

2. 문자열 자료형

- 문자열 : 문자, 단어 등으로 구성된 문자들의 집합
- 파이썬에서 모든 단어는 큰 따옴표(“)로 둘러싸여있으면 문자열로 본다

2.1. 문자열을 만들고 사용하는 방법

기본적으로 만드는 방법

- 큰 따옴표로 둘러싸기
- 작은따옴표로 둘러싸기
- 큰 따옴표 3개를 연속으로 써서 둘러싸기
 - “””
- 작은 따옴표 3개를 연속으로 써서 둘러싸기
 - '''

왜 그렇게 4개나 될까?

문자열 안에 작은 따옴표나 큰따옴표를 포함시키고 싶은 경우

- 작은 따옴표가 문자열 내에 포함될 경우
 - 큰 따옴표로 둘러싼다
- 큰 따옴표가 문자열 내에 포함될 경우
 - 작은 따옴표로 둘러싼다
- 역슬래시 사용 \

```
print('Python\'s favorite food is perl')
```

여러줄인 문자열 변수에 대입

- 줄을 바꾸기 위한 \n (이스케이프 코드)삽입
- 연속된 작은 따옴표 3개, 큰 따옴표 3개 사용

2.2. 문자열 연산하기

문자열 더해서 연결

```
a = "hi, "  
b = "python!"
```

```
print(a+b)
```

hi, python!

문자열 곱하기

```
a = "hi,"  
b = "python!"  
print(a+b*2)
```

hi,python!python!

곱하는 숫자만큼 여러번 반복하라는 의미

2.3. 문자열 길이 구하기

len 함수를 이용하기

```
a = "hi,"  
b = "python!"  
print(len(a+b))
```

문자열에 길이에는 공백, 특수기호도 포함된다

2.4. 문자열 인덱싱과 슬라이싱

- 인덱싱 : 무엇인가를 가리킨다
- 슬라이싱 : 무엇인가를 잘라낸다
- a = "Life is too short, You need Python"

위 코드에서 변수 a 에 저장한 문자열의 각 문자마다 번호를 매겨 보면 다음과 같다.

L	i	f	e		i	s		t	o	o		s	h	o	r	t	,		Y	o	u		n	e	e	d		P	y	t	h	o	n
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33

a[3] 은 e가 된다

- 파이썬은 0부터 숫자를 센다
- 특수문자, 공백도 인덱스 번호를 차지한다

문자열 인덱싱 활용하기

```
a = "hi nahee, i'm Sheldon Cooper"
print(a[0])
```

결과 h

```
a = "hi nahee, i'm Sheldon Cooper"
print(a[-2])
```

-는 뒤에서 부터 읽는 것을 말한다

결과는 e

문자열 슬라이싱

내가 원하는 부분을 빼서 더 할 수 있다

```
a = "hi nahee, i'm Sheldon Cooper"
print(a[0]+a[1]+a[2]+a[14]+a[15]+a[16]+a[17]+a[18]+a[19]+a[20]+a[21])
```

결과는 hi Sheldon

```
a = "lemon"
print(a[0:3])
```

결과는 lem

왜 o는 나오지 않을까?

수식으로 나타내면 $0 \leq a < 3$ 이기 때문이다

- a [시작번호 : 끝번호]
 - 끝번호 생략시 시작부터 그 문자열 끝까지
 - 시작번호 생략시 시작부터 끝번호까지
 - 전부 생략시 처음부터 끝까지
 - • 사용가능 : 뒤에서 부터 읽는다

2.5. 문자열 포매팅

- 문자열 안에 특정한 값을 바꿔야 할 경우가 있을 때 이것을 가능하게 해주는 것
 - 나희는 1학년 2학기때 학점이 3.9 였다

- 나희는 2학년 1학기때 학점이 0이었다

문자열 포매팅 방법

- 숫자 바로 대입

```
a = "I eat the %d cake" % 2
print (a)
```

I eat the 2 cake

- 문자열 대입

```
a = "I eat the %s cake" % "red"
print (a)
```

I eat the red cake

- 숫자 값을 나타내는 변수로 대입

```
red = 2
a = "I eat the %s cake" % red
print (a)
```

I eat the 2 cake

- 2개 이상의 값 넣기

```
a = "I eat the %s %d cake" % ("red", 2)
print (a)
```

I eat the red 2 cake

2.6. 문자열 포맷 코드

코드	설명
%s	문자열 (string)
%c	문자 1개 (character)
%d	정수
%f	부동소수
%o	8진수
%x	16진수
%%	문자 % 자체

- %s의 경우 어떤 형태의 값이든 변환해서 넣을 수 있다
- %를 나타내려는 경우 %%와 같이 써야 한다

2.7. 포맷코드와 숫자 함께 사용하기

- 포맷코드를 숫자와 함께 사용하면 더 유용하다

정렬과 공백

```
print ("%10s" % "hi")
```

A terminal window showing the output of the command. The text 'hi' is displayed in a monospaced font, right-aligned within a field of 10 characters. The first 7 characters are empty space, and the last 3 characters are 'hi'.

%10s

전체길이가 10개인 문자열 공간에서 대입되는 값을 오른쪽으로 정렬한다

그 앞에 나머지는 공백으로 남겨둔다

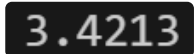
A terminal window showing the output of the command. The text 'hi' is displayed in a monospaced font, left-aligned within a field of 10 characters. The first 3 characters are 'hi', and the remaining 7 characters are empty space.

-10%s

이건 왼쪽정렬이 된다

소수점 표현하기

```
print ("%0.4f" % 3.42134234)
```

A terminal window showing the output of the command. The number '3.4213' is displayed in a monospaced font. The first 4 characters are '3.421', and the last 3 characters are '3'.

%0.4f 이나 %.4f

소수점 4번째 자리까지만 나타내고 싶은 경우

4 앞에 0은 소수점 포인트 앞에 있는 정수의 길이는 상관하지 않겠다는 의미이다

소숫점도 인덱스를 차지한다!

```
print ("%30.4f" % 23523513.42134234)
```

A terminal window showing the output of the command. The number '23523513.4213' is displayed in a monospaced font, right-aligned within a field of 30 characters. The first 17 characters are empty space, and the last 13 characters are '23523513.4213'.

2.8. format 함수를 사용한 포매팅

문자열 대입

```
a = "hello my name is {0}" .format("hee")  
print(a)
```

hello my name is hee

2개 이상의 값 넣기

```
a = "hello my name is {0}, and my age is {1}" .format("hee", 26)  
print(a)
```

hello my name is hee, and my age is 26

{0}은 첫번째 두번째 넣을 곳은 {1} 로 바뀌게 된다

이름으로 넣기

```
"I ate {number} apples. so I was sick for {day} days." .format(number=10, day =3)
```

'I ate 10 apples. so I was sick for 3 days.'

인덱스와 이름을 혼용해서 넣기

```
"I ate {0} apples. so I was sick for {day} days." .format(10, day=3)
```

'I ate 10 apples. so I was sick for 3 days.'

정렬

왼쪽 정렬

```
"{0:<10}".format("hi")
```

- :<10 사용
치환되는 문자열을 왼쪽으로 정렬하고 문자열의 총 자릿수를 10 으로 맞출 수 있다

오른쪽 정렬

```
"{0:>10}".format("hi")
```

- 오른쪽 정렬은 :< 대신 :>을 사용한다

가운데 정렬

```
"{0:^10}".format("hi")
```

- ^ 이용

공백 채우기

```
"{0:=^10}".format("hi")
```

정렬 시 공백 문자 대신 지정한 문자 값으로 채워 넣을 수 있다

소수점 표현하기

```
y = 3.42134234  
"{0:0.4f}".format(y)
```

'3.4213'

format 함수를 이용해 소수점을 4자리까지만 표현한다

```
"{0:10.4f}".format(y)
```

' 3.4213'

자릿수를 10으로 맞출 수도 있다

{또는} 문자 표현하기

```
"{{ and }}".format()
```

{ and }

{} 같은 문자를 나타내고 싶은 경우
{}}처럼 2 개를 연속해서 사용하면 된다

2.9. f 문자열 포매팅

*파이썬 3.6 버전 이상에서만 사용 가능

```
name = '오나희'
age = 30

f'내 이름은 {name}이다. 나이는 {age} 이다.'
```

- 문자열 앞에 f 접두사를 붙이면 f 문자열 포매팅 기능을 사용할 수 있다

2.10. 문자열 관련 함수들

- 문자열 자료형은 자체적으로 함수를 가지고 있다 = 문자열 내장 함수
- 사용 시 문자열 변수 이름 뒤에 . 을 붙인 후 함수 이름을 써주면 된다

문자열 개수 세기 count

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

위치 알려주기

find

```
a = "hi TV"
print(a.find('T'))
```


3

index

```
a = "hi TV"
print(a.index('h'))
```

0

만약 찾는 문자나 문자열이 존재하지 않는다면 오류가 발생한다

문자열 삽입 join

```
print(".".join("helloTV"))
```

h.e.l.l.o.T.V

소문자를 대문자로 바꾸기 upper

```
a = "hi"
print(a.upper())
```

HI

- 이미 문자열이 대문자라면 아무런 변화도 일어나지 않는다
- 변수 값 자체를 바꾸는 것은 아님
 - 대문자로 바꾼 값을 리턴하는 것임

대문자를 소문자로 바꾸기 lower

```
a = "HI"
print(a.lower())
```

hi

- 변수 값 자체를 바꾸는 것은 아님
 - 대문자로 바꾼 값을 리턴하는 것임

왼쪽 공백 지우기 lstrip

```
a = " hi "  
print(a.lstrip())
```

'hi '

오른쪽 공백 지우기 rstrip

```
a = " hi "  
print(a.rstrip())
```

'hi'

양쪽 공백 지우기 strip

```
a = " hi "  
print(a.strip())
```

'hi'

문자열 바꾸기 replace

```
a = "어쩔티비 저쩔티비"  
print(a.replace("티비", "냉장고"))
```

어쩔냉장고 저쩔냉장고

문자열 나누기 split

```
a = "어쩔티비 저쩔티비"  
print(a.split())
```

['어쩔티비', '저쩔티비']