위,경도 -> 주소 변환

```
import threading
import requests
import time
import json
import datetime as dt
```

```
# API KEY
APIKEY: str = "6DBD19F7-1B18-36CE-AD74-FD0E5FAEC01F"

# CSV파일의 경로
INPUT_PATH: str = "input.csv"

# Thread 가동 수
THREAD_COUNT: int = 50

# OpenAPI 연동을 위한 URL 형식
URL_TPL = "https://api.vworld.kr/req/address?
service=address&request=getAddress&version=2.0&crs=epsg:4326&point={lat},
{lng}&format=json&type=both&zipcode=true&simple=false&key={apikey}"
```

```
class GeocodeRequest(threading.Thread):

    # file -> csv 파일 입출력을 위한 파일 핸들
    # data -> 수집해야할 위,경도 리스트
    def __init__(self, file, data):
        threading.Thread.__init__(self)
        self.file = file
        self.data = data

    def run(self):
        global APIKEY
        global URL_TPL

        # 접속 객체 생성
        with requests.Session() as session:
            # 웹 페이지 직접 수집이 아닌 이상 이 구문은 생략 가능
            session.headers.update({
                "Referer": "",
                "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/111.0.0.0 Safari/537.36"
            })

            # data의 형식 --> CSV파일의 일부를 통째로 전달
            for i, c in enumerate(self.data):
                # c ⟶ "124900078","25595","선사초등학
교","127.129285","37.555166","마을버스"
```

```
                # 하나의 행을 쉼표 단위로 나눔
                line = c.strip().split(",")

                # 위경도 추출
                lat = float(line[3].replace('"', '').strip())
                lng = float(line[4].replace('"', '').strip())

                # 수집해야할 URL 구성
                url = URL_TPL.format(lat=lat, lng=lng, apikey=APIKEY)

                # 데이터 수집
                try:
                    r = session.get(url)
                except:
                    print("요청에 실패했습니다.")

                    # 현재의 행에 강제로 빈 값 두개를 추가하고 텍스트 파일에 기록
                    line.append('""')
                    line.append('""')
                    self.file.write(",".join(line))
                    continue

                # 연동 결과 처리
                r.encoding = "UTF-8"
                text = json.loads(r.text)
                result = text['response']['result'][0]
                postcode = result['zipcode'].strip()
                address = result['text'].strip()

                # 원래 데이터의 한 행에 수집 결과를 추가
                line.append('"%s"' % postcode)
                line.append('"%s"' % address)
                print(" >>> [%s] %s" % (postcode, address))

                # 추가된 행을 새로운 파일에 한 줄로 기록하기 위해 콤마를 기준으로 하는 하나의
문자열로 연결
                oneline = ",".join(line)
                self.file.write(oneline)
                self.file.write("\n")
```

```
with open(INPUT_PATH, "r", encoding="euc-kr") as f:
    csv = f.readlines()

    # 제목행 제거
    del(csv[0])

csv[:5]
```

```
count = len(csv)
split_count = count//THREAD_COUNT
print("총 데이터 수:", count, "쓰레드 1개당 처리 할 데이터 수:", split_count)
```

```
split_csv = []
total = 0
```

```
for i in range(0, THREAD_COUNT):
    if i + 1 < THREAD_COUNT:
        part = csv[split_count*i:split_count*(i+1)]
    else:
        part = csv[split_count*i:]

    print(len(part))
    total += len(part)
    split_csv.append(part)

total
```

```
split_csv[0]
```

```
# 수집 결과를 저장할 파일 생성
fname = dt.datetime.now().strftime("%Y%m%d_%H%M%S.csv")

with open(fname, "w", encoding="euc-kr") as f:

    # csv 파일의 첫 줄은 강제로 저장
    f.write("노드 ID,정류소번호,정류소명,X좌표,Y좌표,정류소 타입,우편번호,주소\n")

    # 생성할 thread 목록
    threads = []

    # 데이터 묶음 단위로 쓰레드 생성
    for s in split_csv:
        geocoder = GeocodeRequest(f, s)
        geocoder.start()
        threads.append(geocoder)

    # 모든 쓰레드의 작업이 종료될 때까지 대기
    for t in threads:
        t.join()

    print("작업이 종료되었습니다.")
```