파일 입출력

- ★ 키보드 입력을 표준 입력, 모니터 출력을 표준 출력이라 함
- ★ input( ) 함수는 키보드로부터 데이터를 입력 받는 함수
- ★ print( ) 함수는 모니터로 결과를 출력하는 함수(옵션을 변경하여 파일에 출력 가능)
- ★ 하지만, 입력 데이터의 양이 많아지면 손으로 직접 입력하기 어렵고, 출력 데이터의 양도 많아져 파일에 보관해야 할 필요성도 생김
- ★ 본 강의에서는 문자열로 이루어진 파일 입출력을 다룸
- ★ 파일 입출력은 다음과 같은 세 단계를 통하여 이루어짐
  1. 파일 열기 (File Open)
  2. 파일에서 읽거나 또는 파일에 쓰기
  3. 파일 닫기 (File Close)

file open

★ 형식

```
fp_r = open("in.txt", 'r')          # read mode
fp_w = open("out.txt", 'w')         # write mode
```

- **fp_r**, **fp_w** : 파일 객체를 나타내는 변수
- **in.txt**, **out.txt** : 파일 이름 (사용자 지정 이름)
- File Mode : 파일을 어떤 용도로 사용할지 결정

  'r' : 읽기 전용. 존재하지 않는 파일이면 에러 발생

  'w' : 쓰기 전용. 파일이 없으면 빈 파일을 새로 만들고, 기존 있던 파일이면 내용을 모두 지우고 파일을 쓰기 전용으로 open.

  'a' : append의 약자이며, 기존 파일에 덧붙여서 이어서 쓰기 작업을 함

입출력 파일 위치

- script 파일이 저장되어 있는 폴더에 저장되어 있는 파일을 읽거나 쓰는 경우는 파일 이름만 명시함

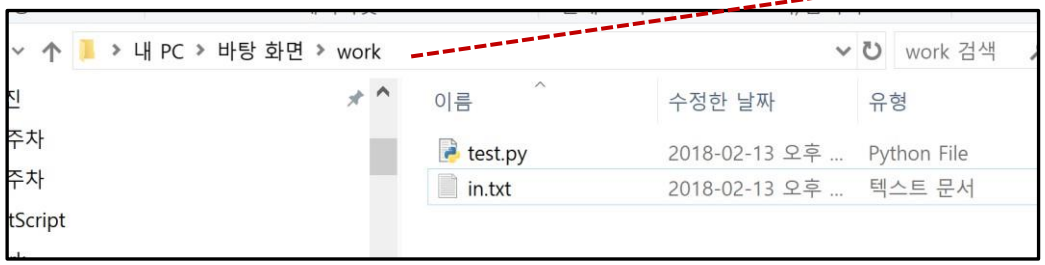
```
fp_r = open("in.txt", 'r')              # read mode
```

- script 파일과 다른 폴더에 저장되어 있는 파일을 읽거나 쓰는 경우는 전체 경로를 파일 이름과 함께 명시해야 함

```
fp_r = open("C:₩₩Users₩₩SOGANG₩₩Desktop₩₩work₩₩in.txt", 'r')
```

₩를 문자열에 표현하기 위해서 ₩₩로!

- 파일의 전체 경로는 파일 탐색기 창에서 아래와 같은 방법으로 얻을 수 있음



마우스 클릭
Ctrl-C (*경로 복사*)
Ctrl-V (script*에 붙이기*)

file read

- open() 함수에서 읽기 모드로 반환 받은 파일 객체를 가리키는 변수를 fp_r 라고 가정
- **fp_r.read( )** : 파일 전체를 하나의 문자열로 읽어 들임
- **fp_r.readlines( )** : 파일을 줄 단위로 읽어 각 줄을 문자열 형식으로 저장, 이들 전체 문자열을 원소로 하는 리스트를 반환
- **fp_r.readline( )** : 한 줄을 문자열 형식으로 읽어 이를 반환

- 파일 전체를 한 줄씩 읽어 처리하는 방법 : for loop를 사용하면 편리

```
for aLine in fp_r :          # 변수 aLine은 fp_r에서 한 줄씩 읽어 들인 문자열
    statements               # 읽어들인 줄에 처리할 명령어들
```

file write

- ★ open() 함수에서 쓰기 모드로 반환 받은 파일 객체를 가리키는 변수를 fp_w 라고 가정
- ★ **fp_w.write(string)** : string 문자열 하나를 파일에 씀
  - 줄바꿈 문자가 자동으로 삽입되지 않음
  - 줄바꿈이 필요한 경우, string의 마지막에 줄바꿈 기호를 추가해야 함
- ★ **print(*objects, file = fp_w)** : 표준 출력 함수 사용 가능
  - 화면 대신 파일에 쓰기 위해서는 file 인수 값 변경
  - print() 함수는 줄바꿈이 자동
  - 줄바꿈을 하지 않으려면, end 인수 값 변경

```
print(*objects, file = fp_w, end = "")
# file = fp_w  : file 인수값을 기본 값인 표준 출력에서 원하는 파일 객체
                 변수명으로 변경
# end = "" : 출력시 마지막 문자열 인수값을 기본 값인 줄바꿈이 아닌
             빈문자열로 변경
```

file close

★ 파일을 열었다면 반드시 파일을 닫아야 함

```
fp_r = open("in.txt", 'r')          # read mode
fp_w = open("out.txt", 'w')         # write mode
```

```
fp_r.close()
fp_w.close()
```

- 파일이 열려있는 상태에서는 파일을 지우거나 이름을 바꿀 수 없음
- Python shell을 종료하거나, restart하거나 또는 close method를 호출하여야 파일이 닫혀 파일 지우기 또는 이름 바꾸기 등을 수행할 수 있음
- 따라서, 파일 작업을 마치면 반드시 파일을 닫도록 함

파일 입출력 예제

★ 한 줄씩 읽어 화면에 출력하기

```
fp = open("in_data.txt", 'r')
for s in fp :                    # 한 줄씩 읽어
    print(s, end = "")           # 화면에 출력
fp.close()
```

출력 후 줄바꿈 자동 출력 안함

in_data.txt(*입력 파일*)

```
1 3 5
7 3
```

파일에 데이터 입력시 줄바꿈 문자가 저장되어 있음(enter 키)

화면 출력

```
1 3 5
7 3
```

print() *함수에서 자동 줄바꿈을 변경 했지만, 기존 데이터 파일의 줄 끝에 줄바꿈 문자가 저장되어 있어서 줄바꿈이 발생*

파일 입출력 예제

★ 파일을 줄 단위로 읽어 문자열 리스트 만들기

```
fp = open("in_data.txt", 'r')
s = fp.readlines()  # 각 줄이 문자열로 저장, 이를 리스트 원소로 저장
print(s)
fp.close()
```

in_data.txt(*입력 파일*)

```
1 3 5
7 3
```

화면 출력

```
['1 3 5₩n', '7 3']
```

줄바꿈 문자가 포함(₩n)

파일 입출력 예제

★ 파일 복사 : 한번에 모두 읽어 복사 하기

```
fp_r = open("in_data.txt", 'r')
fp_w = open("out_data.txt", 'w')
s = fp_r.read()      # 한번에 모두 읽어 하나의 문자열로 저장
fp_w.write(s)        # 읽은 문자열 파일에 쓰기
fp_r.close(); fp_w.close()   # 두 파일 닫기
```

★ 파일 복사 : 한 줄씩 읽어 복사 하기

```
fp_r = open("in_data.txt", 'r')  f
p_w = open("out_data.txt", 'w')
for s in fp_r :      # 한 줄씩 읽어서 쓰기
    fp_w.write(s)  # 또는 print(s, end = "", file = fp_w)

fp_r.close(); fp_w.close()
```

파일 입출력 예제

★ 파일의 각 수를 제곱하여 저장한 파일 만들기

```
fp_r = open("in_mul.txt", 'r')                  #입력파일 in_mul.txt
fp_w = open("out_mul.txt", 'w')                 #출력파일 out_mul.txt

print("The square results are", file = fp_w)    #출력 파일 가리키는 fp_w에 문자열 출력

for s in fp_r :                                 #한 줄씩 읽기
   num = list( int(x) for x in s.split() )      #입력 파일에서 읽어온 문자열 정수로 list에 저장
   for i in range(len(num)) :
         sq = num[i] * num[i]
      print("%2d  " %sq, file = fp_w, end = '')     #출력 후 줄바꿈 자동출력 안함

   fp_w.write("\n")   # 한 줄 처리 후, 줄바꿈
fp_r.close()                                    #입력 파일 close
fp_w.close()                                    #출력 파일 close
```

in_mul.txt

```
1  3  5  8
4  8  9  5  7
```

각 숫자 제곱

out_mul.txt(출력 예시)

```
The square results are
 1   9  25  64
16  64  81  25  49
```

파일 입출력 예제

★ 여러 개의 정수를 입력 받아서 양수와 음수 개수를 출력하는 프로그램
(10장 반복문 예제 파일 입출력 형식으로 변환 –비교 학습)

```
fp = open("numbers.txt","r")                          #파일 입출력을 위한 파일 객체 지정
fp_o = open("result.txt","w")

countP= 0 ; countN= 0; sumNums = 0                    #양수 개수; 음수 개수 ; 전체 합
print("Enter the integer: ", file=fp_o)               # 기존 입력 부분의 출력 내용을  파일에 출력
for s in fp:
   data = list(int(x) for x in s.split())             #기존에는 입력 받으면서 화면에 표시되었던 내용을
print(data, file=fp_o)                                # 읽어와서 리스트로 변환 후 파일에 출력
dataNum = len(data)                                   #입력한 숫자가 없을 경우를 위한 변수

if dataNum == 0:                                      #입력한 숫자가 없는 경우
   print("입력한 숫자가 없습니다", file='fp_o')
else :
   while (dataNum > 0): #dataNum을 인덱스로 반복, dataNum 개수만큼
      if data[dataNum-1] > 0:                                    #양수일 경우
         countP += 1
      elif data[dataNum-1] < 0:                       #음수일 경우
         countN += 1
      sumNums += data[dataNum-1]                      #양수든 음수든 합계 계산
      dataNum -= 1                                    #인덱스 카운트
print("양수:", countP,"개,", "음수 :", countN, "개,", "합계 :", sumNums, file = fp_o)

fp.close()
fp_o.close()
```

★ 출력 예시

```
cat numbers.txt
23 -2 5 89 -43 0 5 0
```

```
cat result.txt
Enter the integer:
[23, -2, 5, 89, -43, 0, 5, 0]
양 수 : 4 개 , 음 수  : 2 개 , 합 계  : 77
```

실습