파일:

* 바이트들이 순차적으로 저장되어 있음
* 맨 끝에는 **EOF(End Of File) 마커**가 있음
* 입출력 동작이 발생하는 위치를 나타내는 **위치 표시자**를 가지고 있음
* 파일을 처음으로 열면 **위치 표시자 = 파일의 첫 번째 바이트**
* **위치 표시자** 자동 업데이트

파일의 종류:

1. 텍스트 파일: 사람이 읽을 수 있는 텍스트가 들어가 있는 파일

ex) 소스 파일

1. 이진 파일: 컴퓨터가 읽을 수 있는 파일

ex) 실행 파일

파일 모드: ( r, w, a 만 잘 알아두면 될 것 같다! )

1. r : 파일의 처음부터 읽는다. 읽기 모드(read mode)
2. w : 파일의 처음부터 쓴다, 파일이 없으면 생성, 있으면 기존 내용 삭제, 쓰기 모드(write mode)
3. a : 파일의 끝에 쓴다, 파일이 없으면 생성, 추가 모드(append mode)
4. r+ : 파일에 읽고 쓸 수 있는 모드, 읽기와 쓰기 모드

파일 열기:

infile = open("input.txt", "r") ## input.txt 이라는 파일을 읽기 모드로 열기

infile = open(r”c:\temp\phones.txt”, “r”) ## 윈도우에서의 절대 경로 설정법

infile = open(“c:\\tmp\\phones.txt”, “r”)

\*\*\* 파일의 이름 앞의 r 접두어 : 문자열을 원천 문자열(raw string)으로 취급하라는 것을 의미

\의 특별한 의미로 해석하지 말라는 것을 의미(이스케이프 시퀀스)

데이터 읽기:

s = infile.read(10) ## 파일 객체에서 문자 10개를 읽어서 출력

s = infile.readline( ) ## 한 줄을 읽음

s = infile.read( ) ## 하나의 문자를 읽음

\*\*\* 일반적으로 파일에 몇 줄이 들어있는지 알 수 없는 경우

line = infile.readline( )

while line !=”” ## 비어있지 않으면 반복문 실행

print(line)

line = infile.readline( ) 다음 줄로 넘어간 상태를 저장

infile.close()

\*\*\* 파일 객체를 문자열의 컨테이너로 간주하므로 for문을 이용해 파일 읽기 가능

infile = open(“input.txt”, “r”)

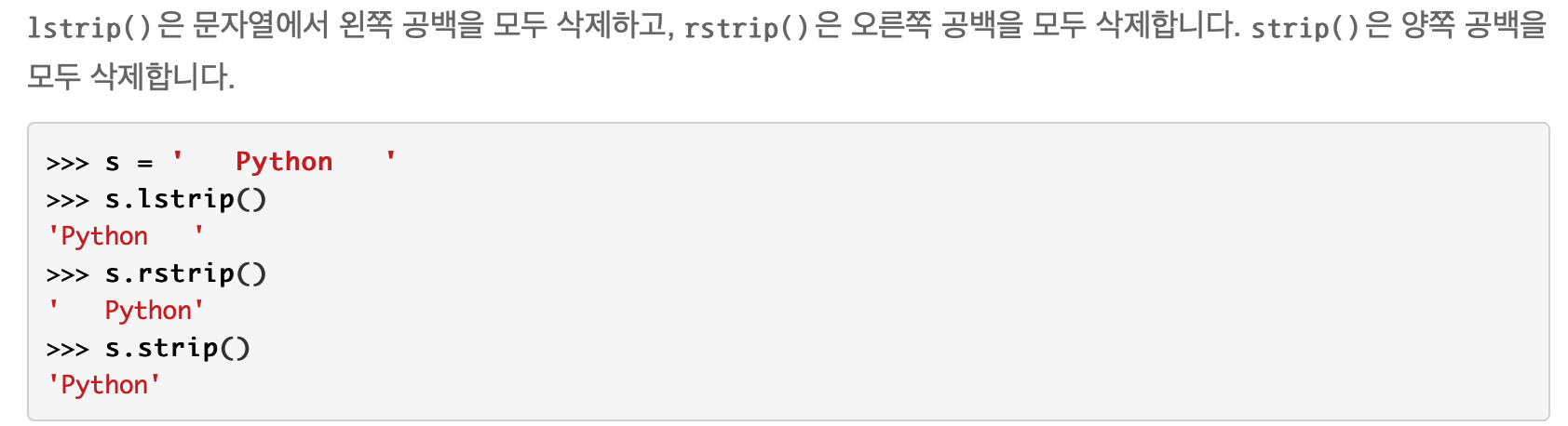
for line in infile:

line = line.rstrip( )

##문자열의 오른쪽(right) 공백 문자 제거 -> 줄 마지막에 붙은 \n 제거 시 유용

print(line)

infile.close( )



데이터 쓰기:

import os.path

outfile = open(“input.txt”, “w”)

if os.path.isfile(“input.txt”): ##파일이 존재하는지 확인

print(“이미 존재합니다.”)

else:

outfile.write(“박세은”)

outfile.write(“강지연”)

outfile.write(“윤재희”)

outfile.wirte(“사케동 먹고 싶은 오윤진”)

outfile.close()

데이터 추가하기:

outfile = open(“input.txt”, “a”)

outfile.write(“으리의 파송송”)

outfile.close( )

파일에서 단어 읽기: 문자열 객체의 split( ) 메소드는 공백 문자를 이용하여 문자열에서 단어 분리

infile = open(“proverbs.txt”, “r”)

for line in infile:

line = line.rstrip( )

word\_list = line.split( ) ##문자열에서 단어 분리

for word in word\_list:

print(word)

infile.close( )

숫자 데이터 처리 방법

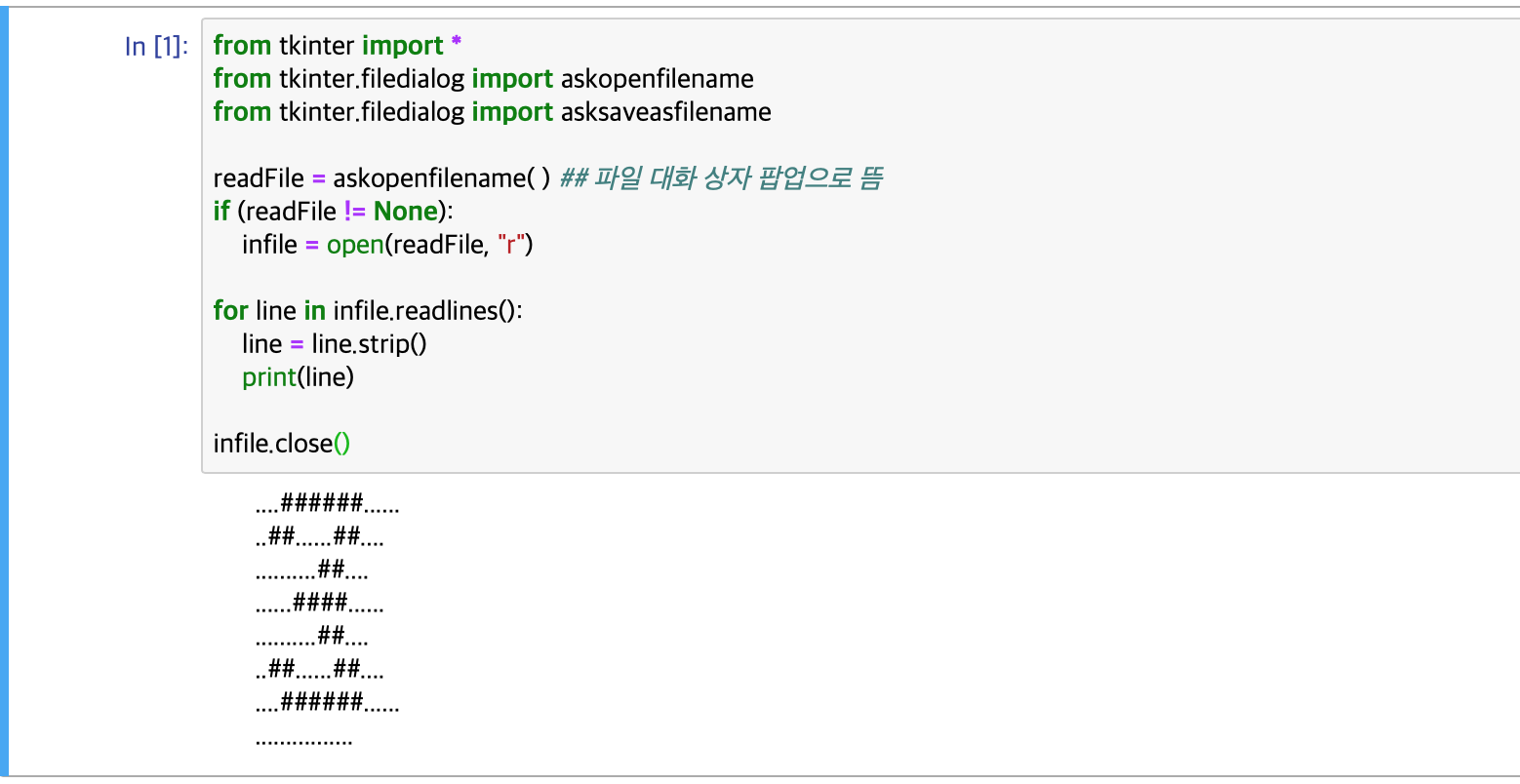
텍스트 파일에는 텍스트 형태의 데이터만 저장 가능

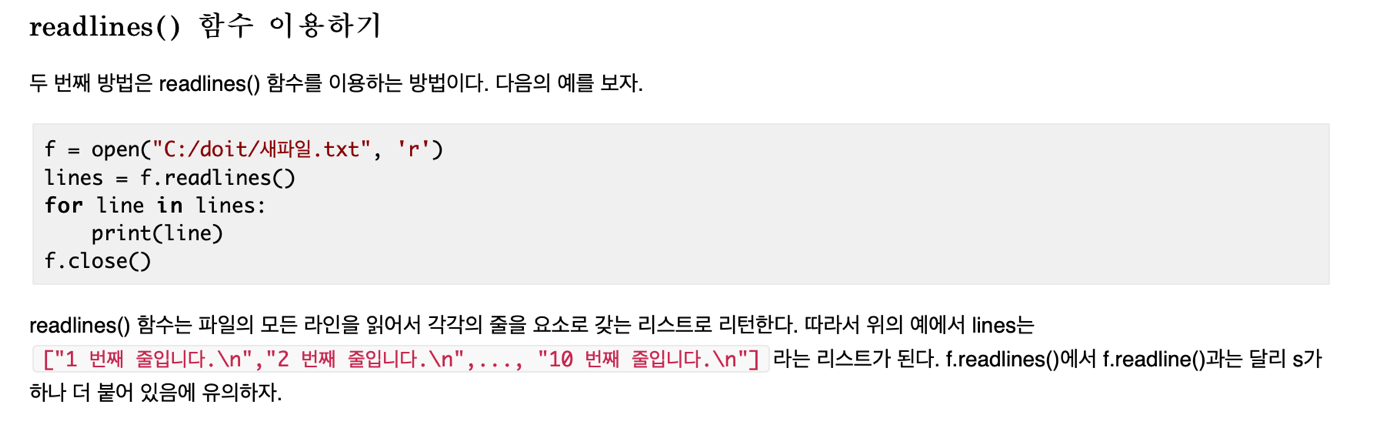
* 숫자 데이터는 문자열로 변환 후에 저장
* str( ) 함수 이용

for i in range(10):

outfile.write(str(i)+” “)

파일 대화 상자





<이진 파일>

* 정수 123456이 문자열로 변환되지 않고 이진수 형태로 그대로 기록되는 것
* 텍스트 파일에 비해 저장 공간도 적게 차지, 시간도 짧게 걸림 🡪 효율적!!
* 이식성 떨어짐 (컴퓨터 시스템마다 데이터 표현 방식이 다를 수 있음)
* 출력 불가능, 파일의 내용 확인 어려움

이진 파일에서 데이터 읽기:

infile = open(filename, “rb”)

입력 파일에서 8바이트 읽기:

bytesArray = infile.read(8) ##bytesArray는 바이트형의 시퀀스, 0~ 255까지의 값들의 모임

바이트 저장:

outfile = open(filename, “wb”)

bytesArray = bytes([255, 128, 0, 1])

outfile.wirte(bytesArray)

## 나도 이해가 어렵다 .. 무슨 말이지?? ##

임의 접근을 하고 싶다!!

임의 접근이란? 원하는 위치에서 읽기/쓰기 하는 것!

infile.seek(position) ##위치 표시자를 조작하는 함수

infile = open(“test.txt”, “r+”)

str = infile.read(10)

position = infile.tell( ) ##현재 위치를 알려주는 함수

position = infile.seek(0, 0) ##파일의 처음으로 이동

infile.close()