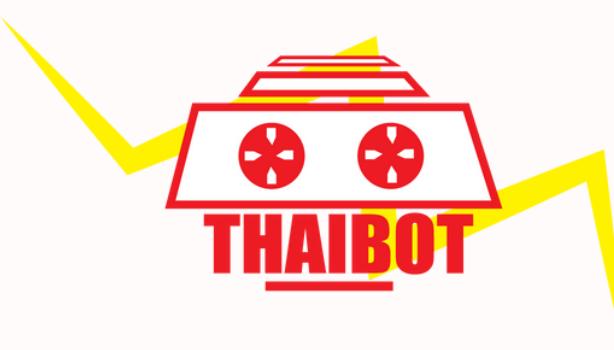
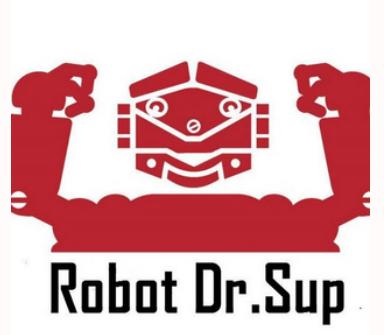
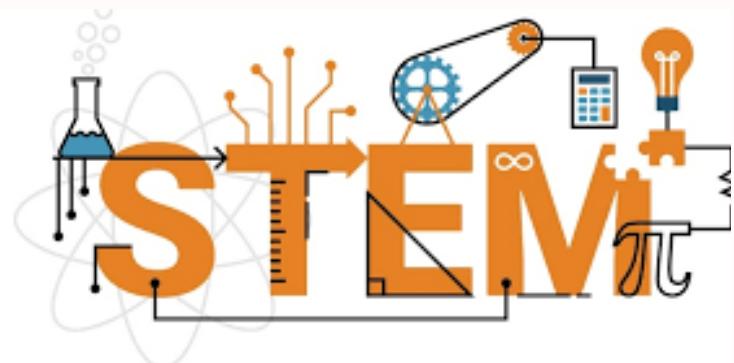




ARDUINO & ROBOTICS





TOPIC

Path 1 : What is programming ?

Path 2 : Arduino คืออะไร ทำไมต้อง Arduino ?

Path 3 : Programing Fundamental 1 กับอุปกรณ์ Output

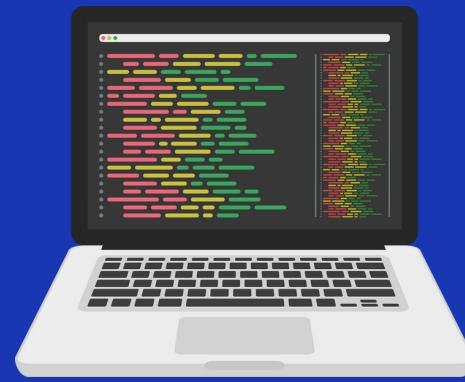
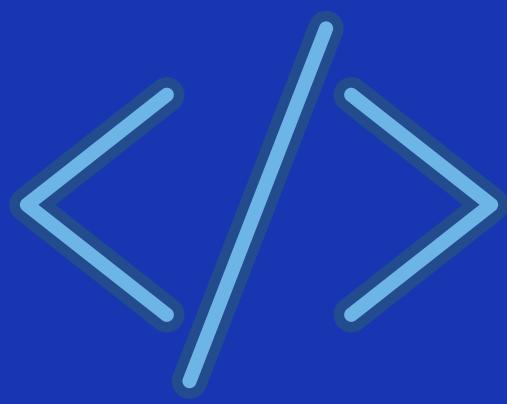
Path 4 : Programing Fundamental 2 กับอุปกรณ์ Input

Path 5 : ทฤษฎีของ Robot Line Follower

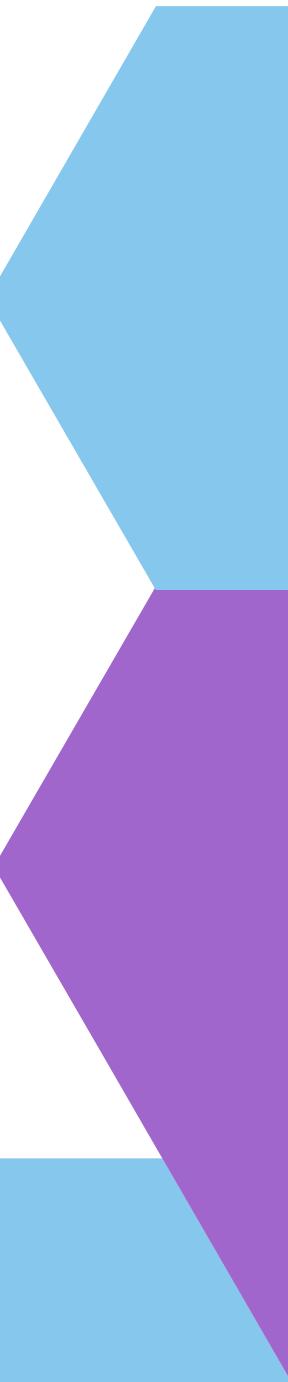
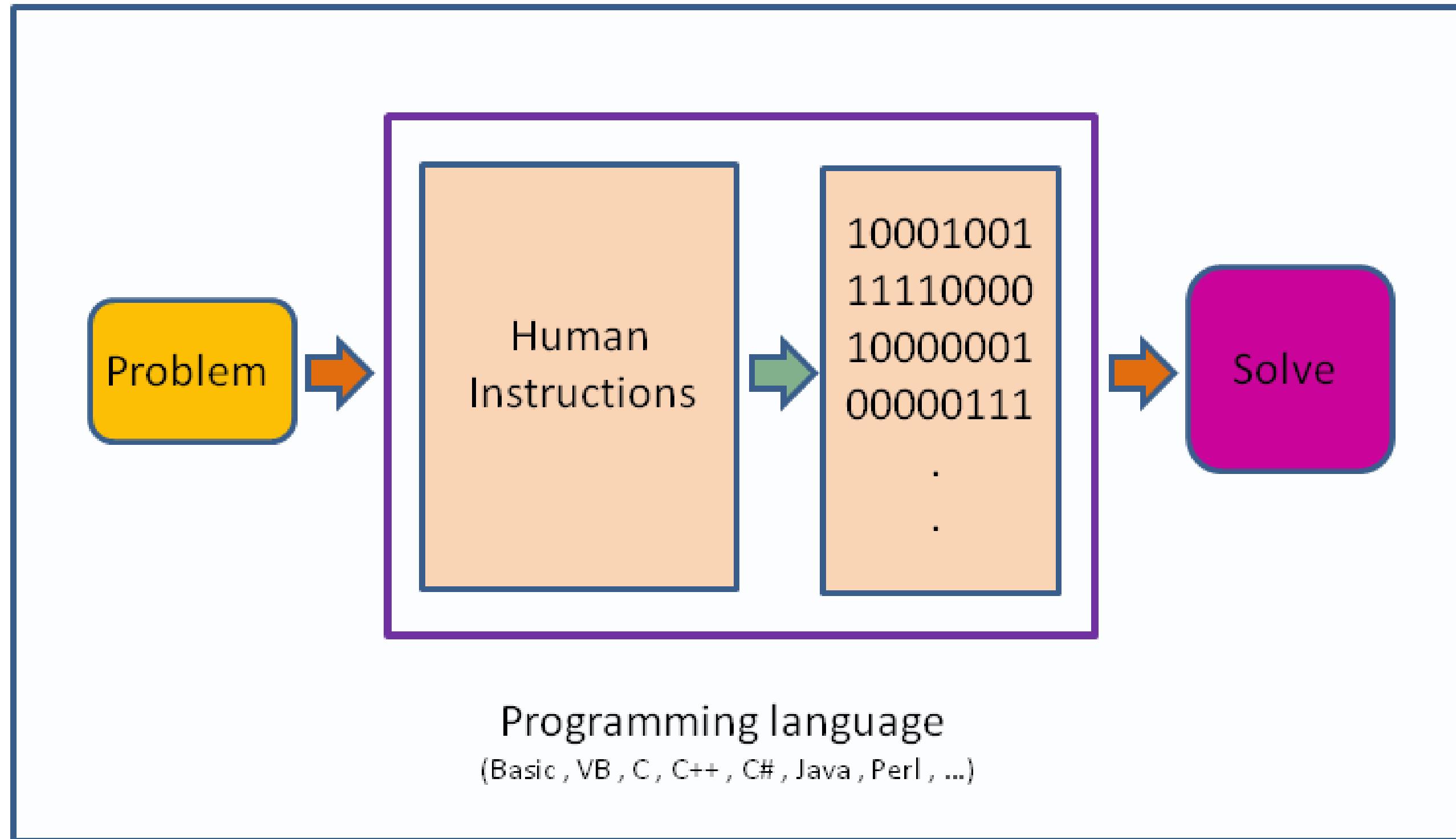
Path 6 : ทดลองโปรแกรม / การประลองโปรแกรม

Path 1

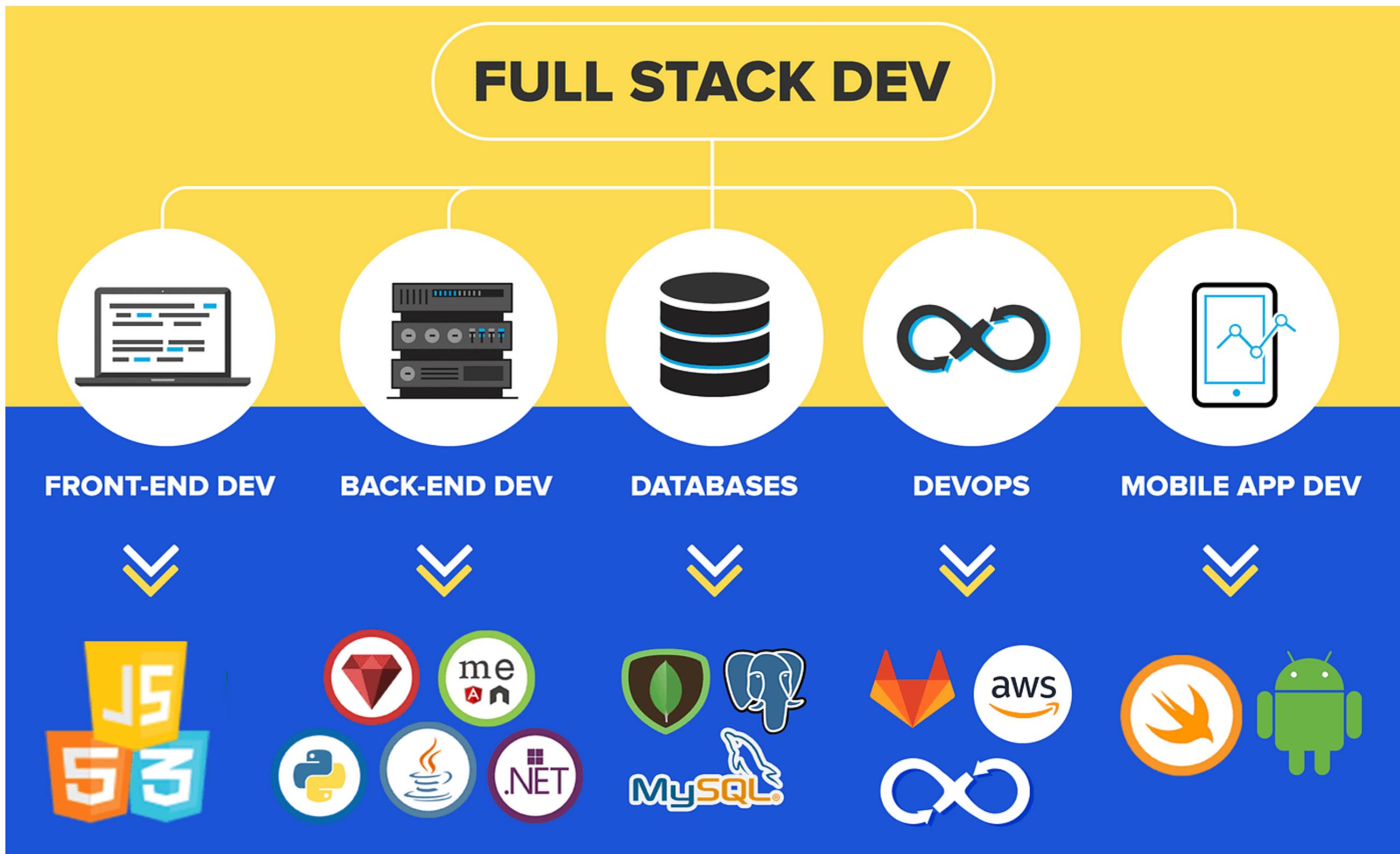
What is programming ?



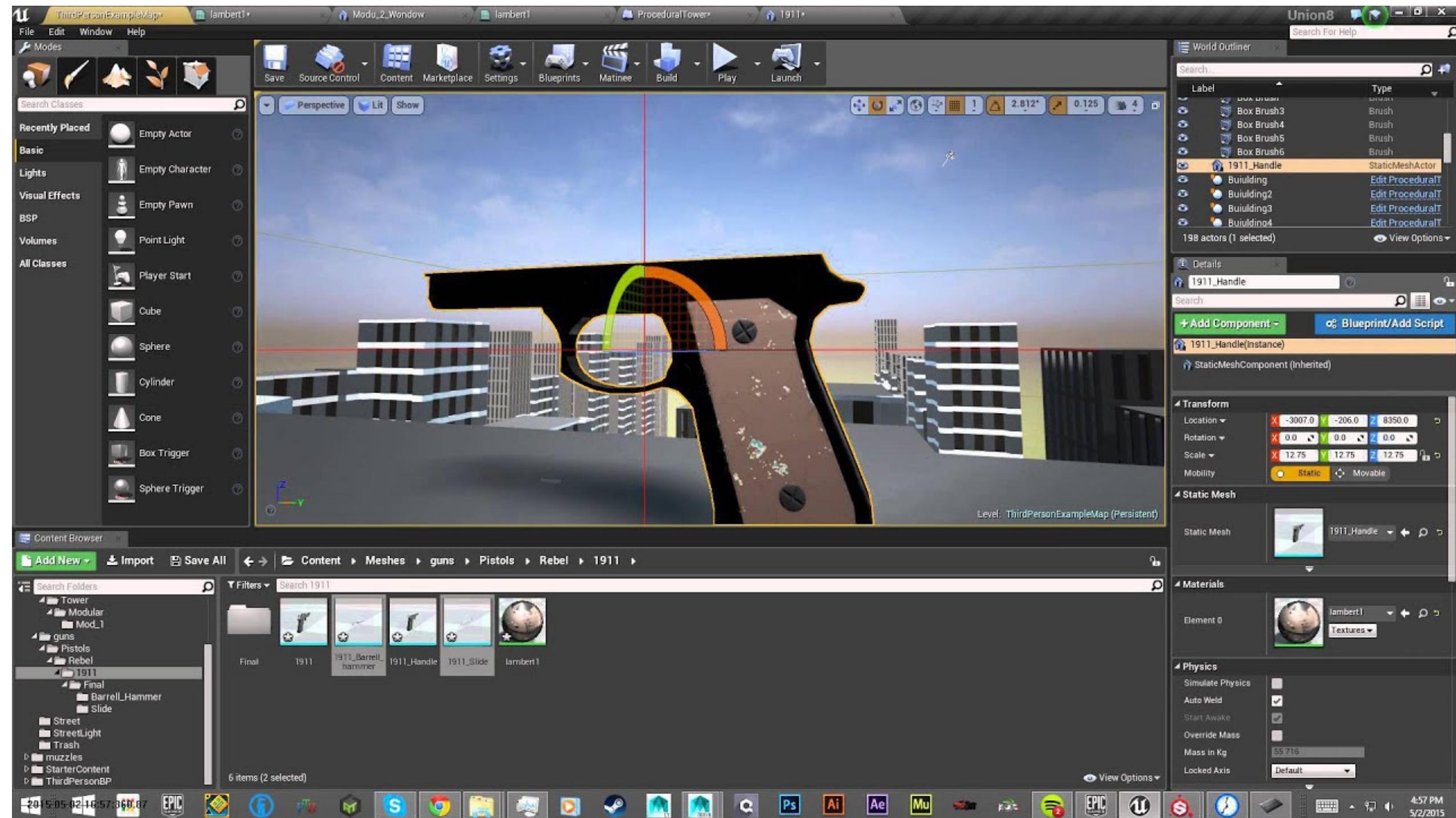
Programming in present



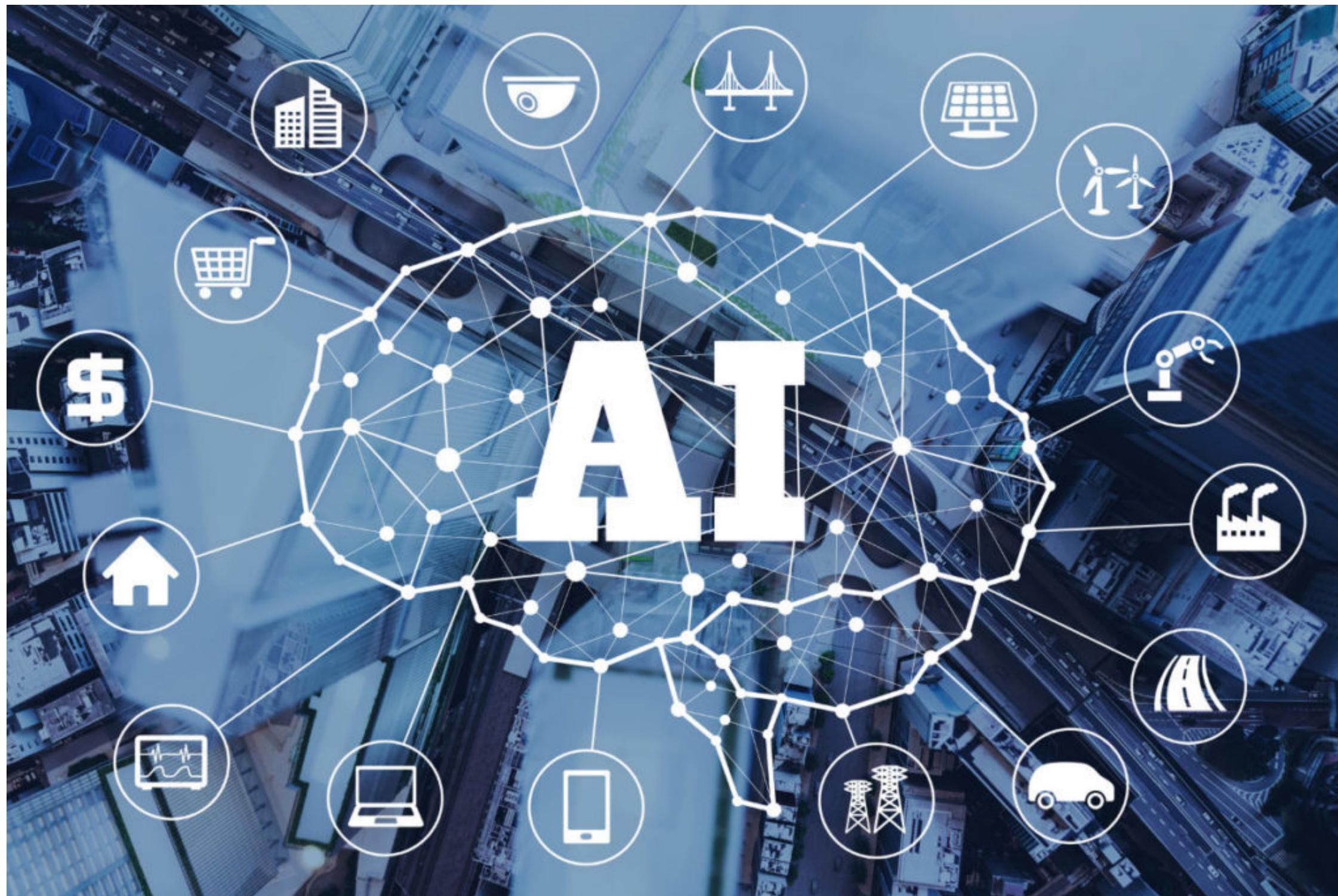
Web developer



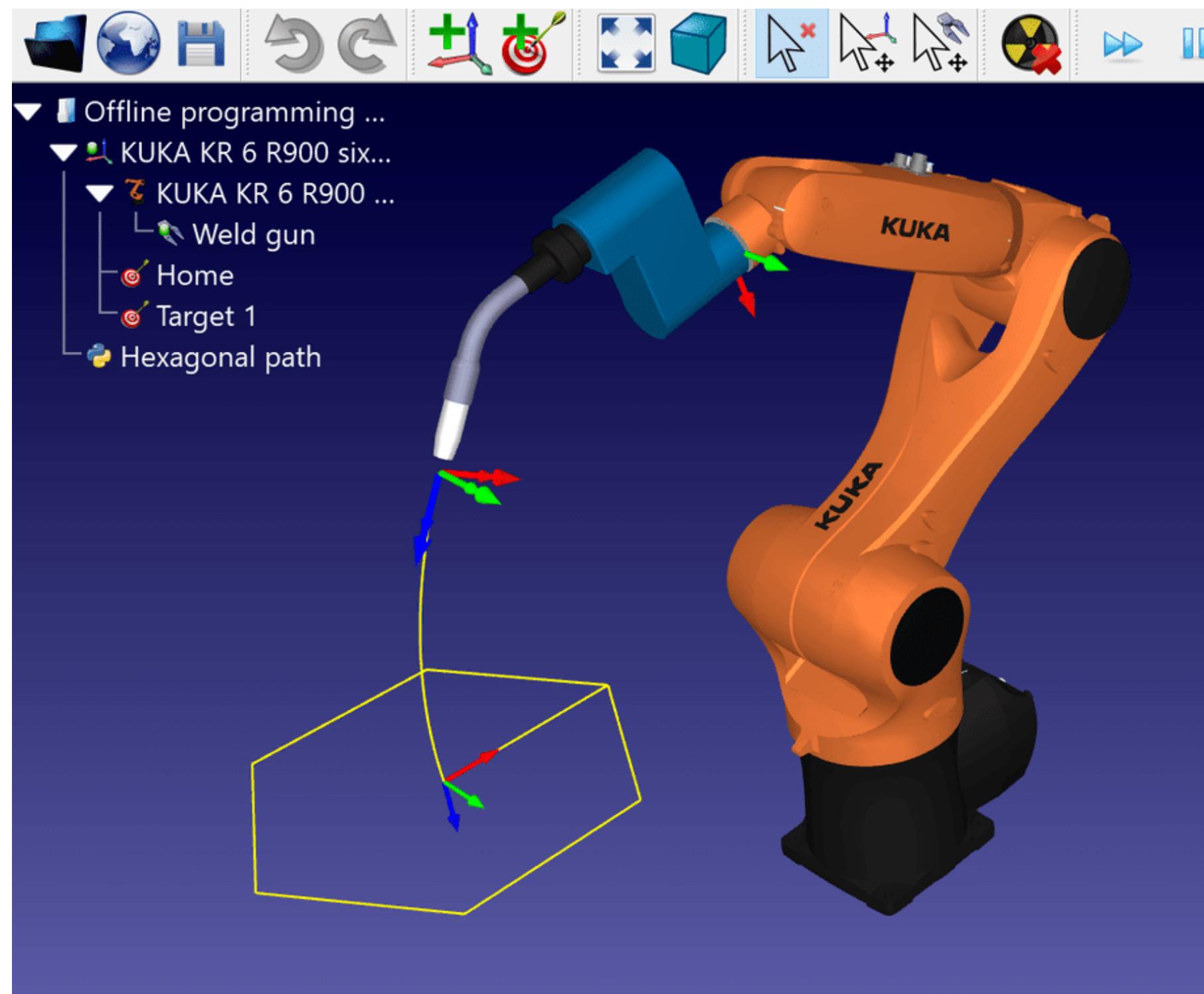
Game developer



AI Engineer



Robotics



```
Hexagonal_path.py - C:\Users\Albert\AppData\Local\Temp\Hexagonal_path.py (3.4.3)
File Edit Format Run Options Window Help
from robolink import *      # API to communicate with RoboDK
from robodk import *        # robodk robotics toolbox

# Any interaction with RoboDK must be done through RDK:
RDK = Robolink()

# get the robot by name:
robot = RDK.Item('', ITEM_TYPE_ROBOT)

# get the home target and the welding targets:
home = RDK.Item('Home')
target = RDK.Item('Target 1')

# get the pose of the target (4x4 matrix representing position)
poseref = target.Pose()

# move the robot to home, then to the Target 1:
robot.MoveJ(home)
robot.MoveJ(target)

# make an hexagon around the Target 1:
for i in range(7):
    ang = i*2*pi/6 #angle: 0, 60, 120, ...
    posei = poseref*rotz(ang)*transl(200,0,0)*rotz(-ang)
    robot.MoveL(posei)

# move back to the center, then home:
robot.MoveL(target)
robot.MoveJ(home)
```

Robotics / AI in real life

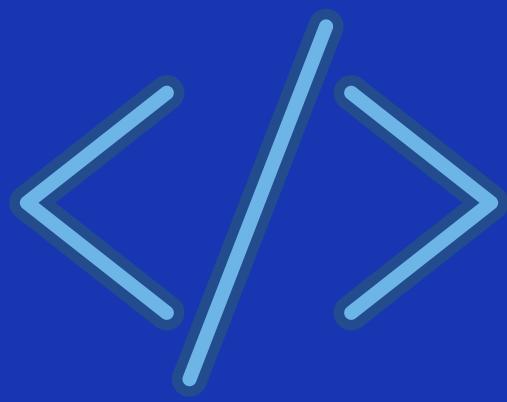


ANDROID AUTHORITY

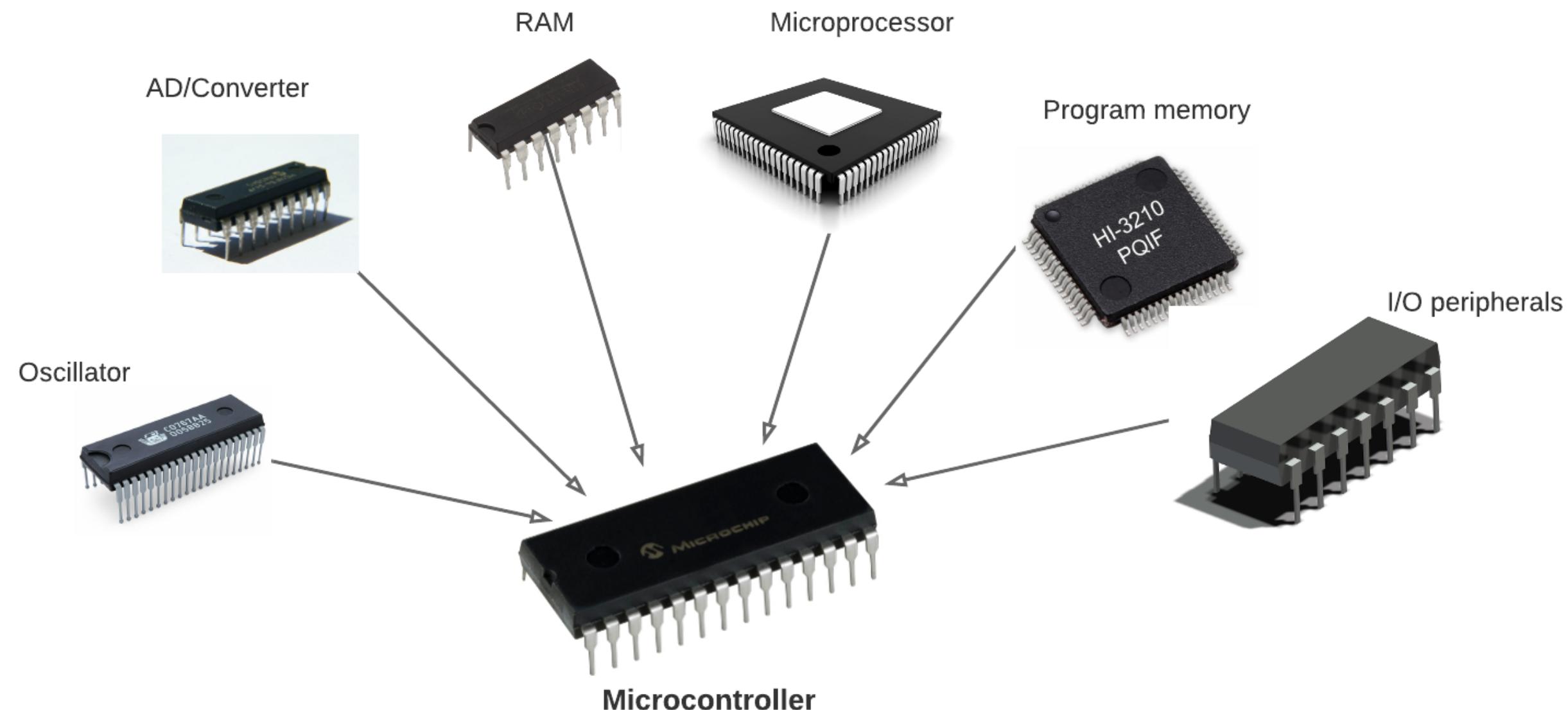


Path 2

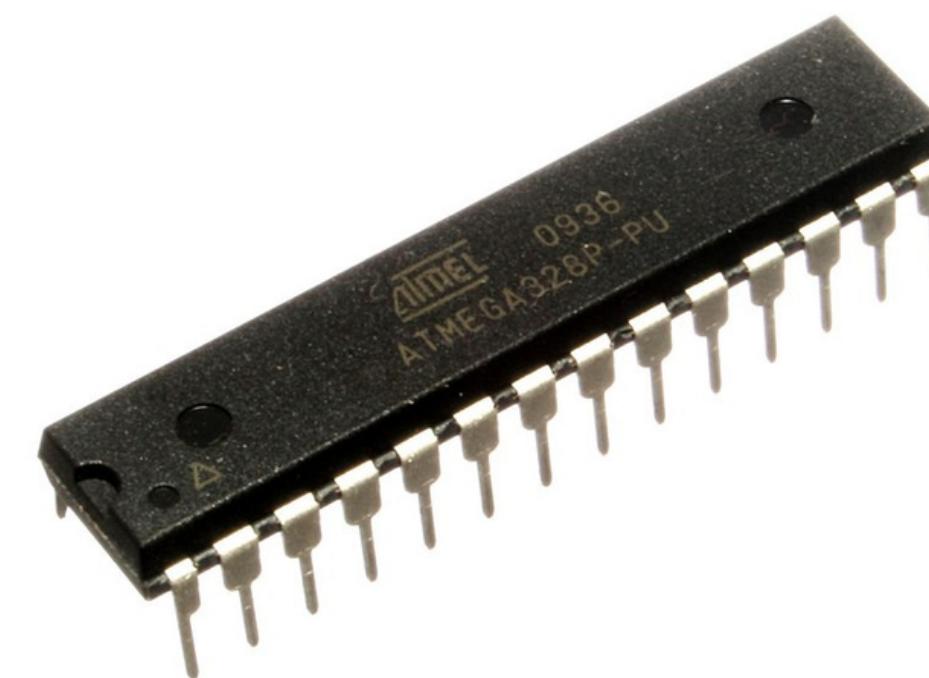
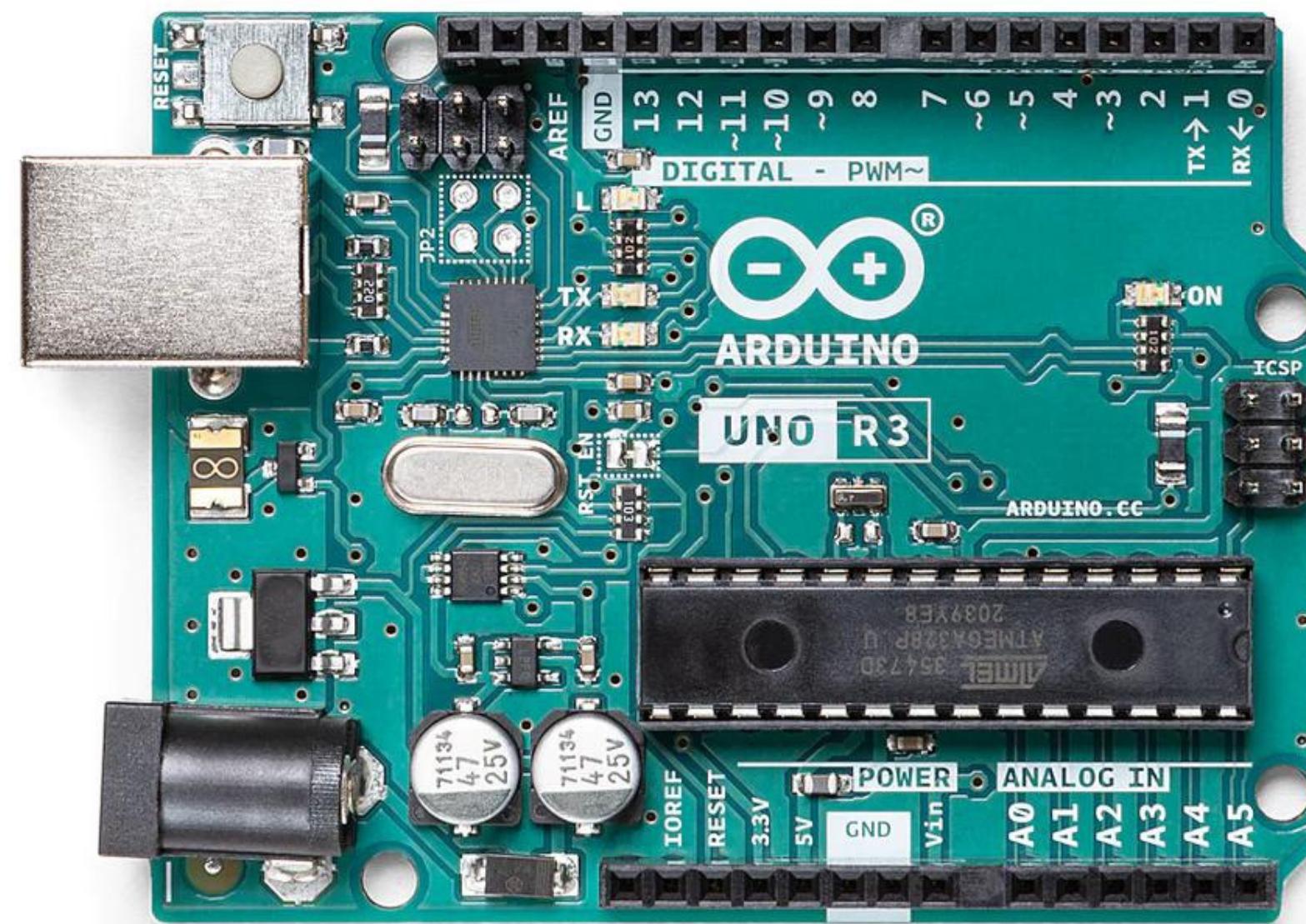
What is Arduino ?



Arduino & Microcontroller ?

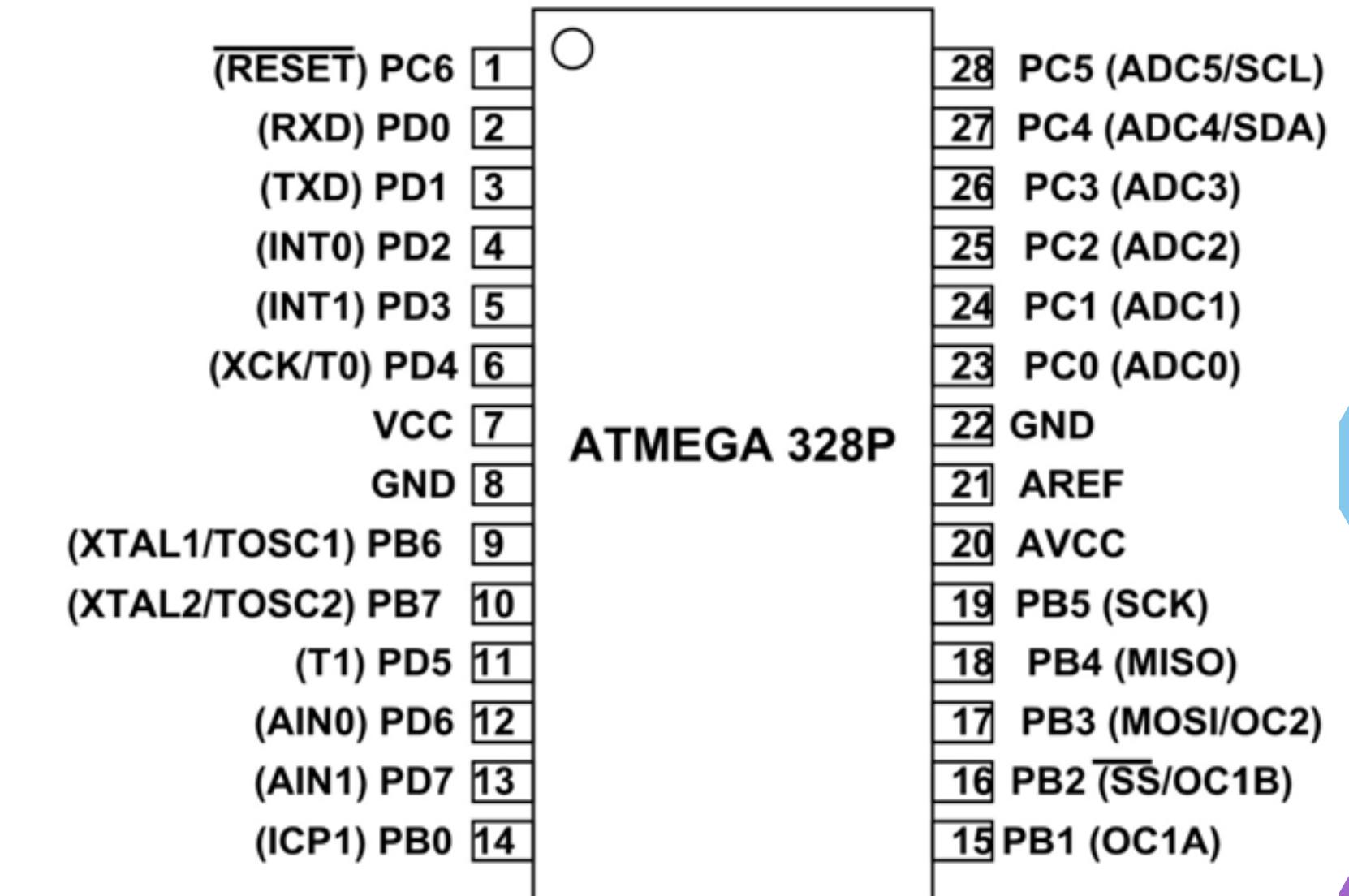


Arduino Uno ?

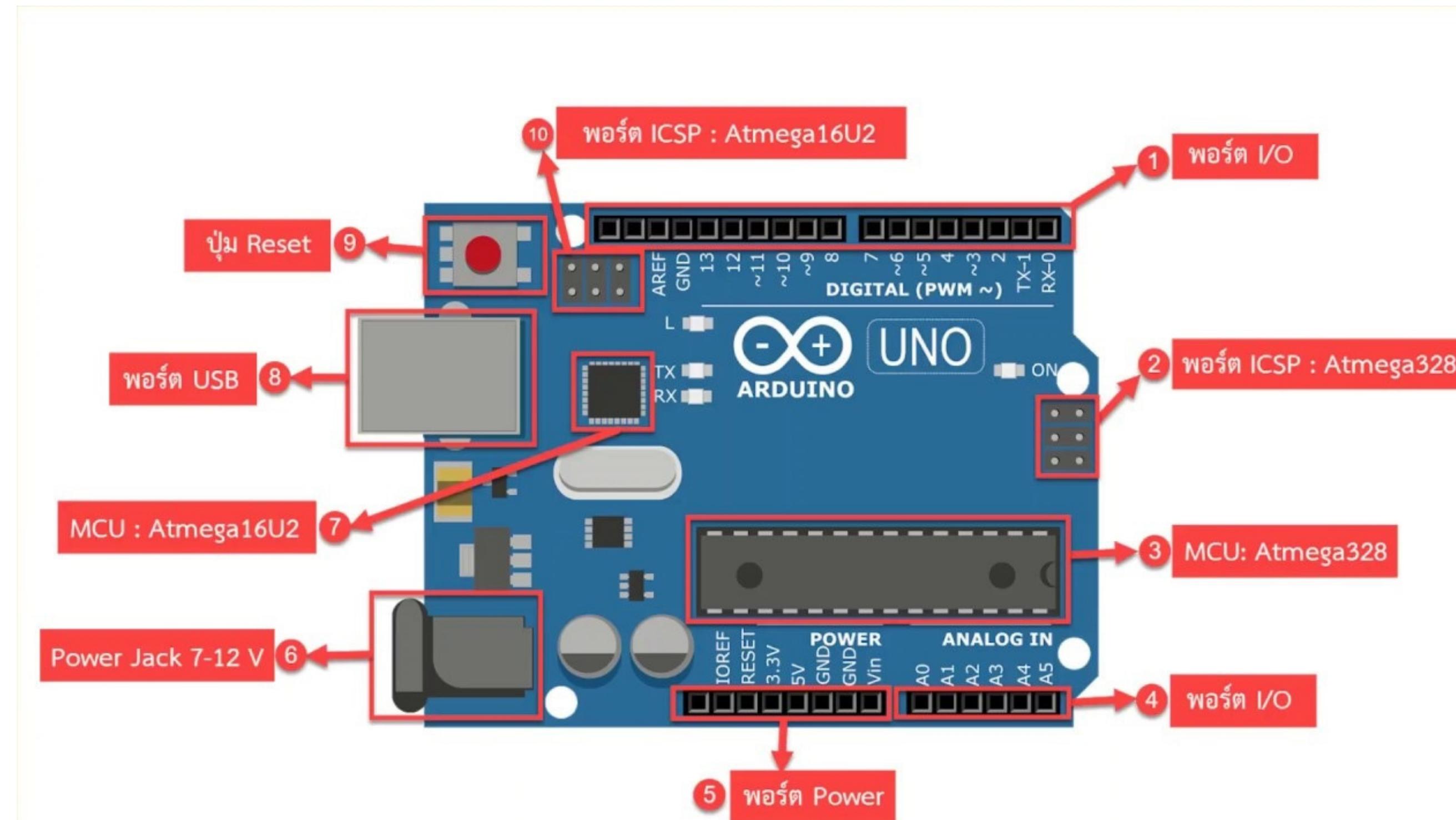


Arduino Uno ?

Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
Length	68.6 mm
Width	53.4 mm
Weight	25 g



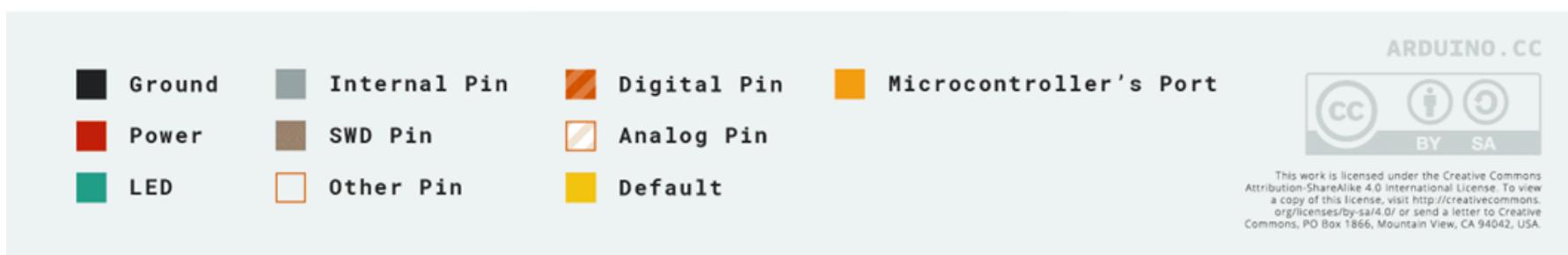
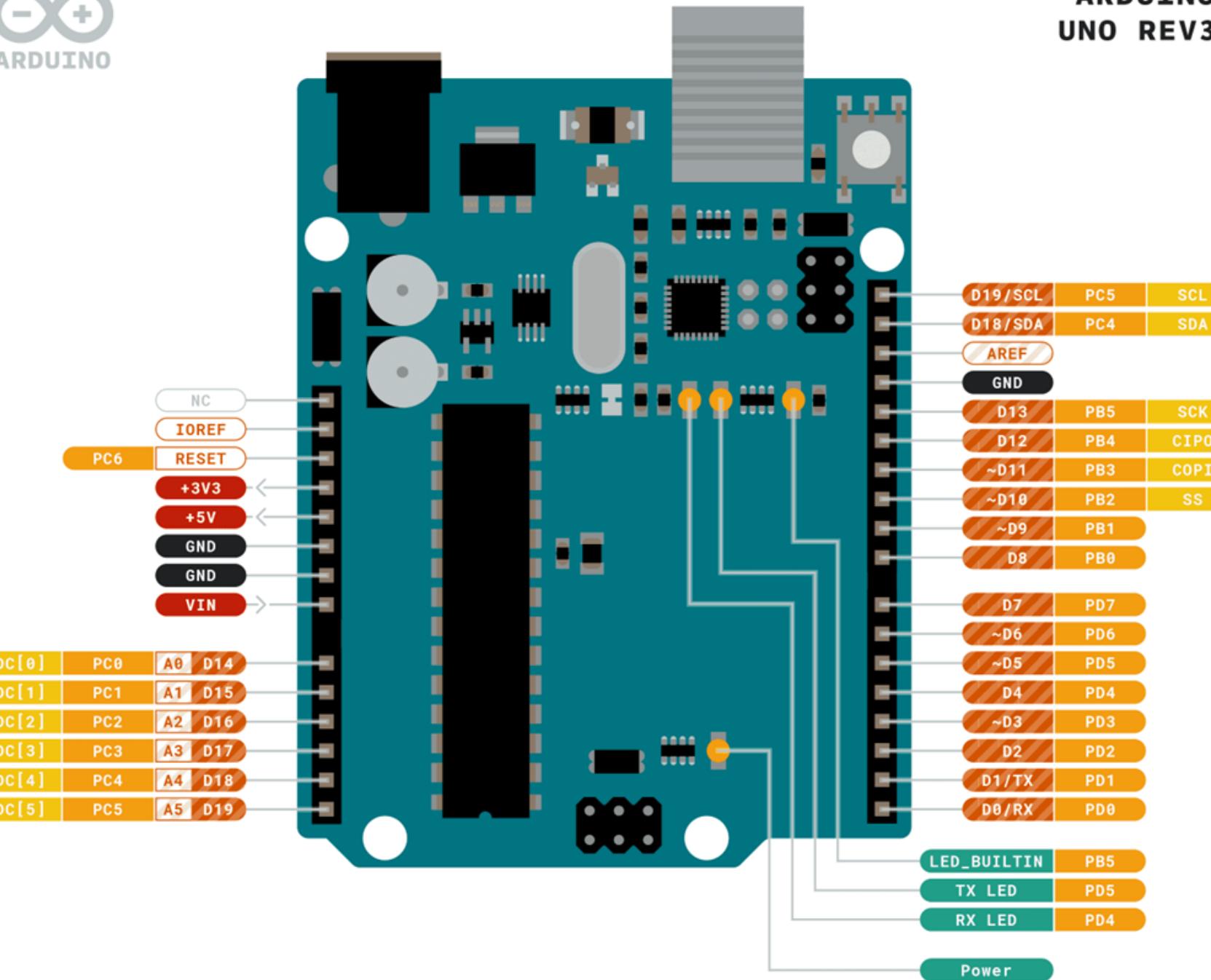
Arduino Uno I/O



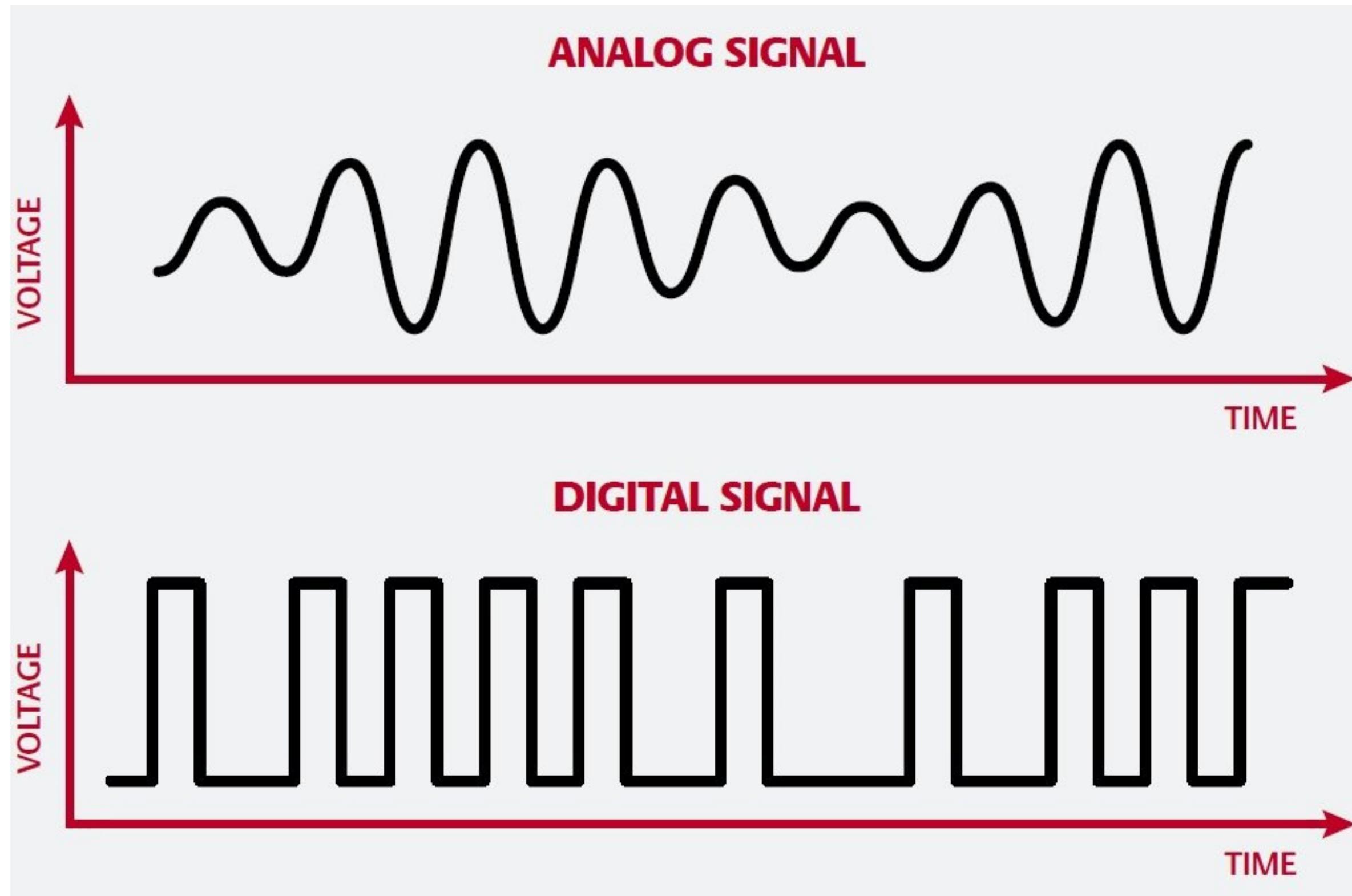
Arduino Uno I/O



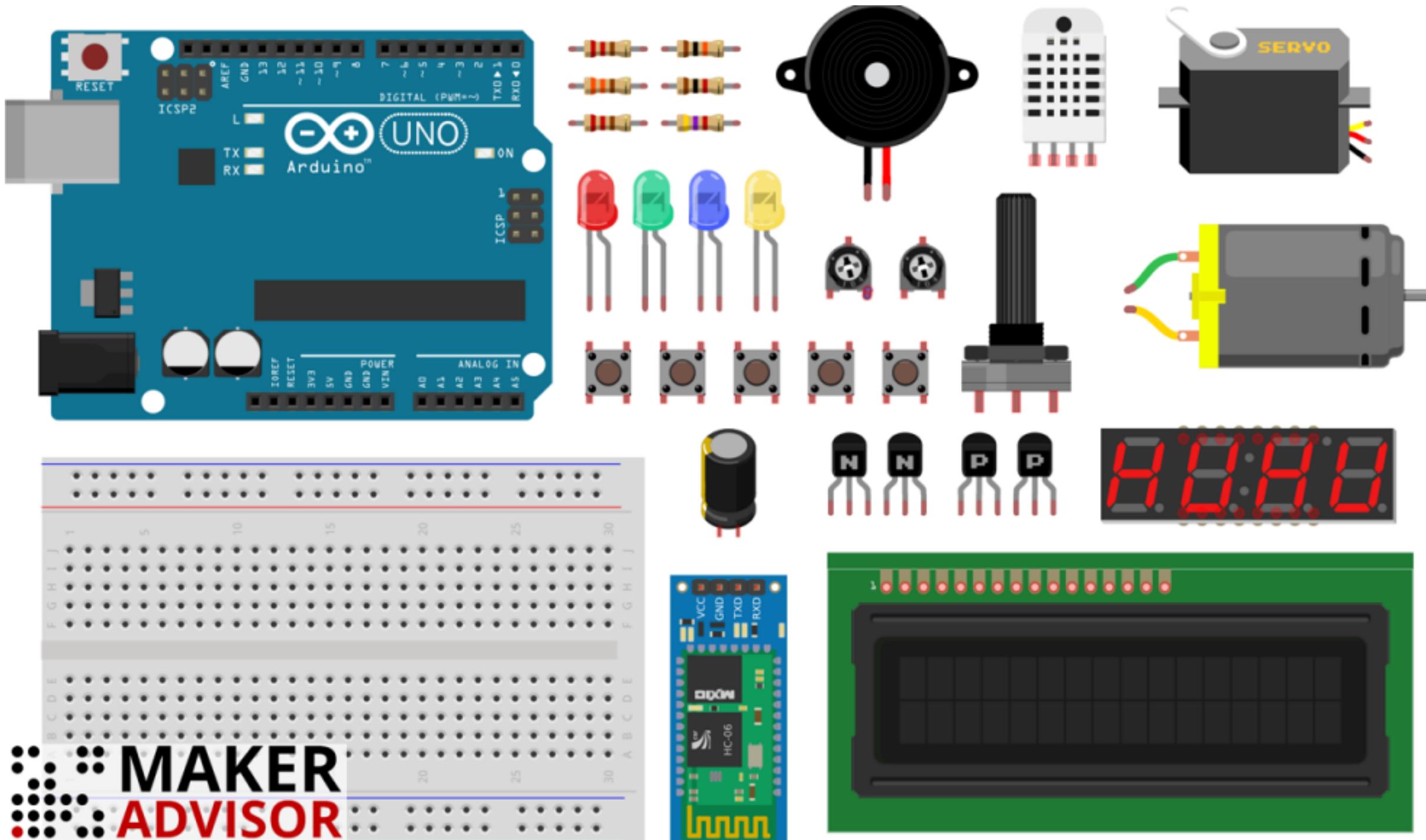
ARDUINO
UNO REV3



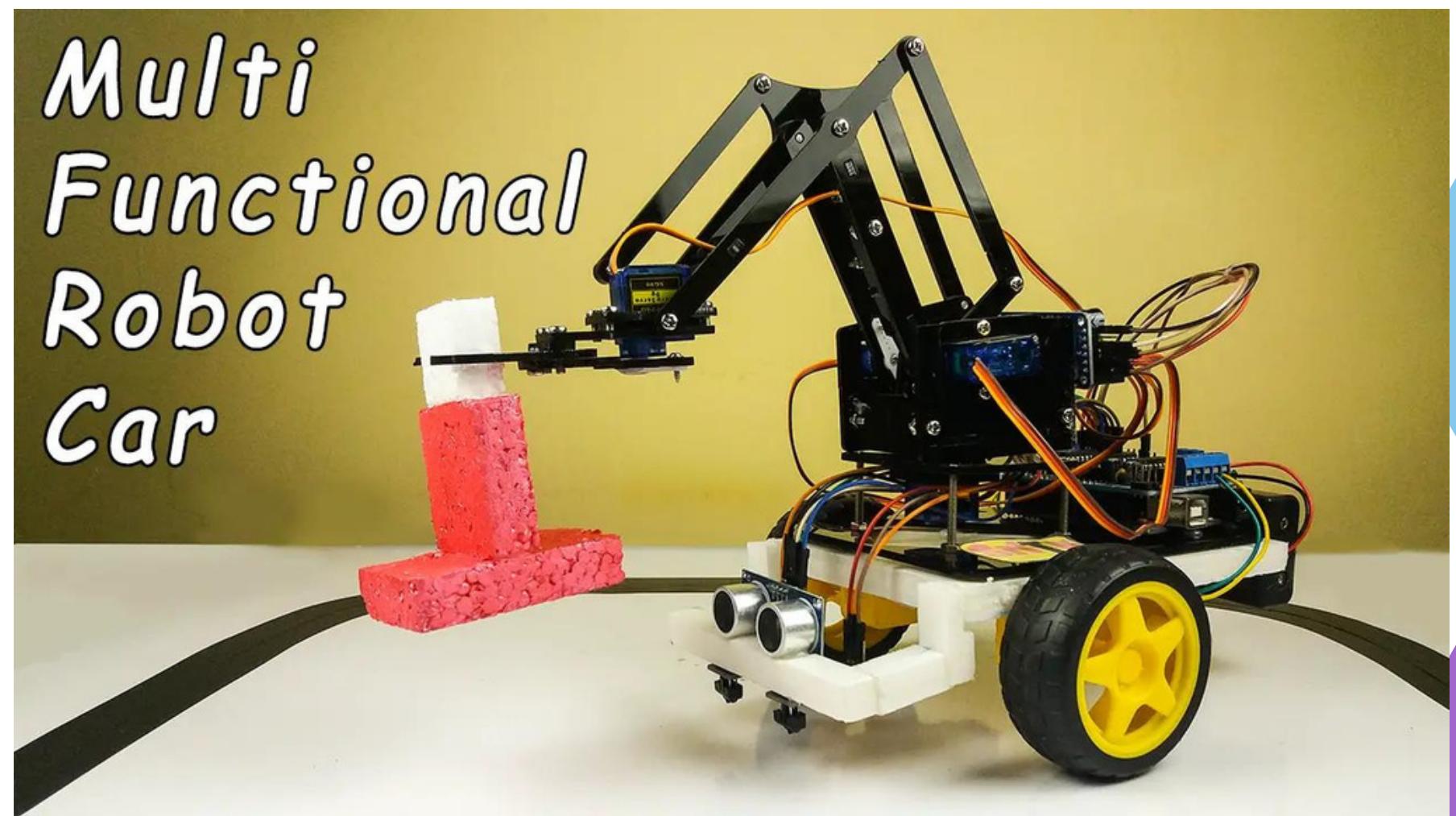
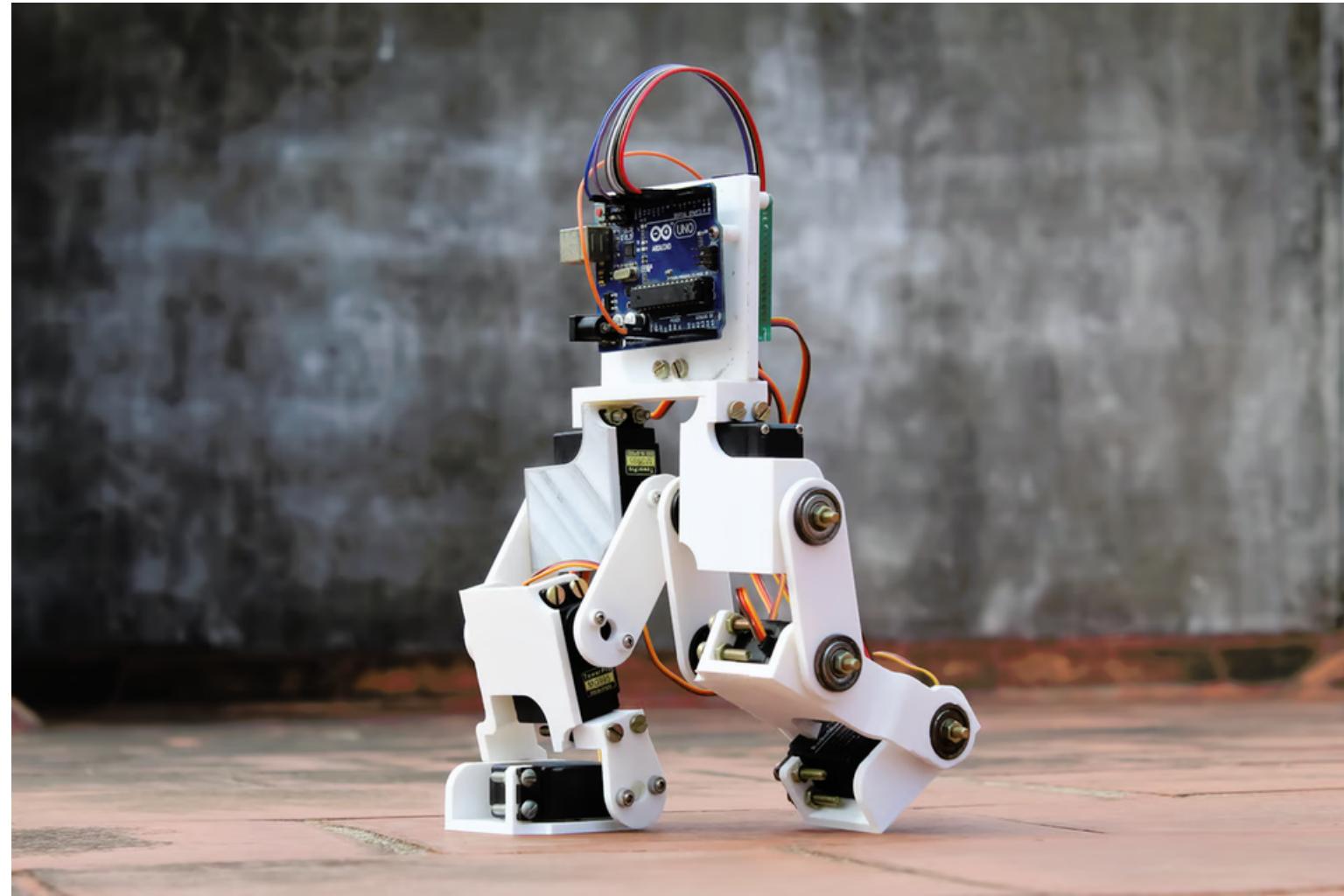
What is Signal ?



What Arduino can do ?



What Arduino can do ?

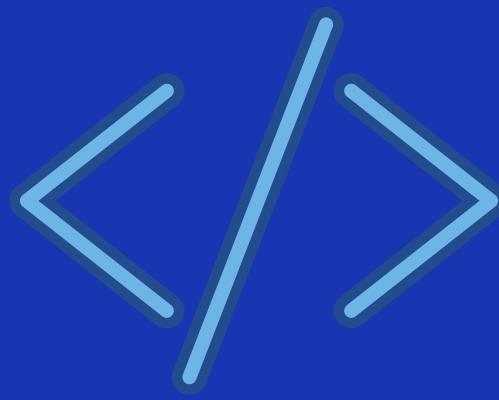


What Arduino can do ?

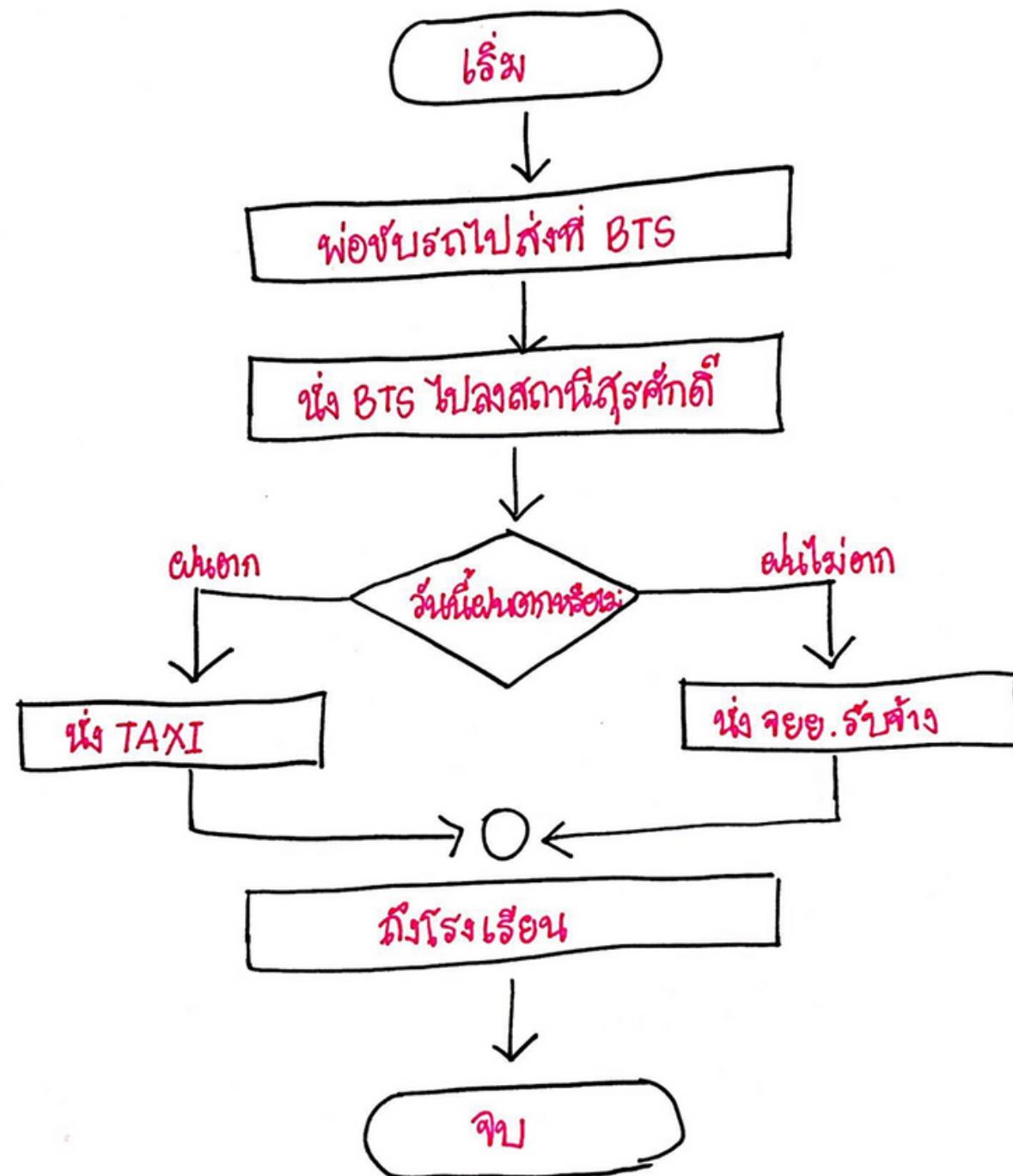


Path 3

Programming Fundamental 1



Flowchart



ขั้นตอนการพัฒนาโปรแกรม

เขียนลำดับขั้น (Algorithm)



แปลงลำดับขั้นเป็นผังงาน (Flowchart)



แปลงผังงานเป็นโปรแกรม

Arduino Code Structure

sketch_oct19a | Arduino 1.8.7

File Edit Sketch Tools Help



```
sketch_oct19a §

1 void setup() {
2     // put your setup code here, to run once:
3
4 }
5
6 void loop() {
7     // put your main code here, to run repeatedly:
8
9 }
```

Arduino Code Structure

ตัวโปรแกรม

ฟังก์ชันที่ทำงานครั้งเดียว `setup()`

ฟังก์ชันที่เวียนกลับทำงานวนซ้ำ `loop()`

(พื้นฐาน)

หัวโปรแกรม

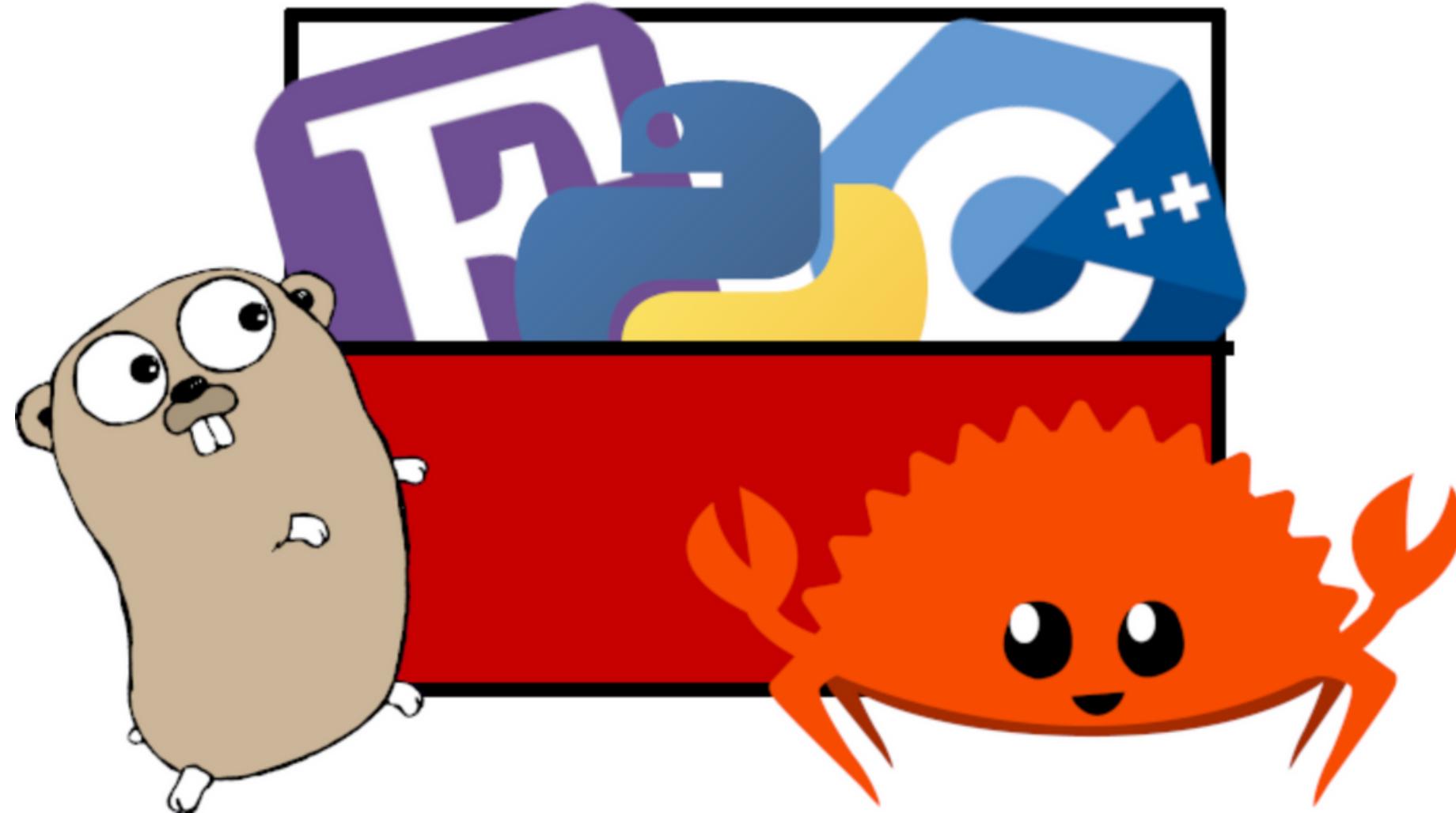
ตัวโปรแกรม

ฟังก์ชันที่ทำงานครั้งเดียว `setup()`

ฟังก์ชันที่เวียนกลับทำงานวนซ้ำ `loop()`

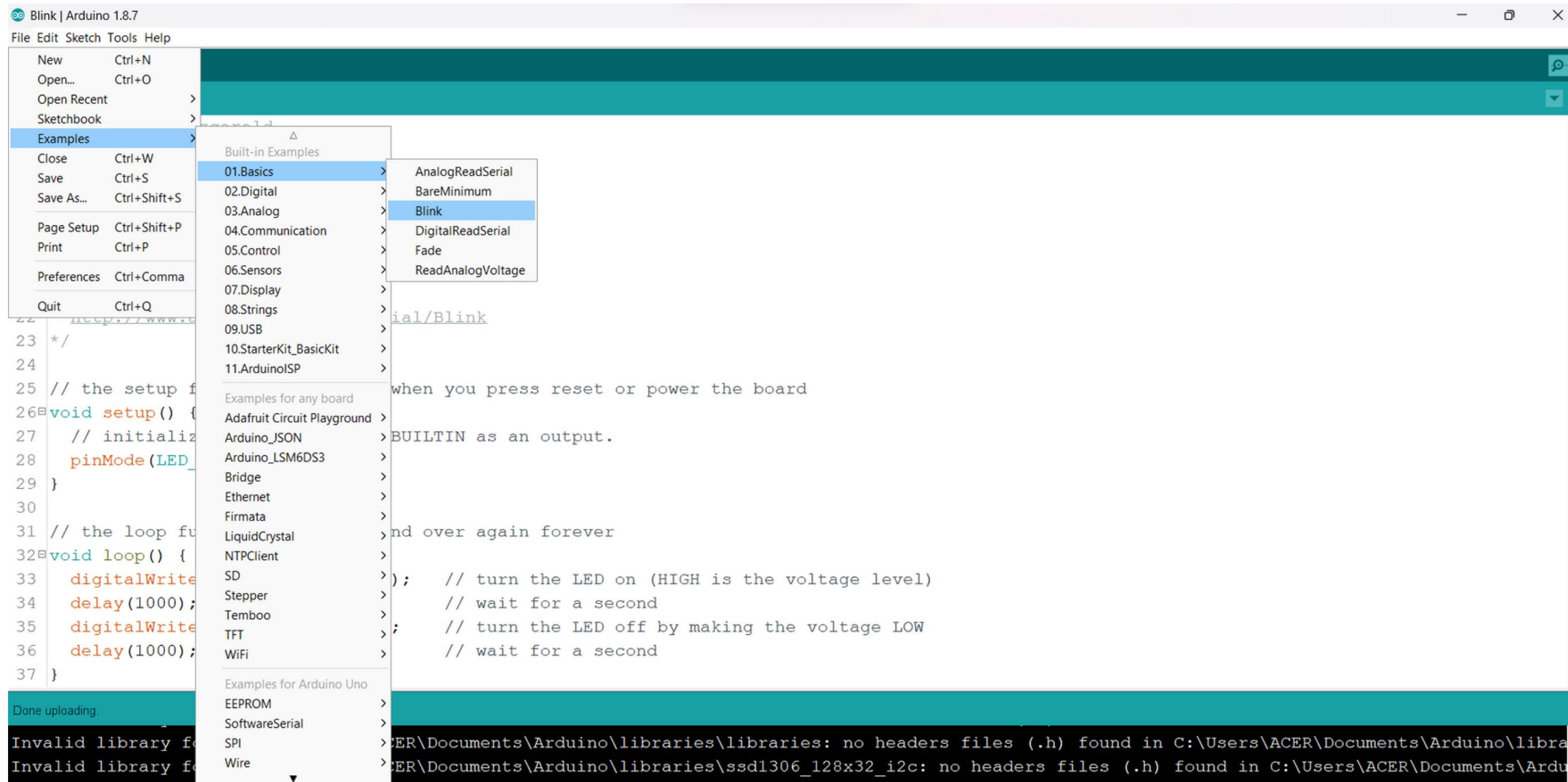
ฟังก์ชันรองที่เขียนขึ้นเอง

(ขั้นสูง)



TRY Example Blynk

Compiler & Upload Code



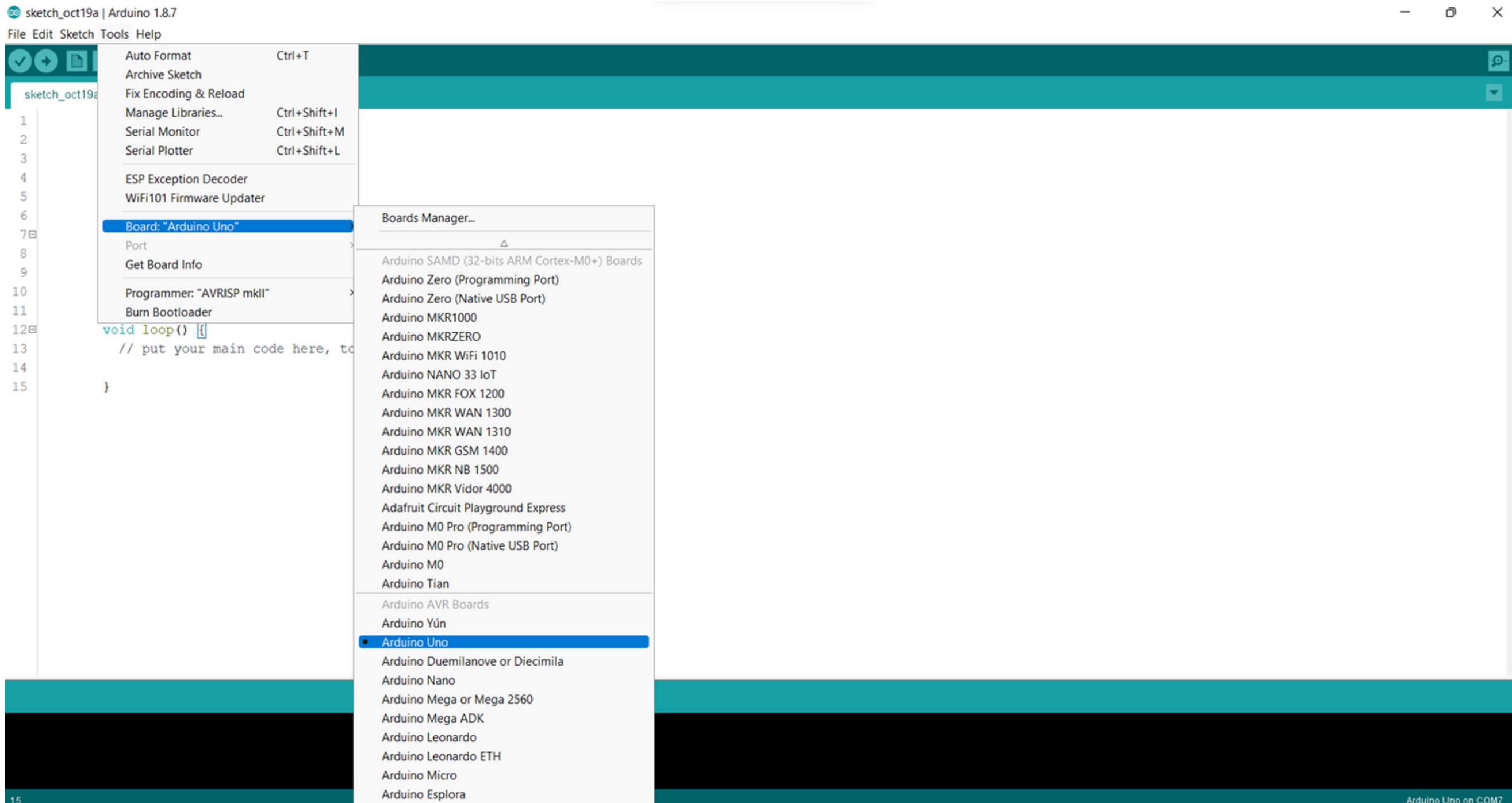
The screenshot shows the Arduino IDE interface with the title bar "Blink | Arduino 1.8.7". The menu bar includes File, Edit, Sketch, Tools, and Help. The "Sketch" menu is open, showing the "Examples" option selected. A submenu titled "01.Basics" is displayed, containing "AnalogReadSerial", "BareMinimum", "Blink" (which is highlighted in blue), "DigitalReadSerial", "Fade", and "ReadAnalogVoltage". The main code editor area contains the "Blink" sketch, which blinks an LED connected to pin 13. The code is as follows:

```
23 */
24
25 // the setup function runs once when you press reset or power the board
26 void setup() {
27   // initialize the LED pin as an output
28   pinMode(LED_BUILTIN, OUTPUT);
29 }
30
31 // the loop function runs over and over again forever
32 void loop() {
33   digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
34   delay(1000); // wait for a second
35   digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
36   delay(1000); // wait for a second
37 }
```

The status bar at the bottom shows "Done uploading." and two error messages: "Invalid library file" and "Invalid library file".

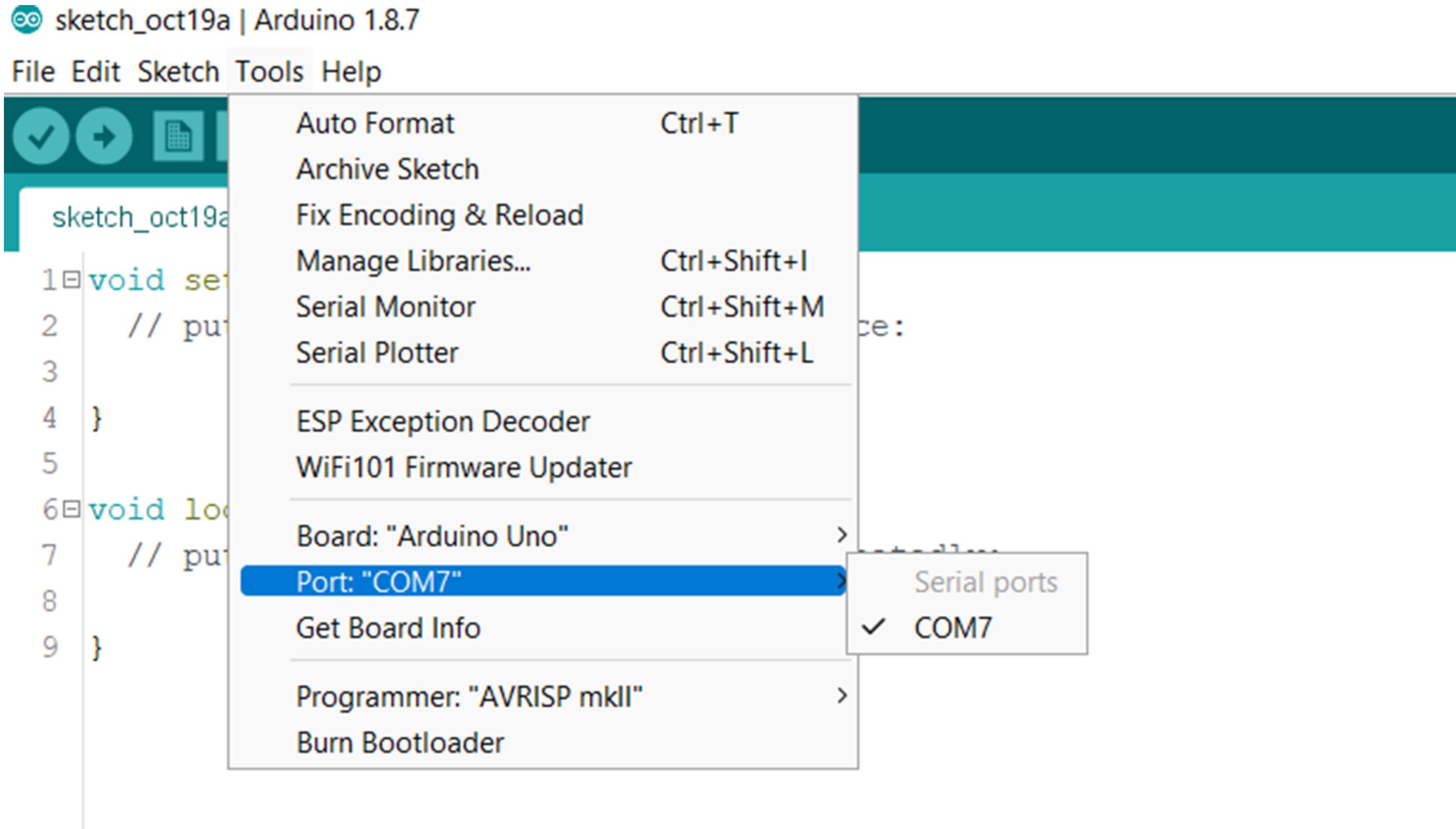
Ľaľwá

Compiler & Upload Code



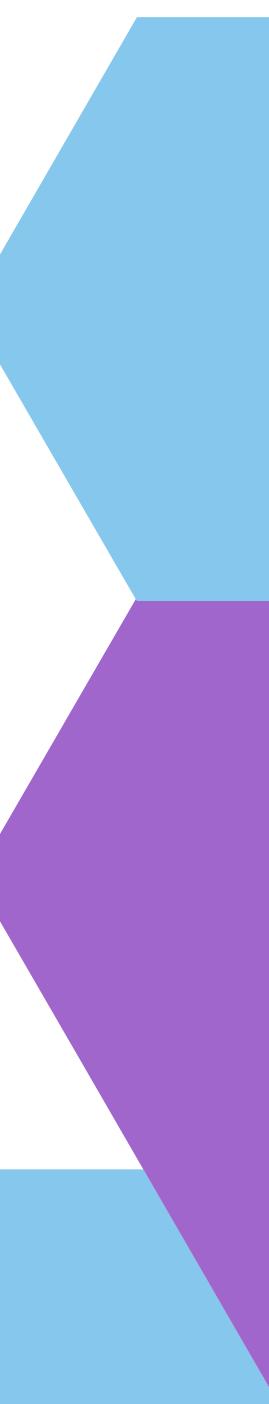
เลือกบอร์ด Arduino Uno

Compiler & Upload Code



เลือก Port USB

Compiler & Upload Code



Screenshot of the Arduino IDE showing the "Blink" example sketch. The code is displayed in the main editor area, and the status bar at the bottom shows the upload command and file path.

```
14 // by Scott Fitzgerald
15 // modified 2 Sep 2016
16 // by Arturo Guadalupi
17 // modified 8 Sep 2016
18 // by Colby Newman
19
20 This example code is in the public domain.
21
22 http://www.arduino.cc/en/Tutorial/Blink
23 */
24
25 // the setup function runs once when you press reset or power the board
26 void setup() {
27     // initialize digital pin LED_BUILTIN as an output.
28     pinMode(LED_BUILTIN, OUTPUT);
29 }
30
31 // the loop function runs over and over again forever
```

Done uploading.
"C:\Users\ACER\AppData\Local\Arduino15\packages\arduino\tools\avr-acc\17.3.0-atmel3.6.1-arduino7/bin/avr-acc-ar" rcs "C:\Users\ACER\Ar

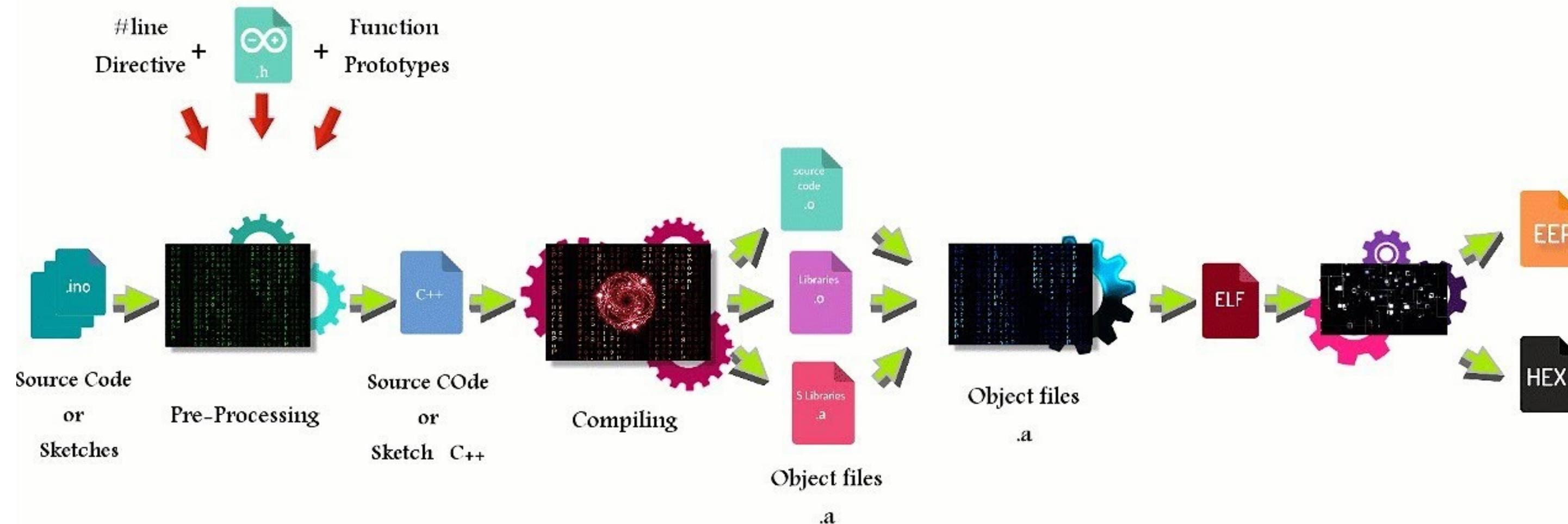
Upload

What is Compiler ?



- คอมไพล์ เป็นโปรแกรมที่แปลงโค้ดโปรแกรมจากภาษาบุษย์เป็นรหัสเครื่อง (ไบนารี) เพื่อให้คอมพิวเตอร์เข้าใจได้.
- ตรวจสอบข้อผิดพลาด เป็นกระบวนการตรวจสอบโค้ดเพื่อหาข้อผิดพลาดและแจ้งเตือนผู้พัฒนา.

How code upload to Arduino ?



Arduino IDE Build Process

Code Explain

Blink | Arduino 1.8.7

File Edit Sketch Tools Help



```
1
2 // the setup function runs once when you press reset or power the board
3 void setup() {
4     // initialize digital pin LED_BUILTIN as an output.
5     pinMode(LED_BUILTIN, OUTPUT);
6 }
7
8 // the loop function runs over and over again forever
9 void loop() {
10    digitalWrite(LED_BUILTIN, HIGH);           // turn the LED on (HIGH is the voltage level)
11    delay(1000);                            // wait for a second
12    digitalWrite(LED_BUILTIN, LOW);           // turn the LED off by making the voltage LOW
13    delay(1000);                            // wait for a second
14 }
```

C++ / Comment Code

Comments

// Single line comment

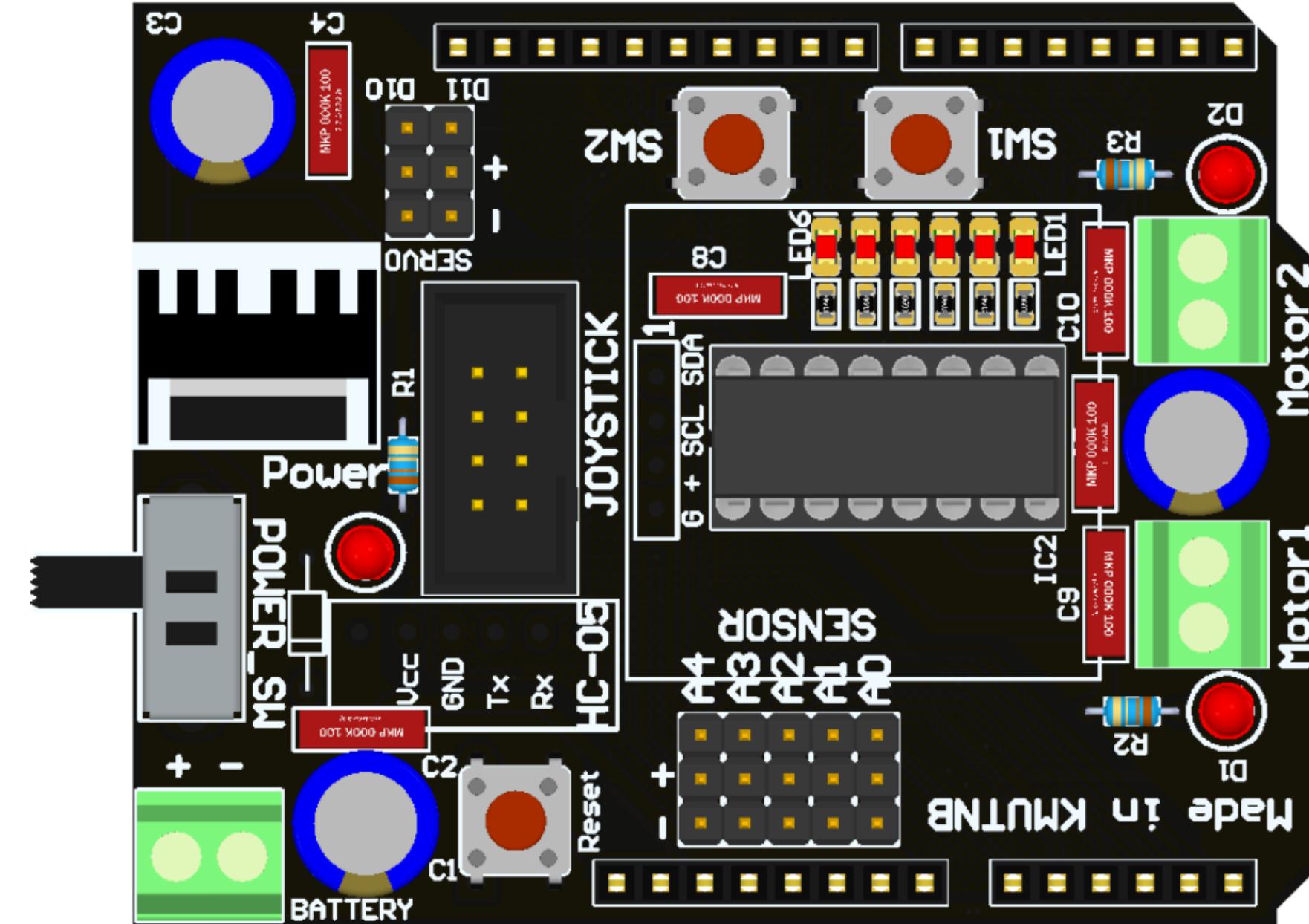
/* Multi-line comment */

๐๖

ตัวอย่าง

```
3 void setup() {  
4  
5  
6 // สวัสดีครับ  
7 /*  
8 // สวัสดีครับ  
9 */
```

Board Shield “Inno KMUTNB”

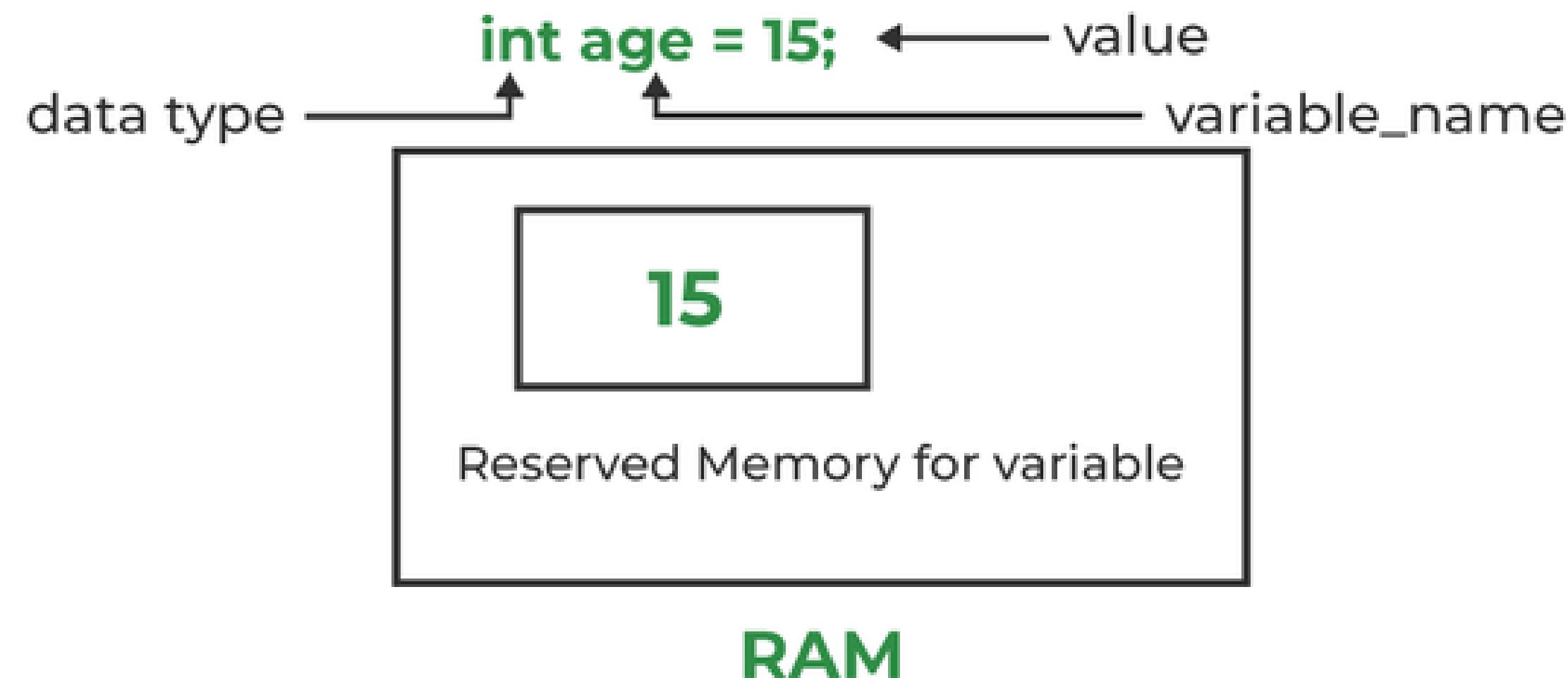


Pinout

Board Inno Kmutnb																					
ลำดับ	อุปกรณ์	จำนวน	Arduino Uno R3																		
			D0	D1	D2	~D3	D4	~D5	~D6	D7	D8	~D9	~D10	~D11	D12	D13	A0	A1	A2	A3	A4
1	Switch	2			SW1			SW2													
2	Sensor IR	3															S2	S3	S4	S5	S1
3	Servo	1												1							
4	Motor	2				PWM2			INB2	INB1	INA1			PWM1	INA2						
5	Bluetooth	1			Tx			Rx													
6	Joy	1														Right	X1	X2	Left	Up	Down
7	OLED	1																	SDA	SCL	
8	LED	6					5			2	1			3	4	6					

C++ / variable

Variable in C++



C++ / Data type

Data Type	Size (Bytes)	Range of Values
void	0	null
bool/boolean	1	True/False
char	1	-128 to +127
unsigned char	1	0 to 255
byte	1	0 to 255
int	2	-32,768 to 32,767
unsigned int	2	0 to 65,535
word	2	0 to 65,535
long	4	-2,147,483,648 to 2,147,483,647
unsigned long	4	0 to 4,294,967,295
float	4	-3.4028235E+38 to 3.4028235E+38
double	4	-3.4028235E+38 to 3.4028235E+38
string	-	character array

C++ / Declare variables

ชนิดของตัวแปร ชื่อตัวแปร

ชนิดของตัวแปร ชื่อตัวแปรที่ 1, ชื่อตัวแปรที่ 2, ชื่อตัวแปรที่ n

```
int x = 0 ;  
byte y = 255;  
char ch = 'A';  
String text = "Hello Arduino";  
float pi = 3.14;
```

C++ /Warning

```
int int = 0;      Syntax Error
String 1Text = 0;
String text no = "No";
```

```
bool isHot = true;  คนล่ำตัวแปรกัน
bool IsHot = true;
```

C++ / Declare variables

ชนิดของตัวแปร ชื่อตัวแปร

ชนิดของตัวแปร ชื่อตัวแปรที่ 1, ชื่อตัวแปรที่ 2, ชื่อตัวแปรที่ n

```
int x = 0 ;  
byte y = 255;  
char ch = 'A';  
String text = "Hello Arduino";  
float pi = 3.14;
```

C++ / Constant Variable

Syntax

```
#define constantName value
```

Parameters

constantName: the name of the macro to define.

value: the value to assign to the macro.

Example Code

```
#define ledPin 3  
// The compiler will replace any mention of ledPin with the value 3 at compile time.
```

C++ / Operator

เครื่องหมาย	การกระทำ	ตัวอย่าง	คำอธิบาย
+	บวก	$x=y+z;$	x เพิ่มค่าในตัวแปร y บวกกับค่าในตัวแปร z
-	ลบ	$x=y-z;$	x เพิ่มค่าในตัวแปร y ลบด้วยค่าในตัวแปร z
*	คูณ	$x=y*z;$	x เพิ่มค่าในตัวแปร y คูณด้วยค่าในตัวแปร z
/	หาร	$x=y/z;$	x เพิ่มค่าในตัวแปร y หารด้วยค่าในตัวแปร z
%	หารเอาเศษ	$x=y \% z;$	x เพิ่มเศษของการหารระหว่างตัวแปร y กับตัวแปร z

เครื่องหมาย	การกระทำ	ตัวอย่าง	คำอธิบาย
++	เพิ่มค่า 1 ค่า	$x++;$	เพิ่มค่า x ขึ้น 1 ค่า
--	ลดค่า 1 ค่า	$x--;$	ลดค่า x ลง 1 ค่า
+=	บวก	$x+=2;$	x ใหม่เท่ากับ x เดิมบวกกับ 2
-=	ลบ	$x-=2;$	x ใหม่เท่ากับ x เดิมลบด้วย 2
=	คูณ	$x=2;$	x ใหม่เท่ากับ x เดิมคูณด้วย 2

C++ /Operator Comparison

เครื่องหมาย	การกระทำ	ตัวอย่าง	คำอธิบาย
>	มากกว่า	if($x > 10$)	× มากกว่า 10 ใช่หรือไม่
<	น้อยกว่า	if($x < 10$)	× น้อยกว่า 10 ใช่หรือไม่
>=	มากกว่าหรือเท่ากับ	if($x \geq 10$)	× มากกว่าหรือเท่ากับ 10 ใช่หรือไม่
<=	น้อยกว่าหรือเท่ากับ	if($x \leq 10$)	× น้อยกว่าหรือเท่ากับ 10 ใช่หรือไม่
= =	เท่ากับ	if($x == 10$)	× เท่ากับ 10 ใช่หรือไม่
!=	ไม่เท่ากับ	if($x != 10$)	× ไม่เท่ากับ 10 ใช่หรือไม่

เครื่องหมาย	การกระทำ	คำอธิบาย
&&	แอนด์	เชื่อมเงื่อนไขระหว่าง 2 เงื่อนไขด้วยคำว่า “และ”
	ออร์	เชื่อมเงื่อนไขระหว่าง 2 เงื่อนไขด้วยคำว่า “หรือ”
!	อินเวิร์ส	ใส่ไว้หน้าตัวแปร จะทำให้ได้ค่าลํอจิกตรงข้าม เช่น if (!x)

C++ / Operator Priority

() []	Operators within parenthesis are performed first	Higher Lower
++, --	Postfix increment / decrement	
++, --	Prefix increment / decrement	
*, /, %	Multiplication, Division, Modulus	
+, -	Addition, Subtraction	
<, <=, >, >=	Less than, Less than or equal to, Greater than, Greater than or equal to	
==, !=	Equal to, Not equal to	
&&	Logical AND	
	Logical OR	
:?	Conditional Operator	
=	Simple Assignment	
+=, -=, *=, /=	Shorthand operators	
,	Comma operator	

Arduino Code with Output

Build-in function < pinMode() >

Syntax

```
pinMode(pin, mode)
```

Parameters

pin: the Arduino pin number to set the mode of.

mode: INPUT, OUTPUT, or INPUT_PULLUP. See the [Digital Pins](#) page for a more complete description of the functionality.

```
void setup() {  
  pinMode(13, OUTPUT);    // sets the digital pin 13 as output  
}  
  
void loop() {  
  digitalWrite(13, HIGH); // sets the digital pin 13 on  
  delay(1000);           // waits for a second  
  digitalWrite(13, LOW);  // sets the digital pin 13 off  
  delay(1000);           // waits for a second  
}
```

Build-in function < digitalWrite() >

Syntax

```
digitalWrite(pin, value)
```

Parameters

pin: the Arduino pin number.

value: HIGH or LOW.

```
void setup() {  
    pinMode(13, OUTPUT);      // sets the digital pin 13 as output  
}  
  
void loop() {  
    digitalWrite(13, HIGH);   // sets the digital pin 13 on  
    delay(1000);             // waits for a second  
    digitalWrite(13, LOW);    // sets the digital pin 13 off  
    delay(1000);             // waits for a second  
}
```

Build-in function < analogWrite() >

Description

Writes an analog value ([PWM wave](#)) to a pin. Can be used to light a LED at varying brightnesses or drive a motor at various speeds. After a call to `analogwrite()`, the pin will generate a steady rectangular wave of the specified duty cycle until the next call to `analogwrite()` (or a call to `digitalRead()` or `digitalWrite()`) on the same pin.

BOARD	PWM PINS	PWM FREQUENCY
Uno, Nano, Mini	3, 5, 6, 9, 10, 11	490 Hz (pins 5 and 6: 980 Hz)

Syntax

```
analogWrite(pin, value)
```

Parameters

`pin`: the Arduino pin to write to. Allowed data types: `int`.

`value`: the duty cycle: between 0 (always off) and 255 (always on). Allowed data types: `int`.

Build-in function < analogWrite() >

Example Code

Sets the output to the LED proportional to the value read from the potentiometer.

```
int ledPin = 9;      // LED connected to digital pin 9
int analogPin = 3;    // potentiometer connected to analog pin 3
int val = 0;          // variable to store the read value

void setup() {
  pinMode(ledPin, OUTPUT); // sets the pin as output
}

void loop() {
  val = analogRead(analogPin); // read the input pin
  analogWrite(ledPin, val / 4); // analogRead values go from 0 to 1023, analogWrite values from 0 to 255
}
```

Build-in function < delay() >

Syntax

```
delay(ms)
```

Parameters

ms: the number of milliseconds to pause. Allowed data types: `unsigned long`.

```
void setup() {  
    pinMode(13, OUTPUT);      // sets the digital pin 13 as output  
}  
  
void loop() {  
    digitalWrite(13, HIGH);   // sets the digital pin 13 on  
    delay(1000);             // waits for a second  
    digitalWrite(13, LOW);    // sets the digital pin 13 off  
    delay(1000);             // waits for a second  
}
```

Arduino with Serial

Description

Used for communication between the Arduino board and a computer or other devices. All Arduino boards have at least one serial port (also known as a UART or USART), and some have several.

BOARD	USB CDC NAME	SERIAL PINS	SERIAL1 PINS	SERIAL2 PINS	SERIAL3 PINS
Uno, Nano, Mini		0(RX), 1(TX)			

Syntax

`Serial.println(val)`

`Serial.println(val, format)`

Parameters

`serial`: serial port object. See the list of available serial ports for each board on the [Serial main page](#).

`val`: the value to print. Allowed data types: any data type.

`format`: specifies the number base (for integral data types) or number of decimal places (for floating point types).

Arduino with Serial

Description

Used for communication between the Arduino board and a computer or other devices. All Arduino boards have at least one serial port (also known as a UART or USART), and some have several.

BOARD	USB CDC NAME	SERIAL PINS	SERIAL1 PINS	SERIAL2 PINS	SERIAL3 PINS
Uno, Nano, Mini		0(RX), 1(TX)			

Syntax

`Serial.println(val)`

`Serial.println(val, format)`

Parameters

`serial`: serial port object. See the list of available serial ports for each board on the [Serial main page](#).

`val`: the value to print. Allowed data types: any data type.

`format`: specifies the number base (for integral data types) or number of decimal places (for floating point types).

Serial.println() & Serial.print()

Syntax

```
Serial.println(val)
```

```
Serial.println(val, format)
```

Parameters

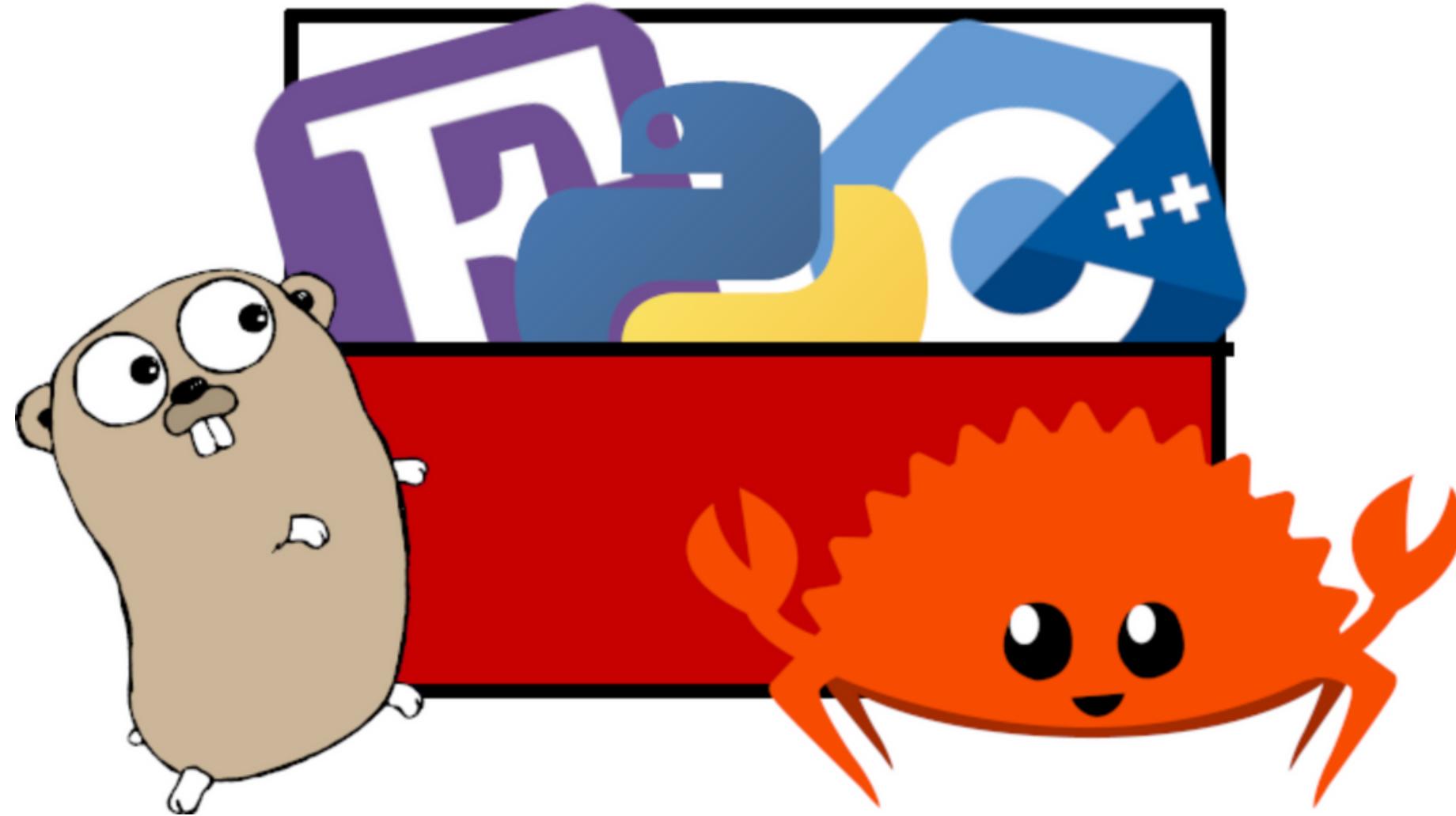
`Serial`: serial port object. See the list of available serial ports for each board on the [Serial main page](#).

`val`: the value to print. Allowed data types: any data type.

`format`: specifies the number base (for integral data types) or number of decimal places (for floating point types).

```
void setup() {  
    // open the serial port at 9600 bps:  
    Serial.begin(9600);  
}  
  
void loop() {  
    // read the analog input on pin 0:  
    analogValue = analogRead(0);  
  
    // print it out in many formats:  
    Serial.println(analogValue);          // print as an ASCII-encoded decimal  
    Serial.println(analogValue, DEC);      // print as an ASCII-encoded decimal  
    Serial.println(analogValue, HEX);      // print as an ASCII-encoded hexadecimal  
    Serial.println(analogValue, OCT);      // print as an ASCII-encoded octal  
    Serial.println(analogValue, BIN);      // print as an ASCII-encoded binary  
  
    // delay 10 milliseconds before the next reading:  
    delay(10);  
}
```

```
void setup() {  
    Serial.begin(9600);  
    Serial.print("Hello World");  
    Serial.println("Hello World");  
}
```



TRY 1 Serial
แสดงชื่อและนามสกุลตัวเองผ่าน Serial monitor

Function

function ไม่รับค่า และไม่ส่งคืนค่า

```
void turnON() {  
    digitalWrite( LED , HIGH );  
}
```

function รับค่า และไม่ส่งคืนค่า

```
void turnON(int pin) {  
    digitalWrite( pin , HIGH );  
}
```

function ไม่รับค่า และส่งคืนค่า

```
int getLED() {  
    return digitalRead(LED);  
}
```

function รับค่า และส่งคืนค่า

```
int getLED(int pin) {  
    return digitalRead(pin);  
}
```

ตัวอย่างการเรียก Function

- turnON();
- turnON(2);
- int data = getLED();
- int data = getLED(2);

Variable Scope

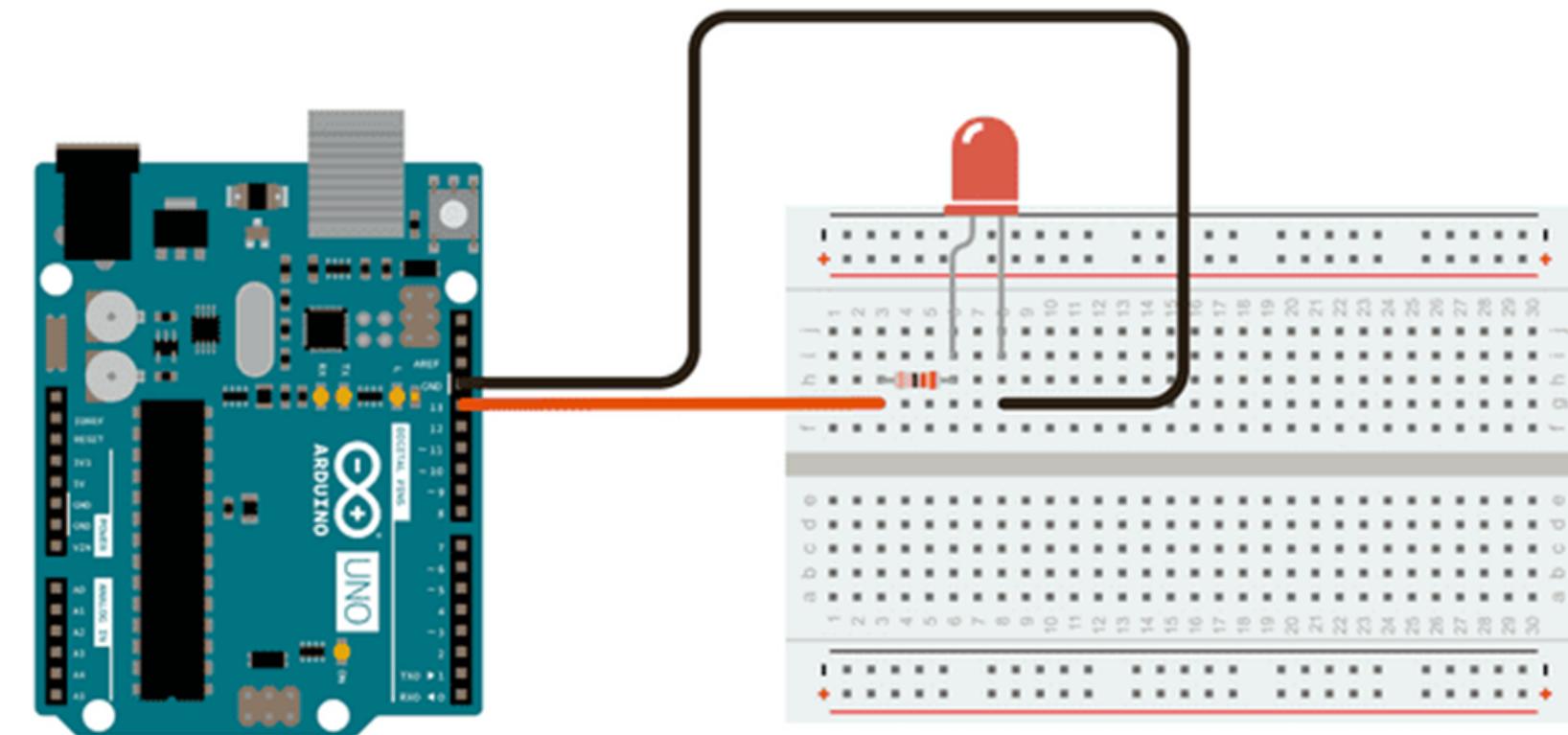
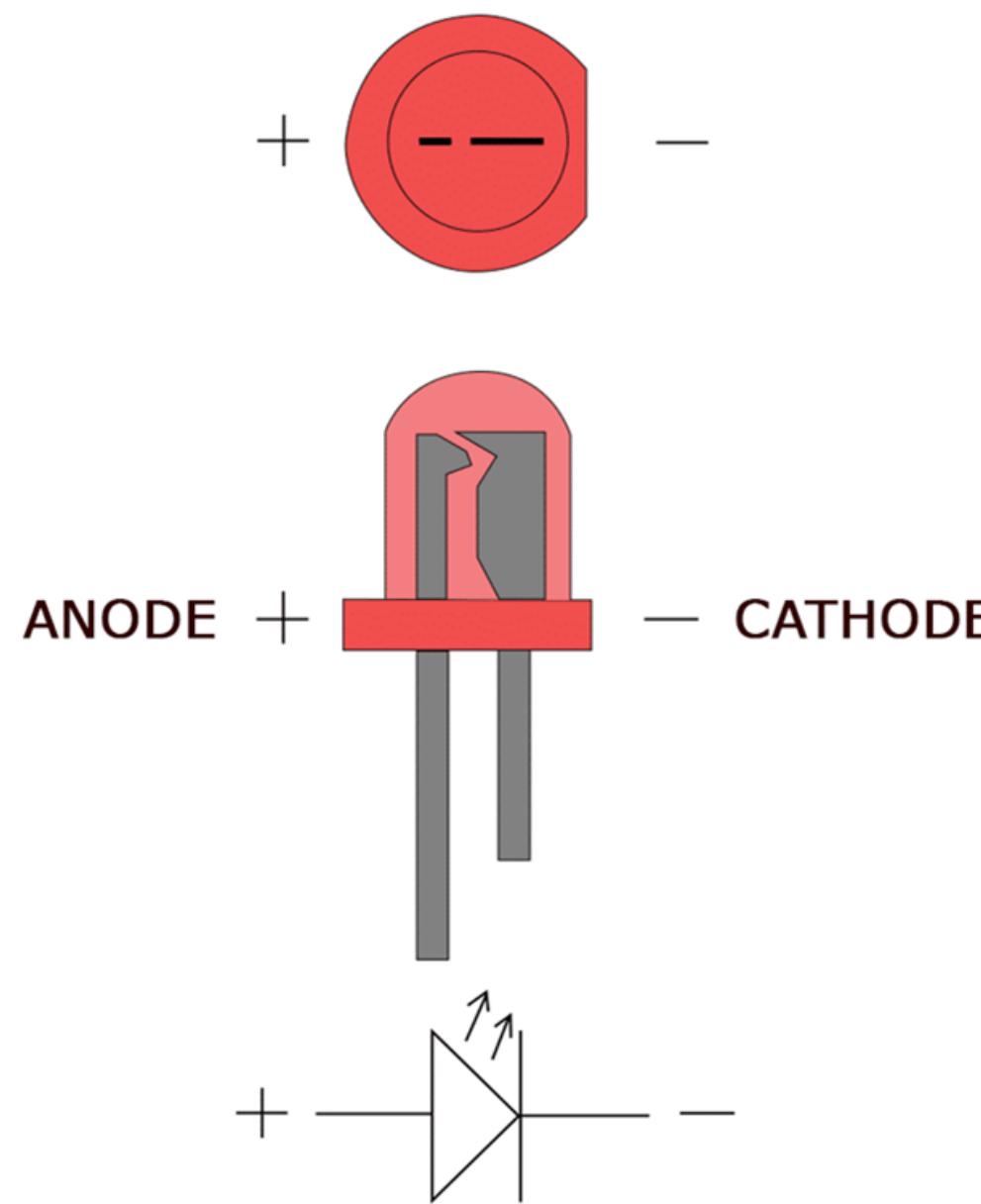
ตัวแปรประเภทโลกบอล (global)

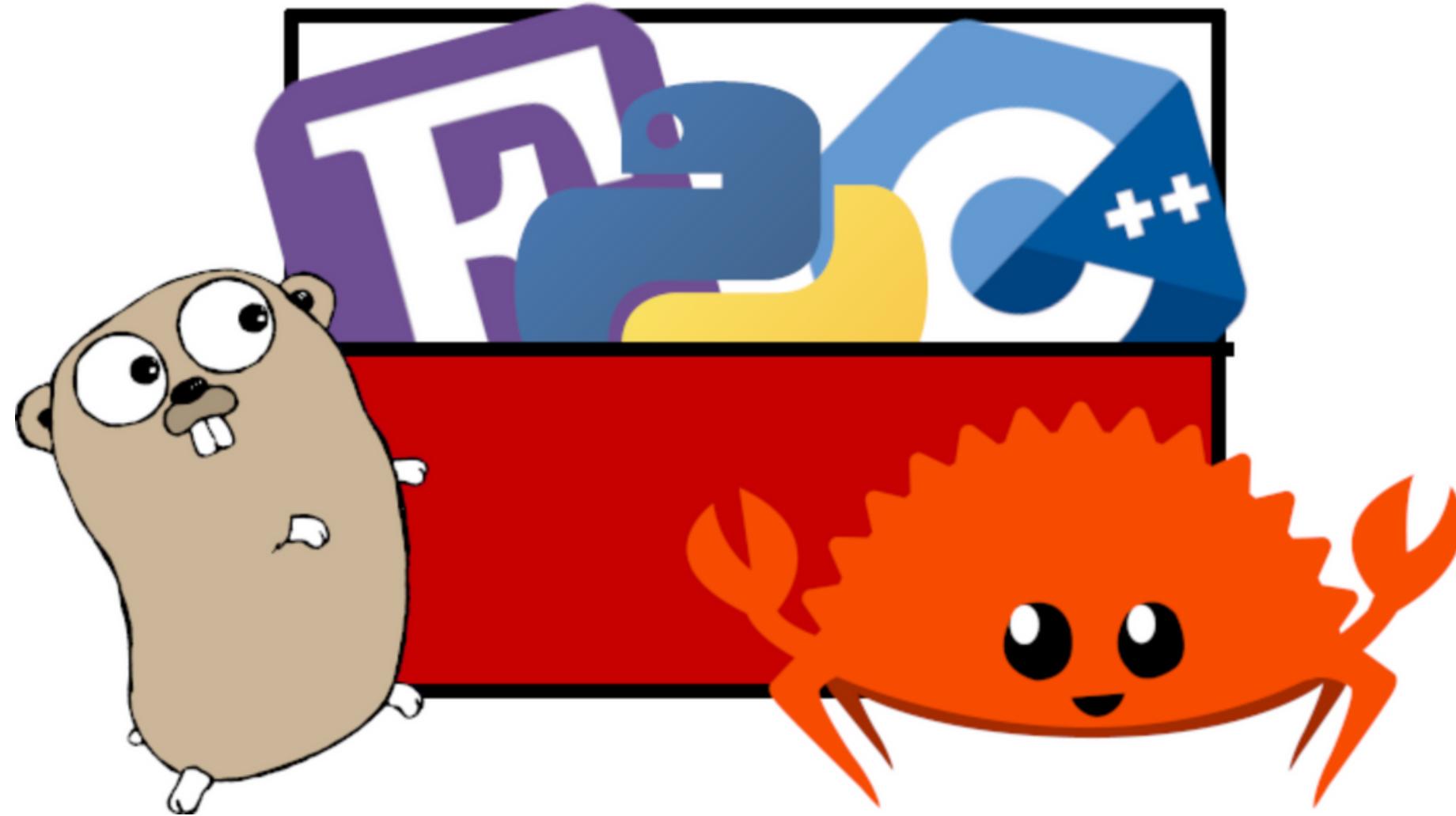
```
int number = 10;  
void setup() {  
    number = 3;  
}  
  
void loop() {  
    number = 5;  
}  
void turnON(int pin) {  
    number = 2;  
    digitalWrite( pin , HIGH );  
}  
int getLED(int pin) {  
    number = 5;  
    return digitalRead(pin);  
}
```

ตัวแปรประเภทโลกออล (local)

```
void setup() {  
    int number = 10;  
    number = 5;  
}  
void loop() {  
    number = 5;  
}
```

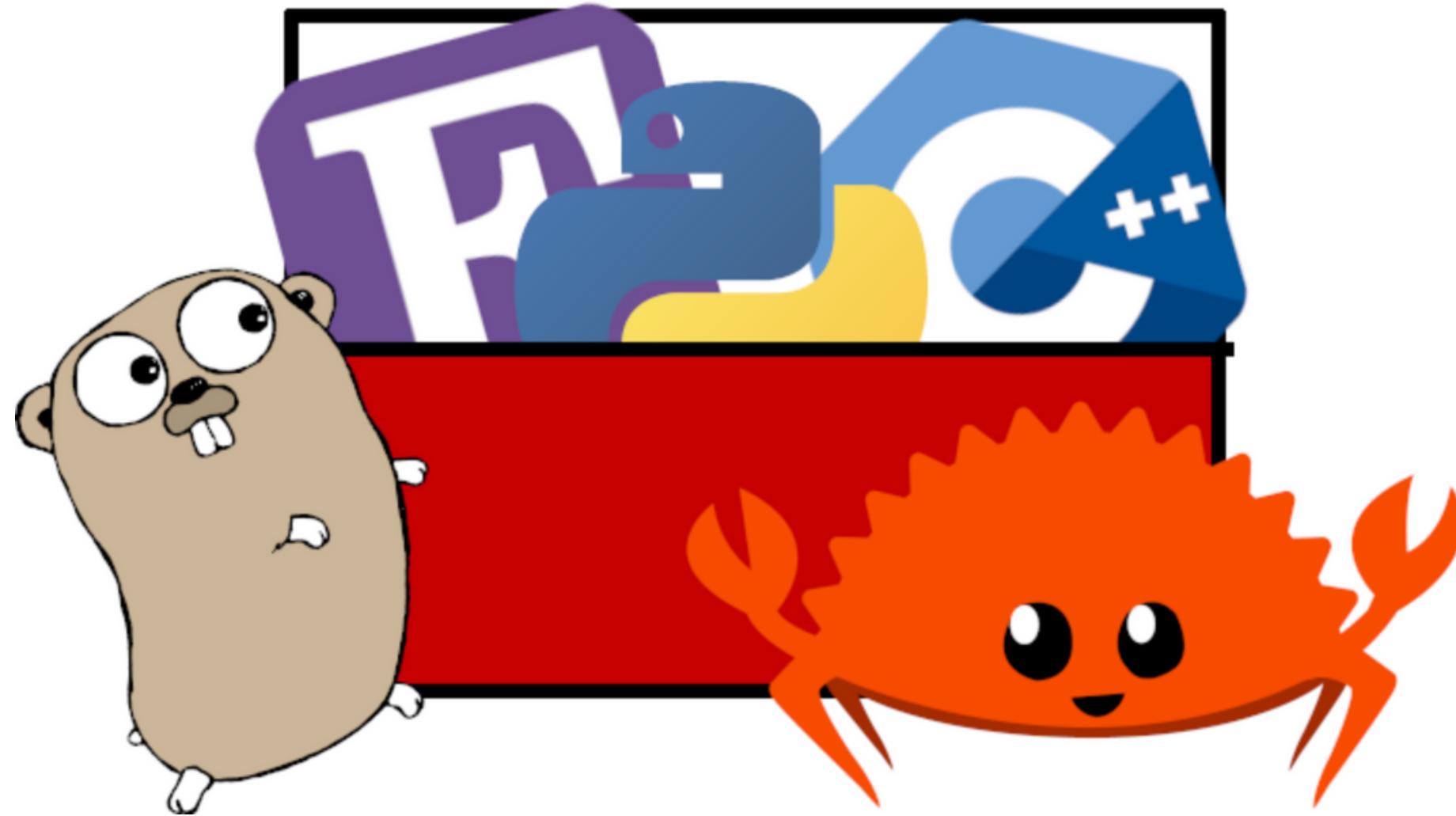
Output equipment / LED (light-emitting diode)





TRY 1 LED

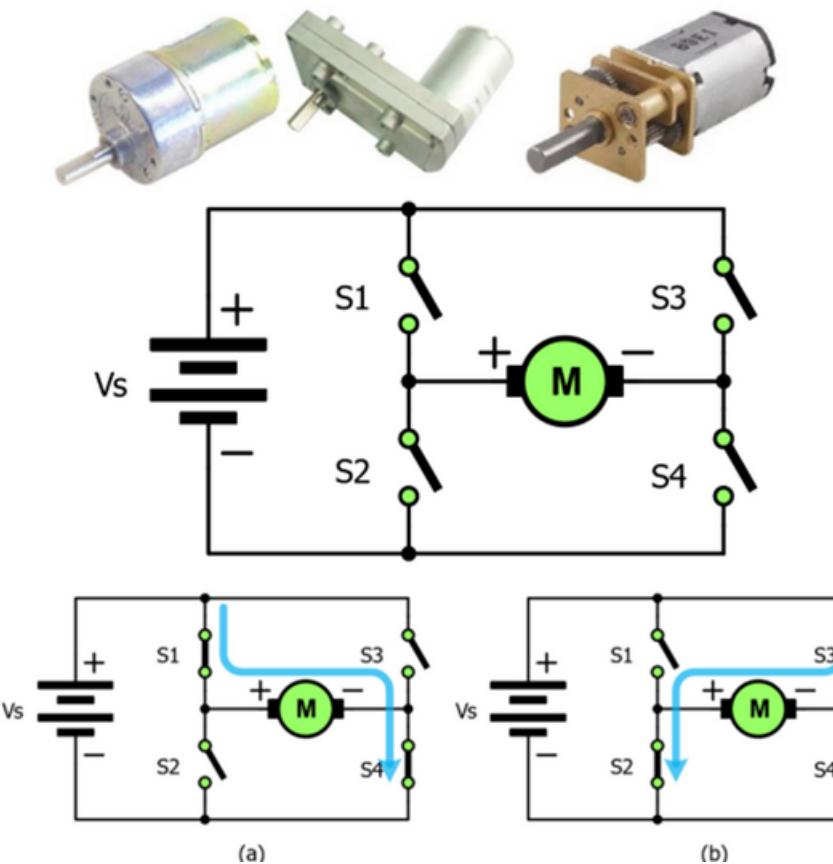
ให้ LED ติดทีละดวงจนถึงดวงสุดท้ายจากนั้นดับทีละ
ดวง เริ่มจากด้วยที่ 1



TRY 2 LED
ให้ LED ดวงที่ 3 ค่อยๆติดกีละ 25 %

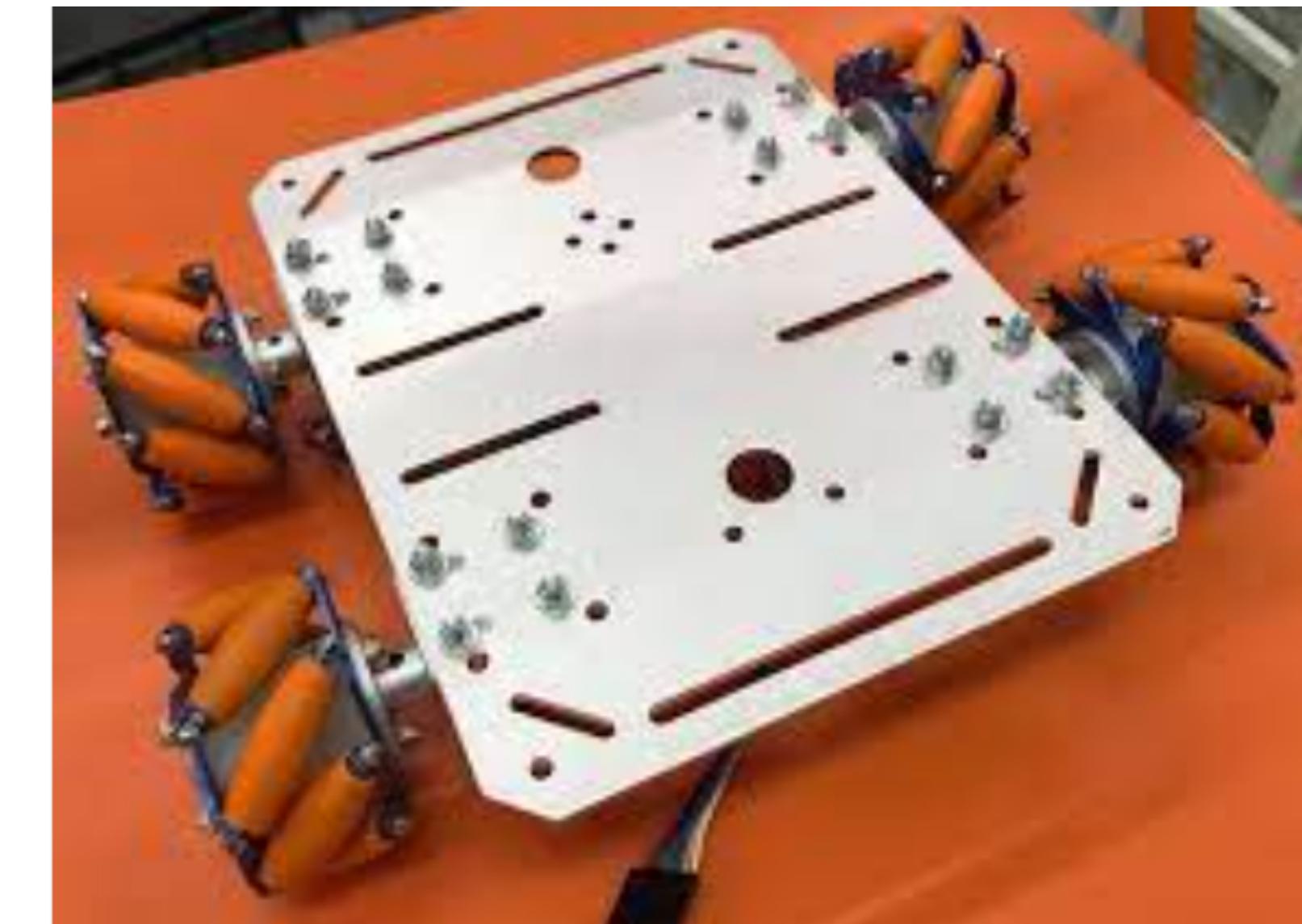
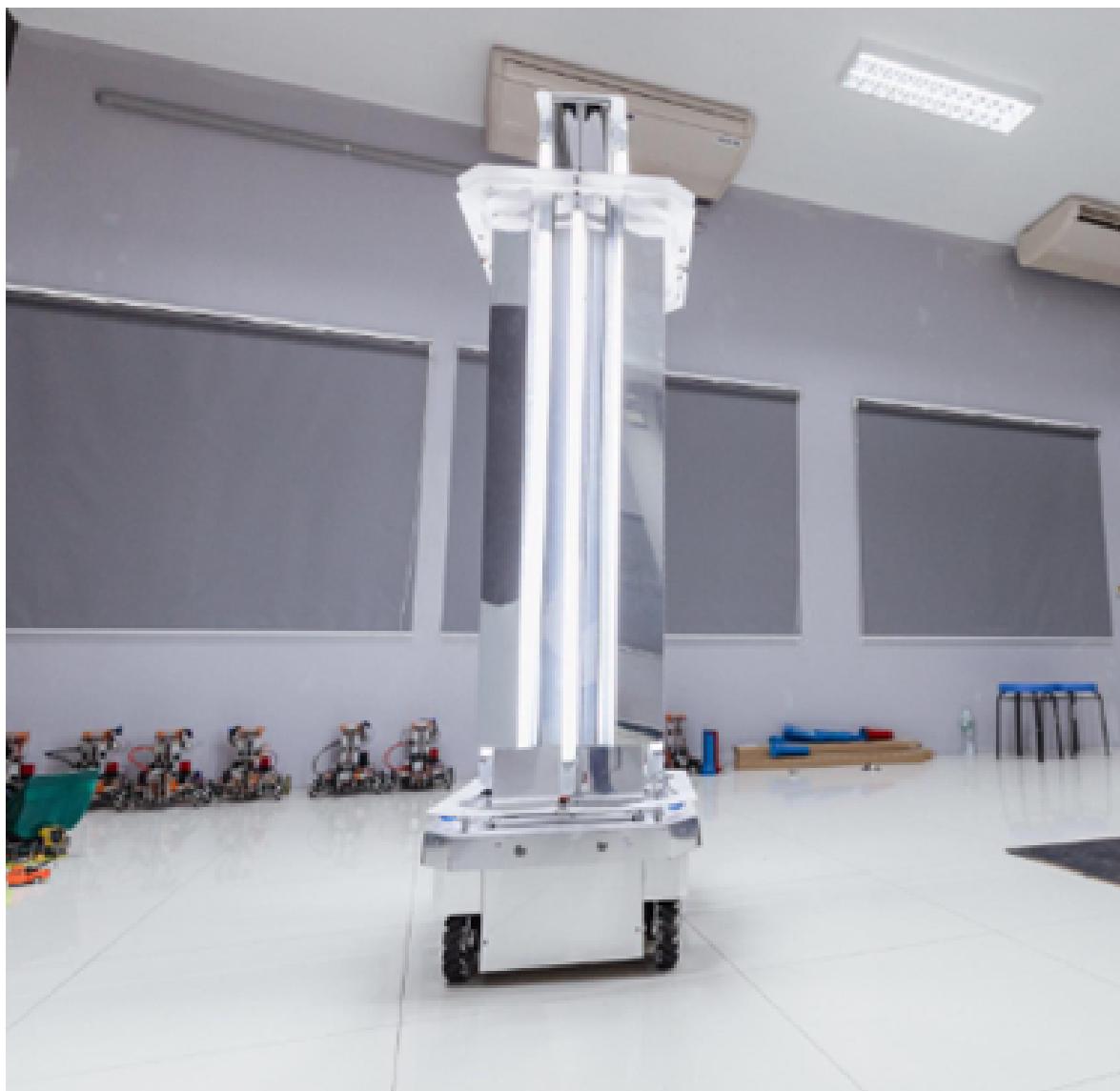
Output equipment / Motor

สถานะของการควบคุมมอเตอร์			สถานะของมอเตอร์ไฟฟ้ากระแสสลับ
INA	INB	PWM	
1	1	0	ไม่หมุน
1	0	0	ไม่หมุน
1	0	(1-255)	หมุนตามเข็มนาฬิกา (CW)
0	1	0	ไม่หมุน
0	1	(1-255)	หมุนทวนเข็มนาฬิกา (CCW)
1	1	255	ไม่หมุน
0	0	255	ไม่หมุน
0	0	0	ไม่หมุน



THEORY OF ROBOT MOVEMENT

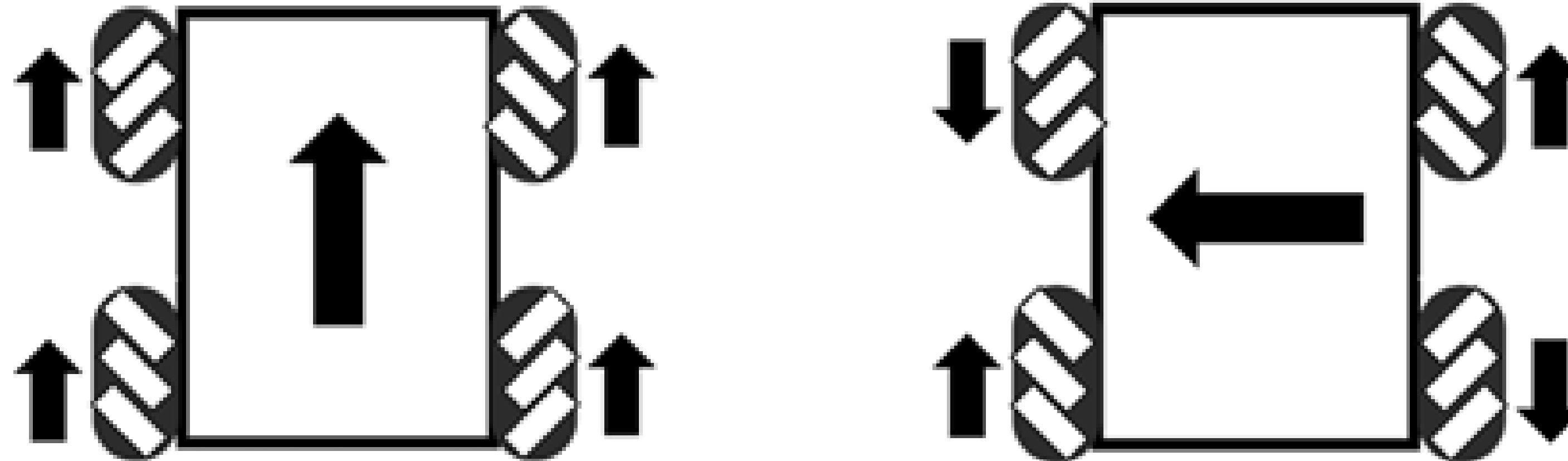
Holonomics movement Mecanum Drive



THEORY OF ROBOT MOVEMENT

Holonomics movement

Mecanum Drive

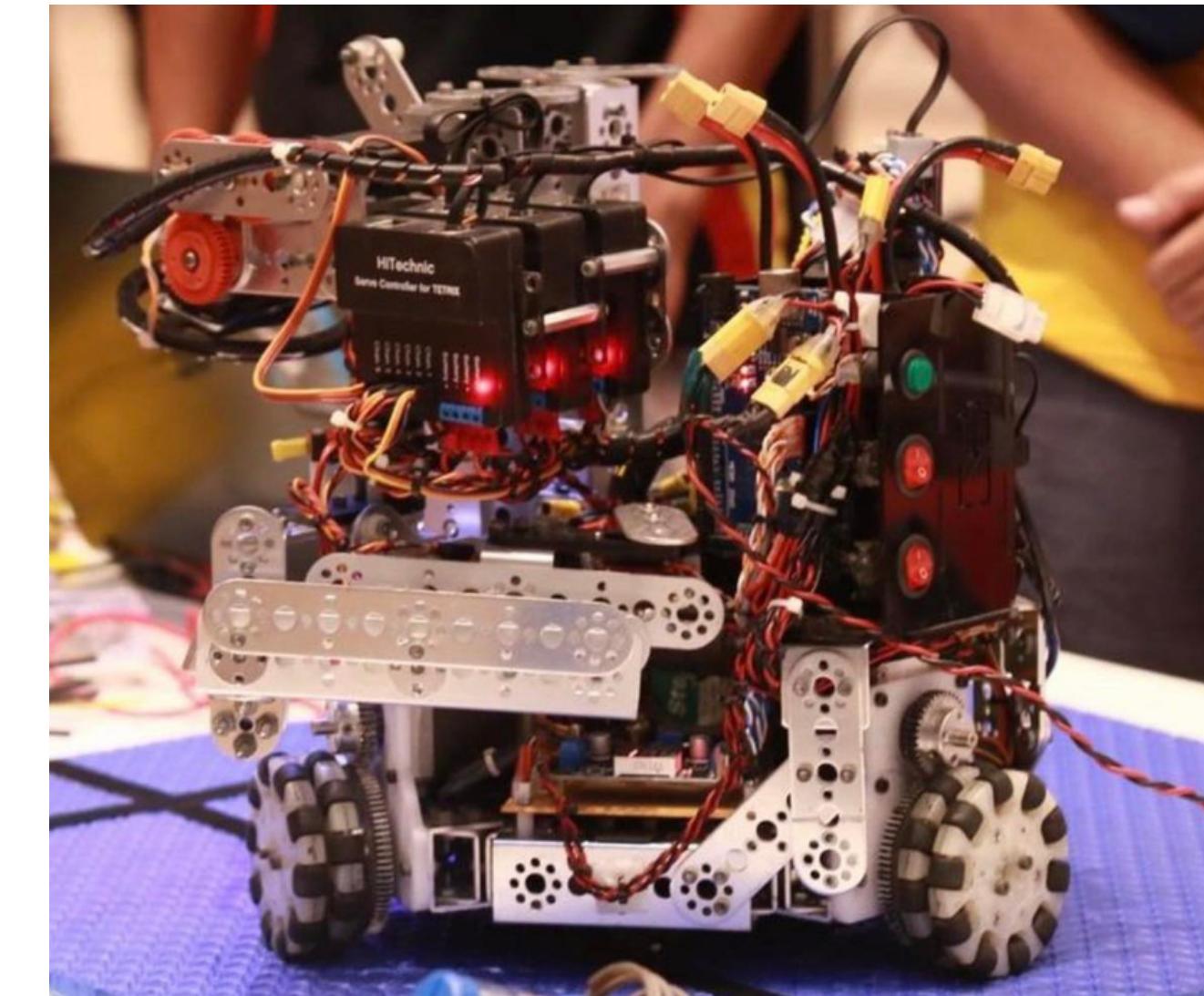
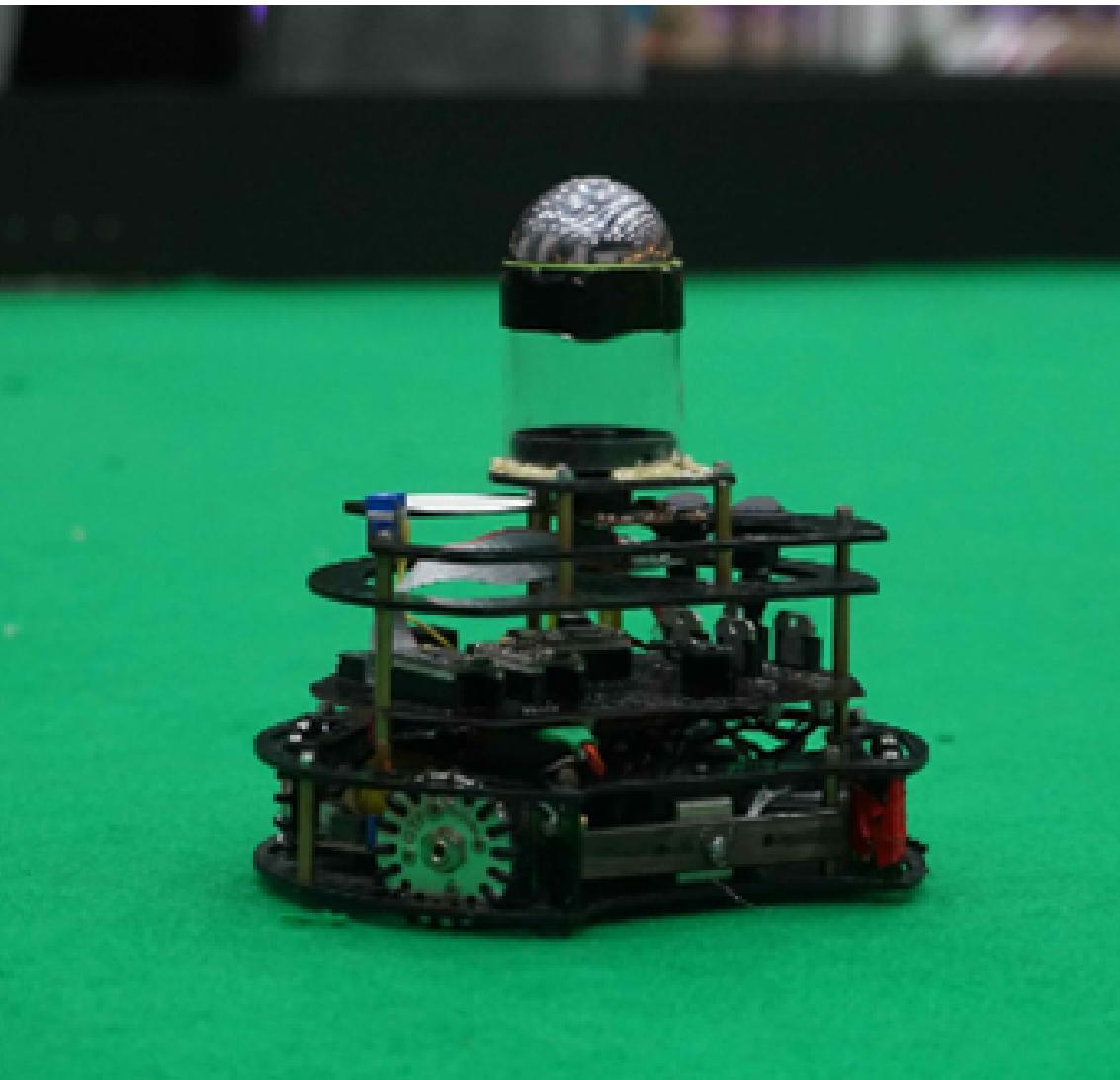


ข้อดี สามารถเคลื่อนที่แบบ Holonomics ได้
ข้อเสีย หาเกิดการลื่นไถล (slip) จะทำให้การเคลื่อนที่ไม่ดี จึง
ต้องออกแบบช่วงล่างทำให้ล้อทุกล้อสัมผัสนิ่มที่สุด

THEORY OF ROBOT MOVEMENT

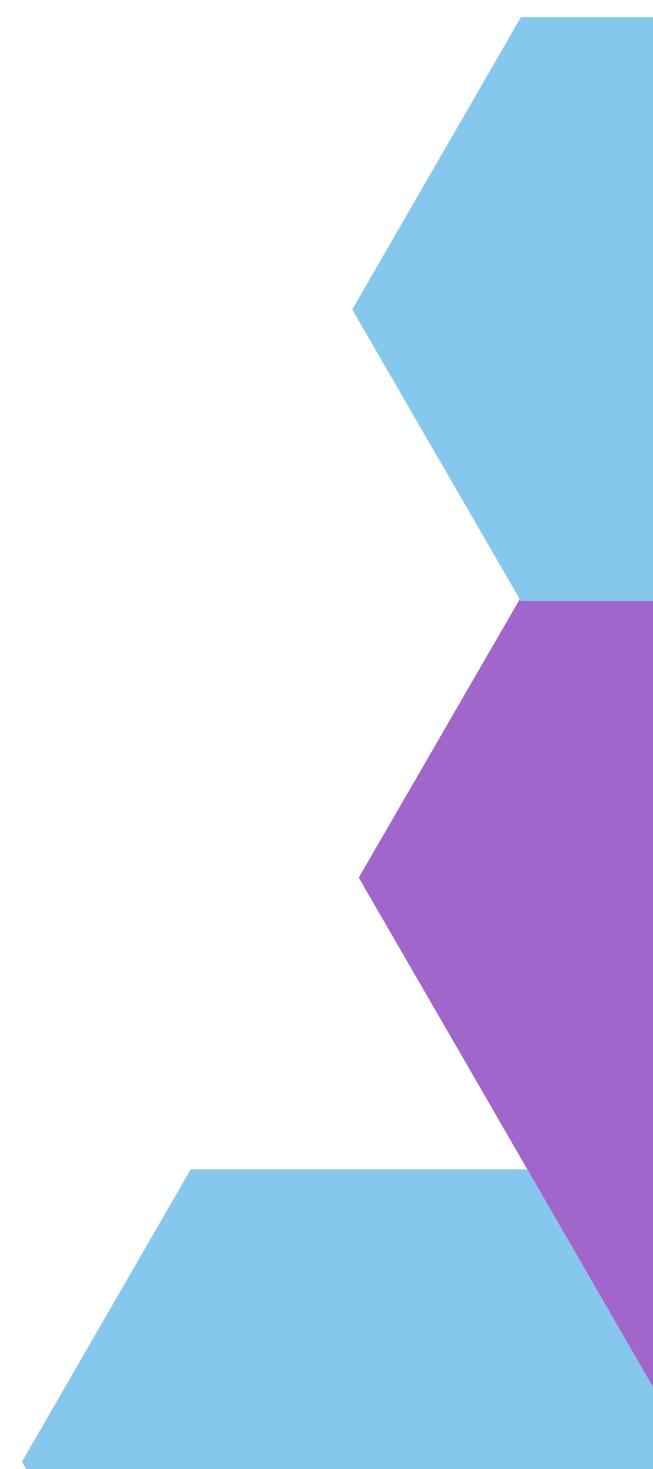
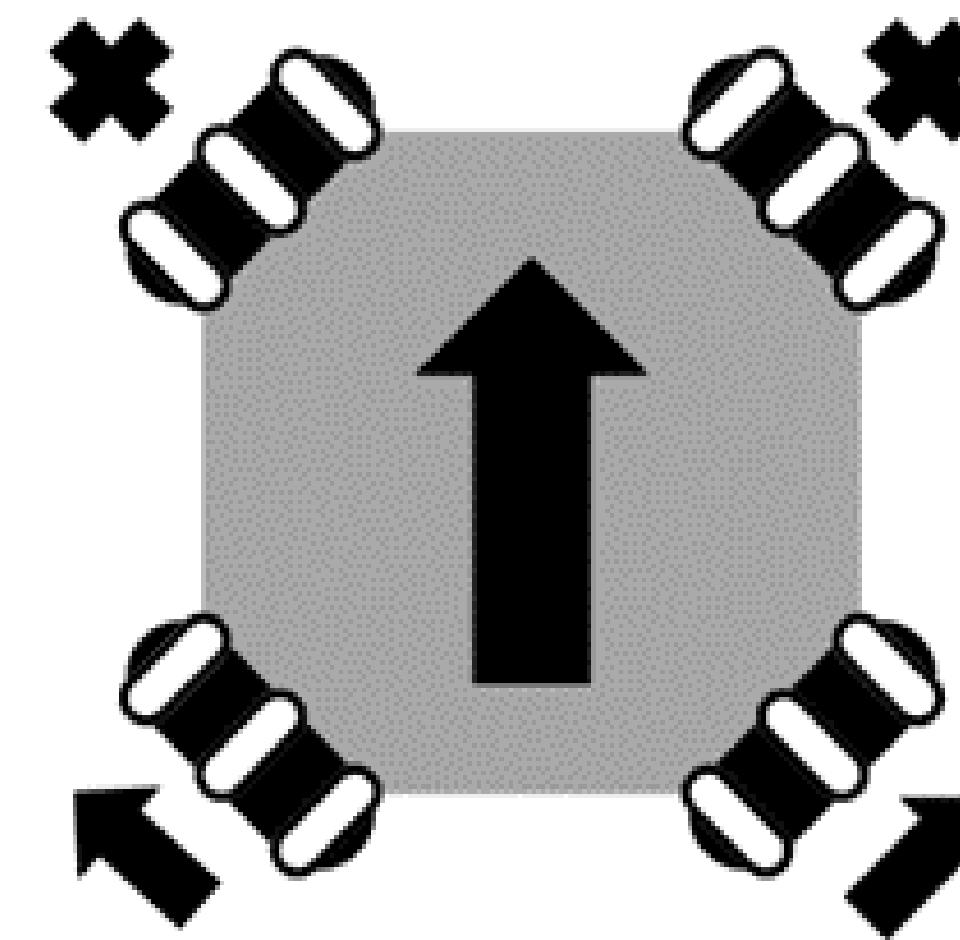
Holonomics movement

Omni Drive



THEORY OF ROBOT MOVEMENT

Holonomics movement Omni Drive



THEORY OF ROBOT MOVEMENT

Non-Holonomics movement

Differential Drive



THEORY OF ROBOT MOVEMENT

Non-Holonomics movement

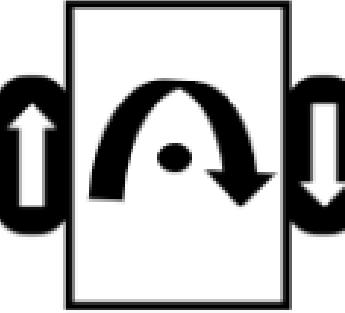
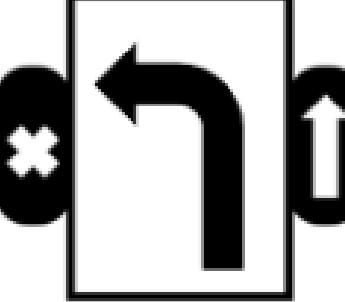
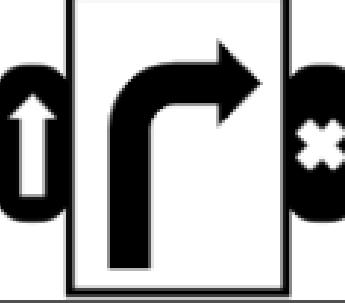
Differential Drive

ลักษณะการเคลื่อนที่	การหมุนของ ส้อซ้าย	การหมุนของ ส้อขวา	ทิศทางการเคลื่อนที่
	หมุนไปด้านหน้า หรือ ทวนเข็มนาฬิกา	หมุนไปด้านหน้า หรือ ตามเข็มนาฬิกา	หุ่นยนต์เคลื่อนที่ไปด้านหน้า
	หมุนไปด้านหลัง หรือ ตามเข็มนาฬิกา	หมุนไปด้านหลัง หรือ ทวนเข็มนาฬิกา	หุ่นยนต์เคลื่อนที่ไปด้านหลัง
	หมุนไปด้านหลัง หรือ ตามเข็มนาฬิกา	หมุนไปด้านหน้า หรือ ตามเข็มนาฬิกา	หุ่นยนต์หมุนซ้าย

THEORY OF ROBOT MOVEMENT

Non-Holonomics movement

Differential Drive

	หมุนไปด้านหน้า หรือ ทวนเข็มนาฬิกา	หมุนไปด้านหลัง หรือ ทวนเข็มนาฬิกา	หุ่นยนต์หมุนขวา
	ไม่หมุน	หมุนไปด้านหน้า หรือ ตามเข็มนาฬิกา	หุ่นยนต์เลี้ยวซ้าย
	หมุนไปด้านหน้า หรือ ทวนเข็มนาฬิกา	ไม่หมุน	หุ่นยนต์เลี้ยวขวา

THEORY OF ROBOT MOVEMENT

Non-Holonomics movement

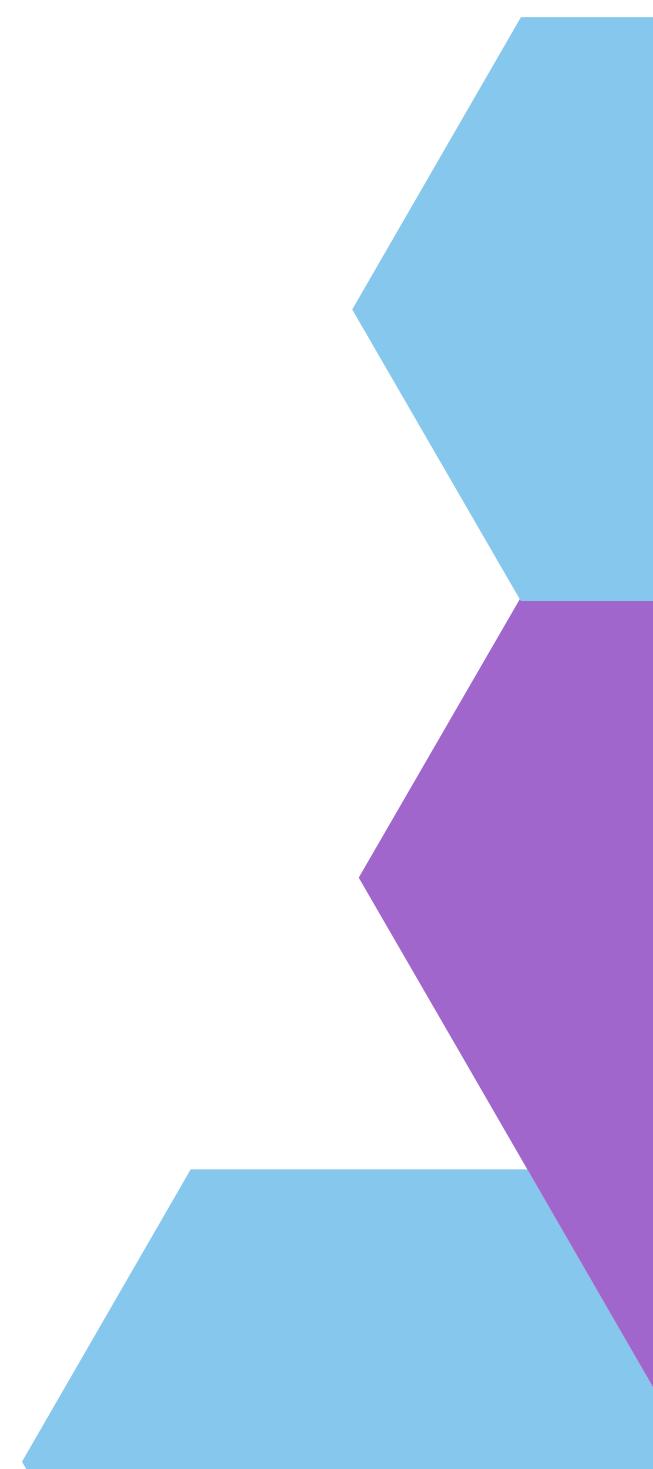
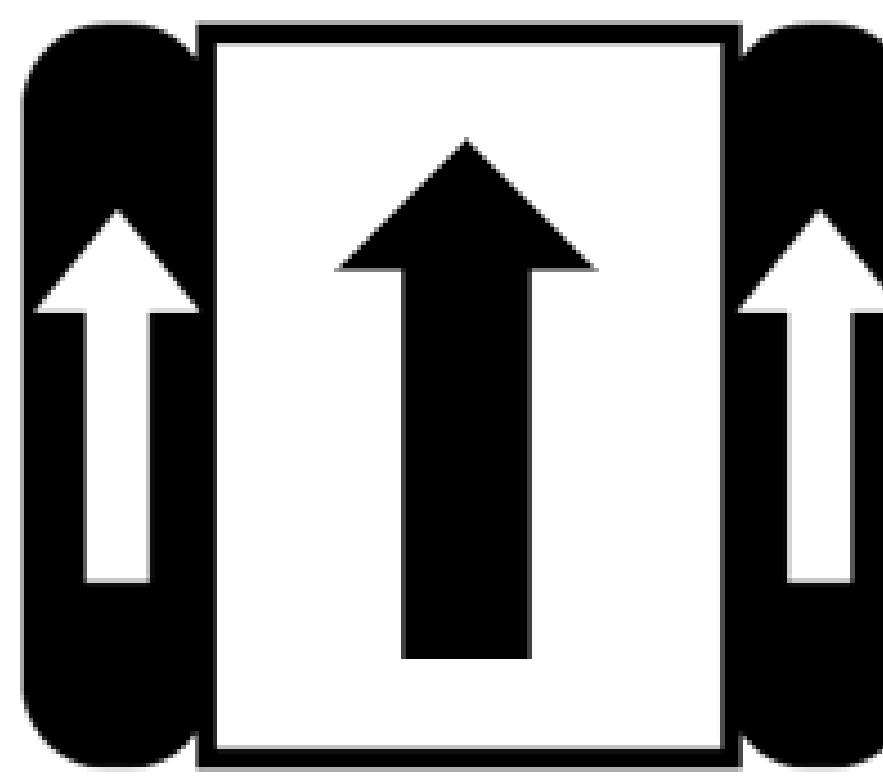
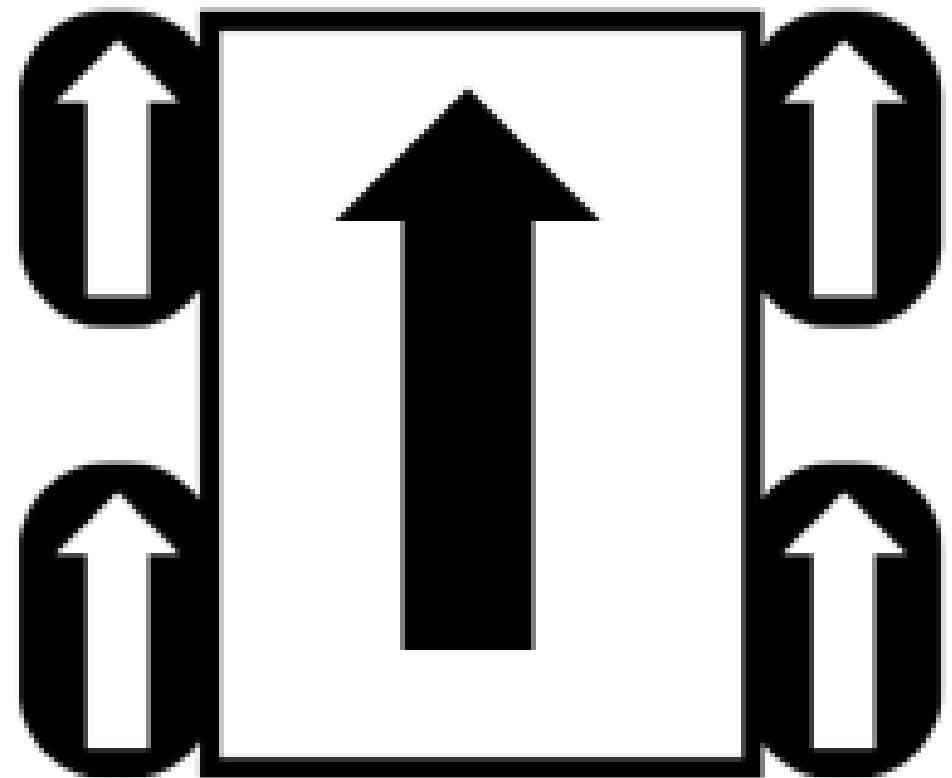
Skid steering Drive



THEORY OF ROBOT MOVEMENT

Non-Holonomics movement

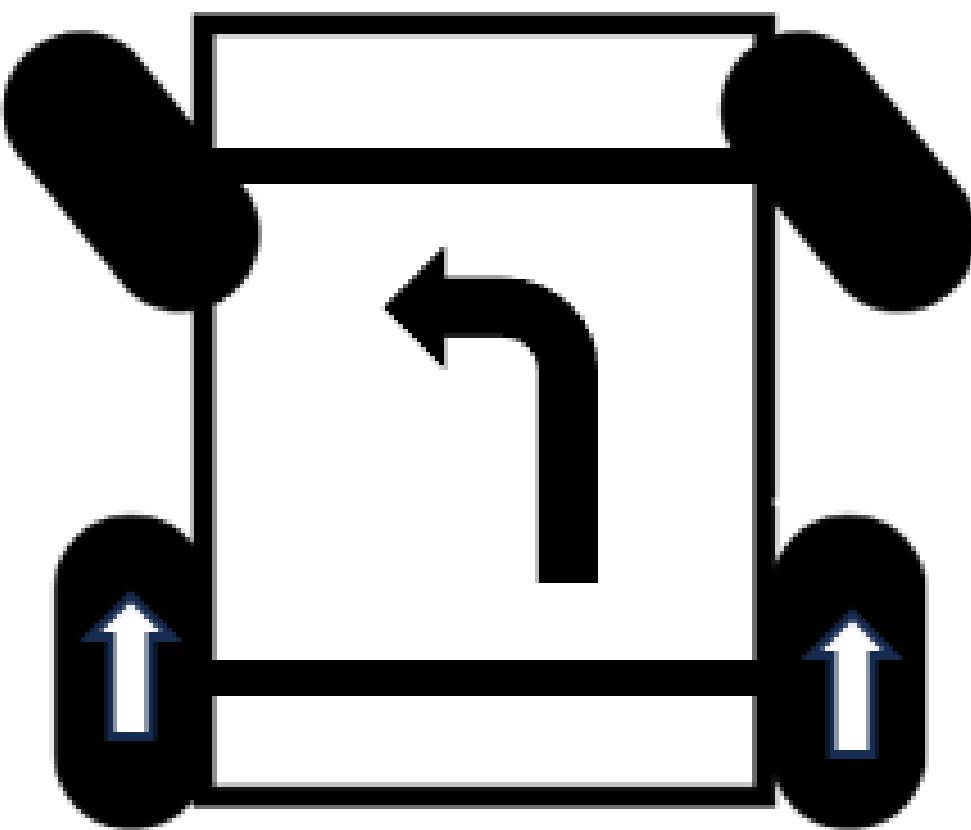
Skid steering Drive

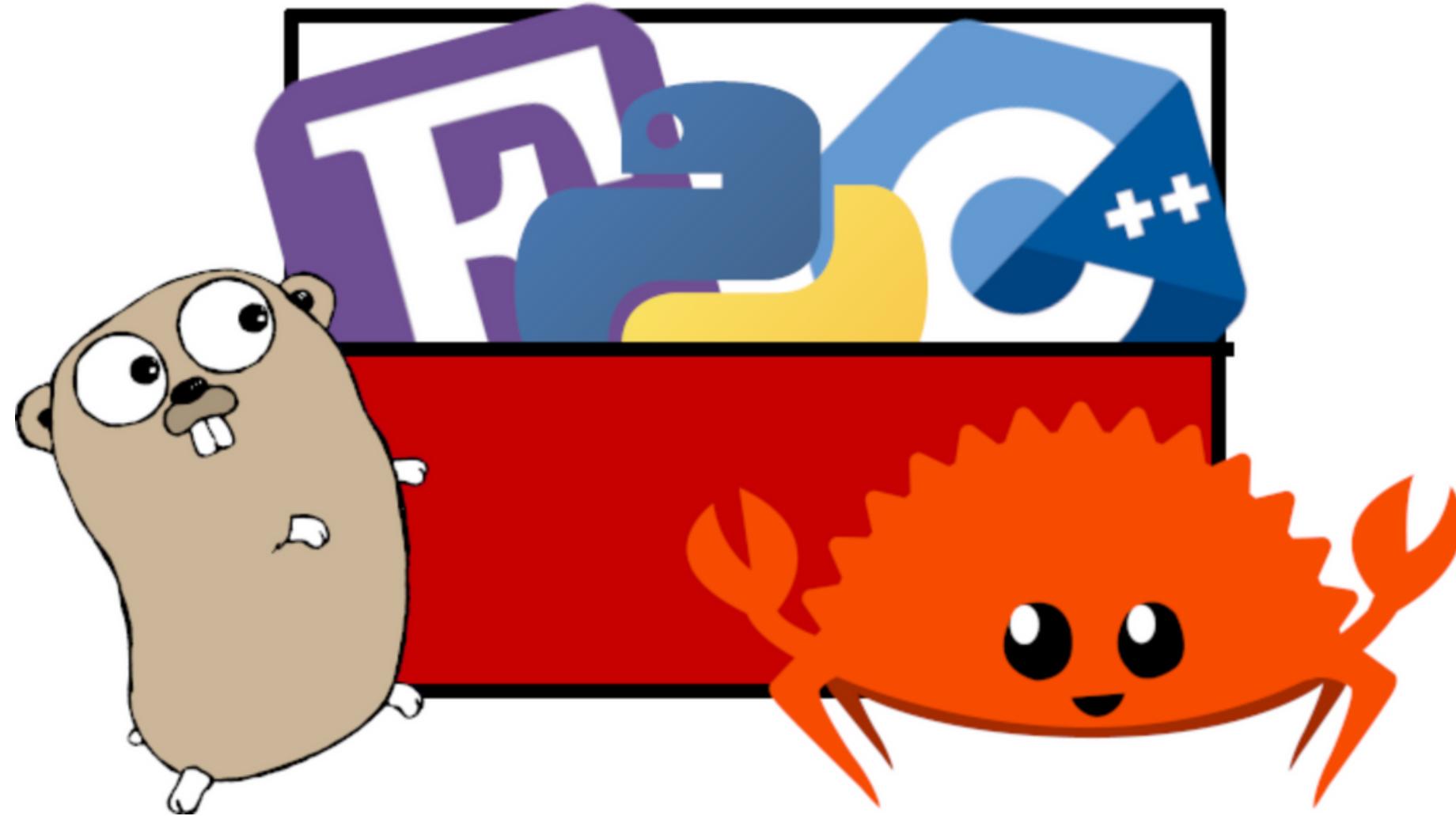


THEORY OF ROBOT MOVEMENT

Non-Holonomics movement

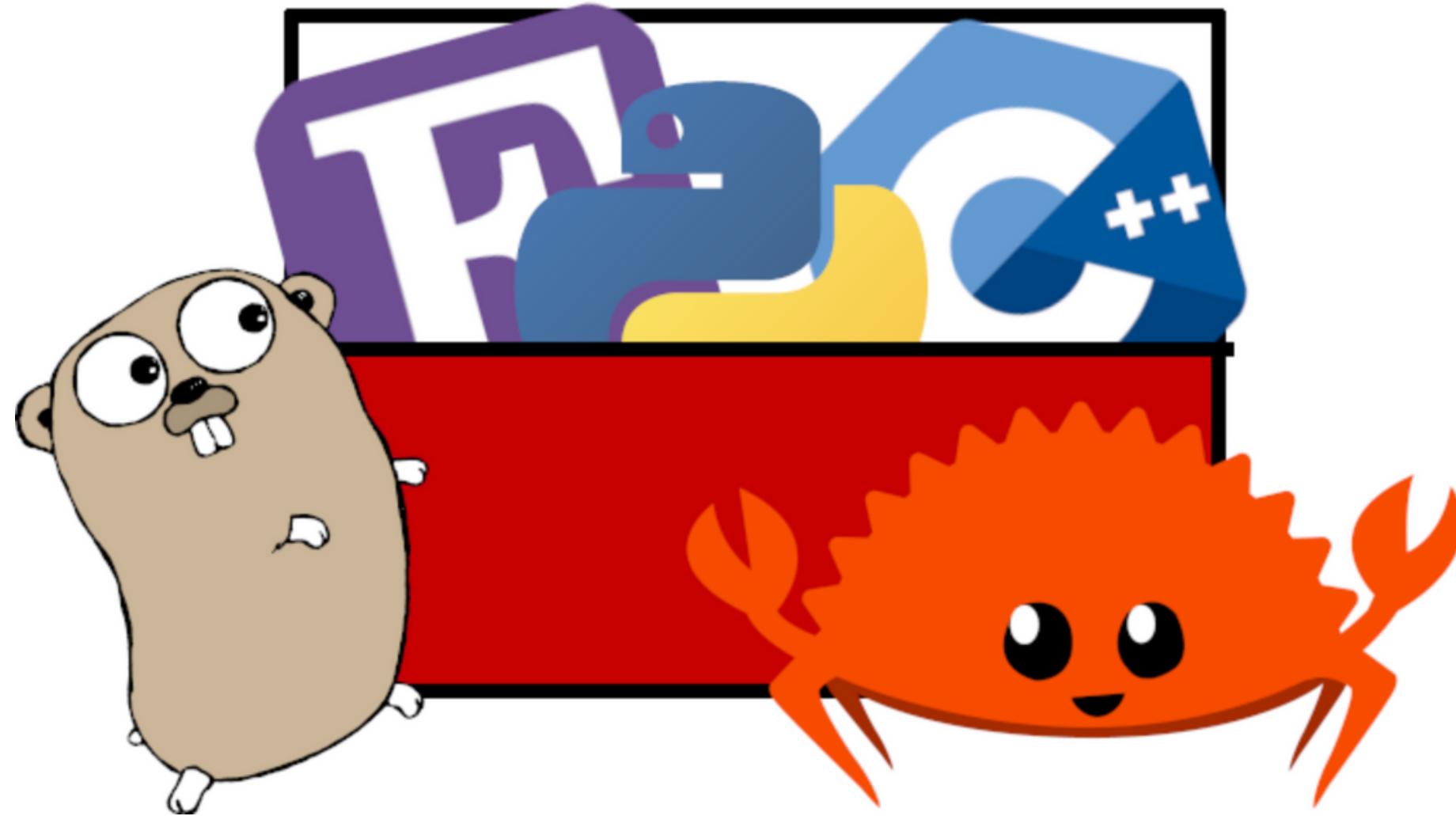
Ackermann steering Drive





TRY 1 Motor

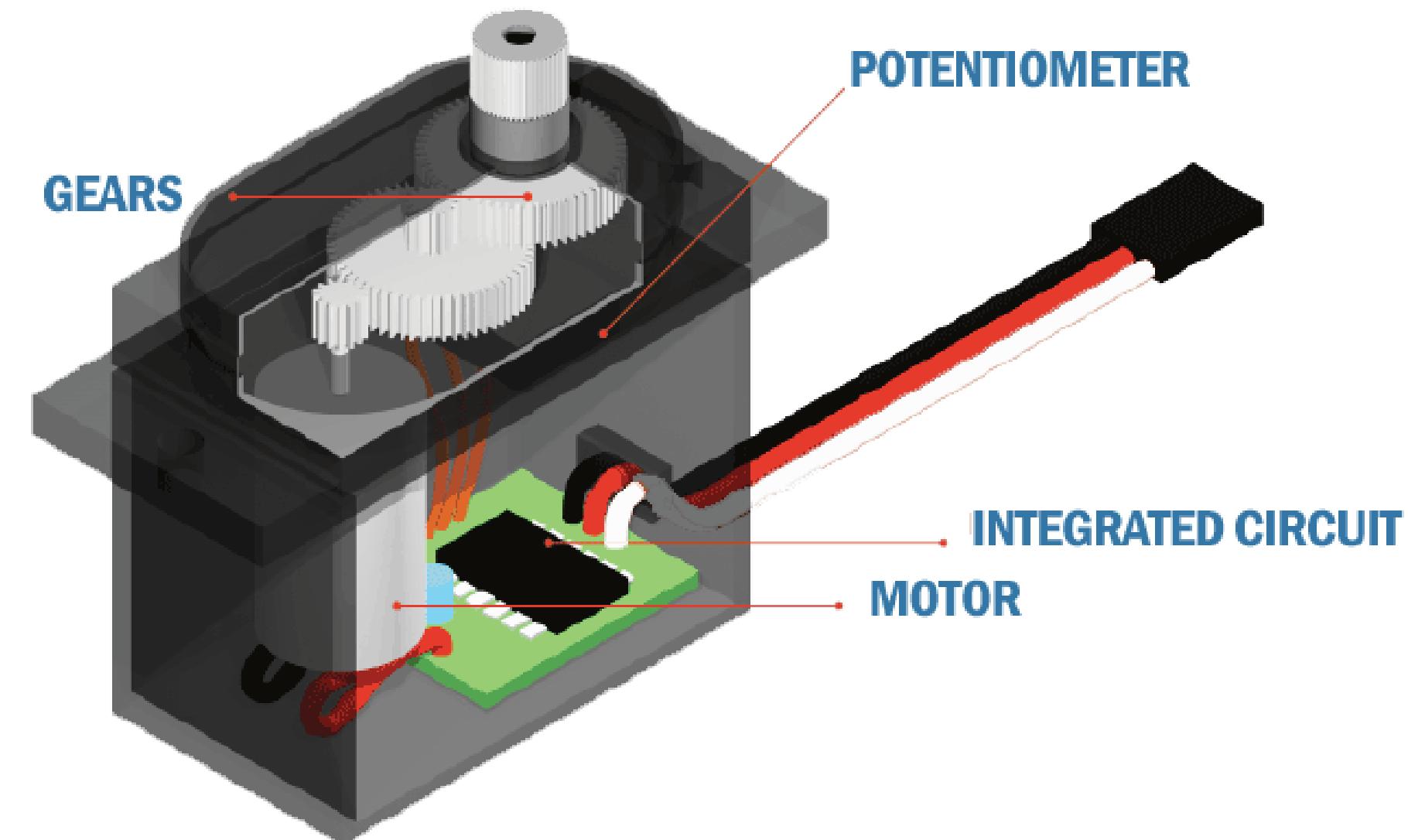
ให้ Motor หมุนตามเข็มทิศ 2 ตัวที่ speed 100 % จากนั้น
ให้ Motor หมุนตามกวนเข็มทิศ 2 ตัวที่ speed 100 %

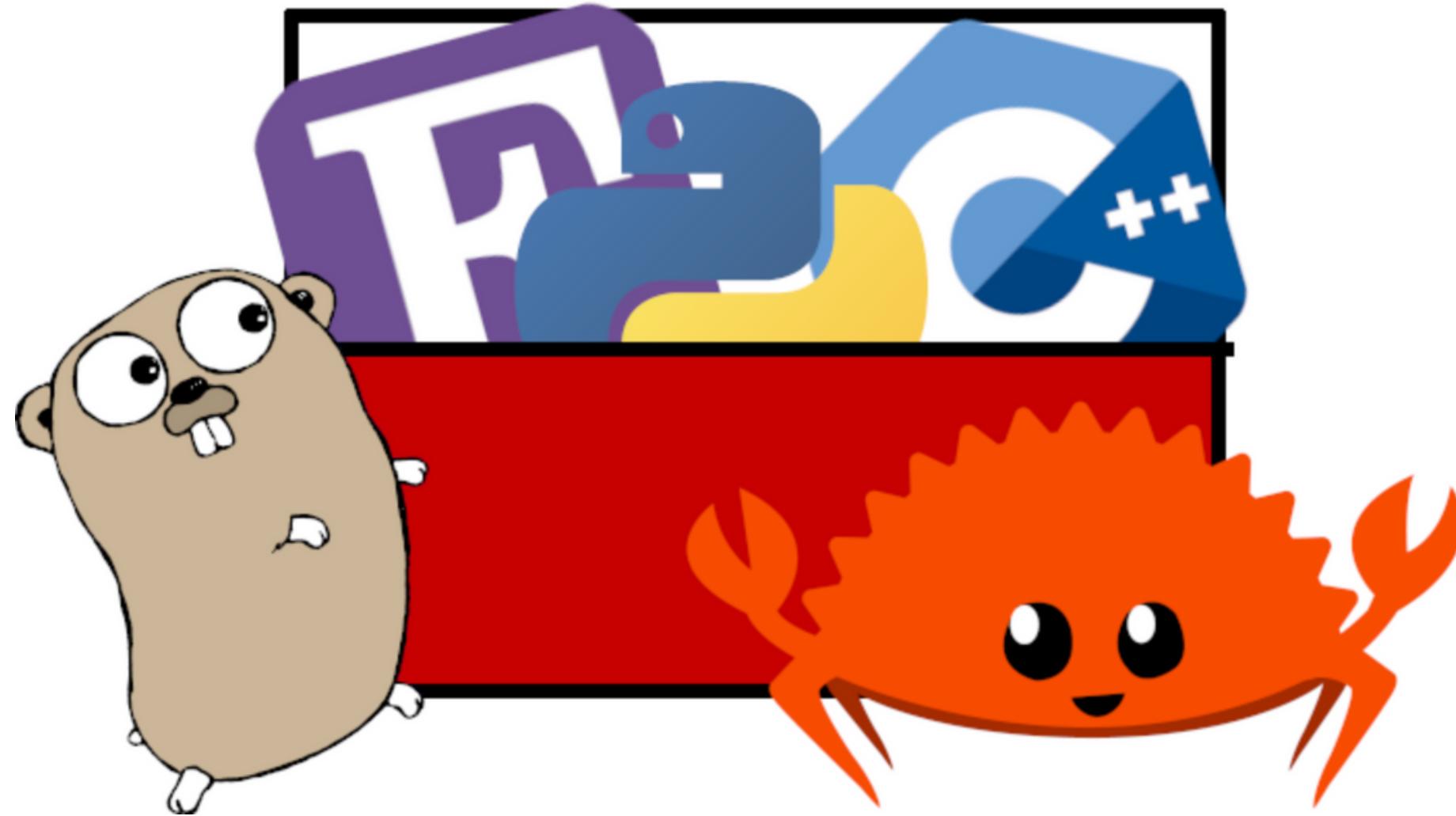


TRY 2 Motor

ให้ Motor เดินไป ข้างหน้า , ถอยหลัง , เลี้ยวซ้าย ,
เลี้ยวขวา ตามลำดับ ครึ่งลั่ 2 วินาที

Output equipment / Servo





TRY 2 Servo
ให้ Servo หมุนไปที่ 0 และ 100 องศา ครึ่งลีบ 5 วินาที



THANK YOU

