่ายกับระบบดิจิทัลและไมโครคอนโทรลเลอร์

## การใช้งานทั่วไป

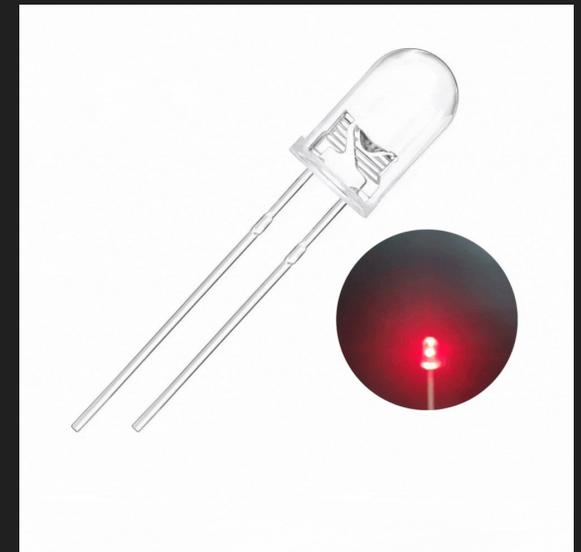
- อุปกรณ์อิเล็กทรอนิกส์
- ระบบควบคุมอัตโนมัติ
- ยานยนต์
- อุปกรณ์ไฟฟ้าในบ้าน

# การโปรแกรมเพื่อควบคุมหลอด LED

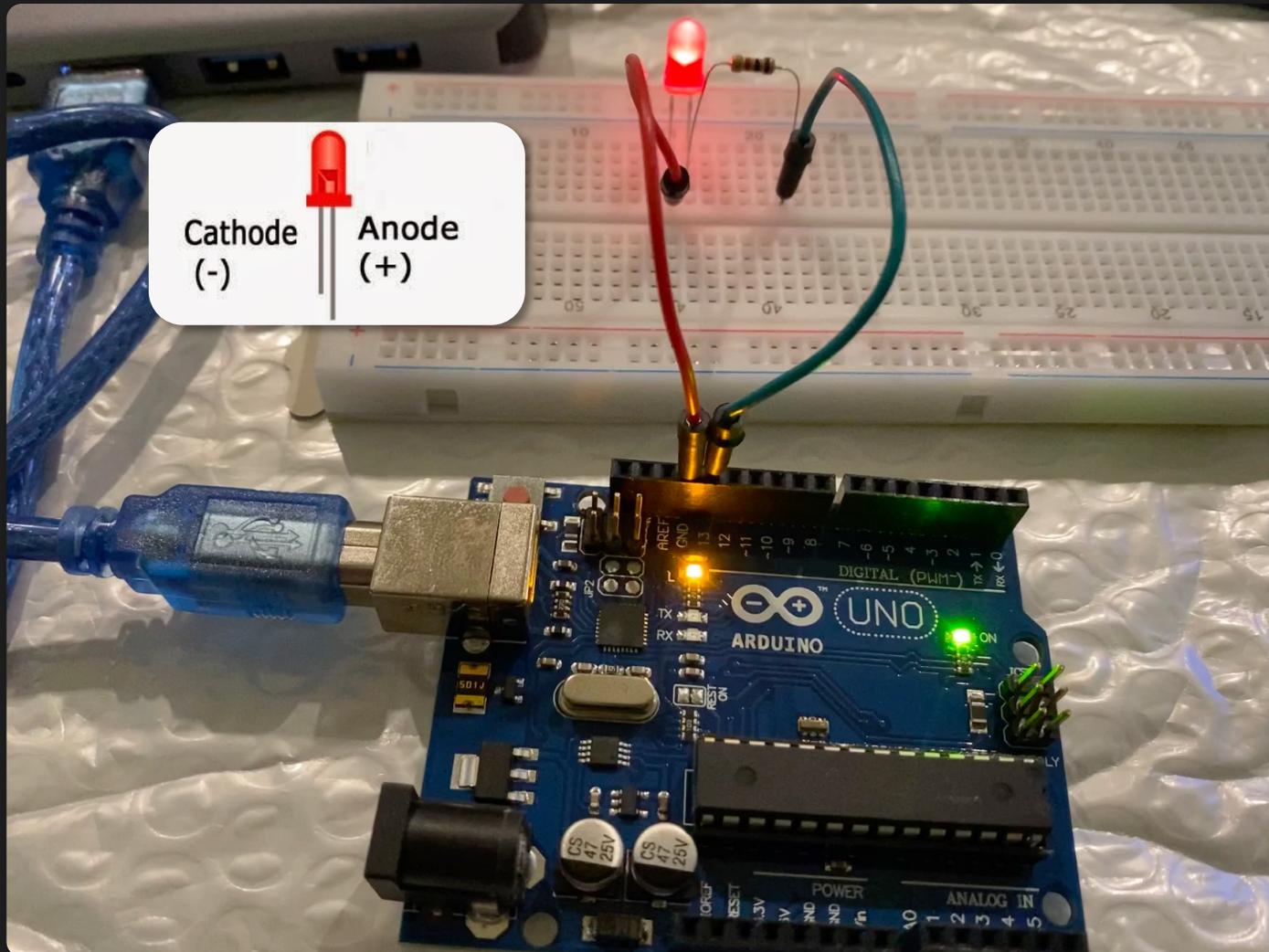


# LED หรือ Light Emitting Diode

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

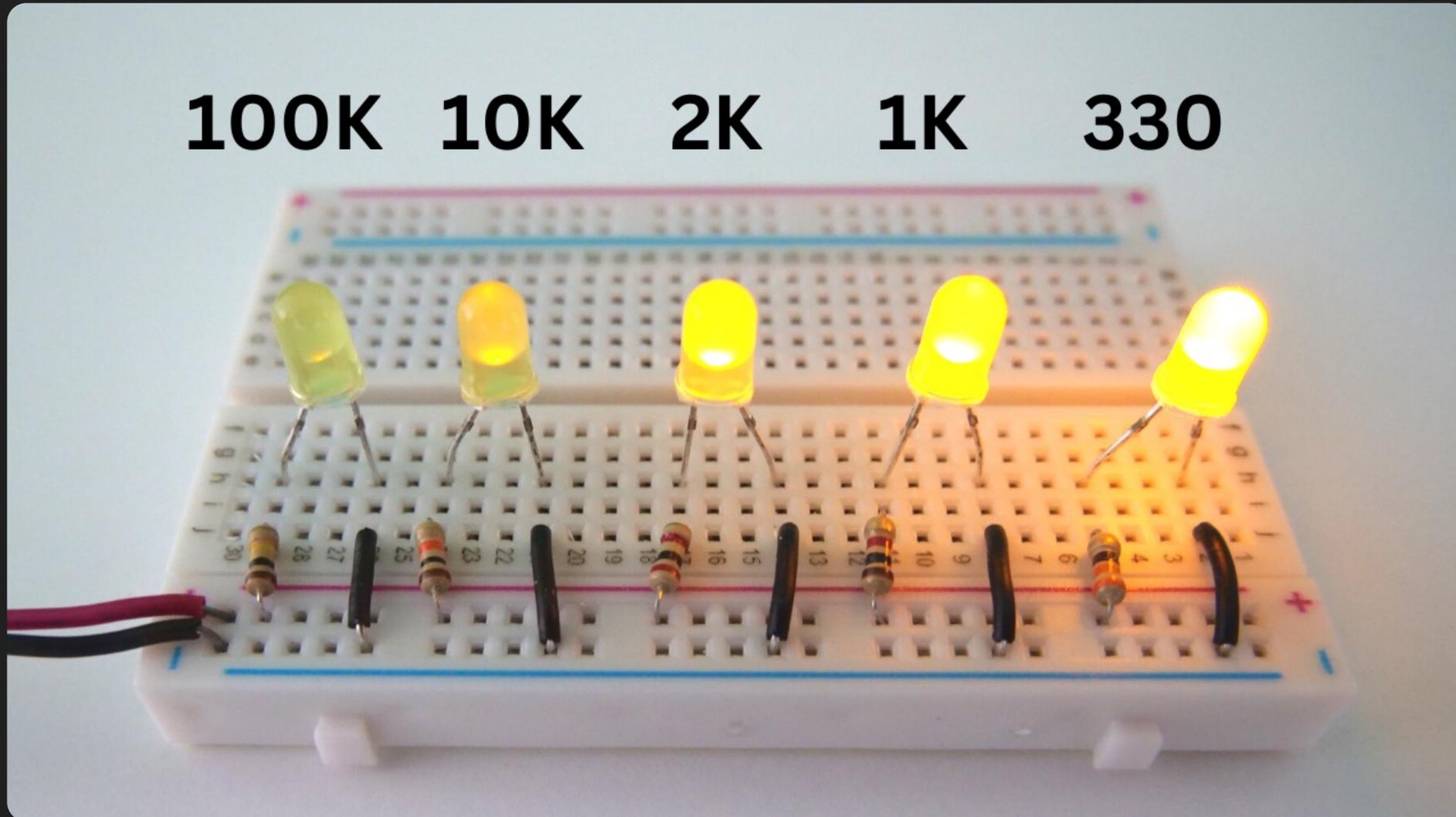


# หลอด LED



```
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 32256 bytes.  
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes free.
```

# ตัวต้านทาน (Resistor)



## การคำนวณค่าตัวต้านทาน

$$R = (V_s - V_f) / (I_f * N)$$

โดยที่

R = ค่าความต้านทานที่ต้องการ (โอห์ม)

V<sub>s</sub> = แรงดันแหล่งจ่าย (โวลต์)

V<sub>f</sub> = แรงดันตกคร่อม LED (โวลต์)

I<sub>f</sub> = กระแสที่ต้องการให้ไหลผ่าน

LED แต่ละหลอด (แอมแปร์)

N = จำนวน LED (ในที่นี้คือ 10)

## ตัวอย่างการคำนวณ

V<sub>s</sub> = 5V (Arduino)

V<sub>f</sub> = 2V

I<sub>f</sub> = 20mA

N = 10 หลอด

$$R = (5 - 2) / (0.02 * 10) = 15 \text{ โอห์ม}$$

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.พินดิจิทัล (Digital pins)

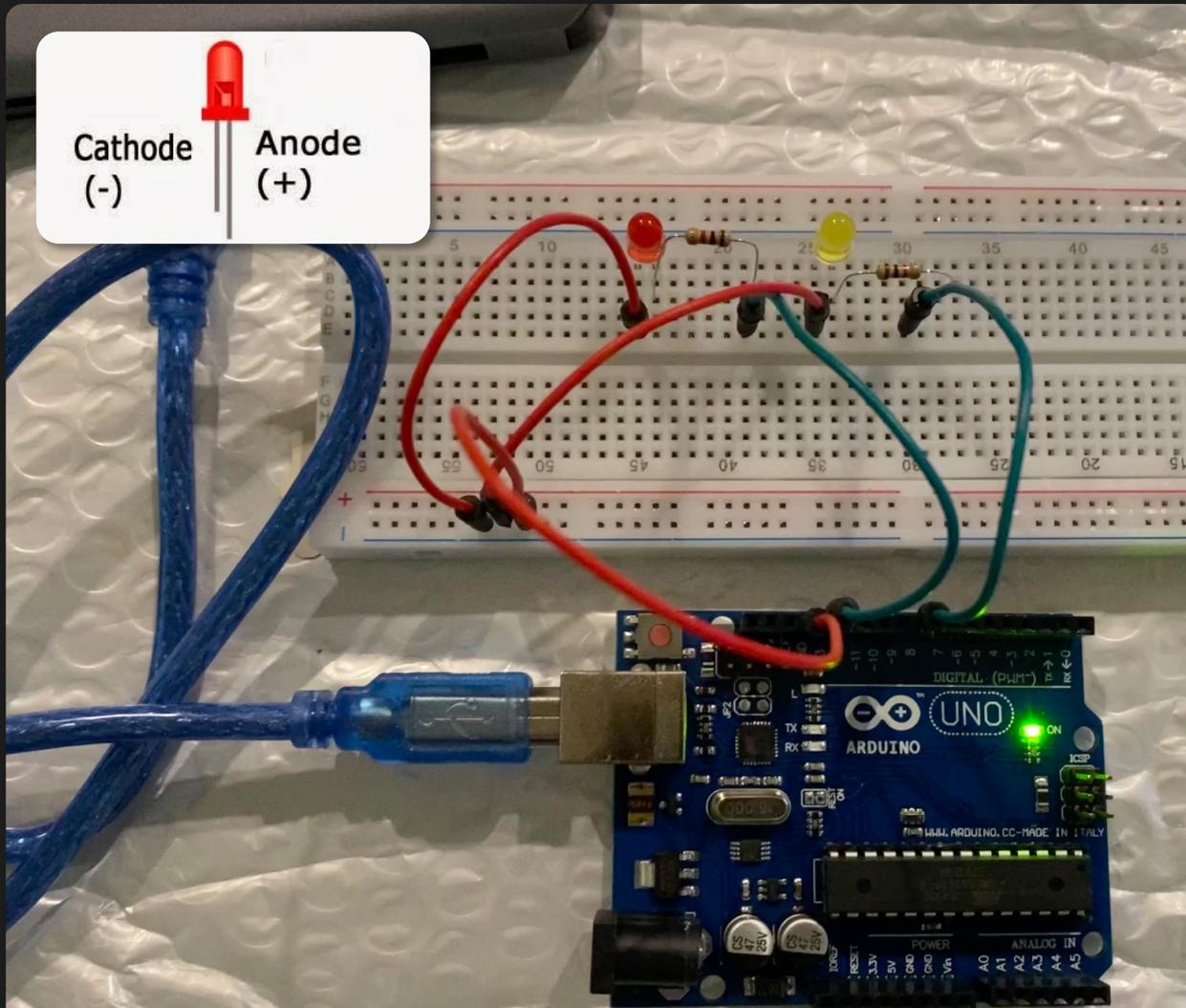
- เมื่อตั้งค่าเป็น OUTPUT และกำหนดให้เป็น HIGH จะจ่ายไฟ 5 โวลต์
- บางรุ่น เช่น Arduino Due จ่ายไฟที่ 3.3 โวลต์

## 3.พิน VIN

- ในกรณีที่ Arduino ได้รับไฟเลี้ยงผ่านแจ็กไฟ พิน VIN จะจ่ายไฟเท่ากับแรงดันที่ป้อนเข้ามา (ก่อนการลดแรงดัน)

สำหรับการต่อ LED โดยตรง มักจะใช้พินดิจิทัลที่จ่ายไฟ 5 โวลต์

# หลอด LED



```
Arduino1_Nutthapat
Arduino1_Nutthapat
void setup() {
  // put your setup code here, to run once:
  pinMode(13,OUTPUT);
  pinMode(8,OUTPUT);
}

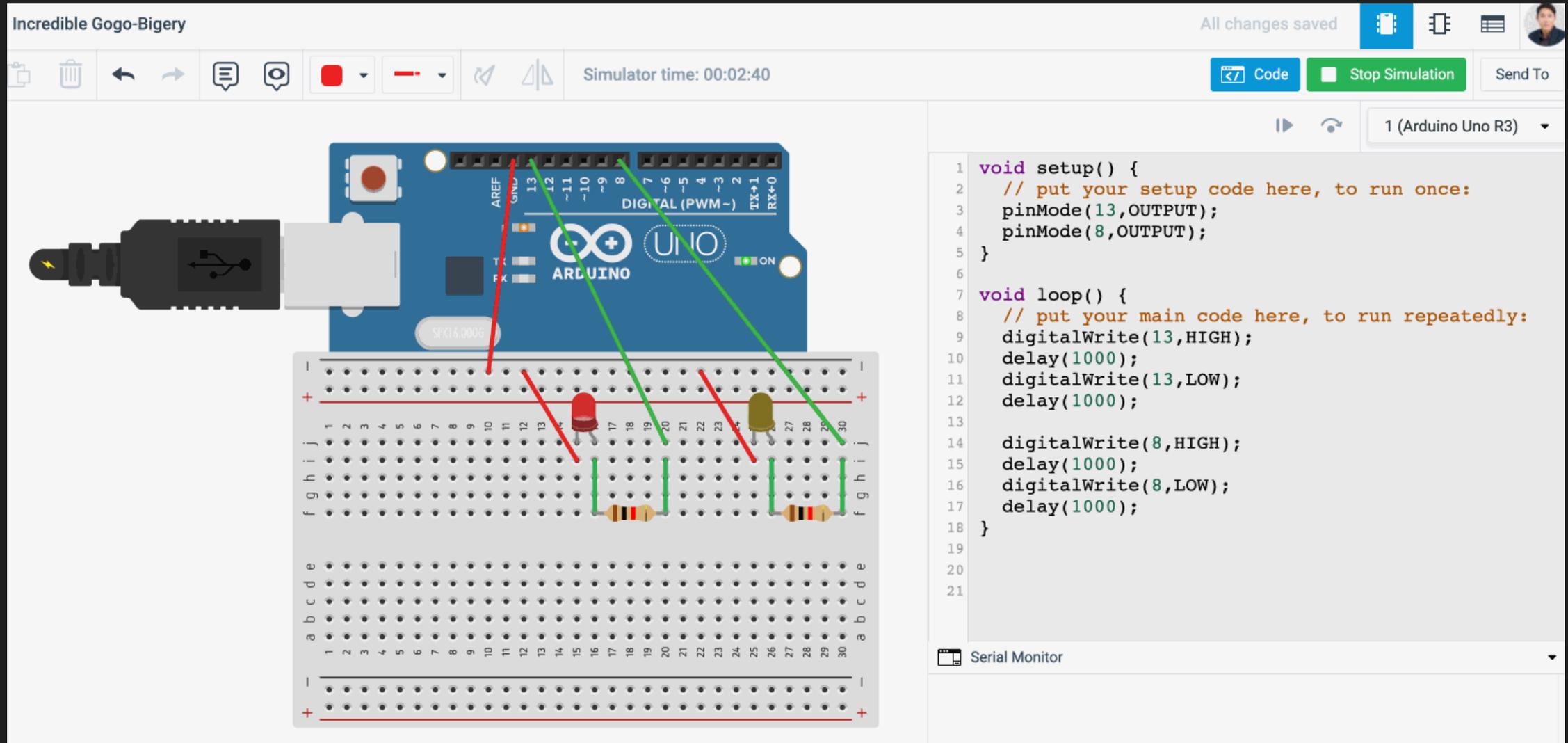
void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(13,HIGH);
  delay(1000);
  digitalWrite(13,LOW);
  delay(1000);

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

Done uploading.
```

Sketch uses 980 bytes (3%) of program storage space. Maximum is 32256 bytes. Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes free.

# การโปรแกรมเพื่อควบคุมหลอด LED

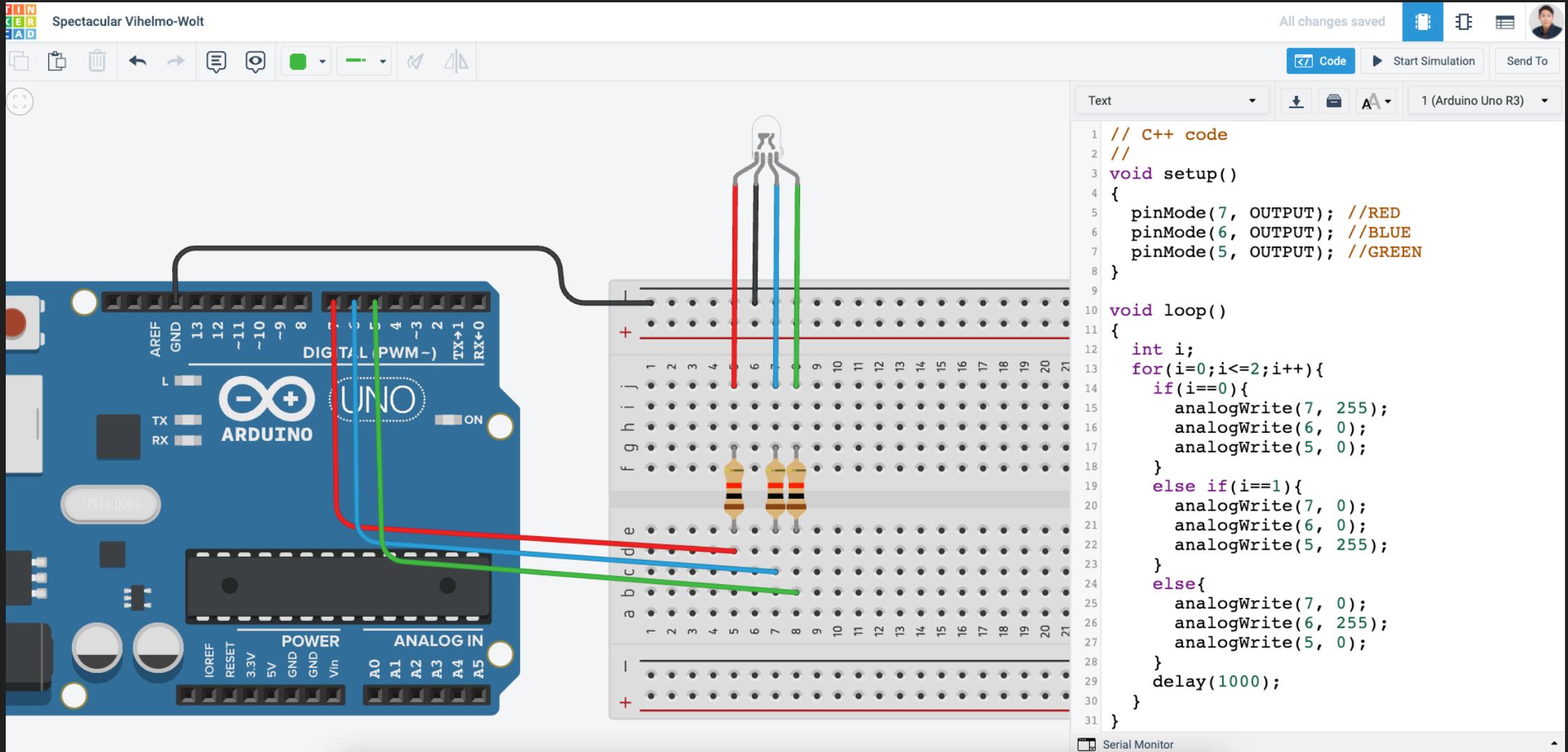


The image shows a screenshot of the Arduino IDE simulator interface. On the left, a blue Arduino Uno R3 board is connected to a breadboard. The breadboard contains two LEDs (one red, one green) and two resistors. Wires connect the LEDs to digital pins 13 and 8 on the Arduino board. The breadboard is populated with components: a red LED connected to pin 13, a green LED connected to pin 8, and two resistors connected to ground. The Arduino board is connected to a USB cable. The simulator interface includes a toolbar with icons for file operations, simulation control, and a 'Code' button. The 'Code' button is highlighted, and the code editor shows the following code:

```
1 void setup() {  
2   // put your setup code here, to run once:  
3   pinMode(13,OUTPUT);  
4   pinMode(8,OUTPUT);  
5 }  
6  
7 void loop() {  
8   // put your main code here, to run repeatedly:  
9   digitalWrite(13,HIGH);  
10  delay(1000);  
11  digitalWrite(13,LOW);  
12  delay(1000);  
13  
14  digitalWrite(8,HIGH);  
15  delay(1000);  
16  digitalWrite(8,LOW);  
17  delay(1000);  
18 }  
19  
20  
21
```

Below the code editor is a 'Serial Monitor' window, which is currently empty. The simulator time is displayed as 00:02:40. The top right corner shows 'All changes saved' and a user profile icon.

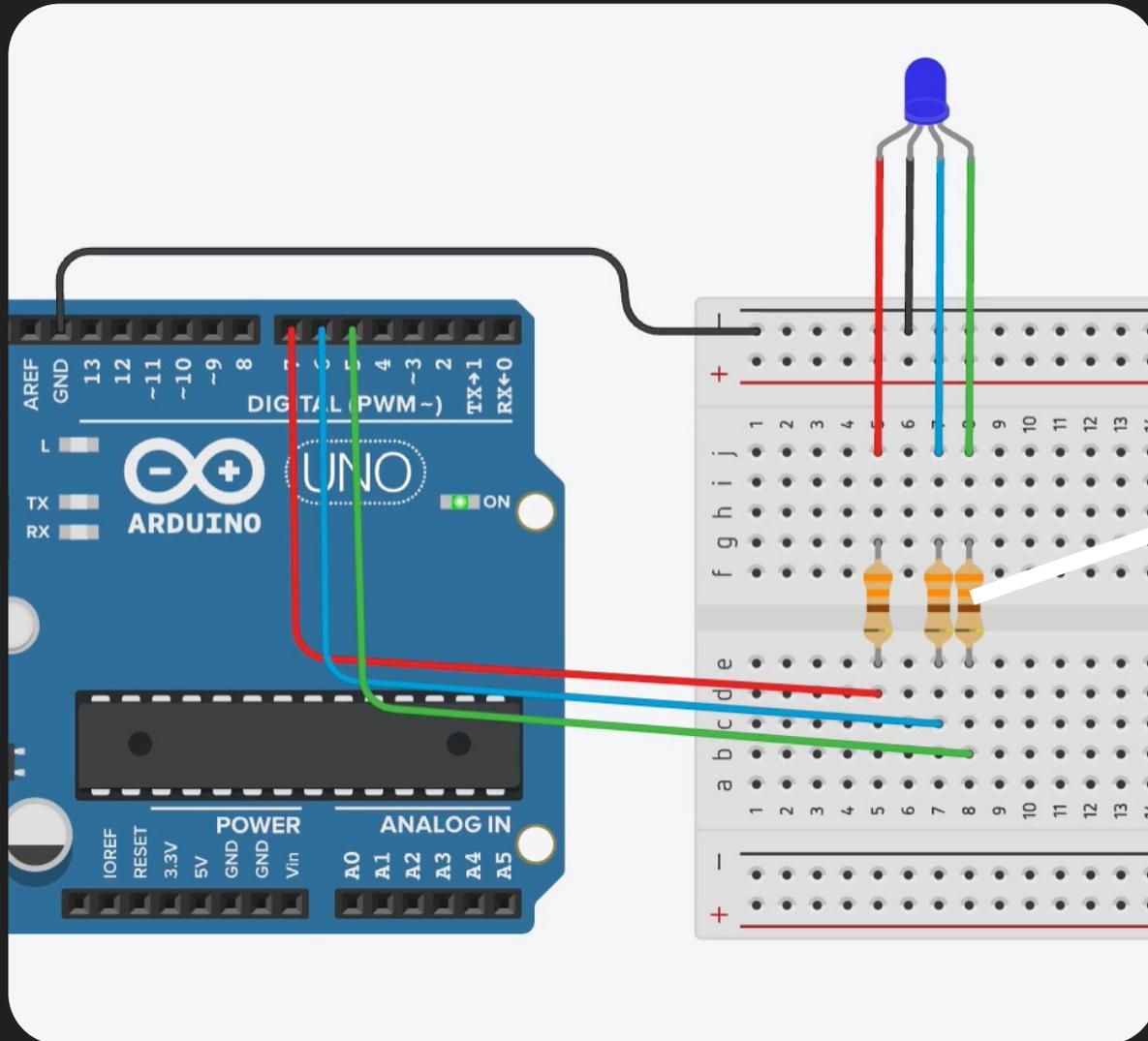
# การโปรแกรมเพื่อควบคุมหลอด LED



The image shows the Arduino IDE interface with a wiring diagram and C++ code for controlling an LED. The wiring diagram shows an Arduino Uno connected to a breadboard. The breadboard has three resistors connected to pins 7, 6, and 5. The LED is connected to pins 7, 6, and 5. The C++ code is as follows:

```
1 // C++ code
2 //
3 void setup()
4 {
5   pinMode(7, OUTPUT); //RED
6   pinMode(6, OUTPUT); //BLUE
7   pinMode(5, OUTPUT); //GREEN
8 }
9
10 void loop()
11 {
12   int i;
13   for(i=0;i<=2;i++){
14     if(i==0){
15       analogWrite(7, 255);
16       analogWrite(6, 0);
17       analogWrite(5, 0);
18     }
19     else if(i==1){
20       analogWrite(7, 0);
21       analogWrite(6, 0);
22       analogWrite(5, 255);
23     }
24     else{
25       analogWrite(7, 0);
26       analogWrite(6, 255);
27       analogWrite(5, 0);
28     }
29     delay(1000);
30   }
31 }
```

# การโปรแกรมเพื่อควบคุมหลอด LED



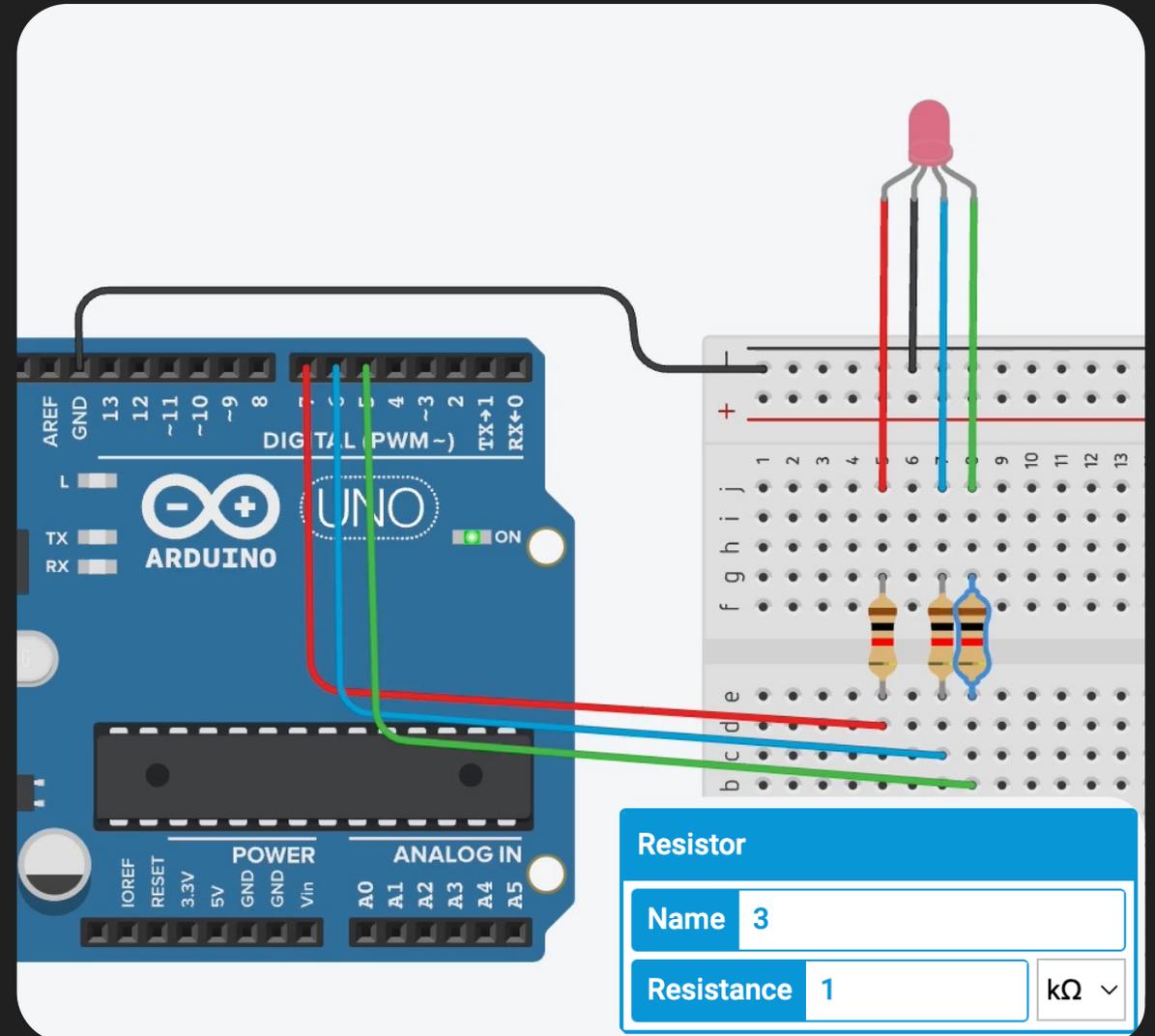
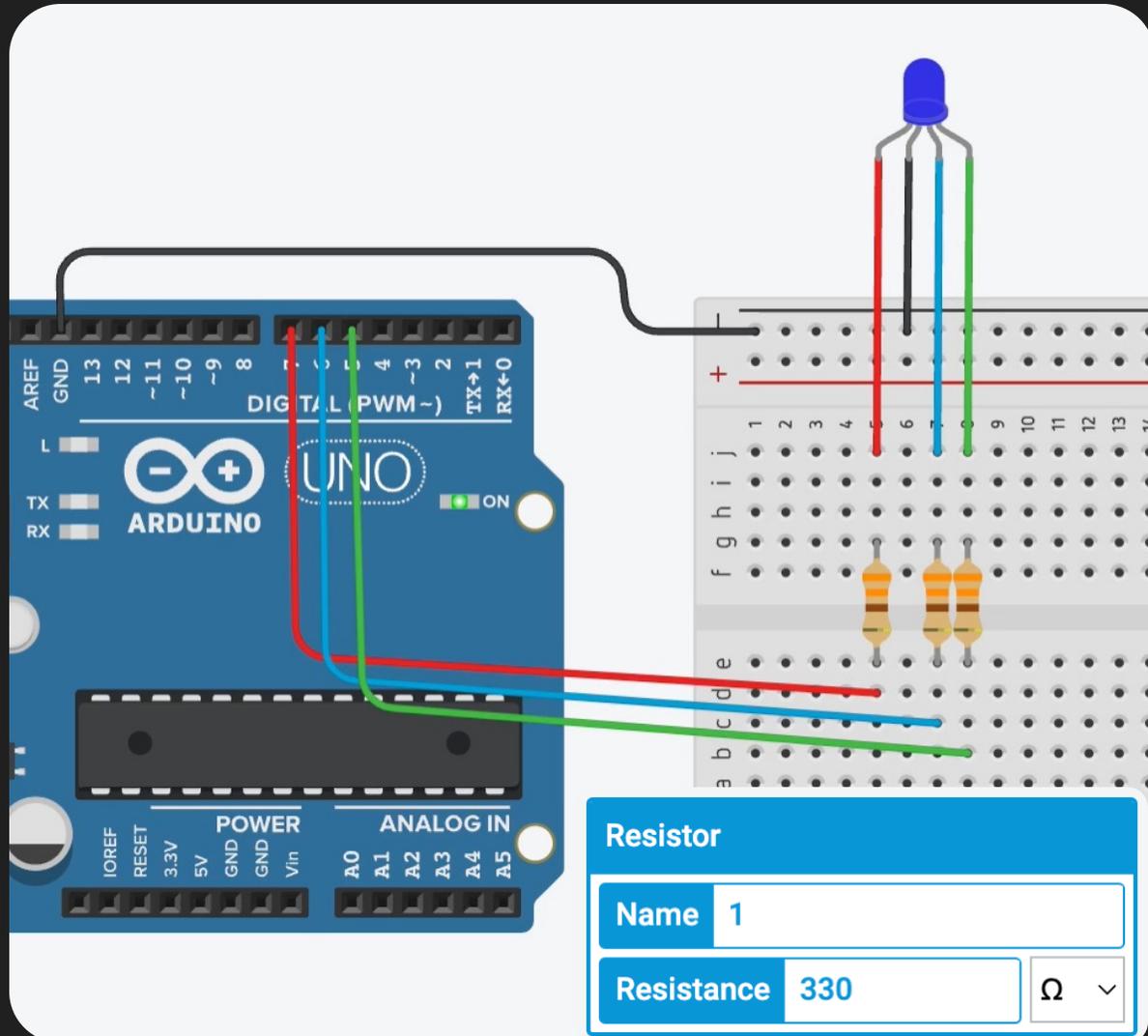
## ตัวต้านทาน

Resistor

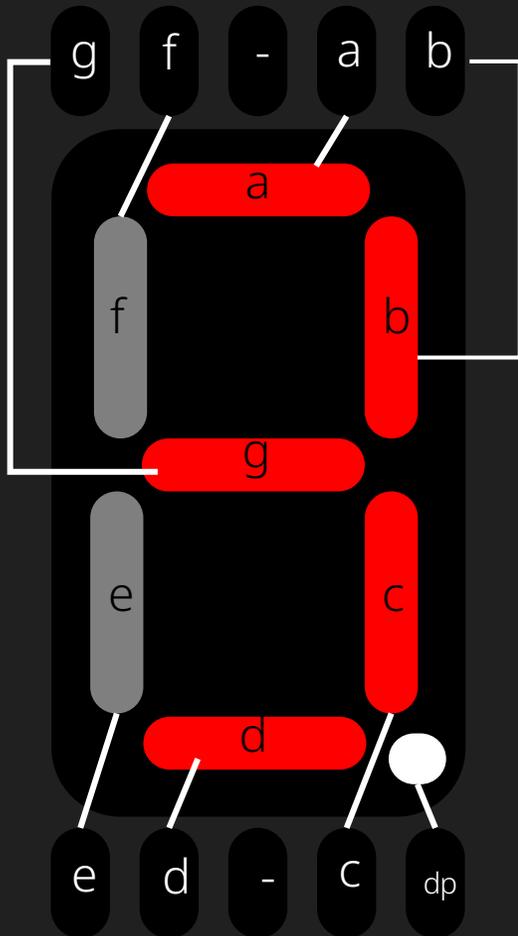
Name 1

Resistance 330  $\Omega$

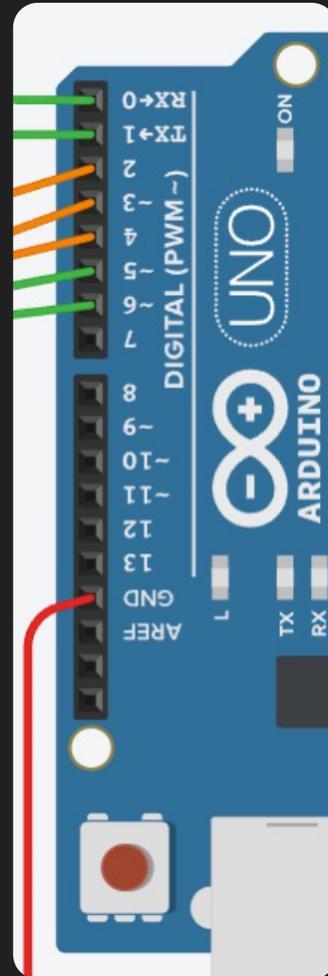
# การโปรแกรมเพื่อควบคุมหลอด LED



# Seven Segment กับ Arduino UNO



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



```

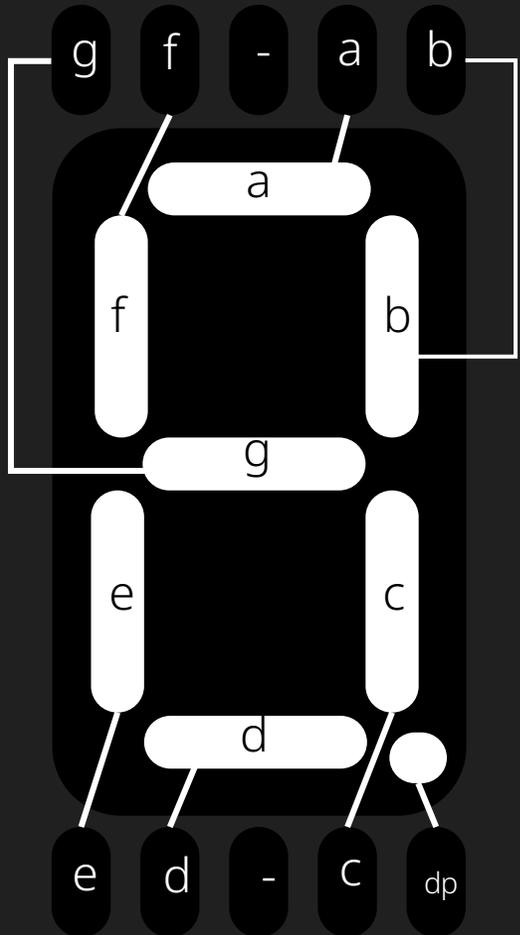
SevenSegment_Three

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



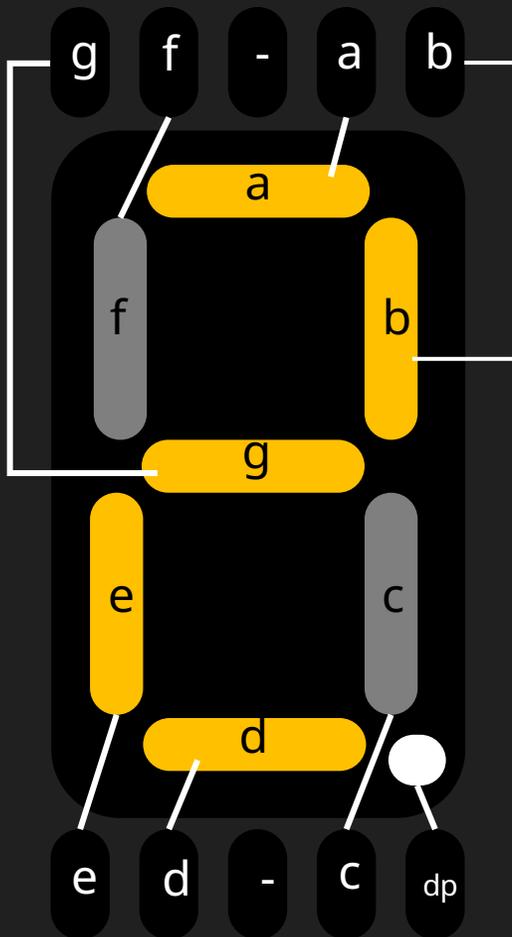
```
SevenSegment |
void setup() {
  // put your setup code here, to run once:
  for(int x=0;x<=6;x++){
    pinMode(x,OUTPUT);
  }
}

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 |
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);
}

```



SAMPLE

# Seven Segment กับ Arduino UNO

The image shows a digital workspace for an Arduino project. On the left, a breadboard circuit is simulated. It features an Arduino UNO connected to a 7-segment display. The display is configured as a common cathode and is currently showing the digit '8'. The circuit includes several resistors connected to the display segments. On the right, a code editor window titled 'SevenSegment I' contains the following C++ code:

```
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);  
}
```