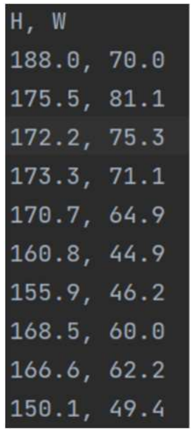
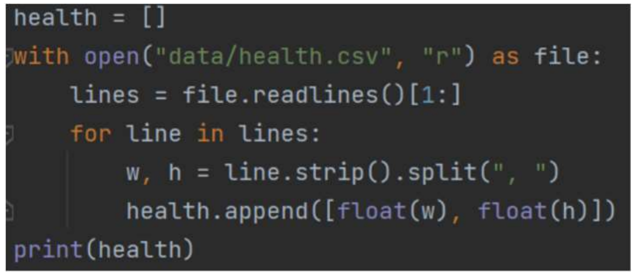
파이썬

파일 입출력

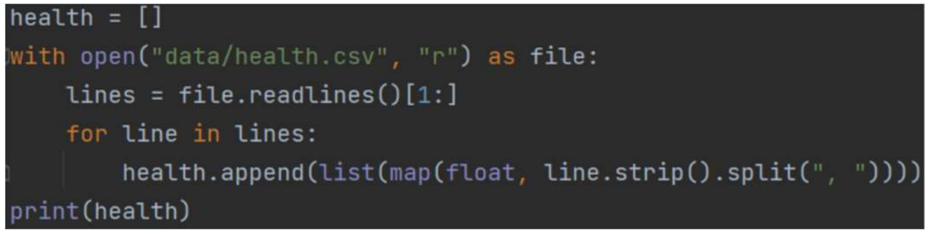


```
H, W
188.0, 70.0
175.5, 81.1
172.2, 75.3
173.3, 71.1
170.7, 64.9
160.8, 44.9
155.9, 46.2
168.5, 60.0
166.6, 62.2
150.1, 49.4
```

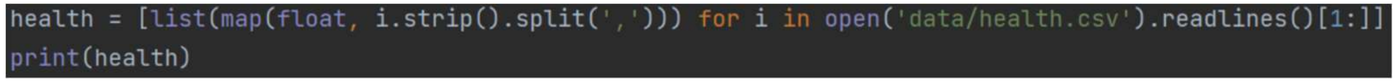
health.csv



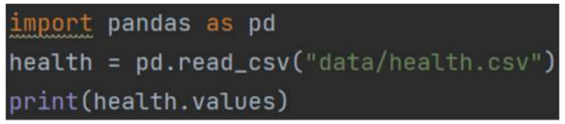
```python
health = []
with open("data/health.csv", "r") as file:
    lines = file.readlines()[1:]
    for line in lines:
        w, h = line.strip().split(", ")
        health.append([float(w), float(h)])
print(health)
```



```python
health = []
with open("data/health.csv", "r") as file:
    lines = file.readlines()[1:]
    for line in lines:
        health.append(list(map(float, line.strip().split(", "))))
print(health)
```

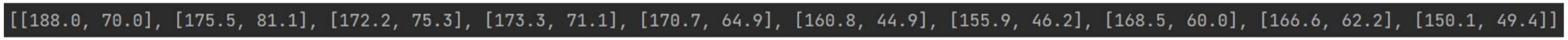


```python
health = [list(map(float, i.strip().split(','))) for i in open('data/health.csv').readlines()[1:]]
print(health)
```



```python
import pandas as pd
health = pd.read_csv("data/health.csv")
print(health.values)
```

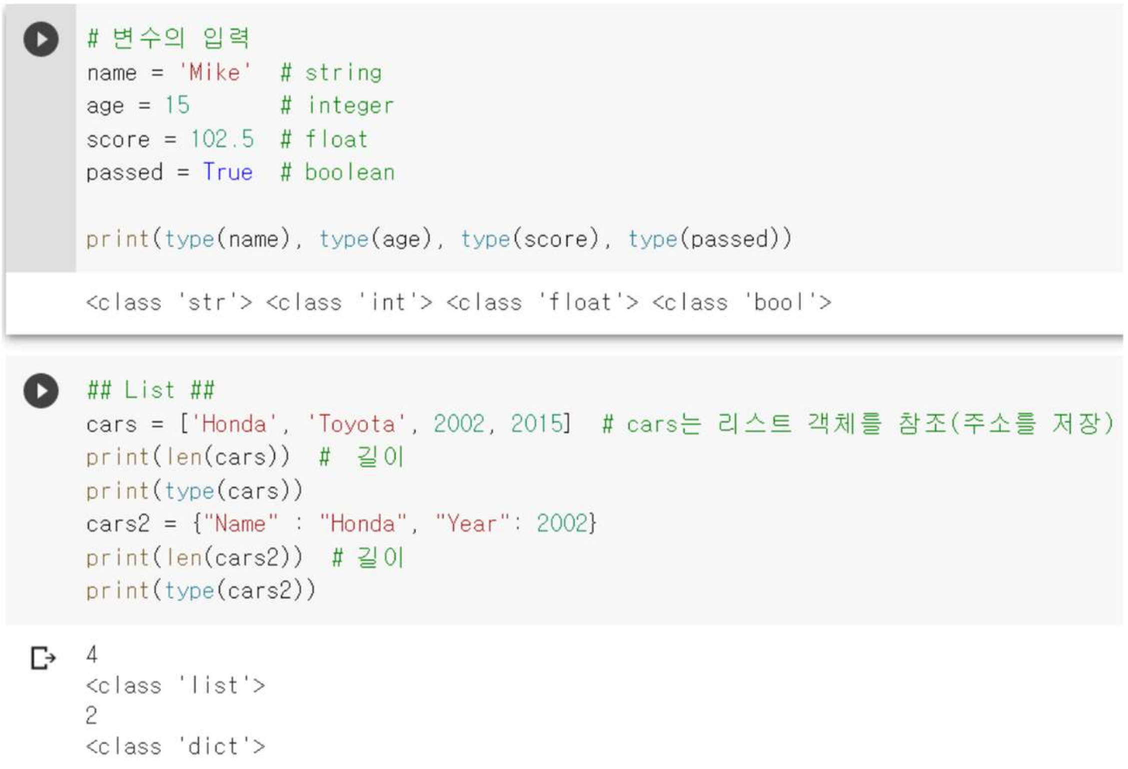
파이썬 코드



```
[[188.0, 70.0], [175.5, 81.1], [172.2, 75.3], [173.3, 71.1], [170.7, 64.9], [160.8, 44.9], [155.9, 46.2], [168.5, 60.0], [166.6, 62.2], [150.1, 49.4]]
```

실행결과

자료형



```
# 변수의 입력
name = 'Mike'  # string
age = 15       # integer
score = 102.5  # float
passed = True  # boolean

print(type(name), type(age), type(score), type(passed))
```

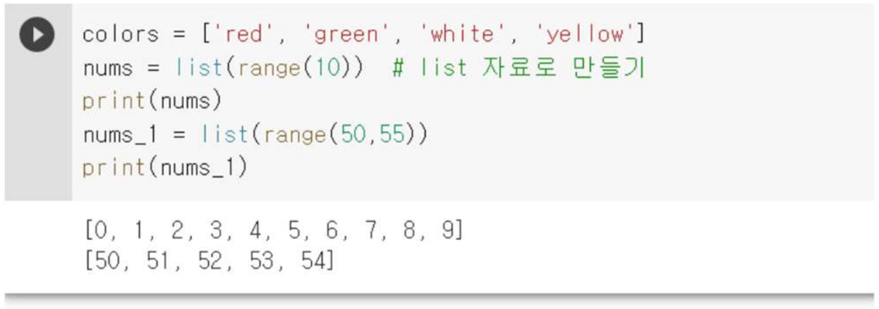
```
<class 'str'> <class 'int'> <class 'float'> <class 'bool'>
```

```
## List ##
cars = ['Honda', 'Toyota', 2002, 2015]  # cars는 리스트 객체를 참조(주소를 저장)
print(len(cars))  #  길이
print(type(cars))
cars2 = {"Name" : "Honda", "Year": 2002}
print(len(cars2))  # 길이
print(type(cars2))
```

```
4
<class 'list'>
2
<class 'dict'>
```

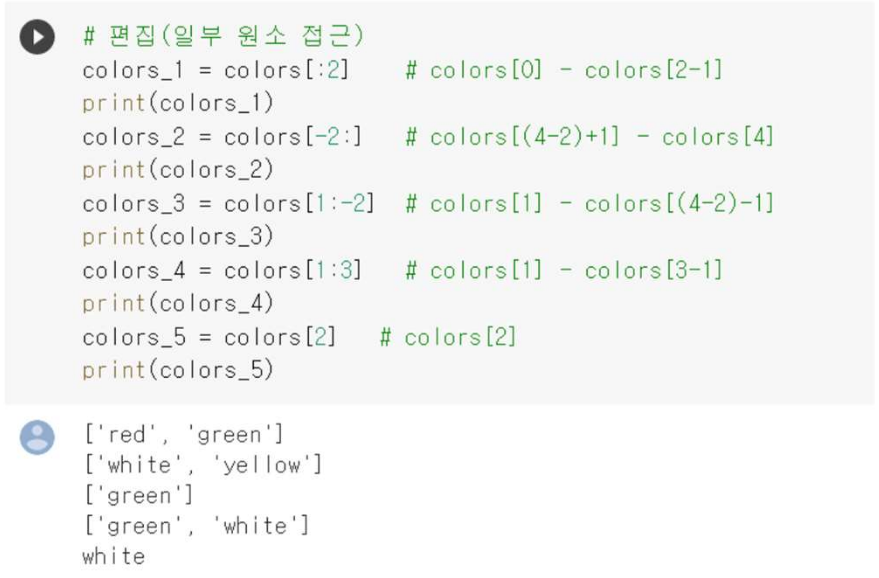
SEJONG UNIVERSITY

리스트 (List)



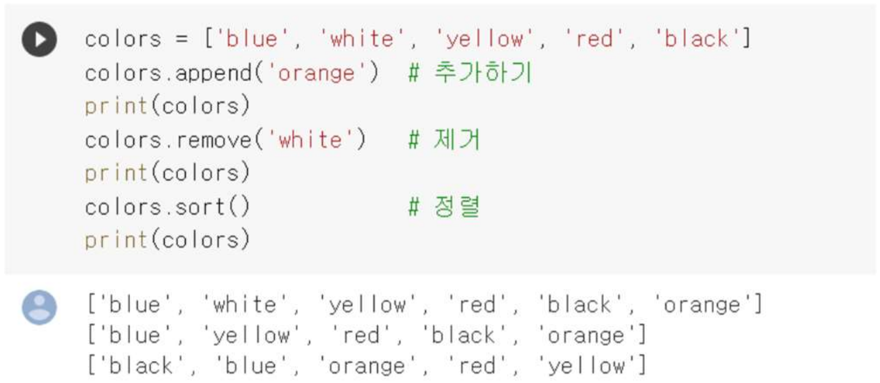
```
colors = ['red', 'green', 'white', 'yellow']
nums = list(range(10))  # list 자료로 만들기
print(nums)
nums_1 = list(range(50,55))
print(nums_1)
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
[50, 51, 52, 53, 54]
```



```
# 편집(일부 원소 접근)
colors_1 = colors[:2]     # colors[0] - colors[2-1]
print(colors_1)
colors_2 = colors[-2:]    # colors[(4-2)+1] - colors[4]
print(colors_2)
colors_3 = colors[1:-2]   # colors[1] - colors[(4-2)-1]
print(colors_3)
colors_4 = colors[1:3]    # colors[1] - colors[3-1]
print(colors_4)
colors_5 = colors[2]    # colors[2]
print(colors_5)
```

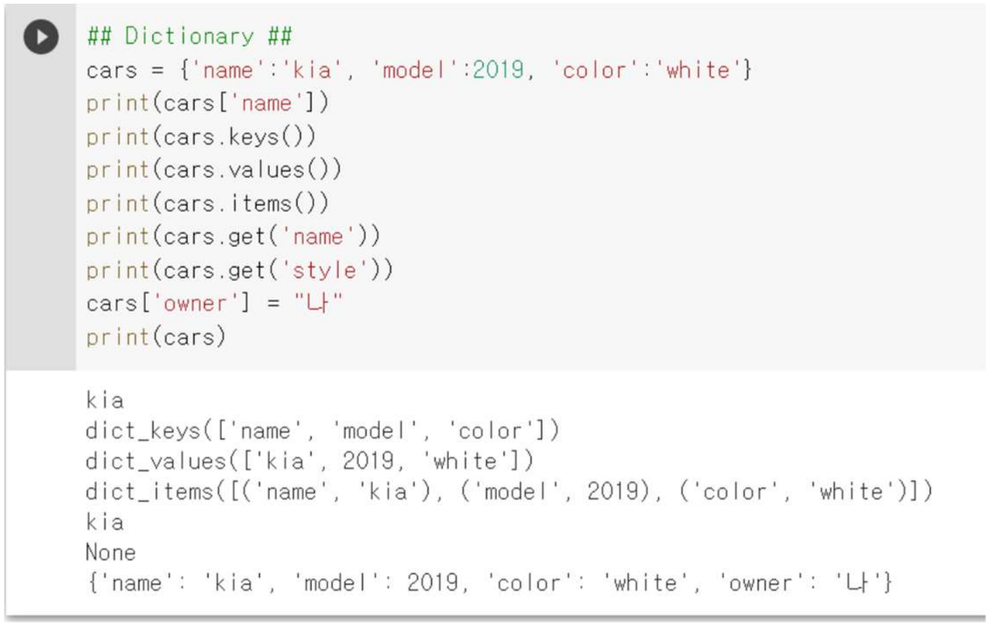
```
['red', 'green']
['white', 'yellow']
['green']
['green', 'white']
white
```



```
colors = ['blue', 'white', 'yellow', 'red', 'black']
colors.append('orange')  # 추가하기
print(colors)
colors.remove('white')   # 제거
print(colors)
colors.sort()            # 정렬
print(colors)
```

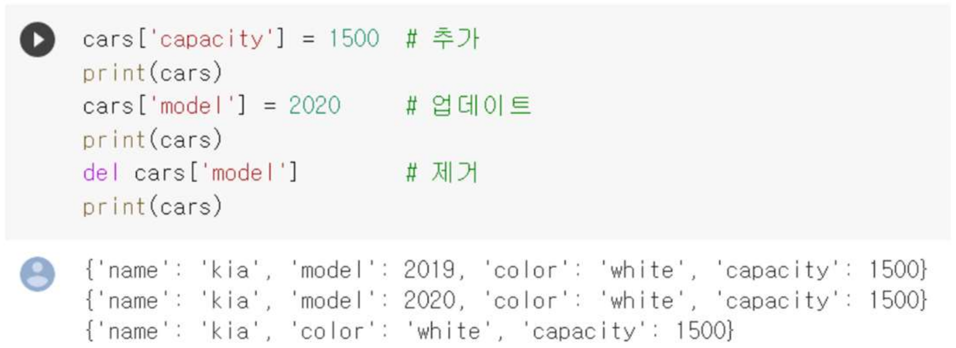
```
['blue', 'white', 'yellow', 'red', 'black', 'orange']
['blue', 'yellow', 'red', 'black', 'orange']
['black', 'blue', 'orange', 'red', 'yellow']
```

딕셔너리 (Dictionary)



```
## Dictionary ##
cars = {'name':'kia', 'model':2019, 'color':'white'}
print(cars['name'])
print(cars.keys())
print(cars.values())
print(cars.items())
print(cars.get('name'))
print(cars.get('style'))
cars['owner'] = "나"
print(cars)
```

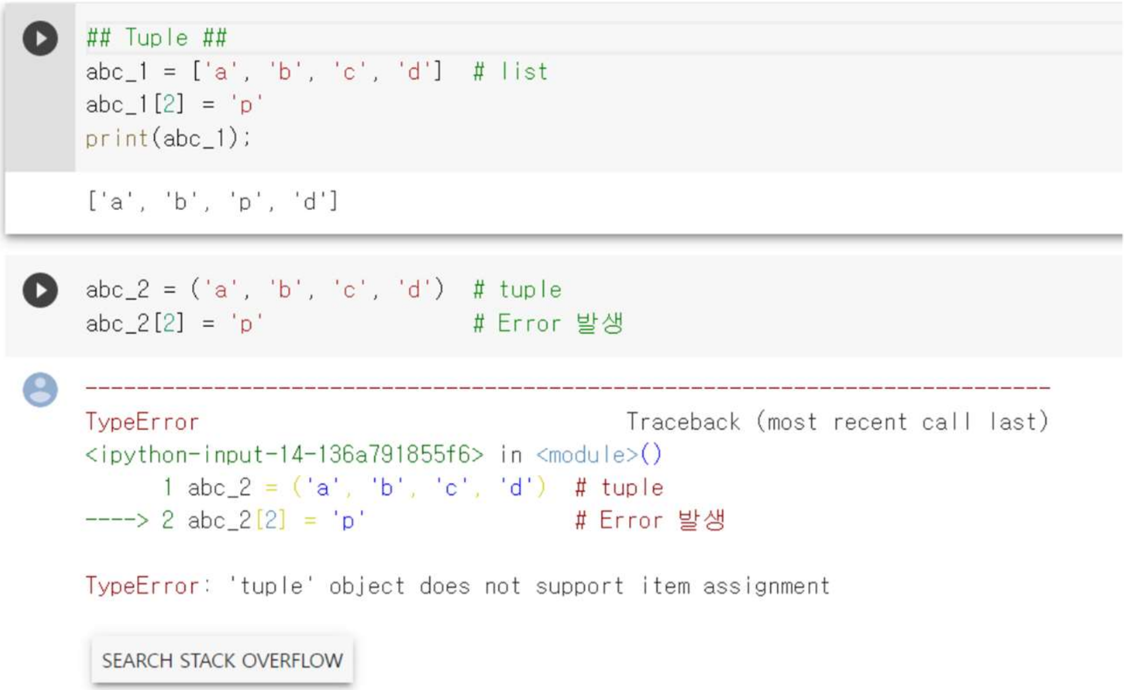
```
kia
dict_keys(['name', 'model', 'color'])
dict_values(['kia', 2019, 'white'])
dict_items([('name', 'kia'), ('model', 2019), ('color', 'white')])
kia
None
{'name': 'kia', 'model': 2019, 'color': 'white', 'owner': '나'}
```



```
cars['capacity'] = 1500   # 추가
print(cars)
cars['model'] = 2020      # 업데이트
print(cars)
del cars['model']         # 제거
print(cars)
```

```
{'name': 'kia', 'model': 2019, 'color': 'white', 'capacity': 1500}
{'name': 'kia', 'model': 2020, 'color': 'white', 'capacity': 1500}
{'name': 'kia', 'color': 'white', 'capacity': 1500}
```

튜플 (Tuple)



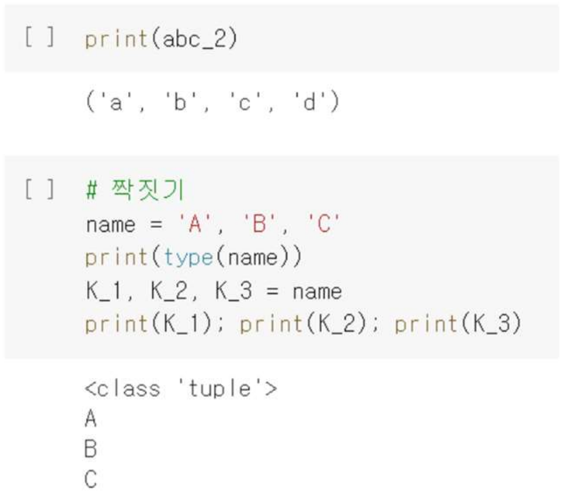
```
## Tuple ##
abc_1 = ['a', 'b', 'c', 'd']  # list
abc_1[2] = 'p'
print(abc_1);

['a', 'b', 'p', 'd']
```

```
abc_2 = ('a', 'b', 'c', 'd')  # tuple
abc_2[2] = 'p'                # Error 발생
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-14-136a791855f6> in <module>()
      1 abc_2 = ('a', 'b', 'c', 'd')  # tuple
----> 2 abc_2[2] = 'p'                # Error 발생

TypeError: 'tuple' object does not support item assignment
```



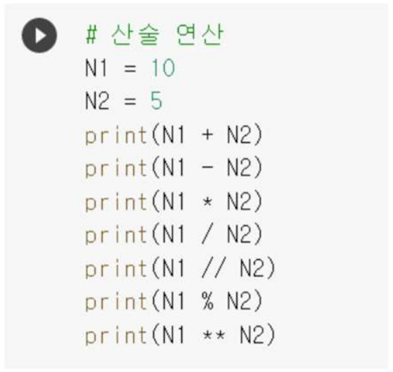
```
[ ]  print(abc_2)

     ('a', 'b', 'c', 'd')
```

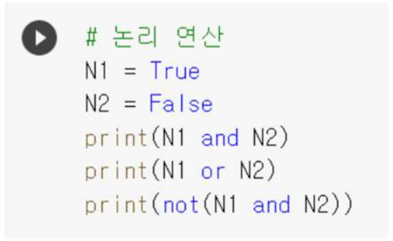
```
[ ]  # 짝짓기
     name = 'A', 'B', 'C'
     print(type(name))
     K_1, K_2, K_3 = name
     print(K_1); print(K_2); print(K_3)

     <class 'tuple'>
     A
     B
     C
```

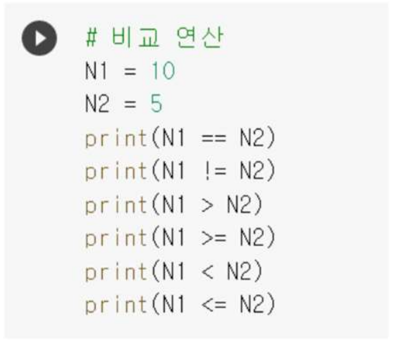
연산



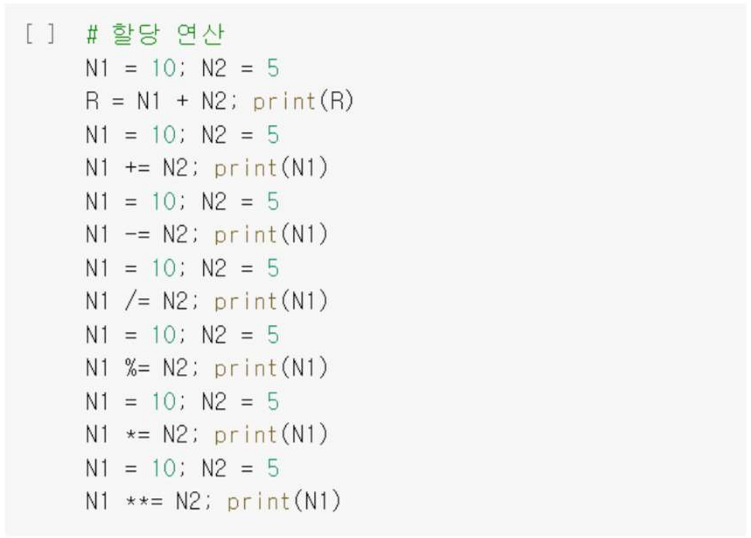
```python
# 산술 연산
N1 = 10
N2 = 5
print(N1 + N2)
print(N1 - N2)
print(N1 * N2)
print(N1 / N2)
print(N1 // N2)
print(N1 % N2)
print(N1 ** N2)
```



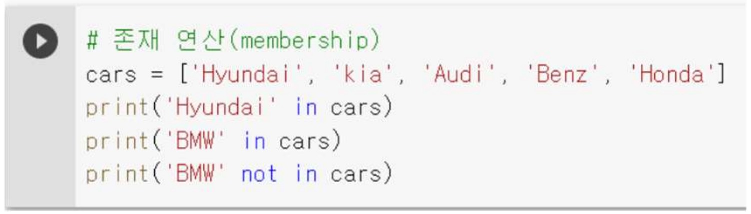
```python
# 논리 연산
N1 = True
N2 = False
print(N1 and N2)
print(N1 or N2)
print(not(N1 and N2))
```



```python
# 비교 연산
N1 = 10
N2 = 5
print(N1 == N2)
print(N1 != N2)
print(N1 > N2)
print(N1 >= N2)
print(N1 < N2)
print(N1 <= N2)
```



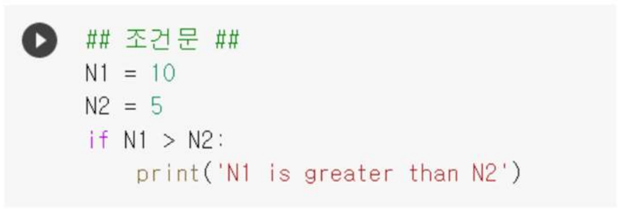
```python
# 할당 연산
N1 = 10; N2 = 5
R = N1 + N2; print(R)
N1 = 10; N2 = 5
N1 += N2; print(N1)
N1 = 10; N2 = 5
N1 -= N2; print(N1)
N1 = 10; N2 = 5
N1 /= N2; print(N1)
N1 = 10; N2 = 5
N1 %= N2; print(N1)
N1 = 10; N2 = 5
N1 *= N2; print(N1)
N1 = 10; N2 = 5
N1 **= N2; print(N1)
```



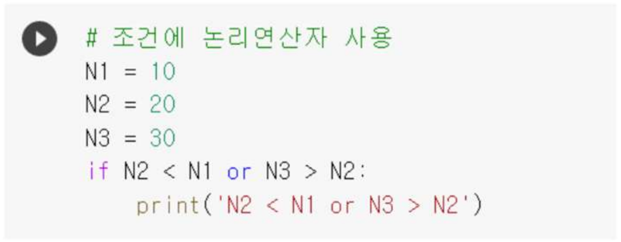
```python
# 존재 연산(membership)
cars = ['Hyundai', 'kia', 'Audi', 'Benz', 'Honda']
print('Hyundai' in cars)
print('BMW' in cars)
print('BMW' not in cars)
```

SEJONG UNIVERSITY

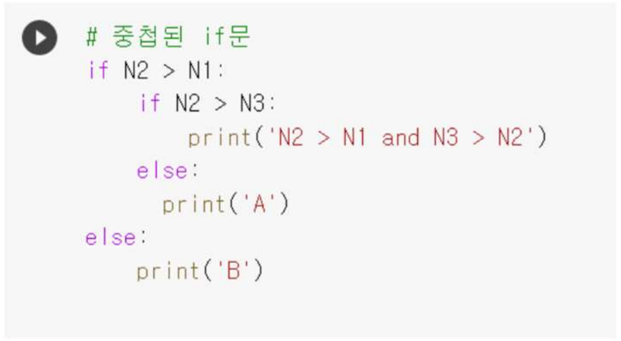
조건문



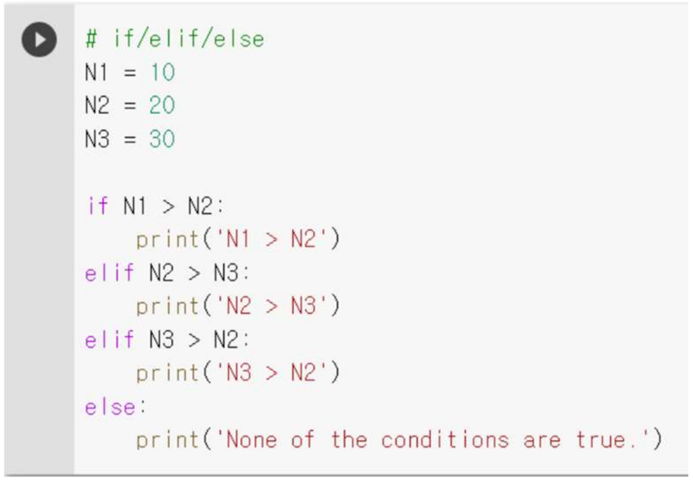
```python
## 조건문 ##
N1 = 10
N2 = 5
if N1 > N2:
    print('N1 is greater than N2')
```



```python
# 조건에 논리연산자 사용
N1 = 10
N2 = 20
N3 = 30
if N2 < N1 or N3 > N2:
    print('N2 < N1 or N3 > N2')
```



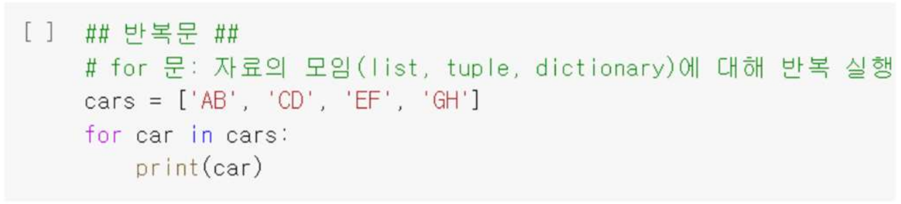
```python
# 중첩된 if문
if N2 > N1:
    if N2 > N3:
        print('N2 > N1 and N3 > N2')
    else:
      print('A')
else:
    print('B')
```



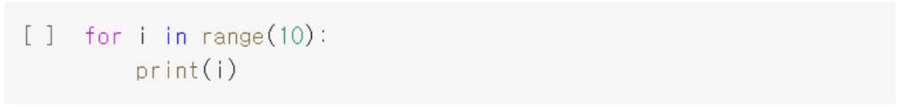
```python
# if/elif/else
N1 = 10
N2 = 20
N3 = 30

if N1 > N2:
    print('N1 > N2')
elif N2 > N3:
    print('N2 > N3')
elif N3 > N2:
    print('N3 > N2')
else:
    print('None of the conditions are true.')
```

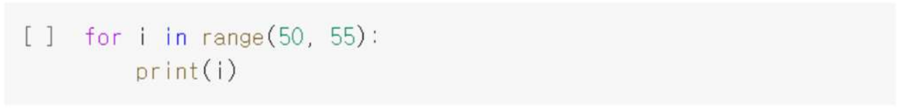
반복문



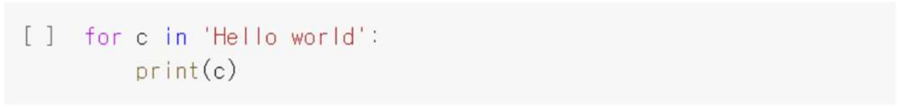
```
[ ]  ## 반복문 ##
     # for 문: 자료의 모임(list, tuple, dictionary)에 대해 반복 실행
     cars = ['AB', 'CD', 'EF', 'GH']
     for car in cars:
         print(car)
```



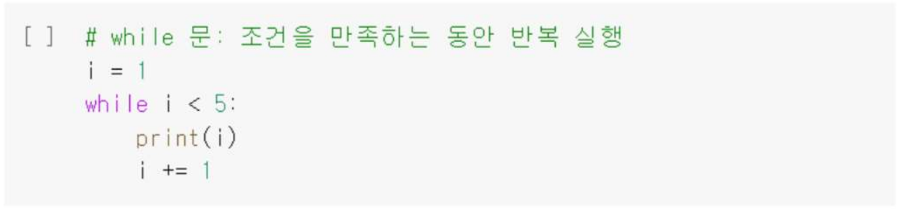
```
[ ]  for i in range(10):
         print(i)
```



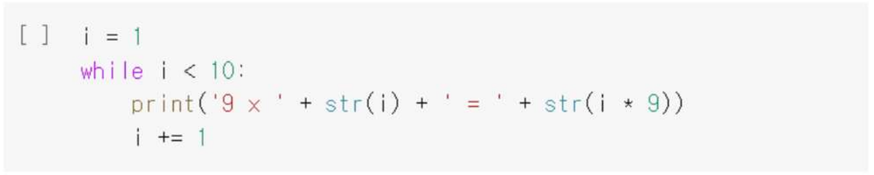
```
[ ]  for i in range(50, 55):
         print(i)
```



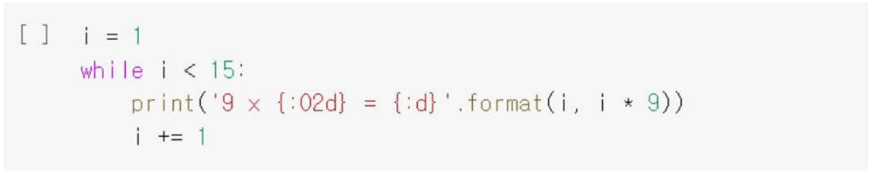
```
[ ]  for c in 'Hello world':
         print(c)
```



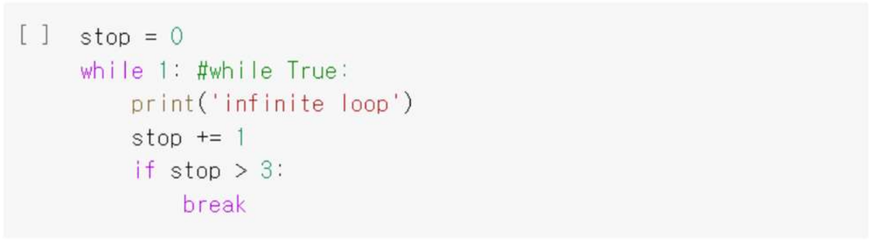
```
[ ]  # while 문: 조건을 만족하는 동안 반복 실행
     i = 1
     while i < 5:
         print(i)
         i += 1
```



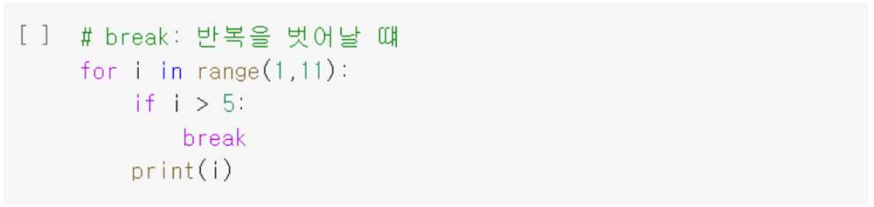
```
[ ]  i = 1
     while i < 10:
         print('9 x ' + str(i) + ' = ' + str(i * 9))
         i += 1
```



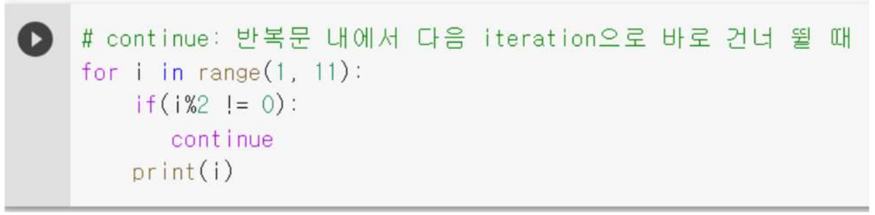
```
[ ]  i = 1
     while i < 15:
         print('9 x {:02d} = {:d}'.format(i, i * 9))
         i += 1
```



```
[ ]  stop = 0
     while 1: #while True:
         print('infinite loop')
         stop += 1
         if stop > 3:
             break
```



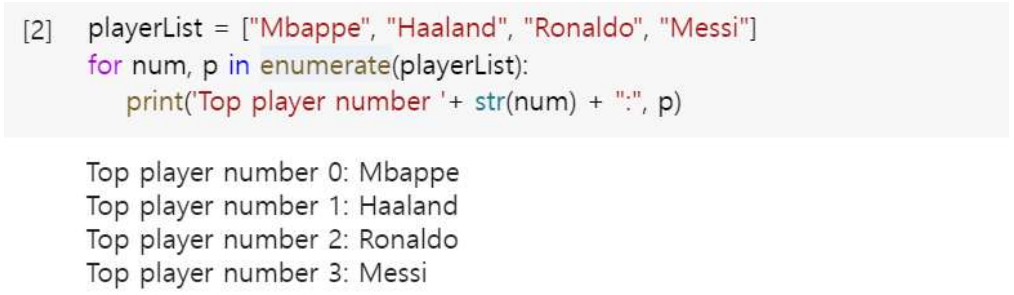
```
[ ]  # break: 반복을 벗어날 때
     for i in range(1,11):
         if i > 5:
             break
         print(i)
```



```
# continue: 반복문 내에서 다음 iteration으로 바로 건너 뛸 때
for i in range(1, 11):
    if(i%2 != 0):
       continue
    print(i)
```

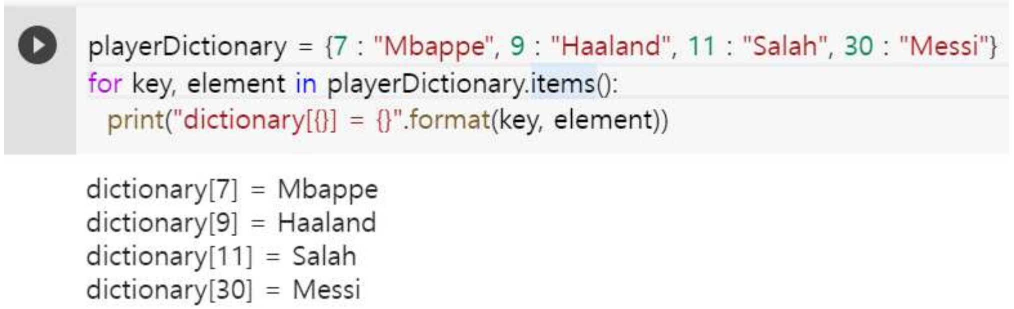
SEJONG UNIVERSITY

리스트/딕셔너리+반복문



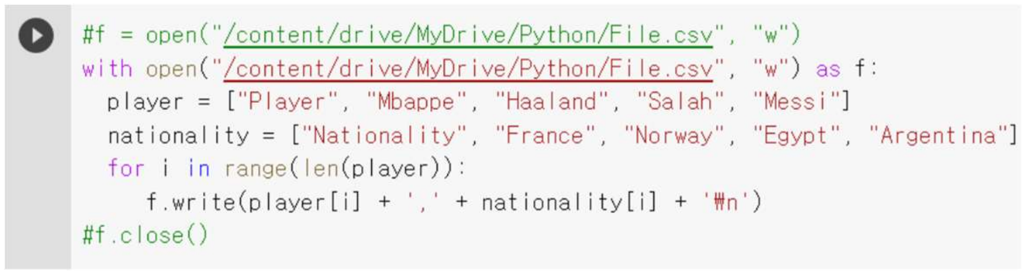
```
[2] playerList = ["Mbappe", "Haaland", "Ronaldo", "Messi"]
    for num, p in enumerate(playerList):
        print('Top player number '+ str(num) + ":", p)
```

```
Top player number 0: Mbappe
Top player number 1: Haaland
Top player number 2: Ronaldo
Top player number 3: Messi
```

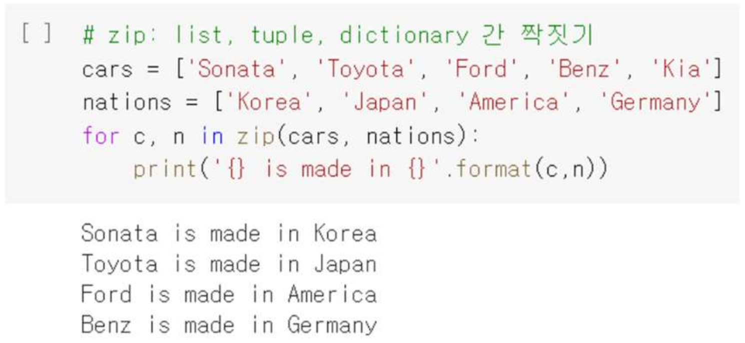


```
playerDictionary = {7 : "Mbappe", 9 : "Haaland", 11 : "Salah", 30 : "Messi"}
for key, element in playerDictionary.items():
  print("dictionary[{}] = {}".format(key, element))
```

```
dictionary[7] = Mbappe
dictionary[9] = Haaland
dictionary[11] = Salah
dictionary[30] = Messi
```

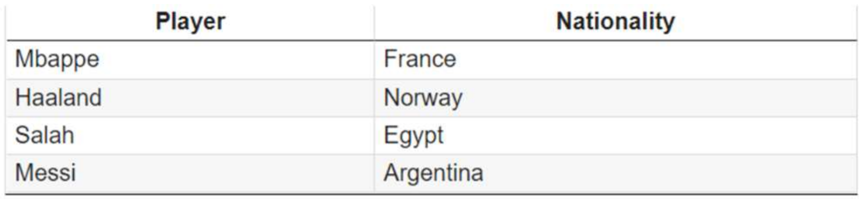


```
#f = open("/content/drive/MyDrive/Python/File.csv", "w")
with open("/content/drive/MyDrive/Python/File.csv", "w") as f:
  player = ["Player", "Mbappe", "Haaland", "Salah", "Messi"]
  nationality = ["Nationality", "France", "Norway", "Egypt", "Argentina"]
  for i in range(len(player)):
      f.write(player[i] + ',' + nationality[i] + '₩n')
#f.close()
```



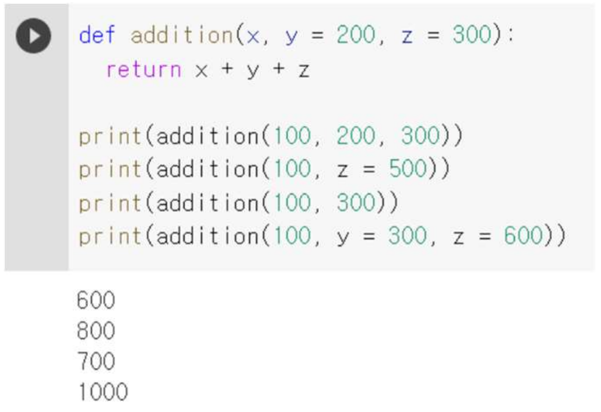
```
[ ] # zip: list, tuple, dictionary 간 짝짓기
    cars = ['Sonata', 'Toyota', 'Ford', 'Benz', 'Kia']
    nations = ['Korea', 'Japan', 'America', 'Germany']
    for c, n in zip(cars, nations):
        print('{} is made in {}'.format(c,n))
```

```
Sonata is made in Korea
Toyota is made in Japan
Ford is made in America
Benz is made in Germany
```



Player	Nationality
Mbappe	France
Haaland	Norway
Salah	Egypt
Messi	Argentina

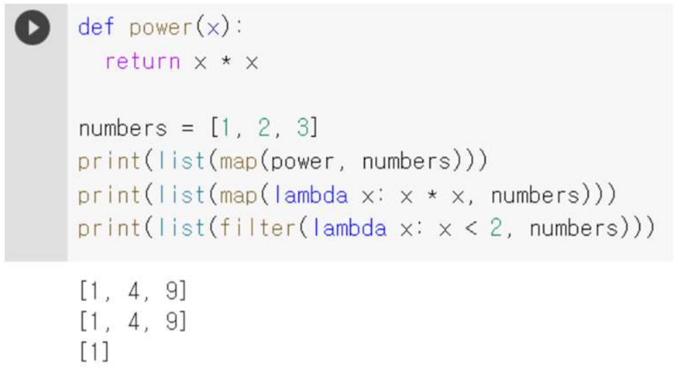
함수



```python
def addition(x, y = 200, z = 300):
  return x + y + z

print(addition(100, 200, 300))
print(addition(100, z = 500))
print(addition(100, 300))
print(addition(100, y = 300, z = 600))
```

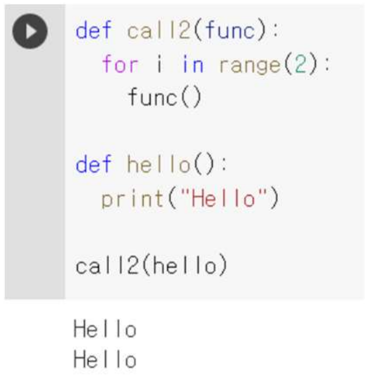
```
600
800
700
1000
```



```python
def power(x):
  return x * x

numbers = [1, 2, 3]
print(list(map(power, numbers)))
print(list(map(lambda x: x * x, numbers)))
print(list(filter(lambda x: x < 2, numbers)))
```

```
[1, 4, 9]
[1, 4, 9]
[1]
```



```python
def call2(func):
  for i in range(2):
    func()

def hello():
  print("Hello")

call2(hello)
```

```
Hello
Hello
```

SEJONG UNIVERSITY

참고자료

- 지능기전공학부 최유경 교수님 자료, https://github.com/sejongresearch/2021.MachineLearning
- 코랩(Colab), https://colab.research.google.com/
- 파이썬(Python), https://www.python.org/doc/
- 사이킷런(sckit-learn), https://scikit-learn.org/stable/index.html
- 판다스(pandas), https://pandas.pydata.org/
- 맷플롯립(matplotlib), https://matplotlib.org/
- 씨본(seaborn), https://seaborn.pydata.org/
- 캐글(Kaggle), https://www.kaggle.com/
- 넘파이(numpy), https://numpy.org/doc/stable/
- 스택오퍼플러우(stackoverflow), https://stackoverflow.com/