# Prob2

영상 파일 검색
영상 파일 출력 프로그램


기타 제출일 : 2020.12.07

작성자 소속 : 산업경영시스템공학과

학번 : 2018316501

성명 : 박민지

## 서론

이번 과제는 디렉토리 안에 모든 파일 중 지정된 영상 파일(jpg, png, tif)을 찾아서 파일들의 개수와 이름, 파일의 크기의 총합을 파악하여 출력하는 프로그램입니다. 또한, 그림 파일 중 용량이 제일 큰 파일과 용량이 제일 적은 파일, 영상의 높이가 높은 파일, 영상의 가로가 제일 넓은 파일 총 4개의 사진을 2X2 서브화면에 출력합니다.

## 본론

### 1. 하위 디렉토리에 존재하는 영상파일 구하기

```
def allFiles(path):  # 하위 디렉토리에 존재하는 jpg, png, tif 구하기
    test = []
    for current, dir, files in os.walk(path):
        currentpath = os.path.join(os.path.abspath(""), current)
        for file in files:  # 파일 확장자가 다음과 같으면 주소와 파일명 합치기
            if file.endswith('.jpg'):
                filepath = os.path.join(currentpath, file)
                test.append(filepath)
            elif file.endswith('png'):
                filepath = os.path.join(currentpath, file)
                test.append(filepath)
            elif file.endswith('.tif'):
                filepath = os.path.join(currentpath, file)
                test.append(filepath)
            elif file.endswith('.JPG'):
                filepath = os.path.join(currentpath, file)
                test.append(filepath)
            elif file.endswith('.PNG'):
                filepath = os.path.join(currentpath, file)
                test.append(filepath)
            elif file.endswith('.TIF'):
                filepath = os.path.join(currentpath, file)
                test.append(filepath)
```

```
    return test
```

➔ test라는 배열을 하나 생성합니다. os모듈을 사용해 파일의 현재 위치와 파일명을 받아옵니다. for문을 돌려 파일의 끝이 '.jpg, .png, .tif'로 끝나면 test배열에 넣습니다. 그리고 test를 반환합니다.

### 2. 영상 크기 구하기

```
def getsize(path_dir):   # 영상 크기 구하기
    file_size = os.path.getsize(path_dir)
    return file_size
```

➔ 원하는 영상파일의 주소를 불러온 뒤 getsize를 사용해 파일 크기를 구합니다.

### 3. 영상 파일 개수 구하기

```
def getSortData(name):
    jpg = []
    png = []
    tif = []

    for i in name:
        if ".jpg" in i:
            jpg.append(i)
        elif ".png" in i:
            png.append(i)
        elif ".tif" in i:
            tif.append(i)
        elif ".JPG" in i:
            jpg.append(i)
        elif ".PNG" in i:
            png.append(i)
        elif ".TIF" in i:
            tif.append(i)
    return len(jpg), len(png), len(tif)
```

➔ 'jpg, png, tif'로 분류하기 위해 위 3개의 배열을 만들고 이름에 jpg가 있으면 jpg배열에 저장하고, png가 있으면 png배열에 저장하고 tif가 있으면 tif배열에 저장합니다.그리고 jpg, png, tif의 개수를 반환합니다.

### 4. 영상 파일 사이즈 제일 작은거 구하기

```
def Min_Size(name, size):  # 파일사이즈 제일 작은거
    lst1 = []  # 파일명
    lst2 = []  # 파일 사이즈

    for i in name:
        lst1.append(i)
    for i in size:
        lst2.append(i)

    for i in range(len(lst2)):
        for j in range(len(lst2)):
            if lst2[i] < lst2[j]:
                temp = lst2[i]
                lst2[i] = lst2[j]
                lst2[j] = temp

                temp = lst1[i]
                lst1[i] = lst1[j]
                lst1[j] = temp

    return lst1[0], lst2[0]
```



➔ 영상 파일명과 영상 파일 사이즈를 같이 입력 받아 lst1, lst1에 집어넣은 뒤 for문을 사용하여 작은 것을 찾아 셔플 합니다. 그리고 제일 작은 파일명(lst1[0])과 제일 작은 파일 크기(lst2[0])를 반환합니다.

### 5. 영상 파일 사이즈 제일 큰거 구하기

```
def Max_Size(name, size):  # 파일 사이즈 제일 큰거
```

```
    lst1 = []  # 파일명
    lst2 = []  # 파일 사이즈

    for i in name:
        lst1.append(i)
    for i in size:
        lst2.append(i)

    for i in range(len(lst2)):
        for j in range(len(lst2)):
            if lst2[i] > lst2[j]:
                temp = lst2[i]
                lst2[i] = lst2[j]
                lst2[j] = temp

                temp = lst1[i]
                lst1[i] = lst1[j]
                lst1[j] = temp

    return lst1[0], lst2[0]
```

➔ 영상 파일명과 영상 파일 사이즈를 같이 입력 받아 lst1, lst1에 집어넣은 뒤 for문을 사용하여 큰 것을 찾아 셔플 합니다. 그리고 제일 큰 파일명(lst1[0])과 제일 큰 파일 크기(lst2[0])를 반환합니다.



### 6. 영상 파일 높이가 높은 파일 구하기

```
def heightSize(name):
    n = []
    hei = []
    wid = []
    channel = []

    for i in name:  # 파일 명을 src_folder로 잘라서 mission 이후의 폴더 이름만 n에 저장
        j = i.split(src_folder)
```

```
        k = j[1]
        l = os.path.join(src_folder, k)
        n.append(l)
    for i in range(len(n)):  # 저장된 주소를 불러와서 img에 저장 후 그 이미지의 h,w,c 구함
        img = cv.imread(n[i])
        heig, wide, chan = img.shape
        hei.append(heig)
        wid.append(wide)
        channel.append(chan)

    for i in range(len(n)):
        for j in range(len(n)):
            if hei[i] > hei[j]:
                temp = hei[i]  # 크기 위치 변경
                hei[i] = hei[j]
                hei[j] = temp

                temp = wid[i]
                wid[i] = wid[j]
                wid[j] = temp

                temp = n[i]  # 크기 위치로 이름 변경
                n[i] = n[j]
                n[j] = temp

    return n[0], hei[0], wid[0]
```



➔ 받아올 파일 이름이 드라이브에서부터 시작하기 때문에 src_folder 위치부터 시작해서 영상파일의 위치를 받아오기 위해 src_folder로 자릅니다. 그리고 그 뒷부분을 k에 저장하고 k에는 src_folder 뒷부분부터 잘랐기 때문에 l에 src_folder와 자른 뒷부분을 합쳐줍니다. 그리고 n에 넣어줍니다.

➔ 저장된 주소를 불러와 imread를 한 후 h,w,c를 구합니다. 높이 기준으로 제일 높은 이미지를 이름과 높이, 너비를 반환합니다.

## 7. 영상 파일 가로가 제일 넓은 파일 구하기

```
def wideSize(name):
    n = []
    hei = []
    wid = []
    channel = []

    for i in name:  # 파일 명을 src_folder 잘라서 mission 이후의 폴더 이름만 n에 저장
        j = i.split(src_folder)
        k = j[1]
        l = os.path.join(src_folder, k)
        n.append(l)

    for i in range(len(n)):  # 저장된 주소를 불러와서 img에 저장 후 그 이미지의 h,w,c 구함
        img = cv.imread(n[i])
        heig, wide, chan = img.shape
        hei.append(heig)
        wid.append(wide)
        channel.append(chan)

    for i in range(len(n)):
        for j in range(len(n)):
            if wid[i] > wid[j]:
                temp = hei[i]  # 크기 위치 변경
                hei[i] = hei[j]
                hei[j] = temp

                temp = wid[i]
                wid[i] = wid[j]
                wid[j] = temp

                temp = n[i]  # 크기 위치로 이름 변경
                n[i] = n[j]
                n[j] = temp
```

```
    return n[0], hei[0], wid[0]
```

➔ 높이 구했던 것처럼 똑같이 실행하지만 너비를 기준으로 제일 긴거로 셔플을 해줍니다.

### 8. 영상 파일 이름 구하기

```
def namesplit(name):
    y = name.split('/')
    y.reverse()
    splitname = y[0]

    return splitname
```

➔ 위치를 제외하고 영상 파일 이름만 구하기 위해 '/'로 자른뒤 뒤집어서 y[0]을 출력합니다.

### 9. 출력하기

```
name1 = allFiles(src_folder)      # 이미지 파일의 디렉토리 주소
name = []
size = []          # 이미지 파일의 크기
for i in name1:
    s = i.replace("₩₩", "/")
    name.append(s)
    size.append(getsize(i))

jpg, png, tif = getSortData(name)          # 각 확장자 마다 이미지 파일의 개수

MaxName, MaxSize = Max_Size(name, size)
MinName, MinSize = Min_Size(name, size)
heina, heihei, heiwid = heightSize(name)   # 세로가 제일 높은 이미지
widna, widhei, widwid = wideSize(name)       # 가로가 제일 높은 이미지
```

```
total = jpg + png + tif       # 이미지 파일의 개수
totalsize = 0    # 이미지 파일의 총 크기
for i in size:
    totalsize += i

print('Total number of picture files : {0}(jpg : {1}, png : {2}, tif : {3})'.format(total, jpg, png, tif))
print('Total size of files : ', end=" "), print_num(totalsize)


# 디렉토리 주소에서 파일명과 확장자명만 출력하기
Maxfilename = namesplit(MaxName)
Minfilename = namesplit(MinName)
widname = namesplit(widna)
heiname = namesplit(heina)


# 영상 출력
img = imread(MaxName)
plt.figure()
plt.subplot(221), plt.imshow(img)
plt.axis('off')
plt.title('max={0} ({1})'.format(Maxfilename, MaxSize))

img = imread(MinName)
plt.subplot(222), plt.imshow(img)
plt.axis('off')
plt.title('min={0} ({1})'.format(Minfilename, MinSize))

img = imread(heina)
plt.subplot(223), plt.imshow(img)
plt.axis('off')
plt.title('tall={0} ({1},{2})'.format(heiname, heihei, heiwid))

img = imread(widna)
```

```
plt.subplot(224), plt.imshow(img)
plt.axis('off')
plt.title('wide={0} ({1},{2})'.format(widname, widhei, widwid))

plt.show()
```

➔ Name과 size에 이미지 파일의 주소와 크기를 넣어줍니다. getSortdata를 사용해 각 확장자별로 이미지 파일을 분류합니다. Max_Size를 사용하여 이미지 파일중 가장 큰 이미지를 구하고, Min_Size를 사용하여 가장 작은 이미지를 구합니다. heightSize를 사용해 세로가 제일 높은 이미지를 구하고 wideSize를 사용해 가로가 제일 높은 이미지를 구합니다. total에는 총 이미지파일 개수를 totalsize에는 총 이미지 파일의 크기를 넣습니다.

➔ Matplotlib를 사용해 첫번째 칸에는 제일 큰 이미지를, 두번째 칸에는 제일 작은 이미지를, 세번째 칸에는 가로가 제일 높은 이미지를, 네번째 칸에는 세로가 제일 긴 이미지를 출력합니다.

## 결론

```
Total number of picture files : 6(jpg : 4, png : 2, tif : 0)
Total size of files :  810,637 ( 81만6백3십7 )
```

이미지 파일은 총 6개로 jpg가 4개 png가 2개입니다. 이미지 파일의 총 크기는 81만 6백 3십7입니다.



plt창을 출력시키면 제일 큰 이미지는 bk, 제일 작은 이미지는 j, 세로가 제일 긴 이미지는 apt, 가로가 제일 긴 이미지는 fighter입니다.