Python 소개 및 활용

■ About Python

- 귀도 반 로섬(Guido van Rossum)이 1991년에 개발한 대화형 프로그래밍 언어이다
- 최근 많은 인기를 얻고 있다. 가장 큰 이유는 생산성이 뛰어나기 때문이다.
  파이썬을 이용하면 간결하면서도 효율적인 프로그램을 빠르게 작성할 수 있다.
- 오픈소스이어서 무료이고 패키지들이 추가되고 있어서 매일 진화하는 언어이기도 하다.
- 파이썬 다운로드: https://www.python.org/ -> 기타 모듈 설치하기 (pip install ...)
- 영상처리, 빅데이터, 인공지능 패키지 통합 IDE: https://www.anaconda.com/ -> Products -> Individual Edition -> Download -> 해당 Download & Install : Anaconda Navigator, Spyder, Jupyter Notebook 활용

■ Hello World

- print("Hello World")
- print("100"+"200") vs print(100+200)
- 참고) console 명령어: ls, pwd, clear, exit/quit...

■ Comment(주석)

- 기본: # , eg) # This is my first fn
- Triple Quotes: ''', """, eg) '''comments''', 참고) docstring 간주

■ Module 사용하기

- import [모듈명], eg) import numpy
- 그래픽 모듈 활용해 보기

```
import turtle as t
t.shape("turtle")
t.forward(100)
t.done()
```

■ 변수(Varible)

- 식별자는 문자와 숫자, 밑줄 문자(_), 밑줄 문자 이외의 특수 문자를 사용할 수 없다.
- 식별자의 첫 글자는 숫자로 시작할 수 없다. 또한 중간에 공백을 가질 수 없다.
- 대문자와 소문자는 구별된다.
- 대입(assignment): l-variable, r-value, eg) height = 180.0
- 예약어 사용 불가: True, False, None, class, return, is finally, if, else, for, while, continue, break, lambda, def, from, nonlocal, and, del, global, not, with, as, elif, try, except, or, yield, assert, import, pass, in, raise
- dynamic typing: 어떤 자료형도 저장 가능, 참고) static typing
- type(): 변수의 자료형 알기, python 자료형 = class

```
height = 181.5
type(height)
```


동의대학교 김남규(ngkim@deu.ac.kr)

■ 데이터 값(value)

- 값의 표현 한계: 0.1+0.1 == 0.2 vs 0.1+0.1+0.1 == 0.3
- 문자열: ‘, “”, \+’,“ eg) ’철수 “안녕” 하고 말함‘, “철수 ’안녕‘ 하고 말함”
- 형 변환: int(...), str(...), float(...), eg) str(100)+“원”“
- 데이터 입력: [l-variable] = input(“...”) eg) a = int(input(“값?”))

참고)

```
x, y, z = map(int, input(“세 정수”).split())
x, y, z   # (10, 20, 30)
```

■ 연산자(operator)

- 대입/할당(assignment): =
  eg) multiple: x=y=100, simultaneous: n1, n2 = 100, 200
- 수식(expression): operand(피연산자), operator(연산자)
- 연산자: +, −, *, **(지수), //(정수 나눗셈), /(실수 나눗셈), %(나머지)
  eg) 피타고라스 정리/유클리드 거리: (a**2 + b**2) ** 0.5
- 복합 할당(augmented assignment): +=, −=, *=, **=, /=, //=, %=
- 비교(comparison): Bool형(True/False) 반환, ==, !=, >, <, >=, <=
- 논리(logic): and, or, not
- 이진(binary): &(and), |(or), ^(xor), ~(not), <<, >> (shift)
  복합 할당 가능: &=, |=, ^=, <<=, >>=
- 연산자 우선 순위 존재: [**], [unary(~,+,−)], [*,/,%,//], [+,−]...

■ Random & Math module

import random	import math
random.random() random.randint(1,7) # 1 <= r <= 7 random.randrange(7) # 0 <= r < 7 random.randrange(1,7) # 1 <= r < 7 random.randrange(0, 10, 2) # 0, 2, 4, 6, 8	math.pow(3,3) # 3^3 math.fabs(−10) # |−10| = 10 math.log(1.5) math.pi # 3.141592653589793 math.sin(math.pi / 2.0) # 90 math.ceil(3.2) # 4 vs floor()

■ 조건문(conditional statement)

- python의 블록은 들여쓰기(indentation)로 완성
- 조건식은 Bool형인 True / False 값을 가짐

if, if−else	elif

if score > 90 :
  print(“합격입니다”)
  print(“합격증을 받아가세요”)
else :
  print(“불합격입니다”)
  print(“다음 기회에”)
	if num > 0 :
	  print(“양수입니다”)
	elif num == 0 :
	  print(“0입니다”)
	else :
	  print(“음수입니다”)

동의대학교 김남규(ngkim@deu.ac.kr)

■ 반복문(iteration statement)

for	while

sum = 0
for i in range(10): #=range(0,10,1)
  sum += i
print("합", sum)
	sum = count = 0
	while count <10:
	  sum += count
	  count += 1
	print("합", sum)

- 반복문 제어: continue, break
- 참고) 출력 형식 제어하기:
  str = 'I like {1} and {0}'.format('Python', 'Java')
  (결과) I like Java and Python
  val = '{0:10.4f}'.format(3.1415926)'
  (결과) □□□□3.1416
  print('{0:3d} {1:4d} {2:5d}'.format(10, 100, 1000))
  (결과) □10□100□1000

■ 함수(function)

- 전달값(인수, argument), 받는변수(매개변수, parameter), 반환(return) 구조

```
def calculate_area(radius):
    area = 3.14 * radius**2
    return area
```

- multiple arguments, parameters and return values, scope 문법

multiple parameters	variables' scope

def calc(n1, n2):
  return n1+n2, n1-n2, n1*n2
 	
n1, n2 = 100, 200
r1, r2, r3 = calc(n1, n2)
	def prn_cnt():
	  global counter
	  count = 200
	
	count = 100
	prn_cnt()

- default and keyword argument 문법

```
def order(num, pickle=True, onion=True):
    print('햄버거{0}, 피클{1}, 양파{2}'.
          format(num, pickle, onion))
```

```
order(2)
order(1, False, True)
order(3, onion=False,
      pickle=False)
```

- 재귀함수(recursive function)

```
def factorial(n):
    if n <= 1 : return 1
    else : return n * factorial(n-1)
```

- 모듈
  - 함수나 변수, 클래스들을 모아놓은 파일 -> import [모듈이름]
  - 파이썬에서 모듈은 파이썬 파일([...].py) 이고 파일이름이 모듈이름

  예) my_func.py 의 mf_print 함수
  import my_func, import my_func as mf,
  from my_func import mf_print, from my_func import *

동의대학교 김남규(ngkim@deu.ac.kr)

■ 리스트(list)

- **여러 데이터를 관리**: heights = [178.9, 173.5, 166.1]
- **다양한 리스트 형태**: bts = [‘V’, ‘Jungkook’, ‘Jimin’]
- **추가 연산**: bts.append(‘Jin’), bts+=[’RM’, ‘Suga’]
- **생성, 반복, 멤버 확인**: list(range(1,11)), [10, 20, 30]*3, ‘V’ in bts
- **묶음 리스트**: [[‘kim’,178.9], [‘park’, 173.5], [‘Lee’, 166.1]]
- **항목 접근**: letters = [‘A’,‘B’,‘C’,‘D’,‘E’,‘F’], letters[0] #A
- **슬라이싱**(slicing): letters[2:5] #C,D,E, letters[:3] #A,B,C
  letters[::2] #A,C,E, letters[::−1] #F,E,D,C,B,A
- **조작**(manipulation)

```
slist = [‘kim’,178.9], [‘park’, 173.5], [‘Lee’, 166.1]
slist[2] = [‘Baik’, 180.4]
slist.insert(4, “Hong”)
slist.insert(5, 168.1)
```

Method	하는 일
index(x)	원소 x를 이용하여 위치를 찾는 기능
append(x)	원소 x를 리스트의 끝에 추가
count(x)	리스트 내에서 x 원소의 개수를 반환
extend([x1,x2])	[x1,x2] 리스트를 기존 리스트에 삽입
insert(index,x)	원하는 index 위치에 x를 추가
remove(x)	x 원소를 리스트에서 삭제
pop(index)	index 위치의 원소를 삭제한 후 반환, index는 생략 가능(마지막)
sort()	값을 오름차순으로 정렬, reverse=True 내림차순 정렬
reverse()	리스트를 역순으로 생성

- **생성**(creation)**과 참조**(reference)

참조	생성
alist = [‘kim’,‘park’,‘Lee’] blist = alist #id(alist) == id(blist)	alist = [‘kim’,‘park’,‘Lee’] blist = list(alist) #blist = alist[:] #id(alist) != id(blist)

- **탐색**(search): for member in bts
- **함축**(comprehension):
  [x*x for x in range(10)]
  [x for x in range(10) if x % 2 == 0]
  s=[“Hello“,“1234“,“World“,“567“] [x for x in s if x.isdigit()]

■ 튜플(Tuple)

- **불변속성**(immutable) **리스트**: numbers = (1,2,3,4,5)
- **함수 리턴값을 튜플로**

```
import math
def calc(r):
    area=math.pi*r*r
    circum=2*math.pi*r
    return area, circum

radius = 10.0
(a, c) = calc(radius)
print(“넓이=”+str(a)+“둘레=”+str(c))
```

- **참고) 튜플을 이용한 변수 교환**: a, b = b, a

동의대학교 김남규(ngkim@deu.ac.kr)

■ 딕셔너리(dictionary)

- 키(key)와 값(value)의 쌍으로 구성(key-value pair)

```
phonebook = {"홍길동":"010-1234-5678", "강감찬":"010-4567-3456",
             "이순신":"010-4123-9045"}
phonebook['이순신']
phonebook.keys()
phonebook.values()
for key in phonebook.keys():
    print(key, ':', phonebook[key])
```

- 람다 표현식(lamda expression)
  - 람다 함수: 식별자에 의해 정의되지 않은 익명(이름없는 anonymous) 함수
  - def add(x,y): return x+y ↔ lambda x, y: x + y
  - print('합: ', (lambda x,y: x+y)(100,200))
  - sorted(phonebook.items(), key=lambda x:x[0])

■ 집합(set)

- 순서없는 자료형이며, 중복이 허용되지 않음

```
numbers = { 2, 1, 3 }
print(numbers)
set([1,2,3,1,2])
```

```
numbers.add(4)
numbers.remove(4)
1 in numbers # True
```

- 집합연산: 합집합, 교집합, 차집합

```
A = {1,2,3}
B = {3,4,5}
```

```
A | B # 합집합, A.union(B) {1,2,3,4,5}
A & B # 교집합, A.intersecion(B) {3}
A - B # 차집합, A.difference(B) {1,2}
A ^ B # 대칭차집합, A.symmetric_difference(B) {1,2,4,5}
```

- 연사니 len, max, min, sorted, sum

■ 파일(file)

- 컴퓨터 저장 장치 내에 데이터를 저장하기 위한 논리적 단위

```
f = open('Hello.txt', 'w')
f.write("Hello World!")
f.close()
```

```
f = open('Hello.txt','r')
s = f.read()
print(s)
f.close()
```

- 파일에서 중복되지 않는 단어 개수 구하기

```
def process(w):
    output =""
    for ch in w:
        if ch.isalpha() :
            output += ch
    return output.lower()
words = set()  # 중복을 방지하기 위해 집합 자료형에 단어를 넣자
fname = input("입력 파일 이름: ")
file = open(fname, "r")   # 파일을 연다
# 파일의 모든 줄에 대하여 반복한다.
for line in file:
      lineWords = line.split()
      for word in lineWords:
          words.add(process(word))   # 단어를 집합에 추가한다.
print("사용된 단어의 개수 =", len(words))
print(words)
```


동의대학교 김남규(ngkim@deu.ac.kr)

■ 텍스트 데이터 처리

- split

s = 'Welcome to Python' s.split() # ['Welcome', 'to', 'Python']	s = '2021.8.15.' s.split('.') #['2021','8','15']

- join

','.join(['apple','graph','pine']) #apple,graph,pine	'-'.join('010.123.456'.split('.')) # '010-123-456' '010.123.456'.replace('.','-') # '010-123-456'

s = 'hello world'
clist = list(s)
''.join(clist)
	# 공백없애기
	clist = s.split()
	' '.join(clist)

- **대문자와 소문자 변환**: lower(), upper()
- **양쪽, 왼쪽, 오른쪽 공백 제거**: strip(),lstrip(),rstrip() # strip('#')
- **찾기**: s.find() # **찾은** index **반환**
- **문자열 등장 횟수 반환**: s.count('.') # s='www.deu.ac.kr', 3
- **가장 큰문자 및 작은 문자**: max(), min()
- **유니코드 값 얻기**: ord(), **코드에 따른 문자 변환**: chr()

동의대학교 김남규(ngkim@deu.ac.kr)