단기 집중교육

1일차 실습

Computational Thinking

연세대학교
YONSEI MIRAE CAMPUS

단기 집중 교육

연세대학교 YONSEI MIRAE CAMPUS

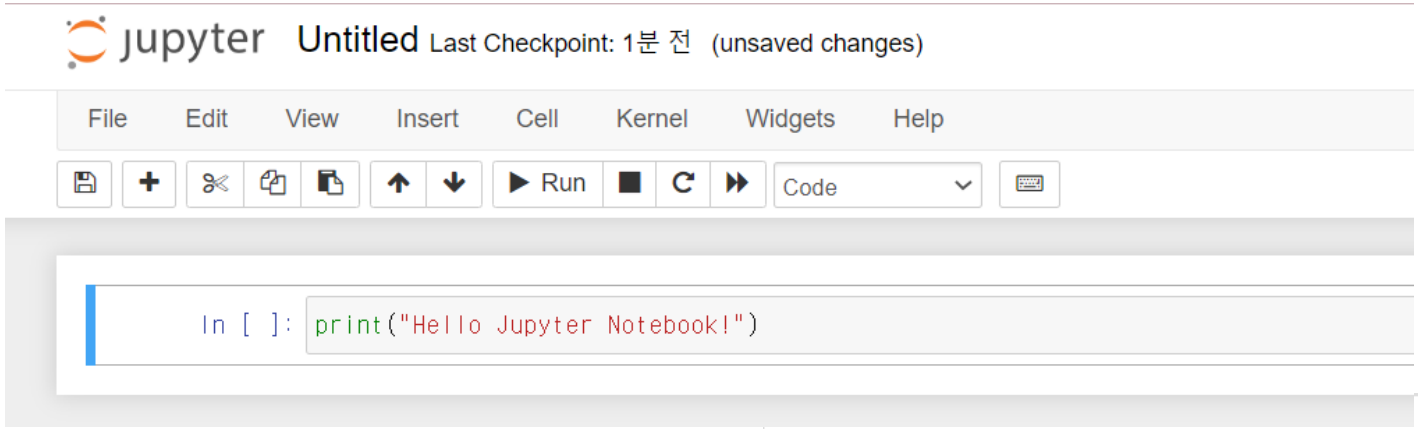
◆ Sw대학 연구조교 유희지

- E-mail: heeji@yonsei.ac.kr

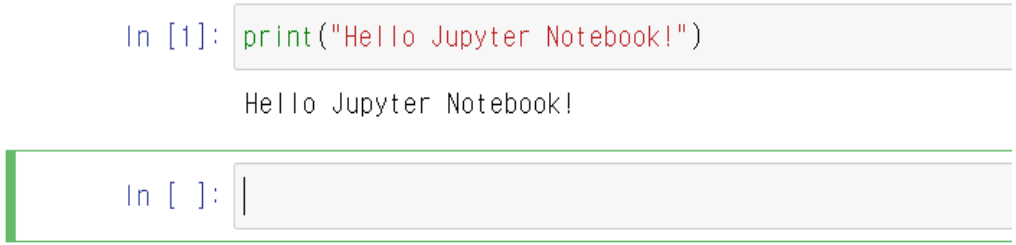
Jupyter Notebook

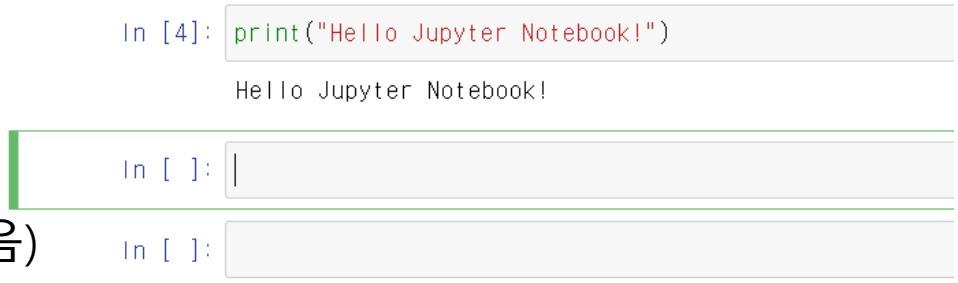
YONSEI MIRAE CAMPUS

Jupyter note book



- 단축키
  - Shift + enter
    : 아래 셀 선택(없으면 추가)
  - Alt+enter
    : 아래 셀 추가
  - Ctrl+enter
    : 현재 셀 수행(셀 추가, 선택 없음)





Jupyter note book

◆ 주피터 노트북에서 문서 작성(markdown)

```
# 1단계 제목 <제목 1번>
## 2단계 제목 <제목 2번 >
### 3단계 제목 <제목 3번 >

**강조** 로 문자를 강조함

* 목록
- 목록1

$$ x^n + y^n = z^n $$
```

1단계 제목 <제목 1번>

2단계 제목 <제목 2번 >

3단계 제목 <제목 3번 >

강조 로 문자를 강조함

- 목록
- 목록1

$$x^n + y^n = z^n$$

< 실행결과화면 >

변수와 자료형

1. 변수
2. 숫자, 문자열, 불(bool)
3. 리스트, 튜플, 세트, 딕셔너리

YONSEI MIRAE CAMPUS

1. 변수

◆ 숫자 할당 예제

- 숫자 12340을 abc 변수에 할당
- abc 변수에 100을 더해서 결과를 출력

```
In [5]: abc = 12340        # 숫자 12340을 abc 변수에 할당
        print(abc + 100)   # abc 변수에 100을 더해서 결과를 출력

        12440
```

◆ 문자 할당 예제

```
In [1]: string1 = "Python is "
        string2 = "powerful."
        print(string1 + string2)

        Python is powerful.
```

2. 숫자, 문자열, 불(bool)

◆ 자료형 확인하는 함수:

type()

◆ **숫자**

- 정수형(int) / 실수형(float)
- 연산순서: 괄호 -> 지수 -> 곱셈, 나눗셈 -> 덧셈, 뺄셈

```
x = 0
```

산술 연산자

연산자	설명	예제
+	더하기	x = 1 + 2 # x = 3
-	빼기	x = 1 - 2 # x = -1
/	나누기	x = 5 / 2 # x = 2.5
//	나눈 몫	x = 5 // 2 # x = 2
%	나눈 나머지	x = 5 % 2 # x = 1
*	곱하기	x = 3 * 3 # x = 9
**	제곱	x = 3 ** 3 # x = 27

```
x = 3
```

대입 연산자

연산자	설명	예제
=	값을 덮어씌움	x = 100 # x = 100
+=	더하기	x += 2 # x = 5
-=	빼기	x -= 2 # x = 1
/=	나누기	x /= 2 # x = 1.5
//=	나눈 몫	x //= 2 # x = 1
%=	나눈 나머지	x %= 2 # x = 1
*=	곱하기	x *= 3 # x = 9
**=	제곱	x **= 3 # x = 27

2. 숫자, 문자열, 불(bool)

◆ 숫자

- 정수형(int) / 실수형(float)
- 연산순서: 괄호 -> 지수 -> 곱셈, 나눗셈 -> 덧셈, 뺄셈

Q. 아래 계산 코드의 결과는 무엇인가?

```
In [ ]: (10/5 + (5-2)) * (1.2+2) / 2**2
```

2. 숫자, 문자열, 불(bool)

◆ 숫자

- 정수형(int) / 실수형(float)
- 연산순서: 괄호 -> 지수 -> 곱셈, 나눗셈 -> 덧셈, 뺄셈

Q. 아래 계산 코드의 결과는 무엇인가?

```
In [ ]: (10/5 + (5-2)) * (1.2+2) / 2**2
```

```
In [2]: (10/5 + (5-2)) * (1.2+2) / 2**2
Out[2]: 4.0
```

2. 숫자, 문자열, 불(bool)

◆ 문자

- " ", ' ' 모두 사용 가능 (단 양쪽에 같은 따옴표를 사용해야 함)

Q. 그는 "파이썬이 무엇입니까?"라고 물었습니다
위 문장을 출력하기 위해서는 어떻게 코드를 작성해야 할까요?

2. 숫자, 문자열, 불(bool)

◆ 문자

- " ", ' ' 모두 사용 가능 (단 양쪽에 같은 따옴표를 사용해야 함)

Q. 그는 "파이썬이 무엇입니까?"라고 물었습니다
위 문장을 출력하기 위해서는 어떻게 코드를 작성해야 할까요?

```
In [3]: print('그는 "파이썬이 무엇입니까?"라고 물었습니다.')
그는 "파이썬이 무엇입니까?"라고 물었습니다.
```

2. 숫자, 문자열, 불(bool)

◆ 문자

- " ", ' ' 모두 사용 가능 (단 양쪽에 같은 따옴표를 사용해야 함)

- 연산자

  \+ -> 문자열끼리 연결

  * -> 문자열 반복해서 연결

- len() 함수 -> 문자열의 문자 개수를 알려줌(공백 포함)

2. 숫자, 문자열, 불(bool)

◆ 불(bool)

- True / False (대소문자 구별)
- 논리연산자:

  and / or / not

and 연산

True and True => True
True and False => False
False and True => False
False and False => False

or 연산

True or True => True
True or False => True
False or True => True
False or False => False

not 연산

not True => False
not False => True

```
a = 10
b = 20
```

비교 연산자

연산자	설명	예제
==	값이 동일하다.	a == b # False
!=	값이 동일하지 않다.	a != b # True
>	왼쪽이 오른쪽보다 크다.	a > b # False
>= (≥)	왼쪽이 오른쪽보다 크거나 같다.	a >= b # False
<	왼쪽이 오른쪽보다 작다.	a < b # True
<= (≤)	왼쪽이 오른쪽보다 작거나 같다.	a <= b # True

2. 숫자, 문자열, 불(bool)

◆ 불(bool)

- True / False (대소문자 구별)
- 논리연산자:

  and / or / not

- 예제) (3 >= 10 or 3 < 7) and 1 > 9 의 결과는?

2. 숫자, 문자열, 불(bool)

◆ 불(bool)

- True / False (대소문자 구별)
- 논리연산자:

  and / or / not

- 예제) (3 >= 10 or 3 < 7) and 1 > 9 의 결과는?

```
In [6]: ( 3 >= 10 or 3 < 7 ) and 1 > 9
Out[6]: False
```

3. 리스트, 튜플, 세트, 딕셔너리

◆ 리스트 [ ]

- 특징: 순서 중요, 요소 변경 가능

- 리스트 연산자: +, *

Q. 아래의 두개의 리스트와 연산자(+, *)를 사용해서 화면에 보이는 대로 출력하시오.

```
list_str1 = ["연세대학교 ", "미래캠퍼스 "]
list_str2 = ["여름 ", "방학 "]
```

결과->

```
['연세대학교 ', '미래캠퍼스 ']
['여름 ', '방학 ']
['여름 ', '방학 ', '여름 ', '방학 ']
['연세대학교 ', '미래캠퍼스 ', '여름 ', '방학 ']
```

3. 리스트, 튜플, 세트, 딕셔너리

◆ 리스트 [ ]

- 리스트 인덱싱

Q. 아래의 질문의 결과를 각각 구하시오.

```
list_mix1 = [1.5, 2.6, '문자열1', '문자열2']
list_mix2 = [4.0, True, 'abc', list_mix1]
```

1. Mix1 리스트 인덱스 3번 출력
2. Mix2 리스트 인덱스 3번의 인덱스 2번 출력
3. Mix2 리스트 인덱스 3번의 인덱스 1번을 제거했을 때 mix2 출력

3. 리스트, 튜플, 세트, 딕셔너리

◆ 리스트 [ ]

- 리스트 인덱싱

Q. 아래의 질문의 결과를 각각 구하시오.

```
list_mix1 = [1.5, 2.6, '문자열1', '문자열2']
list_mix2 = [4.0, True, 'abc', list_mix1]
```

```
In [11]: list_mix1 = [1.5, 2.6, '문자열1', '문자열2']
         list_mix2 = [4.0, True, 'abc', list_mix1]


         print(list_mix1[3])
         print(list_mix2[3][2])
         del list_mix2[3][1]
         print(list_mix2)

         문자열2
         문자열1
         [4.0, True, 'abc', [1.5, '문자열1', '문자열2']]
```

3. 리스트, 튜플, 세트, 딕셔너리

◆ 리스트 [ ]

- 리스트 슬라이싱

Q. 아래의 출력 결과를 각각 구하시오.

In [ ]:
```
list_num3 = [0, 10, 20, 30, 40, 50, 60, 70, 80, 90]

print(list_num3[0:8:2])
print(list_num3[::2])
print(list_num3[8::-2])
del list_num3[0:10:2]
print(list_num3)
```

3. 리스트, 튜플, 세트, 딕셔너리

◆ 리스트 [ ]

- 리스트 슬라이싱

Q. 아래의 출력 결과를 각각 구하시오.

```
In [12]: list_num3 = [0, 10, 20, 30, 40, 50, 60, 70, 80, 90]

         print(list_num3[0:8:2])
         print(list_num3[::2])
         print(list_num3[8::-2])
         del list_num3[0:10:2]
         print(list_num3)
```

```
[0, 20, 40, 60]
[0, 20, 40, 60, 80]
[80, 60, 40, 20, 0]
[10, 30, 50, 70, 90]
```

3. 리스트, 튜플, 세트, 딕셔너리

◆ 튜플 (, , ,) 또는 , , ,

- 특징: 순서 중요, 요소 변경 불가능

Q. 아래의 요소를 가지는 튜플이 있을 때 1, 2번의 결과는?

```
tuple_mixed1 = ('programming', 'language', 'python', 1, 2, 3)
```

1) print(tuple_mixed1[0:4])

2) del tuple_mixed1[3]

3. 리스트, 튜플, 세트, 딕셔너리

◆ 세트(set) { }

- 특징: 요소의 순서가 없음, 중복 요소 없음, 수학적 집합

  -> 인덱싱, 슬라이싱 적용 불가능

  ex) set_num = {10, 100, 1, 2, 3, 4}

  set_num[1] -> 실행결과?

3. 리스트, 튜플, 세트, 딕셔너리

◆ 세트(set) { }

- 특징: 요소의 순서가 없음, 중복 요소 없음 , 수학적 집합

  -> 인덱싱, 슬라이싱 적용 불가능

  ex) set_num = {10, 100, 1, 2, 3, 4}

  set_num[1] -> 실행결과?

```
set_num = {10, 100, 1, 2, 3, 4}
set_num[1]
```

```
---------------------------------------------------------------------
----------
TypeError                                 Traceback (most recent call
last)
Input In [13], in <cell line: 2>()
      1 set_num = {10, 100, 1, 2, 3, 4}
----> 2 set_num[1]

TypeError: 'set' object is not subscriptable
```

3. 리스트, 튜플, 세트, 딕셔너리

◆ 세트(set) { }

- 특징: 요소의 순서가 없음, 중복 요소 없음 , 수학적 집합

Q. 아래의 set을 이용하여 집합 A와 B의 교집합, 합집합, 차집합을 구하는 코드를 작성하시오.

```
set_A = {0, 1, 2, 3, 4}
set_B = {3, 4, 5, 6, 7}
```

3. 리스트, 튜플, 세트, 딕셔너리

◆ 세트(set) { }

- 특징: 요소의 순서가 없음, 중복 요소 없음 , 수학적 집합

Q. 아래의 set을 이용하여 집합 A와 B의 교집합, 합집합, 차집합을 구하는 코드를 작성하시오.

```
set_A = {0, 1, 2, 3, 4}
set_B = {3, 4, 5, 6, 7}
```

```
# &, |, - 연산자 사용
print(set_A & set_B)
print(set_A | set_B)
print(set_A - set_B)

# intersection(), union(), difference() 메서드 사용
print(set_A.intersection(set_B))
print(set_A.union(set_B))
print(set_A.difference(set_B))
```

```
{3, 4}
{0, 1, 2, 3, 4, 5, 6, 7}
{0, 1, 2}
{3, 4}
{0, 1, 2, 3, 4, 5, 6, 7}
{0, 1, 2}
```

3. 리스트, 튜플, 세트, 딕셔너리

◆ 딕셔너리 {key_1:value1, key_2:value2, ,}

- 특징: 키(key)라는 것과 그 키에 해당하는 값(value)을 담을 수 있음
  순서 중요x
- 예제) dict_user = {"이름": "박재민", "나이": 24}
  아래와 같이 출력이 되도록 딕셔너리를 순차적으로 변경하시오.

```
{'이름': '박재민', '나이': 24}
{'이름': '박재민', '나이': 25}
{'이름': '박재민', '나이': 25, '취미': ['게임', '농구']}
```

3. 리스트, 튜플, 세트, 딕셔너리

◆ 딕셔너리 {key_1:value1, key_2:value2, ,}

- 특징: 키(key)라는 것과 그 키에 해당하는 값(value)을 담을 수 있음
  순서 중요x
- 예제) dict_user = {"이름": "박재민", "나이": 24}
  아래와 같이 출력이 되도록 딕셔너리를 순차적으로 변경하시오.

```
dict_user = {"이름": "박재민", "나이": 24}
print(dict_user)

dict_user["나이"] = 25
print(dict_user)

dict_user["취미"] = ["게임", "농구"]
print(dict_user)
```

```
{'이름': '박재민', '나이': 24}
{'이름': '박재민', '나이': 25}
{'이름': '박재민', '나이': 25, '취미': ['게임', '농구']}
```

3. 리스트, 튜플, 세트, 딕셔너리

◆ 딕셔너리 {key_1:value1, key_2:value2, ,}

- 메서드 함수 이용

  -keys() : 키 전체를 리스트로 모아 dict_keys 자료형으로 반환

  -values() : 딕셔너리 값 전체를 리스트로 모아 dict_values 자료형으로 반환

예제) dict_num_alpha = {0: 'a', 1: 'b', 2: 'c', 3: 'd', 4: 'e'}

1) 딕셔너리의 키 전체를 리스트로 출력
2) 딕셔너리의 값 전체를 리스트로 출력
3) key 2번에 대응하는 값 반환
4) 5: 'f' 추가

3. 리스트, 튜플, 세트, 딕셔너리

◆ 딕셔너리 {key_1:value1, key_2:value2, ,}

- 메서드 함수 이용

  -keys() : 키 전체를 리스트로 모아 dict_keys 자료형으로 반환

  -values() : 딕셔너리 값 전체를 리스트로 모아 dict_values 자료형으로 반환

예제)

```
dict_num_alpha = {0: 'a', 1: 'b', 2: 'c', 3: 'd', 4: 'e'}

print(list(dict_num_alpha.keys()))
print(list(dict_num_alpha.values()))
print(dict_num_alpha.get(2))
dict_new = {5: 'f'}
dict_num_alpha.update(dict_new)
print(dict_num_alpha)
```

```
[0, 1, 2, 3, 4]
['a', 'b', 'c', 'd', 'e']
c
{0: 'a', 1: 'b', 2: 'c', 3: 'd', 4: 'e', 5: 'f'}
```

Question and Answer

YONSEI MIRAE CAMPUS