**3강. 함수와 파일 다루기**

**3.1 함수 호출로 입출력받아 처리하기**

앞에서 언급한 것처럼 입출력을 원활히 하려면 프로그램에 ‘변수(variable)’를 잘 활용해야 한다. 변수와 함수(function)를 이용하여 입출력 처리과정을 확인해보자.

파이썬에서는 여러 유용한 내장함수(built-in function)들을 사용할 수 있다. IDLE 인터랙티브 모드에서 사용자로부터 입력을 받을 때는 raw\_input()이라는 함수를 사용하면 된다.

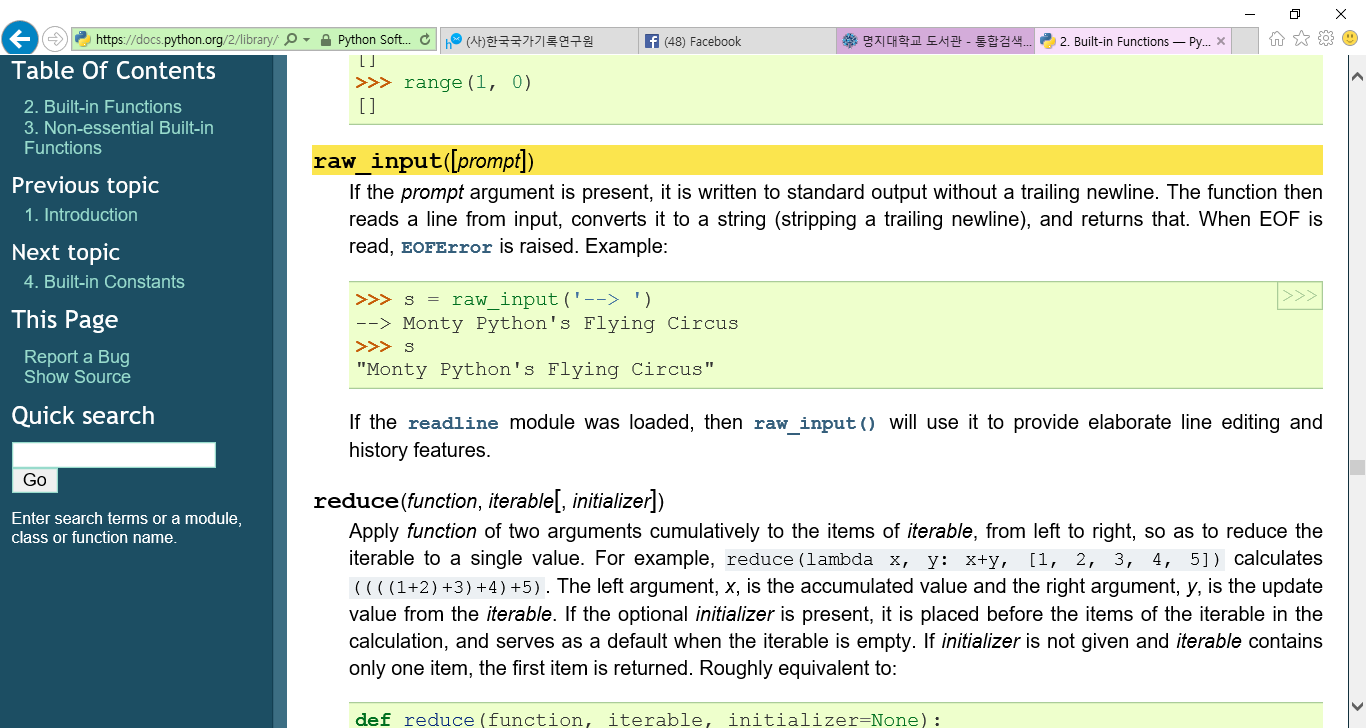
**() 괄호가 있으면 함수다**

구글링을 통해 파이썬의 정보를 찾을 때는 “Python raw\_input()”과 같이 검색을 하면 된다. 다음 URL에 파이썬의 내장함수 목록이 제시되어 있다. raw\_input()을 클릭하면 상세 사용법을 볼 수 있다.

[https://docs.python.org/2/library/functions.html](http://docs.python.org/2/library/functions.html)

▼

>>> s 이름의 변수 [오른쪽] = raw\_input (‘-->’) [넣은 값을 의미]



함수에 대한 설명정보에서 **괄호 안의 이름들은 인수(argument)**들이다. prompt는 인수명으로 함수를 이해하기 쉽게 의미있는 이름을 지어준 것이다. 해당 함수를 호출할 때마다 인수에 다른 입력값을 줄 수 있다.

raw\_input함수의 인수 prompt는 화면에서 입력을 기다리릴 때 띄워주는 신호를 의미한다. []의 의미는 함수를 호출할 때 인수값이 필수적으로 주어져야 하는 것은 아님을 뜻한다.

함수에 대한 설명에서 함수의 수행결과 값으로 무엇이 **반환(return value)** 되는지 확인하는 것이 중요하다. raw\_input()의 경우 사용자가 입력한 값을 문자열(string)으로 변환하여 이를 결과값으로 반환해준다.

실습1) 다음 명령어들을 PracticeRawInput.py 파일로 만들어 실행하여 결과를 확인하세요.

somebody = raw\_input("Enter your name: ")

print "hi!", somebody, "How are you today?"

Enter your name:Mary

hi! Mary How are you today?

*(실행한 후 결과값을 적어보세요)*

위의 실습에서 확인한 바와 같이 raw\_input() 함수는 문자열을 반환해 주고, 이 반환값을 somebody 변수에 저장하여 사용할 수 있다.

위의 실습에서 print 명령어 뒤에 문자열을 복수 개 나열하여 한꺼번에 디스플레이할 수 있음을 확인했다. 콤마(,)로 구분하여 나열하며, 문자열을 따옴표로 싸서 나열하거나 somebody와 같이 문자열을 저장한 변수명을 나열하면 된다.

실습2) 다음 명령어를 PracticePrint.py로 작성하여 print의 추가 기능을 확인해 보세요.

print "I",

print "love",

print "this class~!“

I

love

this class~!

*(실행한 후 결과값을 적어보세요)*

**3.2 파일로 입출력하기**

인터랙티브 모드에서 사용자가 화면에 직접 입출력하는 방식과 달리 파일을 이용해서 입출력을 주고 받을 수 있다.

컴퓨터에서 파일이란 다양한 정보를 저장하고 있는 비트스트림(bitstream)이라고 할 수 있다. 앞에서 말한 바대로 컴퓨터의 메모리나 저장매체의 한 소자는 ON/OFF의 상태를 나타내는 이진값만 표시할 수 있고, 논리적으로 0,1의 값만 가질 수 있다. 고밀도의 소자들에 bit값들이 설정되고, 의미있는 한 덩어리가 만들어지게 되면 그것이 파일로 불리게 된다.

파일은 사용자가 운영체제를 통해 식별하고, 생성하고, 삭제하는 등 관리할 수 있다. 파일은 서로 구별되는 이름을 갖게되고, 이름의 뒷부분에 마침표(.)를 찍고 확장자(extension)을 붙여 파일의 종류를 알려준다.

퀴즈) 다음 확장자를 가진 파일의 종류를 설명하시오.

.jpg 사진, 이미지

.exe 실행파일

.txt 텍스트파일 – 메모장

.pdf PDF 편집안하는 파일

.docx 워드로 만든 문서

.odt 오픈 오피스 – 오픈 다큐먼트 포맷

.mpeg4 동영상

.jar 압축된 파일

.py 파이썬 프로그램파일

파이썬 프로그램에서 웹 상의 특정 위치에 있는 문서의 내용을 읽어 올 수 있다. 읽어 온 내용을 메모리에 저장했다가, 컴퓨터 상의 특정 위치에 파일을 만들고 메모리 내용을 복사하여 저장할 수 있다.

실습3) 다음은 웹의 특정 페이지를 열어 그 내용을 읽어오는 예제입니다. PracticeWebInput.py로 작성하여 실행해보세요.

import urllib2

▲ library

file = urllib2.urlopen(‘https://courses.cs.washington.edu/courses/cse326/02wi/homework/hw5/README-1.txt’)

사이트정보를 넣어서 알려줌

message = file.read()

print message

----------------------

Command Line Arguments

----------------------

Example: ./word-count -b -sort SelectionSort -suf < textfile

-b | -a | -s

(required) Specifies the type of tree for storing (word, count) pair

possible trees are Binary search tree, Avl tree, and Splay tree

\*\*\*생략\*\*\*

For the word-stemming part, two stemming kinds are taken care:

words end with -s and with -ly. Due to our language skill limitations,

we cannot check all the cases and all the exceptions. We have tried our best to check

all of the cases that we think of. Also, since " 'd ", " 's " is in the scope above,

we remove them as well. For this option, one needs to add "-suf" in the command line as

mentioned above.

**file이라는 변수 = urlopen 이라는 함수**

**file. read 읽으라는 함수를 적용**

먼저 urllib2라는 모듈(module)을 import로 가져온 후, urlopen이라는 함수를 호출하여 인수로 준 웹페이지를 열어 file이라는 객체가 가리키도록 한다. 웹페이지의 내용을 한 줄씩 읽어 message 변수에 저장하고 화면에 출력한다.

파이썬에서 모듈이란 함수들의 묶음을 말한다. 모듈의 묶음이 패키지(package)이다. 이들을 프로그램에서 호출하고자할 때는 import라는 명령어를 사용하여 먼저 패키지와 모듈을 불러와야 한다.

이미 완성되고 프로그래머가 손쉽게 사용할 수 있도록 마련된 패키지들을 라이브러리(library)라고 한다. os, glob, sys, stderr, write, re, math, urllib2, smtplib, daetime, data, zlib, timeit, Timer, average 등이 파이썬 매뉴얼에 부록으로 소개되며, 이러한 라이브러리를 잘 알고 활용할 수 있어야 좋은 프로그래머라 할 수 있다.

파일은 컴퓨터의 디렉토리 구조의 특정 위치에 저장된다. 파일의 위치는 디렉토리의 최상위(root)로부터 하위 폴더의 이름을 차례로 적어 주게 되는데 이를 파일의 위치 경로(path)라고 한다.

파일의 위치는 절대경로로 표기하거나 상대경로로 표기한다. 프로그램이 실행되는 순간 현재의 작업디렉토리(working directory)가 정해지게 되며, 그 위치로부터 파일이 놓은 경로를 적어준 것이 상대경로이다.

실습4) 먼저 새로운 파일을 만들어 내용을 저장하는 명령어를 실행해 보세요

my\_file = open('C:\Temp\sample.txt', 'w')

my\_file.write("1. Wake up")

my\_file.write("\n2. Brush teeth")

my\_file.write("\n3. Eat breakfast")

my\_file.close()

다음 명령어를 실행하여 결과를 확인하세요.

>>> my\_file = open('C:\Temp\sample.txt', 'r')

>>> print my\_file.read()

1. Wake up

2. Brush teeth

3. Eat breakfast

이번에는 다음 명령어를 실행하여 결과를 확인하세요

>>> my\_file = open('C:\Temp\sample.txt', 'r')

>>> print my\_file.readlines()

['1. Wake up\n', '2. Brush teeth\n', '3. Eat breakfast']

다음에는 아래와 같이 명령어를 실행하여 결과를 확인하세요.

>>> my\_file = open('C:\Temp\HncDownload\update.log', 'r')

>>> print my\_file.readline()

[Done]

>>> print my\_file.readline()

0404\_141101.1334=End to (Off)

>>> print my\_file.readline()

[LastLog]

>>> print my\_file.readline()

Action=Done

다음 명령어를 실행한다.

>>> my\_file.seek(0)

>>> print my\_file.readlines()

['[Done]\n', '0404\_141101.1334=End to (Off)\n', '[LastLog]\n', 'Action=Done\n', 'Key=0404\_141101.1334\n']

결과의 차이점을 비교해 보면, readline은 파일의 내용을 한 줄씩 읽어오며, readlines는 모든 줄을 읽어 []로 묶어준다. 파일의 첫 부분으로 다시 돌아가려면 seek(0)를 호출한다.

실습5) 파일에 내용을 추가하여 작성하는 사례를 실행해 보세요

print "===================="

my\_file = open('C:\Temp\sample.txt', 'r')

print my\_file.read()

my\_file = open('C:\Temp\sample.txt', 'a')

my\_file.write('\n4. Drink coffee')

my\_file.close()

print "===================="

my\_file = open('C:\Temp\sample.txt', 'r')

print my\_file.read()

1. Wake up

2. Brush teeth

3. Eat breakfast

4. Drink coffee

실습6) print를 이용해 파일에 내용을 추가하는 사례를 실행해 보세요.

my\_file = open('C:\Temp\sample1.txt', 'w')

print >> my\_file, '\n5. Go to school'

my\_file.close()

my\_file = open('C:\Temp\sample1.txt', 'r')

print my\_file.read()

5. Go to school