

파이썬 기초

10 문자열 다루기



홍필두 교수
(파이썬기초)



학습내용

- 01 문자열 다루기
- 02 문자열 함수
- 03 포매팅





학습목표

- 문자열에 대하여 이해하고 사용할 수 있다.
- 문자열 함수에 대하여 이해하고 사용할 수 있다.
- 문자열 포매팅에 대하여 이해하고 사용할 수 있다.



사전 학습

“문자열 다루기”

실제 업무에서 우리가 문자열을 다루어야 되는 경우는 매우 빈번합니다.

프로그램 처리결과를 화면이나 출력물에 원하는 양식으로 출력하기 위해서 문자열을 가공하여야 하는 경우도 많습니다.

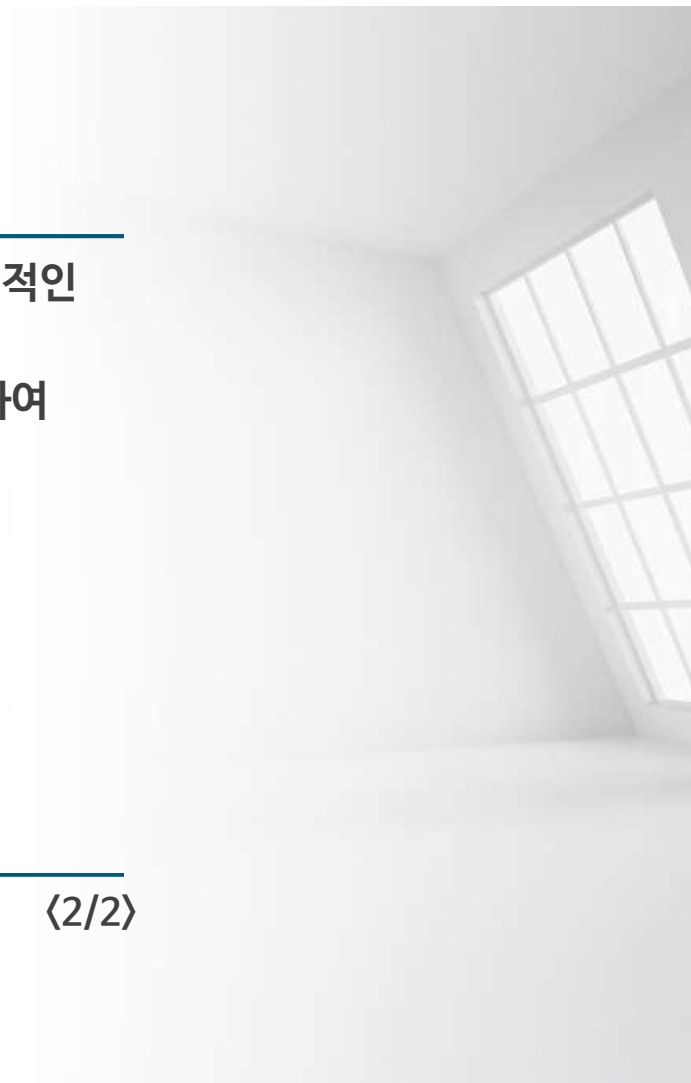
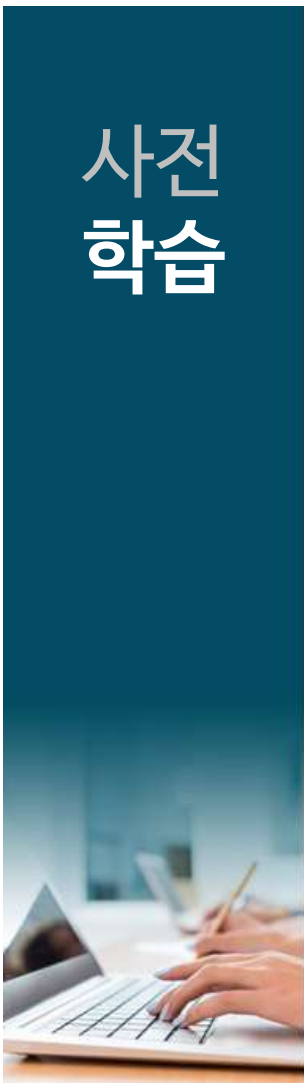
또 수집된 자료를 가공하여 데이터베이스에 적재한다던가, 프로그램간 데이터를 주고 받으면서 해당 프로그램에서 처리를 위하여 일정한 형식에 맞추는 업무 등은 **대부분 문자열을 가공하는 작업**입니다.

사전 학습

“문자열 다루기”

이번 강의에서는 이러한 문자열을 다루는 특징적인 몇 가지 사항을 배웁니다.

여러분은 먼저 해당 내용에 대하여 검색을 통하여 미리 알아보고 조사해 본 후, 본 강의를 듣도록 합시다.





01

문자열 다루기

- 1) 문자열의 성격
- 2) 문자열 자르기

1) 문자열의 성격



문자형 변수의 성격은 컬렉션의 리스트와 유사함



문자리스트, 즉 String(문자열)은 첨자로 대응 됨



한글은 2byte로 처리되나 문자열에서 1개의 첨자에 대응되는 것에 유의할 것

```
>>> a="abcde"
>>> a[0],a[1],a[2],a[3],a[4]
('a', 'b', 'c', 'd', 'e')
>>> a[-1],a[2],a[-3],a[-4],a[-5]
('e', 'c', 'c', 'b', 'a')
>>> a="가나다라마"
>>> a[0],a[1],a[2],a[3],a[4]
('가', '나', '다', '라', '마')
```

2) 문자열 자르기



문자 첨자 처리시
[시작:끝]으로 표시가능



끝에 해당되는 숫자
앞까지 해당됨에 유의



음수 첨자에 유의할 것

```
>>> a="abcde"
>>> a[1:5]
'bcde'
>>> a[1:2]
'b'
>>> a[:6]
'abcde'
>>> a[3:]
'de'
>>> a[2:-2]
'c'
>>> a[2:-3]
''
>>> a[1:-1]
'bcd'
```


02

문자열 함수

- 1) 찾기
- 2) 검사
- 3) 대소문자 바꾸기
- 4) 공백제거
- 5) 구분자로 문자열 나누기
- 6) 구분자 연습

1) 찾기

구분	설명
len(문자열)	문자열의 길이를 반환
str.find(문자)	해당문자를 앞으로부터 찾아서 위치를 반환
str.rfind(문자)	해당문자를 뒤로부터 찾아서 위치를 반환
str.index(문자), str.rindex(문자)	find(), rfind()와 기능은 동일하나 문자열을 못 찾으면 에러발생
str.count(문자)	해당문자를 찾아서 몇 개가 있는지 개수를 반환

1) 찾기

```
>>> a="abcdefghijklmabcdefghijklm"
>>> a.find("f") #앞에서 부터 찾음, 찾은 값의 인덱스임
5
>>> a.rfind("f") #뒤에서 부터 찾음
18
>>> a.find("i"),a.rfind("i") #만일 찾는 문자가 하나라면 결과는 동일
(8, 8)
>>> a.index("b"),a.rindex("b") #index함수도 결과는 동일
(1, 14)
```

```
>>> a.rindex("z") #하지만 찾는 문자가 없으면 에러발생
Traceback (most recent call last):
  File "<pyshell#19>", line 1, in <module>
    a.rindex("z")
ValueError: substring not found
>>> a.count("f") #해당 문자의 개수를 반환
2
>>> a.rfind("abc") #문자열을 넣어도 됨
13
```

2) 검사



“포함문자 in 문자”의 형태로 찾는 것도 가능함



해당 문자를 가지고 있는지 검사함



“만일 “f”가 a안에 있다면”으로 표현되기 때문에
이 방식이 보다 자연어적인 표현일 수 있음

```
>>> a="abcdfegabcdfeg"
>>> if a.find("f") > 0:
        print ("a가 들어 있음")
a가 들어 있음
>>> if "f" in a : #만일 "f"가 a안에 있다면..으로 자연적 영어표현
        print ("a가 들어 있음")

a가 들어 있음
```

3) 대소문자 바꾸기

구분	설명
str.upper()	문자열을 모두 대문자로 바꿈
str.lower()	문자열을 모두 소문자로 바꿈

```
>>> a="abcDefGh"  
>>> a.upper()  
'ABCDEFGH'  
>>> a="abcDefGh"  
>>> a.lower()  
'abcdefgh'
```

4) 공백제거

구분	설명
<code>str.lstrip()</code>	문자열 왼쪽 끝의 모든 공백을 지움
<code>str.rstrip()</code>	문자열 오른쪽 끝의 모든 공백을 지움
<code>str.strip()</code>	문자열 왼쪽 끝과 오른쪽 끝의 모든 공백을 지움
<code>str.replace(" ", "")</code>	이와 같은 표현은 모든 공백을 지움
<code>str.replace(이전_문자열, 바꿀_문자열)</code>	이전 문자열을 바꿀 문자열로 바꿈

4) 공백제거

```
>>> a="  abc  def  hij      "  
>>> a.lstrip()  
'abc  def  hij      '  
>>> a="  abc  def  hij      "  
>>> a.rstrip()  
'  abc  def  hij'  
>>> a="  abc  def  hij      "  
>>> a.strip()  
'abc  def  hij'  
>>> a="  abc  def  hij      "  
>>> a.replace(" ", "")  
'abcdefhij'
```

5) 구분자로 문자열 나누기

구분	설명
<code>a=str.split(구분자)</code>	str의 문자열을 구분자 문자를 찾아서 하나씩 나누어 a에 리스트로 반환

5) 구분자로 문자열 나누기



예제는 ,(콤마)로 구분되어 있는 한 줄의 데이터를
콤마로 구분하여 나누어 줌

```
rec="홍길동,100,95,88"
```

```
item=rec.split(",")
```

```
print("="*10)
```

```
print("이름: ", item[0])
```

```
print("국어: ", item[1])
```

```
print("영어: ", item[2])
```

```
print("수학: ", item[3])
```

```
print("="*10)
```

```
>>>
```

```
RESTART: ***.py
```

```
=====
```

```
이름: 홍길동
```

```
국어: 100
```

```
영어: 95
```

```
수학: 88
```

```
=====
```

5) 구분자로 문자열 나누기



예제는 빈칸으로 구분되어 있는 한 줄의 데이터를
빈칸으로 구분하여 나누어 줌



하지만 빈칸이 여러 개 있을 때 해당 빈칸 하나 하나를
데이터로 인식함

5) 구분자로 문자열 나누기

```
rec="본 예제는 빈칸으로 구분되어 있는 한 줄의 데이터를 빈칸을  구분하여 나누어 줌"
```

```
item=rec.split(" ")

print("="*10)
i=0
for s in item:
    print(i,"==>", s)
    i+=1
print("="*10)
```

공백(빈칸)이 5개

```
>>>
RESTART: ***.py
=====
0 ==> 본
1 ==> 예제는
2 ==> 빈칸으로
3 ==> 구분되어
4 ==> 있는
5 ==> 한
6 ==> 줄의
7 ==> 데이터를
8 ==> 빈칸을
9 ==>  
10 ==>
11 ==>
12 ==>
13 ==>
14 ==> 구분하여
15 ==> 나누어
16 ==> 줌
=====
```

6) 구분자 연습



앞에서 실습한 트와이스 학급 성적표 출력 계속



해당 데이터가 리스트에서 다음과 같은 콤마 구분
데이터의 리스트인 경우 기존 작성된 프로그램을 변경할 것

이전 데이터

```
score=[["나연",100,90,100],  
        ["정연",90,90,100],  
        ["모모",80,70,90],  
        ["사나",90,90,80],  
        ["지효",100,80,80],  
        ["미나",50,90,90],  
        ["다현",80,60,100],  
        ["채영",70,80,90],  
        ["쯔위",100,90,90]]
```

이번 실습 데이터

```
dili_score=["나연,100,90,100",  
            "정연,90,90,100",  
            "모모,80,70,90",  
            "사나,90,90,80",  
            "지효,100,80,80",  
            "미나,50,90,90",  
            "다현,80,60,100",  
            "채영,70,80,90",  
            "쯔위,100,90,90"]
```

6) 구분자 연습

[콤마 구분자로 나눔, 숫자형 변환, 리스트에 추가]를 유의하여 프로그래밍 함

```
dili_score=["나연,100,90,100",
            "정연,90,90,100",
            "모모,80,70,90",
            "사나,90,90,80",
            "지효,100,80,80",
            "미나,50,90,90",
            "다현,80,60,100",
            "채영,70,80,90",
            "쯔위,100,90,90"]

score=[]
for one_rec in dili_score:
    one_score=one_rec.split(",")
    one_score[1]=int(one_score[1]) #국어점수 숫자형 변환
    one_score[2]=int(one_score[2]) #영어점수 숫자형 변환
    one_score[3]=int(one_score[3]) #수학점수 숫자형 변환
    score.append(one_score.copy()) # copy에 주의할 것..값 복사

print(score)
```

6) 구분자 연습

[콤마 구분자로 나눔, 숫자형 변환, 리스트에 추가]를 유의하여 프로그래밍 함



```
>>>
RESTART:**.py
[['나연', 100, 90, 100], ['정연', 90, 90, 100], ['모모', 80, 70, 90],
['사나', 90, 90, 80], ['지효', 100, 80, 80], ['미나', 50, 90, 90],
['다현', 80, 60, 100], ['채영', 70, 80, 90], ['쯔위', 100, 90, 90]]
```

Tip

- CSV파일 : CSV(comma-separated values)는 몇 가지 필드를 쉼표(,)로 구분한 텍스트 데이터 및 텍스트 파일을 의미
- 많은 실무 데이터가 형태로 CSV를 사용하고 있으며, 또한 JSON등 통신처리에서도 이러한 방식이 많이 사용되므로 구분자 포맷을 처리하는 방법은 매우 중요한 사항



03

포맷팅

- 1) 포맷팅 정의
- 2) 포맷팅 문자표현
- 3) 숫자에 콤마 찍기
- 4) 포맷팅 실습

1) 포매팅 정의

포매팅 (Formatting)

출력 형식을 지시하여 출력하는 방식

1) 포매팅 정의

 다음 예제를 먼저 무작정 실습함

```
item="오징어땅콩"    #품목
unit=1000            #단가
qty=10               #수량
tot=unit*qty         #합계

print("case1") #지금까지 사용한 print방법, 칸이 띄어서 출력
print("품목:",item," 단가:",unit,"원 수량:",qty,"개 합계:",tot,"원")

print("case2") #하나의 긴 문자열을 만들어 출력하면 공백없이 출력이
가능
print("품목:"+item+" 단가:"+str(unit)+"원 수량:"+str(qty)+"개
합계:"+str(tot)+"원")

print("case3") #출력포맷문자를 지정하여 원하는 포맷으로 출력이 가능
print("품목:%s 단가:%d원 수량:%d개 합계:%d원"%(item,unit,qty,tot))

print("case4") #포매팅 예
print("품목:%-10s 단가:%05d원 수량:0x%02x개
합계:%10.3f원"%(item,unit,qty,tot))
```

1) 포매팅 정의

 다음 예제를 먼저 무작정 실습함



```
>>>
  RESTART: ***.py
case1
품목: 오징어땅콩   단가: 1000 원   수량: 10 개   합계: 10000 원
case2
품목:오징어땅콩   단가:1000원   수량:10개   합계:10000원
case3
품목:오징어땅콩   단가:1000원   수량:10개   합계:10000원
case4
품목:오징어땅콩           단가:01000원   수량:0x0a개   합계: 10000.000원
```

2) 포매팅 문자표현

표시

print("포매팅 문자포함 나열"%(대응 순서별 변수나열))

구분	대표표현	사용 예
정수표현	%d	<ul style="list-style-type: none">▪ %5d : 5칸으로 숫자표시하며 오른쪽으로 정렬하고 앞은 공백으로 채움▪ %05d: 5칸으로 숫자를 표시하고 앞에서 부터 0채움▪ %-5d: 5칸으로 숫자를 표현,단 왼쪽부터 채우고 넘어가면 잘림
실수표현	%f (정수부와 소수 부) %e (지수형표시)	<ul style="list-style-type: none">▪ %10.2f:10칸으로 숫자를 표시하고 소숫점이하 2자리를 출력, 소숫점도 한자리를 차지함▪ %e : 1.00000e+04형식으로 나오며 안에 숫자를 기입

〈1/2〉

2) 포매팅 문자표현

표시

print("포매팅 문자포함 나열"%(대응 순서별 변수나열))

구분	대표표현	사용 예
문자열표현	%s	<ul style="list-style-type: none">▪ %5s : 5칸으로 문자표현 ,오른쪽으로 정렬하며 앞은 공백으로 채움▪ %-5s: 5칸으로 문자표현, 왼쪽으로 정렬
문자하나표현	%c	
진법표현	%x (16진수,%#x) %o (8진수)	<ul style="list-style-type: none">▪ %x : 단순한 16진수 표현▪ %#x : 0xb 와 같은 형식으로 표현
기타	%%	<ul style="list-style-type: none">▪ 55% 와 같이 %기호를 출력하려면 %%를 사용함

〈2/2〉

2) 포매팅 문자표현

포매팅 예제

```
item="땅콩"    #문자
unit=1000      #숫자

print("[%5d][%-5d][%05d]"%(unit,unit,unit))
print("[%5s][%-5s][%05s]"%(item,item,item))
print("[%d]=[%x][%#x][%o]"%(unit,unit,unit,unit))
print("[%5d%]"%unit)

#한글은 1byte로 자르면 에러
#print("[%c][%c][%c]"%(unit[0],unit[1]))
a="abcd"
print("[%c][%c][%c]"%(a[0],a[1],a[2]))
```

```
>>>
RESTART: ***.py
[ 1000][1000 ][01000]
[   땅콩][땅콩   ][   땅콩]
[1000]=[3e8][0x3e8][1750]
[ 1000%]
[a][b][c]
```

3) 숫자에 콤마 찍기



숫자 (특히 돈)에 관련된 경우 3자리 마다 콤마를 찍어 표현


└→ 100000원 → 100,000원

3) 숫자에 콤마 찍기



이런 경우 다음 예제를 사용함

```
import locale #locale라이브러리를 임포트 함, 설치는 필요없음
locale.setlocale(locale.LC_ALL, "") # 시스템 기본 로케일 사용
n = -1234567890.123
s = locale.format_string("%.3f", n, 1)
print(s)
```



```
>>>
RESTART: ***.py
[ 1000][1000 ][01000]
[   땅콩][땅콩   ][   땅콩]
[1000]=[3e8][0x3e8][1750]
[ 1000%]
[a][b][c]
```

4) 포매팅 실습

 5강에서 환율계산 문제에 대한 결과 출력을 깔끔하게 처리 할 것

환전 문제

- 만일 100만 원(1000000)을 달러로 환전한다고 하자
- 미화 환율은 1달러당 1010.12원이고
은행은 0.002%의 수수료를 환전수수료로 받는다고 하였을 때
- 100만 원에 대하여 받는 달러금액, 은행수수료,
한화 거스름돈을 구하시오.

4 포매팅 실습

코딩 예제 (계산은 동일함)

이번에는 소스 안지우고 보여주니 감을 느끼시오.

```
import locale #locale라이브러리를 임포트 함, 설치는 필요없음
locale.setlocale(locale.LC_ALL, "") # 시스템 기본 로케일 사용

myWon=1000000 #100만원
moneyEx=1010.12 # 달러환율
commission_rate=0.002 # 은행 수수료 율
a = moneyEx + moneyEx*commission_rate #1달러당 환율금액 과 해당금액에 대한 수수료의 합
usd = int (myWon /a) # 나의 돈을 위의 금액으로 나오면 소수점이 나오기 때문에
                    # 해당 소수점을 버린다 즉 100만원에 989.99달러 나오면
                    #989달러를 준다. 소수점 아래는 버린다
remain = int(myWon - usd * a) # 거스름돈 계산 =100만원 - 달러액*(환율+달러당 수수료)
                    # 소수점 아래는 버린다.
commission = usd * moneyEx * commission_rate # 환전 수수료총액 계산,
                    #이 금액은 소수점이하 올림처리해 보자
if(commission != float(int(commission))) : # 이 비교문은 소수점이 있는지 체크
    commission = int(commission) +1 #소수점이 있다면 올려서 은행이 처리
else:
    commission = int(commission) #소수점 아래 숫자가 없다면 단순 정수형 처리
```

-- 뒷장 계속 --

4) 포매팅 실습

코딩 예제 (계산은 동일함)

```
print("*****")
print("총금액: %s원 \t달러환율: %s"%( \
    locale.format_string("%12d", myWon, 1), \
    locale.format_string("%-12d", moneyEx, 1)) )

print("지급달러: %s"% locale.format_string("%-12d", usd, 1))
print("거스름돈: %s원 \t은행수수료: %s원"%( \
    locale.format_string("%6d", remain, 1), \
    locale.format_string("%6d", commission, 1)) )
print("*****")
```

4) 포매팅 실습

코딩 예제 (계산은 동일함)



```
>>>
RESTART: ***.py
*****
총금액:      1,000,000원      달러환율: $1,010
지급달러: $988
거스름돈:      5원   은행수수료:  1,996원
*****
```

04

실습하기 I

- 1) 문자열 다루기 - 정의, 문자열 다루기
- 2) 문자열 함수 - 찾기, 검사, 대소문자 바꾸기
- 3) 문자열 함수 - 공백제거, 구분자 처리

실습내용

- 1) 문자열 다루기 - 정의, 문자열 다루기
- 2) 문자열 함수 - 찾기, 검사, 대소문자 바꾸기
- 3) 문자열 함수 - 공백제거, 구분자 처리

05

실습하기 II

- 1) 문자열 함수 - 구분자 연습
- 2) 포매팅 - 정의, 포매팅 문자표현
- 3) 포매팅 - 포매팅 연습

실습내용

- 1) 문자열 함수 - 구분자 연습
- 2) 포매팅 - 정의, 포매팅 문자표현
- 3) 포매팅 - 포매팅 연습



학습활동

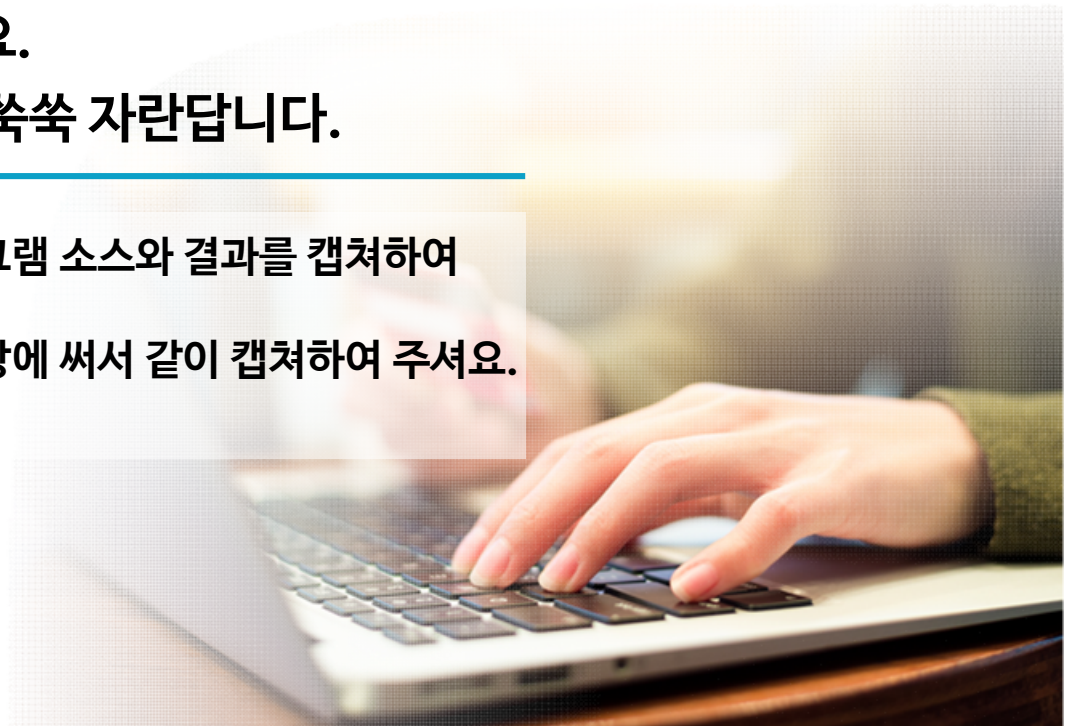
일시정지 버튼을 누른 후, 아래의 학습활동에 참여하세요.

Q

오늘 배운 내용을 스스로 실습하여
자유게시판에 올려 주세요.

이렇게 정리하면 실력이 쑥쑥 자란답니다.

- ① 본인이 실습한 내용을 프로그램 소스와 결과를 캡처하여 올려주세요.
- ② 본인의 학번과 이름을 메모장에 써서 같이 캡처하여 주세요.
- ③ 그리고 설명도 달아 주세요.





학습활동에 대한 교수님 의견

Q

오늘 배운 내용을 스스로 실습하여 자유게시판에 올려 주세요.
이렇게 정리하면 실력이 쑥쑥 자란답니다.

A

[오늘 학습한 내용의 실습 사항]

- ① 문자열 다루기 - 정의, 문자열 다루기
- ② 문자열 함수 - 찾기, 검사, 대소문자 바꾸기
- ③ 문자열 함수 - 공백제거, 구분자 처리
- ④ 문자열 함수 - 구분자 연습
- ⑤ 포매팅 - 정의, 포매팅 문자표현
- ⑥ 포매팅 - 포매팅 연습

학습 평가

Q1

Q2

Q3

Q4

Q5

Q1

다음 a 를 $a = \text{"abcde"}$ 로 정의하였을 때
잘못된 내용은?

- 1 $a[0]$ 은 a 이다.
- 2 $a[0:1]$ 은 ab 이다.
- 3 $a[-2]$ 는 b 이다.
- 4 $a[5]$ 는 정의될 수 없는 오류이다.

학습 평가

Q1

Q2

Q3

Q4

Q5

Q1

다음 a 를 $a = \text{"abcde"}$ 로 정의하였을 때
잘못된 내용은?

1 $a[0]$ 은 a 이다.

☒ $a[0:1]$ 은 ab 이다.

3 $a[-2]$ 는 b 이다.

4 $a[5]$ 는 정의될 수 없는 오류이다.

정답

2번

해설

$a[0:-1]$ 은 첨자 기준으로 0부터 1까지(1은 제외)로
 a 입니다.

학습 평가

Q1

Q2

Q3

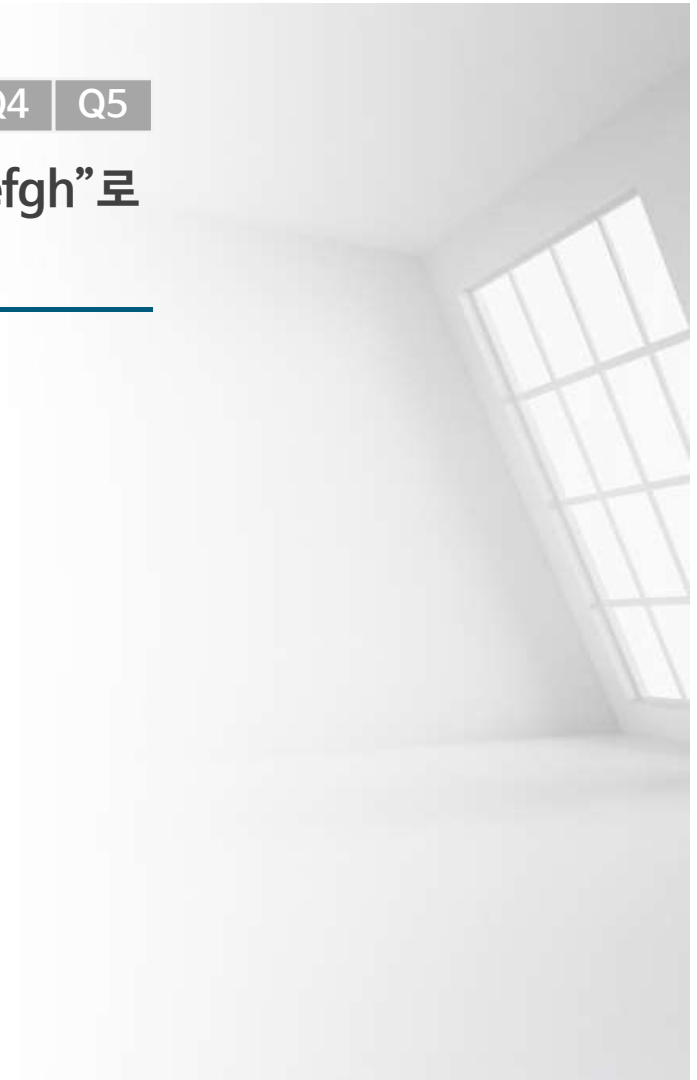
Q4

Q5

Q2

다음 a를 a="abcdefghijklmabcdefgh"로 정의하였을 때 **잘못된** 내용은?

- 1 a.find("f")는 5이다.
- 2 a.rfind("f")는 18이다.
- 3 a.find("z")는 에러가 발생한다.
- 4 a.index("f")는 5이다.



학습 평가

Q1

Q2

Q3

Q4

Q5

Q2

다음 a를 a="abcdefghijklmabcdefgh"로 정의하였을 때 **잘못된** 내용은?

- 1 a.find("f")는 5이다.
- 2 a.rfind("f")는 18이다.
- ☒ 3 a.find("z")는 에러가 발생한다.
- 4 a.index("f")는 5이다.

정답

3번

해설

find 함수에서 문자열을 못 찾는 경우 -1 을 리턴합니다.

학습 평가

Q3

Q1

Q2

Q3

Q4

Q5

다음 문장 수행 이후 설명 중 **잘못된** 것은?

```
rec="홍길동,100,95,88";  
item = rec.split(",")
```

- 1 split함수의 구분자는 ,(comma)이다.
- 2 item[1]에 홍길동이 들어있다.
- 3 item에는 콜렉션으로 값이 들어간다.
- 4 구분자는 문자열로도 사용이 가능하다.

학습 평가

Q1

Q2

Q3

Q4

Q5

Q3

다음 문장 수행 이후 설명 중 **잘못된** 것은?

```
rec="홍길동,100,95,88";
```

```
item = rec.split(",")
```

- 1 split함수의 구분자는 ,(comma)이다.
- ☒ 2 item[1]에 홍길동이 들어있다.
- 3 item에는 콜렉션으로 값이 들어간다.
- 4 구분자는 문자열로도 사용이 가능하다.

정답

2번

해설

item[1]은 100이라는 문자열이 들어갑니다.

학습 평가

Q1

Q2

Q3

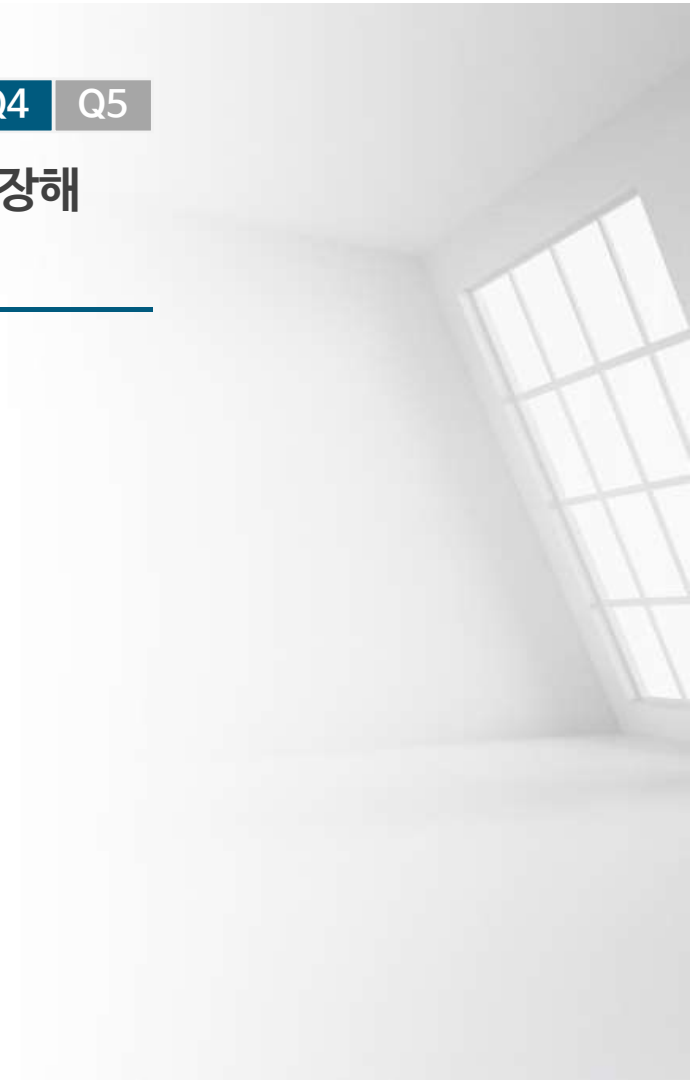
Q4

Q5

Q4

다음 중 콤마를 구분자로 데이터를 저장해
놓은 파일을 의미하는 용어는?

- 1 XML파일
- 2 HTML파일
- 3 JSON파일
- 4 CSV파일



학습 평가

Q1

Q2

Q3

Q4

Q5

Q4

다음 중 콤마를 구분자로 데이터를 저장해 놓은 파일을 의미하는 용어는?

- 1 XML파일
- 2 HTML파일
- 3 JSON파일
- ☒ 4 CSV파일

정답

4번

해설

집합연산의 CSV(Comma-Separated Values)는 몇 가지 필드를 쉼표(,)로 구분한 텍스트 데이터 및 텍스트 파일을 의미합니다.

학습 평가

Q1

Q2

Q3

Q4

Q5

Q5

다음 `print("[% -5d]"%256)`으로
작성하였을 때 올바른 출력결과는?
(단, b는 빈칸 출력을 의미함)

1 [bb256]

2 [256bb]

3 [b-256]

4 [-256b]



학습 평가

Q1

Q2

Q3

Q4

Q5

Q5

다음 `print("[%5d]"%256)`으로
작성하였을 때 올바른 출력결과는?
(단, b는 빈칸 출력을 의미함)

1 [bb256]

☒ 2 [256bb]

3 [b-256]

4 [-256b]

정답

2번

해설

`[%-5d]`는 왼쪽부터 숫자를 출력하라는
포맷팅 문자입니다.



정리하기

문자열 다루기

- ✓ 문자형 변수의 성격은 컬렉션의 리스트와 유사함
- ✓ 문자 첨자 처리시 [시작:끝] 으로 표시가 가능함
(끝에 해당되는 숫자 앞까지 해당됨에 유의)





정리하기

문자열 함수

- ✓ 문자열 함수로 찾기, 검사, 대소문자 바꾸기, 공백제거, 구분자로 문자열 나누기 등이 가능함
- ✓ `A=str.split(구분자)` : str의 문자열을 구분자 문자를 찾아서 하나씩 나누어 a에 리스트로 반환함
- ✓ 많은 실무 데이터가 형태로 CSV를 사용하고 있으며, 또한 JSON 등 통신처리에서도 이러한 방식이 많이 사용되므로 구분자 포맷을 처리하는 방법은 매우 중요한 사항임





정리하기

포매팅

- ✓ 포매팅(Formatting)은 출력 형식을 지시하여 출력하는 방식을 의미함
- ✓ 포매팅 문자표현은 `print("포매팅 문자포함 나열"%(대응 순서별 변수나열))`으로 표시함

