[평생] 멀티미디어 일반형 양식

[원고] 빅데이터 수집 시스템 개발

12회차 : 문자열 처리와 그룹연산

내용전문가	백현숫	**교수설계(한기대)**	홍길동
협력업체	업체명	**교수설계(협력업체)**	홍길동

NCS 분류 정보	20-01-02-09	정보통신 - 정보기술 - 정보기술개발 - 빅데이터플랫폼구축
능력단위 정보	03	빅데이터 수집시스템 개발
능력단위 요소 정보	2001020903_17v1.1	빅데이터 수집시스템 설계하기

업무	작성자	버전	작성일	특이사항
원고 작성	백현숙	v.1.0	2019/09/11	2회차 원고
한기대 피드백		v.1.1		
원고 보완		v.2.0		
자문 진행		v.2.1		
원고 보완				

◈ 학습 목차

- 학습 목차를 작성해 주세요. (NCS 비적용 과정일 경우에는 NCS 및 능력단위 정보를 기입하지 않아도 됩니다.)

NCS 분류 정보	20-01-02-09	정보통신 - 정보기술 - 정보기술개발 - 빅데이터플랫폼구축
능력단위 정보	능력단위코드 기입	빅데이터 수집시스템 개발

모듈명	회차명	능력단위 요소		수행준거	
1. 모듈명 기입	01. 회차명 기입	1	능력단위요소명 기입	1.1	수행준거 기입
	02. 회차명 기입	1	능력단위요소명 기입	1.2	수행준거 기입
	03. 회차명 기입	1	능력단위요소명 기입	1.3 1.4	수행준거 기입 수행준거 기입
	04. 회차명 기입	1	능력단위요소명 기입	1.4	수행준거 기입
2. 모듈명 기입	05. 회차명 기입	2	능력단위요소명 기입	2.1 2.2	수행준거 기입 수행준거 기입

과정명		모듈명		회차명	

◈ 학습열기

파이썬은 문자열의 취급방법도 다른 언어들에 비해서 접근하기 쉽게 되어 있습니다. 문자열은 컴퓨터와 사람간에 가장 간극이 큰부분이라서 대부분의 언어에서 문자열을 다루기가 쉽지 않습니다. 초보 개발자분들이 프로그램 할때 가장 어려워 하는 부분중에 하나이기도 합니다. 그러나 파이썬은 애초에 언어를 만든 사람이 재미로 만들었다는 성격이 문자열에서도 드러납니다. 문자열의 연산이나 슬라이싱도 기본적으로 리스트와 큰 차이가 나지 않아서 사용하기 쉽습니다.

이 장에서는 문자열간의 연산, 문자열 형태의 숫자를 처리하는 방법, 문자열내에서 다른 서브 문자열을 검색하거나, 토큰의 분리와 결합등 문자열을 처리하는 방법등을 알아보도록 하겠습니다.

내레이션	학습자가 학습 화면에 구성한 내용을 보고 있다고 가정하고, 학습 설명으로 함께 들어야 할 음성 내용을 자세하게 기입해 주세요. **➔ 이건 나래이션해요? 저도 이런건 나래이션하면 좋겠어요..ㅜㅜ**

과정명		모듈명		회차명	

◈ 학습내용

- 문자열 처리와 그룹연산

◈ 학습목표

- 문자열로 부터 원하는 문자를 추출하거나, 수정하거나 제거할 수 있다.
- 서식문자열을 이용하여 다양한 형태의 데이터들을 문자열로 만들 수 있다
- 문자열을 분해하여 리스트로 만들거나 리스트 형태의 문자열을 결합하여 한 문장을 만들 수 있다.

내레이션

(기입하지 않습니다.)

간지 부분

◈ 문자열처리

문자열 인덱싱과 슬라이싱

문자열 포매팅

문자열 연산

내레이션

(기입하지 않습니다.)

◈ 문자열

- 문자열(String)이란 문자, 단어 등으로 구성된 문자들의 집합을 의미합니다
- 파이썬에서 문자열은 “ ”(큰따옴표) 나 ‘ ‘(작은 따옴표)로 둘러쌓아 만듭니다
- 우리눈에 숫자로 보이더라도 따옴표 안에 있는 데이터는 문자열입니다.
- 문자열의 예시

  “대한민국은 민주공화국이다“, “1234”, ‘ 도토리 3개’, ‘beautiful day’
- 문자열을 여러줄에 걸쳐 써야 할 경우에는 “ “ “ 를 세개 쓰거나 ‘ ‘ ‘ 를 세개 쓰면 됩니다.

```
mystr = """
        문장이 한줄을 초과할 경우에는
        이런식으로 작성하면 됩니다
        """""
"""
좌변에 변수가 없을 경우에는 문자열이 아니라 여러줄에 걸친 주석으로
판단됩니다.
"""
```

내레이션

◈ 문자열

파일명 : exam12_1.py

```
#문자열(String)
msg1 = "Hello Python"
msg2 = 'Python is easy'

msg3 = """
나보기가 역겨워 가실때에는
말없이 고이 보내드리우리다
영변에 약산 진달래꽃 아름따다
가실길에 뿌리오리다
"""

print(msg1)
print(msg2)
print(msg3)
```

변수=""" 형태를 유지해야 한다. 다음 데이터를 바로 써도 되고 옆에 처럼 한줄내려서 기술해도 된다.

```
Hello Python
Python is easy

나보기가 역겨워 가실때에는
말없이 고이 보내드리우리다
영변에 약산 진달래꽃 아름따다
가실길에 뿌리오리다
```

내레이션

◈ 숫자로 구성된 문자열 처리하기

- input 함수를 이용하여 데이터를 받아들이면 모두 문자열로 인식합니다.
- 문자열을 숫자로 변환하여 계산하려면 string 을 int 타입으로 전환해야 합니다.
- 타입전환은 하나의 타입을 다른 타입으로 전환하는데 파이썬의 타입전환은 변수 = (전환타입)수식 형태로 이루어집니다.

```
value1 = int( input("정수를 을 입력하세요") )
value2 = float( input("실수를 입력하세요 "))
만일 입력자가 정수를 입력하지 않으면 예외가 발생한다
```

내레이션

◈ 입력된 문자열이 숫자로 이루어졌는지 확인하는 함수

- input 함수를 이용하여 받아들인 데이터가 만일 숫자가 아닌 문자가 포함되어 있다면 파이썬은 예외를 발생시킵니다.
- 예외를 발생시키지 않고 사전에 오류를 점검하고 싶다면 우선 타입전환이 가능한지 확인을 해야 합니다.
- 문자열이 숫자로 구성되었는지 확인하기 위한 함수가 isdigit, isdecimal, isnumeric 세개가 있습니다.

isdecimal : 문자열이 숫자로만 구성되었으면 True, 아니면 False반환

isdigit : 특수 기호 중에서 어깨위에 제곱이나 세제곱을 표시하는 문자처럼 '숫자처럼 생긴' 모든 글자를 다 숫자로 판단합니다.

idnumeric : 숫자값 표현에 해당하는 텍스트까지 숫자로 인정해줍니다 . 예) "½"

내레이션

◈ 숫자로 구성된 문자열 처리하기

파일명 : exam12_2.py

```
value1 = input("정수를 입력하세요 ")
value2 = input("정수를 입력하세요 ")

#print( value1.isdigit() )
if value1.isdecimal() and  value2.isdecimal():
    result = value1 + value2
    print("결과는 ", result, " 입니다 ")

    #위의 코드 수정하기
    result2 = int(value1) + int(value2)
    print("결과는 ", result2, " 입니다 ")
else:
    print("정수를 입력하지 않아서 연산에 실패했습니다")
```

입력받은 데이터가 숫자로 전환가능한지 확인한다. 둘다 True일때 수행을 하고 아니면 정수가 입력되지 않아 연산에 실패했다는 오류메시지가 발생한다

두개의 정수를 입력받아 그 결과를 출력하려고 합니다. 12와 34를 입력하였다면 결과값이 46가 나오는 것이 아니라 1234 가 나옵니다. input 함수는 반환이 무조건 string 타입입니다.
파이썬에서 string 타입에 string 타입을 더하면 결과는 문자열을 결합 합니다. 그래서 결과가 1234로 나옵니다.

```
정수를 입력하세요 12
정수를 입력하세요 34
결과는  1234  입니다
결과는  46  입니다
```

내레이션

◈ 문자열 출력

- 문자열을 출력시 줄바꿈이나 특별한 문자등을 출력하기 위해 이스케이프 문자를 사용합니다.
- 이스케이프 코드란 프로그래밍할 때 사용할 수 있도록 미리 정의해 둔 "문자 조합 " 입니다.
- 출력을 보기 좋게 정렬하는 용도로 사용 합니다. 문자열 안에 함께 사용합니다
- 파이썬에서는 \(역슬래쉬) 와 함께 사용하는데 키보드의 우축 상단에 백스페이스 아래에 있는 문자입니다.
- 영문폰트에서는 \ 로 보이고 한글폰트에서는 \ 로 보입니다. 다음 표에 있는 내용이 자주 사용하는 문자입ㄴ다

문자	의미
\n	문자열에서 줄바꿈을 할 때 사용합니다.
\t	문자열에서 탭키를 적용할 때 사용합니다.
\\	역슬래쉬 문자 자체를 출력하고자 할 때 사용합니다
\’	작은 따옴표를 출력하고자 할 때 사용합니다.
\”	큰 따옴표를 출력하고자 할 때 사용합니다

내레이션

◈ 문자열 출력

```
print("₩"파이썬₩"은 데이터분석에 사용되는 언어입니다 ")
print("₩'파이썬₩'은 ₩'딥러닝₩'에도 사용됩니다")
print("파이썬₩n데이터분석₩n크롤링")
print("파이썬₩t데이터분석₩t크롤링")
print("c:₩₩pythonwork_space₩₩exam12₩₩exam12_1.py")

msg = "파이썬₩t뷰티플스프₩n사이킷런"
print(msg)

path = r"c:₩pythonwork_space₩exam12₩exam12_1.py"
print(path)
```

escape문자를 사용해서 다양한 기능을 구현할 수 있습니다. 가끔 역슬래쉬 문자를 무력화 시켜야 할 때가 있습니다. 그럴때는 문자열 앞에 r을 붙여주면 됩니다. r을 붙이면 escape 문자가 무력화되서 역슬래쉬를 를 두개씩 붙이지 않아도 됩니다.

```
"파이썬"은 데이터분석에 사용되는 언어입니다
'파이썬'은 '딥러닝'에도 사용됩니다
파이썬
데이터분석
크롤링
파이썬	데이터분석	크롤링
c:\pythonwork_space\exam12\exam12_1.py
파이썬	뷰티플스프
사이킷런
c:\pythonwork_space\exam12\exam12_1.py
```

내레이션

◈ 문자열 연산하기

- 문자열을 더하거나 곱할 수 있습니다. 두가지 연산만 허용됩니다.
- 더하기 연산은 두개의 문자열을 합쳐서 하나의 문자열을 만듭니다.
- 곱하기 연산은 특정 문자열을 지정한 회수만큼 반복하여 긴 문자열을 만들때 사용합니다 .

```
s = 'Hello'
s2 = 'Python'
s3 = s + " " + s2
print(s3)              결과 : Hello Python


line =  " - "  * 5
print(line)            결과 -----      앞의 문자를 * 뒤에 숫자만큼 반복한다
```

내레이션

◈ 문자열 연산하기

```
#exam12_4.py

print()
s1 = "Hello"
s2 = "Python"

s = s1 + " " + s2
print(s)

s = s1 * 3
print(s)

line = "-" * 50
print(line)

for i in range(1, 11):
    print("*" * i)
print()
```

```
Hello Python
HelloHelloHello
--------------------------------------------------
*
**
***
****
*****
******
*******
********
*********
**********
```

내레이션

◈ 문자열 인덱싱

- 인덱싱을 이용해 문자열의 요소들을 하나씩 접근할 수 있습니다. 문자열의 인덱싱은 문자열변수[위치값]의 형태로 데이터를 한글자씩 읽을 수 있습니다. 인덱싱은 값을 읽기 위한 용도로만 사용되고 쓰기 위한 용도로는 사용되지 않습니다.
- 원래의 글자 범위를 벗어나면 예외가 발생합니다.
- 인덱싱을 –를 사용할 수 도 있습니다

```
s = “Hello”
print( s[0] )
print( s[1] )

s[0] = “h” <- 이건 허용되지 않습니다.
첫글자를 소문자로 바꾸려면 별도의 문자열 객체를 만들어서 복사해야 합니다.

s2 = “h” + s[1] + s[2[ + s[3] + s[4]
여러개의 글자의 경우 실제로 이렇게는 사용하지 않고 다음 페이지의 슬라이딩을 사용합니다.

슬라이딩 예) s2 = "h" + s[1:]
```

내레이션

◈ 문자열 슬라이딩

- 슬라이딩은 창문을 열었다 닫았다 하는것처럼 보여서 붙인겁니다. 인덱싱은 한번에 하나의 글자만 읽을 수 있는데 반해서 슬라이딩은 어디부터 어디까지라고 지정을 할 수 있습니다.
- 문자열변수[시작:종료] 형태로 콜론(:) 연산자를 이용해서 시작과 끝을 지정하는데 시작과 콜론만 지정하면 시작위치부터 마지막 문자까지, 콜론과 마지막 위치를 지정하면 처음부터 지정한 위치까지 문자열을 추출할 수 있습니다

```
s = “Hello”
print( s[0:5] )      Hello
print( s[3:] )       Hel , 0부터 시작해서 마지막에 지정한 3번을 제외한 나머지를 추출합니다
print( s[:3] )       lo , 3번방부터 시작해서 마지막까지

s = “blue diamond”
s2 = s[0:4]          blue
s3 = s[5:]           diamond
```

내레이션

◈ 문자열 인덱싱과 슬라이싱

```
#exam12_5.py
str = "Hello Python"

print( str[0] )
print( str[1] )
print( str[2] )
print( str[3] )
print( str[4] )

print( str[-1] )
print( str[-2] )
print( str[-3] )

print( str[0:5] ) #0~5번째 문자열
print( str[:5] )
print( str[6:] )

#인덱싱을 이용한 문자열 뒤집기
for i in range(-1, -(len(str)+1), -1):
    print(str[i], end='')
print()
```

인덱싱을 -를 사용하면 뒤쪽에서부터 시작합니다

```
H
e
l
l
o
n
o
h
Hello
Hello
Python
nohtyP olleH
```

내레이션

◈ 문자열 인덱싱과 슬라이싱

```
#exam12_6.py
print()
str = "korea"
#str[0] ='K' – 인덱싱으로 값을 바꿀 수 없다.
print(str)

str = 'K' + str[1:]
print(str)

str = "pyton programing"

s1 = str[0:3] + "h" + str[3:5]
s2 = str[6:12] +"m" + str[12:]
print(s1, s2)
print()
```

문자열 인덱싱은 데이터를 읽기위한 용도로만 사용된다. 에러발생

문자열 추출과 +연산을 통해 새로운 문자열을 만들어야 한다

```
korea
Korea
python programming
```

내레이션

◈ 서식문자열 처리하기

- 출력값이 문자와 숫자(정수, 실수등) 그리고 여러개의 문자열로 구성되었을 경우 정해진 형태로 출력하고자 하는것을 문자열 포매팅이라고 합니다.
- 파이썬은 다음처럼 두가지 방식을 지원하고 있습니다 .

방법1) s = "가로 %d 세로 % d인 사각형의 면적은 %d 입니다" % (width, height, width*height)
%d는 서식문자열입니다. 문자열 중간 중간 삽입할 데이터가 있을때 그 위치에 놓은 값들의 형태에 따라 다른 문자를 씁니다. %d는 정수를 그 자리에 놓겠다는 의미입니다. "문자열" % (튜플) 형태로 값을 전달하면 문자열에 있는 기호 %d 들에 괄호안의 값들이 차례로 전달됩니다.
%d말고도 사용되는 서식문자열은 다음 페이지에 있습니다.
방법2) s = "가로 {} 세로 {} 인 사각형의 면적은 {}입니다.".format(width, height, width*height)
{} 위치에 값들이 투입됩니다.

내레이션

◈ 문자열 포매팅

- 아래 표는 문자열 포매팅에서 자주 사용하는 형시입니다

코드	설명
%d	정수
%f	실수
%s	문자열
%c	한문자
%o	8진수
%x	16진수
%%	%기호 자체를 출력하고자 할때

내레이션

◈ 문자열 포매팅

```
#정수는 d
s = "결과는 %d 입니다." % 10
print(s)

s = "안녕하세요 %s님" % "홍길동"
print(s)

#여러개
s = "%s 님의 주급은 %d 입니다." % ("홍길동", 500000)
print(s)


#실수는 f를 사용한다
print( "%f  X %f = %f" % (2.4, 3.5, 2.4*3.5))
print( "%7.2f  X %7.2f = %7.2f" % (2.4, 3.5, 2.4*3.5))

print("현재 목표달성율은 %d%% 입니다." % 50)


#정렬
print("%5d*"  % 12)  #5자리 확보 후 숫자를 오른쪽부터 채워서 출력한다
print("%-5d*" % 12)  #5자리 확보 후 숫자를 왼쪽부터 채워서 출력한다
```

문자열 중간에 값을 끼워넣을 위치에 %d를 넣어놓고 문자열 뒤에 %기호 그리고 데이터를 주면 %d 대신에 데이터가 그 위치에 들어갑니다.
%s는 데이터가 문자열의 경우입니다

여러개의 데이터를 문자열 중간에 넣으려면 %뒤에 데이터를 튜플 형식으로 전달하면 됩니다. 튜플은 여러개의 데이터를 하나로 묶어주는 타입인데 반드시 괄호를 해야 합니다.

```
결과는 10 입니다.
안녕하세요 홍길동님
홍길동 님의 주급은 500000 입니다.
2.400000  X 3.500000 = 8.400000
   2.40  X    3.50 =    8.40
현재 목표달성율은 50% 입니다.
   12*
12   *
```

내레이션

◈ 문자열 포매팅

#exam12_8.py

```
print("{} + {} = {}".format(3,4,3+4))
print("{1} + {0} = {2}".format(3,4,3+4))
print("{:.2f} X {:.2f} = {:.2f}".format(2.4, 3.5, 2.4*3.5))
print("{} is {}".format("flower", "beautiful"))

print("{:<5d}*".format(12))
print("{:>5d}*".format(12))

for i in range(1,6):
    print("{:04d}".format(i))
```

" ".format() 함수는 문자열에 여러가지 형태의 데이터를 끼워넣어 하나의 문자열을 만들때 사용합니다.
" {0} {1} {2}".format(3, 4, 5) 형태로 기술한다. {} 기호안에 값이 들어갑니다. {} 개수만큼 format함수에 값을 전달해야 합니다.
{0} {2} {1} 식으로 숫자을 지정해주면 전달해주는 값의 순서가 맞추어서 차례대로 전달됩니다. 0의 자리에는 첫번째로 전달된 3이 2에는 세번째로 전달된 인자 5가 1의 자리에는 두번째로 전달된 값 4가 출력됩니다.
순서대로 전달될때는 굳이 숫자를 써주지 않으면 차례대로 전달됩니다.

자릿수 지정은 { :4d } 형태로 기술해줍니다. :뒤에 4d라고 쓰면 정수 들어갈 자리 4자리를 확보하고 오른쪽 끝에서부터 채워옵니다. 만일 왼쪽 정렬을 하려면 {:>4d} 형태로 기술하면 됩니다.

{:04d} 는 자릿수를 4자리로 맞추고 오른쪽부터 숫자를 채우는데 빈 공간으 0으로 채우라는 의미입니다.

내레이션

간지 부분

◈ 문자열 함수들

문자열의 결합과 분리, 위치 찾기, 대소문자변경하기

내레이션

(기입하지 않습니다.)

◈ 문자열 위치 찾기

- 문자열에서 특정 문자열이 몇번 존재하는지 찾기 위해서는 count 함수를 사용합니다 .
- count 함수는 찾는 문자열이 몇번 등장했는지를 반환합니다.
- 문자열에서 특정 문자열이 어디에 존재하지는 찾기 위해서는 index함수를 사용합니다.
- index 함수는 문자열의 위치를 반환하는데 만일 찾는 문자열이 없을 경우에는 예외를 반환하므로 보통 count함수나 in 연산자를 활용해서 먼저 데이터가 있는지 확인하고 나서 index함수를 사용해야 합니다 .

```
s="I like star, red star yello star, blue star“

print( "star 개수 : ", s.count("star"))
print( "star 단어 존재 여부 : ", "star" in s)
print( "star 첫번째 위치 : ", s.index("star"))
print( "star 두번째 위치 : ", s.index("star", 12))
```

```
Traceback (most recent call last):
  File "exam12_9.py", line 54, in <module>
    print( s.index("gray"))
ValueError: substring not found
```

index함수에 존재하지 않는 값을 전달할 때 에러메시지

내레이션

◈ 문자열 위치 찾기

- count 함수와 index 함수를 이용해서 문자열내의 특정 문자열을 모두 찾아보는 코드를 작성해보겠습니다.

```
s="I like star, red star yello star, blue star“

print("star 위치")
key = "star"
pos=0
while  s.count(key, pos)!=0:
   pos = s.index(key, pos)
   pos = pos + len(key) #단어길이만큼 다음 위치로 이동한다
   print(pos)
```

내레이션

◈ 문자열 분리하기

- split 함수는 문자열을 특정 문자를 기준으로 분리하여 문자열의 list 형태로 반환합니다.
- split 함수의 매개변수에 특정 문자를 지정하지 않으면 공백으로 토큰을 나눕니다.

```
s="I like star red star yello star blue star“

words = s.split()
print( words )  리스트 형태입니다.

s = "소나무,전나무,오동나무,사시나무,향나무"
s2 = s.split(",")
print(s2)
```

내레이션

◈ 문자열 변환하기

- upper 함수는 문자열을 대문자로 변환하는 함수입니다
- lower 함수는 문자열을 소문자로 변환하는 함수입니다.
- replace 함수는 문자열을 특정 문자열로 전환합니다.

```
s="I like star, red star yello star, blue star“
s = s.upper()
print(s)
결과 : I LIKE STAR RED STAR YELLO STAR BLUE STAR

s = s.lower()
print(s)
결과 : i like star red star yello star blue star

s=s.replace(“star’, “moon”)
print(s)
결과 : i like moon red moon yello moon blue moon
```

내레이션

◈ 문자열 앞뒤 공백 제거 함수

- 문자열의 앞뒤에 있는 공백문자를 제거하기 위해서 strip 함수를 사용합니다.
- 양쪽을 다 제거할때는 strip,왼쪽의 공백만 제거하면 lstrinp, 오른쪽 공백만 제거하면 rstrip함수를 사용합니다.

```
s = "   space remove   "
print( "*" + s + "*")
print( "*" + s.lstrip() + "*")
print( "*" + s.rstrip() + "*")
print( "*" + s.strip() + "*")

*   space remove   *          문자열 앞뒤로 공백이 있다
*space remove   *             문자열 왼쪽의 공백이 제거되었다
*   space remove*             문자열 오른쪽이 공백이 제거되었다
*space remove*                문자열 양쪽의 공백이 제거되었다
```

내레이션

◈ 문자열 결합하기(join)

- join의 문자열 리스트를 전달받아 하나의 문자열을 만들때 사용합니다.

```
#문자열 결합
s = ",".join(["apple", "orange", "banana", "mango", "cherry"])
print(s)

s = " ".join(["apple", "orange", "banana", "mango", "cherry"])
print(s)


apple,orange,banana,mango,cherry
apple orange banana mango cherry
```

내레이션

◈ 문자열 관련 함수들

```
#exam12_9.py
s="I like star, red star yello star, blue star"
words = s.split()
print( words )

s1 = "소나무,전나무,오동나무,사시나무,향나무"
woods = s1.split(",")
print(woods)

print( "문자열 길이 : ", len(s) )

#특정 문자의 개수
print ("star 개수 : ", s.count("star"))

#star 의 위치
print("star 첫번째 위치 : ", s.index("star"))
#모든 star 찾기

print("star 위치")
key = "star"
pos=0
while  s.count(key, pos)!=0:
    pos = s.index(key, pos)
    pos = pos + len(key) #단어길이만큼 다음 위치로 이동한다
    print(pos)
```

내레이션

◈ 문자열 관련 함수들

```
print("소문자 대문자 체인지")
s2 = s.upper()
print(s2)

s1 = s2.lower()
print(s1)

#문자열 대체하기
#star 을 moon으로 대체
s = s.replace("star", "moon")
print(s)

#공백제거
s = "    space remove    "
print( "*" + s.lstrip() + "*")
print( "*" + s.rstrip() + "*")
print( "*" + s.strip() + "*")

#문자열 삽입
s = ",".join(["apple", "orange", "banana", "mango", "cherry"])
print(s)
s = " ".join(["apple", "orange", "banana", "mango", "cherry"])
print(s)
```

```
['I', 'like', 'star', 'red', 'star', 'yello', 'star', 'blue', 'star']
['소나무', '전나무', '오동나무', '사시나무', '향나무']
문자열 길이 :  41
star 개수 :  4
star 첫번째 위치 :  7
star 위치
11
20
31
41
소문자 대문자 체인지
I LIKE STAR RED STAR YELLO STAR BLUE STAR
i like star red star yello star blue star
I like moon red moon yello moon blue moon
*   space remove    *
*space remove    *
*   space remove*
*space remove*
apple,orange,banana,mango,cherry
apple orange banana mango cherry
```

내레이션

◈ 적용하기

1. data 폴더내의 data.txt 파일을 읽어서 토큰을 분리하고 토큰의 개수는 몇개인지 출력하시오

파일읽기 예

```
file = open("./data/data.txt", encoding="utf8")
data = file.read()
print(data)
file.close()
```

2. Too 와 Time 단어가 각각 몇번 등장했는지 횟수를 출력하시오

3. who 등장 위치를 리스트에 추가하여 어떤 위치에 등장했는지 위치값 리스트를 출력하시오

4. 모든 토큰을 대문자로 바꾸어서 토큰의 번호, 대문자로 전환한 토큰을 출력하시오

내레이션

(기입하지 않습니다.)

◈ 적용하기- 풀이

```
file = open("./data/data.txt", encoding="utf8")
data = file.read()
print(data)
file.close()

#1)토큰분리하기
tokens = data.split()
print( len(tokens))

#2)Too 와 Time 단어가 각각 몇번 등장했는지
print("Too 개수 : ", data.count("Too"))
print("Time 개수 : ", data.count("Time"))
```

내레이션

(기입하지 않습니다.)

◈ 적용하기 -> 풀이

```
#3) who 등장 위치를 배열로
key = "who"
pos = 0
positions = list()
while data.count(key, pos)!=0:
    pos = data.index(key, pos)
    positions.append(pos)
    pos = pos + len(key)
print(positions)

#4)모든 토큰을 대문자로 바꾸어서 토큰의 번호, 대문자로 전환한 토큰 출력하기
for i, token in enumerate(tokens):
    print(i, token.upper() )
```

```
34
Too 개수 :  4
Time 개수 :  2
[29, 58, 86, 117, 143]
0 TIME
1 IS..
2 TOO
3 SLOW
4 FOR
5 THOSE
6 WHO
7 WAIT
8 TOO
9 SWIFT
10 FOR
11 THOSE
12 WHO
13 FEAR
14 TOO
```

리스트가 너무 많아 생략

내레이션

(기입하지 않습니다.)

◈ 문제풀기

문제 1) 다음 연산의 수행 결과는 ?

x1 = “123”

x2 = “456”

result = x1 + x2

① 123456

② 579

③ 예외발생

④ 142536

정답) 1

해설) 문자열간의 + 연산은 문자열을 덧붙인다

난이도) 5(1:아주어려움, 2:어려움 3: 보통 4: 쉬움 5: 아주 쉬움)

관련페이지번호) 8

내레이션

(기입하지 않습니다.)

번호	문제	정답	난이도	해설	관련학습 보기
2	디음처럼 문장이 있을때 수행 결과를 작성하시오 s = “DeepLearning” print(s[4:])	Learning	3	4번째 데이터부터 슬라이딩이므로 Learning가 된다.	16
3	**다중 중 에러가 발생하는 부분은 ?** s = “machineLearning” ① s[0]=“M” ② s2 = “M” + s2[:1] ③ s2 = s[0] + s[1] + s[2] + s[3[+ s[4] ④ s2 = “machine”*2	1	3	인덱스를 이용해서 데이터를 수정할 수 없습니다	15
4	**다음 프로그램의 수행 결과는** **s = “{1} {3} {0} {2}”.format(1,2,3,4)** **print(s)** **1) 1 2 3 4** **2) 3 2 1 4** **3) 2 4 1 3** **4) 4 3 2 1**	3	3	{1} {3} {0} {2} 는 format 함수에 전달되는 인자들의 위치값입니다. 1 -0, 2 -1, 3-2, 4-3 에 해당됩니다. 그래서 2 4 1 3 의 결과가 나옵니다	22

번호	문제	정답	난이도	해설	관련학습보기
5	토큰을 분리하려고 합니다. 다음 중 적당한 함수는? ① index ② isdecimal ③ strip ④ split	4	4	토큰을 분리하기 위해서는 split 함수를 사용합니다.	26
6	**다음처럼 토큰 리스트가 있을때 모든 토큰을 결합사여 하나의 문자열로 만들 수 있는 함수는?** **msg = ["I", "like", "star"]** **① join** **② append** **③ concatenation** **④ add**	1	5	join 함수를 이용해서 토큰들을 하나로 묶을 수 있습니다	29
7	**다음 단어를 대문자로 출력하고자 합니다. 적절한 함수는 ?** **① capital** **② upper** **③ lower** **④ capitalize**	2	4	대문자로 변환하는 함수는 upper 입니다 .	27

번호	문제	정답	난이도	해설	관련학습보기
8	다음 프로그램의 수행 결과는 ? s = “ My name is Tom ” print (“*” + s.lstrip() + “*”) 1) *My name is Tom * 2) * My name is Tom * 3) *My name is Tom* 4) *MynameisTom*	1	5	lstrip 함수는 문자열 왼쪽의 공백을 삭제해줍니다	28
9	s1 = “35” s2 = “120” result = ( ) 일때 연산의 결과가 155가 나오려면 어떻게 해야 하는지 비어있는 공간의 코드를 완성하시오	int(s1) + int(s2)	3	문자열의 경우 먼저 정수형태로 전환을 해야 한다 정수 전환 연산자는 int 이다	8
10	**다음 처럼 데이터가 출력되길 원할때 올바른 format 방법은** **00001** **00002** **00003** **00004** **{ } 이부분에 포맷팅을 완성하세요**	:05d {:05d}	1	{ :05d} 정수 출력을 위한 5개의 자리를 잡고 나머지 빈자리를 지정한 문자 0으로 채우시오	22

과정명		모듈명		회차명	

◈ 핵심요약

▶.문자열

- 문자열은 인덱싱과 슬라이싱을 통해 접근하거나 추출한다. 파이썬에서 인덱싱을 이용하거나 슬라이싱을 이용해 데이터를 추출할 수 있지만 데이터의 수정은 불가능하다. 인덱싱과 슬라이싱한 추출한 문자열과 다른 문자열을 결합하여 새로운 문자열을 만들 수 있다
- index 함수는 서브문자열의 위치를 알려주는 함수지만, 문자열이 없을 경우 예외를 발생한다. 프로그램 상에서 실제로 적용할 경우에는 count 함수를 이용해 단어가 존재하는지 확인하거나 in 연산자를 이용해서 확인한 후 사용해야 한다
- 포매팅(%나 forma함수) 다양한 데이터들을 하나의 문자열로 만들어 출력을 아름답게 꾸밀 수 있다.
- 가급적 format 함수를 사용하자.

▶문자열 처리 함수들

- 문자열이 숫자만으로 구성되었는지 확인하는 함수에는 isdigit, isdecimal, isnumeric 함수가 있다
- 문자열 양쪽의 쓸데 없는 공백을 제거하가 위해서는 strip(양쪽), lstrip(왼쪽), rstrip(오른쪽) 함수를 사용한다
-

내레이션

(기입하지 않습니다.)

◈ 학습맺음

- 이번 회차에서는 문자열과 문자열을 처리하는 함수들에 대해서 살펴보았습니다 .

◈ 참고자료

- https://docs.anaconda.com/anaconda/
- https://code.visualstudio.com/docs/getstarted/themes
- https://pandas.pydata.org/pandas-docs/stable/getting_started/tutorials.htm
- https://wikidocs.net/13#joinl

내레이션

(기입하지 않습니다.)

◈ 회차 메타데이터

- 주요학습내용 : 해당 회차 또는 레슨의 주요학습내용을 자세히 기입해 주세요.
- 검색 키워드 : 학습자가 검색창에 어떤 검색어를 입력하면 본 회차 또는 본 레슨이 검색될 수 있을지 검색 키워드를 5개 기입해 주세요.

제목	주요학습내용	검색 키워드1	검색 키워드2	검색 키워드3	검색 키워드4	검색 키워드5
12. **문자열 처리와 그룹 연산**	문자열과 문자열 처리함수	join	splite	strip	index	count