

# 양송이농장 데이터 측정 메뉴얼

트레셋

이상호

[sangho@microschool.kr](mailto:sangho@microschool.kr)

# 차례

1. CO2센서 소개
2. 구성품확인
3. CO2 측정하기
4. 데이터 저장하기

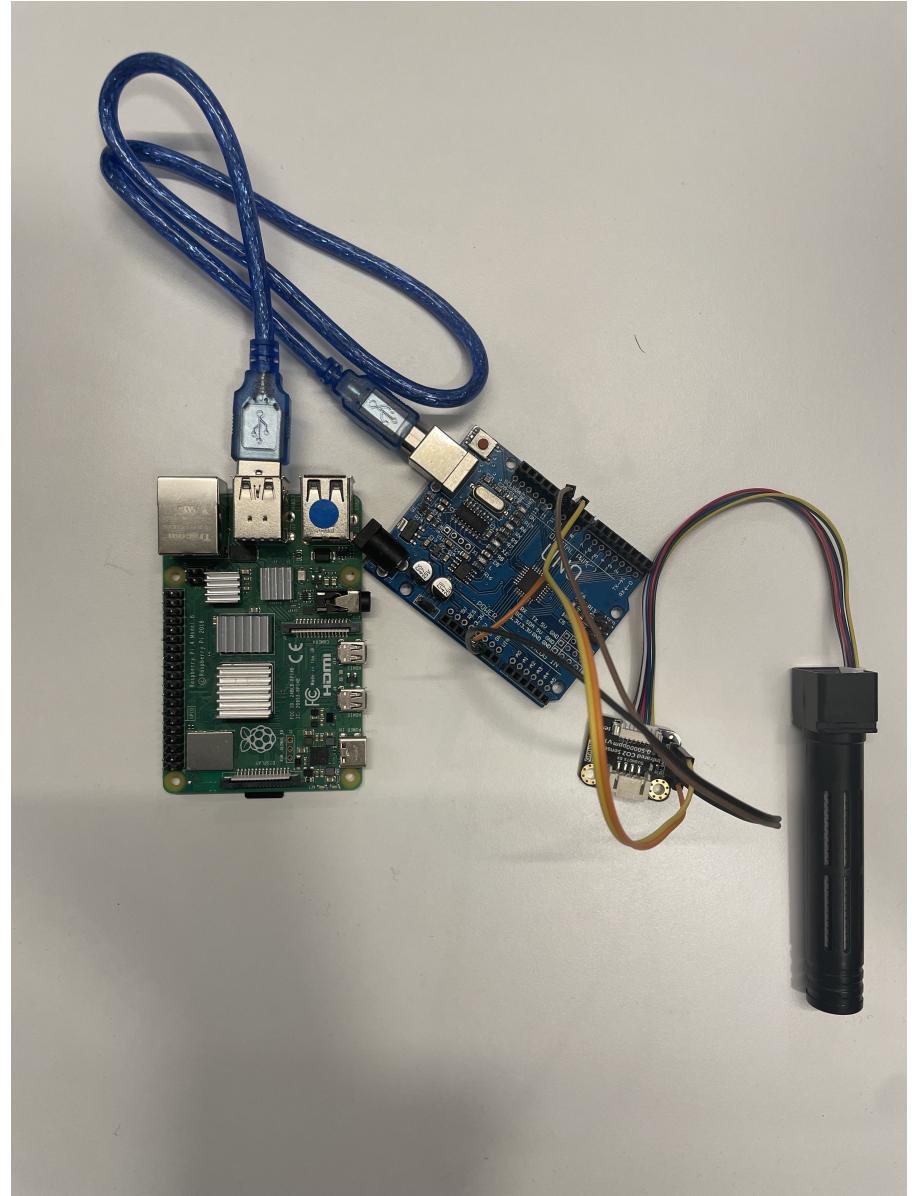
## CO2센서 소개



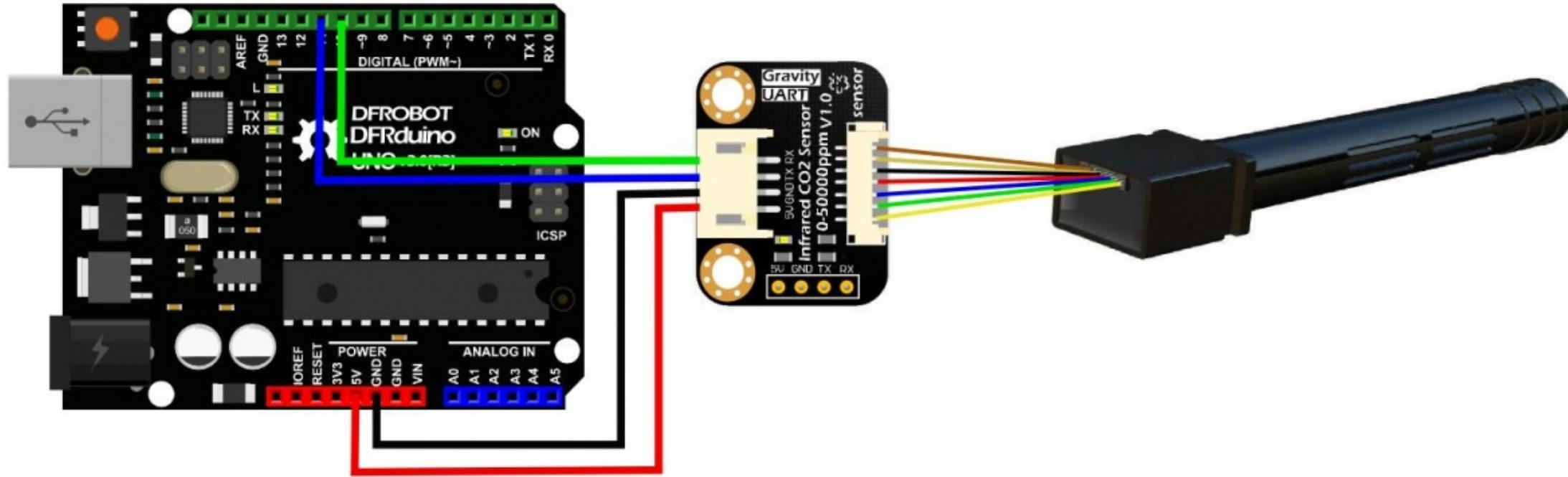
- 가스가 흡수하는 적외선 양으로 CO2를 측정합니다.
- 특정 가스 농도를 구체적으로 구분 해서 측정합니다.
- 주변의 산소농도나 온도에 영향을 받지 않습니다.
- 유효범위가 0~50000pm으로 광범위합니다.  
(기존의 센서는 400~5000pm)

# 구성품확인

1. 라즈베리파이 4
2. 아두이노
3. 아두이노케이블
4. CO2 센서



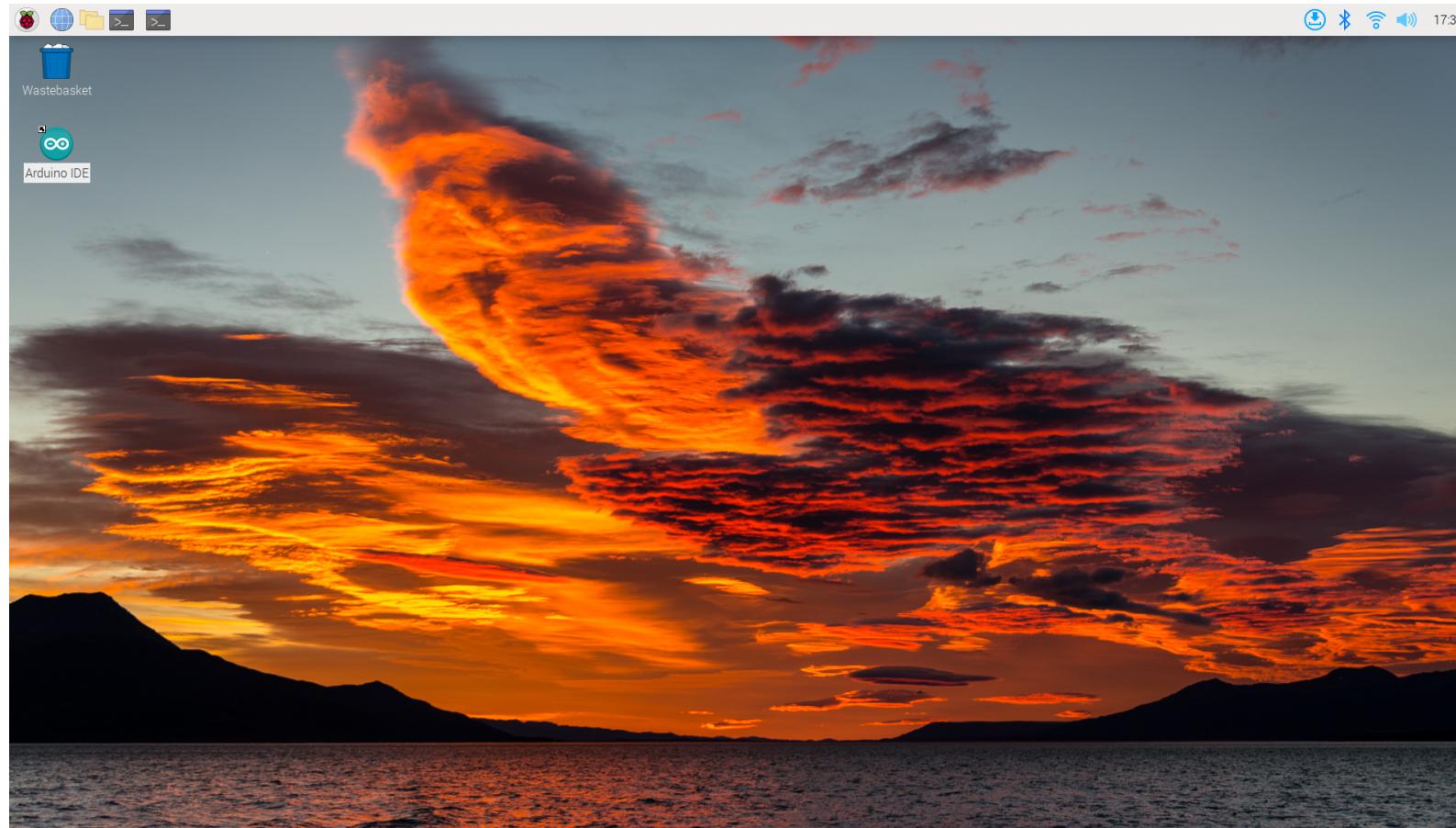
# 회로도



참고: 이미 연결된 상태이므로 센서가 아두이노의 10과 11번 핀에 연결된다는 것만 확인하면 됩니다.

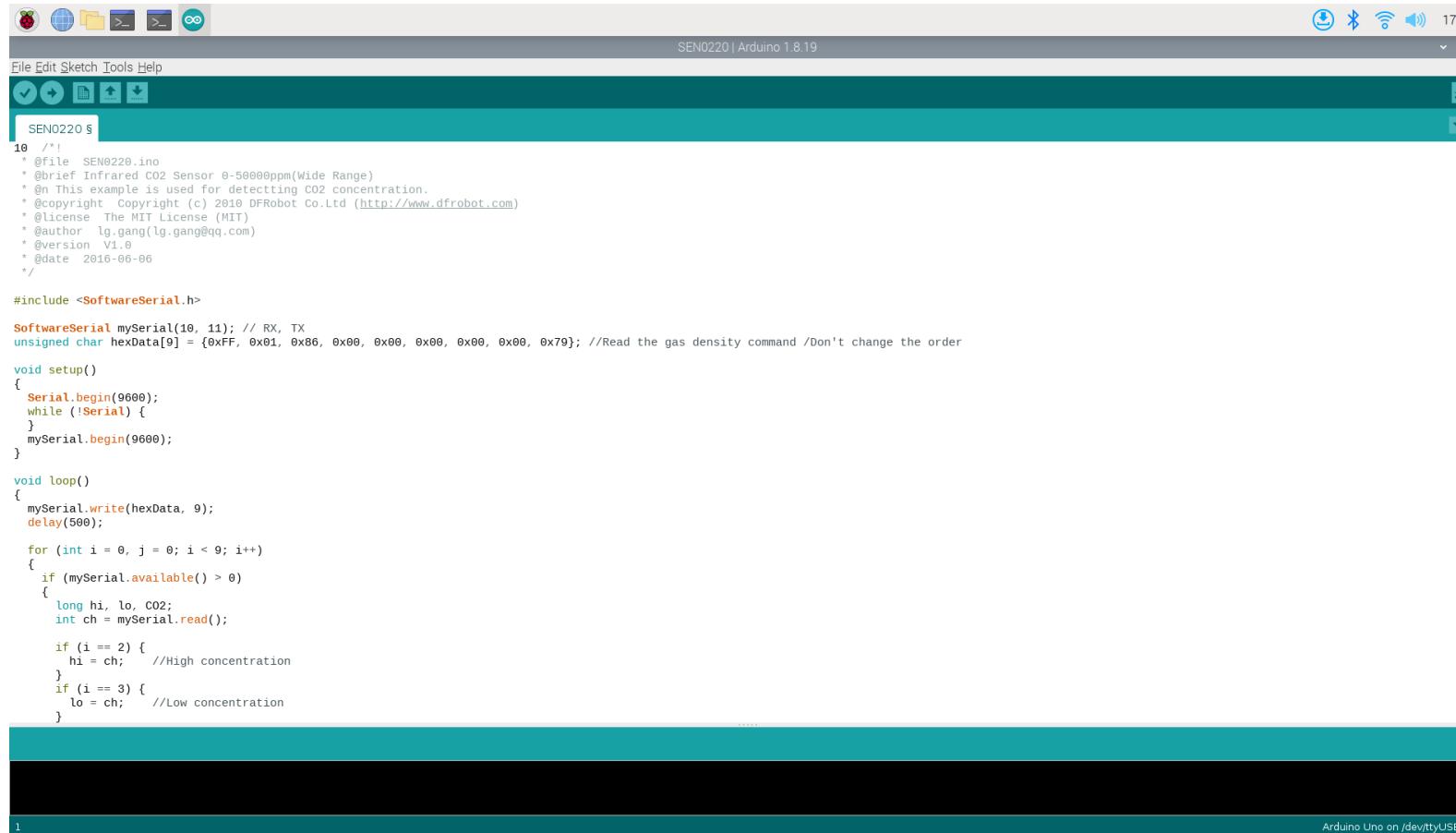
# CO<sub>2</sub> 측정하기

# 라즈베리파이 실행하기



전에 보내드렸던 마우스와 모니터를 연결한후에 전원케이블을 연결합니다.

# 아두이노 실행하기



The screenshot shows the Arduino IDE interface with the title bar "SENO220 | Arduino 1.8.19". The menu bar includes File, Edit, Sketch, Tools, and Help. Below the menu is a toolbar with various icons. The main area displays the code for "SENO220.ino". The code uses SoftwareSerial to communicate with a CO2 sensor. It initializes the serial port at 9600 bps, sends a command to read gas density, and then reads the response to extract CO2 concentration values (hi and lo) from bytes 2 and 3.

```
10 /*!
 * @file SENO220.ino
 * @brief Infrared CO2 Sensor 0-5000ppm(Wide Range)
 * @n This example is used for detecting CO2 concentration.
 * @copyright Copyright (c) 2016 DFRobot Co.Ltd (http://www.dfrobot.com)
 * @license The MIT License (MIT)
 * @author lg.gang(lg.gang@qq.com)
 * @version V1.0
 * @date 2016-06-06
 */
#include <SoftwareSerial.h>
SoftwareSerial mySerial(10, 11); // RX, TX
unsigned char hexData[9] = {0xFF, 0x01, 0x86, 0x00, 0x00, 0x00, 0x00, 0x00, 0x79}; //Read the gas density command /Don't change the order
void setup()
{
  Serial.begin(9600);
  while (!Serial) {
  }
  mySerial.begin(9600);
}
void loop()
{
  mySerial.write(hexData, 9);
  delay(500);
  for (int i = 0, j = 0; i < 9; i++)
  {
    if (mySerial.available() > 0)
    {
      long hi, lo, co2;
      int ch = mySerial.read();
      if (i == 2) {
        hi = ch; //High concentration
      }
      if (i == 3) {
        lo = ch; //Low concentration
      }
      .....
    }
  }
}
```

Arduino Uno on /dev/ttyUSB0

라즈베리파이 바탕화면에 있는 아두이노를 실행합니다.

Compile과 Upload를 통해 아두이노에 코드를 넣어줍니다.

# 아두이노 코드

```
#include <SoftwareSerial.h>

SoftwareSerial mySerial(10, 11); // RX, TX
unsigned char hexData[9] = {0xFF, 0x01, 0x86, 0x00, 0x00, 0x00, 0x00, 0x00, 0x79}; //Read the gas density command /Don't change the order

void setup()
{
    Serial.begin(9600);
    while (!Serial) {
    }
    mySerial.begin(9600);
}

void loop()
{
    mySerial.write(hexData, 9);
    delay(500);

    for (int i = 0, j = 0; i < 9; i++)
    {
        if (mySerial.available() > 0)
        {
            long hi, lo, CO2;
            int ch = mySerial.read();

            if (i == 2) {
                hi = ch;      //High concentration
            }
            if (i == 3) {
                lo = ch;      //Low concentration
            }
            if (i == 8) {
                CO2 = hi * 256 + lo; //CO2 concentration
                //Serial.print("CO2 concentration: ");
                //Serial.print(CO2);
                Serial.println(CO2);
                //Serial.println("ppm");
            }
        }
    }
}
```

아두이노의 시리얼 모니터로 0.5초 간격으로 CO2의 ppm 농도를 출력합니다.

# 측정시간을 1분으로 변경하기

```
#include <SoftwareSerial.h>

SoftwareSerial mySerial(10, 11); // RX, TX
unsigned char hexData[9] = {0xFF, 0x01, 0x86, 0x00, 0x00, 0x00, 0x00, 0x00, 0x79}; //Read the gas density command /Don't change the order

void setup()
{
    Serial.begin(9600);
    while (!Serial) {
    }
    mySerial.begin(9600);
}

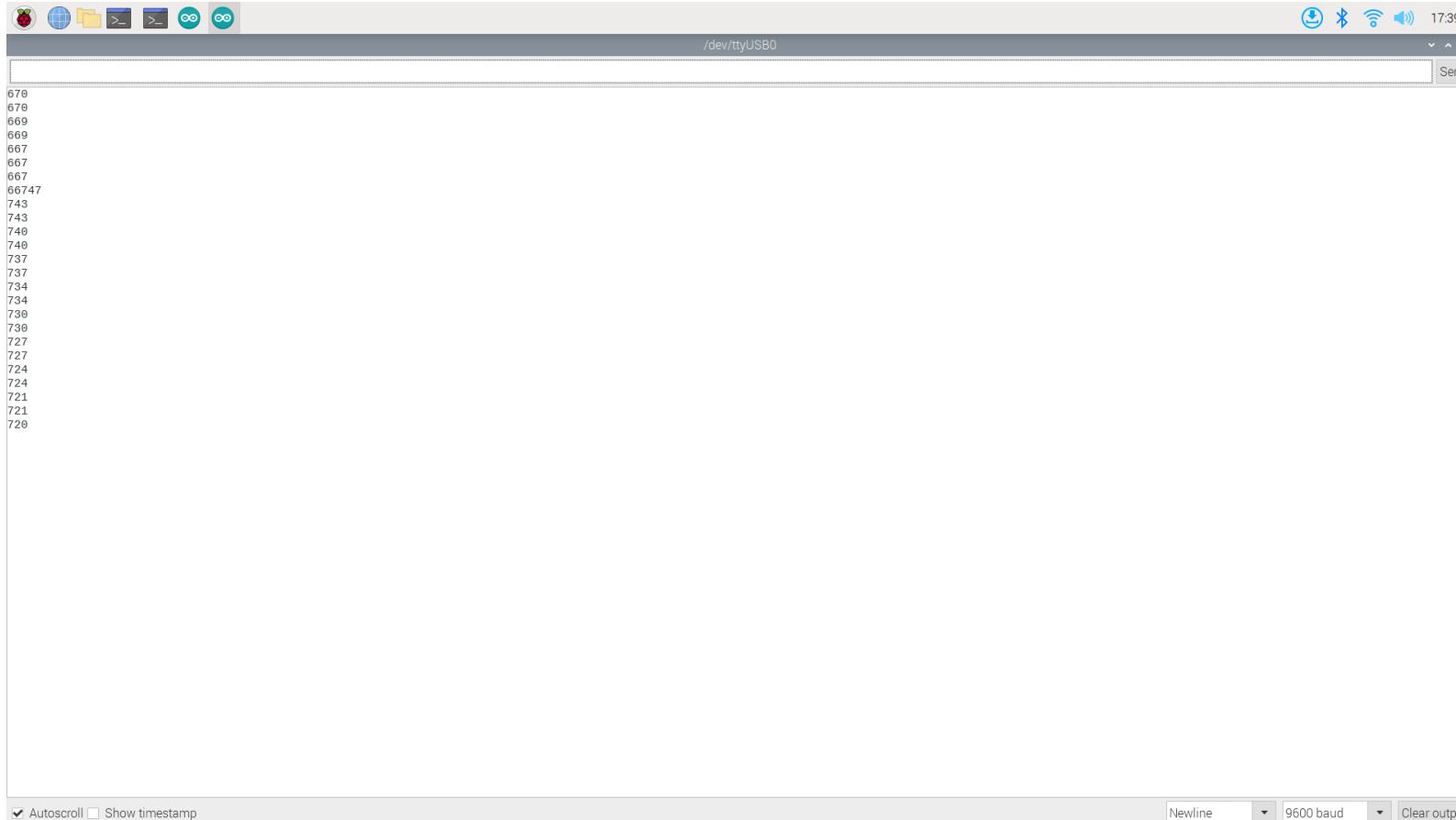
void loop()
{
    mySerial.write(hexData, 9);
    delay(6000);

    for (int i = 0, j = 0; i < 9; i++)
    {
        if (mySerial.available() > 0)
        {
            long hi, lo, CO2;
            int ch = mySerial.read();

            if (i == 2) {
                hi = ch;      //High concentration
            }
            if (i == 3) {
                lo = ch;      //Low concentration
            }
            if (i == 8) {
                CO2 = hi * 256 + lo; //CO2 concentration
                //Serial.print("CO2 concentration: ");
                //Serial.print(CO2);
                Serial.println(CO2);
                //Serial.println("ppm");
            }
        }
    }
}
```

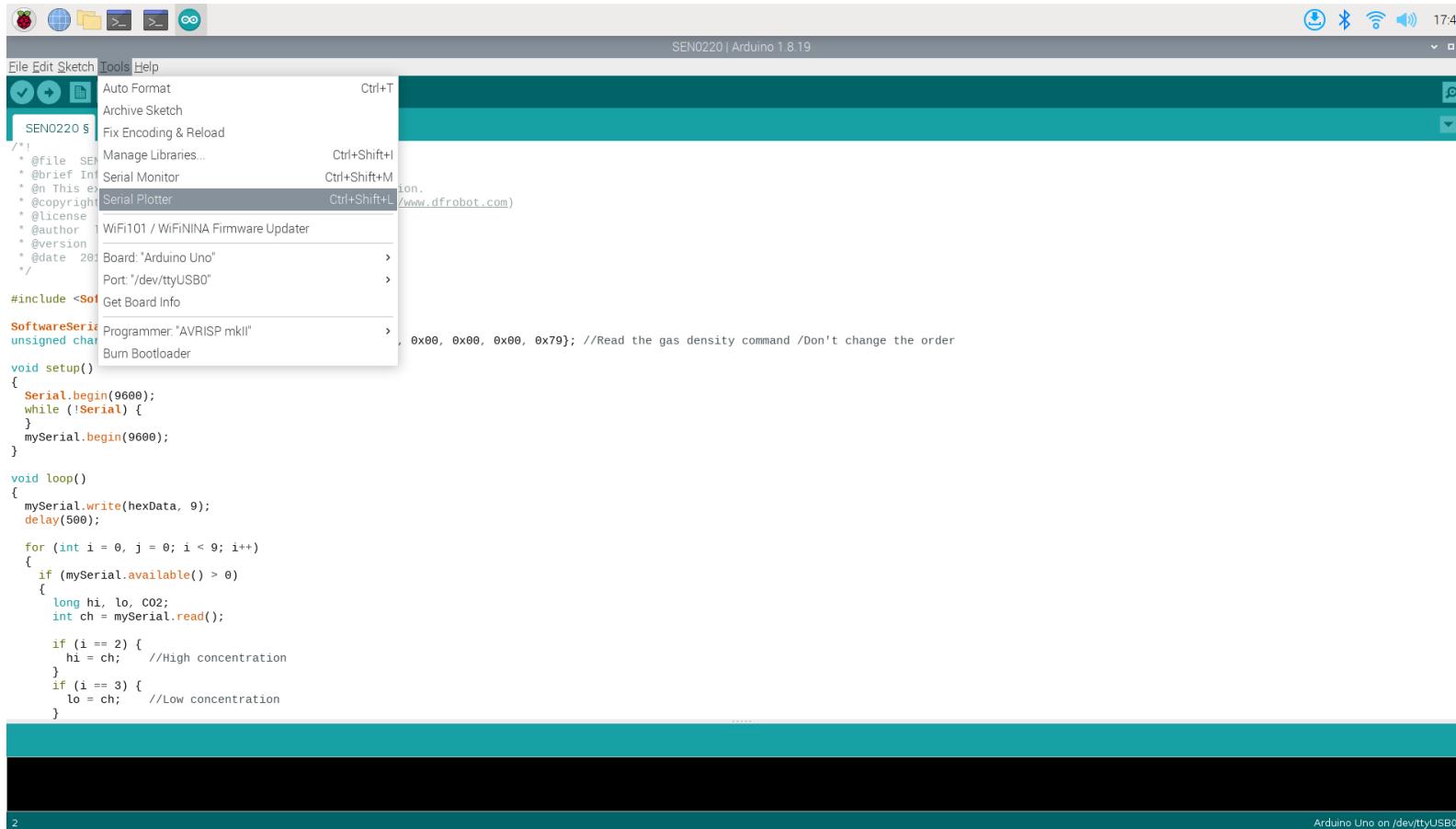
아두이노 코드에서 `delay(500)`을 `delay(6000)`로 변경하고 **Compile**한 후에 **Upload**하면 변경됩니다.

# 시리얼 모니터로 데이터 확인하기



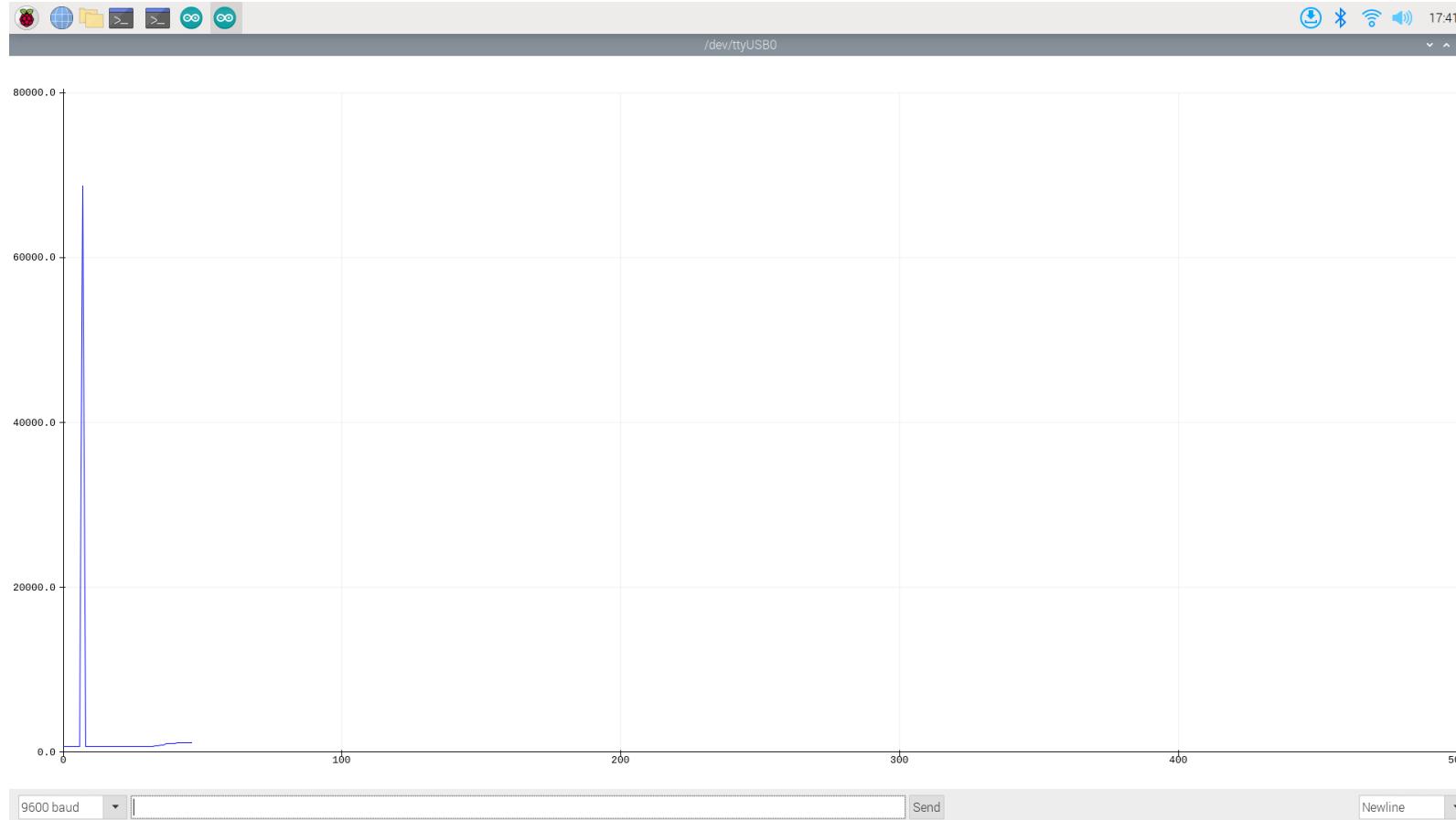
아두이노의 우측 상단에 돌보기 모양 아이콘을 누르면 데이터가 출력되는 것을 확인할 수 있습니다.

# 시리얼 플로터 실행하는 법



Tools에서 Serial Plotter를 실행합니다.

# 시리얼 플로터 데이터 확인하기



시리얼 플로터로 데이터의 추이를 그래프로 확인할 수 있습니다.

# 데이터 저장하기

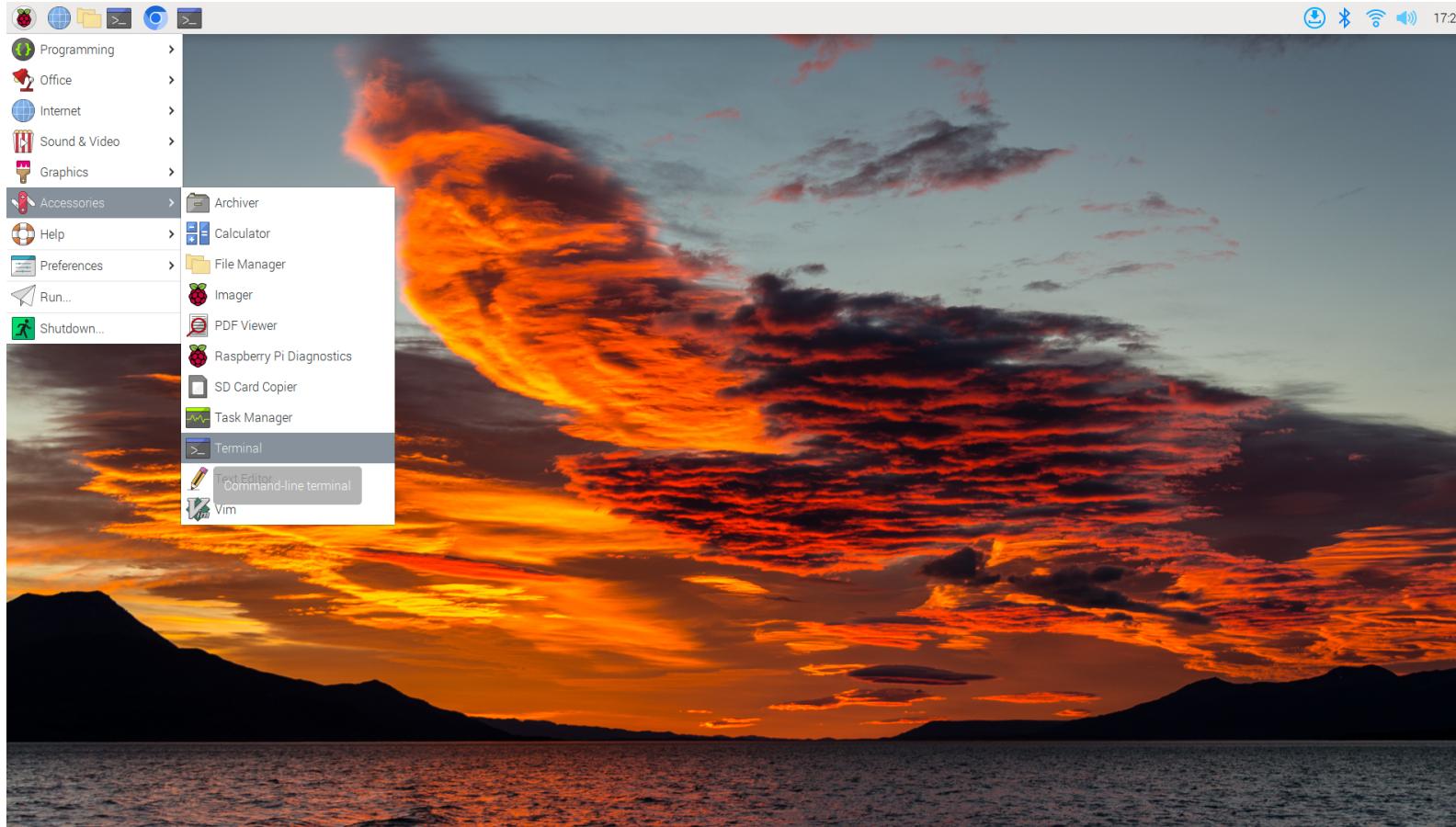
# 소개

- 아두이노에 연결된 센서로 부터 데이터를 라즈베리파이에 저장합니다.

# 실행방법

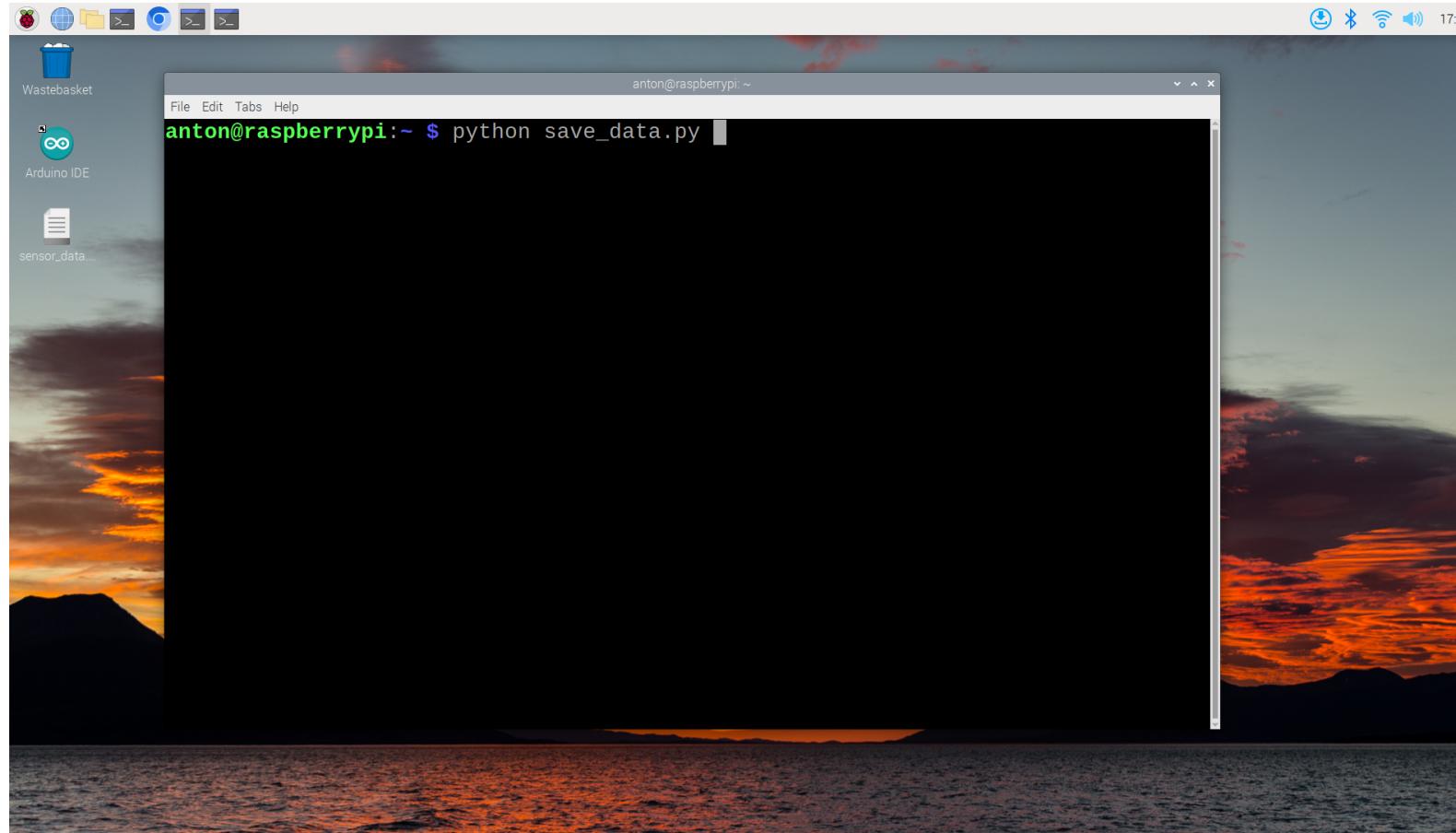
1. 터미널창 열기
2. Python 파일 실행하기
3. 저장된 파일 열기
4. 데이터를 그래프로 만들기

# 터미널창을 열기



좌측상단의 라즈베리파이를 누르고 Accessories 안에 Terminal을 선택합니다.

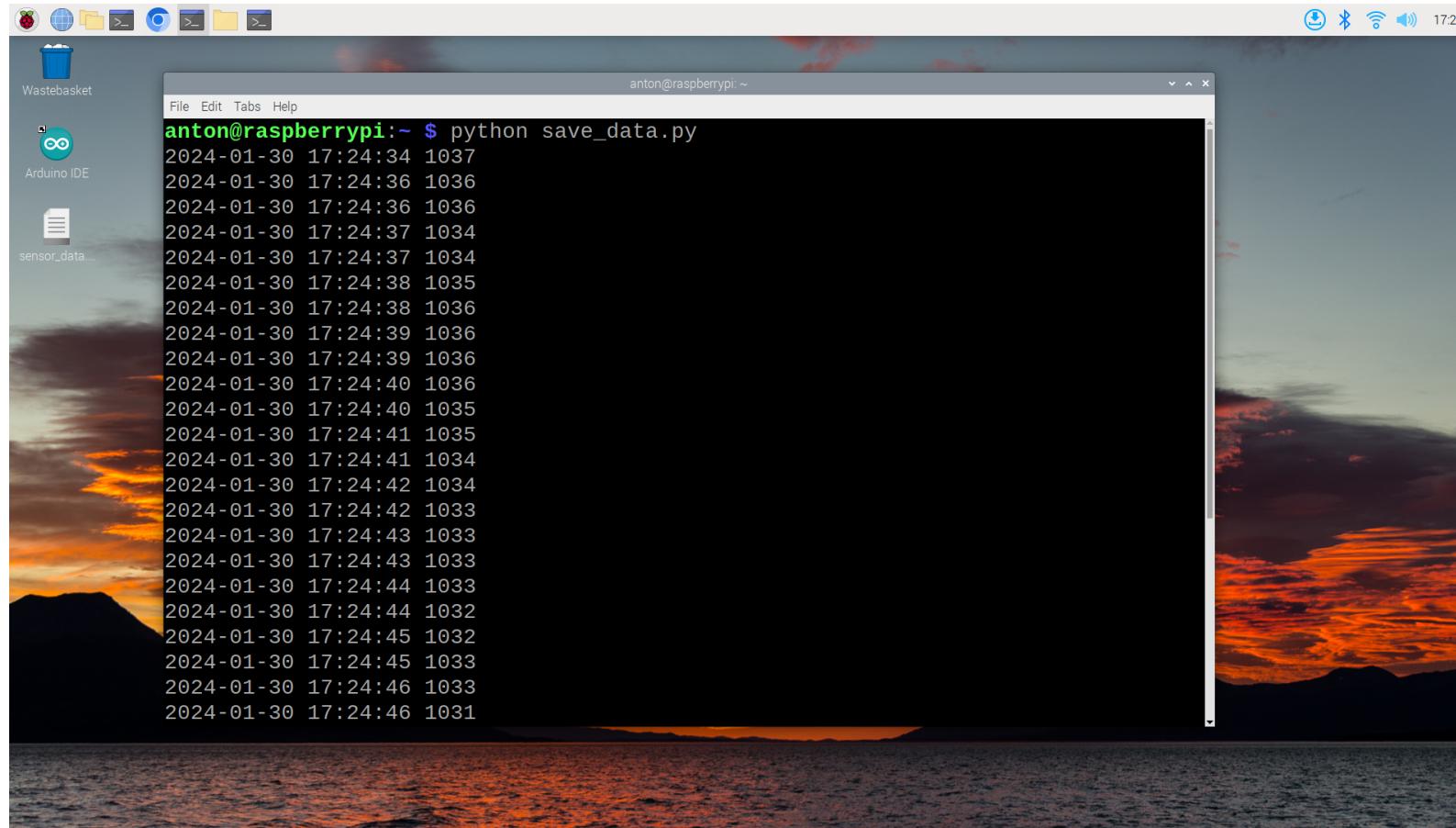
# python 파일을 실행하기



아래 명령어를 실행합니다.

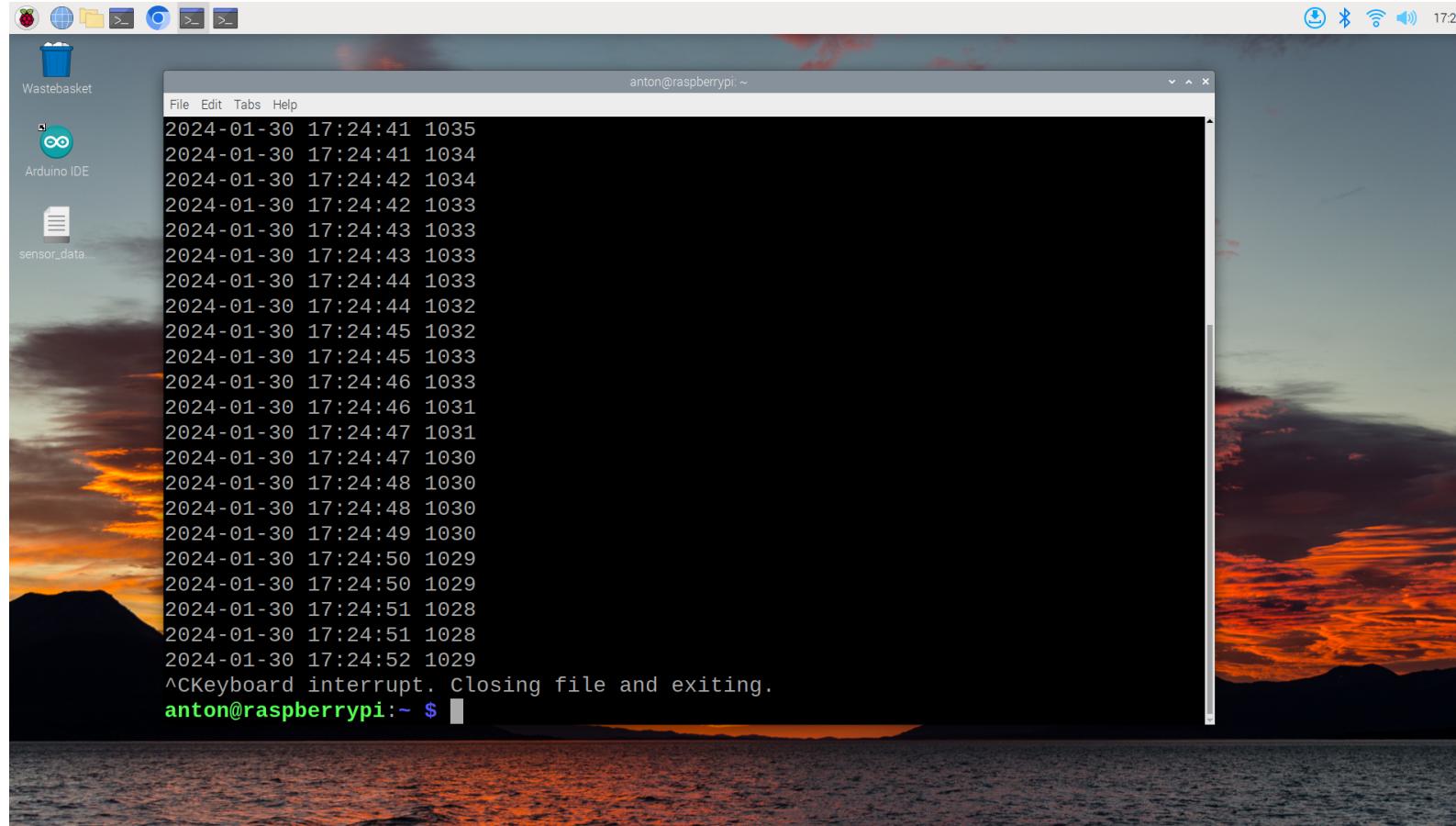
```
python save_data.py
```

# 터미널창에 데이터 나오는 것을 확인하기



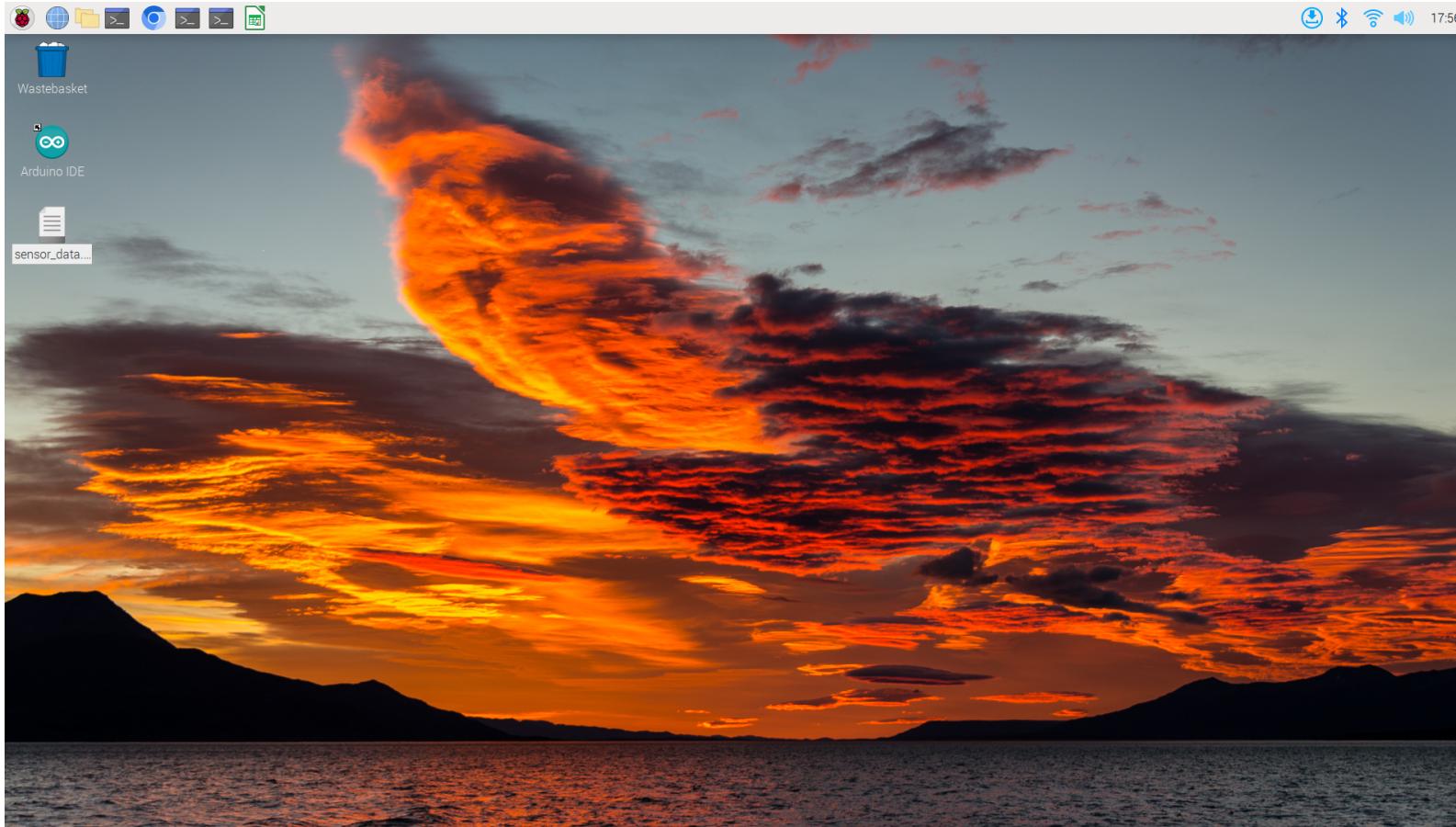
터미널에서 데이터가 1초 단위로 나오면서 저장을 시작합니다.

# 데이터가 저장되는 것을 중지시키기



Ctrl + C 명령어로 데이터저장을 종료시킵니다.

# 저장된 데이터가 있는 파일 열기



바탕화면에 있는 `sensor_data.csv` 파일을 더블클릭해서 실행합니다.

# 엑셀파일열기 (LibOffice)

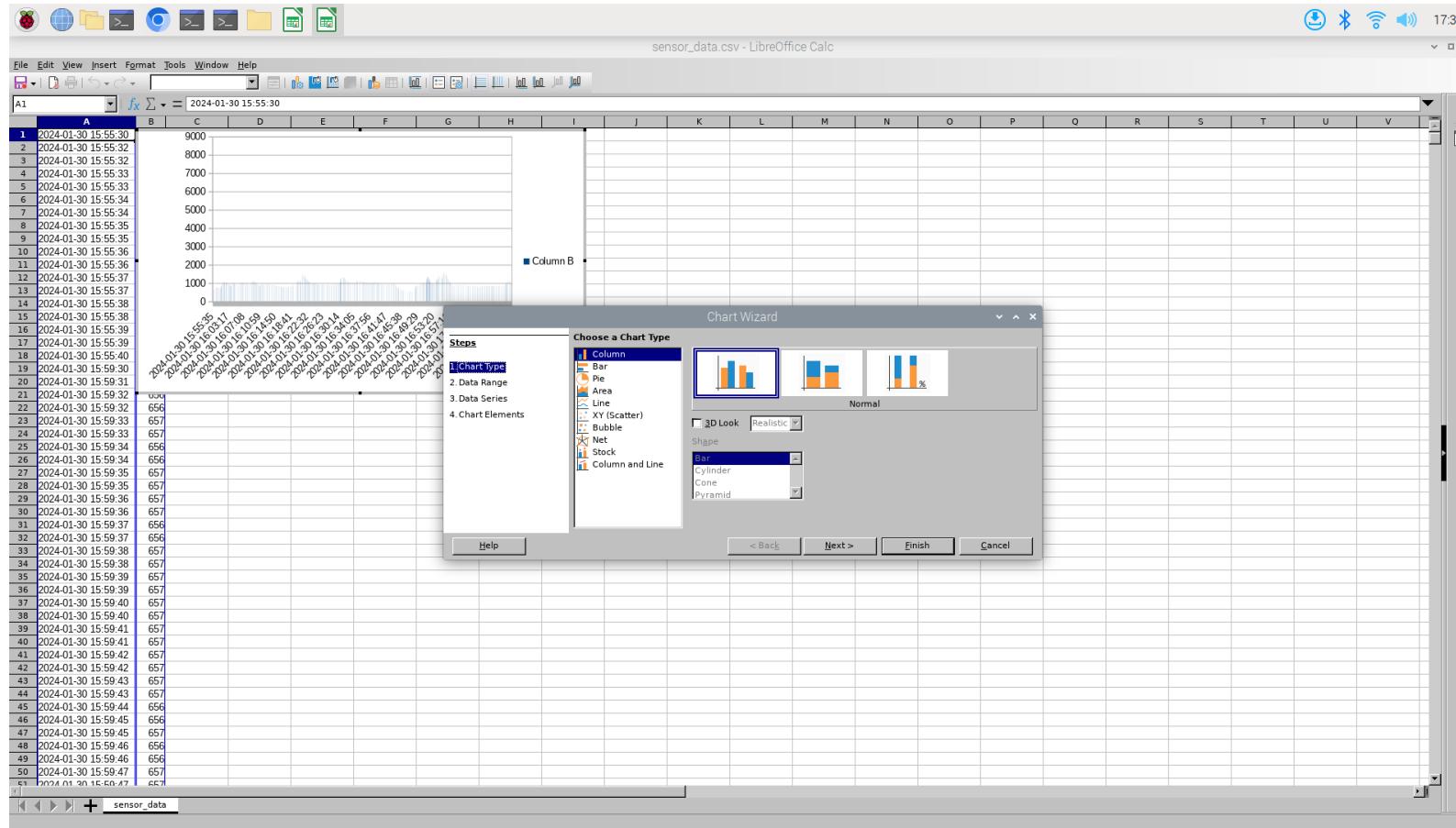
The screenshot shows the LibreOffice Calc application window. The title bar reads "sensor\_data.csv - LibreOffice Calc". The status bar at the bottom right shows the time as 17:31. The main spreadsheet area displays a single column of data starting from row 1. The first few rows of data are as follows:

	A
1	2024-03-30 15:55:30
2	761
3	2024-03-30 15:55:32
4	762
5	2024-03-30 15:55:32
6	762
7	2024-03-30 15:55:33
8	762
9	2024-03-30 15:55:34
10	762
11	2024-03-30 15:55:36
12	762
13	2024-03-30 15:55:37
14	762
15	2024-03-30 15:55:38
16	762
17	2024-03-30 15:55:39
18	764
19	2024-03-30 15:55:40
20	764
21	2024-03-30 15:59:30
22	657
23	2024-03-30 15:59:31
24	657
25	2024-03-30 15:59:32
26	656
27	2024-03-30 15:59:32
28	656
29	2024-03-30 15:59:33
30	657
31	2024-03-30 15:59:33
32	657
33	2024-03-30 15:59:34
34	656
35	2024-03-30 15:59:34
36	656
37	2024-03-30 15:59:35
38	657
39	2024-03-30 15:59:37
40	656
41	2024-03-30 15:59:38
42	657
43	2024-03-30 15:59:38
44	657
45	2024-03-30 15:59:43
46	657
47	2024-03-30 15:59:44
48	656
49	2024-03-30 15:59:45
	656
	2024-03-30 15:59:45
	657
	2024-03-30 15:59:46
	656
	2024-03-30 15:59:46
	656

엑셀파일처럼 열리는 것을 확인합니다.

\*데이터는 마지막 기록시간에서 추가됩니다.

# 데이터를 차트로 만들기



상단의 메뉴바에서 Insert 안에있는 Chart를 실행합니다.

# 차트형태를 선으로 만들기

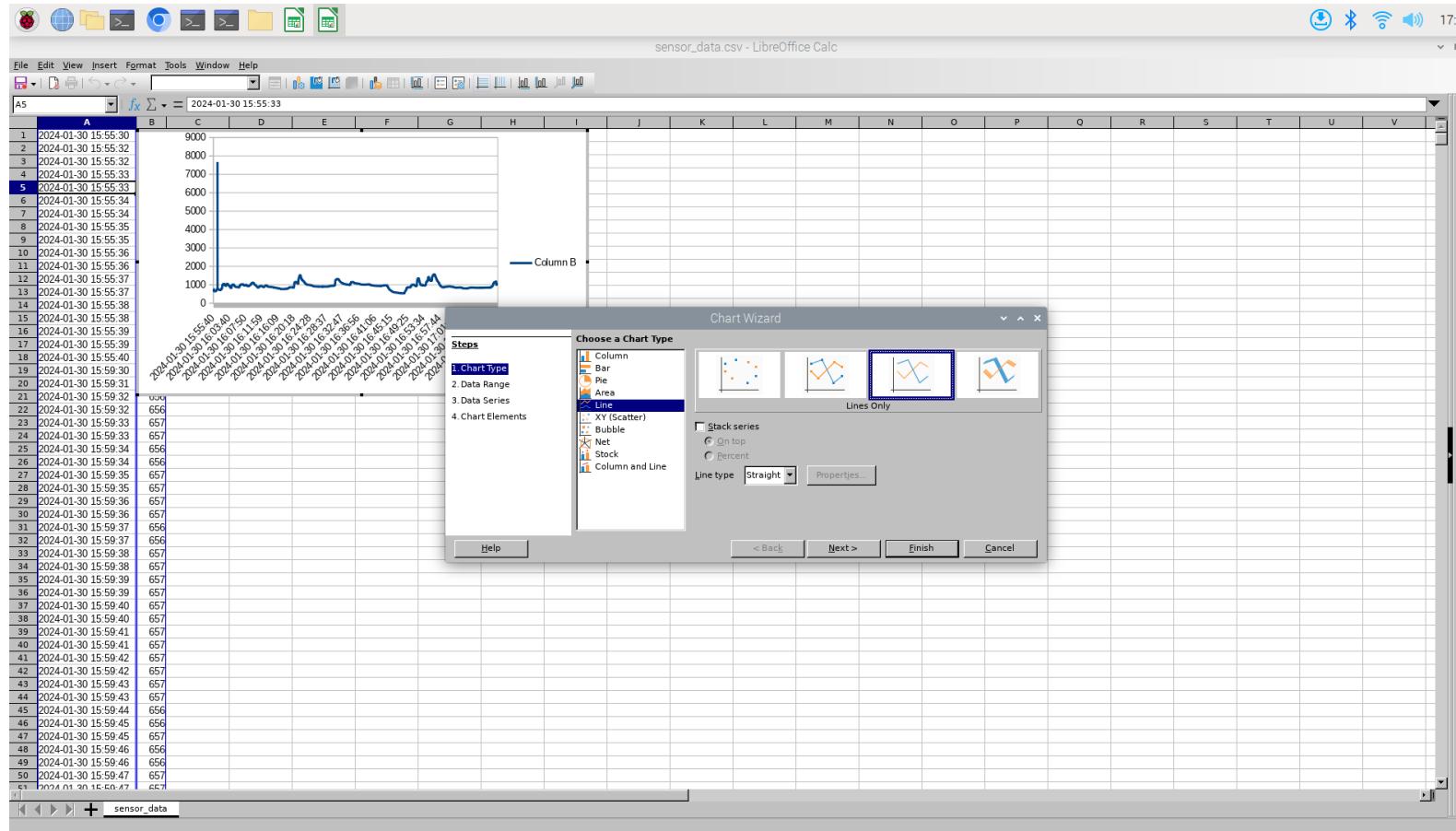
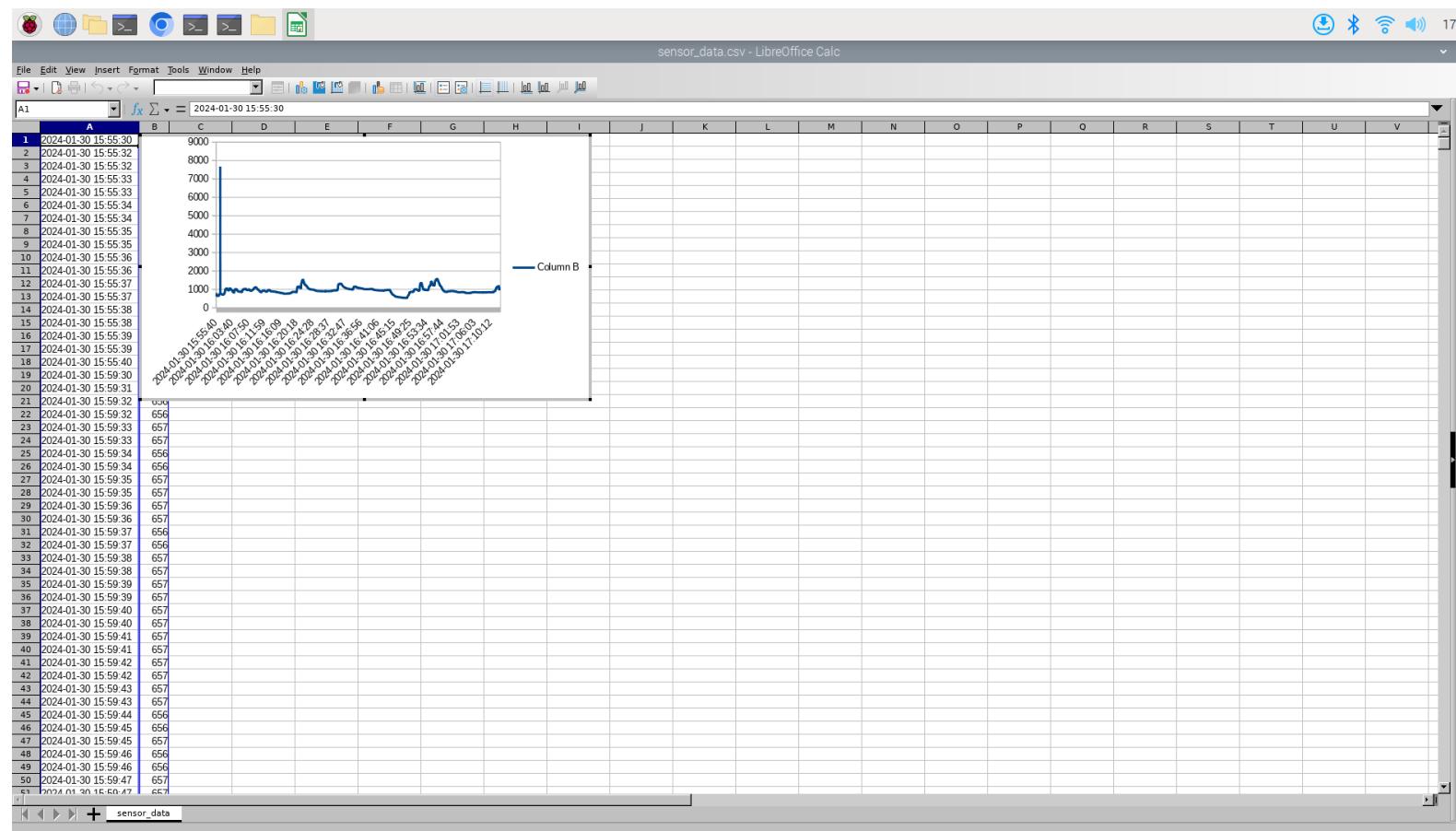


Chart 형태를 선으로 변경합니다.

# 완성



# 참고사항

1. 초반 3분 결과는 예열시간에 나오는 데이터이므로 무시해도됩니다.
2. 센서가 필터링 알고리즘을 내부에서 실행하므로 평균값을 출력합니다.
3. 데이터 저장하는 python 프로그램을 실행할 때는 Arduino IDE를 종료시켜야합니다.
4. 데이터 구분하려면 senser\_data.csv파일을 새로운 이름으로 저장합니다.

예)sensor\_data\_1.csv로 저장

## 요청사항

1. CO<sub>2</sub>값의 추이를 확인하면서 센서성능을 확인부탁합니다.
2. 적당한 데이터 저장 횟수와 시간을 정해주세요.

감사합니다.