python™

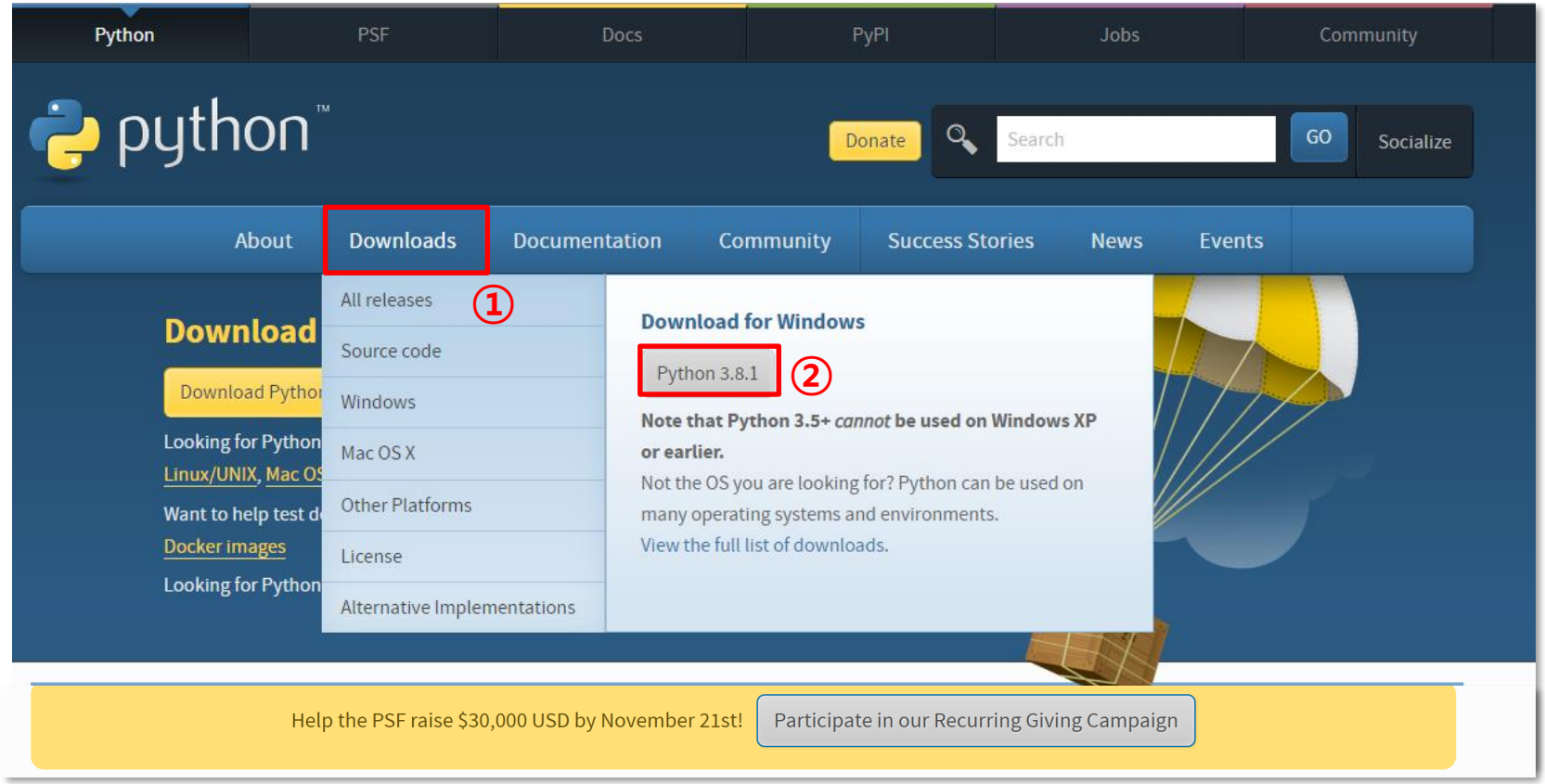
CoDrone Python

Robot Edutainment Company
ROBOLINK

INDEX

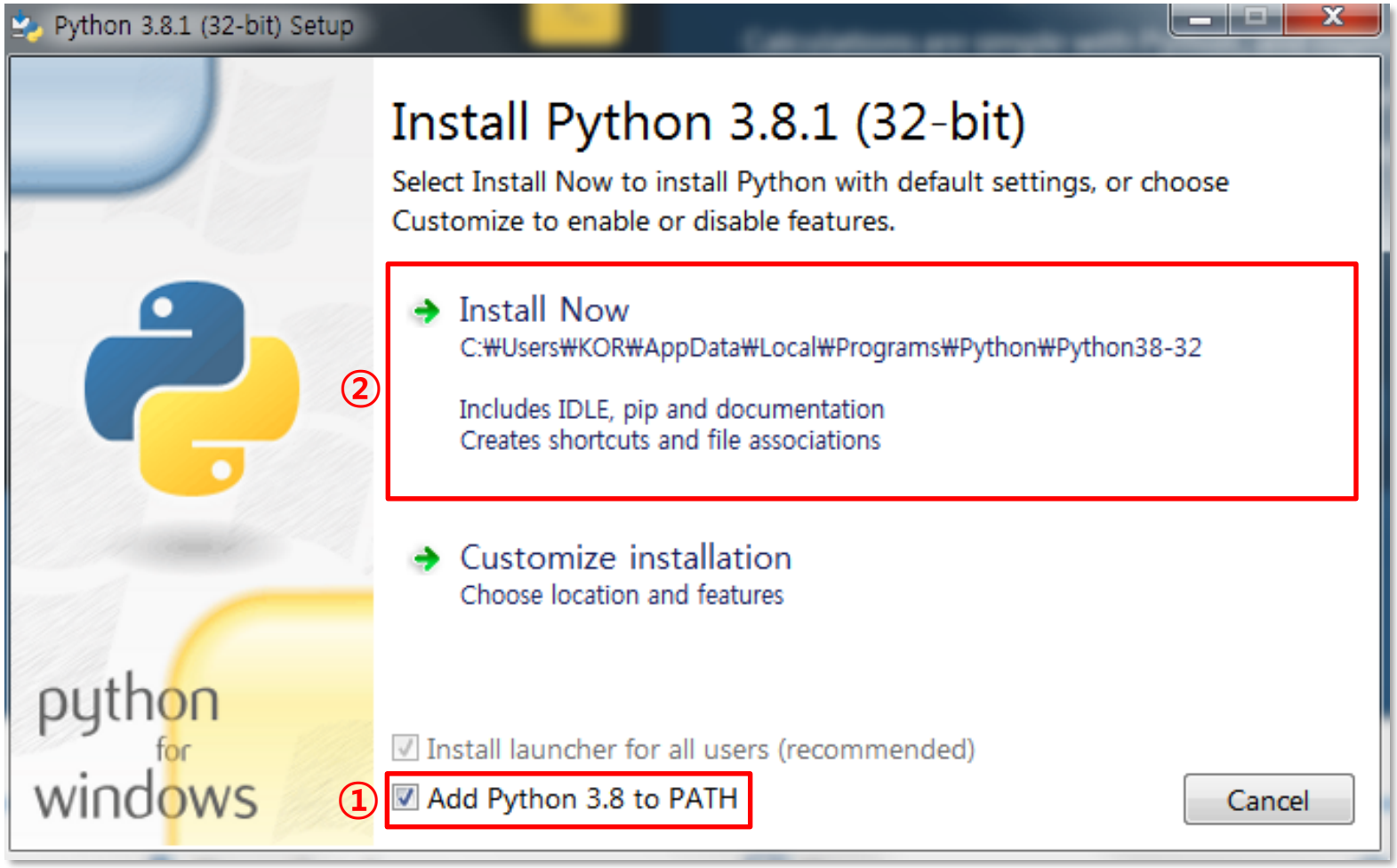
Python Installation

1. Python Installation



https://www.python.org/downloads/

1. Python Installation



Installation Library

2. Installation Library

PIP Python PIP : Package Manager

◆ **What's PIP?**

'**PIP**' is a package for Python packages, and 'Python' is easy to install and update the various software and libraries you need if you're online.

* If Python ver 3.4 or higher is installed, PIP is included by default.

◆ **What's Package?**

The '**Package**' includes everything needed for the module,
and '**Modules**' are libraries of Python code for inclusion in the project

2. Installation Library

PIP Upgrade

cmd.exe

```
C:₩Users₩metal>python -m pip install --upgrade pip
Collecting pip
  Downloading https://files.pythonhosted.org/packages/54/0c/d01aa759fdc5
2ce14b5845662c13f3/pip-20.0.2-py2.py3-none-any.whl (1.4MB)
    100% |                                | 1.4MB 3.9MB/s
Installing collected packages: pip
  Found existing installation: pip 10.0.1
    Uninstalling pip-10.0.1:
      Successfully uninstalled pip-10.0.1
Successfully installed pip-20.0.2

C:₩Users₩metal>
```

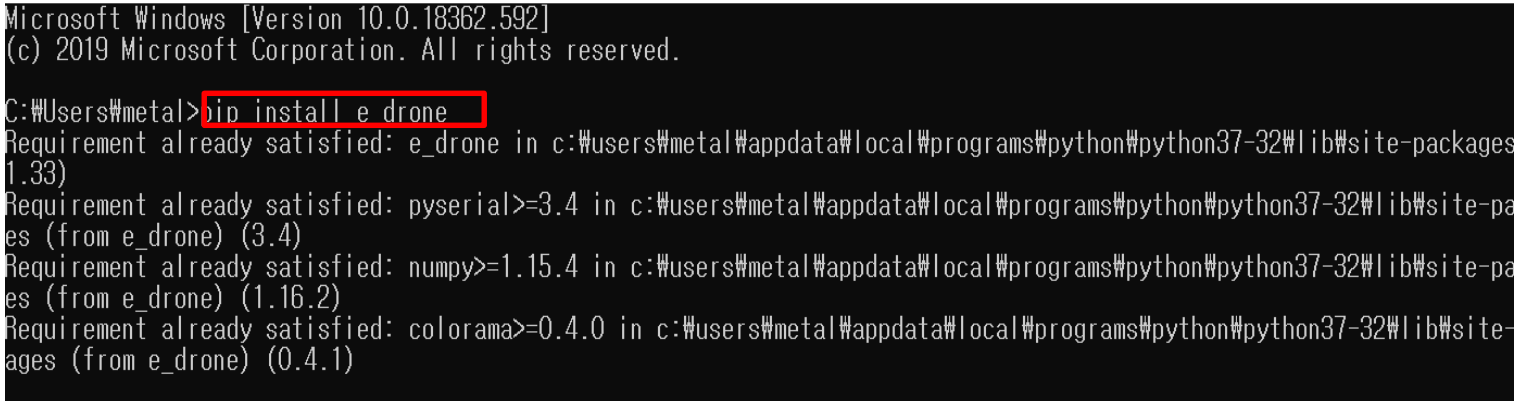
If the 'pip' version is older, we will upgrade it.

Enter **'Python –m pipe install –upgrade pipe'** to upgrade 'pip'.

2. Installation Library

Install e_drone

cmd.exe



Enter '**pip install e_drone**' to install the library.

If installed, the currently installed version information is output.

2. Installation Library

Upgrade e_drone

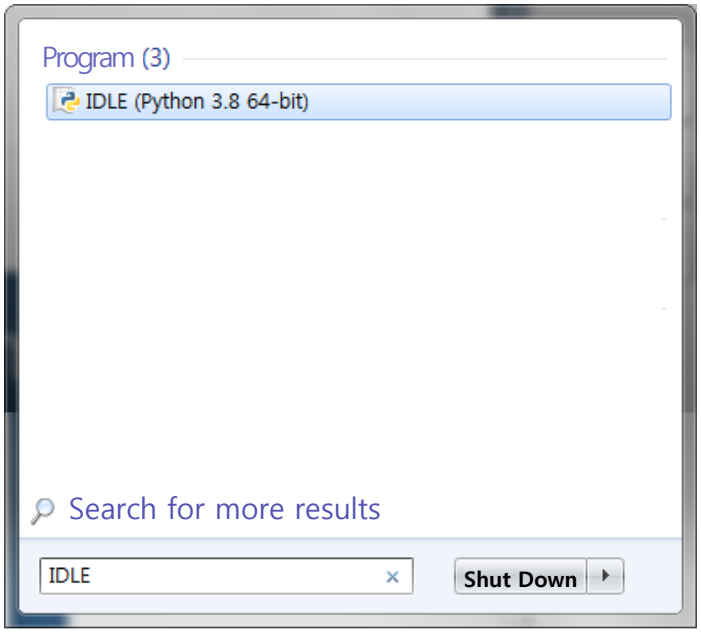
cmd.exe

```
C:₩Users₩metal>pip install --upgrade e_drone
Requirement already up-to-date: e_drone in c:₩users₩metal₩appdata₩local₩programs₩
.1.33)
Requirement already satisfied, skipping upgrade: pyserial>=3.4 in c:₩users₩metal₩
32₩lib₩site-packages (from e_drone) (3.4)
Requirement already satisfied, skipping upgrade: numpy>=1.15.4 in c:₩users₩metal₩
32₩lib₩site-packages (from e_drone) (1.16.2)
Requirement already satisfied, skipping upgrade: colorama>=0.4.0 in c:₩users₩meta
7-32₩lib₩site-packages (from e_drone) (0.4.1)

C:₩Users₩metal>_
```

Enter 'pip install --upgrade e_drone' to upgrade.

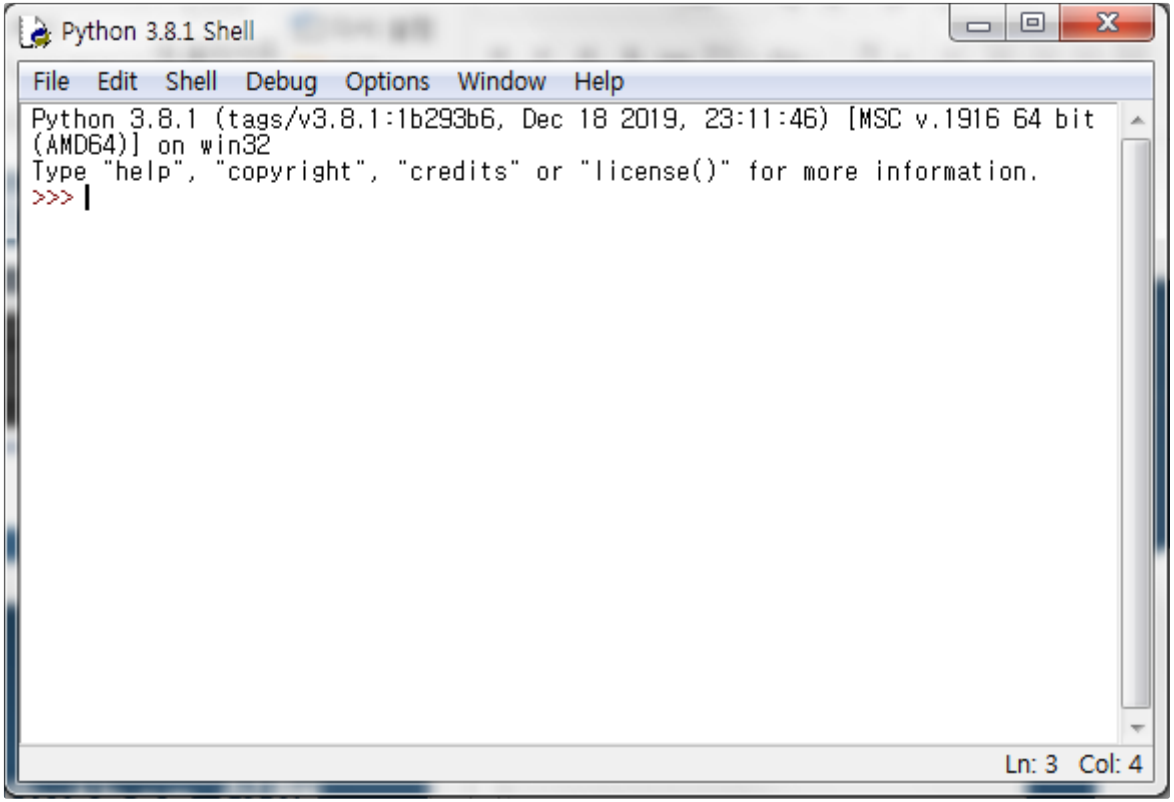
Python Execution

3. Python Execution



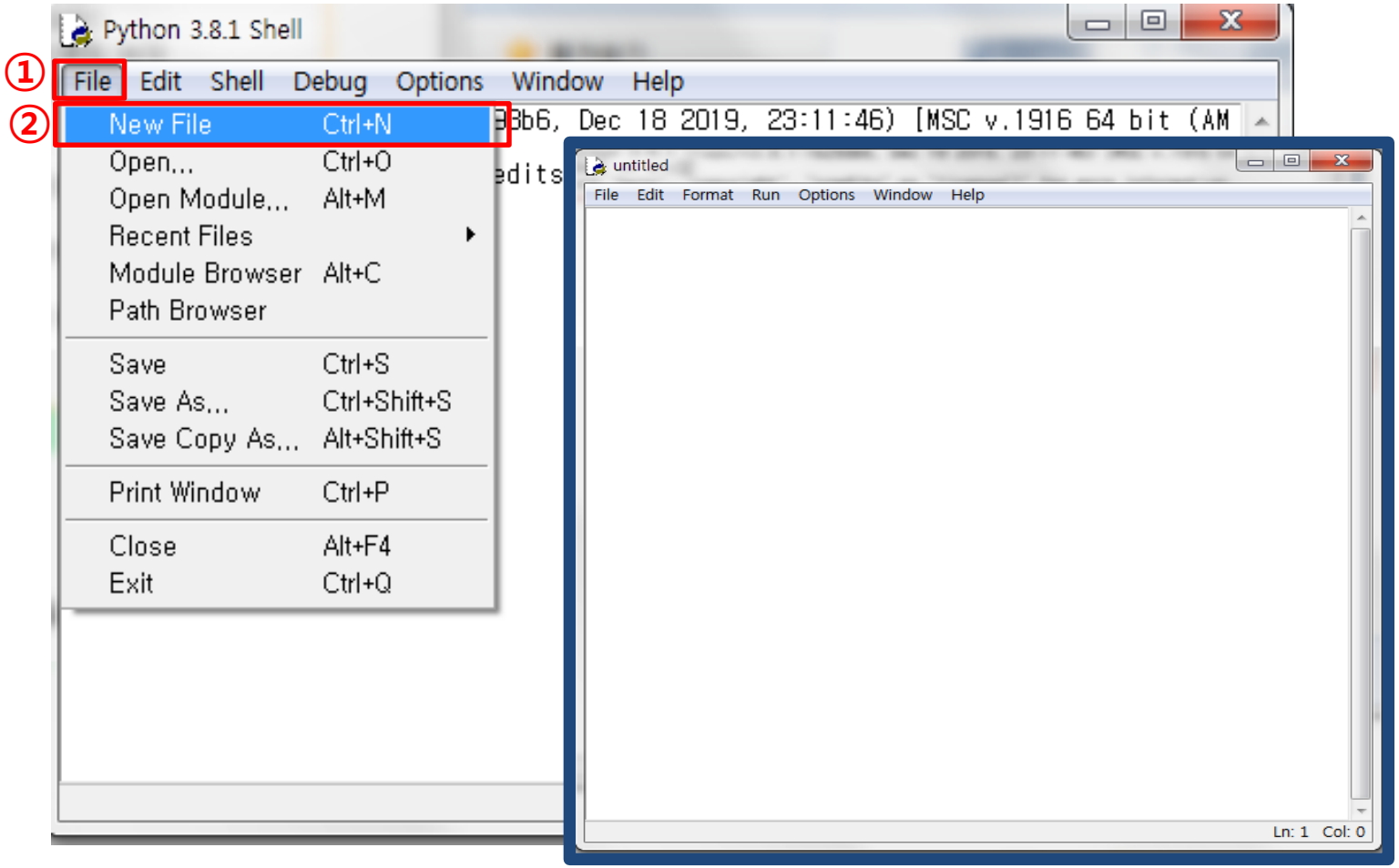
Window –> **IDLE(Python 3.8 64-bit) Click** to run

3. Python Execution



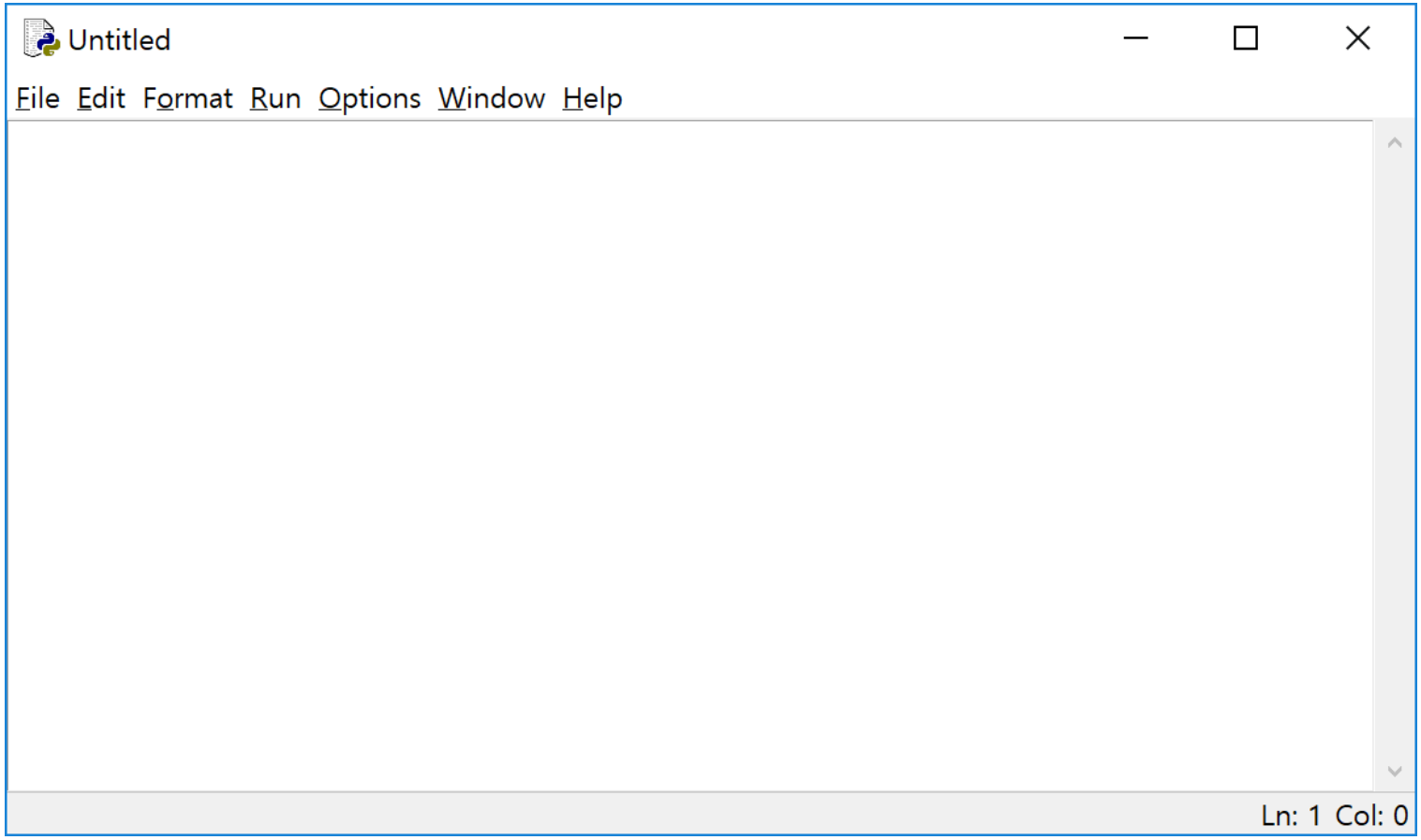
When Python runs, a **New Window** opens.

3. Python Execution



File -> New File Click

3. Python Execution



When a new window opens, create a **Python Program** here.

Programming 'CoDrone'

Example List

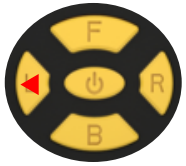
- Changing LEDs Red-Green-Blue
- Randomizing LEDs
- Take-off - Landing
- Take-off – Forward - Landing
- Square Shape Pattern Flight
- Circular Pattern Flight
- Read Gyro Sensor Value
- Read Altitude Sensor Value

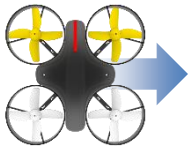
4. CoDrone Programming

- Precautions

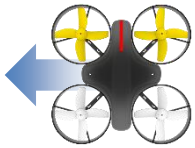
※ Before programming Python, fine-tune the drone to prevent it from flowing.

Fine Adjustment : ROLL





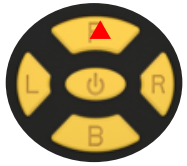
If the drone flows to the right, press the left button to adjust it.

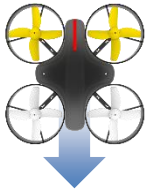


If the drone flows to the left, press the right button to adjust it.

Fine Adjustment : PITCH

If the drone flows to the front, press the back button to adjust it.



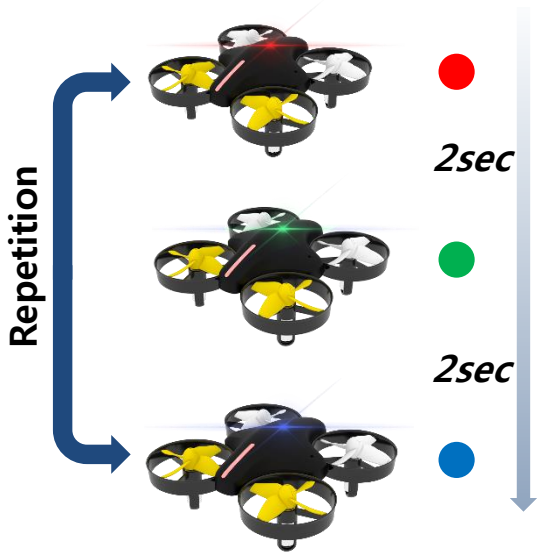


If the drone flows to the back, press the front button to adjust it.

4. CoDrone Programming

- LED R,G and B repeated flashing patterns for 2 seconds

Variable Name	Type or Range	Description
lightMode	UInt8	Operating LED Mode
interval	0~65535	Invoke Internal Brightness Control Function
R	0 ~ 255	Red
G	0 ~ 255	Green
B	0 ~ 255	Blue




```
def sendLightDefaultColor(self, lightMode, interval, r, g, b):
```

untitled

File Edit Format Run Options Window Help

```
import random
from time import sleep

from e_drone.drone import *
from e_drone.protocol import *


if __name__ == '__main__':

    drone = Drone(True, True, True, True, True)
    drone.open()

    while True:

        drone.sendLightDefaultColor(LightModeDrone.BodyDimming, 1, 255, 0, 0)
        sleep(2)
        drone.sendLightDefaultColor(LightModeDrone.BodyDimming, 1, 0, 255, 0)
        sleep(2)
        drone.sendLightDefaultColor(LightModeDrone.BodyDimming, 1, 0, 0, 255)
        sleep(2)

    drone.close()
```

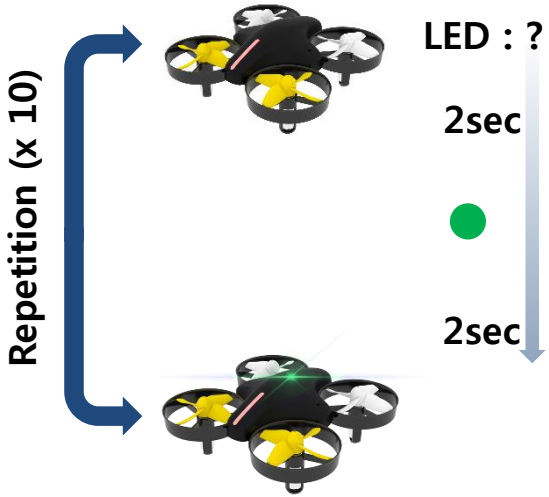
LED RGB (0~255)

Ln: 21 Col: 0

4. CoDrone Programming

- Runs 10 commands that gradually lighten and darken with LED random

Variable Name	Type or Range	Description
lightMode	UInt8	Operating LED Mode
interval	0~65535	Invoke Internal Brightness Control Function
R	0 ~ 255	Red
G	0 ~ 255	Green
B	0 ~ 255	Blue




```
def sendLightDefaultColor(self, lightMode, interval, r, g, b):
```

untitled

File Edit Format Run Options Window Help

```
import random
from time import sleep

from e_drone.drone import *
from e_drone.protocol import *


if __name__ == '__main__':

    drone = Drone(True, True, True, True, True)
    drone.open()

    for i in range(0, 10, 1):

        r    = int(random.randint(0, 255))
        g    = int(random.randint(0, 255))
        b    = int(random.randint(0, 255))

        dataArray = drone.sendLightDefaultColor(LightModeDrone.BodyDimming, 1, r, g, b)
        print("{0} / {1}".format(i, convertByteArrayToString(dataArray)))

        sleep(2)

    drone.close()
```

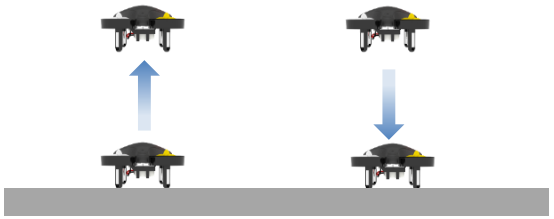
Ln: 21 Col: 0

4. CoDrone Programming

- Take-off, Hover, Land.

Variable Name	Type or Range	Description
Roll	-100 ~ 100	Roll
Pitch	-100 ~ 100	Pitch
Yaw	-100 ~ 100	Yaw
throttle	-100 ~ 100	Throttle

- sendTakeOff()
- sendControlWhile()
- sendLanding()



```
def sendControlWhile(self, roll, pitch, yaw, throttle, timeMs):
```

untitled

File Edit Format Run Options Window Help

```
from time import sleep

from e_drone.drone import *
from e_drone.protocol import *

if __name__ == '__main__':

    drone = Drone()
    drone.open()

    print("TakeOff")
    drone.sendTakeOff()
    sleep(0.01)

    print("Hovering")
    drone.sendControlWhile(0, 0, 0, 0, 5000)

    print("Go Stop")
    drone.sendControlWhile(0, 0, 0, 0, 1000)


    print("Landing")
    drone.sendLanding()
    sleep(0.01)
    drone.sendLanding()
    sleep(0.01)

    drone.close()
```

Ln: 21 Col: 0

Drone commands should give you about 0.01 second of delay

a routine that stops with a 5-sec hover flight of a drone

Wait a second

The drone landing function is called twice to stop

4. CoDrone Programming

- Take-off, Hover, Advance, Land.

- sendTakeOff()
- sendControlWhile()
- sendLanding()

untitled

File Edit Format Run Options Window Help

```
from time import sleep

from e_drone.drone import *
from e_drone.protocol import *

if __name__ == '__main__':

    drone = Drone()
    drone.open()

    print("TakeOff")
    drone.sendTakeOff()
    sleep(0.01)

    print("Hovering")
    drone.sendControlWhile(0, 0, 0, 0, 5000)

    print("Go Start")
    drone.sendControlWhile( 0, 50, 0, 0, 2000)

    print("Go Stop")
    drone.sendControlWhile(0, 0, 0, 0, 1000)


    print("Landing")
    drone.sendLanding()
    sleep(0.01)
    drone.sendLanding()
    sleep(0.01)

    drone.close()
```

Take-off Command

Hover Command

Forward (Pitch) 50% Speed Command for 2 seconds

Wait a second

Landing Command

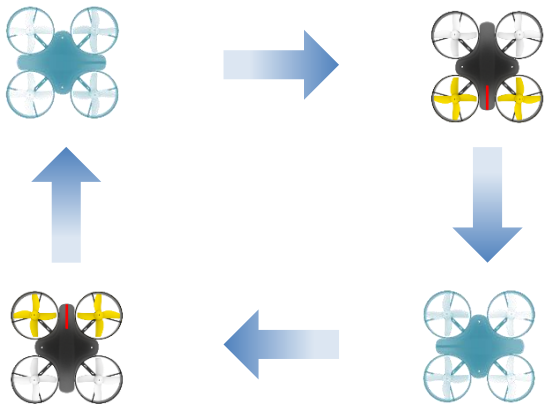
Ln: 21 Col: 0

4. CoDrone Programming

- Square Pattern Flight

- sendTakeOff()
- sendControlWhile()
- sendLanding()

Let's adjust the speed and time so we can fly it finely.



untitled

File Edit Format Run Options Window Help

```
from time import sleep

from e_drone.drone import *
from e_drone.protocol import *

if __name__ == '__main__':

    drone = Drone()
    drone.open()

    print("TakeOff")
    drone.sendTakeOff()
    sleep(0.01)

    print("Hovering")
    drone.sendControlWhile(0, 0, 0, 0, 4000)

    print("Go Start")
    drone.sendControlWhile( 0, 50, 0, 0, 1000)
    drone.sendControlWhile(0, 0, 0, 0, 1000)
    drone.sendControlWhile( 50, 0, 0, 0, 1000)
    drone.sendControlWhile(0, 0, 0, 0, 1000)
    drone.sendControlWhile( 0, -50, 0, 0, 1000)
    drone.sendControlWhile(0, 0, 0, 0, 1000)
    drone.sendControlWhile( -50, 0, 0, 0, 1000)
    drone.sendControlWhile(0, 0, 0, 0, 1000)
    print("Go Stop")

    print("Landing")
    drone.sendLanding()
    sleep(0.01)
    drone.sendLanding()
    sleep(0.01)

    drone.close()
```

Take-off Command

Hover Command

1 second forward (Pitch 50)

Right side for 1 second (Roll 50)

Reverse for 1 second (Pitch -50)

Left side for 1 second (Roll -50)

Landing Command

Ln: 21 Col: 0

4. CoDrone Programming

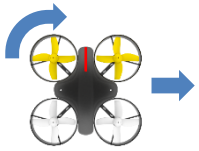
- Circular Pattern Flight

- sendTakeOff()
- sendControlWhile()
- sendLanding()

Let's change the speed to resize the circle and fly it elaborately.

Clockwise Rotation

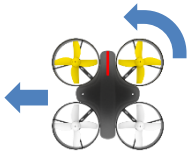
```
drone.sendControlWhile(50, 0, -50, 0, 4000)
```



Roll (50) **Yaw** (-50)

CounterClockwise Rotation

```
drone.sendControlWhile( -50, 0, 50, 0, 4000)
```



Roll (-50) **Yaw** (50)

untitled

File Edit Format Run Options Window Help

```
from time import sleep

from e_drone.drone import *
from e_drone.protocol import *

if __name__ == '__main__':

    drone = Drone()
    drone.open()

    print("TakeOff")
    drone.sendTakeOff()
    sleep(0.01)

    print("Hovering")
    drone.sendControlWhile(0, 0, 0, 0, 4000)

    print("Go Start")
    drone.sendControlWhile( 50, 0, -50, 0, 4000)

    print("Go Stop")
    drone.sendControlWhile(0, 0, 0, 0, 1000)

    print("Landing")
    drone.sendLanding()
    sleep(0.01)
    drone.sendLanding()
    sleep(0.01)

    drone.close()
```

Take-off Command

Hover Command

4-sec Revolving Flight

Landing Command

Ln: 21 Col: 0

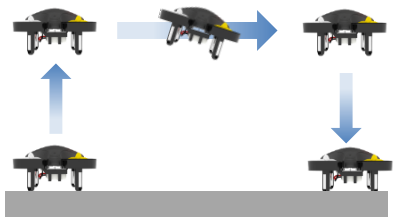
4. CoDrone Programming

- Take-off, Hover, Forward, Landing Test.

```
def sendControl(self, roll, pitch, yaw, throttle):
```

- sendControl()

Variable Name	Type or Range	Description
Roll	-100 ~ 100	Roll
Pitch	-100 ~ 100	Pitch
Yaw	-100 ~ 100	Yaw
throttle	-100 ~ 100	Throttle



```
from time import sleep

from e_drone.drone import *
from e_drone.protocol import *

if __name__ == '__main__':

    drone = Drone()
    drone.open()

    print("TakeOff")
    drone.sendTakeOff()
    sleep(0.01)
    for i in range(3, 0, -1):
        print("{0}".format(i))
        sleep(1)

    print("Hovering")
    for i in range(1, 0, -1):
        print("{0}".format(i))
        drone.sendControlWhile(0, 0, 0, 0, 1000)
        sleep(1)
```

Take-off Command

Flying Time Screen Output

Hover Command

```
    print("Go Start")
    drone.sendControl(0, 50, 0, 0)
    for i in range(3, 0, -1):
        print("{0}".format(i))
        sleep(1)

    print("Go Stop")
    drone.sendControl( 0, 0, 0, 0)
    for i in range(2, 0, -1):
        print("{0}".format(i))
        sleep(1)

    print("Landing")
    drone.sendLanding()
    sleep(0.01)

drone.close()
```

Advance for 1 second

Landing Command

4. CoDrone Programming

- Read Gyro, Acceleration Sensor Values

- Motion
- sendRequest()

Variable Name	Type or Range	Description
deviceType	DeviceType	Destination device to transfer
dataType	DataType	Type of Data

```
def sendRequest(self, deviceType, dataType):
```

untitled

File Edit Format Run Options Window Help

```
from time import sleep

from e_drone.drone import *
from e_drone.protocol import *


def eventMotion(motion):
 print("eventMotion()")
 print("- Accel: {0:5}, {1:5}, {2:5}".format(motion.accelX, motion.accelY, motion.accelZ))
 print("-  Gyro: {0:5}, {1:5}, {2:5}".format(motion.gyroRoll, motion.gyroPitch, motion.gyroYaw))
 print("- Angle: {0:5}, {1:5}, {2:5}".format(motion.angleRoll, motion.anglePitch, motion.angleYaw))


if __name__ == '__main__':

    drone = Drone()
    drone.open()

    # Register event handling function
      drone.setEventHandler(DataType.Motion, eventMotion)

    while True:

        # Range Information Request
        drone.sendRequest(DeviceType.Drone, DataType.Motion)
        sleep(1)

    drone.close()
```

Ln: 21 Col: 0

4. CoDrone Programming

- Read Altitude Sensor Value

- Attitude
- sendRequest()

Variable Name	Type or Range	Description
deviceType	DeviceType	Destination device to transfer
dataType	DataType	Type of Data

```
def sendRequest(self, deviceType, dataType):
```

untitled

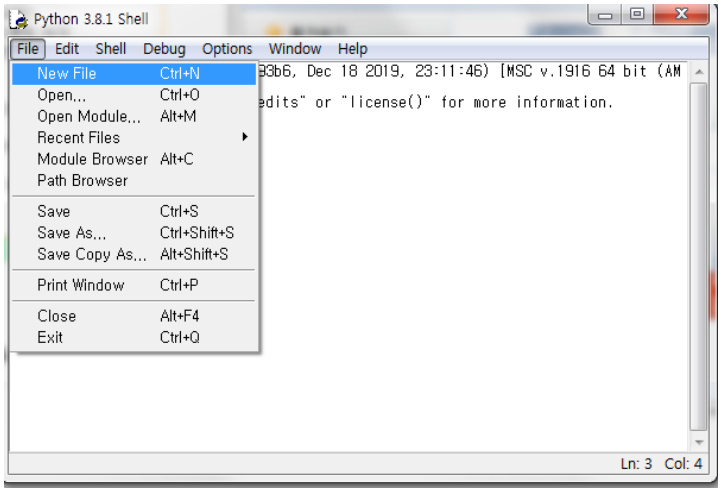
File Edit Format Run Options Window Help

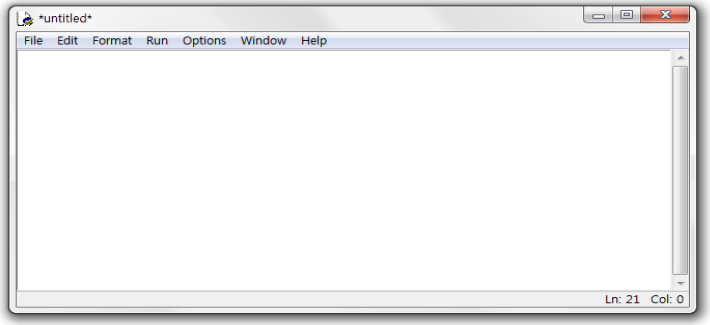
```
from time import sleep

from e_drone.drone import *
from e_drone.protocol import *


def eventAltitude(altitude):
    print("eventAltitude()")
    print("-  Temperature: {0:.3f}".format(altitude.temperature))
    print("-      Pressure: {0:.3f}".format(altitude.pressure))
    print("-      Altitude: {0:.3f}".format(altitude.altitude))
    print("- Range Height: {0:.3f}".format(altitude.rangeHeight))

if __name__ == '__main__':

    drone = Drone()
    drone.open()

    # Register event handling function
      drone.setEventHandler(DataType.Altitude, eventAltitude)

    while True:

        # Altitude Information Request
        drone.sendRequest(DeviceType.Drone, DataType.Altitude)
        sleep(1)

    drone.close()
```

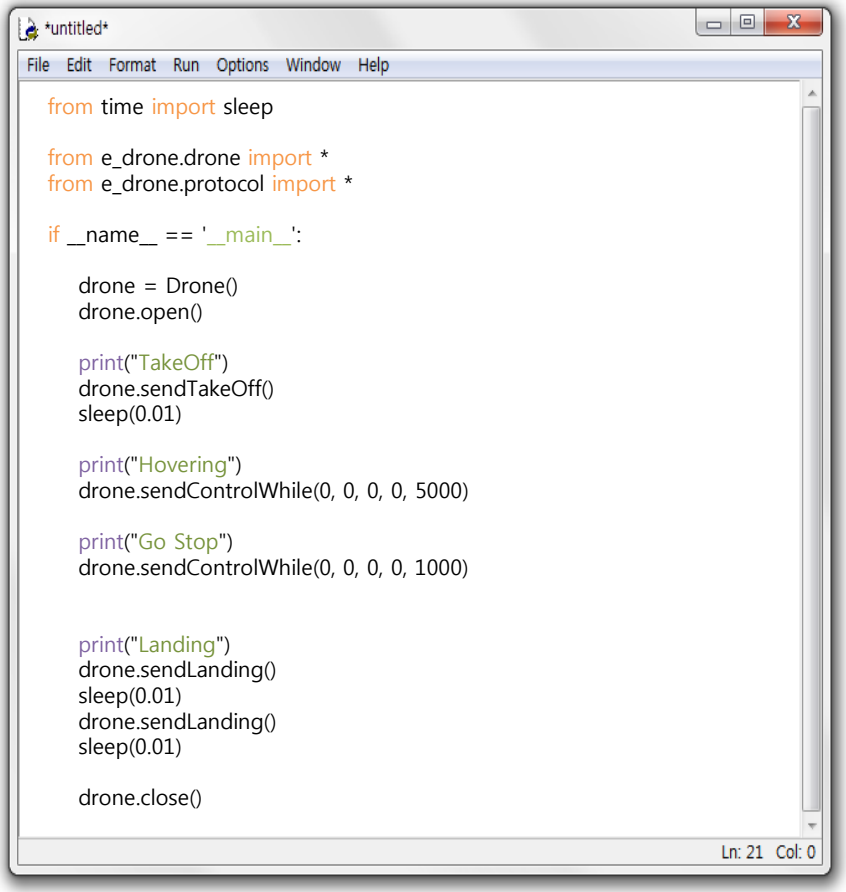
Ln: 21 Col: 0

Python Debugging

5. CoDrone Programming

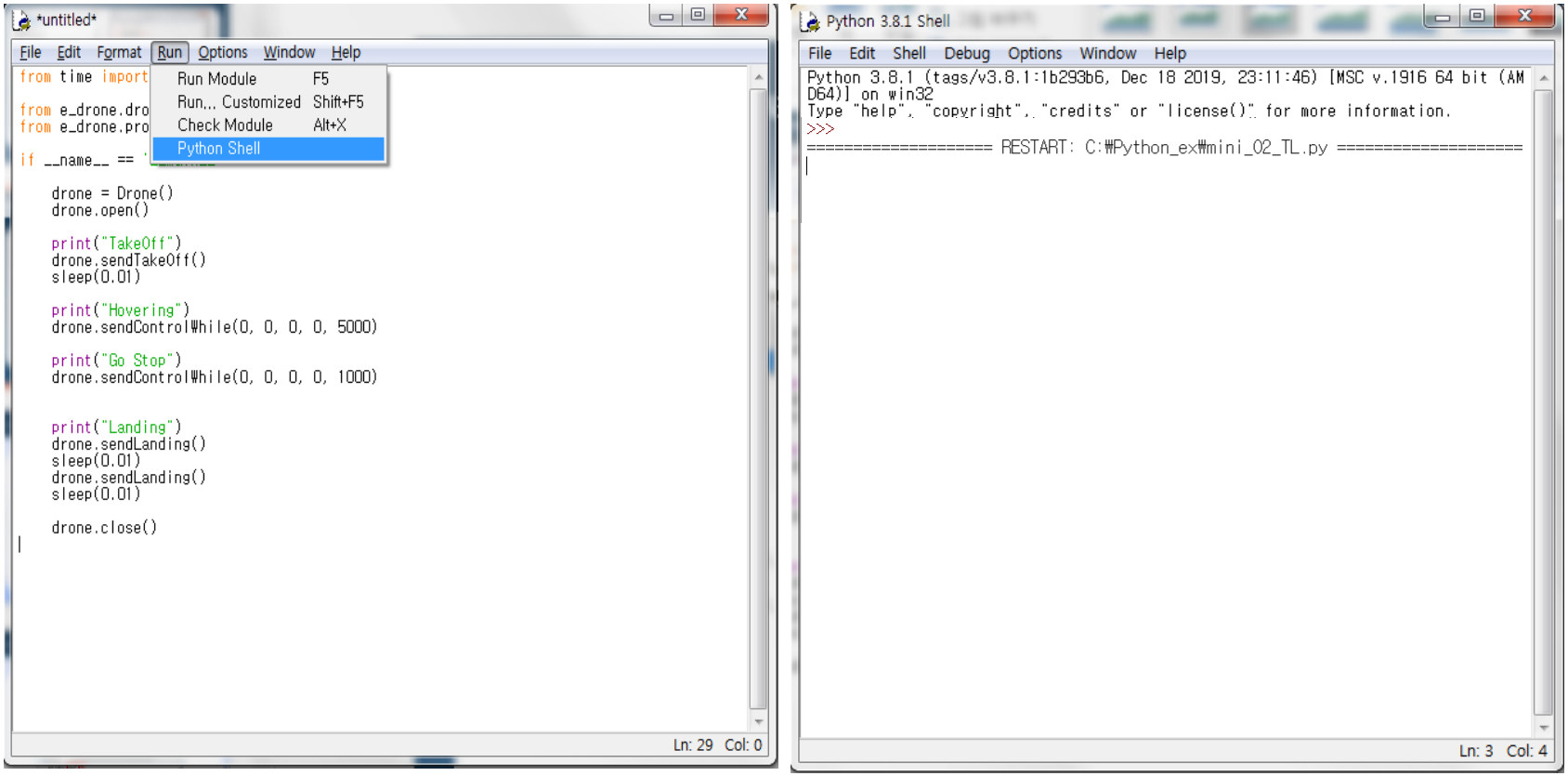






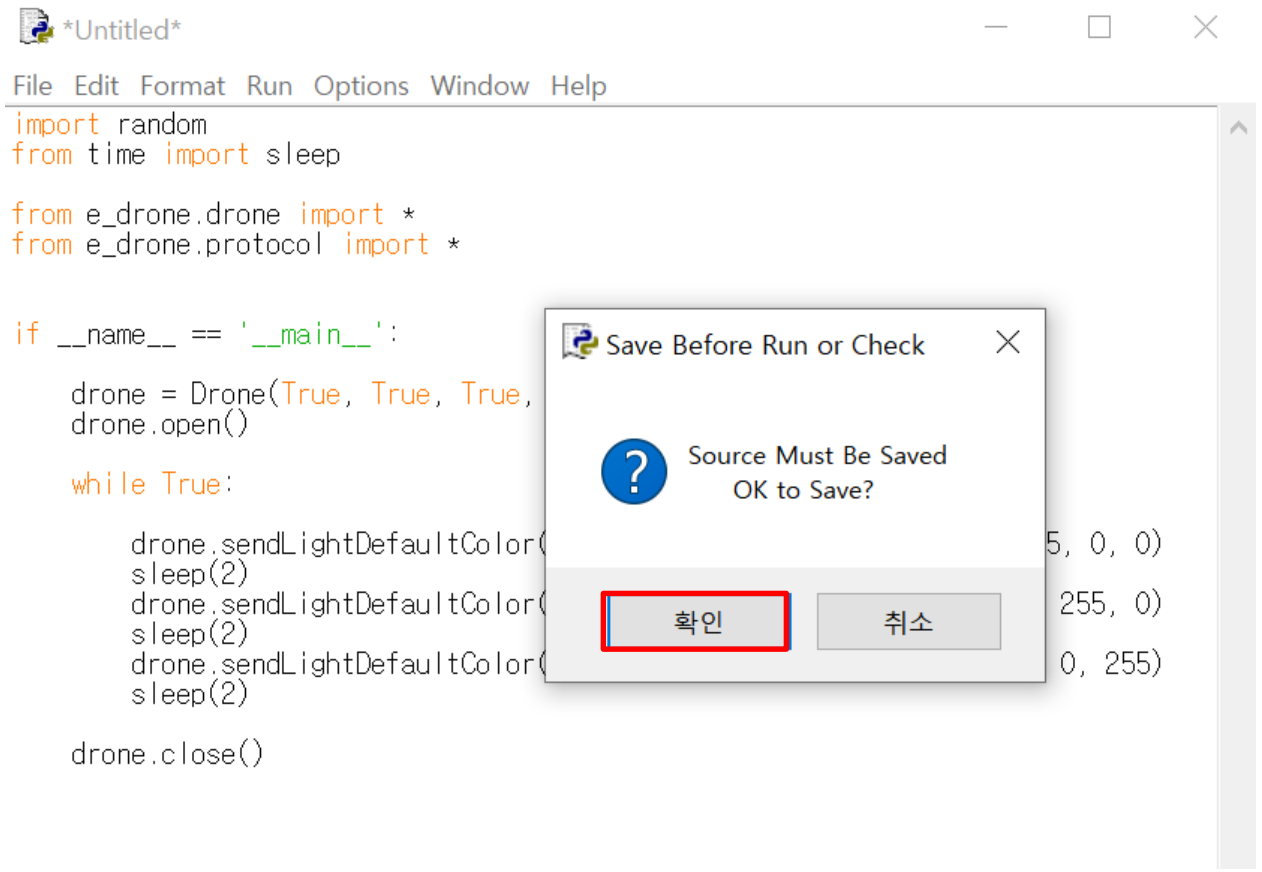
In the **Python IDLE** program, open **New File** and type or copy the program.

5. CoDrone Programming



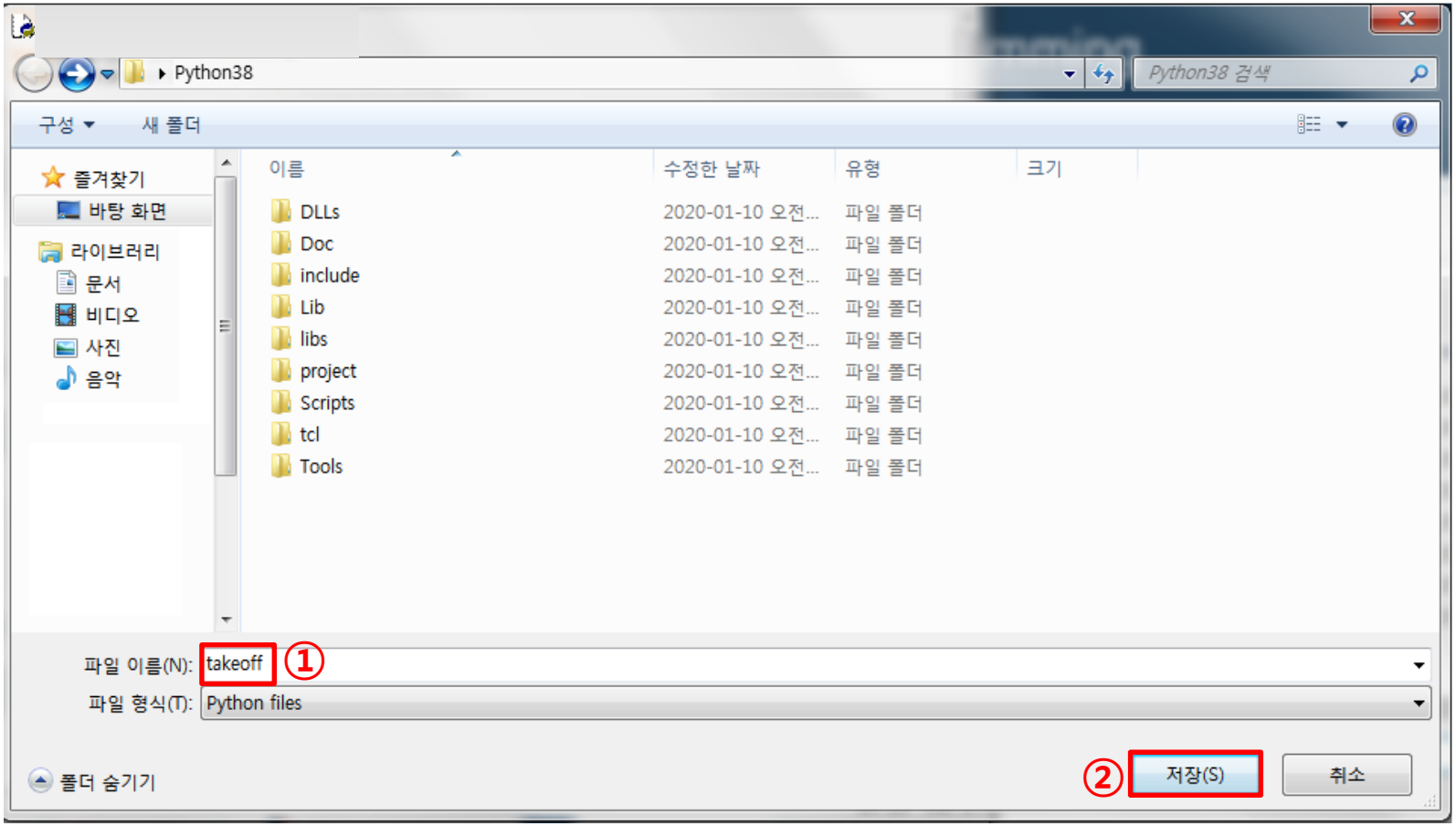
Run – Run Module Click or Press **F5** to run the program.

5. CoDrone Programming



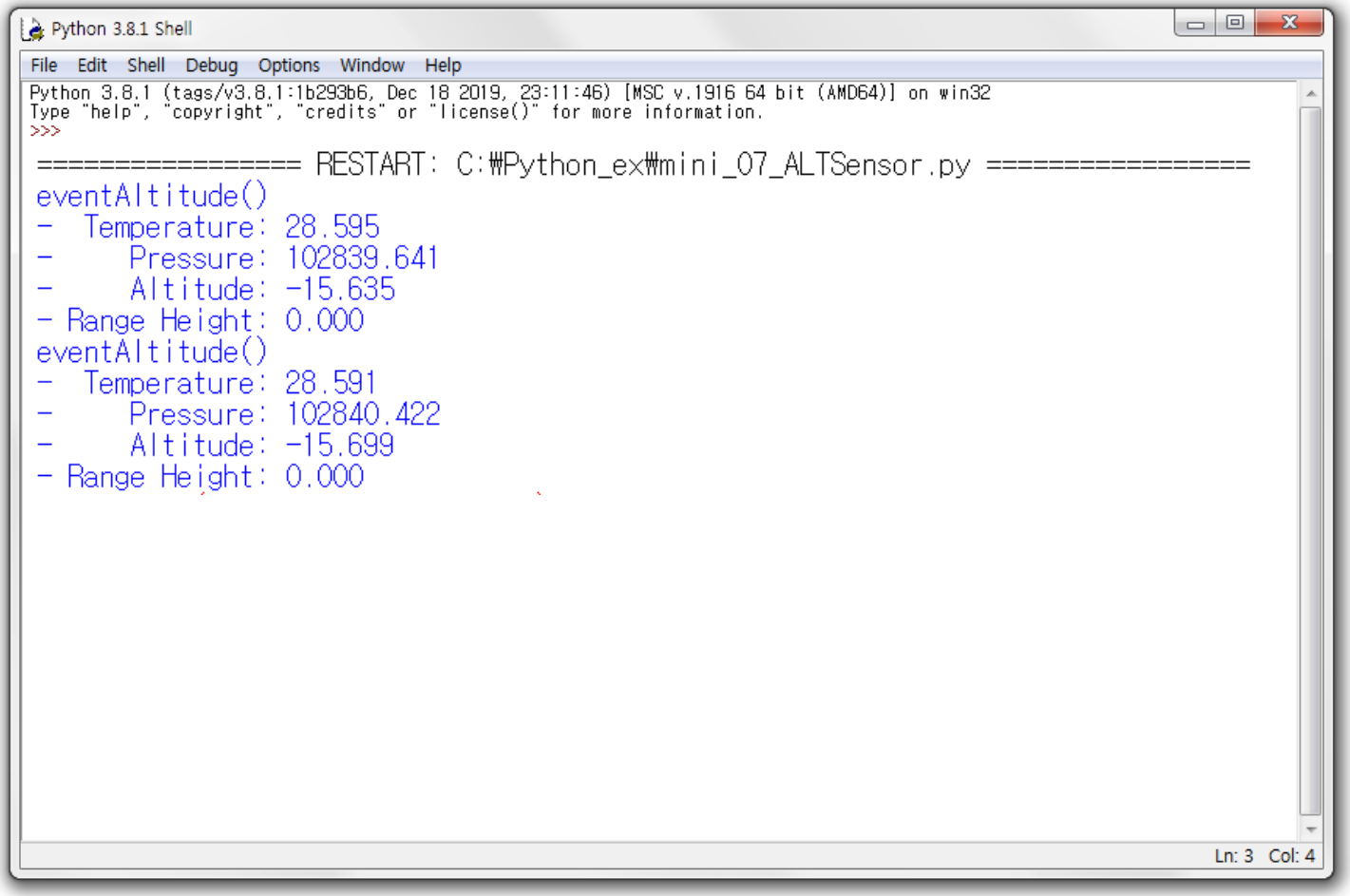
Click OK because the program must be saved to run.

5. CoDrone Programming



Enter the File Name and click Save.

5. CoDrone Programming



A Shell window opens and the program runs after the screen output.

5. CoDrone Programming

Python Library can see functions and protocols through the links below.

http://dev.byrobot.co.kr/documents/kr/products/e_drone/library/python/e_drone/

e_drone for python

1. Intro
2. System
3. Protocol
4. Drone
5. Examples - Ping
6. Examples - Information
7. Examples - Pairing
8. Examples - Control
9. Examples - Sensor
10. Examples - Motor
11. Examples - Setup
12. Examples - Buzzer
13. Examples - Vibrator
14. Examples - Light
15. Examples - Display
16. Examples - Input
17. Examples - Error

Examples - Ping

Examples - Information

Examples - Pairing

Examples - Control

Examples - Sensor

Examples - Motor

Examples - Setup

Examples - Buzzer

Examples - Vibrator

Examples - Light

Examples - Display

Examples - Input