

미래형자동차 교육 워크숍

Subject: Arduino

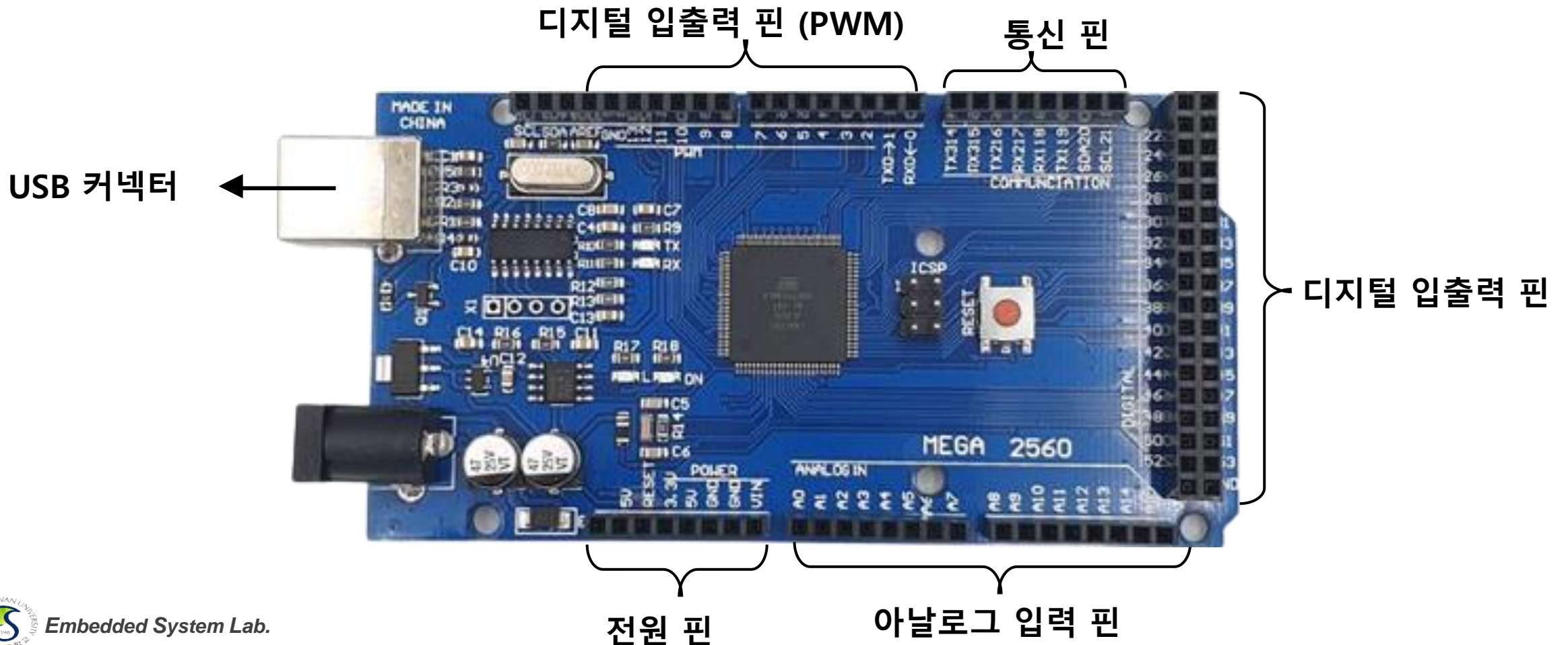
Embedded System Lab.



Introduction

■ Arduino mega 2560

→ 여러 장치 및 센서를 제어하는 초소형 컴퓨터



Introduction

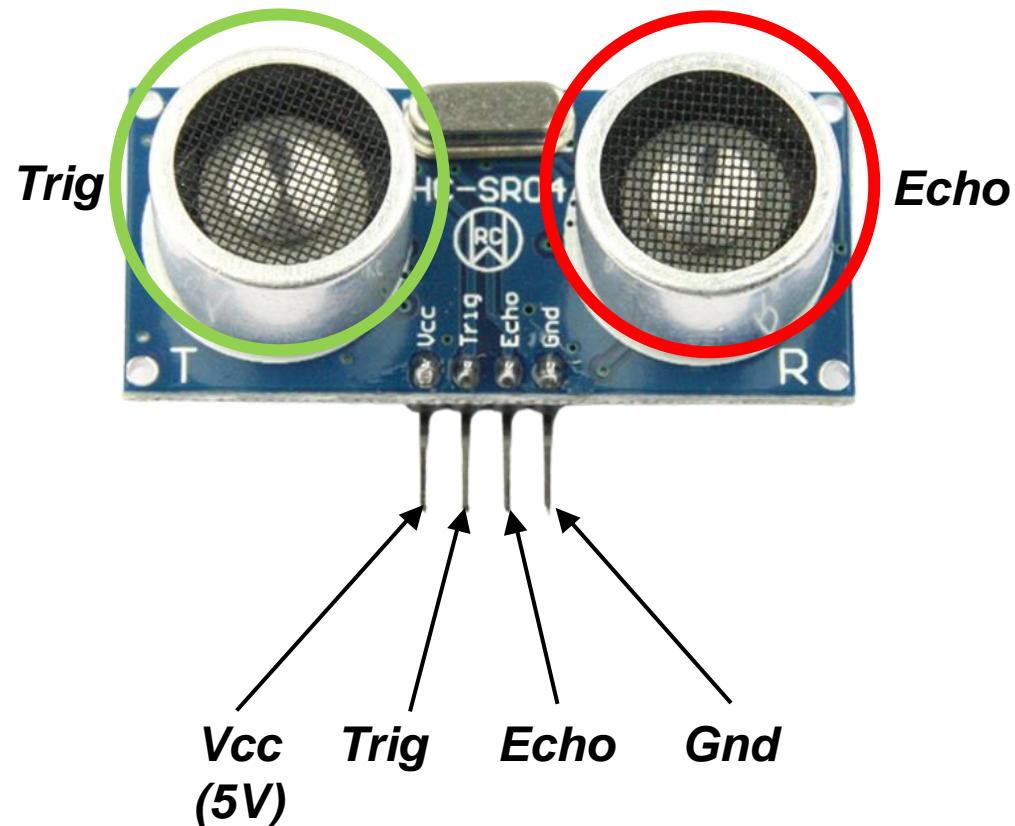
■ Arduino mega 핀 설명

- 전원 핀: 아두이노에 연결되는 장치의 전원 공급에 사용되는 핀
 - (+) 전원: 5V 혹은 3.3V, VIN
 - (-) 전원: GND
- 디지털 입출력 핀: 0(LOW)또는 1(HIGH)의 디지털 신호를 읽거나 보내는 핀
- 아날로그 입력 핀: 0~1023 사이의 정수값으로 아날로그 신호를 읽어오는 핀
- 디지털 입출력(PWM) 핀: 디지털 신호를 아날로그 신호처럼 변환, 0~255 값
- USB 커넥터: PC와의 통신을 위해 사용되는 커넥터, 아두이노 프로그램 업로드

Introduction

■ 초음파 센서

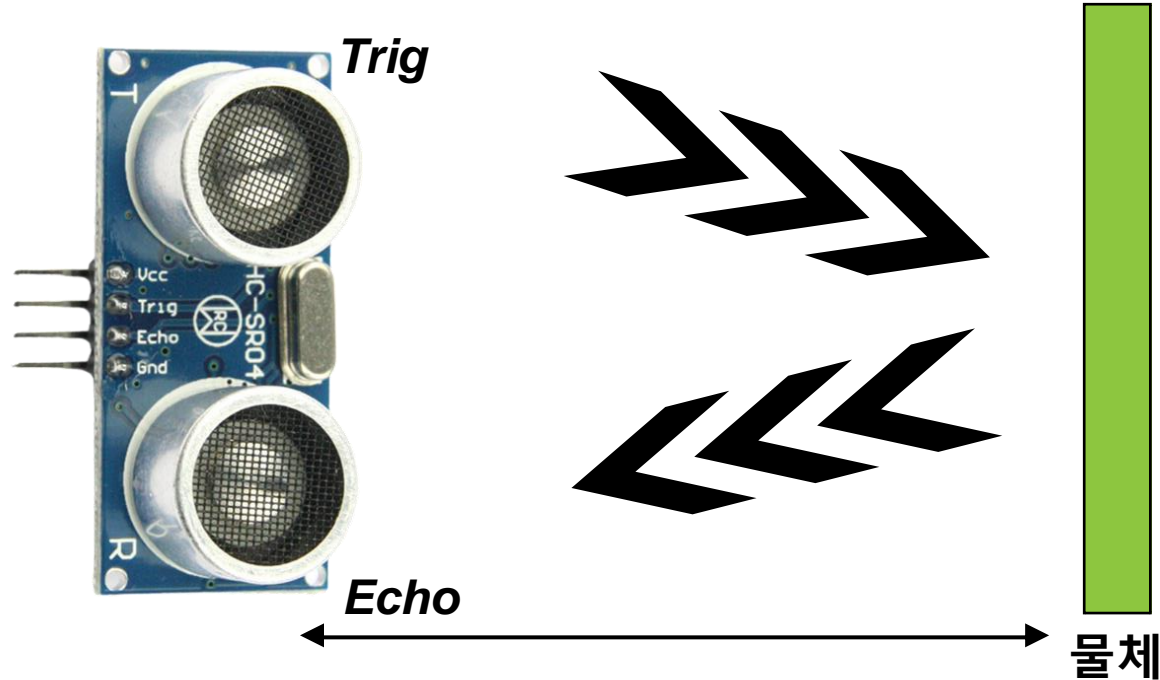
→ 초음파를 이용하여 앞쪽에 위치한 물체와의 거리를 인식하는 센서



Introduction

■ 초음파 센서 설명

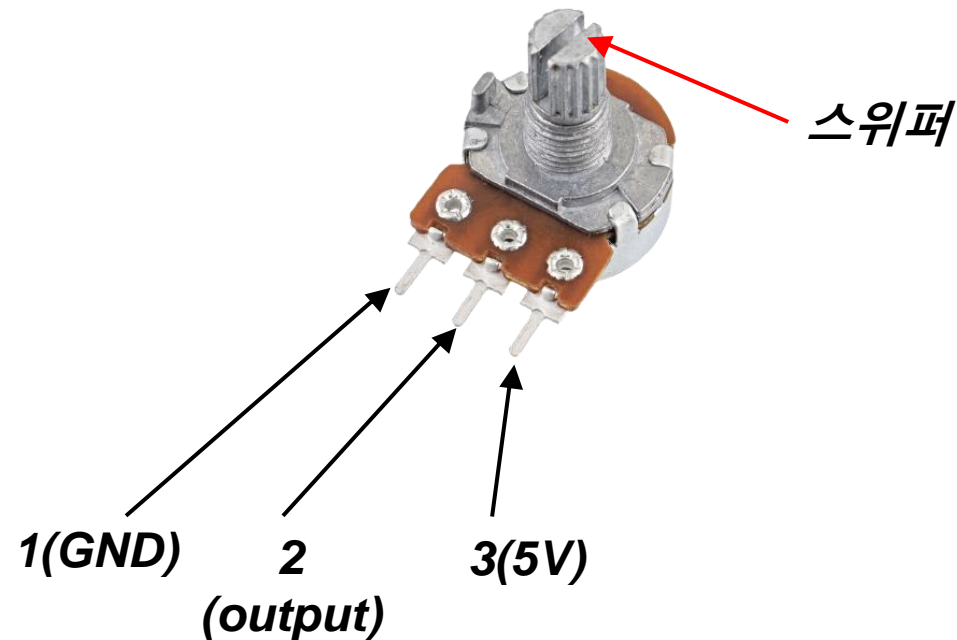
- 송신부(Trig)에서 초음파를 보낸 후 반사되어 돌아오는 초음파를 수신부(Echo)에서 인식
- 초음파 송신 후 수신까지 걸린 시간을 측정하여 거리 계산
$$((340 * \text{duration}) / 1000) / 2$$



Introduction

■ 가변 저항

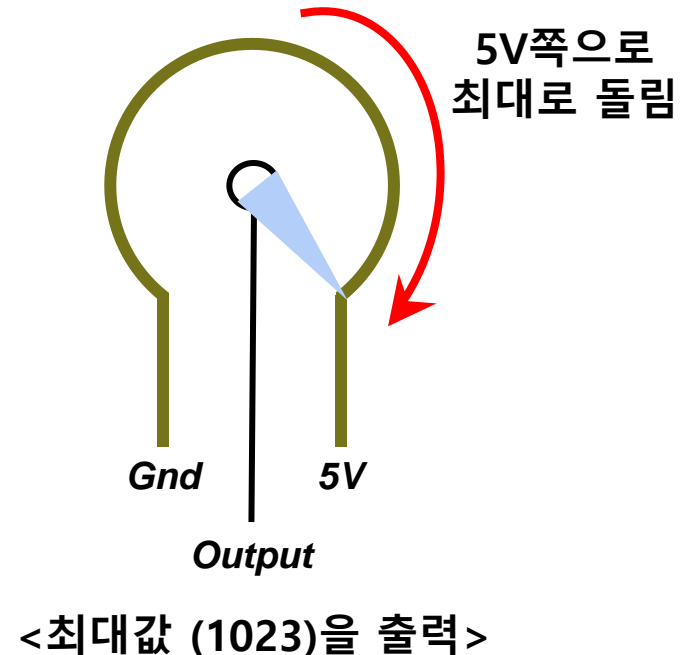
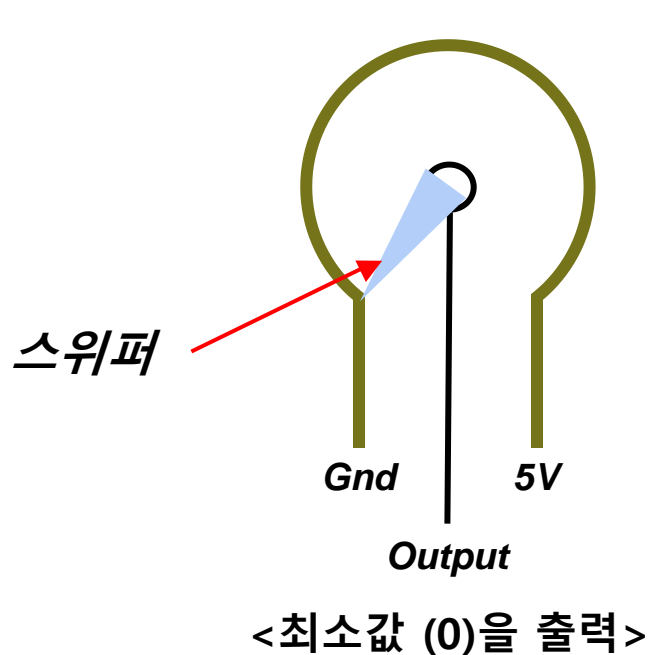
- 회로에 흐르는 전류 값을 변화시킬 수 있는 저항
- 저항의 세기를 아날로그 값으로 Output 핀을 통해 출력



Introduction

■ 가변 저항 설명

- 스위퍼를 움직여 가변 저항의 저항 세기 결정
- 스위퍼를 5V와 연결된 쪽으로 돌리면 최대값, Gnd와 연결된 쪽으로 돌리면 최소값을 출력



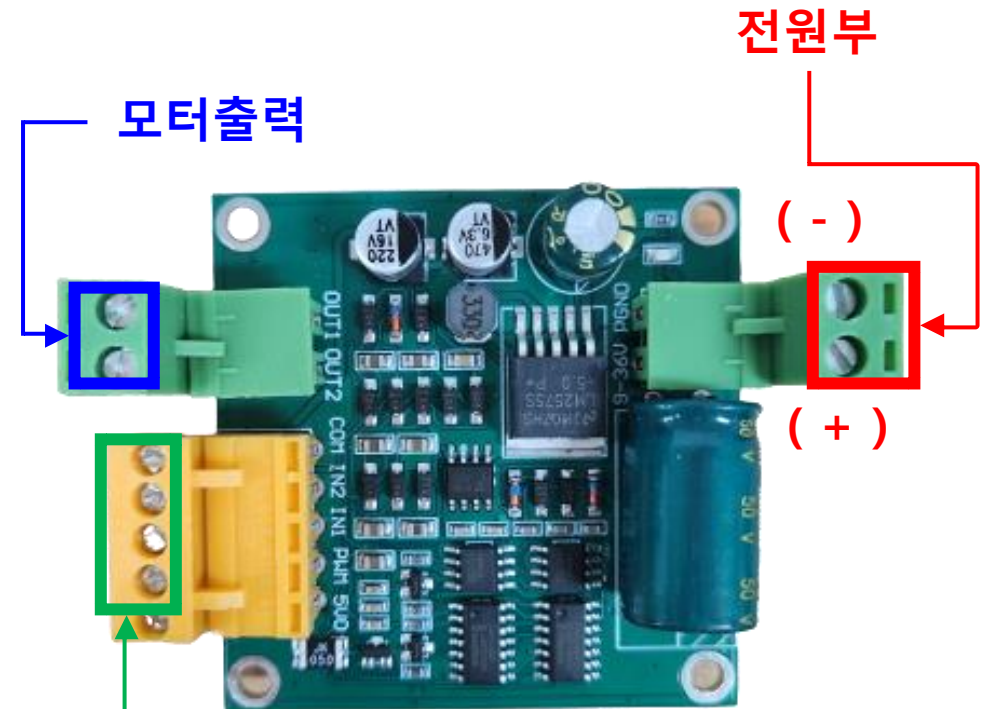
Introduction

■ 기어 모터 & 모터 드라이버

- 높은 토크를 발생하도록 설계된 모터
- 모터 전원 공급을 위한 모터 드라이버



기어 모터 (12V)



모터제어

모터 드라이버

Introduction

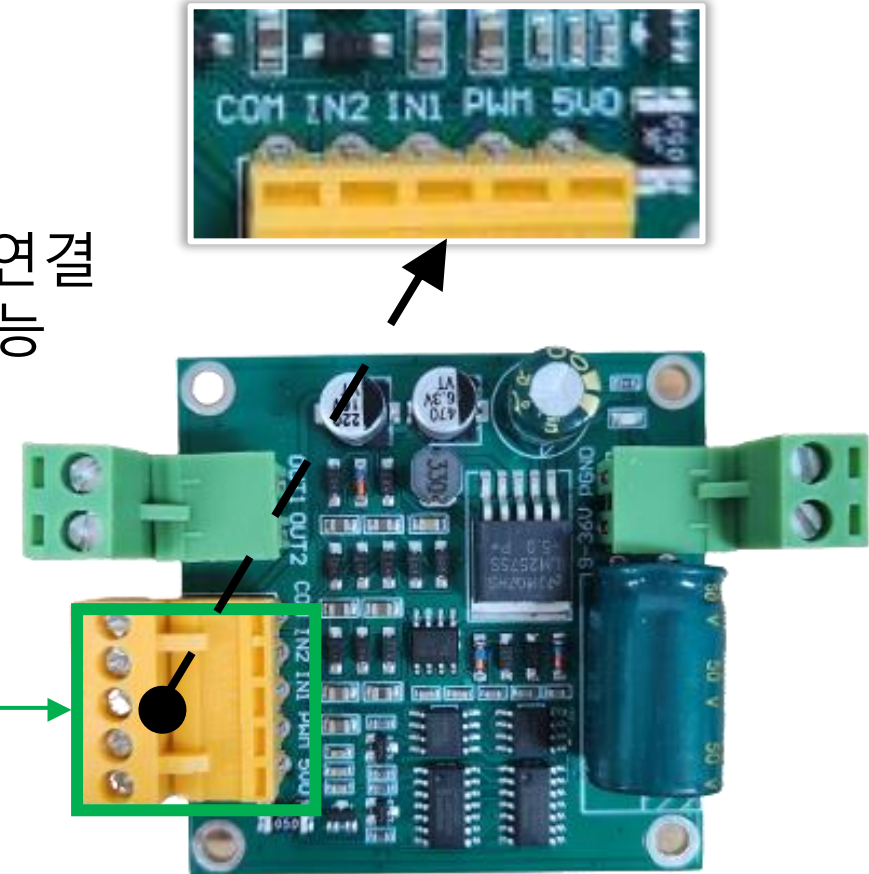
■ 모터 드라이버 제어 설명

- COM 핀: 아두이노의 GND와 연결
- PWM 핀: 아두이노의 5V와 연결
- IN1, IN2핀: 모터의 회전 방향 결정, 디지털핀에 연결
PWM핀에 연결하여 속도 조절도 가능

IN1	IN2	출력
0	0	정지
1	0	정방향 회전
0	1	역방향 회전

모터제어

모터 드라이버



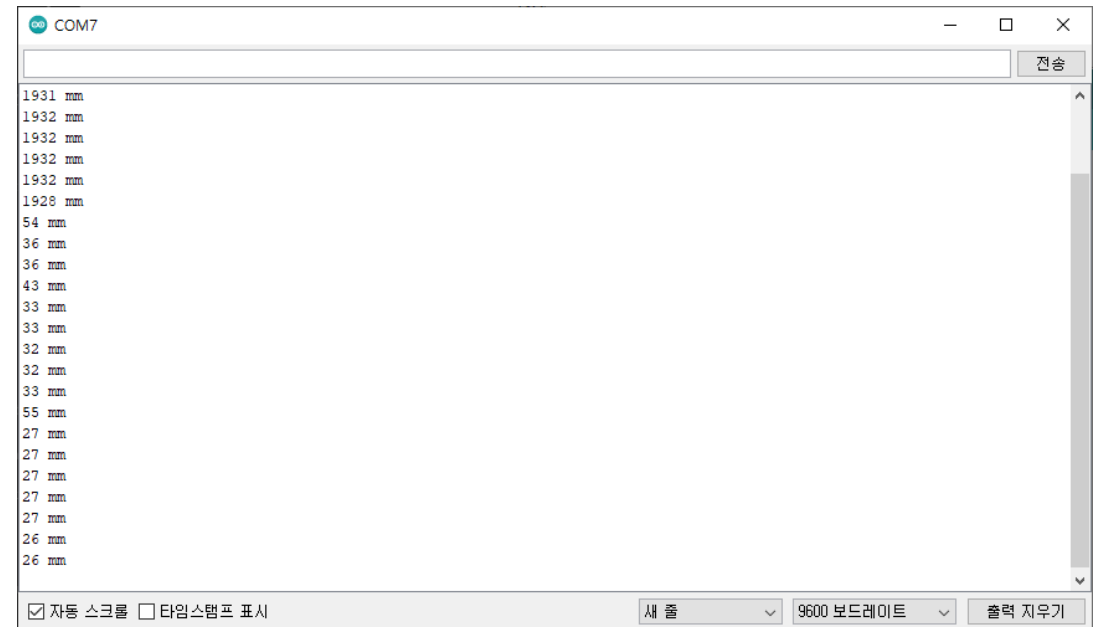
Introduction

■ 시리얼 통신

- 아두이노가 PC와 통신하는 방식
- 아두이노로 데이터를 전송하거나, 데이터를 받는 것이 가능



PC(노트북)과 USB 연결



<시리얼 모니터 화면>

Exercise

Embedded System Lab.



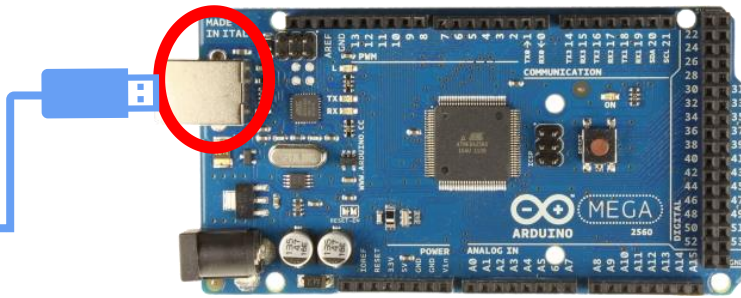
1. Hardware Setting

■ Arduino Hardware Setting

- 소프트웨어 세팅 전에 선행되어야함
- USB 케이블을 노트북(PC) USB 포트에 직접 연결



Notebook(PC)



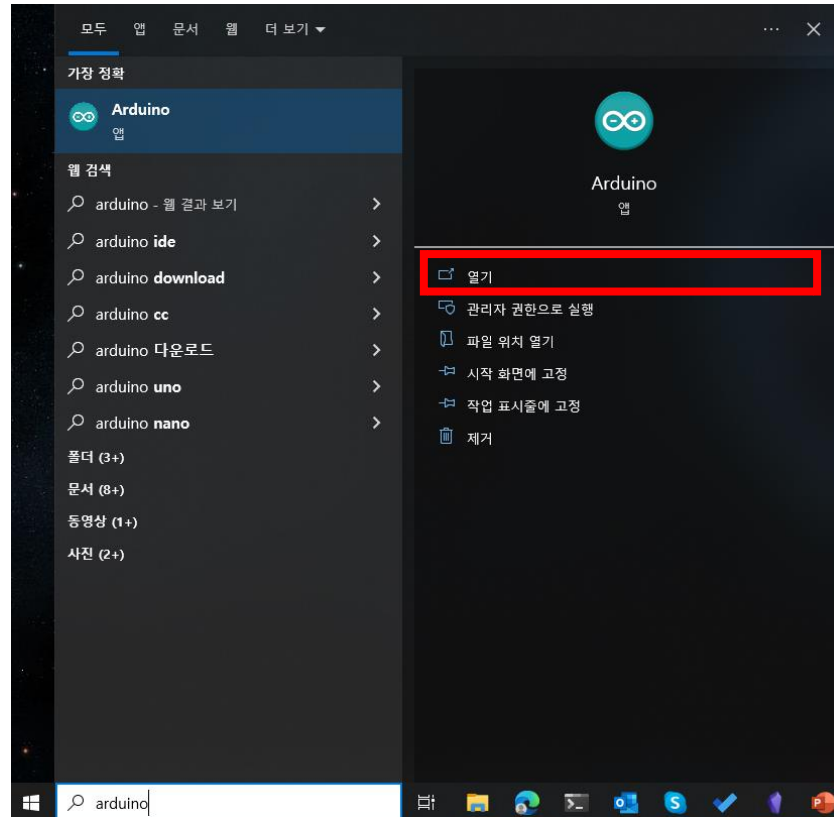
Arduino 보드



2. Software Setting

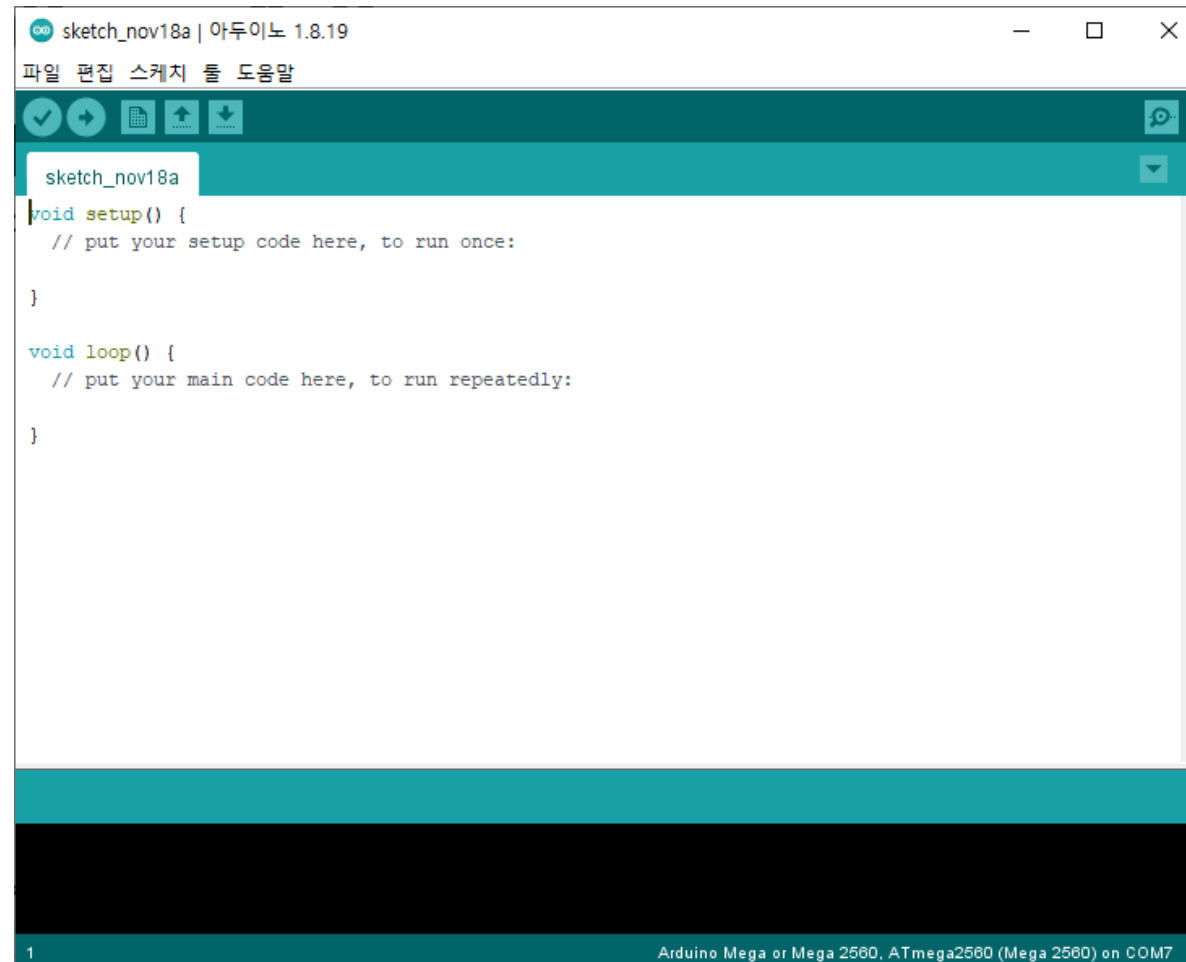
■ 아두이노 앱 실행

- 키보드에 "Win키+q" 입력 후, "arduino" 입력 후 "열기" 클릭
- 앱 설치는 "환경설정" 강의자료 참고!



2. Software Setting

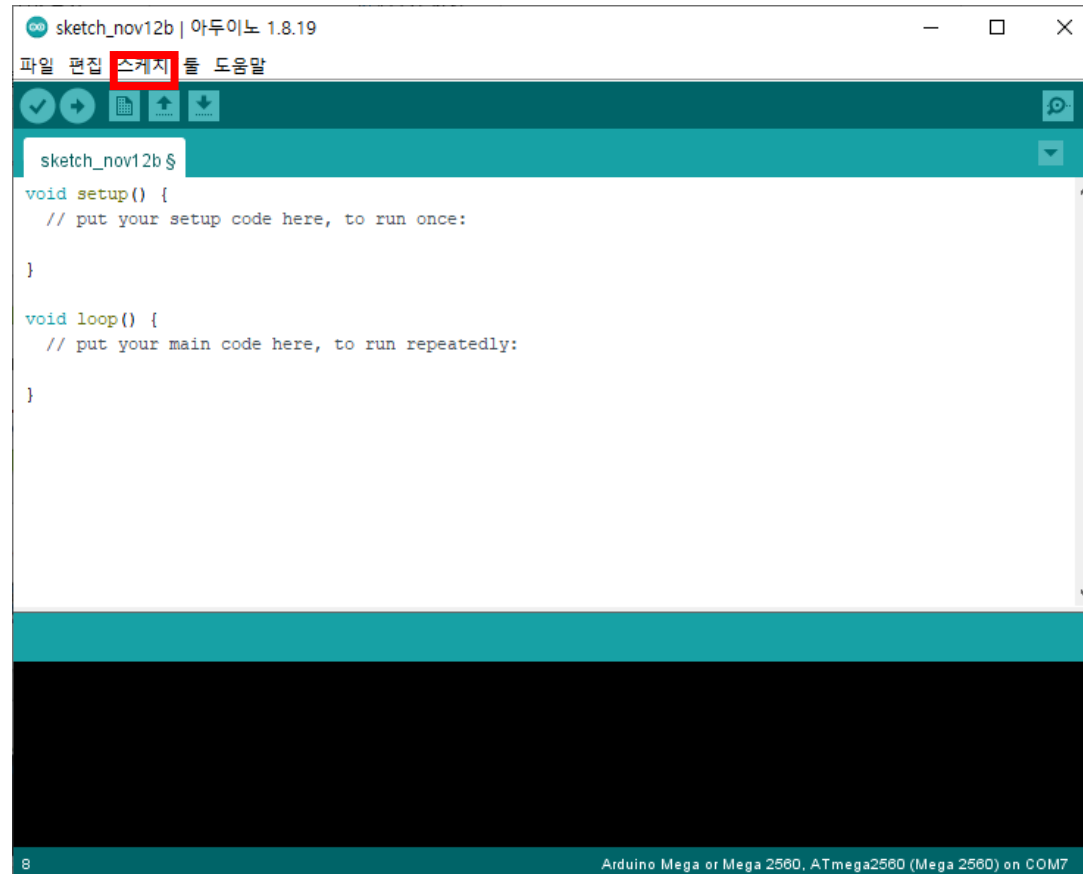
■ 아두이노 앱 초기화면



2. Software Setting

■ 라이브러리 추가

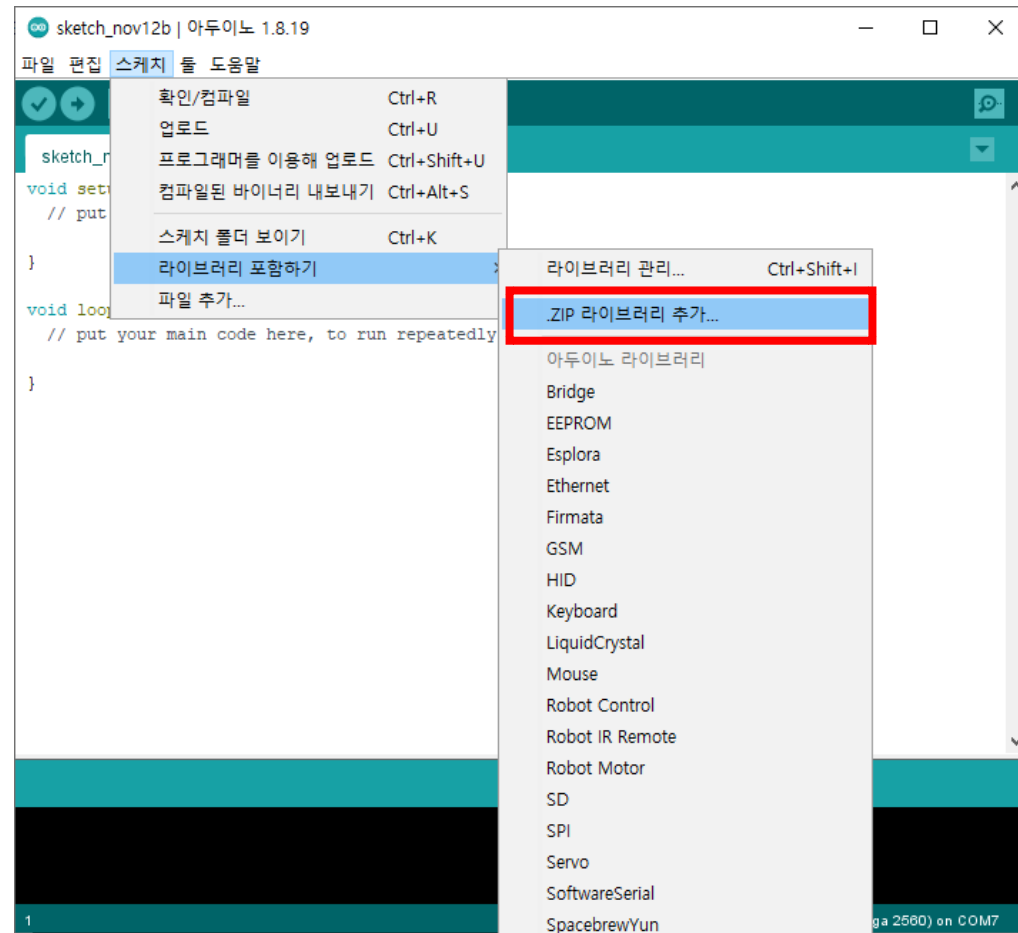
→ "스케치" 클릭



2. Software Setting

■ 라이브러리 추가

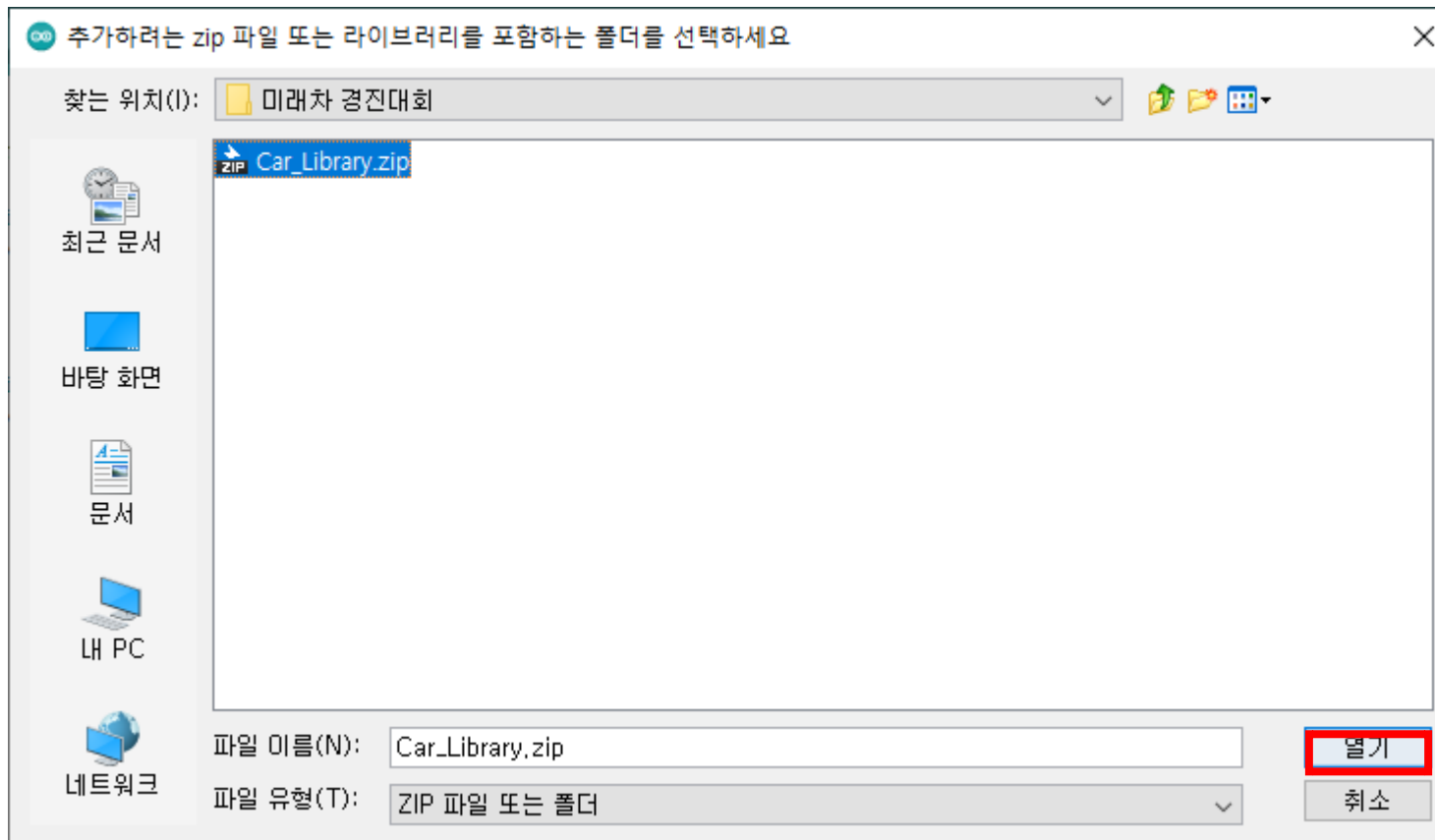
→ “라이브러리 포함하기” -> .ZIP 라이브러리 추가... 클릭



2. Software Setting

■ 라이브러리 추가

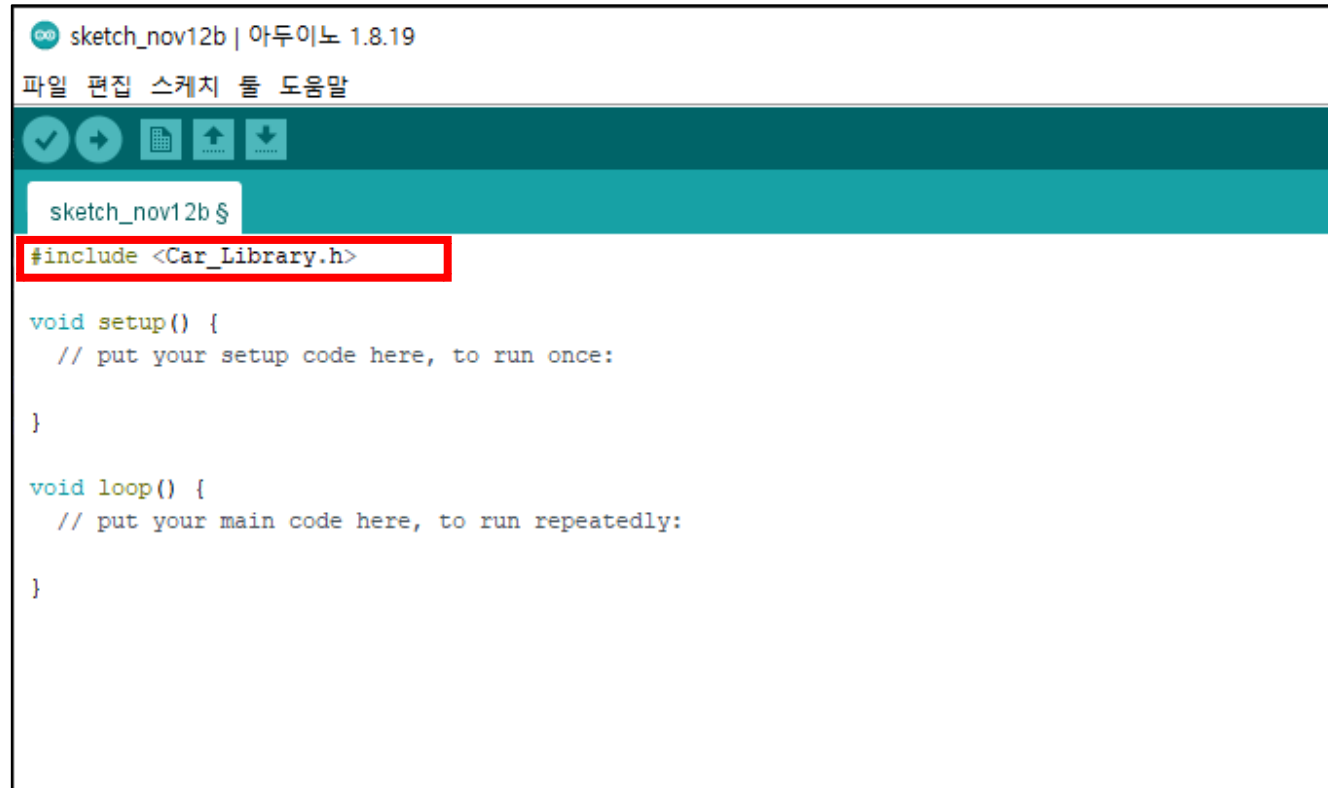
→ 해당 창이 뜨면 Github에서 다운로드 받은 "Car_Library" 압축 파일 선택 후 "열기" 클릭



2. Software Setting

■ 라이브러리 추가

→ 소스코드 상단에 "#include <Car_Library.h>" 입력



The screenshot shows the Arduino IDE interface for a sketch named 'sketch_nov12b'. The top menu bar includes '파일' (File), '편집' (Edit), '스케치' (Sketch), '툴' (Tools), and '도움말' (Help). Below the menu bar is a toolbar with icons for saving, running, and uploading. The main text area shows the following code:

```
sketch_nov12b $
#include <Car_Library.h>

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

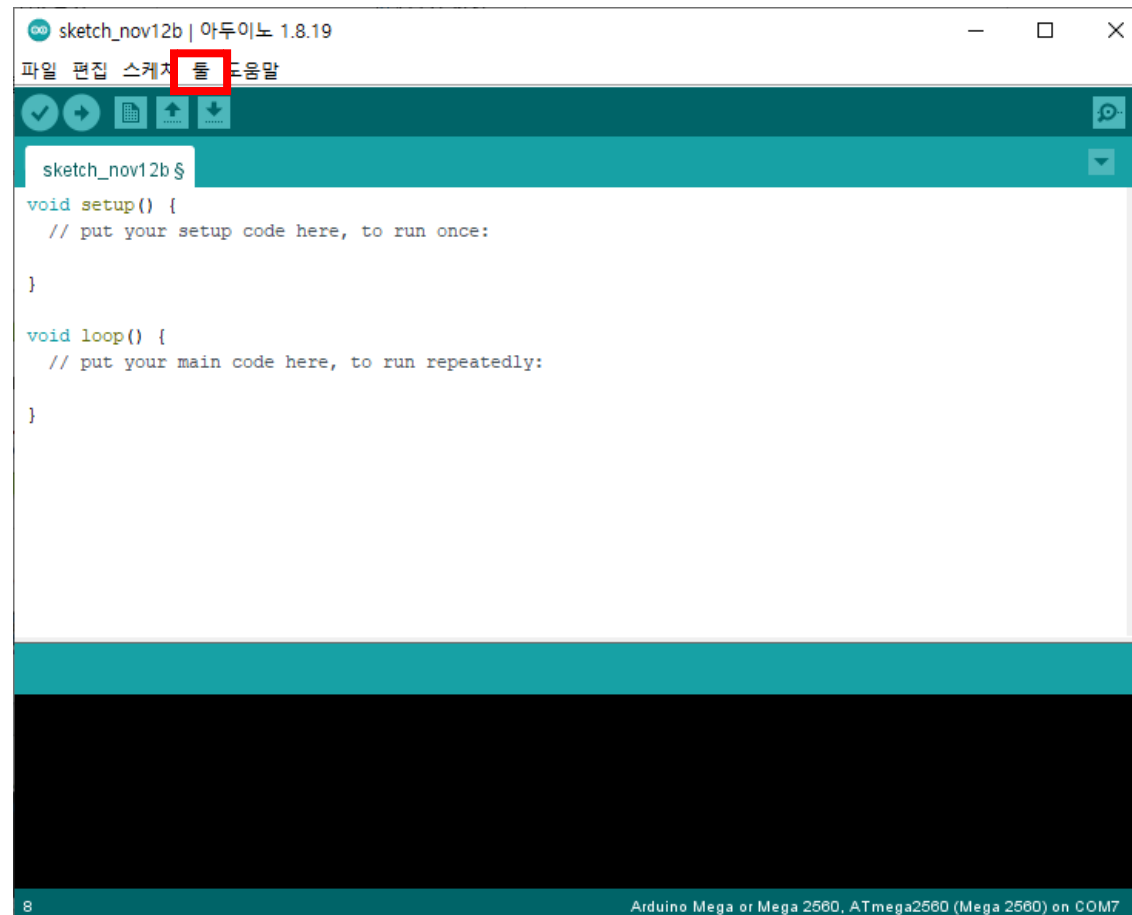
void loop() {
  // put your main code here, to run repeatedly:
}
```

The line `#include <Car_Library.h>` is highlighted with a red rectangular box, indicating the step to add the library header.

2. Software Setting

■ 보드 설정

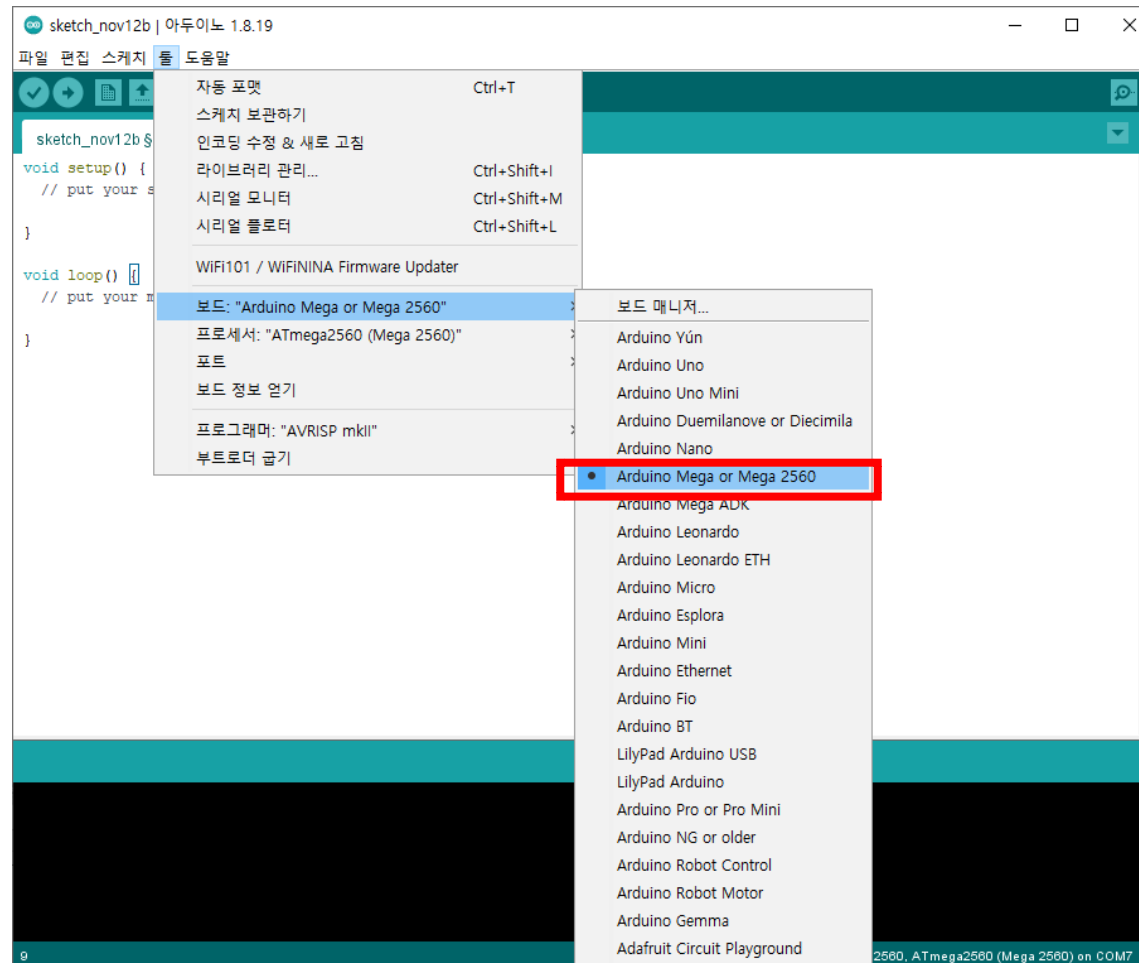
→ 아두이노 앱 실행 후, "툴" 클릭



2. Software Setting

■ 보드 설정

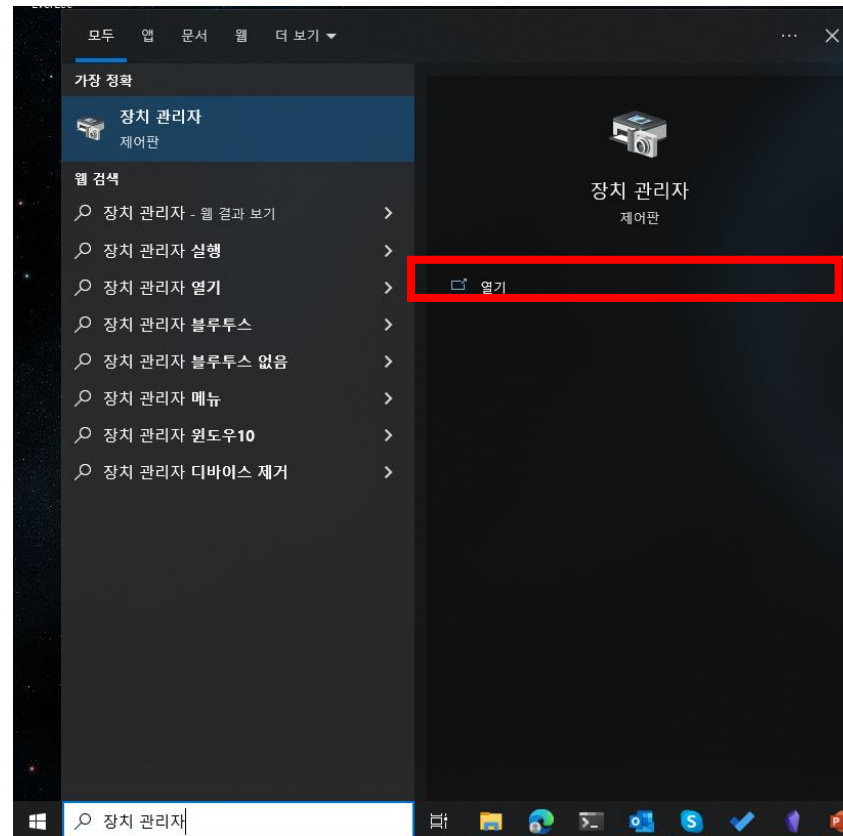
→ 툴 -> 보드 -> "Arduino Mega or Mega 2560" 선택



2. Software Setting

■ 시리얼 연결

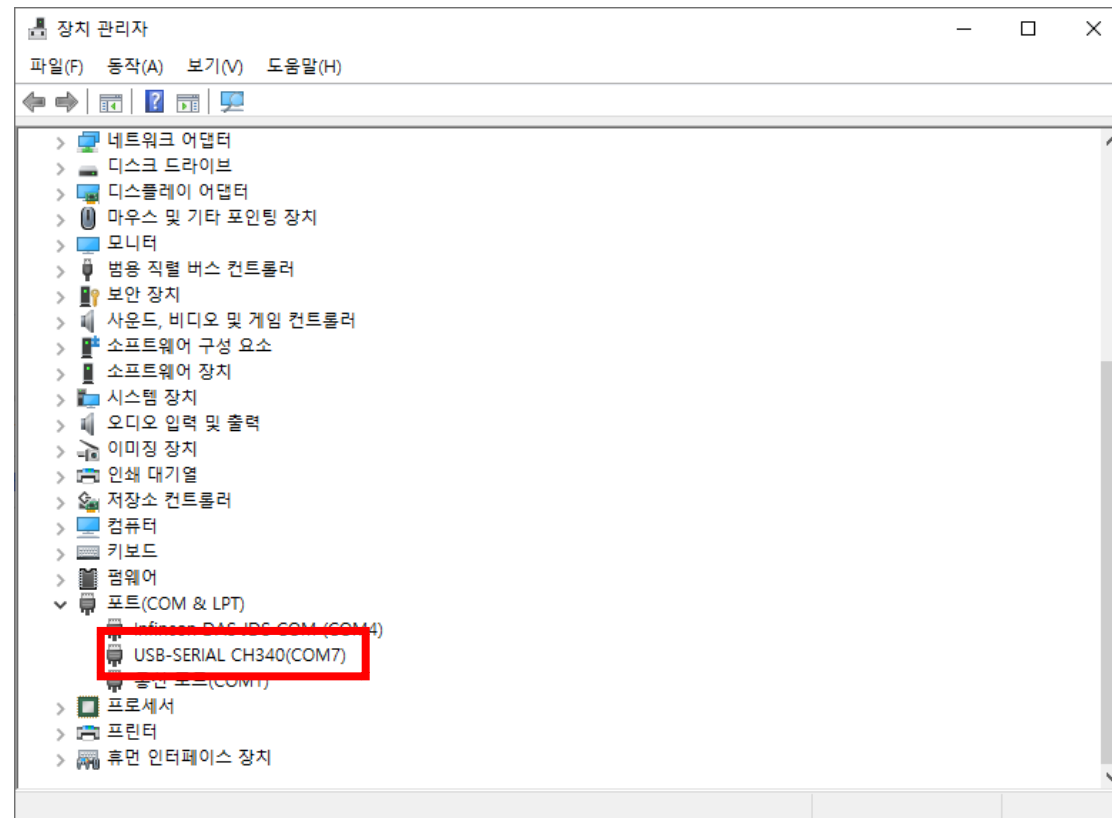
- 키보드에 "Win키+q" 입력 후, "장치 관리자" 입력 후 "열기" 클릭
- 아두이노와 PC가 반드시 연결되어 있어야함



2. Software Setting

■ 시리얼 포트 번호 확인

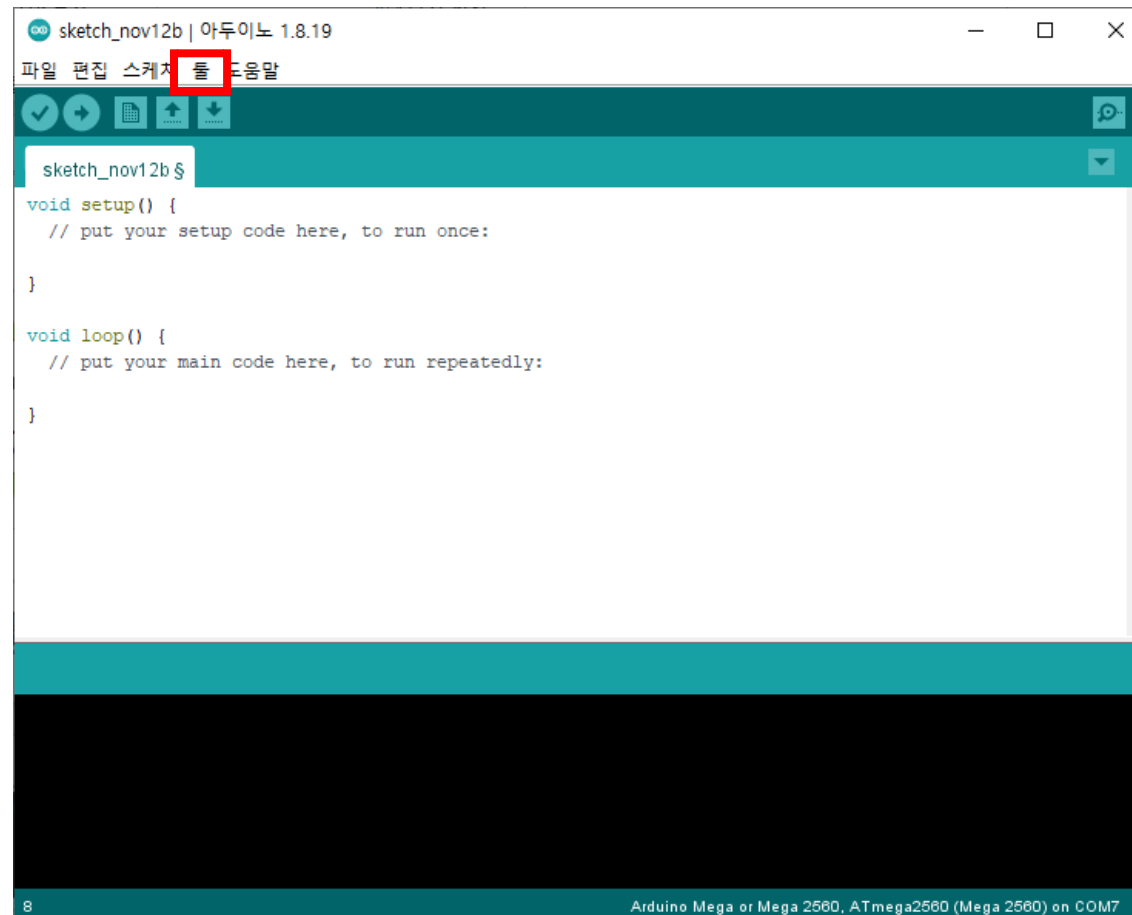
- 장치 관리자 -> 포트 (COM & LPT) -> USB-SERIAL 포트 번호 확인 (COM*)
- 해당 번호는 USB 케이블을 처음 연결한 포트가 아닌 다른 포트에 연결하면 변경될 수 있음



2. Software Setting

■ 시리얼 통신 연결

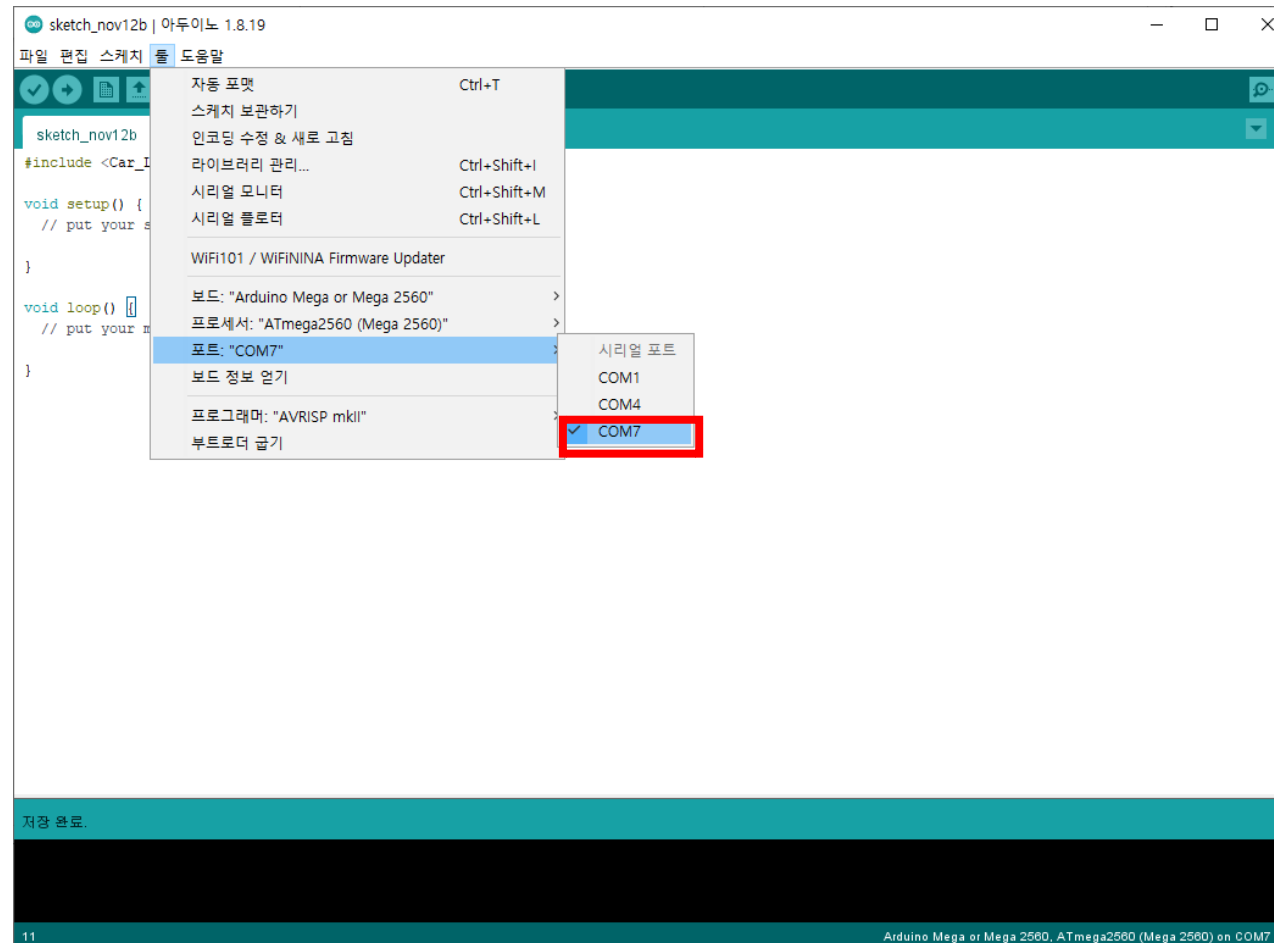
→ 아두이노 앱 실행 후, "툴" 클릭



2. Software Setting

■ 시리얼 포트 설정

→ 툴 -> 포트 -> 이전에 확인 포트 번호 선택

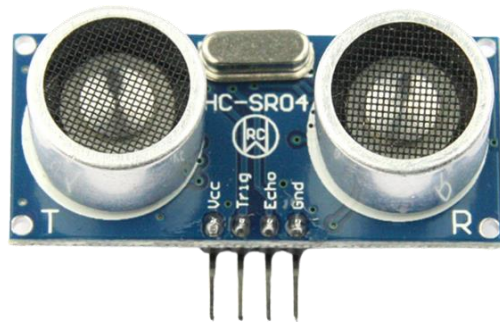


Exercise 1

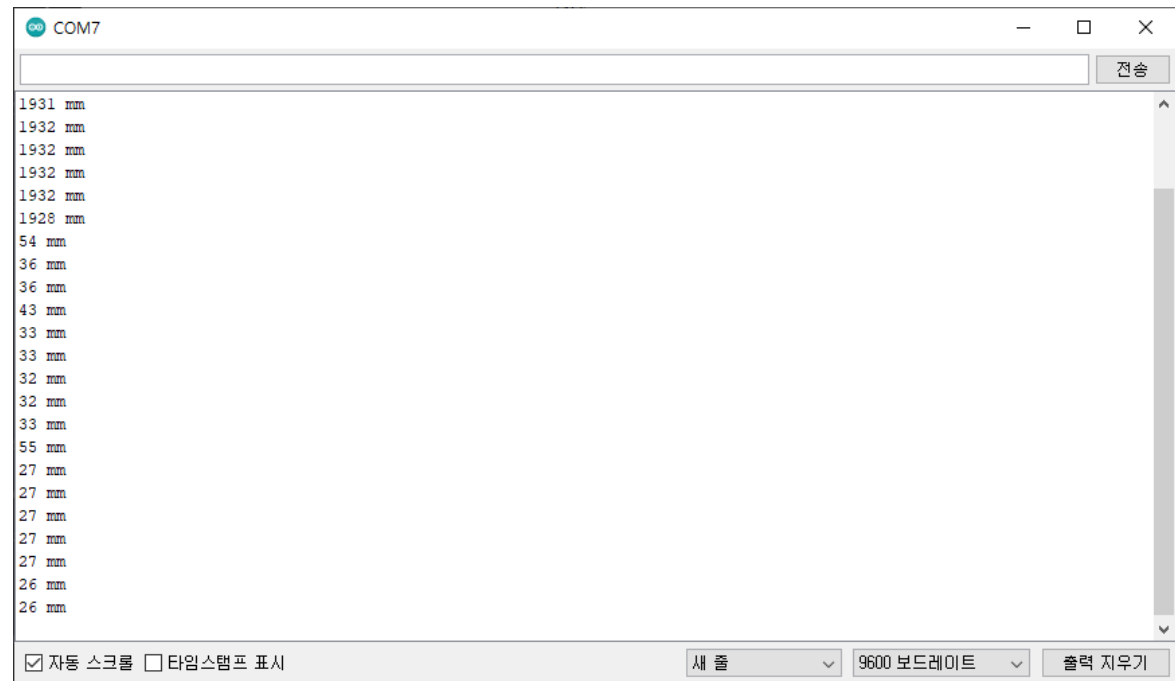
Exercise 1

■ 초음파 센서로 거리 측정하기

- 초음파 센서를 이용하여 앞쪽에 위치한 물체와의 거리를 측정
- 측정한 거리를 시리얼 통신으로 출력



초음파 센서

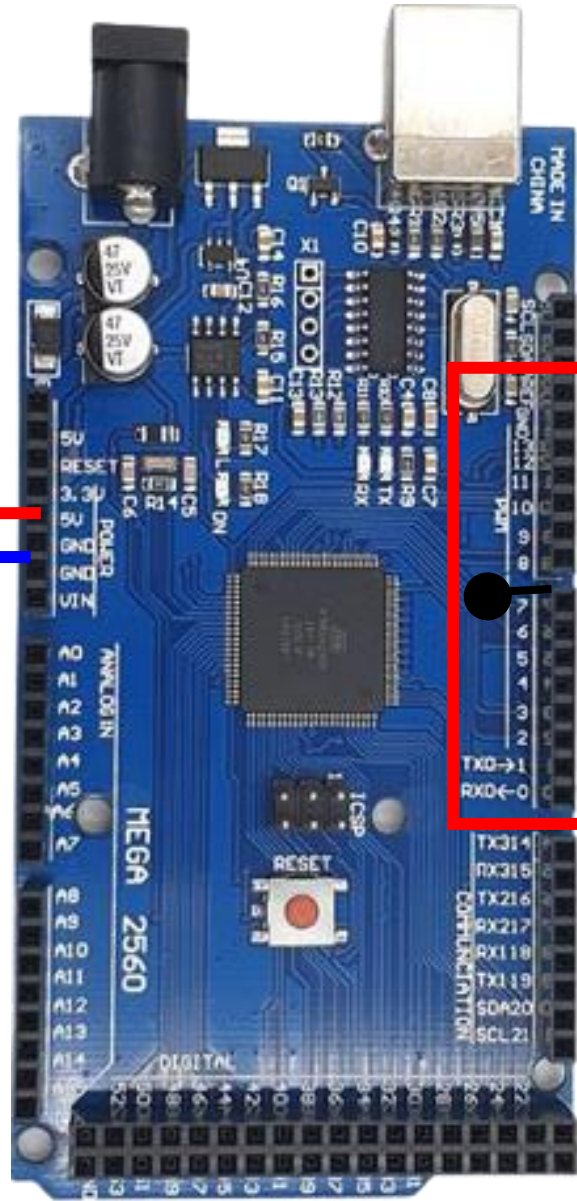


결과 화면

Exercise 1

■ 선 연결

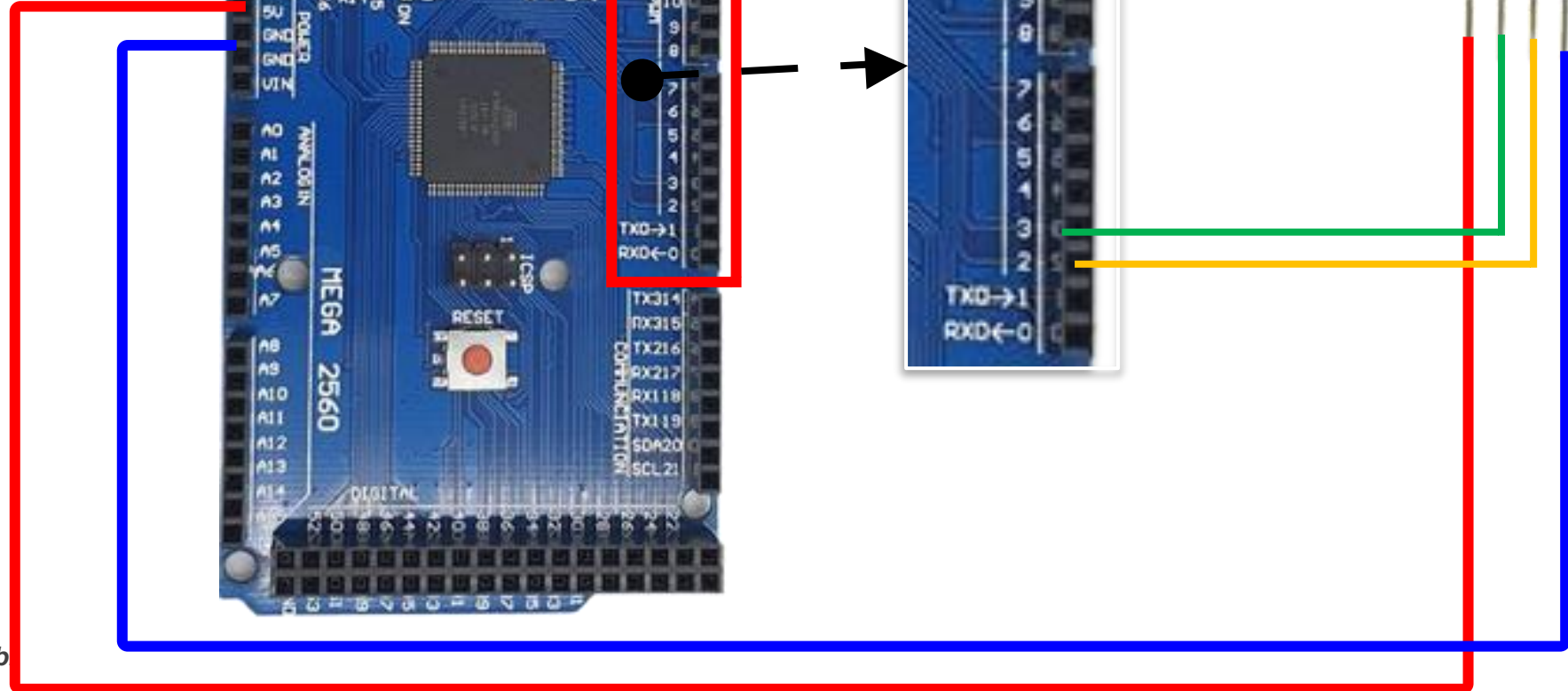
Arduino 보드



PWM 핀



초음파 센서



Exercise 1

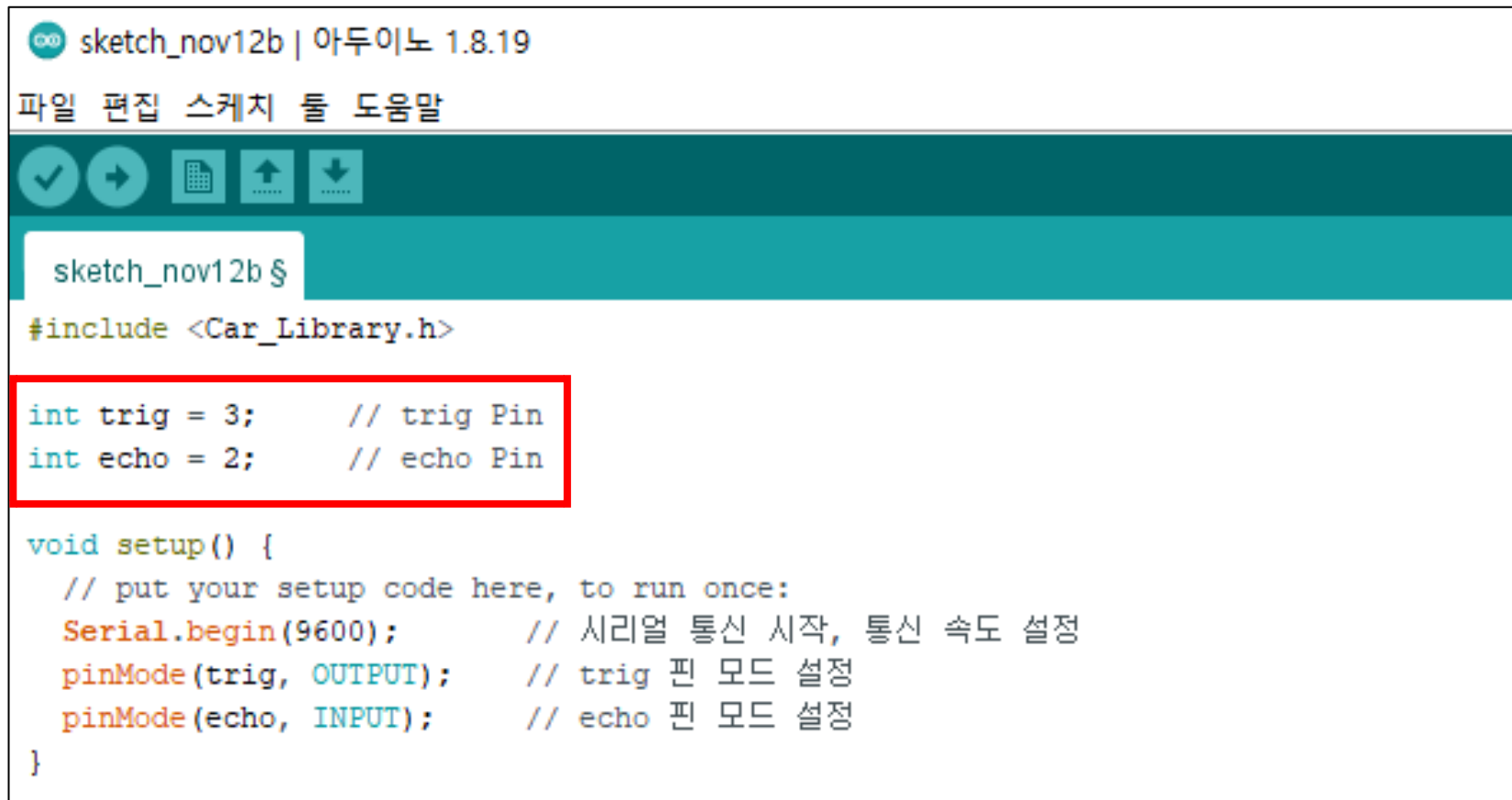
■ 선 연결 (Cont'd)

- 초음파 센서 **VCC** – 아두이노 **5V**에 연결
- 초음파 센서 **GND** – 아두이노 **GND**에 연결
- 초음파 센서 **Trig** – 아두이노 **PWM 3번** 핀에 연결
- 초음파 센서 **Echo** – 아두이노 **PWM 2번** 핀에 연결

Exercise 1

■ 핀 번호 변수 선언

→ 초음파 센서의 trig, echo와 연결된 아두이노 핀 번호 변수 선언



```
sketch_nov12b | 아두이노 1.8.19
파일 편집 스케치 툴 도움말

sketch_nov12b $
#include <Car_Library.h>

int trig = 3;    // trig Pin
int echo = 2;    // echo Pin

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);    // 시리얼 통신 시작, 통신 속도 설정
  pinMode(trig, OUTPUT); // trig 핀 모드 설정
  pinMode(echo, INPUT);  // echo 핀 모드 설정
}
```

Exercise 1

■ 시리얼 통신 및 핀 모드 설정

→ 시리얼 통신 시작 및 연결된 핀의 모드 설정

```
sketch_nov12b | 아두이노 1.8.19
파일 편집 스케치 툴 도움말
✓ → 📄 ⬆ ⬇
sketch_nov12b $
#include <Car_Library.h>

int trig = 3;    // trig Pin
int echo = 2;    // echo Pin

void setup() {
    // put your setup code here, to run once:
    Serial.begin(9600);    // 시리얼 통신 시작, 통신 속도 설정
    pinMode(trig, OUTPUT); // trig 핀 모드 설정
    pinMode(echo, INPUT);  // echo 핀 모드 설정
}
```

Exercise 1

■ 함수 설명

- **Serial.begin(통신속도)** `Serial.begin(9600);` // 시리얼 통신 시작, 통신 속도 설정
→ 입력한 통신속도로 시리얼 통신 시작
- **pinMode(핀번호, mode)** `pinMode(trig, OUTPUT);` // trig 핀 모드 설정
 `pinMode(echo, INPUT);` // echo 핀 모드 설정
→ 입력받은 핀의 모드를 설정
→ OUTPUT(출력) 또는 INPUT(입력)으로 설정 가능
→ 해당 예제에서 trig는 초음파를 보내기 때문에 OUTPUT(출력),
echo는 반사된 초음파를 받기 때문에 INPUT(입력)으로 설정

Exercise 1

■ 변수 선언 및 거리 측정 함수 실행

→ 거리 값을 저장할 변수 선언, 초음파센서 거리 측정 함수 실행

```
void loop() {  
    long distance;      // 거리 값 저장할 변수 선언  
  
    distance = ultrasonic_distance(trig, echo);  
  
    // Serial 모니터로 출력  
    Serial.print(distance);  
    Serial.println(" mm");  
  
    // 1초마다 출력  
    delay(1000);  
}
```


Exercise 1

■ “ultrasonic_distance()” 설명

→ 초음파 송신 후, 수신까지 걸린 시간 측정하고 거리를 계산하는 함수

①

```
float ultrasonic_distance(int trigPin, int echoPin)
```

```
distance = ((float)(340 * duration) / 1000) / 2;
```

```
return distance;
```

②

① Input: 연결한 trig핀과 echo 핀의 번호 입력

② Output: 계산한 거리를 출력 (mm 단위)

Exercise 1

■ 측정 값 출력

→ 측정한 거리 값을 시리얼 모니터로 출력

```
void loop() {  
    long distance;      // 거리 값 저장할 변수 선언  
  
    distance = ultrasonic_distance(trig, echo);  
  
    // Serial 모니터로 출력  
    Serial.print(distance);  
    Serial.println(" mm");  
  
    // 1초마다 출력  
    delay(1000);  
}
```

Exercise 1

■ 함수 설명

- **Serial.print(데이터)**

```
// Serial 모니터로 출력  
Serial.print(distance);  
Serial.println(" mm");
```

- 시리얼 모니터를 통해 데이터를 출력
- 문자열을 출력하고 싶으면 큰따옴표("") 사용
- Serial.println()은 데이터 출력 후 줄 바꿈을 해주는 함수

- **Delay(시간)**

```
// 1초마다 출력  
delay(1000);
```

- 입력 받은 시간만큼 아두이노의 동작을 멈추는 함수
- 단위는 밀리세컨드, ms (천분의 1초)

Exercise 1

■ 컴파일 및 업로드

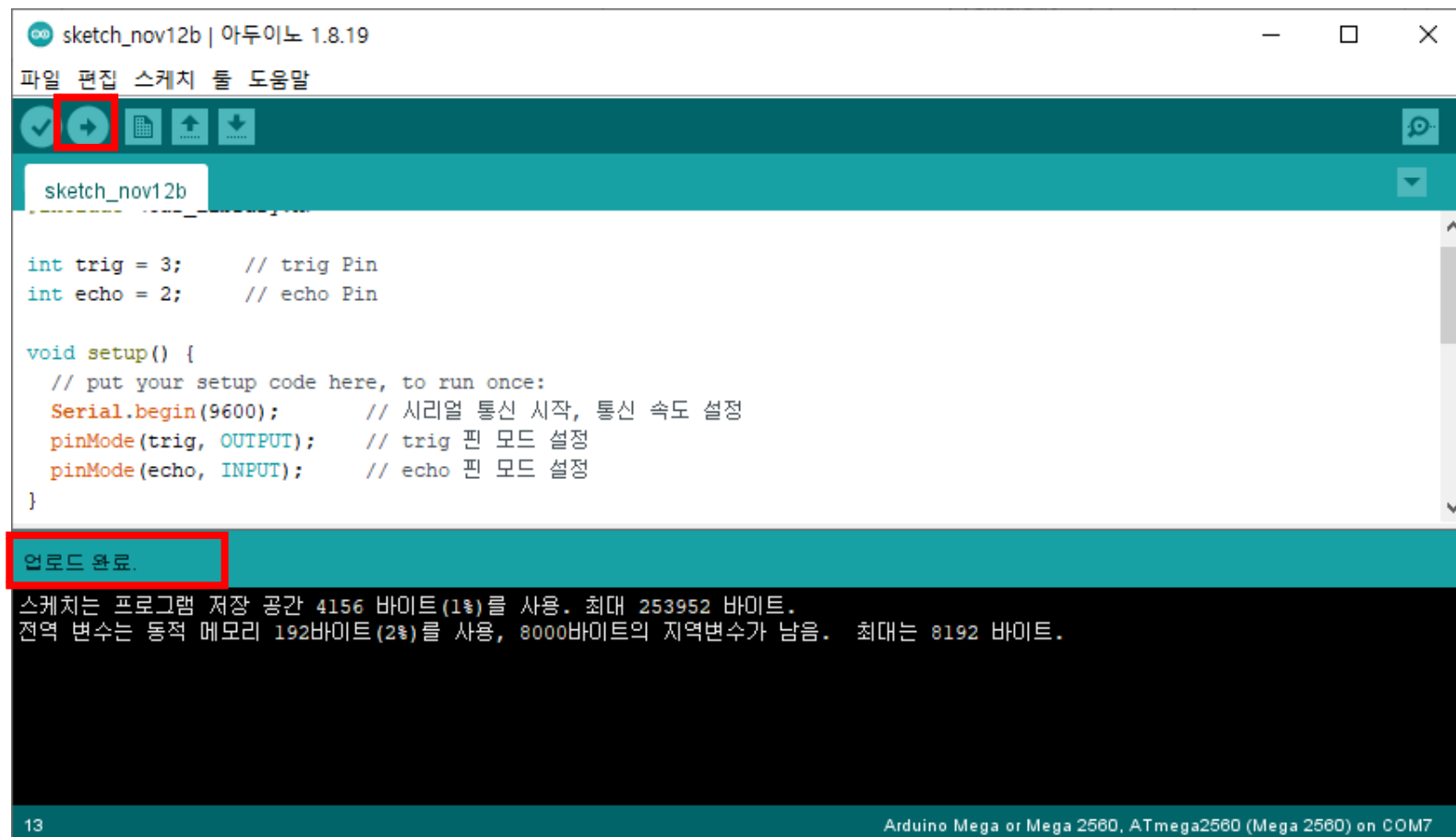
- 소스 코드 작성 완료 후, 확인 버튼 클릭하여 컴파일 및 컴파일 완료 확인
- 저장하라는 창이 뜰 경우 저장하거나 취소해도 상관없음



Exercise 1

■ 컴파일 및 업로드

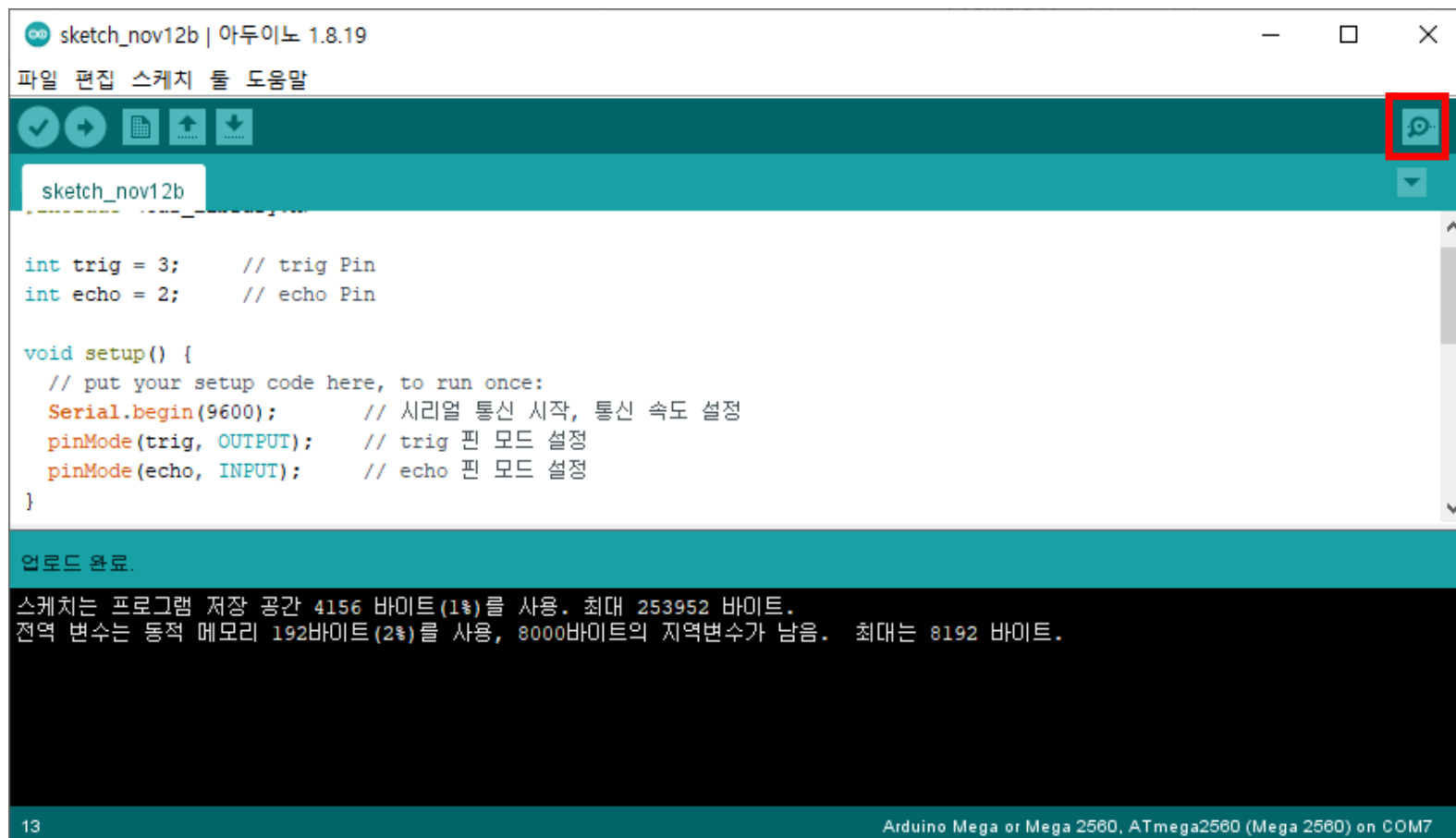
→ 업로드 버튼 클릭하여 아두이노 보드에 업로드 및 업로드 완료 확인



Exercise 1

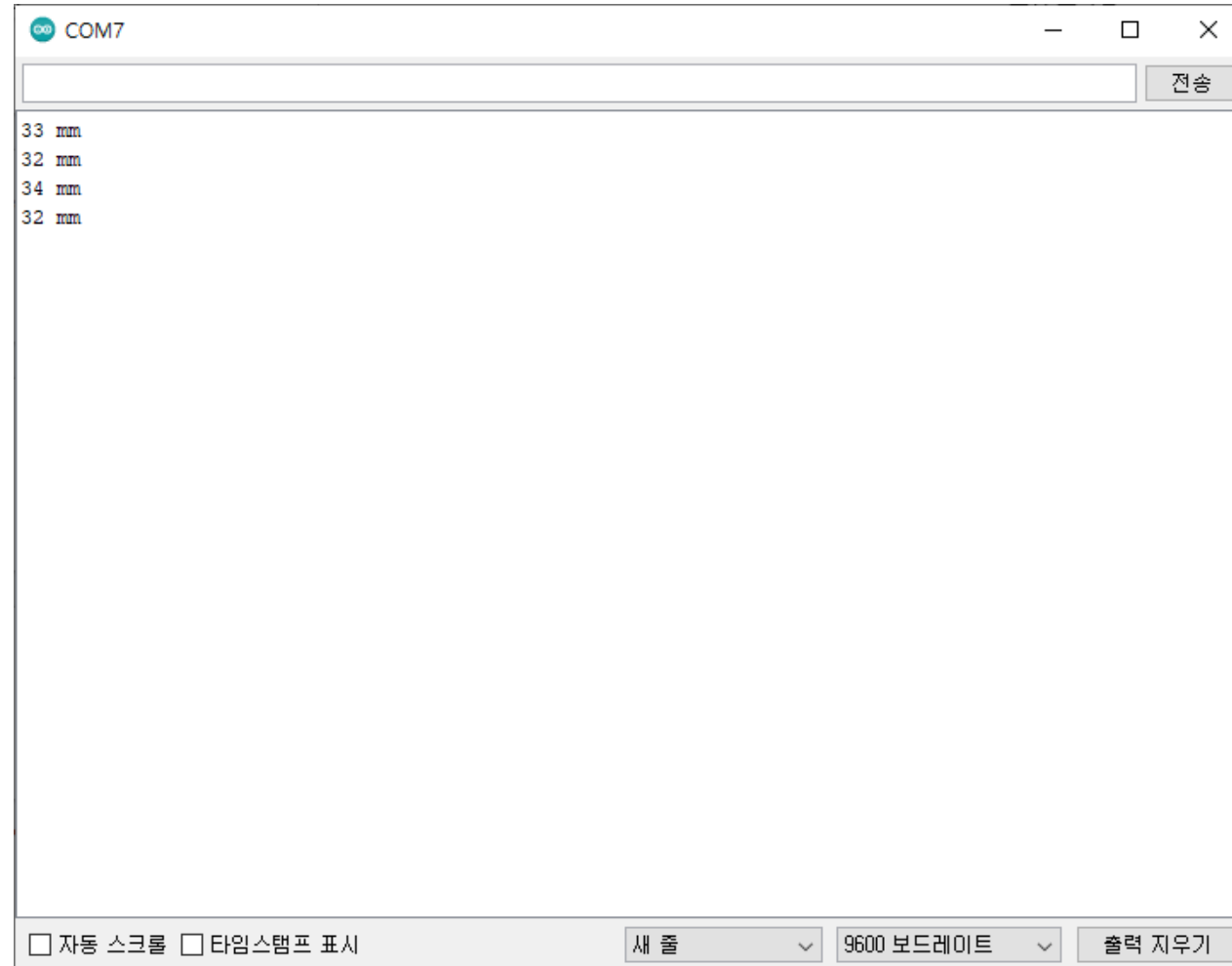
■ 시리얼 모니터로 결과 확인

→ 시리얼 모니터 버튼 클릭 혹은 "Ctrl + Shift + m" 입력



Exercise 1

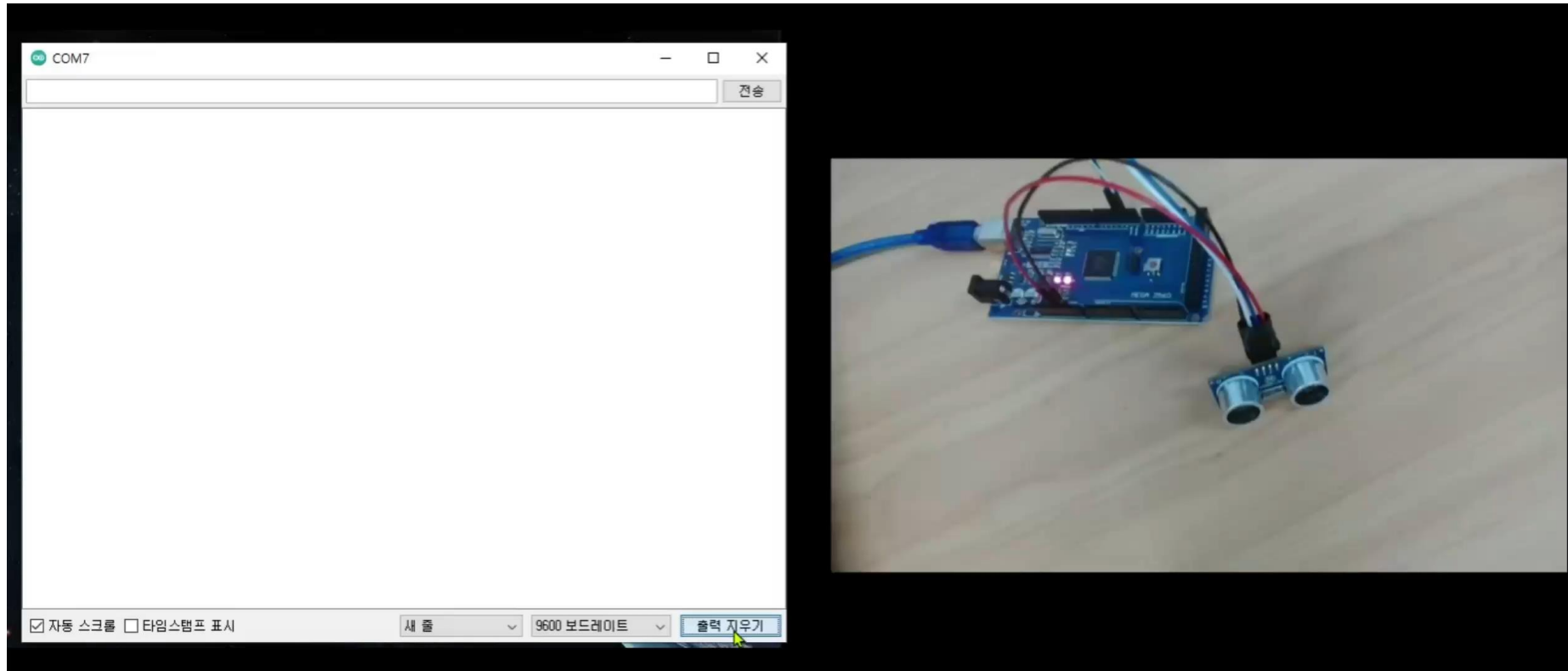
■ 시리얼 모니터 창



Exercise 1

■ 시리얼 모니터로 결과 확인

→ 거리 값 출력 확인

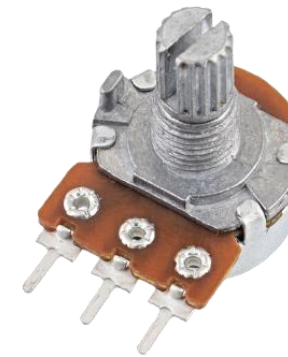
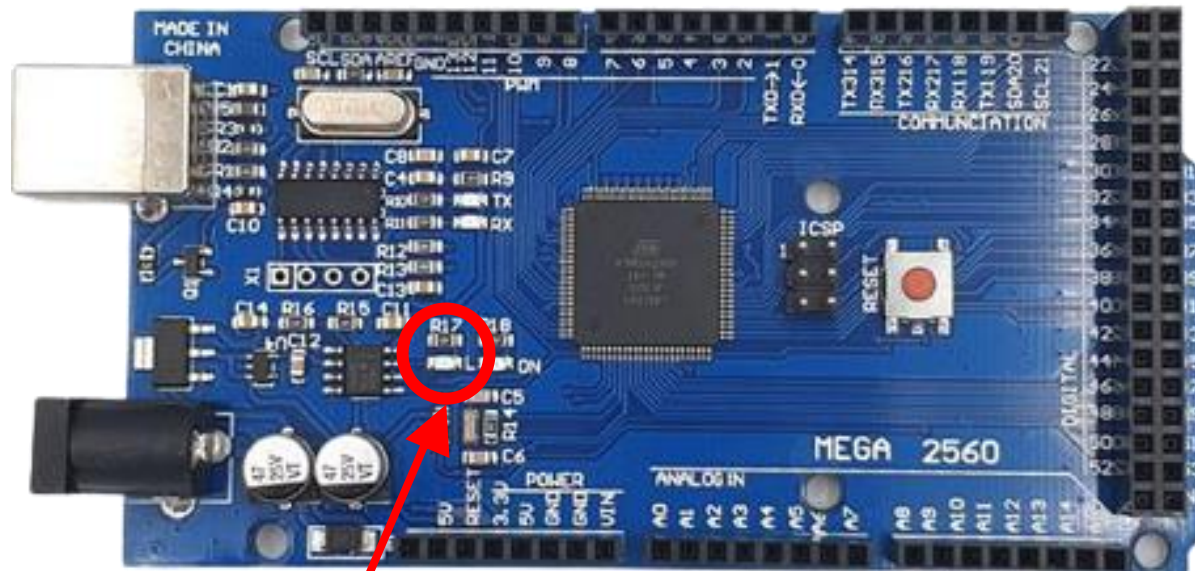


Exercise 2

Exercise 2

■ 가변저항을 이용한 LED 밝기 조절

→ 가변 저항을 이용하여 아날로그 입력으로 LED 밝기 조절



가변 저항

메가 2560 LED

Exercise 2

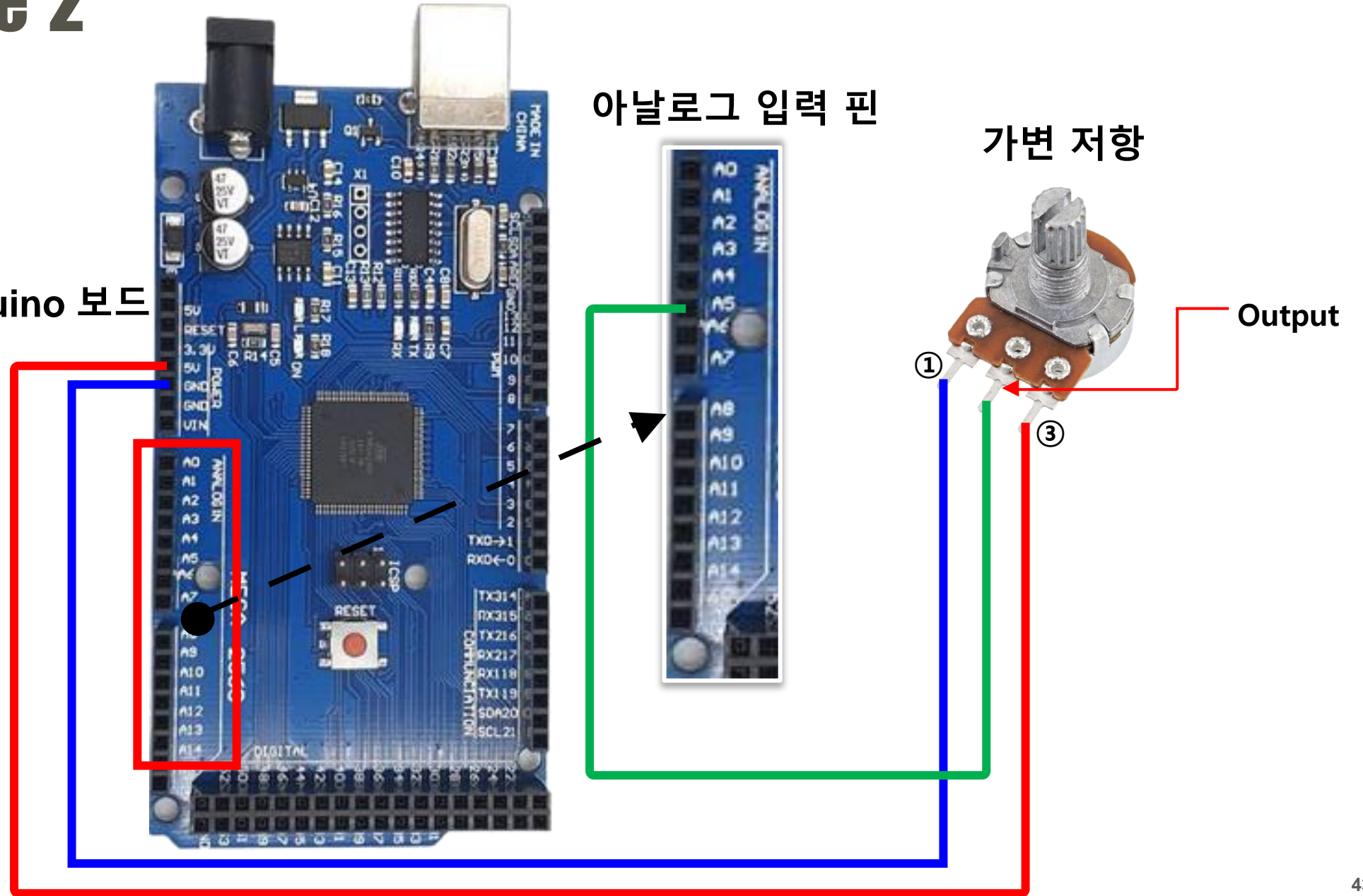
■ 선 연결

Arduino 보드

아날로그 입력 핀

가변 저항

Output



Exercise 2

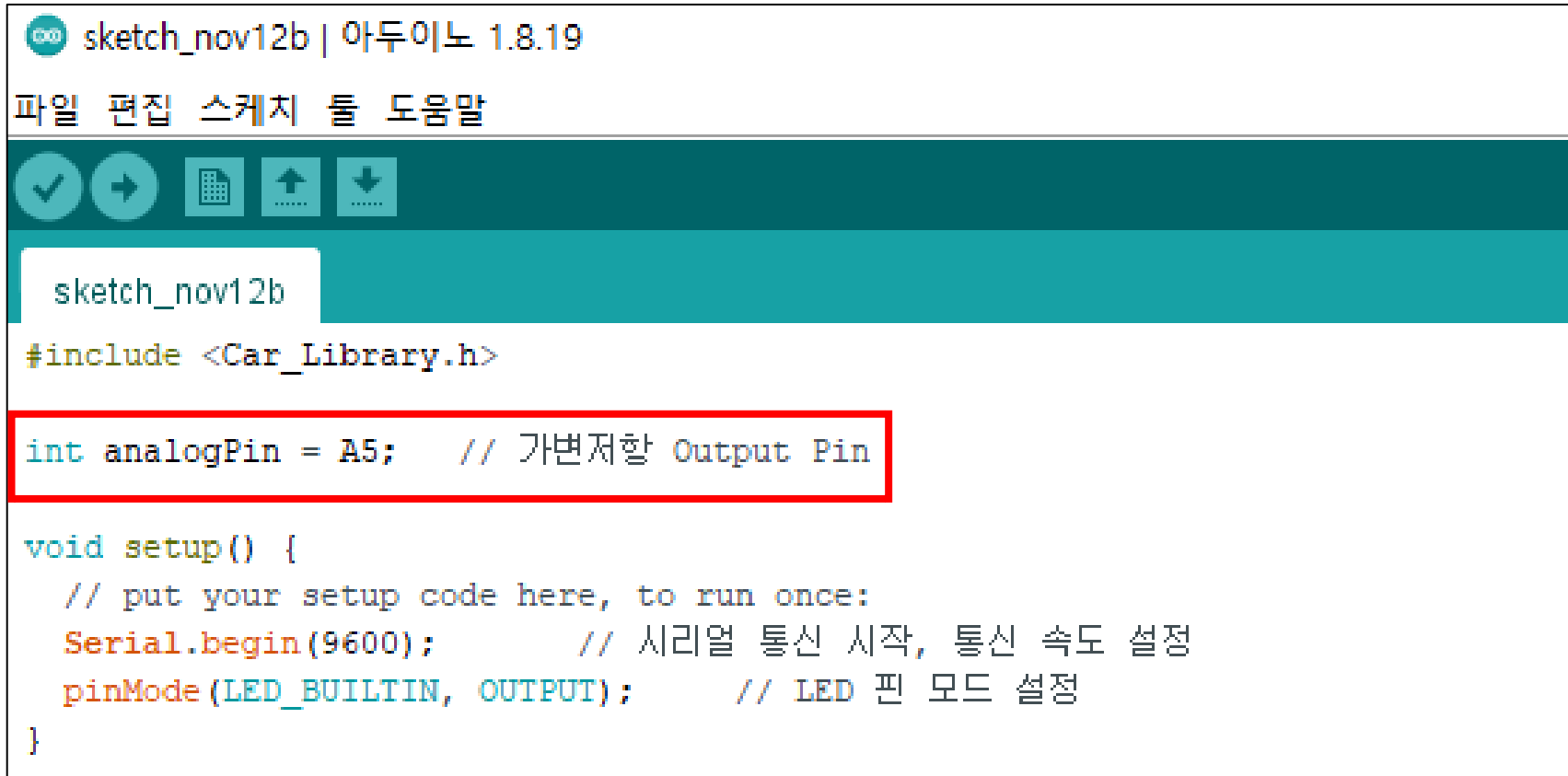
■ 선 연결 (Cont'd)

- 가변 저항 1번 핀 – 아두이노 GND에 연결
- 가변 저항 3번 핀 – 아두이노 5V에 연결
- 가변 저항 Output – 아두이노 Analog A5번 핀에 연결

Exercise 2

■ 핀 번호 변수 선언

→ 가변 저항 Output 핀과 연결된 아두이노 핀 번호 변수 선언



```
sketch_nov12b | 아두이노 1.8.19
파일 편집 스케치 툴 도움말

sketch_nov12b
#include <Car_Library.h>

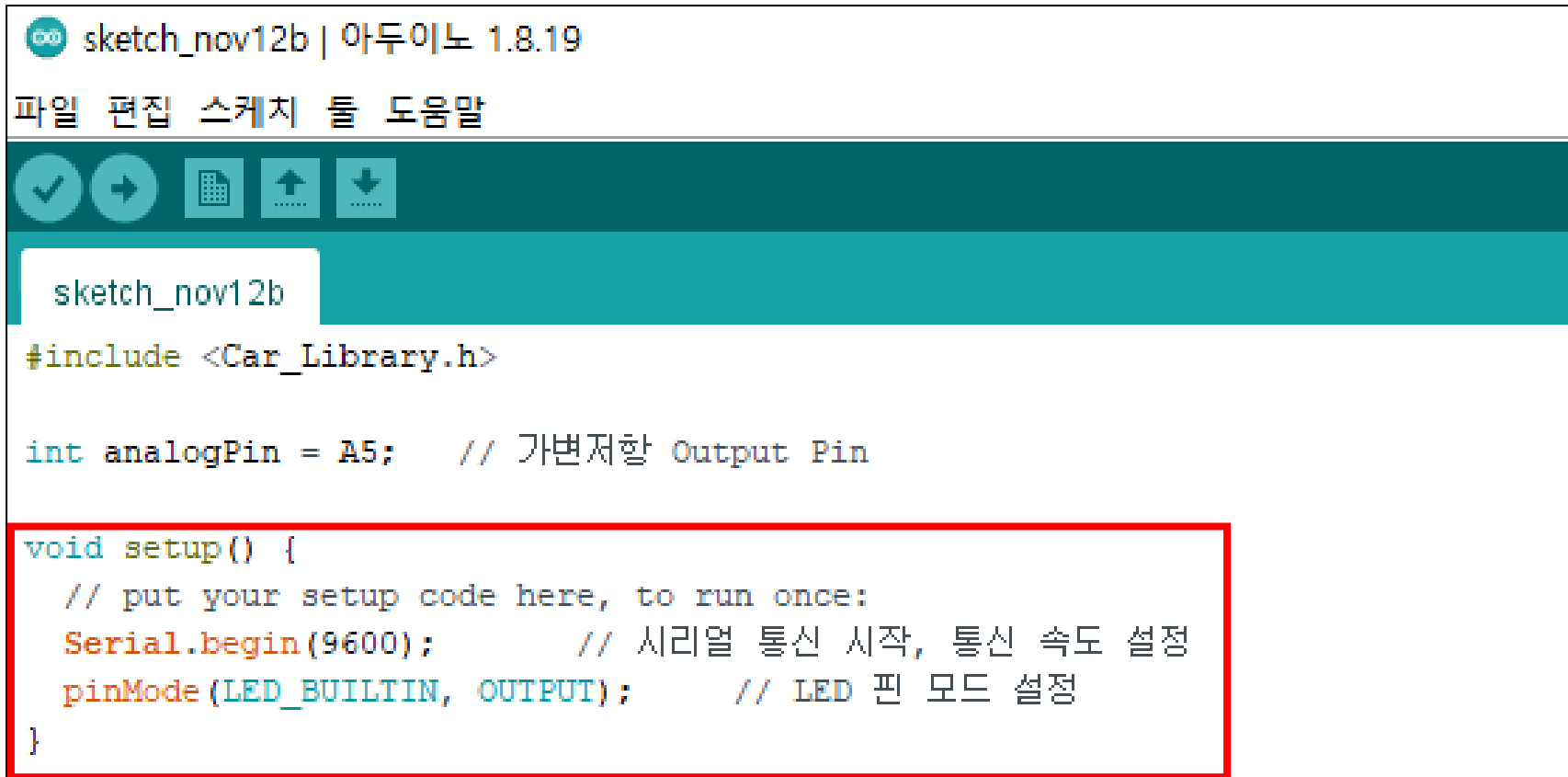
int analogPin = A5;    // 가변저항 Output Pin

void setup() {
    // put your setup code here, to run once:
    Serial.begin(9600);    // 시리얼 통신 시작, 통신 속도 설정
    pinMode(LED_BUILTIN, OUTPUT);    // LED 핀 모드 설정
}
```

Exercise 2

■ LED핀 모드 설정

→ 아두이노에 연결된 가변저항 핀 모드 설정



```
sketch_nov12b | 아두이노 1.8.19
파일 편집 스케치 툴 도움말

✓ → 📄 ⬆ ⬇

sketch_nov12b
#include <Car_Library.h>

int analogPin = A5;    // 가변저항 Output Pin

void setup() {
    // put your setup code here, to run once:
    Serial.begin(9600);    // 시리얼 통신 시작, 통신 속도 설정
    pinMode(LED_BUILTIN, OUTPUT);    // LED 핀 모드 설정
}
```

Exercise 2

■ 변수 선언 및 저항 값 읽어오는 함수 실행

→ 저항 값을 저장할 변수 선언, 가변저항의 저항 값을 읽어오는 함수 실행

```
void loop() {  
    // put your main code here, to run repeatedly:  
    int val;      // 저항값 저장할 변수 선언  
  
    // 가변저항의 저항값을 읽어오는 함수 실행  
    val = potentiometer_Read(analogPin);  
  
    // Serial 모니터로 출력  
    Serial.println(val);  
  
    // 가변 저항 값을 LED로 보내 출력  
    analogWrite(LED_BUILTIN, val);  
}
```

Exercise 2

■ “potentiometer_Read()” 설명

→ 가변저항의 저항 값을 읽어오는 함수

```
int potentiometer_Read(int pin)
```

①

```
value = analogRead(pin) / 4;
```

②

```
return value;
```

① Input: 가변저항의 Output핀과 연결된 핀 번호 입력

② Output: 최댓값을 255로 매핑한 저항 값을 출력

Exercise 2

■ 저항 값 시리얼 출력 및 LED 출력

- 가변저항의 저항 값을 시리얼 모니터에 출력
- 저항 값을 LED로 보내 밝기 조절

```
void loop() {  
    // put your main code here, to run repeatedly:  
    int val;          // 저항값 저장할 변수 선언  
  
    // 가변저항의 저항값을 읽어오는 함수 실행  
    val = potentiometer_Read(analogPin);  
  
    // Serial 모니터로 출력  
    Serial.println(val);  
  
    // 가변 저항 값을 LED로 보내 출력  
    analogWrite(LED_BUILTIN, val);  
}
```

Exercise 2

■ 함수 설명

- `analogWrite(핀 번호, 값)`

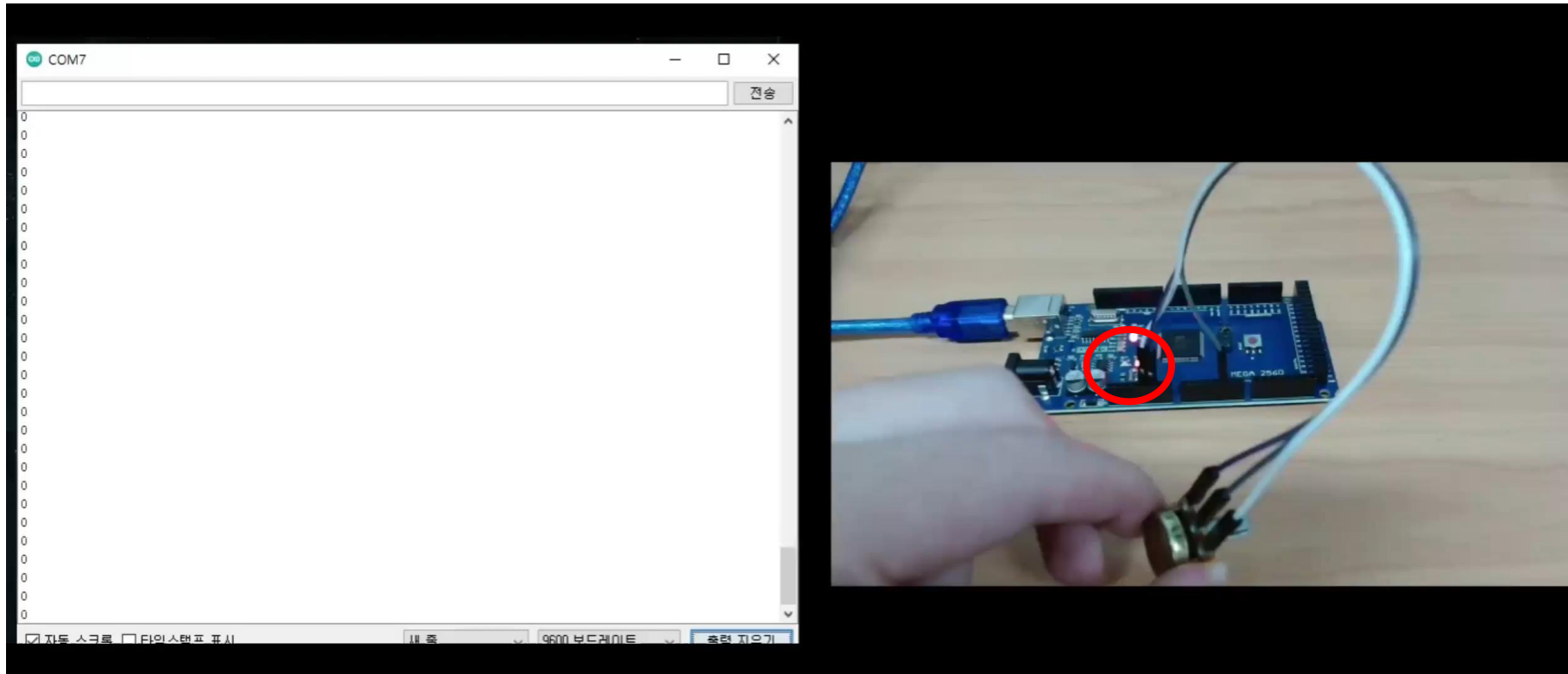
```
// 가변 저항 값을 LED로 보내 출력  
analogWrite(LED_BUILTIN, val);
```

→ 핀의 출력 전압을 입력 받은 값(0~255)으로 설정

Exercise 2

■ LED 밝기 조절 확인

→ 가변 저항으로 LED 밝기 조절 확인



Exercise 3

Exercise 3

■ 모터 제어

→ 아두이노에 모터 드라이버와 기어 모터를 연결하여 모터 제어



기어 모터

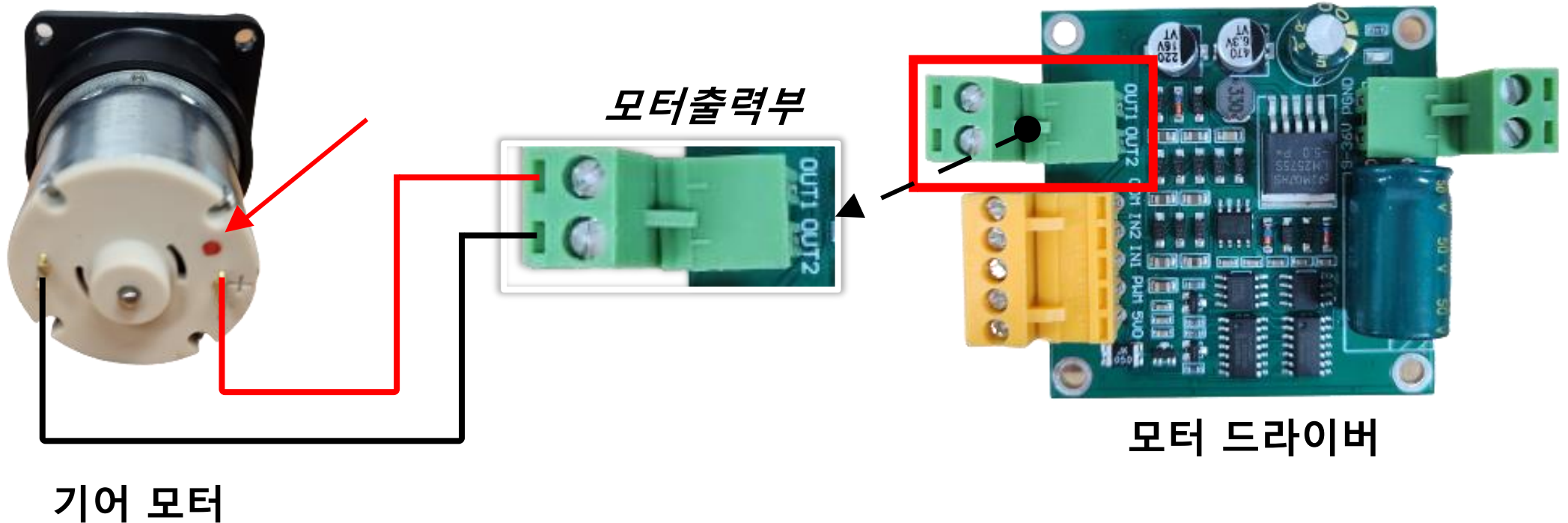


모터 드라이버

Exercise 3

■ 선 연결 - ①모터 드라이버:기어 모터

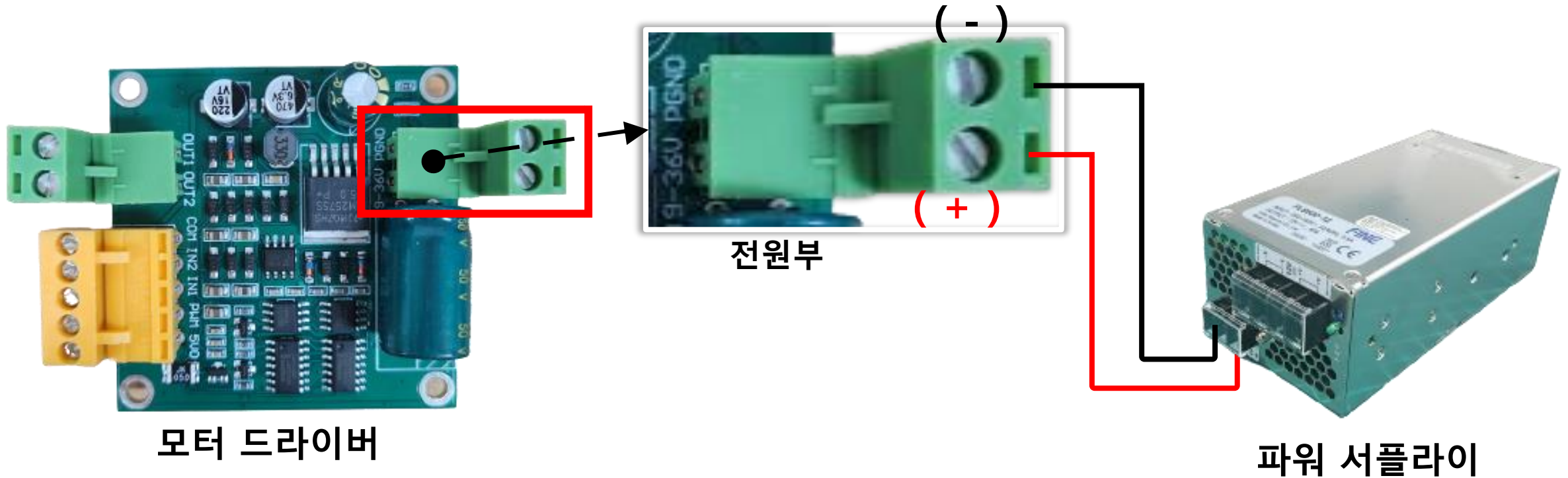
- 기어 모터 빨간색 표시 부분 - 모터 드라이버 OUT1에 연결
- 기어 모터 표시 안된 부분 - 모터 드라이버 OUT2에 연결



Introduction

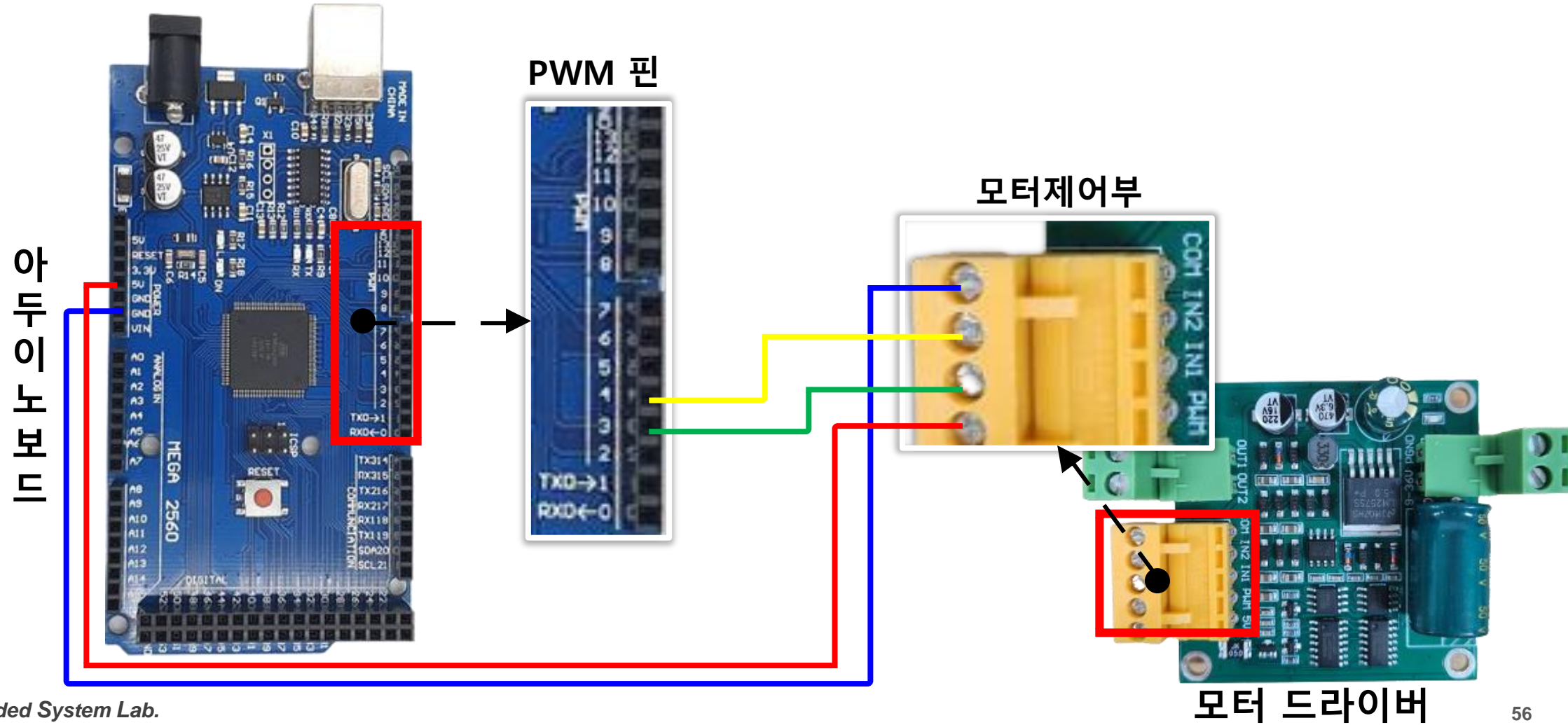
■ 선 연결 - ② 모터 드라이버:전원부

- 12V 모터를 사용하기 때문에 12V 전원 필요
- 파워 서플라이 양극(+) - 모터 드라이버 전원부 9-36V 부분에 연결
- 파워 서플라이 음극(-) - 모터 드라이버 전원부 PGND 부분에 연결



Introduction

■ 선 연결 - ③ 아두이노:모터 드라이버



Introduction

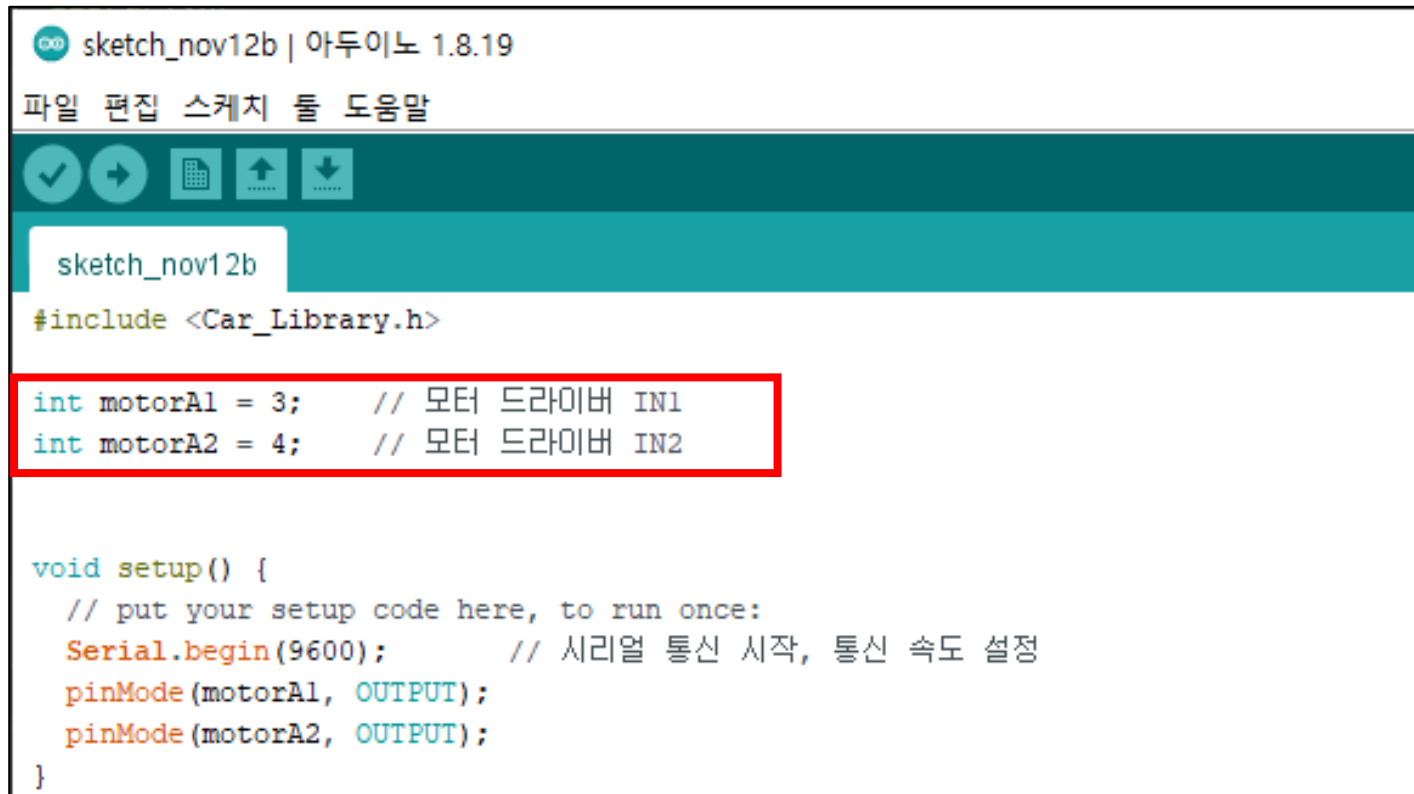
■ 선 연결 - ③ 아두이노:모터 드라이버 (Cont'd)

- 모터 드라이버 제어부의 COM – 아두이노드의 GND 핀
- 모터 드라이버 제어부의 PWM – 아두이노드의 5V 핀
- 모터 드라이버 제어부의 IN1 – 아두이노드의 PWM 3번 핀
- 모터 드라이버 제어부의 IN2 – 아두이노드의 PWM 4번 핀

Exercise 3

■ 핀 변수 선언

→ 드라이버의 제어 핀과 연결된 아두이노 핀 번호 변수 선언



The screenshot shows the Arduino IDE interface for a sketch named 'sketch_nov12b' using Arduino 1.8.19. The code editor displays the following code:

```
sketch_nov12b
#include <Car_Library.h>

int motorA1 = 3;    // 모터 드라이버 IN1
int motorA2 = 4;    // 모터 드라이버 IN2

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);    // 시리얼 통신 시작, 통신 속도 설정
  pinMode(motorA1, OUTPUT);
  pinMode(motorA2, OUTPUT);
}
```

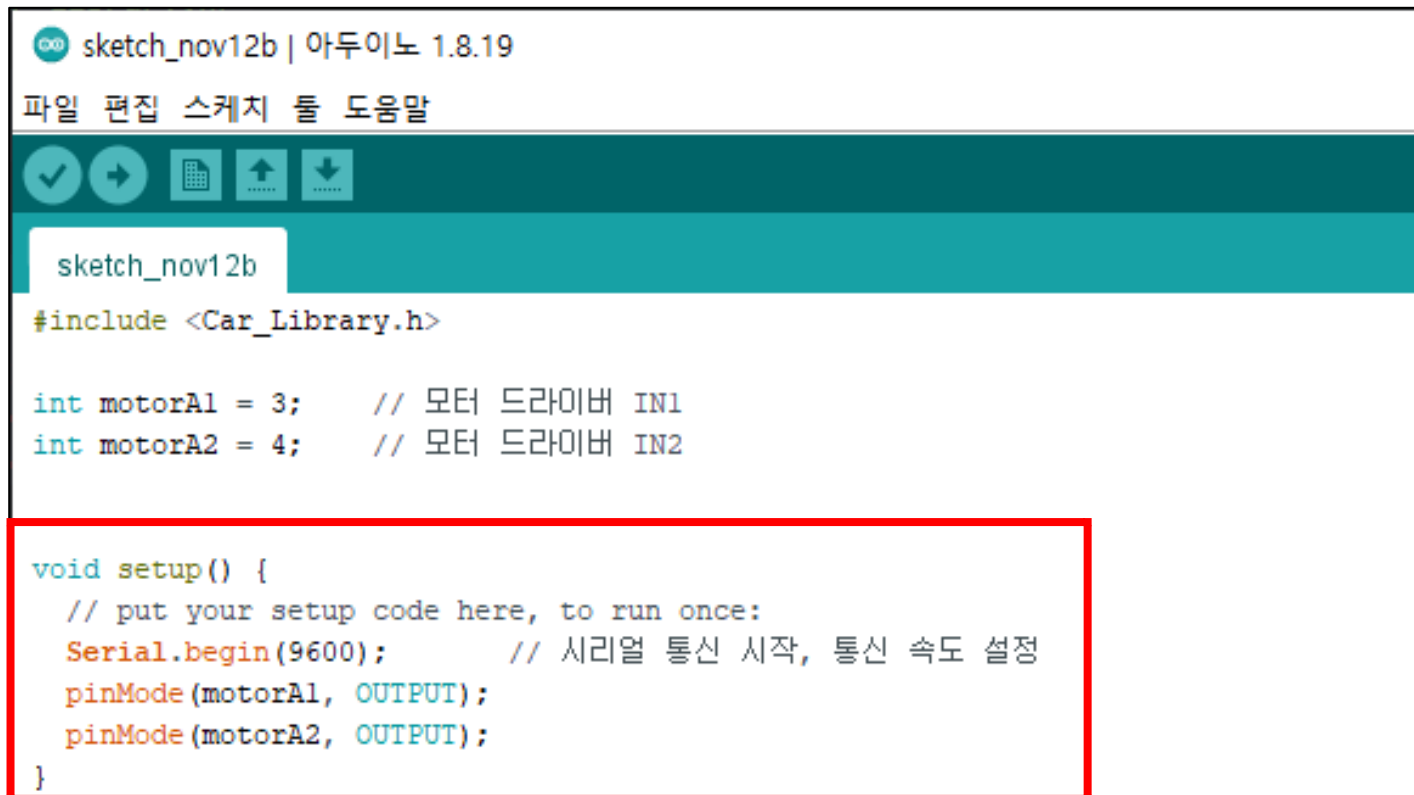
The two lines defining the motor pins are highlighted with a red box:

```
int motorA1 = 3;    // 모터 드라이버 IN1
int motorA2 = 4;    // 모터 드라이버 IN2
```

Exercise 3

■ 핀 모드 설정

→ 모터 제어 핀 모드 설정



The screenshot shows the Arduino IDE interface for a sketch named 'sketch_nov12b' using Arduino 1.8.19. The code includes the 'Car_Library.h' header and defines two motor pins: motorA1 (pin 3) and motorA2 (pin 4). The 'setup()' function is highlighted with a red box and contains the following code:

```
#include <Car_Library.h>

int motorA1 = 3;    // 모터 드라이버 IN1
int motorA2 = 4;    // 모터 드라이버 IN2

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);    // 시리얼 통신 시작, 통신 속도 설정
  pinMode(motorA1, OUTPUT);
  pinMode(motorA2, OUTPUT);
}
```

Exercise 3

■ loop() 코드

```
void loop() {  
    // put your main code here, to run repeatedly:  
  
    // Forward  
    Serial.println("Motor Forward");  
    motor_forward(motorA1, motorA2, 75);  
    delay(3000);  
  
    // Backward  
    Serial.println("Motor Backward");  
    motor_backward(motorA1, motorA2, 150);  
    delay(3000);  
  
    // Hold  
    Serial.println("Motor Hold");  
    motor_hold(motorA1, motorA2);  
    delay(3000);  
}
```

Exercise 3

■ 모터 정방향 회전

→ "motor_forward()" 함수 실행, 제어 핀 번호와 모터 속도를 입력

```
void loop() {  
    // put your main code here, to run repeatedly:  
  
    // Forward  
    Serial.println("Motor Forward");  
    motor_forward(motorA1, motorA2, 175);  
    delay(3000);  
  
    // Backward  
    Serial.println("Motor Backward");  
    motor_backward(motorA1, motorA2, 175);  
    delay(3000);  
  
    // Hold  
    Serial.println("Motor Hold");  
    motor_hold(motorA1, motorA2);  
    delay(3000);  
}
```

Exercise 3

■ “motor_forward()” 설명

→ 모터를 정방향을 회전시키는 함수

```
void motor_forward(int IN1, int IN2, int speed) ①
```

```
{  
    analogWrite(IN1, speed);  
    analogWrite(IN2, LOW);  
}
```

 ②

- ① Input: 모터 드라이버 IN1, IN2와 연결된 핀 번호, 모터 회전속도(0~255) 입력
- ② 모터가 정방향으로 입력 받은 회전속도로 회전

Exercise 3

■ 모터 역방향 회전

→ "motor_backward()" 함수 실행, 제어 핀 번호와 모터 속도를 입력

```
void loop() {  
  // put your main code here, to run repeatedly:  
  
  // Forward  
  Serial.println("Motor Forward");  
  motor_forward(motorA1, motorA2, 175);  
  delay(3000);  
  
  // Backward  
  Serial.println("Motor Backward");  
  motor_backward(motorA1, motorA2, 175);  
  delay(3000);  
  
  // Hold  
  Serial.println("Motor Hold");  
  motor_hold(motorA1, motorA2);  
  delay(3000);  
}
```

Exercise 3

■ “motor_backward()” 설명

→ 모터를 역방향을 회전시키는 함수

```
void motor_backward(int IN1, int IN2, int speed) ①
```

```
{  
    analogWrite(IN1, LOW);  
    analogWrite(IN2, speed);  
}
```

 ②

① Input: 모터 드라이버 IN1, IN2와 연결된 핀 번호, 모터 회전속도 입력

② 모터가 역방향으로 입력 받은 회전속도로 회전

Exercise 3

■ 모터 정지

→ “motor_hold()” 함수 실행, 제어 핀 번호를 입력

```
void loop() {  
    // put your main code here, to run repeatedly:  
  
    // Forward  
    Serial.println("Motor Forward");  
    motor_forward(motorA1, motorA2, 175);  
    delay(3000);  
  
    // Backward  
    Serial.println("Motor Backward");  
    motor_backward(motorA1, motorA2, 175);  
    delay(3000);  
  
    // Hold  
    Serial.println("Motor Hold");  
    motor_hold(motorA1, motorA2);  
    delay(3000);  
}
```

Exercise 3

■ “motor_hold()” 설명

→ 모터를 정지시키는 함수

```
void motor_hold(int IN1, int IN2) ①
```

```
{  
    analogWrite(IN1, LOW);  
    analogWrite(IN2, LOW);  
}
```

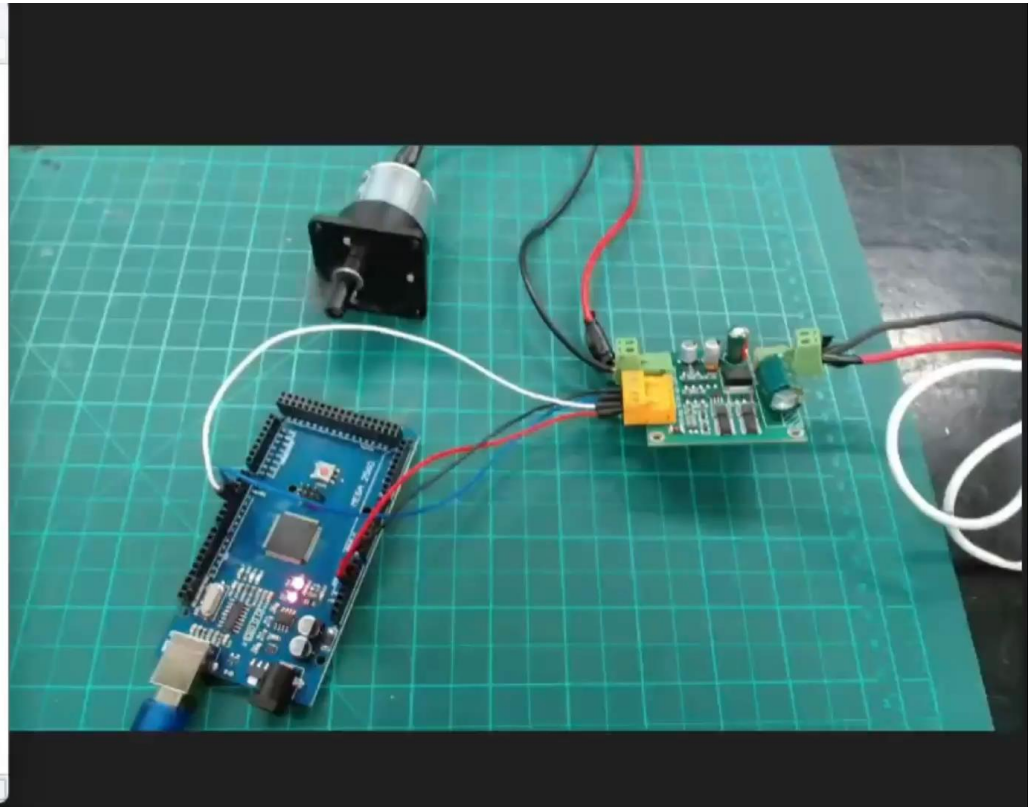
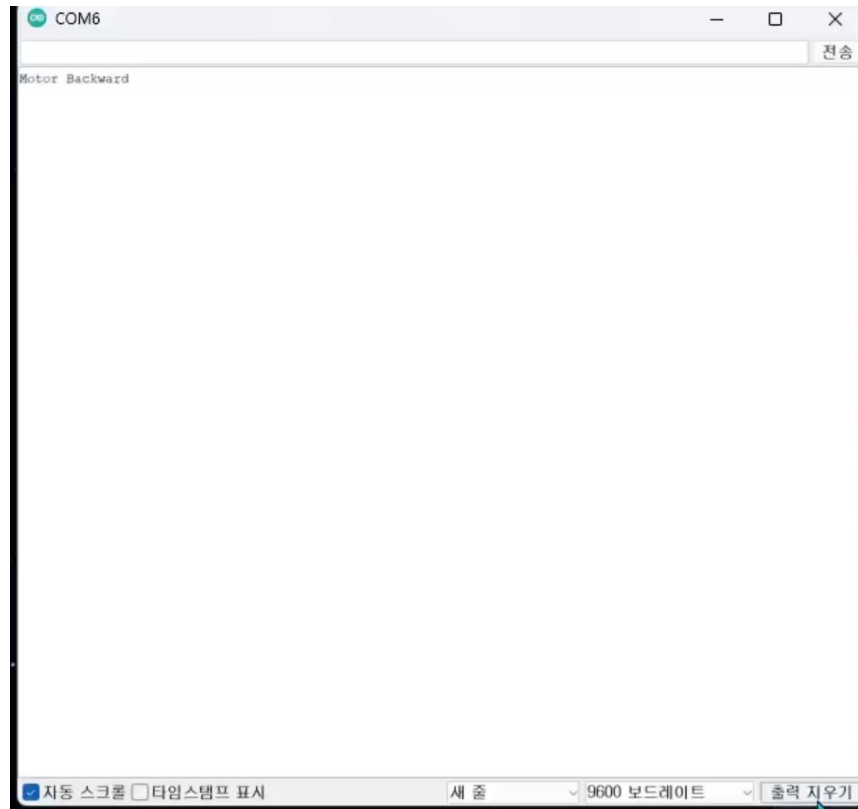
 ②

① Input: 모터 드라이버 IN1, IN2와 연결된 핀 번호 입력

② 모터 회전 정지

Exercise 3

■ 결과 확인



Exercise 4

Exercise 4

■ 시리얼 통신 이용한 Python 프로그램으로 아두이노 제어

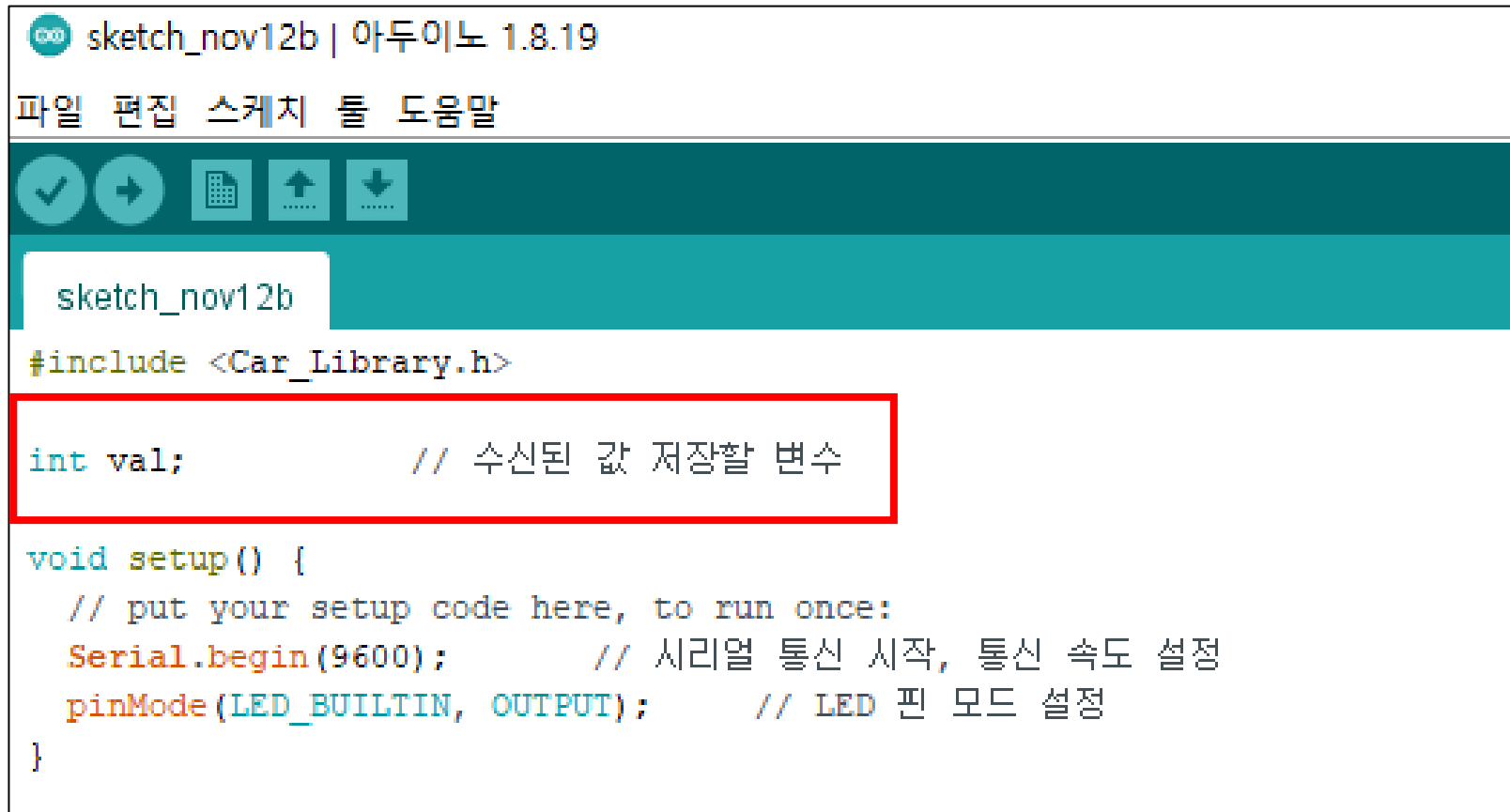
→ 시리얼 통신을 이용하여 Python 프로그램으로 아두이노에게 정수 값을 전송하여 LED 밝기 제어

```
while True > if input_value == 'q'
Run: main x
C:\Users\autolab\anaconda3\envs\Test\python.exe C:\Users\autolab\PycharmProjects\pythonPro
아두이노로 전송할 저항 값(0~255): 255
아두이노로 전송할 저항 값(0~255): 100
아두이노로 전송할 저항 값(0~255): 75
아두이노로 전송할 저항 값(0~255): 10
아두이노로 전송할 저항 값(0~255): 1
아두이노로 전송할 저항 값(0~255): 255
아두이노로 전송할 저항 값(0~255): 50
아두이노로 전송할 저항 값(0~255): 255
아두이노로 전송할 저항 값(0~255): 10
아두이노로 전송할 저항 값(0~255): 0
아두이노로 전송할 저항 값(0~255):
```

Exercise 4

■ 변수 선언 및 시리얼 통신 설정

→ 시리얼로 전송된 값을 저장할 변수 선언



```
sketch_nov12b | 아두이노 1.8.19
파일 편집 스케치 툴 도움말

sketch_nov12b
#include <Car_Library.h>

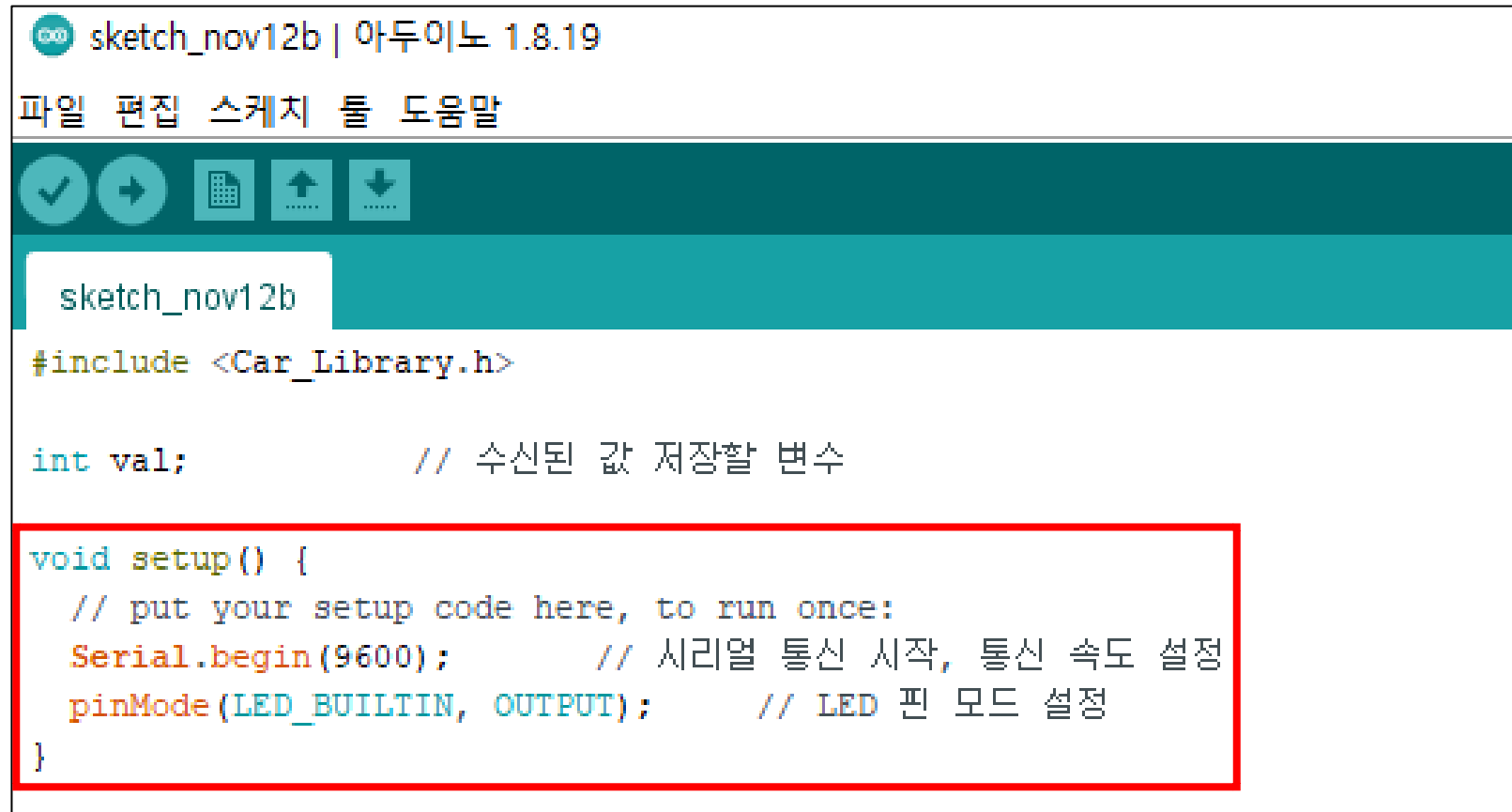
int val;           // 수신된 값 저장할 변수

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);      // 시리얼 통신 시작, 통신 속도 설정
  pinMode(LED_BUILTIN, OUTPUT); // LED 핀 모드 설정
}
```

Exercise 4

■ 변수 선언 및 시리얼 통신 설정

→ 시리얼 통신 시작 및 통신 속도 설정, LED 핀 모드 설정



The screenshot shows the Arduino IDE interface. At the top, it says "sketch_nov12b | 아두이노 1.8.19". Below that are menu items: "파일", "편집", "스케치", "툴", "도움말". A toolbar with icons for check, run, upload, and download is visible. The file name "sketch_nov12b" is in the editor title bar. The code in the editor is as follows:

```
#include <Car_Library.h>

int val;          // 수신된 값 저장할 변수

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);      // 시리얼 통신 시작, 통신 속도 설정
  pinMode(LED_BUILTIN, OUTPUT);  // LED 핀 모드 설정
}
```

The `void setup()` block is highlighted with a red rectangle.

Exercise 4

■ 시리얼 통신 데이터 수신

- 시리얼 데이터 읽기 및 데이터 정수형으로 변환
- 전송된 값을 LED로 보내 LED 출력

```
void loop() {  
  // put your main code here, to run repeatedly:  
  if(Serial.available()) {  
    val = Serial.parseInt();  
  
    if(val >= 0) {  
      analogWrite(LED_BUILTIN, val);  
    }  
  }  
}
```


Exercise 4

■ 함수 설명

- **Serial.available()**

→ 시리얼 포트로부터 수신 받은 데이터가 있는지 확인

```
if (Serial.available()) {  
    val = Serial.parseInt();  
}
```

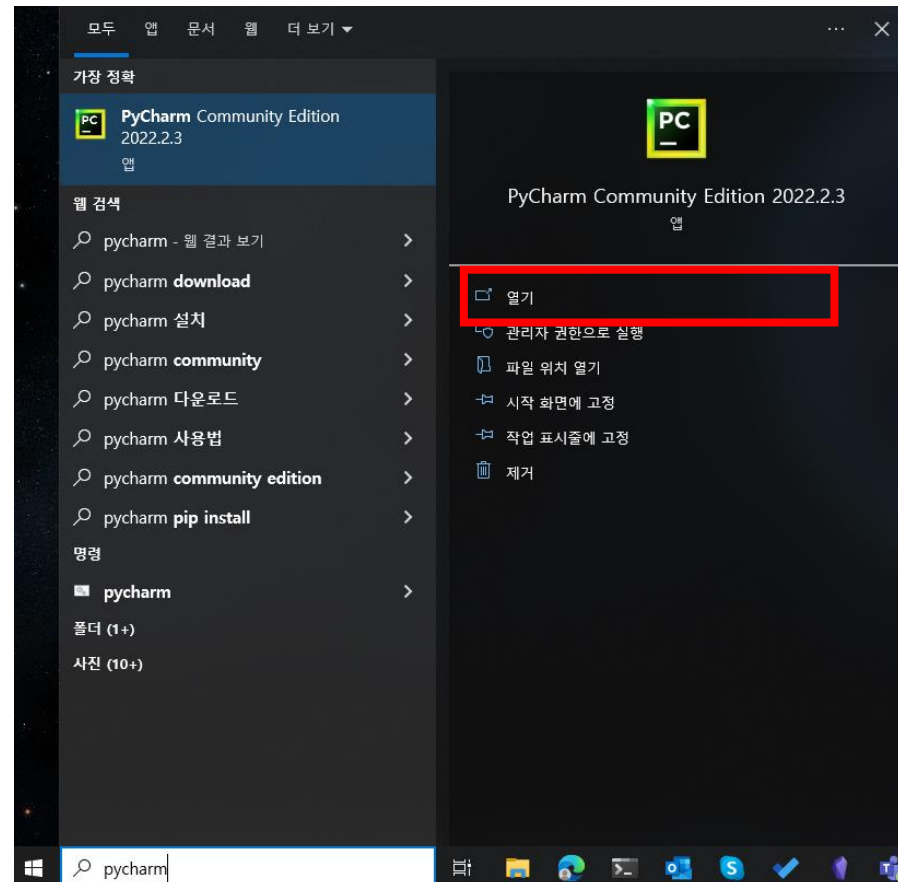
- **Serial.parseInt()**

→ 시리얼 포트로부터 수신 받은 데이터를 정수로 변환

Exercise 4

■ Pycharm 앱 실행

- 키보드에 "Win+q" 입력 후, "pycharm" 입력 후 "열기" 클릭
- 앱 설치는 "환경세팅" 강의자료 참고!



Exercise 4

■ Pycharm 프로젝트 생성 및 라이브러리 설치

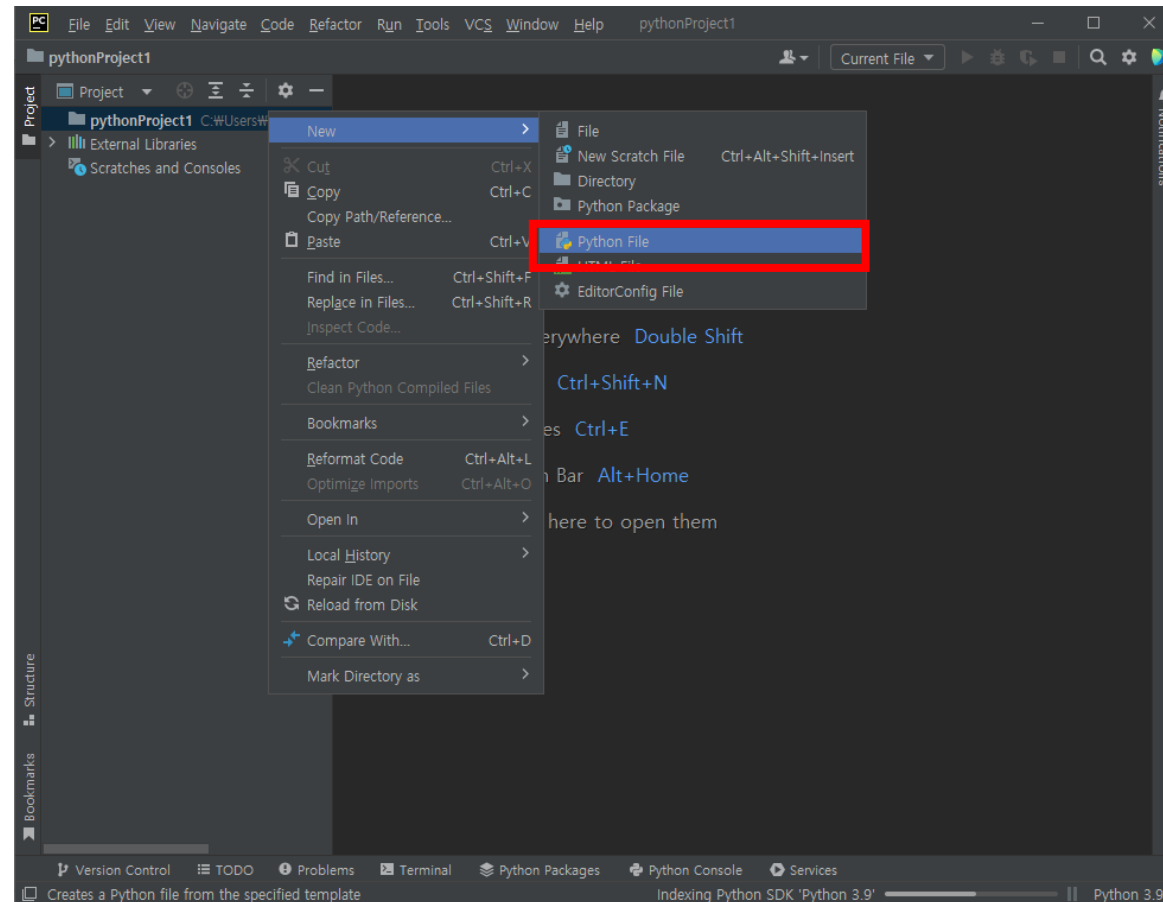
- 해당 예제 진행을 위해 Python "pyserial" 라이브러리 설치 필요
- 프로젝트 생성 및 라이브러리 설치 방법은 "환경설정" 강의 참고!

Exercise 4

■ main.py 파일 생성

→ 프로젝트에 main.py 파일 만들기

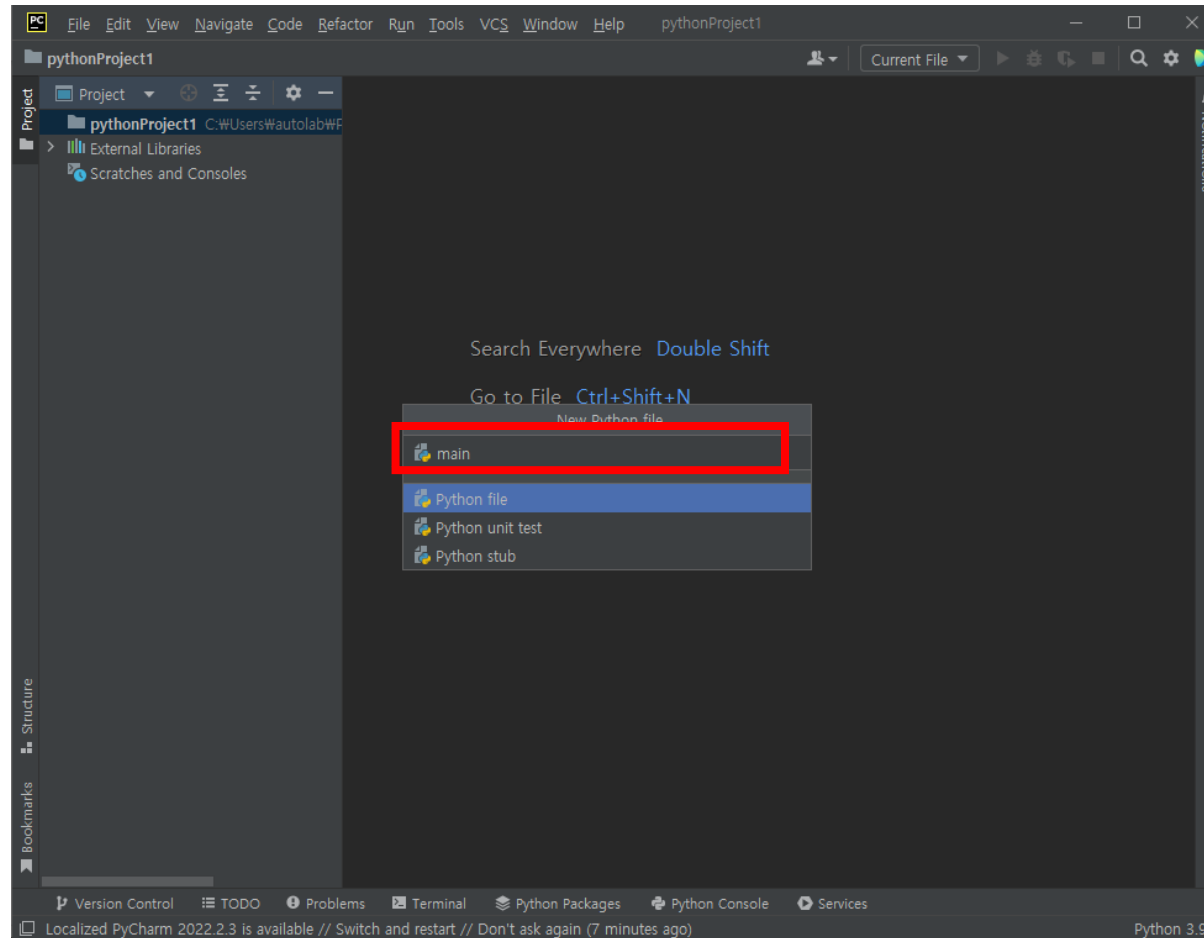
→ 좌측에 프로젝트 폴더 우클릭 -> New 선택 -> Python File 클릭



Exercise 4

■ main.py 파일 생성

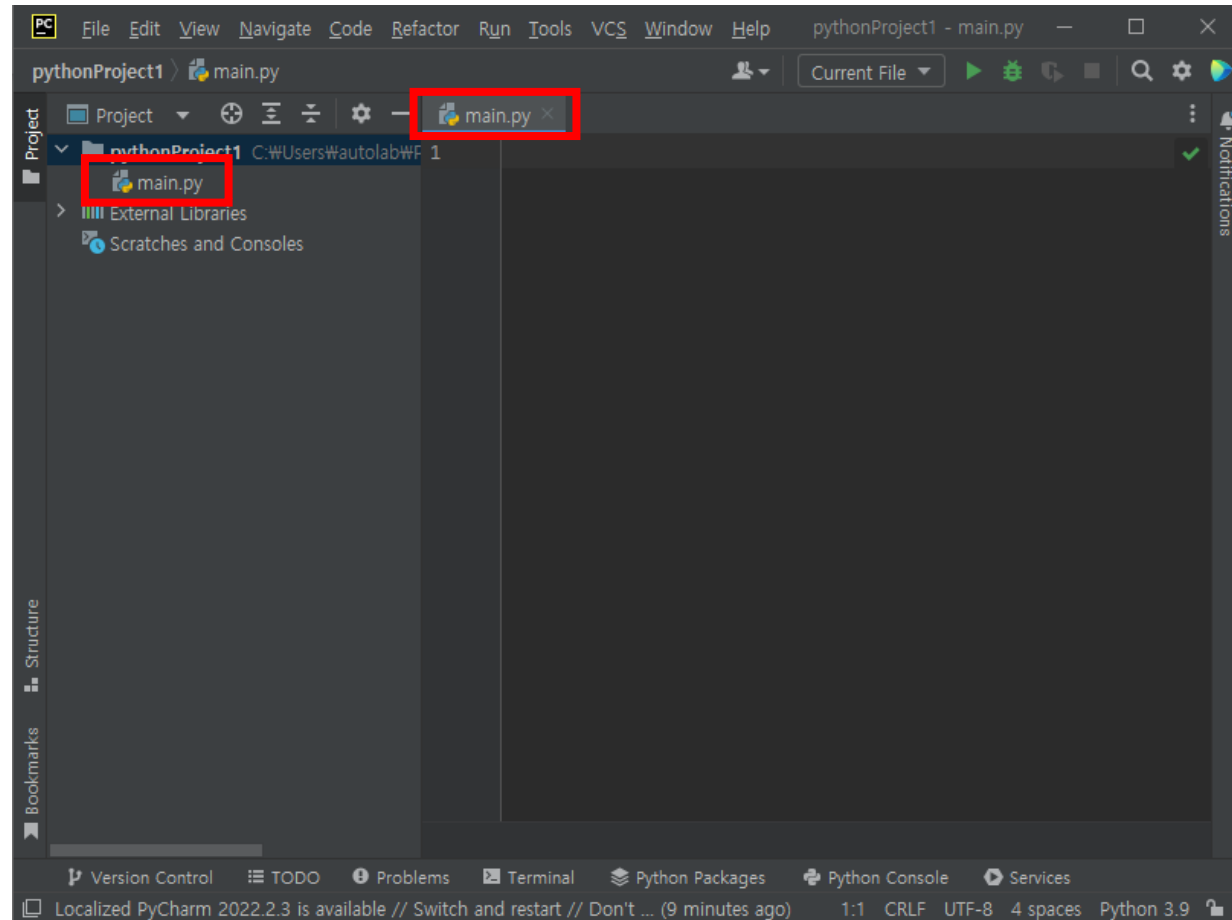
→ 키보드로 "main" 입력 후 엔터



Exercise 4

■ main.py 파일 생성

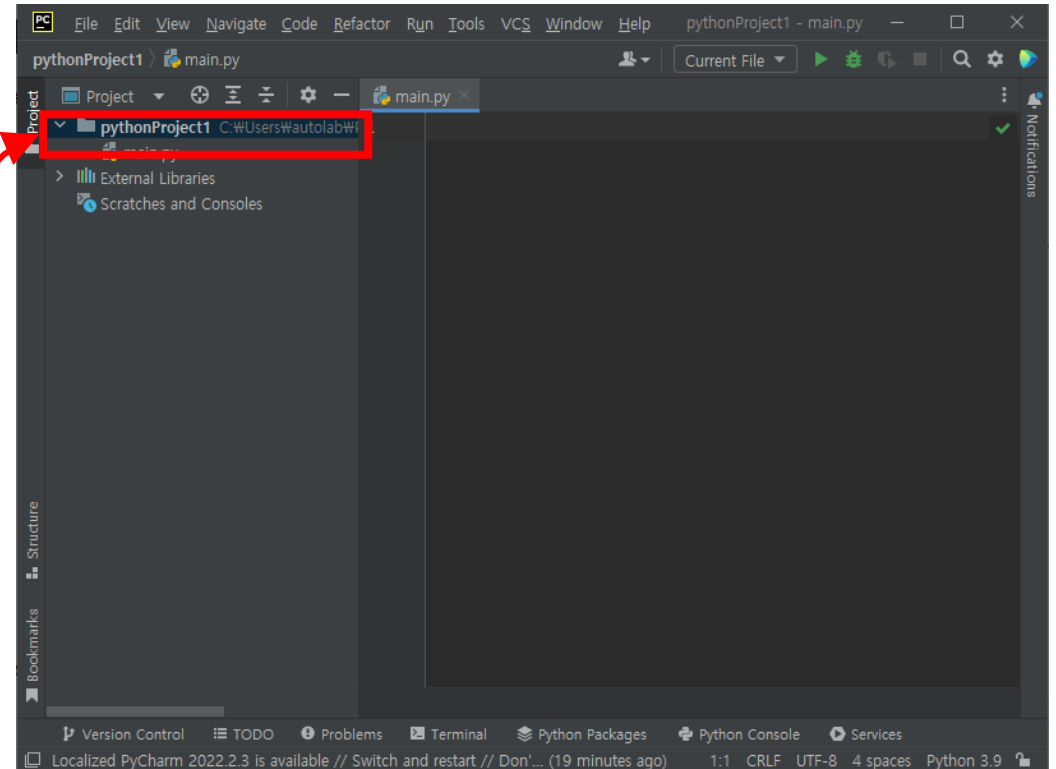
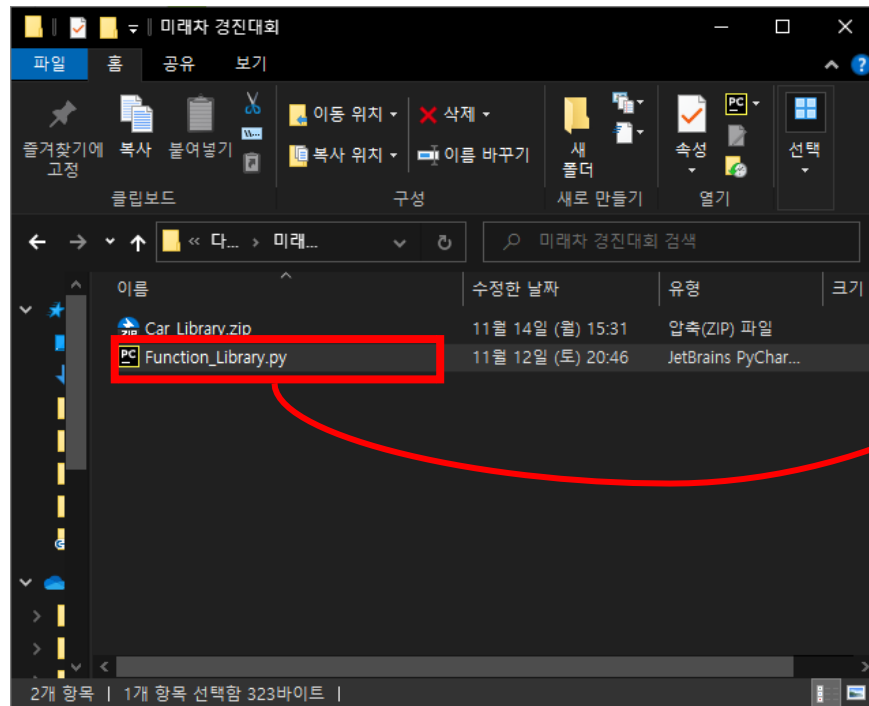
→ main.py 파일 생성 확인



Exercise 4

■ Function_Library.py 파일 추가

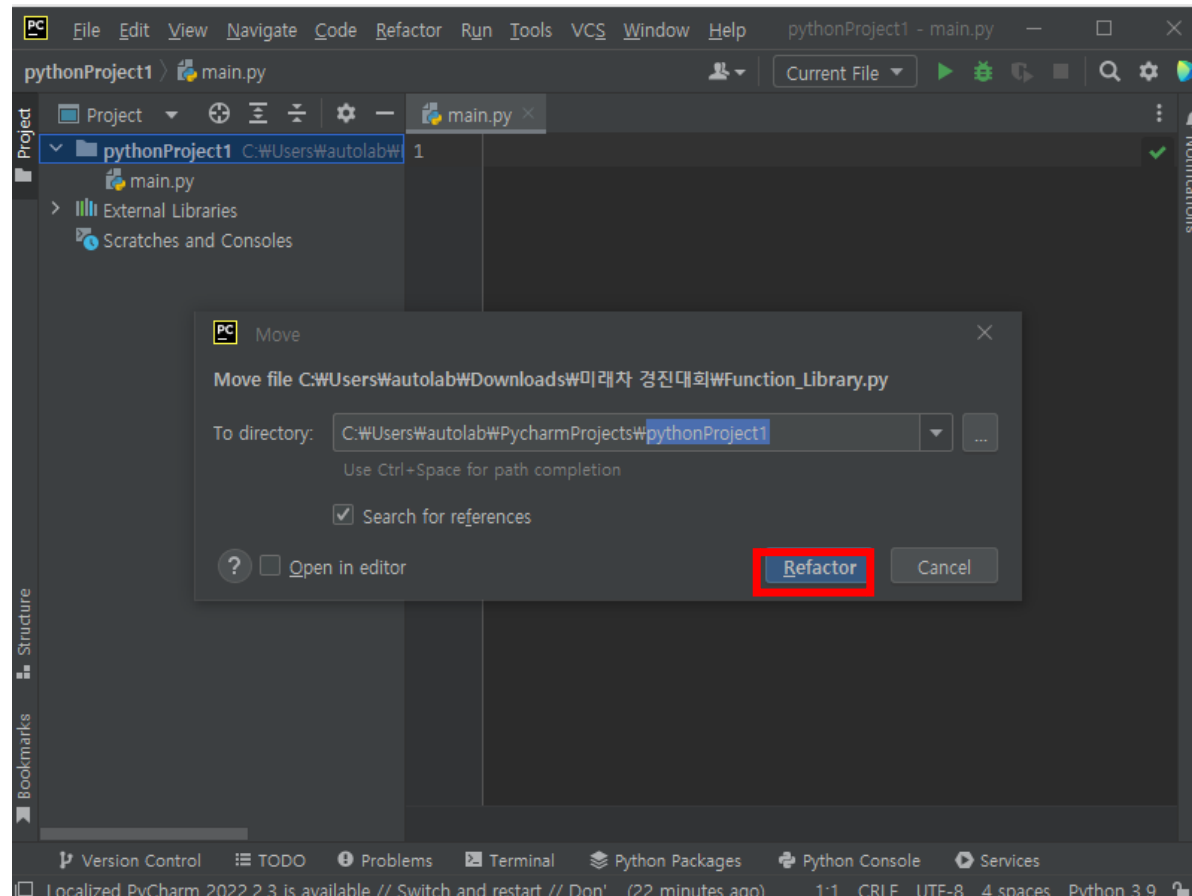
- 깃허브에서 다운받은 "Function_Library.py"를 Python Project에 추가
- 마우스로 파일 선택 후 PythonProject로 드래그앤드롭



Exercise 4

■ Function_Library.py 파일 추가

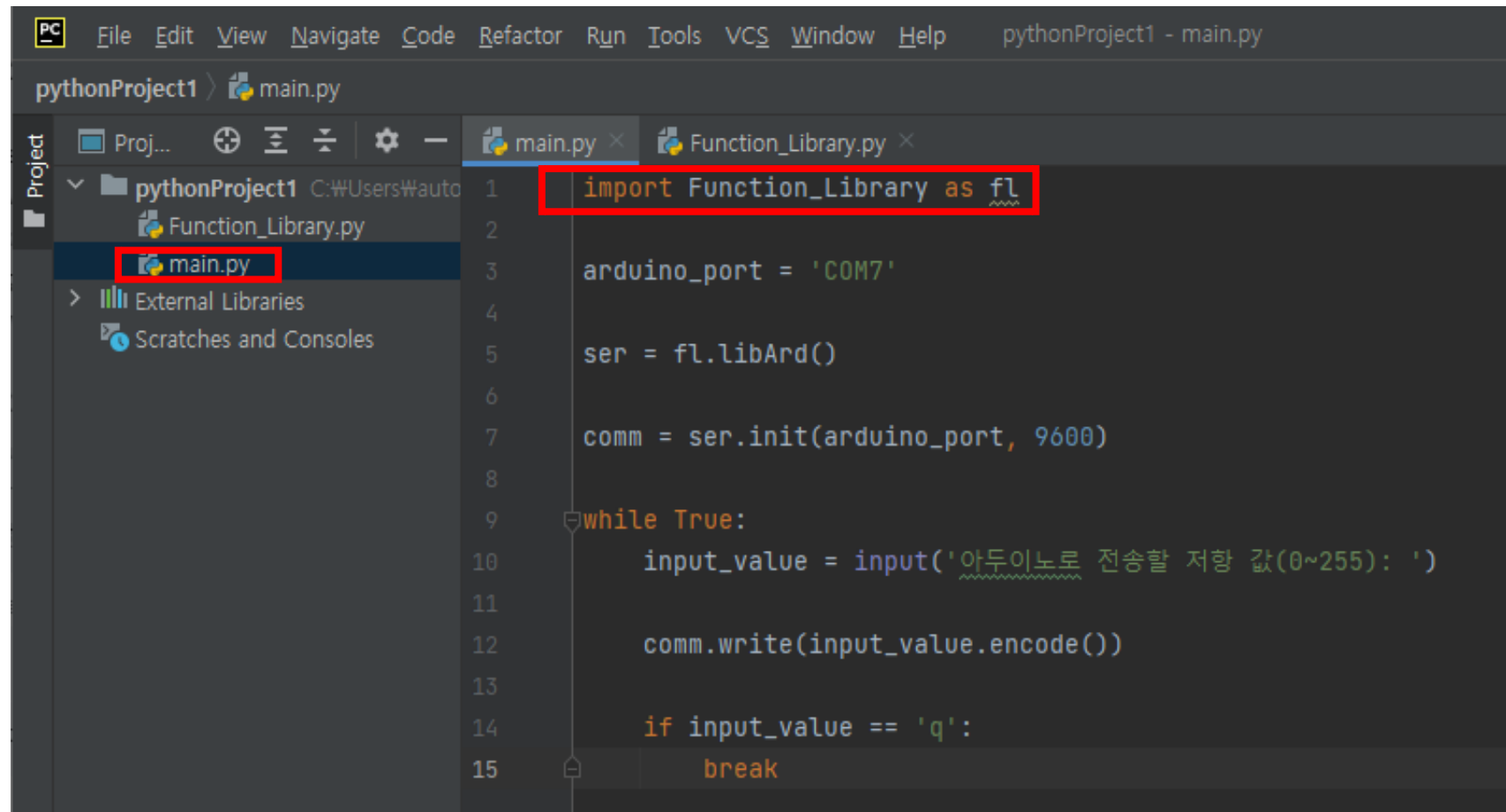
→ 해당 창이 뜨면 엔터 누르거나 Refactor 버튼 클릭



Exercise 4

■ (Python) 라이브러리 Import

→ main.py 소스코드에 “import Function_Library as fl” 입력



The screenshot shows an IDE window titled "pythonProject1 - main.py". The left sidebar displays the project structure with "main.py" selected. The main editor area shows the following Python code:

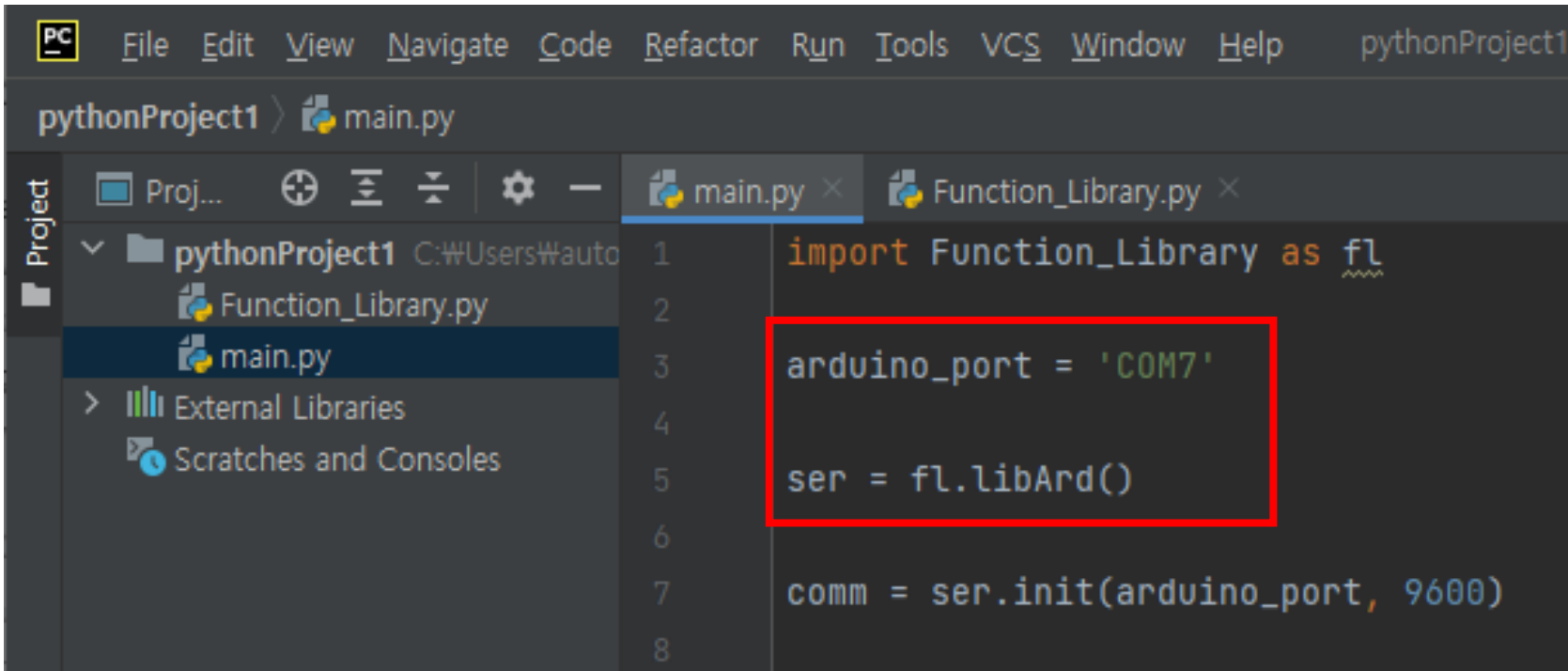
```
1 import Function_Library as fl
2
3 arduino_port = 'COM7'
4
5 ser = fl.libArd()
6
7 comm = ser.init(arduino_port, 9600)
8
9 while True:
10     input_value = input('아두이노로 전송할 저항 값(0~255): ')
11
12     comm.write(input_value.encode())
13
14     if input_value == 'q':
15         break
```

The first line, `import Function_Library as fl`, is highlighted with a red rectangle.

Exercise 4

■ 변수 선언 및 클래스 불러오기

- 시리얼 포트 번호 변수 선언
- "Function_Library.py"의 libArd() 클래스를 불러옴
- libArd() 클래스는 실습에 필요한 함수들의 집합체

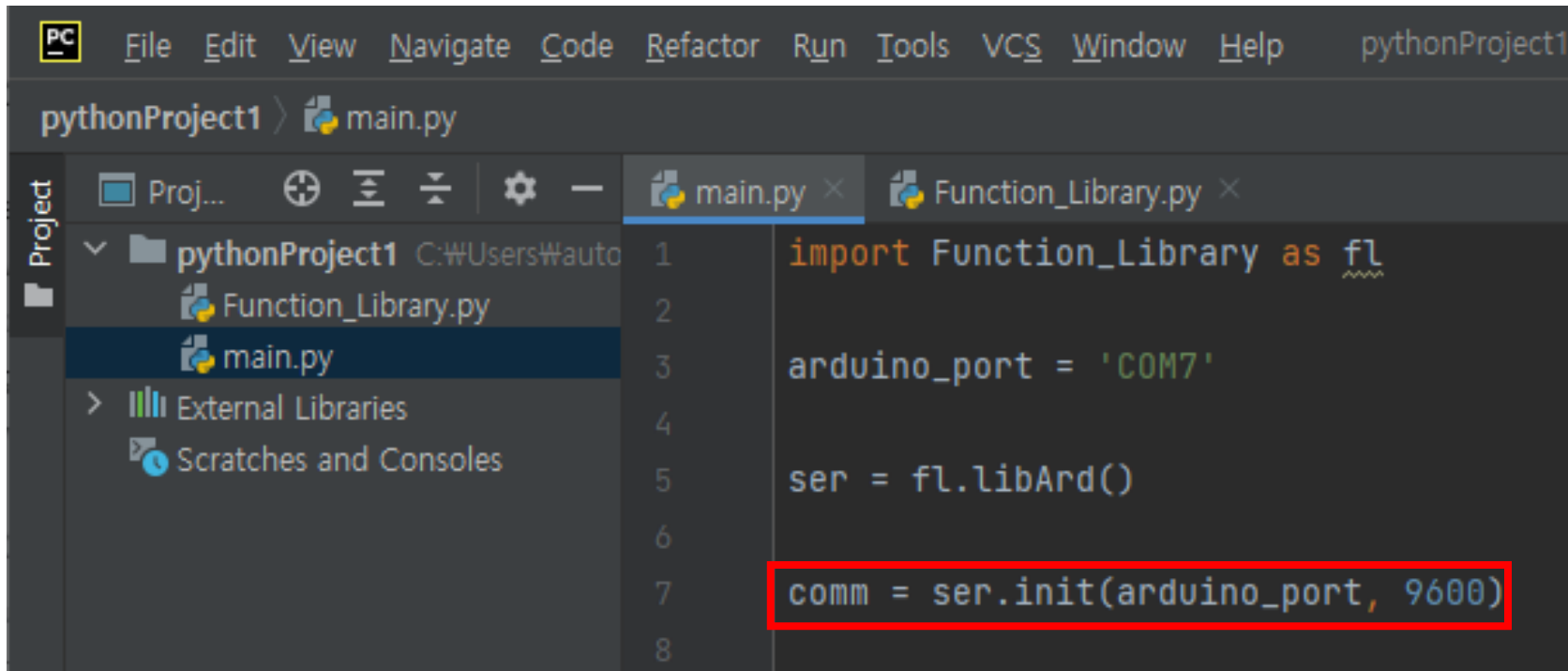


```
pythonProject1 > main.py
1 import Function_Library as fl
2
3 arduino_port = 'COM7'
4
5 ser = fl.libArd()
6
7 comm = ser.init(arduino_port, 9600)
8
```

Exercise 4

■ 시리얼 통신 설정

- ser.init()은 파이썬에서 시리얼 포트 번호 설정 및 통신 속도 설정 함수
- 포트번호 변수와 통신 속도를 입력



The screenshot shows an IDE window titled 'pythonProject1'. The left sidebar displays a project tree with 'pythonProject1' containing 'Function_Library.py' and 'main.py'. The 'main.py' file is selected and open in the editor. The code in 'main.py' is as follows:

```
1 import Function_Library as fl
2
3 arduino_port = 'COM7'
4
5 ser = fl.libArd()
6
7 comm = ser.init(arduino_port, 9600)
8
```

The line `comm = ser.init(arduino_port, 9600)` is highlighted with a red rectangle.

Exercise 4

■ 시리얼 통신 전송

- input(): 지정한 문장을 출력하고 사용자에게 값을 입력 받음
- comm.write(): 입력 받은 값을 시리얼 통신으로 전송

```
while True:
    input_value = input('아두이노로 전송할 저항 값(0~255): ')
    comm.write(input_value.encode())

    if input_value == 'q':
        break
```

Exercise 4

■ 아두이노 컴파일 및 업로드

→ 소스 코드 컴파일 후 아두이노 보드에 업로드



```
sketch_nov12b | 아두이노 1.8.19
파일 편집 스케치 툴 도움말
[Checkmark] [Upload] [New] [Open] [Save]
sketch_nov12b
#include <Car_Library.h>

int val;          // 수신된 값 저장할 변수

void setup() {
    // put your setup code here, to run once:
    Serial.begin(9600);      // 시리얼 통신 시작, 통신 속도 설정
    pinMode(LED_BUILTIN, OUTPUT);    // LED 핀 모드 설정
}

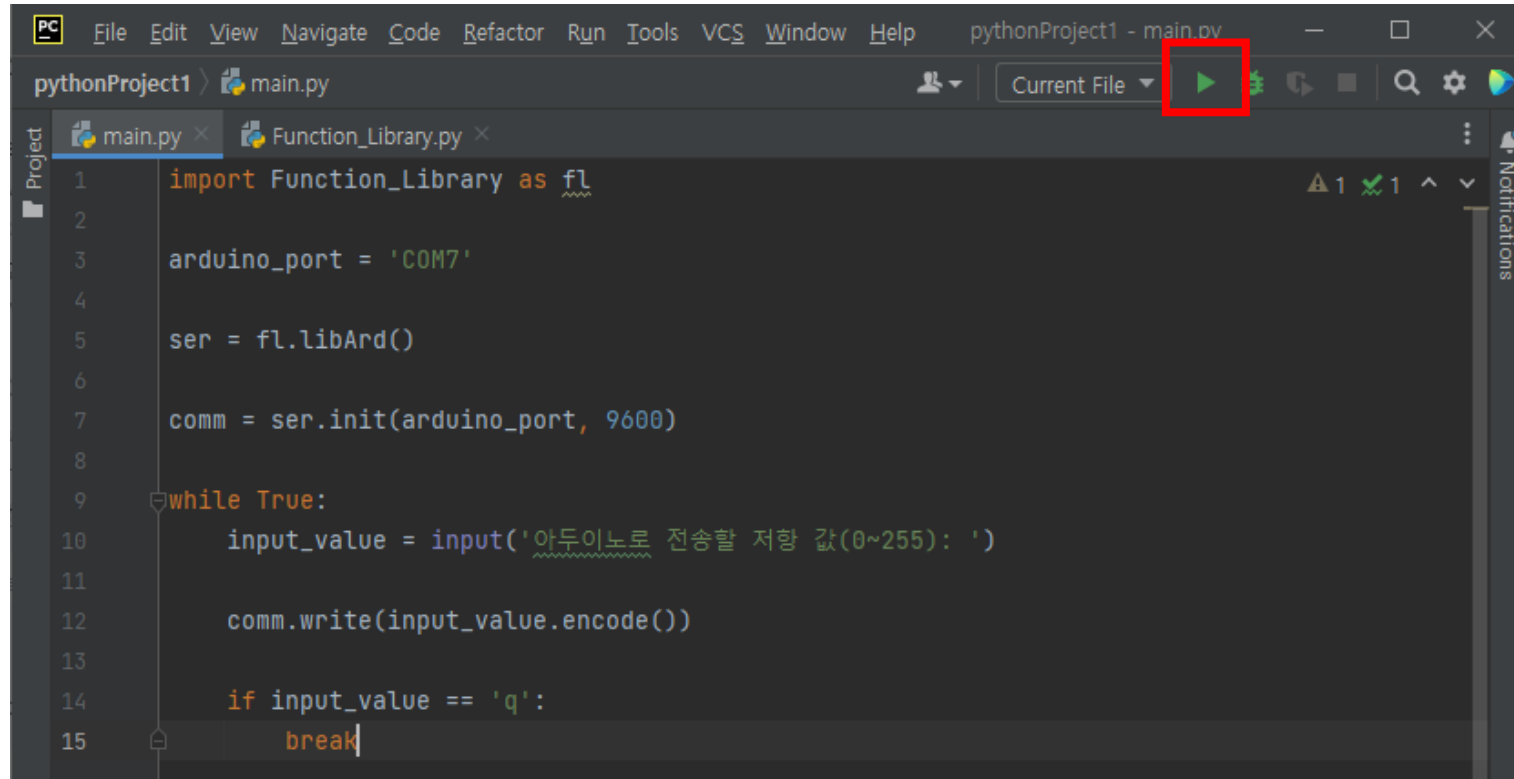
void loop() {
    // put your main code here, to run repeatedly:
    if(Serial.available()) {
        val = Serial.parseInt();

        if(val >= 0) {
            analogWrite(LED_BUILTIN, val);
        }
    }
}
```

Exercise 4

■ 파이썬 소스 코드 실행

- "Run" 버튼 클릭 혹은 Shift + F10을 눌러 소스 코드 실행
- 아두이노 시리얼 모니터가 닫혔는지 꼭 확인(열려 있으면 에러 발생)

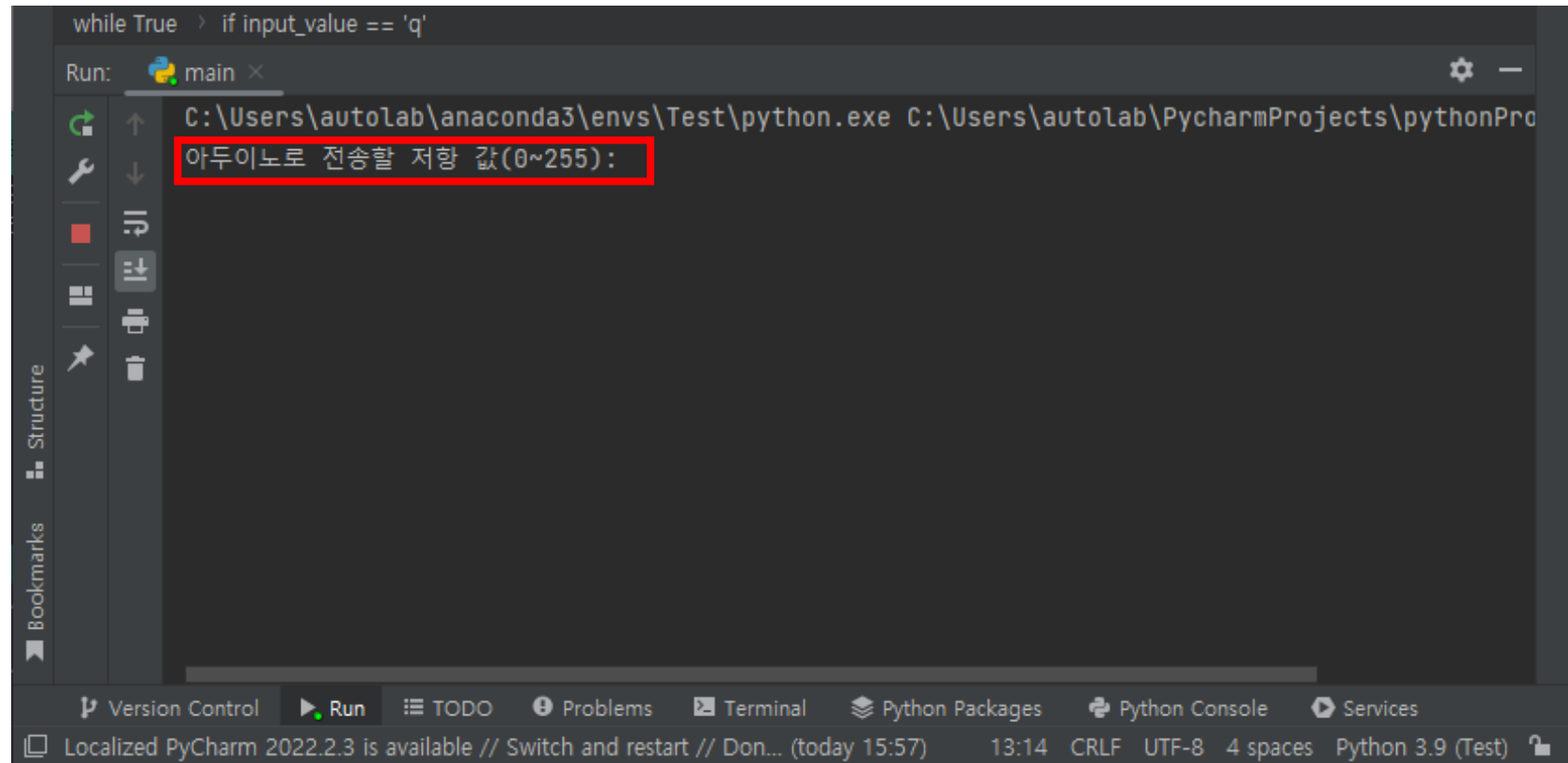


```
pythonProject1 - main.py
pythonProject1 > main.py
main.py x Function_Library.py x
1 import Function_Library as fl
2
3 arduino_port = 'COM7'
4
5 ser = fl.libArd()
6
7 comm = ser.init(arduino_port, 9600)
8
9 while True:
10     input_value = input('아두이노로 전송할 저항 값(0~255): ')
11
12     comm.write(input_value.encode())
13
14     if input_value == 'q':
15         break
```

Exercise 4

■ 저항 값 입력

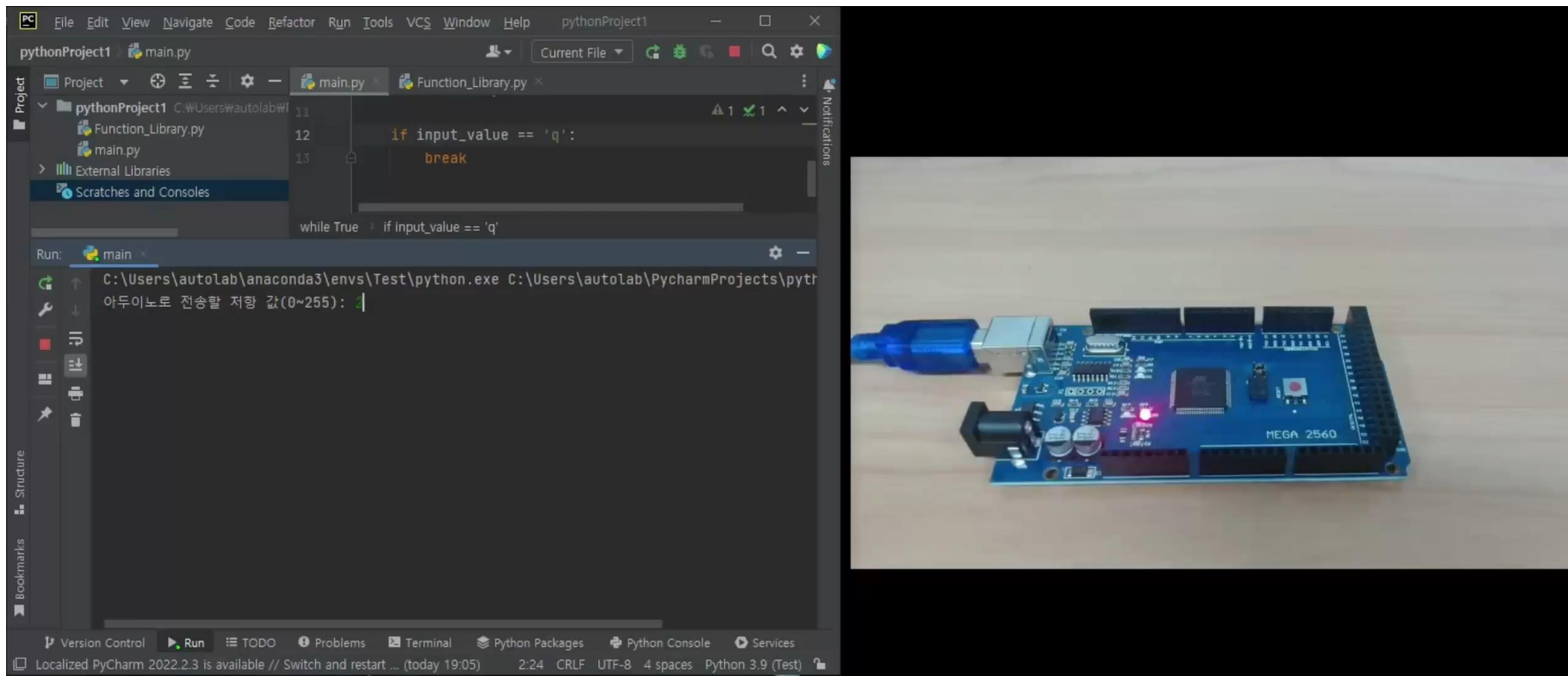
- 실행을 누르면 pycharm 프로그램 하단에 그림과 같은 창 나타남
- 아두이노에 전송할 저항 값을 입력하고 Enter



Exercise 4

■ 결과 확인

→ 입력 값에 따라 LED 밝기 변화 확인



Thank You!

Embedded System Lab.

