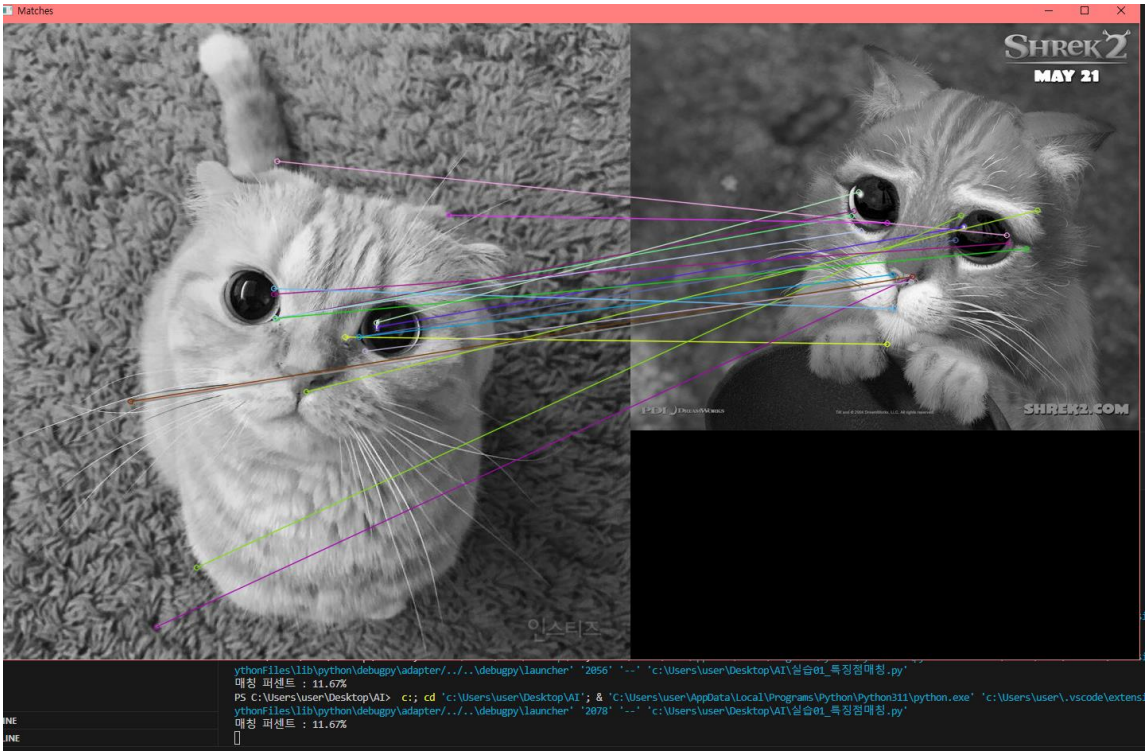
이미지 처리과제 #04

**전진희**

**실습01. 특징점 매칭**

```
import cv2

# 이미지 불러오기
img1 = cv2.imread("./data/compare_cat01.jpg", cv2.IMREAD_GRAYSCALE)
img2 = cv2.imread("./data/compare_cat02.jpg", cv2.IMREAD_GRAYSCALE)

# 특징점 검출기 생성
ord = cv2.ORB_create()

# 특징점 검출과 디스크립터 계산
keypoint01, descriptor01 = ord.detectAndCompute(img1, None)   # orb 검출기
사용하여 이미지에서 특징점 검출하고 거기서 디스크립터 계산
keypoint02, descriptor02 = ord.detectAndCompute(img2, None)

# print(keypoint01, keypoint02, descriptor01, descriptor02) # 특징점과
디스크립터 잘 나오나 확인
# exit()

# 매칭기 생성 알고리즘
bf = cv2.BFMatcher(cv2.NORM_HAMMING, crossCheck=True) # norm_hamming 즉 해밍
거리(같은 길이를 가진 이진 문자열 간의 차이를 측정하는 거리)를 이용하여 디스크립터
거리 비교하여 매칭

# 특징점 매칭
matches = bf.match(descriptor01, descriptor02)

# 매칭 결과 정렬
matches = sorted(matches, key=lambda x:x.distance)

# 매칭 결과 그리기
result  = cv2.drawMatches(img1, keypoint01, img2, keypoint02, matches[:10],
None, flags=cv2.DrawMatchesFlags_NOT_DRAW_SINGLE_POINTS)

# 매칭 결과 출력
cv2.imshow("Matches", result)

# 매칭 퍼센트 계산
num_matches = len(matches)
num_good_matches = sum(1 for m in matches if m.distance < 50) # 거리 임계값
matching_percent = (num_good_matches / num_matches) * 100
```

```python
# 매칭 퍼센트 출력
print("매칭 퍼센트 : %.2f%%" % matching_percent)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

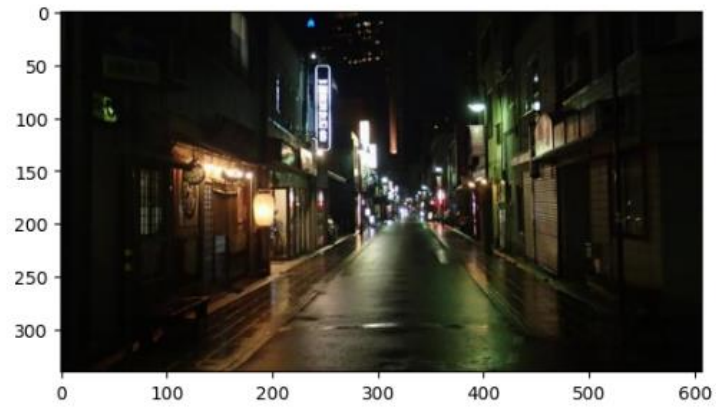



매칭 퍼센트 : 11.67%

## 실습02.이미지 밝기 조정

# 조명 조건에 따라 이미지 보정 간단한 실습

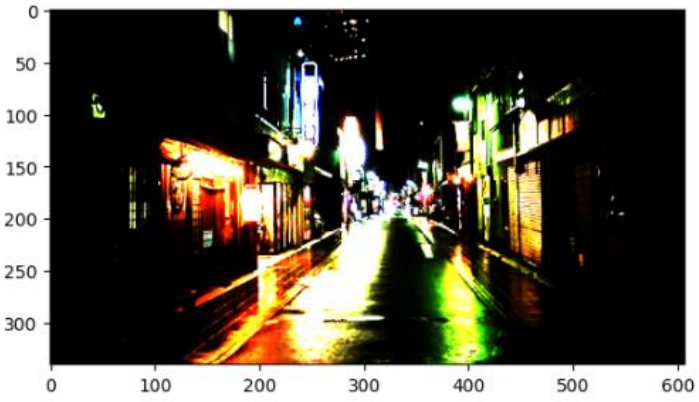
In [6]:
```
import cv2
import matplotlib.pyplot as plt

# < 이미지 읽기>
image = cv2.imread("./data/night_street.png")

# BGR -> RBG 변경
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

plt.imshow(image)
plt.show()
```




# 이미지 밝기 조정

:
```
gamma = 4.5
img_corrected = cv2.pow(image/255.0, gamma)
img_corrected = img_corrected*255.0

plt.imshow(img_corrected)
plt.show()
```

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for in
```



## 실습03. 웹크롤링 – request, beautifulSoup

# 웹크롤링 Requests, BeautifulFoup 라이브러리 사용해서 크롤링 해보기

```
!pip install requests
!pip install beautifulsoup4==4.11.1
```

```
Collecting requests
  Using cached requests-2.31.0-py3-none-any.whl (62 kB)
Collecting charset-normalizer<4,>=2
  Downloading charset_normalizer-3.1.0-cp38-cp38-win_amd64.whl (96 kB)
     ---------------------------------------- 0.0/96.4 kB ? eta -:--:--
     ---------------------------------------- 96.4/96.4 kB 5.7 MB/s eta 0:00:00
Collecting urllib3<3,>=1.21.1
  Downloading urllib3-2.0.3-py3-none-any.whl (123 kB)
     ---------------------------------------- 0.0/123.6 kB ? eta -:--:--
     ---------------------------------------- 123.6/123.6 kB ? eta 0:00:00
Requirement already satisfied: idna<4,>=2.5 in c:\users\user\anaconda3\envs\ai\lib\site-packages (from requests) (3.4)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\user\anaconda3\envs\ai\lib\site-packages (from requests) (202
3.5.7)
Installing collected packages: urllib3, charset-normalizer, requests
Successfully installed charset-normalizer-3.1.0 requests-2.31.0 urllib3-2.0.3
Collecting beautifulsoup4==4.11.1
  Downloading beautifulsoup4-4.11.1-py3-none-any.whl (128 kB)
     ---------------------------------------- 0.0/128.2 kB ? eta -:--:--
     ---------------------------------------- 128.2/128.2 kB 7.9 MB/s eta 0:00:00
Requirement already satisfied: soupsieve>1.2 in c:\users\user\anaconda3\envs\ai\lib\site-packages (from beautifulsoup4==4.1
1.1) (2.4.1)
Installing collected packages: beautifulsoup4
  Attempting uninstall: beautifulsoup4
    Found existing installation: beautifulsoup4 4.12.2
    Uninstalling beautifulsoup4-4.12.2:
      Successfully uninstalled beautifulsoup4-4.12.2
Successfully installed beautifulsoup4-4.11.1
```

```
# !pip list   # 패키지 리스트 확인(requests==2.28.1, beautifulsoup4==4.11.1
```

```
import requests
from bs4 import BeautifulSoup
import os
```

```
# 크롤링하고 싶은 키워드
query = "수박"
url = f'https://www.google.com/search?q={query}&source=lnms&tbm=isch'  # q= 가 검색어 들어가는 부분으로 쿼리

# https://www.google.com/search?q=검색어&source=lnms&tbm=isch로 검색창 주소를 긁어와 사용해도 된다.

header = {
      "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Sa
}

response = requests.get(url, headers=header)
# print(response.text) # 이 중에서 img 태그를 가진것들이 이미지 파일 정보를 가진 것들

soup = BeautifulSoup(response.text, "html.parser") # parser해서 한번 정제(파싱(Parsing)은 주어진 데이터나 문서를 구문 분석하0
#print(soup)

img_tags = soup.find_all("img") # img가 포함된 정보만 가져오기
#print(img_tags)



#이제 src만 따로 뽑아와야 한다
urls_list = []

for img_tag in img_tags:
    try:
        urls_list.append(img_tag['src']) # src 태그가 있으면 append하라
    except KeyError :
        try :
            urls_list.append(img_tag['data-src'])# 위에 것이 키에러로 안되면 이것으로
        except KeyError:
            try:
                urls_list.append(img_tag['data-iurl']) # 위에 것이 키에러로 안되면 이것으로
            except KeyError:
                pass
print(len(urls_list))



# 이미지 저장
os.makedirs("./image01_data/", exist_ok=True) # 저장 파일 생성

for i, url in enumerate(urls_list) :
    print(i,url)
    try :
        img_data = requests.get(url, headers=header).content
        with open(f"./image01_data/watermelon_{i}", "wb") as f:
            f.write(img_data)

        if os.path.getsize(file_path) == 0 :  # 사이즈가 0인 파일은 삭제
            os.remove(file_path)

    except:
        pass
```

```
21
0 /images/branding/searchlogo/1x/googlelogo_desk_heirloom_color_150x55dp.gif
1 https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcQ5mHtmhpQxZmHV5K8BayJ2P87Q8Ai_Ohw2-TrayKTPctbOHeXTPLp2qYTdvvY&s
2 https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcT5zslUHE42Q5RQQzSGbGd1-m7VkSIIrjTmspdukKPUfqd4MQweycVvKf4oF2O&s
3 https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcSDm2Wr8ADTyCHQw4rBLMlesKYHd_j8yw7SMyOtEOFolTLadXEe3Sj4C6jZI98&s
4 https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcQhT9rREbI-3kTnXG6kh9Fn_g8_hduMsBLSbQSW0anFkgjtGSIUi-dMPM3x4tk&s
5 https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcShTwqmmWUhTaW09QoMckOMW92RL7CFxftdSCgRIn5ZF0K2rpdXg9--MTOOjKY&s
6 https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcQ8I7c0DawAsEHBUPwPXQcplcaAoUxTnjhYO1Qo-D5pAA6rLxTGjbohjnKrDA&s
7 https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcS-xvGi8bm5-dtJX4EPkBXIkYbDely4Yb_LU8ovutXqlvfOSOIg89BU8-IVOg&s
8 https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcSCq_Ik5JmtoxzYPs_Y2qP9Wq48Oc-mjsxw0XqENF_t-HeAphXDI2ZibhNCHb8&s
9 https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcRGk6mwUcxaJJKDb4yzfNv4PEXneUjrDGBRDpT1COXOHBQS6a3kzZGBK6OMOQ&s
10 https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcRJ9d2KUbaE31ZH_5yFA-FXnVRfB7BvQKuJNRCnVf9EDkPomQcAy6fT1wONPJM&s
11 https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcTthTibVIphgdGIXDFZhgJELuuQBd3828T34PcxnmJ6gPARbATUUPsSQI5-HA&s
12 https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcSygT_i1Wj__cannN1HLyH4EO4kpYoqfByz8n9FfUri6XaeBNKH981vER6xLA&s
13 https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcStbxZtReJVWVUtHOoEp9nYGoA_hpGOfW_DZuTOoHOsWvnNv_OM_9IxmfkdSg&s
14 https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcT6BwYqrdw-MdW6CHDg_ESGOlxlTyTwp-xlyRIGLzZBIkhaqKoTEBEueHcuBR8&s
15 https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcQSVXOEFRqaOQPzPmnJColvwZuQMebT36mU5yDtkq15quz4ZWZnGOsJDYQnYQ&s
16 https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcTM6fNFKEDSd5jiuuy71CML5vDsr-OK9S_HMvCELnCQEgycAyOL_YswaXYtJqU&s
17 https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcTsLnFTLYqxuxVGhVLReZu8QBCtSZOwOSrXMYXhQV5pcyAPzkwz2qv0-nf8cA&s
18 https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcQxcvlEjnZjMOC6Jfbpx6gidP488AD4-xi2ROBfMPrQOoXokmi2PR_706nl9pw&s
19 https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcSYPs2Ni6H9hQQbQjA-XrRdJvP21Ce_1NI6B8LOtRyarOpYlbM_NjPqmDQq1KI&s
20 https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcSAsL-pBnaS8k6abTcf1vrUb26kcoY5vmYMXIGVUDqZ884MommbH2YZOPfJQEA&s
```

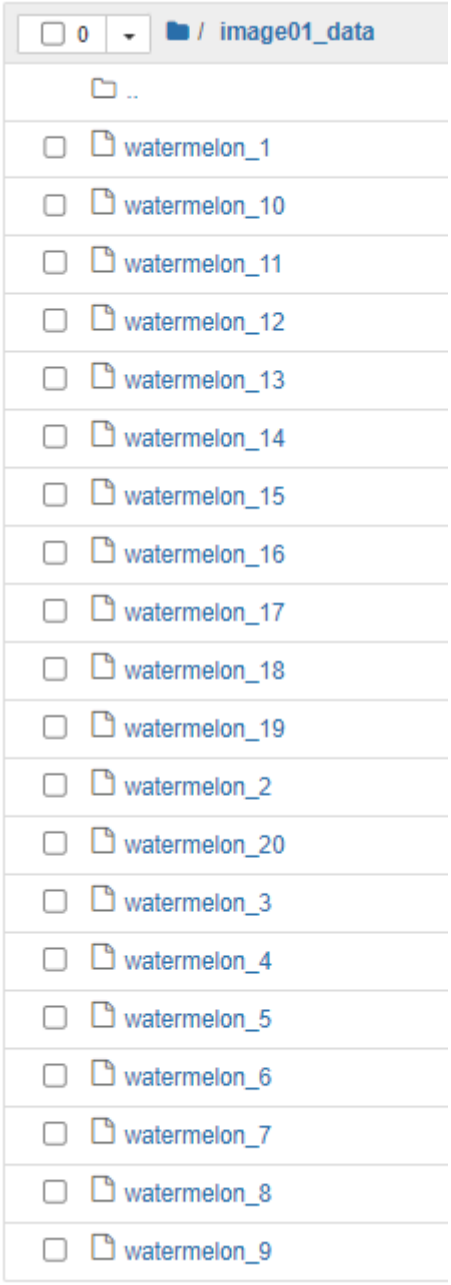



Image01_data 폴더 아래 이미지들 저장된다

## 실습04. 웹크롤링02 – selenium

```
import selenium
```

크롬 버전을 설정 -> 크롬정보에서 확인하고 해당 버전에 맞는 chromedriver 아래 링크에서 찾아 설치 https://chromedriver.chromium.org/downloads

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.chrome.service import Service

import time
import os
import urllib, requests
```

```
# 리눅스나 맥 사용자의 경우 실행 권한 주기
!chmod +x chromedriver
```

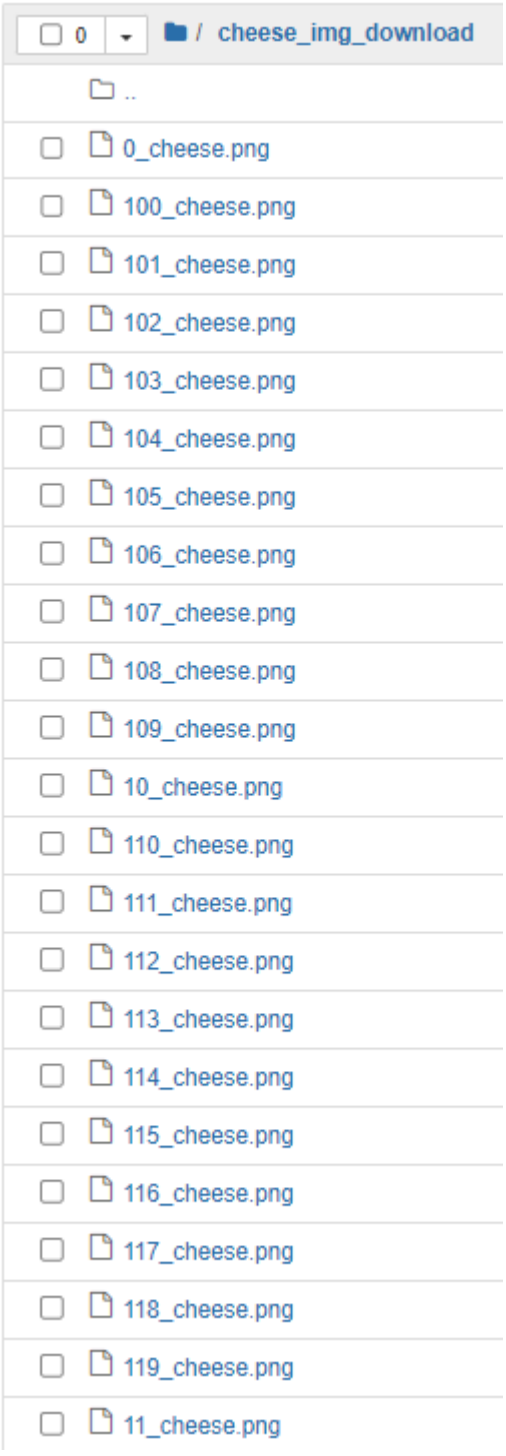
```
'chmod'은(는) 내부 또는 외부 명령, 실행할 수 있는 프로그램, 또는
```

```
### 1. 쿼리 선언 / Chromedriver 실행
query = "cheese"
service = Service("./chromedriver")
driver = webdriver.Chrome(service=service)

### 2. 쿼리 검색창에 추가
driver.get("https://www.google.co.kr/imghp?hl=ko")

 # 크롬 창 열리면 개발자 도구 열어서 검색창 선택하여 검색창 엘레먼트 셀렉트로 선택한 후 엘레먼트 창에 선택된 거 copy full Xpa.

keyword = driver.find_element_by_xpath("/html/body/div[1]/div[3]/form/div[1]/div[1]/div[1]/div/div[2]/textarea")
keyword.send_keys(query) # 이제 검색창에 검색어기 들어가 있는 게 보인다

### 3. 검색창에 입력어 들어오면 검색 실행
driver.find_element_by_xpath("/html/body/div[1]/div[3]/form/div[1]/div[1]/div[1]/button").click()

driver.implicitly_wait(3)

### 4. 스크롤 자동으로 내리고 더보기 버튼 나오면 클릭 하기
print(f"{query} 스크롤 내리는 중...")
elem = driver.find_element_by_tag_name('body')
for i in range(200):
    elem.send_keys(Keys.PAGE_DOWN)
    time.sleep(0.1)

try:
    driver.find_element_by_class_name('mye4qd').send_keys(Keys.ENTER)
    for i in range(200):
        elem.send_keys(Keys.PAGE_DOWN)
        time.sleep(0.1)
except Exception:
    pass
```

```
### 5. 이미지 개수 파악 하기
links = []
images = driver.find_elements_by_css_selector('img.rg_i.Q4LuWd')
for image in images:

    if image.get_attribute('src') != None :
        links.append(image.get_attribute('src'))
    elif image.get_attribute('data-scr') != None :
        links.append(image.get_attribute('data-src'))
    elif image.get_attribute('data-iurl') != None:
        links.append(image.get_attribute('data-iurl'))

print("찾은 이미지 개수 : ", len(links))



### 6. 이미지 다운로드
count = 0
for i in links :
    start = time.time()
    url = i
    os.makedirs(f"./{query}_img_download/", exist_ok=True)
    while True:
        try:
            urllib.request.urlretrieve(url, f"./{query}_img_download/{str(count)}_{query}.png")
            print(f"{str(count + 1)} / {str(len(links))} / {query} / 다운로드 시간 : {str(time.time() - start)[:5]} 초")
            break
        except urllib.error.HTTPError as e:
            print(f"HTTPError 발생 ({e}): 재시도 중...")
            time.sleep(5)
        except Exception as e:
            print(f"Error 발생 ({e}): 재시도 중...")
            time.sleep(5)
        if time.time() - start > 60:
            print(f"{query} 이미지 다운로드 실패")
            break
    count += 1

print(f"{query} 이미지 다운로드 완료")
driver.close()
```

```
찾은 이미지 개수 :  538
1 / 538 / cheese / 다운로드 시간 : 0.001 초
2 / 538 / cheese / 다운로드 시간 : 0.001 초
3 / 538 / cheese / 다운로드 시간 : 0.000 초
4 / 538 / cheese / 다운로드 시간 : 0.000 초
5 / 538 / cheese / 다운로드 시간 : 0.001 초
6 / 538 / cheese / 다운로드 시간 : 0.000 초
7 / 538 / cheese / 다운로드 시간 : 0.000 초
8 / 538 / cheese / 다운로드 시간 : 0.001 초
9 / 538 / cheese / 다운로드 시간 : 0.001 초
10 / 538 / cheese / 다운로드 시간 : 0.000 초
11 / 538 / cheese / 다운로드 시간 : 0.000 초
12 / 538 / cheese / 다운로드 시간 : 0.001 초
13 / 538 / cheese / 다운로드 시간 : 0.000 초
```

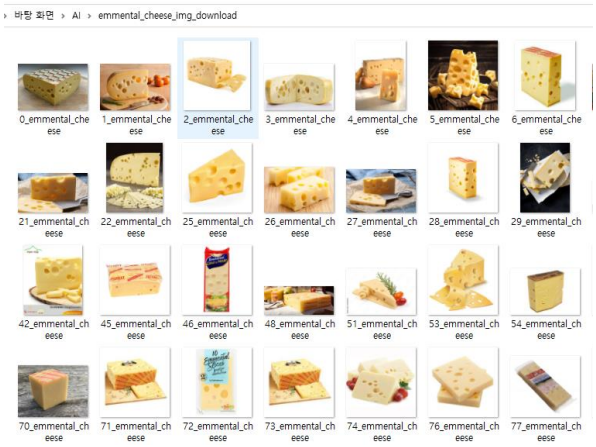
찾은 이미지 개수 출력



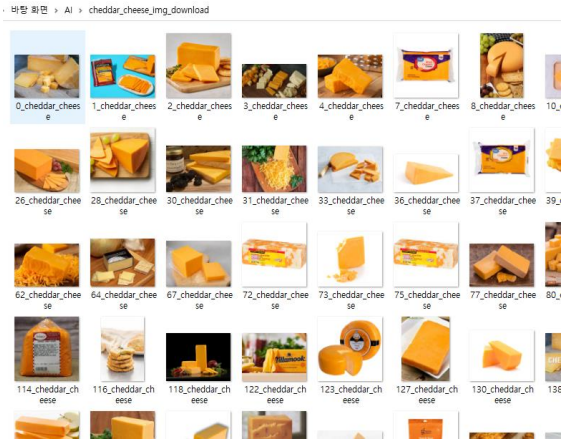

이미지 저장

**실습05.이미지 정제**

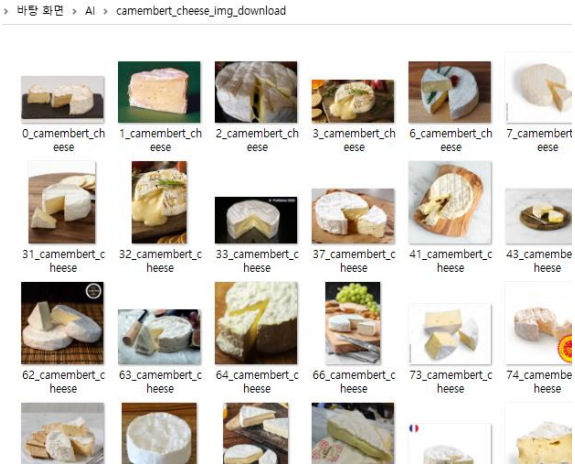
분류 기준을 치즈 종류로 하기로 결정, 4가지 종류의 치즈로 분류하기로 하였다.(4가지 치즈 종류로 이미지 다시 크롤링하여 재진행 : emmental /cheddar/camembert/blue )



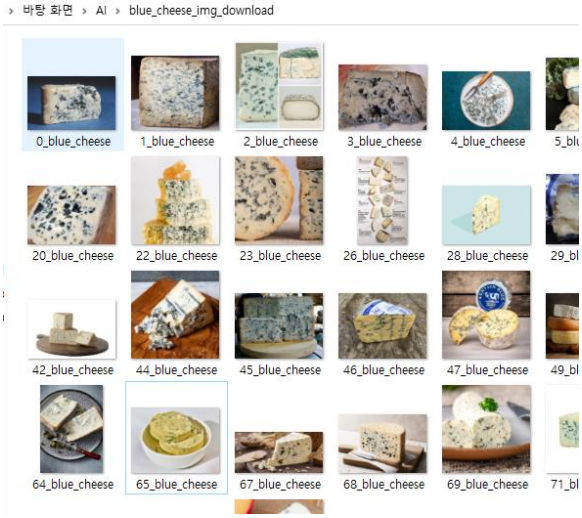

에멘탈




체다




까망베르




블루