

# Ch 7-A: Python Code Reading Recitation-A

파란색 테두리 안의 공간이 부족하여 다소 중구난방하게 적어놓은 점  
죄송합니다. 입출력 예시는 빨간색으로, 설명은 파란색으로 적었습니다.  
참고해주시면 감사하겠습니다.

# Code using List [1/7]

```
bicycles = ['trek', 'cannondale', 'redline', 'specialized']
message = "My first bicycle was a " + bicycles[0].title() + "."
print(message)
```



출력: My first bicycle was a Trek.

설명: bicycles[0] = 'trek'이며 title()은 단어마다 맨 앞은 대문자, 나머지는 소문자로 바꾸어주므로 'Trek'이 된다. +로 문자열들을 합치면 출력 결과가 나온다.

```
motorcycles = ['honda', 'yamaha', 'suzuki', 'ducati']
print(motorcycles)
```

```
too_expensive = 'ducati'
```

```
motorcycles.remove(too_expensive)
```

```
print(motorcycles)
```

```
print("\nA " + too_expensive.title() + " is too expensive for me.")
```



출력:

['honda', 'yamaha', 'suzuki', 'ducati']  
['honda', 'yamaha', 'suzuki']

설명 : remove()로 리스트에서 'ducati'를 찾아 제거한다. 또한 title()을 사용하여 'ducati'->'Ducati'

A Ducati is too expensive for me.

# Code using List [2/7]

```
cars = ['bmw', 'audi', 'toyota', 'subaru']
print("Here is the original list:")
print(cars)

print("\nHere is the sorted list:")
print(sorted(cars))

print("\nHere is the reverse alphabetical list:")
print(sorted(cars,reverse=True))

print("\nHere is the original list again:")
print(cars)
```

출력:

Here is the original list:  
['bmw', 'audi', 'toyota', 'subaru']

Here is the sorted list:  
['audi', 'bmw', 'subaru', 'toyota']

Here is the reverse alphabetical list:  
['toyota', 'subaru', 'bmw', 'audi']

Here is the original list again:  
['bmw', 'audi', 'toyota', 'subaru']

설명 : sorted(cars)는 사전순으로 정렬된 차 이름의 리스트를 반환한다. 또한 reverse=True로 놓으면 역순으로 정렬된다. sorted(cars)는 cars 자체를 변화시키지 않기 때문에 print(cars)는 처음 설정해 놓은 그 리스트(cars)를 출력하게 된다.



# Code using List [3/7]

```
magicians = ['alice', 'david', 'carolina']
```

```
for magician in magicians:
```

```
    print(magician.title() + ", that was a great trick!")
```

```
    print("I can't wait to see your next trick, " + magician.title() + ".\n")
```

```
print("Thank you everyone, that was a great magic show!")
```

출력:

Alice, that was a great trick!

I can't wait to see your next trick, Alice.

David, that was a great trick!

I can't wait to see your next trick, David.

Carolina, that was a great trick!

I can't wait to see your next trick, Carolina.

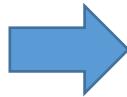
Thank you everyone, that was a great magic show!



설명 : magicians라는 리스트 안에 magician들의 이름이 있다. title()은 이름의 첫글자를 대문자로 바꾸어 준다. 반복문을 사용하여 좋은 마술이었다는 칭찬을 각각의 마술사에게 해준 후, 마지막 문자열은 한번만 출력하는 코드이다.

# Code using List [4/7]

```
numbers = list(range(1,6))  
print(numbers)
```



출력: [1, 2, 3, 4, 5]

설명 : list(range(1,6))은 1에서 5까지의 수를 차례대로 담은 리스트를 의미한다. 따라서 다음과 같은 결과가 출력된다.

```
even_numbers = list(range(2,11,2))  
print(even_numbers)
```



출력: [2, 4, 6, 8, 10]

설명 : list(range(2,11,2))는 2에서 10까지 간격 2로 주어진 수열을 담은 리스트를 의미하며, 이는 곧 2에서 10까지의 모든 짝수의 리스트이다.

# Code using List [5/7]

```
squares = []
for value in range(1,11):
    square = value **2
    squares.append(square)

print(squares)
```



출력: [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]

설명 : range(1,11)은 1에서 10까지의 자연수, 따라서 squares는 1~10까지의 자연수의 제곱수를 넣은 리스트이며, 출력하면 위 결과가 나온다.



```
players = ['charles', 'martina', 'michael', 'florence', 'eli']

print("Here are the first three players on my team:")
for player in players[:3]:
    print(player.title())
```



출력:

Here are the first three players on my team:  
Charles  
Martina  
Michael

설명 : players[:3]는 처음부터 2번 인덱스까지 players를 슬라이싱 한 리스트이다. 따라서 이 코드는 출력 첫 문장의 뜻대로 players 리스트의 처음 세 명의 이름을 title()을 이용해 적절히 바꾼 후 출력해준다.

# Code using List [6/7]

```
my_foods = ['pizza', 'falafel', 'carrot cake']
friend_foods = my_foods[:]
```

```
my_foods.append('cannoli')
friend_foods.append('ice cream')
```

```
print("My favorite foods are:")
print(my_foods)
```

```
print("\nMy friend's favorite foods are:")
print(friend_foods)
```



출력:

My favorite foods are:

['pizza', 'falafel', 'carrot cake', 'cannoli']

My friend's favorite foods are:

['pizza', 'falafel', 'carrot cake', 'ice\_cream']

설명 : friend\_foods = my\_foods[:]를 해줌으로써 my\_foods의 physical copy를 만든다. 이후 my\_foods와 friend\_foods의 원소는 서로의 변화에 무관하다. 따라서 my\_foods에 'cannoli'를 추가하고, friend\_foods에 'ice cream'을 추가하면 두 리스트는 3개의 같은 원소에 1개의 다른 원소를 가진 리스트가 된다. 이를 개행문자로 구별해 출력해줌으로써 각자가 선호하는 음식을 알려주는 코드이다.

# Code using List

[7/7]

```
dimensions = (200,50)
print("Original dimensions:")
for dimension in dimensions:
    print(dimension)

dimensions = (400,100)
print("\nModified dimensions:")
for dimension in dimensions:
    print(dimension)
```

출력:  
Original dimensions:

200  
50

Modified dimensions:

400  
100



설명 : dimensions는 튜플이다. 따라서 튜플 내의 원소의 변화는 불가능하지만, dimensions=(400,100)처럼 튜플 전체를 다시 선언하는 것은 가능하다. 따라서 처음 반복문에서는 (200,50)의 원소를 개행문자로 구별해 각각 출력해주고, 나중 반복문에서는 (400,100)의 원소를 출력해준다.

# Code with IF Statement

[1/5]

```
cars = ['audi', 'bmw', 'subaru', 'toyota']
```

```
for car in cars:  
    if car == 'bmw':  
        print(car.upper())  
    else:  
        print(car.title())
```

출력:

Audi  
BMW  
Subaru  
Toyota



설명 : 반복문을 통해 cars의 원소들을 차례대로 출력하면서, 'bmw'를 제외한 문자열은 title()을 사용해 첫 글자를 대문자로 만들고, 'bmw'의 경우에는 upper()를 사용해 모든 글자를 대문자로 바꾸어주었다.

# Code with IF Statement

[2/5]

```
available_toppings = ['mushrooms', 'olives', 'green peppers',
                      'pepperoni', 'pineapple', 'extra cheese']
```

```
requested_toppings = ['mushrooms', 'french fries', 'extra cheese']
```

```
for requested_topping in requested_toppings:
    if requested_topping in available_toppings:
        print("Adding " + requested_topping + ".")
    else:
        print("Sorry, we don't have " + requested_topping + ".")
print("\nFinished making your pizza!")
```

출력:

Adding mushrooms.  
Sorry, we don't have french fries.  
Adding extra cheese.

Finished making your pizza!

설명 : requested된 토핑을 담은 리스트에 대해 반복문을 돌리면서 if문을 사용하여, 만약 리스트 안의 원소가 available\_toppings에 있으면 더해준다는 내용의 문장을 출력하고 아니면 토핑이 존재하지 않는다는 내용의 문장을 출력해준다. 반복문이 끝나면 피자 만들기가 끝났다는 내용의 문장을 출력한다.



# Code with IF Statement

[3/5]

answer = 17

```
if answer != 42:  
    print("That is not the correct answer. Please try again!")
```



출력: That is not the correct answer. Please try again!

설명 : answer가 42(미리 정해놓은 정답)가 아니면  
틀렸다는 문장을 출력하는 if문이다. answer을 17  
로 초기화하였으므로 이 if문 안의 내용이 실행된다.

```
banned_users = ['andrew', 'carolina', 'david']  
user = 'marie'
```

```
if user not in banned_users:  
    print(user.title() + ", you can post a response if you wish.")
```



출력: Marie, you can post a response if you wish.

설명 : 금지된 유저 리스트에 지금 유저의 이름(marie)  
가 있지 않을 경우에 응답이 가능하다는 문장을 출력해  
준다. if 안이 참이기 때문에 이는 실행된다.

# Code with IF Statement

[4/5]

```
age = 17  
if age >= 18:  
    print("You are old enough to vote!")  
    print("Have you registered to vote yet?")  
else:  
    print("Sorry, you are too young to vote.")  
    print("Please register to vote as soon as you turn 18!")
```

출력 :

Sorry, you are too young to vote.  
Please register to vote as soon as you turn 18!



설명 : age(나이)가 18 이상이면 선거 가능하다는 문장과 선거여부를 묻는 문장을 출력하고, 아닐 경우에는 18세가 된 후에 투표하러 오라는 문장을 출력하는 if문이다. age는 17로 초기화되었고, 18보다 적으므로 else: 에 해당하는 코드가 실행된다.

# Code with IF Statement

[5/5]

VS

age = 12

```
if age < 4:  
    price = 0  
elif age < 18:  
    price = 5  
elif age < 65:  
    price = 10  
elif age >= 65:  
    price = 5
```

```
print ("Your admission cost is $" + str(price) +  
      ".")
```

age = 12

```
if age < 4:  
    price = 0  
elif age < 18:  
    price = 5  
elif age < 65:  
    price = 10  
else  
    price = 5
```

```
print ("Your admission cost is $" + str(price) + ".")
```

출력 (code2에 else가 else: 라는 가정하에):  
Your admission cost is \$5.  
출력은 두 코드 모두 동일하다.



설명 : 첫번째 코드에서는 if - elif문의 마지막까지도 elif를 사용하여 조건을 따진다. 따라서 마지막 elif 문의 조건이  $age \geq 70$ 과 같은 것이었다면 65~69처럼 특정 범위의 나이에 해당하는 가격을 대입하지 못할 수도 있다. 하지만 두번째 코드에서는 마지막에 else를 사용하며 나머지 모든 나이대에 대해서 가격을 대입하도록 한다.(어느 상황에서도 가격을 매길 수 있다.) 주어진 코드에서 출력은 동일하다.(elif  $age < 18$ 에 해당하는 코드가 실행되어 price=5가 됨)

# Code with Dictionary [1/5]

```
alien_0 = {'x_position' : 0, 'y_position' : 25, 'speed': 'medium'}
print("Original position: " + str(alien_0['x_position']))
```

```
# Move the alien to the right.
# Figure out how far to move the alien based on its speed.
if alien_0['speed'] == 'slow':
    x_increment = 1
elif alien_0['speed'] == 'medium':
    x_increment = 2
else:
    # This must be a fast alien.
    x_increment = 3
```

```
# The new position is the old position plus the increment.
alien_0['x_position'] = alien_0['x_position'] + x_increment
```

```
print("New Position: " + str(alien_0['x_position']))
```

설명 : alien\_0는 외계인에 관한 정보를 담고 있는 dictionary이다. 우선 현재의 x\_position을 출력 한다.(0) if문에서는 'speed'에 해당하는 값에 따라 x좌표의 증가량을 정한다. 'speed'가 빠를수록 x\_increment가 커지며, 속도가 'medium'이므로 x\_increment=2이다. alien\_0['x\_position']에 x\_increment를 추가한 후, 새로운 x\_position을 출력한다.(2)



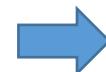
출력 :  
Original position: 0  
New Position: 2

# Code with Dictionary

[2/5]

```
favorite_languages = {  
    'jen' : 'python',  
    'sarah' : 'c',  
    'edward' : 'ruby',  
    'phil' : 'python',  
}
```

```
for name,language in  
favorite_languages.items():  
    print(name.title() + "'s favorite language is "  
        + language.title() + ".")
```

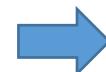


출력 :

Jens favorite language is Python.  
Sarahs favorite language is C.  
Edwards favorite language is Ruby.  
Phils favorite language is Python.

```
user_0 = {'username' : 'efermi',  
          'first' : 'enrico',  
          'last' : 'fermi',  
}
```

```
for key,value in user_0.items():  
    print("\nKey: " + key)  
    print("Value: " + value)
```



출력 :

Key: username  
Value: efermi

Key: first  
Value: enrico

Key: last  
Value: fermi

설명 : dict에 item()을 적용시키면 for loop로 key와 value를 동시에 다룰 수 있다. 이를 이용해서 사람 이름(key)과 그 사람이 다루는 프로그래밍 언어(value)를 한 문장에 출력시키는 코드이다.

설명 : 위의 코드와 유사하게 item()을 사용하여 key와 value를 for loop 안에서 동시에 다룬다. user의 여러 정보(유저명, 성, 이름)를 출력하는 for loop이다.

# Code with Dictionary

[3/5]

```
# Make an empty list for storing aliens.
```

```
aliens = []
```

```
# Make 30 green aliens.
```

```
for alien_number in range (0, 30):
```

```
    new_alien = {'color': 'green', 'points': 5, 'speed':  
'slow'}
```

```
    aliens.append(new_alien)
```

```
for alien in aliens[0:3]:
```

```
    if alien['color'] == 'green':
```

```
        alien['color'] = 'yellow'
```

```
        alien['speed'] = 'medium'
```

```
        alien['points'] = 10
```

```
    elif alien['color'] == 'yellow':
```

```
        alien['color'] = 'red'
```

```
        alien['speed'] = 'fast'
```

```
        alien['points'] = 15
```

```
# Show the first 5 aliens:
```

```
for alien in aliens[0:5]:
```

```
    print(alien)
```

```
    print("...")
```

출력 :

{'color': 'yellow', 'points': 10, 'speed': 'medium'}

{'color': 'yellow', 'points': 10, 'speed': 'medium'}

{'color': 'yellow', 'points': 10, 'speed': 'medium'}

{'color': 'green', 'points': 5, 'speed': 'slow'}

{'color': 'green', 'points': 5, 'speed': 'slow'}

...

설명 : alien에 관한 특성을 dictionary에 저장하였다. aliens를 0~2까지 슬라이싱한 리스트에 대해 for loop을 돌린다. 초기에 모든 alien은 'color': 'green'으로 초기화되기 때문에 if문에서 if alien['color'] == 'green': 다음 부분이 실행되며 0~2번째 alien은 color가 yellow로 바뀌고 기타 등등 값이 변화한다. elif문은 한번도 시행되지 않는다. 그리고 마지막에 for loop로 처음 5마리 alien의 속성을 보여준다.



# Code with Dictionary

[4/5]

```
# Store information about a pizza being ordered.  
pizza = {  
    'crust': 'thick',  
    'toppings' : ['mushrooms', 'extra cheese'],  
}  
  
# Summarize the order.  
print("You ordered a " + pizza['crust'] + "-crust pizza " +  
    "with the following toppings:")  
  
for topping in pizza['toppings']:  
    print("\t" + topping)
```

출력 :

You ordered a thick-crust pizza with the following toppings

mushrooms  
extra cheese



설명 : 피자 주문내역에 관한 정보가 dictionary에 들어가있다. 그 정보를 이용하여 피자 주문 내역을 한 문장 출력하고, topping 리스트를 for loop로 순회하며 출력한다.(\t로 tab 한번 누른 효과)

# Code with Dictionary

[5/5]

```
users = {'aeinstein' : {'first' : 'albert',
                        'last': 'einstein',
                        'location': 'princeton'},
         'mcurie': {'first' : 'marie',
                     'last' : 'curie',
                     'location' : 'paris'},
        }
```

```
for username, user_info in users.items():
    print("\nUsername: " + username)
    full_name = user_info['first'] + " " + user_info['last']
    location = user_info['location']

    print("\tFull name: " + full_name.title())
    print("\tLocation: " + location.title())
```

출력 :

Username: aeinstein  
Full name: Albert Einstein  
Location: Princeton

Username: mcurie  
Full name: Marie Curie  
Location: Paris



설명 : users dictionary안에 user의 정보(username, 성, 이름, 위치)가 저장되어 있다. users.items()를 사용해 for loop를 key와 value에 대해 동시에 돌려주고, user의 정보를 문장으로 출력한다. 성과 이름을 합쳐서 full\_name으로 만든다.

# Code with While Loop

[1/5]

```
prompt = "\nTell me something, and I will repeat it back to you:  
prompt += "\nEnter 'quit' to end the program. "
```

```
active = True  
while active:  
    message = input(prompt)  
    if message == 'quit':  
        active = False  
    else:  
        print(message)
```

입출력 예시 :

Tell me something, and I will repeat it back to you:  
Enter 'quit' to end the program. abcdefg  
abcdefg

Tell me something, and I will repeat it back to you:  
Enter 'quit' to end the program. hijklmn  
hijklmn

Tell me something, and I will repeat it back to you:  
Enter 'quit' to end the program. quit



설명 : input()을 이용해 입력을 받고, quit이 아닌 경우에는 그 입력을 그대로 다시 출력해주는 코드이다. quit을 입력하면 active 가 False가 되며 while문에서 빠져나올 수 있다.

```
prompt = "If you tell us who you are, we can personalize the messages you see."  
prompt += "\nWhat is your first name? "
```

```
name = input(prompt)  
print("\nHello, " + name + "!")
```

입출력 예시 :

If you tell us who you are, we can personalize the messages you see.  
What is your first name? KangHyeon



설명 : input()을 이용해 이름을 입력받고 입력받은 이름을 이용해서 환영인사를 출력해주는 코드이다.

Hello, KangHyeon!

# Code with While Loop [2/5]

```
height = input("How tall are you, in inches? ")
```

```
height = int(height)
```

```
if height >= 36:
```

```
    print("\nYou're tall enough to ride!")
```

```
else:
```

```
    print("\nYou'll be able to ride when you're a little older.")
```

입출력 예시1:

How tall are you, in inches? 38

You're tall enough to ride!

입출력 예시2:

How tall are you, in inches? 30

You'll be able to ride when you're a little older.

설명 : input()을 이용해 키를 입력받고, int()를 이용해 숫자로 바꾼 후, 36과 대소를 비교해서 더 크면 탈 수 있다는 문장을 출력하고, 작으면 더 크고 오라는 문장을 출력한다. 만약 input()에서 숫자가 아닌 것이 입력되면 int()에서 오류가 난다.



```
number = input("Enter a number, and I'll tell you if it's even or odd: ")
```

```
number = int(number)
```

```
if number % 2 == 0:
```

```
    print("\nThe number " + str(number) + " is even.")
```

```
else:
```

```
    print("\nThe number " + str(number) + " is odd.")
```

입출력 예시1:

Enter a number, and I'll tell you if it's even or odd: 3

The number 3 is odd.

입출력 예시2:

Enter a number, and I'll tell you if it's even or odd: 4

The number 4 is even.

설명 : input()과 int()를 이용해 정수를 입력받고, if문을 통해 만약 홀수이면 홀수라는 문장을, 아니면 짝수라는 문장을 출력한다.



# Code with While Loop [3/5]

```
current_number = 1  
while current_number <= 5:  
    print(current_number)  
    current_number +=1
```

출력 :

1  
2  
3  
4  
5



설명 : while문을 통해 현재 숫자가 5이 하일 경우에는 출력하고 1을 더해준다. 따라서 1부터 5까지의 정수가 출력된다.

```
prompt = "\nPlease tell me a city you have visited:  
prompt += "\n(Enter 'quit' when you are finished.)"
```

출력 :

```
while True:  
    city = input(prompt)  
  
    if city == 'quit':  
        break  
    else:  
        print("I'd love to go to " + city.title() + "!")
```

Please tell me a city you have visited:  
(Enter 'quit' when you are finished.)seoul  
I'd love to go to Seoul!

Please tell me a city you have visited:  
(Enter 'quit' when you are finished.)quit



설명 : while문과 input()을 통해 city를 계속 입력받아주고, quit이면 즉시 while문을 탈출한다. quit이 아닌 경우에는 city.title()로 첫 문자를 대문자로 바꾸고, 그 도시에 정말 가보고 싶다는 문장을 출력한다.

# Code with While Loop [4/5]

```
# Start out with some users that need to be verified,  
# and an empty list to hold confirmed users.  
unconfirmed_users = ['alice', 'brian', 'candace']  
confirmed_users = []
```

```
# Verify each user, until there are no more unconfirmed users.  
# Move each verified user into the list of confirmed users.  
while unconfirmed_users:
```

```
    current_user = unconfirmed_users.pop()
```

```
    print("Verifying user: " + current_user.title())  
    confirmed_users.append(current_user)
```

```
# Display all confirmed users.  
print("\nThe following users have been confirmed:")  
for confirmed_user in confirmed_users:  
    print(confirmed_user.title())
```

```
pets = ['dog', 'cat', 'dog', 'goldfish', 'cat', 'rabbit', 'cat']  
print(pets)
```

```
while 'cat' in pets:  
    pets.remove('cat')  
  
print(pets)
```

설명 : current\_user는 .pop()을 사용해서 unconfirmed\_users의 뒤부터 하나씩 빼낸 요소이다. while문을 사용해 current\_user를 하나하나 verify해주고, confirm된 user의 리스트에 추가해서 마지막에 출력해준다. 이름은 .title()을 이용해 표기해준다.



출력 :

Verifying user: Candace  
Verifying user: Brian  
Verifying user: Alice

The following users have been confirmed:  
Candace  
Brian  
Alice



출력 :

['dog', 'cat', 'dog', 'goldfish', 'cat', 'rabbit', 'cat']  
['dog', 'dog', 'goldfish', 'rabbit']

설명 : 리스트를 출력한 후 while 문과 remove()를 이용하여 리스트에 있는 'cat'을 모두 삭제하고, 남아있는 리스트를 출력한다.

# Code with While Loop [5/5]

```
responses = {}
```

```
# Set a flag to indicate that polling is active.  
polling_active = True
```

```
while polling_active:
```

```
    # Prompt for the person's name and response.
```

```
    name = input("\nWhat is your name?")
```

```
    response = input("Which mountain would you like to climb someday? ")
```

```
# Store the response in the dictionary:
```

```
responses[name] = response
```

```
# Find out if anyone else is going to take the poll.
```

```
repeat = input("Would you like to let another person respond? (yes/no) ")
```

```
if repeat == 'no':
```

```
    polling_active = False
```

```
# Polling is complete, show the results.
```

```
print("\n--- Poll Results ---")
```

```
for name, response in responses.items():
```

```
    print(name + " would like to climb " + response + ".")
```

입출력 예시:

What is your name?Cho

Which mountain would you like to climb someday? Gwanak

Would you like to let another person respond? (yes/no) yes

What is your name?Kim

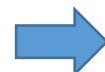
Which mountain would you like to climb someday? Namhan

Would you like to let another person respond? (yes/no) no

--- Poll Results ---

Cho would like to climb Gwanak.

Kim would like to climb Namhan.



설명 : while문을 이용해서 이름과 가고싶은 산을 입력 받는다. 다른 사람에게 물어볼지 여부를 입력받으며 'no'가 입력되기 전까지 이를 계속 반복한다. 'no'가 입력되면 지금까지 입력된 응답들을 저장해놓은 리스트를 for loop를 돌며 출력한다