문자열 처리

UPDATE : 2018.06

Contents

- 문자열 자료형 선언
- 문자열 연산자
- 문자열 인덱싱
- 문자열 슬라이싱
- 문자열 포맷팅 - %
- 문자열 포맷팅 – {}.format
- 공백 처리
- 문자 개수 반환
- 문자 값 반환
- 문자열 조작
- 문자열 자료형의 리스트화

문자열 자료형 선언

문자열 자료형 만드는 4가지 방법

```
"Hello World"

'Python is fun'

"""Life is too short, You need python"""

'''Life is too short, You need python'''
```

문자열 자료형 선언

문자열에 따옴표 포함시키기

```
>>> food = "Python's favorite food is perl"

>>> say = '"Python is very easy." he says.'

>>> food = 'Python\'s favorite food is perl'

>>> say = "\"Python is very easy.\" he says."
```

문자열 자료형 선언

여러 줄로 이루어진 문자열

```
>>> multiline = "Life is too short\nYou need python"
```

```
>>> multiline='''
... Life is too short
... You need python
... '''
```

문자열 연산자

문자열 더해서 연결하기 (Concatenation) : + 이용

```
>>> head = "Python"
>>> tail = " is fun!"
>>> head + tail
'Python is fun!'
```

문자열 곱하기 : * 이용

```
>>> a = "python"
>>> a * 2
'pythonpython'
```

문자열 인덱싱

인덱싱(Indexing)

```
>>> a = "Life is too short, You need Python"
>>> a[0]
'L'
>>> a[12]
's'
>>> a[-1]
'n'
```

파이썬은 0부터 숫자를 센다

문자열 슬라이싱

슬라이싱(Slicing)

- 문자열변수[start:end]
- 문자열변수[:end]
- 문자열변수[start:]

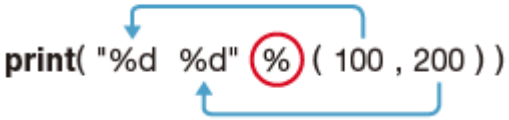
```
>>> a = "Life is too short, You need
Python"
>>> a[0:4]
'Life'
```

```
>>> a = "20010331Rainy"
>>> date = a[:8]
>>> weather = a[8:]
>>> date
'20010331'
>>> weather
'Rainy'
```

문자열 포맷팅 1

- %서식 이용

```
print( " %style1 %style2 "  % (value1, vlue2) )
```



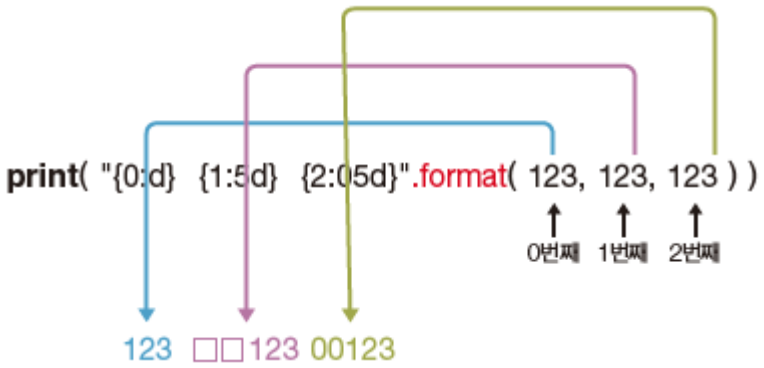
서식	값의 예	설명
%d, %x, %o	10, 100, 1234	정수(10진수, 16진수, 8진수)
%f	0.5, 1.0, 3.14	실수(소수점이 붙은 수)
%c	"b", "한"	한 글자
%s	"안녕", "abcdefg", "a"	두 글자 이상인 문자열

% 서식과 함께 퍼센트(%)를 표시하고 싶다면?
>>>print('%.2f 퍼센트(%%)'%(34.5678))
34.57 퍼센트(%)

문자열 포맷팅 2

- { } .format()

```
print( “ {index1:style1} {index2:style2} ” .format(value1, vlue2))
```



Index를 이용하면 출력 순서를 설정할 수 있다.

```
print("{2:d} {1:d} {0:d}".format(100, 200, 300))
```

문자열 포맷팅 2

- { } .format() – 변수 이용하기

```
print( “ {variable1} {variable2} ” .format(variable1=value1, variable2=value2))
```

>>> print(‘value1 : { a } , value2 : { b }’.format(a=‘dog’, b=20))

value1 : dog , value2 : 20

이스케이프 문자

이스케이프 문자	역할	설명
\n	새로운 줄로 이동	Enter를 누른 효과
\t	다음 탭으로 이동	Tab을 누른 효과
\b	뒤로 한 칸 이동	Backspace를 누른 효과
\\	\ 출력	
\'	' 출력	
\"	" 출력	

```
1 print("\n줄바꿈\n연습 ")
2 print("\t탭키\t연습")
3 print("글자가 \"강조\"되는 효과1")
4 print("글자가 \'강조\'되는 효과2")
5 print("\\\\\\ 역슬래시 세 개 출력")
6 print(r"\n \t \" \\를 그대로 출력")
```

문자열 개수 반환

문자열 개수 세기(count) – 문자열변수.count(string)

```
>>> a = "hobby"
>>> a.count('b')
2
```

문자 위치 반환

위치 알려주기1(find)

```
>>> a = "Python is best choice"
>>> a.find('b')
10
>>> a.find('k')
-1
```

문자가 처음 나온 위치 반환.
없다면 -1 반환

문자 위치 반환

위치 알려주기2(index)

```
>>> a = "Life is too short"
>>> a.index('t')
8
>>> a.index('k')
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
ValueError: substring not found
```

문자가 처음 나온 위치 반환.
없다면 오류발생

문자열 조작

문자열 삽입(join)

```
>>> a= ","
>>> a.join('abcd')
'a,b,c,d'
```

문자열 각각에 a 변수의 문자 , 를 삽입한다.

문자열 조작

대문자를 소문자로 바꾸기(lower)

```
>>> a = "HI"
>>> a.lower()
'hi'
```

소문자를 대문자로 바꾸기(upper)

```
>>> a = "hi"
>>> a.upper()
'HI'
```

문자열 조작

양쪽 공백 지우기(strip)

```
>>> a = " hi "
>>> a.strip()
'hi'
```

왼쪽 공백 지우기(lstrip)

오른쪽 공백 지우기(rstrip)

문자열 조작

문자열 바꾸기(replace)

```
>>> a = "Life is too short"
>>> a.replace("Life", "Your leg")
'Your leg is too short'
```

문자열 자료형의 리스트화

문자열 나누기(split)

- 문자열변수.split()
- 문자열변수.split(구분자)

```
>>> a = "Life is too short"
>>> a.split()
['Life', 'is', 'too', 'short']
>>> a = "a:b:c:d"
>>> a.split(':')
['a', 'b', 'c', 'd’]
```

공백 처리

%양수숫자s(왼쪽 공백)

%음수숫자s(오른쪽 여백)

```
>>> "%10s" % "hi"
'        hi'
>>> "%-10sjane." % 'hi'
'hi        jane.'
```

%-숫자 로 지정시 오른쪽에 여백생성이 되면서 오른쪽 정렬된다.