

(1) 기초 자료형, 입출력, 연산자

송기태 (kitae040522@gmail.com)

Soongsil Univ. (Computer Science and Engineering)

Content

기본 입출력 함수

기초 자료형 (int, str, float ...)

변수

연산자

```
print($[something], sep=' ', end='\n')
```

- `$(something)`을 ' '으로 구분하고, 끝에 `\n`을 추가하여 출력하는 함수이다.
- 이때, `sep`과 `end`는 `print` 함수의 옵션으로 넣어줘도 되고 안 넣어줘도 된다.
- 다만, 값을 안 넣어줬다면 default 값이 들어간다.
- `$(something)`의 갯수는 몇개가 되든지 맘껏 넣어도 된다.

`input($[something])`

- `$[something]`을 출력하고, 입력받은 값을 문자열로 리턴하는 함수이다.

숫자형

파이썬은 아래와 같은 4가지의 숫자형 데이터를 핸들링할 수 있다.

- 정수 (ex. 123, -123, 0, ...)
- 실수 (ex. 123.45, 3.14, -123.0, 1.e-2)
- 8진수, 앞에 0o를 입력하고 숫자를 입력하면 8진수로 인식한다. (ex. 0o34, 0o25)
- 16진수, 앞에 0x를 입력하고 숫자를 입력하면 16진수로 인식한다. (ex. 0xAB, 0xff)

문자열형

파이썬은 문자 하나를 다루는 자료형은 없다.

파이썬에서는 신기하게도 문자열에 `+`, `-`, `*`와 같은 연산자를 사용할 수 있는데,

예를 들어 `print("안녕하세요" + "\n제 이름은 송기태입니다.")`를 해보면

`print("안녕하세요\n제 이름은 송기태입니다.")`로 연산이 된다.

또한, `print("안녕" * 3)`를 해보면 `print("안녕안녕안녕")`로 연산이 된다.

문자열의 길이

문자열의 길이는 어떻게 하면 쉽게 구할 수 있을까? → `len($[something])` 함수를 사용하자!

`len("안녕하세요!")`의 값을 print 해보면, 6이 출력된다.

문자열의 길이에는 공백 문자도 포함되므로, 주의하도록 하자.

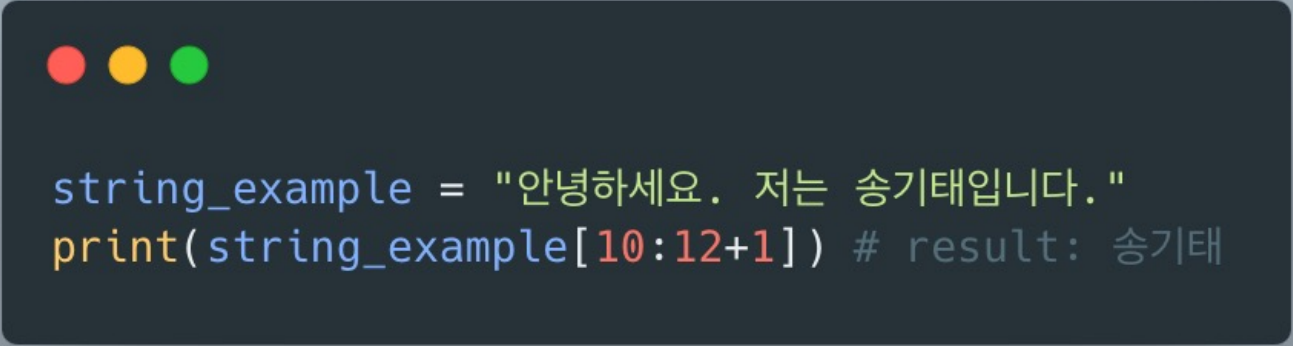
문자열 인덱싱 및 슬라이싱

또한, 문자열은 우리가 원하는 대로 잘라낼 수도 있고, 추출할 수도 있다.

문자열 뒤에 대괄호를 입력하고, 원하는 위치를 입력하면 해당하는 문자를 추출할 수 있다.

- Tip!
 - 파이썬은 0 based index이다. 즉 시작이 0부터!
 - 대괄호 안에는 정수만 입력할 수 있다는 점을 꼭 알아두자!

문자열 인덱싱 및 슬라이싱



```
string_example = "안녕하세요. 저는 송기태입니다."  
print(string_example[10:12+1]) # result: 송기태
```

알아두면 좋은 문자열 메소드

1. 위치 알려주기

"안녕하세요. 저는 송기태입니다."에서 '요'의 위치를 알고 싶을 때에는

`\$[문자열].find(\$[찾고 싶은 거])` 함수를 사용하면 된다. 문자가 존재한다면 해당하는 문자의 위치를 리턴하고, 존재하지 않으면 -1을 리턴한다.

2. 문자열 삽입, "\$[문자열 사이에 넣을 거]".join(\$[원래 문자열])

3. 문자열 나누기, \$[원래 문자열].split(\$[어떤 기준으로 문자열을 나눌 것인가?])

부울형

True와 False로 이루어져 있으며, 각각 0과 1의 값을 갖는다.

변수란 무엇일까?

데이터를 담는 상자

- 변수를 새롭게 만들기 위해서는 $\$[변수명] = \$[값]$ 의 형태로 작성하면 된다.
- 예를 들어 'x'라는 변수에 정수 10을 넣고 싶으면, $x = 10$ 으로 짜주면 된다.
- 파이썬은 동적 타이핑이기 때문에, 프로그램이 실행 될 때 변수의 타입을 검사한다.
- 따라서 변수를 선언할 때, 타입을 명시해 줄 필요가 없다.

변수명 규칙

변수명을 지을 때에는 작성 규칙이 있는데, 아래와 같다.

- 첫 문자는 알파벳이나 언더 바(_)로 시작한다.
- 첫 문자 이외에는 숫자를 사용해도 괜찮다. -> 첫 문자에 숫자를 사용할 수 없다.
- 변수 명에 공백을 삽입할 수 없다.
- 이미 파이썬 내부에서 사용되고 있는 '예약어'와 겹치는 단어는 사용할 수 없다.
- 대소문자를 구분한다.

*예약어 종류는 `import keyword; print(keyword.kwlist)`로 확인 가능

형변환 (Type Casting)

변수의 값을 다른 타입의 값으로 변경해주는 행위.

형변환 함수	설명
<code>int(\$[something])</code>	<code>\$[something]</code> 을 정수형으로 바꿈
<code>float(\$[something])</code>	<code>\$[something]</code> 을 실수형으로 바꿈
<code>str(\$[something])</code>	<code>\$[something]</code> 을 문자열로 바꿈
<code>tuple()</code> , <code>list()</code> , <code>set()</code>	나중에 설명함

산술 연산자는 둘 이상의 타입 값에 적용할 수 있는 기호로,
Overloaded Operator라고 부른다.

기호	연산자 의미	예시	결괏값
+	덧셈	$12 + 3.3$	15.3
-	뺄셈	$6 - 12$	-6
*	곱셈	$8.8 * 5$	44.0
/	나눗셈	$11 / 2$	5.5
//	몫	$7 // 5$	1
%	나머지	$8.5 \% 3.5$	1.5
**	거듭제곱	$2 ** 6$	64

연산자 결합 방식

연산식에서 같은 연산자가 연속해서 나올 경우에,
연산 순서를 왼쪽부터 취할 것인지 아니면 오른쪽에서부터 취할 것인지 결정하는 것

연산자	결합 방식
**	오른쪽 → 왼쪽
- (음수)	왼쪽 → 오른쪽
*, /, //, %	왼쪽 → 오른쪽
+, - (뺄셈)	왼쪽 → 오른쪽

연산자 우선 순위 1

연산식에서 같은 연산자가 연속해서 나올 경우에, 무엇을 먼저 계산할 것인지 결정하는 것
지금 몰라도, 후에 강의를 진행하면서 자연스럽게 알 수 있는 내용이니까 그냥 알아만 두자!

우선순위	연산자	설명
1	(연산식...), (값...), [값...], {키: 값...}, {값...}	괄호, 튜플, 리스트, 딕셔너리, 세트 생성...
2	x[인덱스], x[인덱스:인덱스], x(인수...), x.속성	리스트(튜플, 문자열) 인덱싱, 슬라이싱, 함수 호출, 속성 참조
3	**	거듭제곱
4	*, /, //, %	곱셈, 나눗셈, 몫, 나머지

연산자 우선 순위 2

연산식에서 같은 연산자가 연속해서 나올 경우에, 무엇을 먼저 계산할 것인지 결정하는 것
지금 몰라도, 후에 강의를 진행하면서 자연스럽게 알 수 있는 내용이니까 그냥 알아만 두자!

우선순위	연산자	설명
5	$+$, $-$	덧셈, 뺄셈
6	$<<$, $>>$	비트 시프트 연산
7	$\&$	비트 AND 연산
8	\wedge	비트 XOR 연산

연산자 우선 순위 3

연산식에서 같은 연산자가 연속해서 나올 경우에, 무엇을 먼저 계산할 것인지 결정하는 것
지금 몰라도, 후에 강의를 진행하면서 자연스럽게 알 수 있는 내용이니까 그냥 알아만 두자!

우선순위	연산자	설명
9		비트 OR 연산
10	in, is, <, <=, >, >=, !=, ==	포함 연산자, 객체 비교 연산자, 비교 연산자
11	not	논리 NOT 연산
12	and	논리 AND 연산

연산자 우선 순위 4

연산식에서 같은 연산자가 연속해서 나올 경우에, 무엇을 먼저 계산할 것인지 결정하는 것
지금 몰라도, 후에 강의를 진행하면서 자연스럽게 알 수 있는 내용이니까 그냥 알아만 두자!

우선순위	연산자	설명
13	or	논리 OR 연산
14	if ~ else	조건부 표현식
15	lambda	람다 표현식
16	:=	할당 표현식

복합 대입 연산자

복합 대입 연산자는 연산과 할당을 동시에 표현하는 것

연산자	의미
$a += 3$	$a = a + 3$
$a -= 3$	$a = a - 3$
$a *= 3$	$a = a * 3$
$a /= 3$	$a = a / 3$
$a //= 3$	$a = a // 3$
$a \% = 3$	$a = a \% 3$

연습 문제

숫자 5를 2로 나누었을 때 생기는 몫과 나머지를 출력해보자!

Tip!

`$(var1)`을 `$(var2)`로 나누었을 때 생기는 몫과 나머지를 동시에 구하고 싶으면,
``divmod($(var1), $(var2))`` 함수를 사용해보자!