

데이터분석입문

Lecture 03. 파이썬 프로그래밍 입문

동양미래대학교
인공지능소프트웨어학과
강 환수

- ❖ 01. 출력과 입력 그리고 변수
- ❖ 02. 연산자 사용하기
- ❖ 03. 함수 불러오기
- ❖ 04. 반복과 선택
- ❖ 05. 순서가 있는 저장 공간 리스트

01. 출력과 입력 그리고 변수

02. 연산자 사용하기

03. 함수 불러오기

04. 반복과 선택

05. 순서가 있는 저장 공간 리스트

❖ print() 함수로 출력하기

- 괄호 안의 값을 모니터에 출력해줌

◆ ① 숫자

```
print(10 * 7)
```

◆ ② 문자열 (String)

```
print('Hi, everyone')
```

```
print("Hi, everyone")
```

작은 따옴표 (' ') 또는
큰 따옴표 (" ") 안에
출력하고자 하는
문자열을 작성하면 됩니다

❖ 변수(Variable)

- 하나의 값으로 고정되어 있지 않고 변하는 값

◆ ① 숫자

```
a = 2022  
print(a)
```

◆ ② 문자열 (String)

```
b = '여러분'  
print(b + ' 안녕하세요!')
```

◆ ③ 숫자와 문자열

```
c = '새해'  
print(a, '년에도' + c + ' 복 많이 받으세요!')
```

- vary
(동사) 다르다 (달라지다)
- variable
(형용사) 변동이 심한
(명사) 변수

변수의 이름은 숫자로 시작하면 안 됩니다.
예) 1a, 1004_b 등

❖ input() 함수로 문자열 값 입력 받기

```
name = input('이름을 입력하세요: ')
print(name + '님 안녕하세요!')
```

```
birth = input('출생 연도를 입력하세요: ')
print(2022 - birth, '세이시군요!')
```

TypeError: unsupported operand type(s) for -: 'int' and 'str'

```
birth = input('출생 연도를 입력하세요: ')
print(2022 - int(birth), '세이시군요!')
```

birth의 자료형은 str (문자열)이라는 점을 주의하세요!

2022에서 birth를 빼기 위해서는 birth의 자료형을 int (정수)로 변환해 주어야 합니다!

❖ input() 함수로 문자열 값 입력 받기

```
name = input('이름을 입력하세요: ')
birth = input('출생 연도를 입력하세요: ')
age = 2022 - int(birth)
print('안녕하세요! ' + name + '님! ' + str(age) + '세이시군요!')
```

age의 자료형은 int (정수) 입니다.
따라서 + 연산자를 이용하여 연결하기 위해서는
자료형을 str (문자열)로 변환해 주어야 합니다!

02. 연산자 사용하기

- 01. 출력과 입력 그리고 변수
- 03. 함수 불러오기
- 04. 반복과 선택
- 05. 순서가 있는 저장 공간 리스트

❖ 산술 연산자(Arithmetic Operator)

<code>print(3 + 4)</code>	<code># + 덧셈 연산자</code>
<code>print(3 - 4)</code>	<code># - 뺄셈 연산자</code>
<code>print(3 * 4)</code>	<code># * 곱셈 연산자</code>
<code>print(3 ** 4)</code>	<code># ** 거듭 제곱 연산자</code>
<code>print(3 % 4)</code>	<code># % 나머지 연산자</code>
<code>print(3 / 4)</code>	<code># / 실수 나눗셈 연산자</code>
<code>print(3 // 4)</code>	<code># // 정수 나눗셈 연산자</code>

❖ 비교 연산자(Comparison Operator)

<code>print(3 > 4)</code>	<code># 3은 4보다 크다</code>	<code>→ False (거짓)</code>
<code>print(3 >= 4)</code>	<code># 3은 4보다 크거나 같다</code>	<code>→ False</code>
<code>print(3 < 4)</code>	<code># 3은 4보다 작다</code>	<code>→ True (참)</code>
<code>print(3 <= 4)</code>	<code># 3은 4보다 작거나 같다</code>	<code>→ True</code>
<code>print(3 == 4)</code>	<code># 3은 4와 같다</code>	<code>→ False</code>
<code>print(3 != 4)</code>	<code># 3은 4와 같지 않다</code>	<code>→ True</code>

❖ 논리 연산자(Logical Operator): ① 논리곱(AND)

논리곱 진리표

논리식 (A and B)	결과
True and True	True
True and False	False
False and True	False
False and False	False

```
print((3 > 4) and (2 < 8))
```

False and True → False

```
print(((8 % 2) == 0) and ((3 % 2) != 0))
```

True and True → True

❖ 논리 연산자(Logical Operator): ② 논리합(OR)

논리합 진리표

논리식 (A or B)	결과
True or True	True
True or False	True
False or True	True
False or False	False

```
print((3 > 4) or (2 < 8))
```

```
# False or True → True
```

```
print(((8 % 2) == 0) or ((3 % 2) != 0))
```

```
# True or True → True
```

❖ 논리 연산자(Logical Operator): ③ 논리 부정(NOT)

논리 부정 진리표

논리식 (not A)	결과
not True	False
not False	True

```
print(not (2 < 8))    # not True → False
```

```
print(not (3 > 7))    # not False → True
```

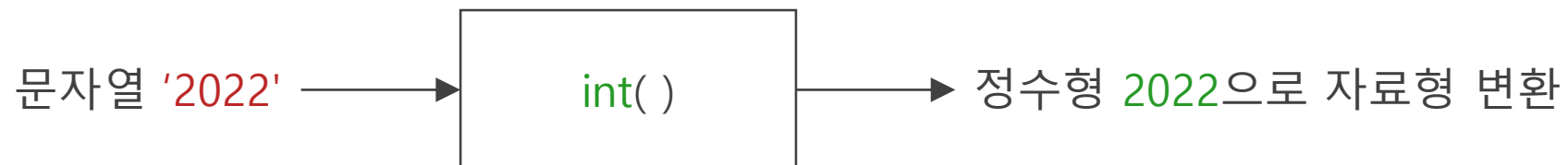
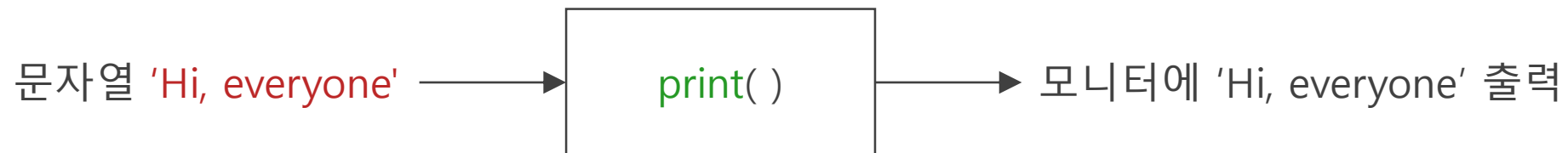
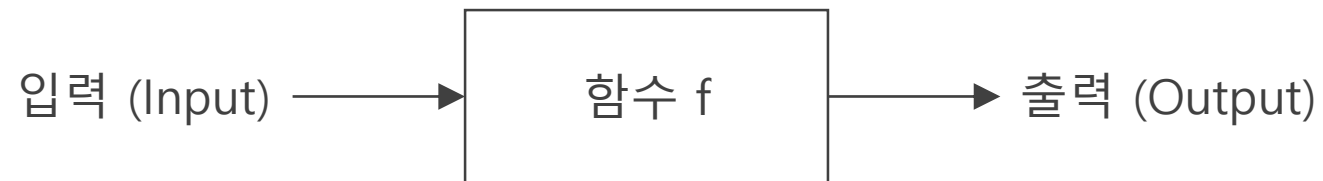
```
print(not 0)          # True (0은 False로 약속합니다)
```

```
print(not 5)          # False (0을 제외한 숫자는 모두 True로 약속합니다)
```

03. 함수 불러오기

- 01. 출력과 입력 그리고 변수
- 02. 연산자 사용하기
- 04. 반복과 선택
- 05. 순서가 있는 저장 공간 리스트

❖ 함수(Function)란?



❖ 파이썬 내장 함수

내장 함수 모두를 외울 필요는 전혀 없습니다!

내장 함수				
abs()	delattr()	hash()	memoryview()	set()
all()	dict()	help()	min()	setattr()
any()	dir()	hex()	next()	slice()
ascii()	divmod()	id()	object()	sorted()
bin()	enumerate()	input()	oct()	staticmethod()
bool()	eval()	int()	open()	str()
breakpoint()	exec()	isinstance()	ord()	sum()
bytearray()	filter()	issubclass()	pow()	super()
bytes()	float()	iter()	print()	tuple()
callable()	format()	len()	property()	type()
chr()	frozenset()	list()	range()	vars()
classmethod()	getattr()	locals()	repr()	zip()
compile()	globals()	map()	reversed()	__import__()
complex()	hasattr()	max()	round()	

❖ 내장 함수 이외의 함수를 불러오기 위해서는?

- import 명령어를 사용하여 사용하고자 하는 함수가 정의되어 있는 라이브러리를 불러옵니다

◆ ① random 라이브러리

```
import random
```

```
dice = random.randint(1, 6)  
print(dice)
```

1과 6사이의 임의의 정수를 반환합니다

- import
(명사) 수입(품)
(동사) 수입하다
(동사) 불러오다

◆ ② math 라이브러리

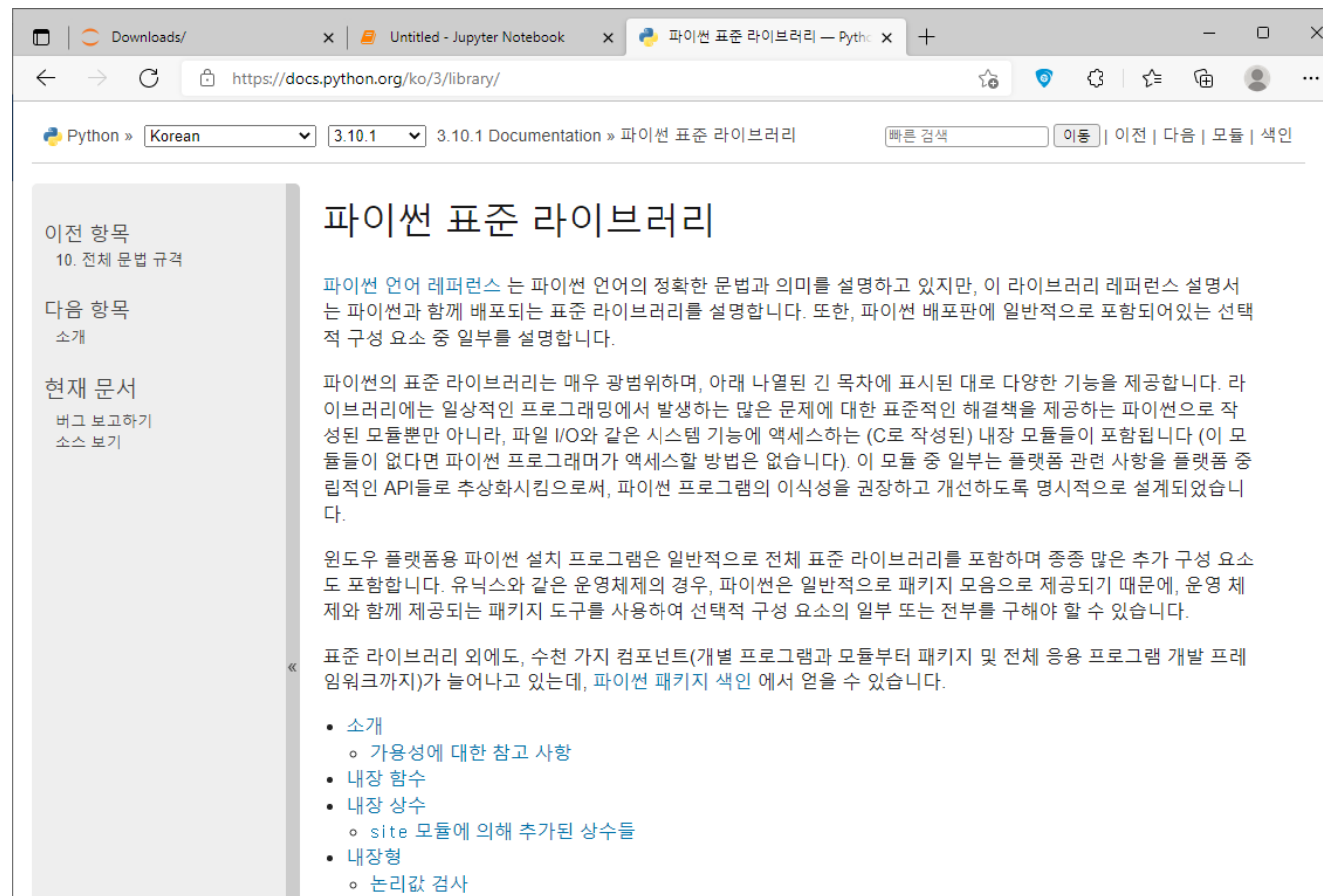
```
import math
```

```
print(math.sqrt(2))  
print(math.sqrt(4))
```

*# 2의 제곱근 (square root)을 반환합니다
4의 제곱근을 반환합니다*

❖ 내가 원하는 기능을 수행하는 함수가 어느 라이브러리에 있는지 어떻게 알 수 있을까?

- 인터넷 검색 → 구글링
- 파이썬 공식 문서에서 찾기 (<https://docs.python.org/ko/3/library/>)



04. 반복과 선택

- 01. 출력과 입력 그리고 변수
- 02. 연산자 사용하기
- 03. 함수 불러오기
- 05. 순서가 있는 저장 공간 리스트

❖ for 반복문 (1/2)

- 주어진 데이터 세트를 순회하거나 명령을 원하는 횟수만큼 반복하고 싶을 때 사용
- for 반복문의 구조

for A **in** B:

반복할 명령1

...

반복할 명령n

맨 뒤에 콜론 (:)을 반드시 잊지 말고 적어주세요

들여쓰기 꼭 해주세요

탭 (tab) 키를 이용하면 쉽게 들여쓰기 할 수 있어요

❖ for 반복문 (2/2)

```
for num in range(10):  
    print(num)
```

*# 0이상 10미만의 정수에 대해서 0부터 순차적으로 변수 num에 대입
range(start, stop, step) → range(0, 10, 1)*

```
test_score = [80, 70, 90, 100]  
for score in test_score:  
    print(score)
```

리스트 test_socre에 저장되어 있는 값을 하나씩 읽어 변수 score에 대입

```
test_name = ['국어', '영어', '수학', '과학']  
for name in test_name:  
    print(name)
```

리스트 test_name에 저장되어 있는 값을 하나씩 읽어 변수 name에 대입

❖ if 조건문 (1/2)

- 주어진 조건이 참(True)인 경우에만 명령을 실행시키고 싶을 때 사용
- if 조건문의 구조

if 조건:

조건이 참일 경우 실행할 명령1

...

조건이 참일 경우 실행할 명령n

맨 뒤에 콜론 (:)을 반드시 잊지 말고 적어주세요

들여쓰기 꼭 해주세요

탭 (tab) 키를 이용하면 쉽게 들여쓰기 할 수 있어요

❖ if 조건문 (2/2)

```
if (10 > 0):  
    print('안녕하세요!')
```

```
password = input('패스워드를 입력하세요: ')
```

```
if (int(password) == 12345):
```

```
    print('로그인 되었습니다.')
```

if 조건문이 참일 경우 실행

```
elif (int(password) == 11111):
```

```
    print('관리자 권한입니다.')
```

if 조건문이 거짓이고 elif 조건문이 참일 경우 실행

```
else:
```

```
    print('로그인에 실패하였습니다.')
```

if, elif 조건문이 모두 거짓일 경우 실행

05. 순서가 있는 저장 공간 리스트

- 01. 출력과 입력 그리고 변수
- 02. 연산자 사용하기
- 03. 함수 불러오기
- 04. 반복과 선택

❖ 순서가 있는 데이터?

- 예) 은행 창구 대기 명단, 출석 번호, 학번 등
- 순서가 있는 데이터를 다룰 때, 리스트(List) 자료형을 사용합니다

```
arr1 = [1, 2, 3, 4, 5]
```

```
print(type(arr1))
```

type(): 변수 arr의 자료형을 반환합니다

```
arr2 = ['월', '화', '수', '목', '금', '토']
```

```
print(type(arr2))
```

```
arr3 = [1, 2, 3, 4, 5, '월', '화', '수', '목', '금', '토']
```

```
print(type(arr3))
```

❖ 인덱스(index)를 이용하여 리스트 값에 접근하기 (1/2)

```
arr2 = ['월', '화', '수', '목', '금', '토']  
print(arr2)
```

리스트 arr2에서 n번째 값만 출력하고 싶다면, 어떻게 해야 할까요?

```
arr2 = ['월', '화', '수', '목', '금', '토']  
print(arr2[0])  
print(arr2[1])  
print(arr2[2])  
print(arr2[3])  
print(arr2[4])  
print(arr2[5])
```

❖ 인덱스(index)를 이용하여 리스트 값에 접근하기 (2/2)

- 음수 인덱스 활용하기
 - ◆ 인덱스 -1: 뒤에서 1번째
 - ◆ 인덱스 -n: 뒤에서 n번째

```
arr2 = ['월', '화', '수', '목', '금', '토']  
print(arr2[-1])           # 뒤에서 1번째: '토'  
print(arr2[-2])           # 뒤에서 2번째: '금'  
print(arr2[-3])           # 뒤에서 3번째: '목'  
print(arr2[-4])           # 뒤에서 4번째: '수'  
print(arr2[-5])           # 뒤에서 5번째: '화'  
print(arr2[-6])           # 뒤에서 6번째: '월'
```

❖ 인덱스(index)를 이용하여 데이터의 일부 자르기(Slicing, 슬라이싱)

```
arr2 = ['월', '화', '수', '목', '금', '토']  
print(arr2[0:2])           # 0이상 2미만의 구간  
print(arr2[1:4])           # 1이상 4미만의 구간  
print(arr2[1:])             # 1이상의 구간  
print(arr2[:3])             # 3미만의 구간  
print(arr2[:])              # 맨 앞에서부터 맨 끝까지
```

❖ 리스트에 값 추가하기

```
arr1 = [1, 2, 3, 4, 5]
print(arr1)
print(len(arr1))           # len(): 리스트의 길이 (Length)를 반환합니다.
arr1.append(6)
print(arr1)
print(len(arr1))
```

```
arr2 = ['월', '화', '수', '목', '금', '토']
print(arr2)
print(len(arr2))
arr2.append('일')
print(arr2)
print(len(arr2))
```

- append
(동사) 덧붙이다
(동사) 첨부하다

- ❖ 01. 출력과 입력 그리고 변수
- ❖ 02. 연산자 사용하기
- ❖ 03. 함수 불러오기
- ❖ 04. 반복과 선택
- ❖ 05. 순서가 있는 저장 공간 리스트

THANK YOU!

Q & A

- Name: 강환수
- Office: 동양미래대학교 2호관 706호 (02-2610-1941)
- E-mail: hsknag@dongyang.ac.kr
- Homepage: <https://github.com/ai7dnn/2023-DA>