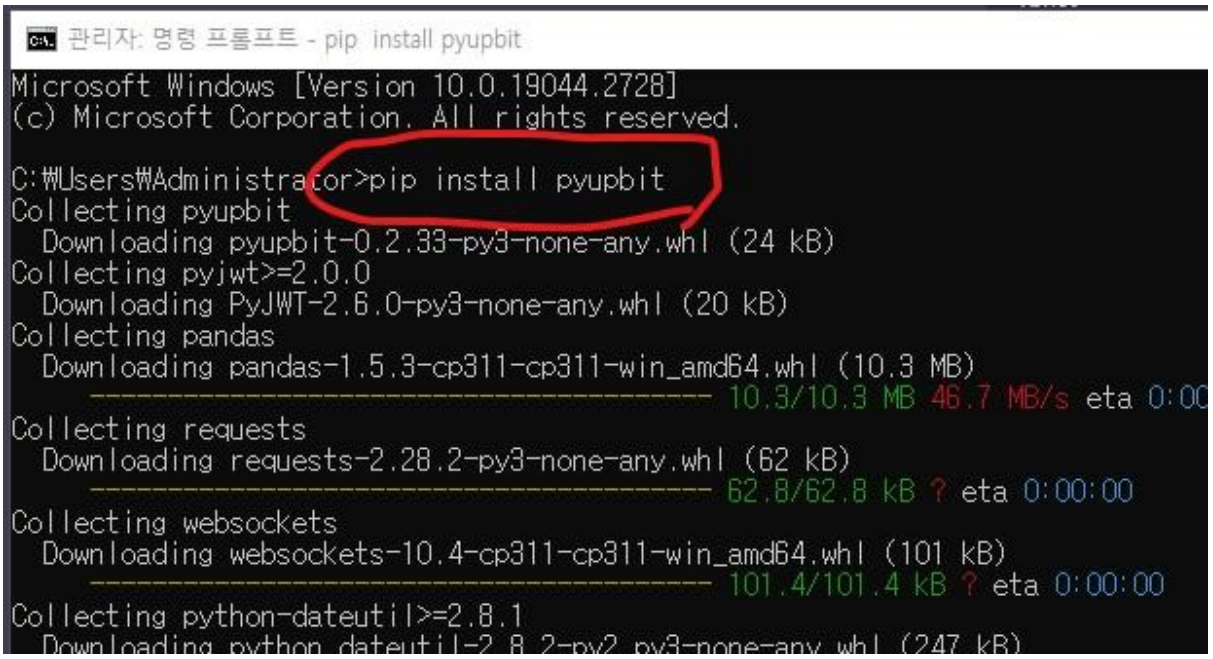
Upbit 자동매매를 위한 **Python** 설정을 하기 위해서는 다음과 같은 단계를 따르면 됩니다.

Written by 말과의미

1) Python 설치**: Python 3.x (64**비트**)** 버전을 다운로드하고 설치합니다**.** 공식 홈페이지에서 다운로드 받을 수 있습니다**.** **https://www.python.org/downloads/**

2) 필요한 라이브러리 설치**: Upbit** 자동매매를 위해 필요한 라이브러리인 **pyupbit**과 **ta-lib**을 설치합니다**. CMD**에서 다음과 같이 명령어를 입력합니다**.**

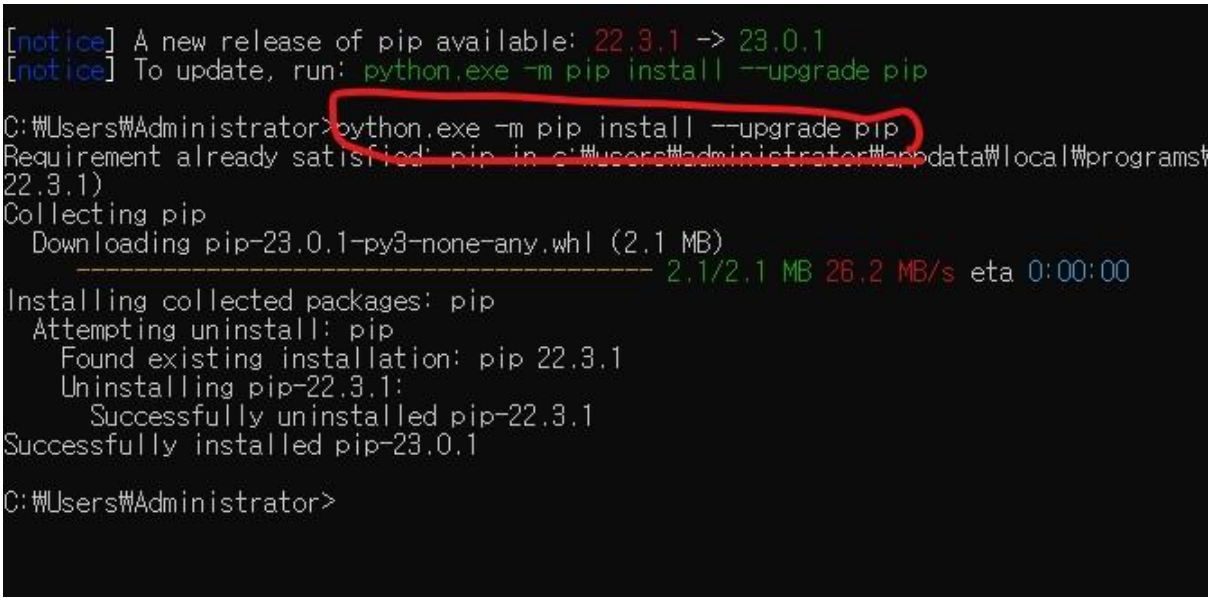
pip install pyupbit
pip install ta-lib



a) pip install에서 필요한 경우 다음과 같이 **Upgrade**도 해줍니다**.**

[notice] A new release of pip available: 22.3.1 -> 23.0.1
[notice] To update, run: python.exe -m pip install --upgrade pip

C:\Users\Administrator>python.exe -m pip install --upgrade pip



3) 개발 환경 설정**: Python** 개발을 위해 통합 개발 환경**(IDE)**을 설치합니다. **PyCharm, VS Code** 등을 사용할 수 있습니다.

a) Visual Studio Code는 **Microsoft**에서 개발한 무료 오픈 소스 코드 에디터입니다. **Python** 개발을 위해 많은 개발자들이 사용하는 인기 있는 **IDE** 중 하나입니다.

Visual Studio Code를 다운로드하려면 다음 단계를 따르세요.
https://code.visualstudio.com/ 접속합니다.

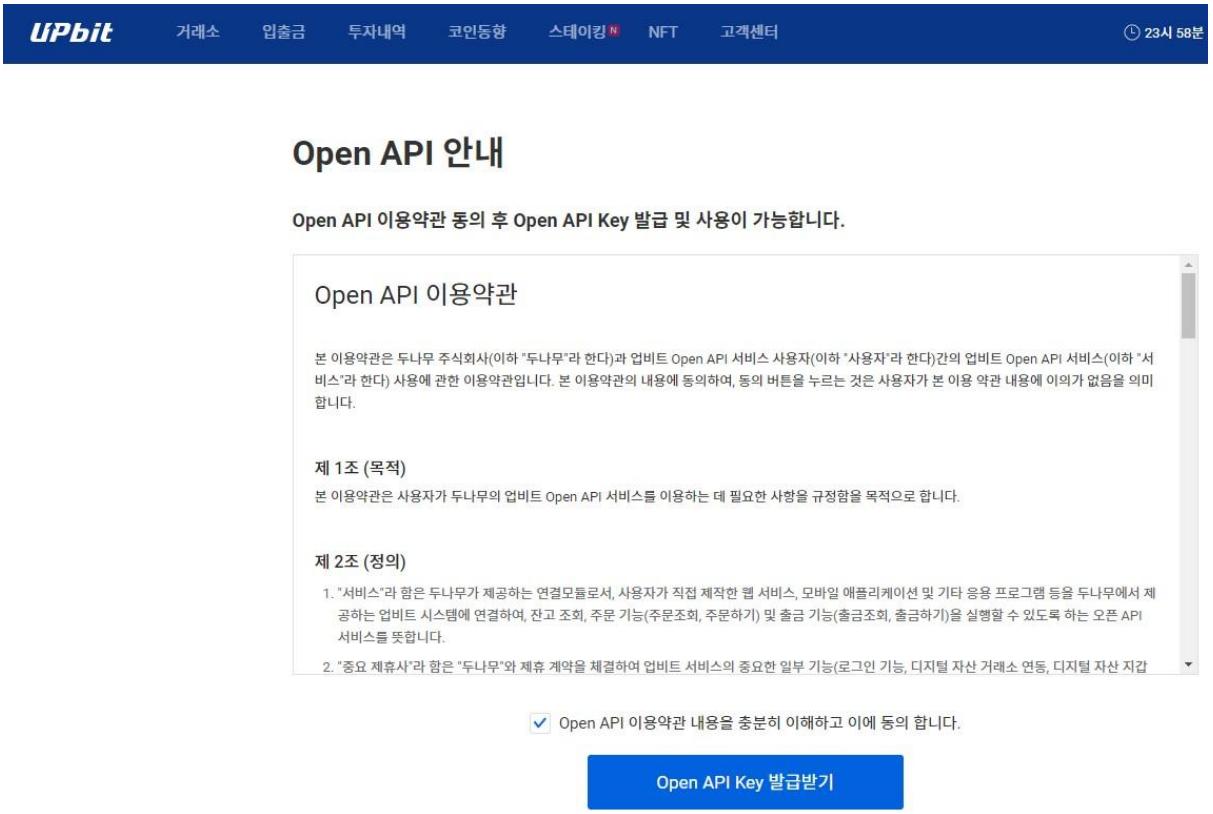
설치가 완료되면 **Visual Studio Code**를 실행합니다.

이제 **Python** 확장 프로그램을 설치하면 **Python** 개발을 시작할 수 있습니다. **Visual Studio Code**에서는 다른 언어와 통합하여 사용할 수 있습니다. 따라서, 다른 언어로 개발할 때도 유용하게 사용할 수 있습니다.

b) VS code 실행후 언어팩을 한글로 설치한다.

4) Upbit API 설정**: Upbit** 자동매매를 위해 **Upbit API**를 사용해야 합니다. 따라서, **Upbit API** 인증키를 발급받고 **API** 호출을 위한 인증 정보를 **Python** 코드에서 사용할 수 있도록 설정합니다.

a) Upbit API 인증키는 **Upbit** 고객센터 **-> Open API-> Open AIP** 신청에서 발급받습니다.



b) API 신청하려면 **IP** 주소를 넣어야 합니다**. IP** 주소는 **CMD**에서

C:\Users\Administrator>ipconfig

IP address 확인 가능

c) API 인증키는 일반적으로 **access_key**와 **secret_key**로 구성되어 있습니다**.** 이후에는 발급받은 **API** 인증키를 **Python** 코드에서 사용하여 **API** 호출을 할 수 있습니다**.**

5) 자동매매 코드 작성**: Upbit API**와 필요한 라이브러리를 이용하여 자동매매 코드를 작성합니다**.** 예를 들어**,** 다음과 같이 **Python** 코드에서 **Upbit API**를 호출할 수 있습니다**.**

```
import pyupbit

access_key = "access key를 입력하세요"
secret_key = "secret key를 입력하세요"
upbit = pyupbit.Upbit(access_key, secret_key)

# 예시: 현재 BTC의 가격을 조회합니다.
ticker = "KRW-BTC"
price = pyupbit.get_current_price(ticker)
```

```
print(ticker, price)
```

6) 자동매매 실행**:** 작성한 코드를 실행하여 자동매매를 시작합니다**.**

7) Upbit의 **KRW-BTC** 시세를 **10**분마다 텔레그램에 보내주는 코딩 참고

```
import time
import pyupbit
import requests

# Telegram Bot API 설정
bot_token = "Bot Token을 입력하세요"
chat_id = "Chat ID를 입력하세요"

# Upbit API 인증키 설정
access_key = "Access Key를 입력하세요"
secret_key = "Secret Key를 입력하세요"
upbit = pyupbit.Upbit(access_key, secret_key)

# KRW-BTC 티커
ticker = "KRW-BTC"

# 10분마다 실행되는 함수
def send_message():
    # 현재 시세 조회
    current_price = pyupbit.get_current_price(ticker)
    message = f"현재 {ticker} 시세: {current_price} KRW"

    # Telegram 메시지 전송

requests.get(f"https://api.telegram.org/bot{bot_token}/sendMessage?chat_id={
chat_id}&text={message}")

# 프로그램 시작
while True:
    send_message()
    time.sleep(600)  # 10분 대기
```

8) Telegram Bot API를 사용하여 텔레그램 메시지를 전송하려면**,** 먼저 봇을 생성하고 봇의 **Token** 값을 발급받아야 합니다**.** 이 **Token** 값은 **BotFather**라는 텔레그램 봇을 통해 발급받을 수 있습니다**.**

Telegram for Korean https://desktop.telegram.org/?setln=ko

a) **BotFather** 봇을 검색하고, **"Start"** 버튼을 눌러 봇을 시작합니다. **"/newbot"** 명령어를 입력하여 새로운 봇을 생성합니다. 봇의 이름과 사용자 이름을 입력하고, 봇의 **Token** 값을 발급받습니다.

b) 발급받은 **Token** 값을 이용하여 봇 **API**에 접근합니다. 발급받은 **Token** 값을 이용하여 **Python** 코드에서 텔레그램 봇을 호출할 수 있습니다. 이때, 메시지를 전송할 대상인 **Chat ID**가 필요합니다. **Chat ID**는 다음과 같이 얻을 수 있습니다.

c) 봇 **API**에 메시지를 전송하고자 하는 대상과 봇을 추가합니다.

@getmyid_bot

userI와 **chat_id** 값을 확인합니다.

따라서, 위 코드에서는 봇 **Token** 값과 **Chat ID** 값을 미리 입력하여, 프로그램 실행 시 바로 메시지를 전송할 수 있도록 설정하였습니다.

9) coding for getting a Bot's chat id and Channel' Chat ID

아래 **python** 코드는 **botFather**에서 만든 **Bot**의 **Chat ID**를 구하는 **python** 코드와 해당 **bot**이 관리자로 추가된 채널의 **Chat ID**를 구하는 **python** 코딩이다.

채널의 **ChatID**는 해당 **bot**을 해당 채널의 관리자로 추가한 후 '안녕하세요' 등의 메세지를 남긴 후에 **python code**를 실행 해야 한다. 아래 **token**와 **id**는 임의의 숫자와 문자다.

a) bot의 Chat ID를 구하는 코드

```
import requests

# 봇 API Token을 입력하세요.
bot_token = "6277aUlnZEvlwdeHEY"

# Chat ID를 확인할 대상과 봇이 추가된 채팅창에서 아무 메시지나 입력합니다.
# 대상과 봇이 추가된 채팅창을 열고, 아무 메시지나 입력하세요.

# Telegram API를 이용하여 대화 내역을 가져옵니다.
response =
requests.get(f"https://api.telegram.org/bot{bot_token}/getUpdates").jso
n()

# 가져온 대화 내역에서 "chat" 항목의 "id" 값을 확인하여 Chat ID 값을 얻습니다.
```

```
chat_id = response["result"][0]["message"]["chat"]["id"]

print(f"Chat ID: {chat_id}")
```

b) 아래는 채널 **Chat ID** 구하는 **python code**

```
import requests

__TOKEN = "627734eHEY"    # bot의 액세스토큰
url = f"https://api.telegram.org/bot{__TOKEN}/getUpdates?"
res = requests.get(url).json()

for item in res.get("result", []):
    if "channel_post" in item:
        chat = item["channel_post"]["sender_chat"]
        print(f"channel: {chat['title']}({chat['id']})")

        # 채널에 경우 이 값은 -100으로 시작하는 숫자값입니다.
```

c) 최종적으로 확인된 **Chat ID**에 메세지 보내서 확인하기

```
import requests

# 봇 API Token과 메시지 전송할 Chat ID 값을 입력합니다.
bot_token = "627734EvlwdeHEY"
# chat_id = "413" 개인 bot chat id
chat_id = "-1001"  # channel chat ID

# Chat ID를 확인하기 위한 메시지를 전송합니다.
message = "Chat ID를 확인했습니다. Hello!"
requests.get(f"https://api.telegram.org/bot{bot_token}/sendMessage?chat
_id={chat_id}&text={message}")
```