모든 계열을 위한 기초 교육

지능형로봇프로그램밍의 이해

3 : 파이썬 기초

한양대학교 ERICA 소프트웨어학부
조상욱 교수

목차

1. 자료형이란?

한양대학교 ERICA
Education Research Industry Cluster @ Ansan

1.1 자료형의 개념

- 자료형(Data Type)이란 변수나 상수의 종류를 의미한다. 파이썬에서 사용되는 모든 값에는 특정한 타입이 있고, 타입에 따라 값들을 조합할 때 동작이 달라진다. 파이썬에 다룰 수 있는 정보의 종류(자료형)는 다양하며, 그 중에서 자주 사용하는 네가지는 3과 15 같은 값은 정수인 int(integer) 타입, 3.5와 15.0 같은 값은 소수인 float(floating point) 타입, 'abc'와 '가을' 같은 문자열 str(string) 타입 그리고 True와 False중 한 가지만 저장할 수 있는 불인 Bool(Boolean) 타입이 있다.

1	**int형 (정수형: integer)**
1.0	**float형 (실수형: floating point)**
abc	**Str형 (문자열형: string)**
True	**Bool 형 (불형: Boolean)**

1. 자료형이란?

- 표현식의 피연산자가 둘 다 float로 이루어진 표현식은 결괏값을 float로 생성한다.
- >>> 15.0 / 5.0
- 3.0
- 표현식의 피연산자가 int와 float면 파이썬은 자동으로 int를 float로 변환한다. 따라서 두 표현식이 반환하는 값은 같다.
- >>> 15.0 / 5.0
- 3.0
- >>> 15.0 / 5
- 3.0

1. 자료형이란?

한양대학교 ERICA
Education Research Industry Cluster @ Ansan

1.2 자료형의 확인하기

- 데이터의 종류를 확인하려면 바로 Type() 함수를 사용한다. 아래 결과를 보면 <class ‘int’>라고 출력되었는데 class는 일단 무시하고 int만 보면 int는 Integer의 약자로 정수라는 의미이다.

```
>>> type(100)
<class 'int'>
```

- 이번에는 실수형, 문자열형, 불형을 넣고 확인한다. 아래 결과를 보면 3.44는 실수인 float로, ‘파이썬’은 문자열인 str로, True는 불인 Bool로 출력되었다.

```
>>> type(3.44)
<class 'float'>
>>> type('파이썬')
<class 'str'>
>>> type(True)
<class 'bool'>
```

2. 연산자

2.1 산술 연산자

- 산술 연산자는 둘 이상의 타입 값에 적용할 수 있는 기호로 재정의 연산자(overloaded operator)라고 부른다. 아래 표를 보면 이러한 연산자를 다양한 값에 적용할 때 어떻게 동작하는지 알 수 있다.

기호	연산자	예	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
**	제곱	2 ** 6	64

2. 연산자

2.2 연산자 결합 법칙

- 연산식에서 같은 연산자가 연속해서 나올 경우에 연산 순서를 왼쪽부터 취할 것인지 아니면 오른쪽에서부터 취할 것인지 결정하는 것으로 아래 표를 보면 그 순서를 알 수 있다.

연산자	결합 법칙
** (제곱)	오른쪽 → 왼쪽
- (음수)	왼쪽 → 오른쪽
* (곱), / (나누기), // (몫), % (나머지)	왼쪽 → 오른쪽
+ (더하기), - (빼기)	왼쪽 → 오른쪽

2. 연산자

한양대학교 ERICA
Education Research Industry Cluster @ Ansan

2.3 연산자 우선 순위

- 표현식에서 하나 이상의 연산자가 썩여 있을 때에는 우선 순위 규칙을 따른다. 그리고 괄호는 가장 높은 우선 순위를 가짐으로 괄호 안의 수식이 먼저 계산된다.
- 예를 들어 화씨를 섭씨로 변환하기 위해서는 화씨에서 32를 뺀 후 5/9를 곱하면 섭씨가 된다.
- >>> 220 – 32 * 5 / 9
- 202.22222222222223
- 파이썬 결과는 섭씨 202.22222222222223도인데 실제로는 104.44444444444444이어야 한다. 곱셈과 나눗셈이 뺄셈보다 우선순위가 높아서 생긴 문제이다. 아래처럼 하위 표현식의 앞뒤에 괄호를 넣으면 우선순위를 바꿀 수 있다.
- >>> (220 – 32) * 5 / 9
- 104.44444444444444

2. 연산자

연산자 우선순위	연산자	설명
1	( ) [ ] { }	괄호, 리스트, 딕셔너니, 세트 등
2	**	제곱
3	+ - ~	단항 연산자
4	* / % //	산술 연산자
5	+ -	산술 연산자
6	《》	비트 시프트 연산자
7	&	비트 논리곱
8	^	비트 배타적 논리합
9	|	비트 논리합
10	< > >= <=	관계 연산자
11	== !=	동등 연산자
12	= %= /= //= -= += *= **=	대입 연산자
13	not	논리 연산자
14	and	논리 연산자
15	or	논리 연산자
16	if ~ else	비교식

2. 연산자

한양대학교 ERICA
Education Research Industry Cluster @ Ansan

2.4 비교 연산자

- 비교 연산자는 데이터 간의 비교를 통해 어느 쪽의 데이터가 더 크거나 작은지를 확인하는 연산자입니다. 비교 연산자도 기본 연산자와 같이 파이썬만이 아닌 다른 프로그래밍 언어에서도 기본으로 제공하는 연산자이다. 비교 연산이 필요한 경우에는 다음의 표와 같이 기호를 사용하여 데이터를 비교할 수 있다.

- 프로그래밍을 처음 배우는 상황이라면 **=(등호)와 ==(관계 연산자)의 사용법**이 헷갈릴 수도 있다. 두 기호는 비슷하지만 용법은 완전히 다르기 때문에 각각의 용법에 대해 확실하게 이해하고 넘어가는 것이 좋다.

  우선 등호(=)는 변수에 값을 대입할 때 사용한다. 그리고 관계 연산자(==)는 두 개의 값이 같은지를 비교할 때 사용한다.

연산자	설명
>	x는 y보다 크다
>=	x는 y와 같거나 크다
<	x는 y보다 작다
<=	x는 y와 같거나 작다
==	x는 y와 같다
!=	x와 y는 같지 않다

2. 연산자

2.4 비교 연산자

- a = 5 > 2
- print('a = ', a)

- b = 5 >= 2
- print('b = ', b)

- c = 5 < 2
- print('c = ', c)

- d = 5 <= 2
- print('d = ', d)

- e = 5 == 2
- print('e = ', e)

- f = 5 != 2
- print('f = ', f)

2. 연산자

2.5 논리 연산자

- and 연산자와 or 연산자, 그리고 not 연산자는 조금 특별한 연산자로 복합적인 조건식을 만들 때 사용하는 논리 연산자이다. 논리 연산자는 두 개 이상의 조건식을 조합하여 표현할 수 있으며, 부울 연산자라고도 한다. 사용하는 연산자는 and, or, not의 세 가지 연산자이며, 다른 프로그래밍 언어에서는 각각의 연산자를 &&, ||, ! 로 표현하기도 한다.

- **and**
- **and** 연산자는 **A와 B가 True인 경우에만 True**가 되며 그 외의 경우에는 **False**가 된다. 다음 표에서 **A**와 **B**의 상태에 따른 결과를 확인해 보자.

A	B	Result
True	True	True
True	False	False
False	True	False
False	False	False

2. 연산자

- or
- or 연산자는 and 연산자와는 달리 A가 True이거나 B가 True인 경우에 True가 되고, A와 B가 모두 False인 경우에만 False가 된다. 다음 표에서 A와 B의 상태에 따른 결과를 확인해보자.

A	B	Result
True	True	True
True	False	True
False	True	True
False	False	False

2. 연산자

- not
- not 연산자는 **단항 연산자로 참과 거짓을 반대로 바꿀 때 사용**한다. 참고로 단항 연산자란 피연산자가 하나인 연산자를 말한다. 다음 표에서 A의 상태에 따른 결과를 확인해 보자.

A	Not
True	False
False	True

3. 변수

한양대학교 ERICA
Education Research Industry Cluster @ Ansan

- 1.1 변수란

- 우리가 다루는 모든 대상에는 이름이 있어서 대화할 때 대상을 더 명확하게 가리킬 수 있다. 예를 들어 '전화나 문자를 주고받거나, 비디오 영상을 볼 때 사용하는 직사각형의 전자 장치'라고 하는 대신에 '핸드폰'이라는 이름을 사용하면 명확하고 좀 더 쉽게 대화할 수 있다. 이와 같이 프로그래밍에서도 원하는 사건이 어떤 순서로 작동할지, 그 사건이 어떤 대상에게 일어나야 할지를 지칭할 때 변수를 사용해 사물을 가리킨다. 프로그램에서 만들어내는 모든 사물에는 이름이 있다. 따라서 나중에 그 이름으로 그 사물을 가리킬 수 있으며 이런 이름을 변수(variable) 라고 하고 이 변수를 사용해 객체를 가리킨다. 그리고 하나 주의 할 점은 수학 시간에 사용하는 변수는 프로그래밍의 변수와는 다르다는 것이다. 수학에서 등식인 'x=1'은 같음을 뜻하지만 프로그램 코드에서의 등호(=)는 대입을 뜻한다.

3. 변수

1.2 할당문

▪ 변수는 값을 저장하는 메모리 공간이다. a = 1의 경우, a라는 이름이 붙은 객체 변수에 1이라는 값이 저장하라는 의미이며, 우리는 이 값을 수학 연산을 적용할 수 있다.

■ 할당문은 변수에 값을 지정하기위해 사용된다.

- 할당문의 왼쪽 : 변수
- 할당문의 오른쪽 : 값

■ 할당문의 예

- a = 30 : a라는 변수에 30이라는 값을 할당함
- b = 50 : b라는 변수에 50이라는 값을 할당함
- b = a : 변수간의 할당문에서는 실제로 참조를 할당함

3. 변수

1.3 할당되지 않은 변수는 에러

▪변수는 사용되기 전에 반드시 할당 되어야 한다. 할당되지 않은 변수는 사용할 수 없다. 아래 출력결과처럼 할당되지 않은 변수 number는 에러가 발생한다. 반대로 할당된 number는 출력되는 결과를 확인할 수 있다.

```
>>> number
Traceback(most recent call last):
  File "<pyshell#0>", line 1, in <module>
    number
NameError: name 'number' is not defined
>>> number = 5
>>> number
5
>>>
```

3. 변수

한양대학교 ERICA
Education Research Industry Cluster @ Ansan

1.4 변수명 규칙

▪변수 이름을 사용하는 이유는 코드를 읽는 사람이 코드를 쉽게 이해할 수 있게 하기 위함이다. 파이썬 프로그래밍 언어에서는 변수에 사용할 수 있는 이름을 제한한다. 변수 이름을 결정할 때에 고려해야할 규칙은 다음과 같다.

	변수명 규칙
1	변수의 이름은 문자, 숫자 그리고 Underscore(_)로만 이루어진다. 다른 기호를 사용하면 구문 에러(Syntax Error)이다. (예) Money$: 구문 에러, $는 사용할 수 없다.
2	변수명은 문자 또는 Underscore로만 시작해야 한다. 즉 숫자로 시작 하면 안된다. (예) 3up, 7brothers : 숫자로 시작했기 때문에, 역시, 구문 에러이다.
3	파이썬 지정단어 (Keyword, Reserved word)들은 변수명으로 사용할 수 없다. (지정 단어 목록 참조)
4	파이썬에서는 대문자와 소문자를 구분한다. (예) hour 와 Hour는 다른 변수이다.
5	공백을 포함 할 수 없다.

3. 변수

한양대학교 ERICA
Education Research Industry Cluster @ Ansan

▪ 의미 있는 변수명

▪변수명은 의미 있게 만들어져야 한다. 역할에 맞는 변수명으로 만들어야 프로그램의 검토, 협업, 공유 시 도움이 된다. 다음 예제를 통해, 실습 A와 실습 B의 차이를 확인하자.

▪예제) 3개의 시험 성적이 주어졌을 때, 시험 성적의 총합과 평균을 구하여라.

▪ 수학: 26점, 영어: 54점, 역사: 96점

▪예제 실습 A

```
a = 26

b = 54

c = 96

d = a + b + c

f = d / 3
```

예제 실습 B

```
math = 26

english = 54

history = 96

total = math + english + history

average = total / 3
```

3. 변수

1.5 파이썬 지정 단어(Keyword, Reserve Word)

- keyword를 import 하면 파이썬에서 지정한 단어들을 확인 할 수 있다. 파이썬 지정 단어는 변수명으로 사용 할 수 없음에 유의하자.
- keyword를 import하여 파이썬 지정 단어를 확인해 보자.

1) keyword.kwlist 명령어는 파이썬 지정 단어를 배열 형태로 나열해 준다.

2) len 함수를 사용해서 파이썬 지정 단어의 개수도 확인해 보자.

```
>>> import keyword
>>> keyword.kwlist
['False', 'None', 'True', '__peg_parser__', 'and', 'as', 'assert', 'async', 'await', 'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']
>>> len(keyword.kwlist)
35
```

3. 변수

한양대학교 ERICA
Education Research Industry Cluster @ Ansan

1.6 여러 변수 할당 (Multiple Assignments)

- 한번에 여러 개 변수에 값을 할당할 수 있다. 단, 한번에 여러 변수에 할당 할 경우에는 변수와 값의 개수가 일치해야 한다. 일치하지 않을 경우에는 에러가 발생한다.
- 다음 명령문을 직접 실행시켜본 후 결과를 비교해보자.
- 1) number_1, number_2 = 550
- 2) number_1, number_2 = 2, 4, 8
- 3) number_1, number_2 = 7, 14

```
>>> number_1, number_2 = 550
Traceback (most recent call last):
  File "<pyshell#18>", line 1, in <module>
    number_1, number_2 = 550
TypeError: cannot unpack non-iterable int object
>>> number_1, number_2 = 2,4,8
Traceback (most recent call last):
  File "<pyshell#19>", line 1, in <module>
    number_1, number_2 = 2,4,8
ValueError: too many values to unpack (expected 2)
>>> number_1, number_2 = 7,14
>>> number_1
7
>>> number_2
14
>>>
```

3. 변수

1.6 변수에 변수 할당

- 변수에 변수를 할당하여 사용할 수 있다. 단 할당문의 오른쪽에 문자가 올 때는 반드시 먼저 값을 할당 받은 후, 할당문의 오른쪽에 문자를 사용해야 한다.

- 다음 명령문을 직접 실행시켜본 후 결과를 비교해보자.
- 1) number_3 = number_4
- 2) number_4 = 550
  - number_3 = number_4

```
> number_3 = number_4
Traceback (most recent call last):
  File "<pyshell#2>", line 1, in <module>
    number_3 = number_4
NameError: name 'number_4' is not defined
>>> number_4 = 550
>>> number_3 = number_4
>>> number_4
550
```

3. 변수

1.7 복합 대입 연산자

▪복합 대입 연산자는 연산과 할당을 동시에 표현하는 것으로 '+=, -=, *=, /=, //=, %=' 등이 있다. 복합 대입 연산자의 의미는 다음과 같다.

복합 대입 연산자	의미
a += 1	a = a + 1
a -= 1	a = a – 1
a *= 1	a = a * 1
a /= 1	a = a / 1
a //= 1	a = a // 1
a %= 1	a = a % 1

3. 변수

1.7 복합 대입 연산자

```
>>> x = 5
>>> x = x + 3
>>> print(x)
    8
```

```
>>> x = 5
>>> x += 3
>>> print(x)
    8
```

4. 입력문을 위한 함수

1.1 input()_입력문을 위한 함수

▪파이썬에서 키보드로 입력 받을 때 가장 손쉽게 사용할 수 있는 명령어로는 input()이 있다. input() 명령은 입력 명령이지만 엄밀히 말하면 출력 명령을 포함한다. 따라서 명령어는 2가지 형식으로 나눌 수 있는데 아래와 같다.

변수이름 = input()	사용자로부터 입력을 받는다.
변수이름 = input('문자열')	'문자열'에 해당하는 내용을 출력 후 사용자로부터 입력을 받는다

▪input() 함수는 모든 것을 문자열로 입력 받는다. 그러므로 사용자에게 문자열을 입력 받으려면 input()을 그대로 사용하면 된다. 사용자에게 이름을 입력 받아서 출력하는 프로그램은 아래와 같다.

```
>>> name = input()
Gildong
>>> name
'Gildong'
>>>
```

```
>>> name = input('What is your first name? ')
What is your first name? Gildong
>>> name
'Gildong'
>>>
```

4. 입력문을 위한 함수

1.2 input() 함수의 자료형 변환

- 앞서 input() 함수는 모든 것을 문자열로 입력 받는다고 설명하였다. 따라서input() 함수의 입력 결과는 '문자열'이다. 산술 계산을 하기 위해서는 문자열을 정수 또는 실수 자료형으로 변환하여야 가능하다. 아래의 자료형을 이용하여 다음 명령문을 직접 실행시켜본 후 결과를 비교해보자.

* 데이터 타입을 지정하여 입력 받는 방법

```
>>> a = input("입력:")
입력:50
>>> print("자료형:",type(a))
자료형: <class 'str'>
>>> print(a+100)
Traceback (most recent call last):
  File "<pyshell#5>", line 1, in <module>
    print(a+100)
TypeError: can only concatenate str (not "int") to str
```

```
>>> a = input("입력:")
입력:50
>>> print("자료형:",type(a))
자료형: <class 'str'>
>>> print(int(a)+100)
>>> 150
```

```
>>> a = input("입력:")
입력:50
>>> print("자료형:",type(a))
자료형: <class 'str'>
>>> print(float(a)+100)
>>> 150.0
```

4. 입력문을 위한 함수

1.3 input() 함수의 자료형 지정

- 앞서 input() 함수는 모든 것을 문자열로 입력 받는다고 설명하였다. 따라서input() 함수의 입력 결과는 '문자열'이다. 하지만 입력 받을 자료형의 종류를 지정하여 입력 받을 수 있다. 문자열인 경우에는 input(), 정수로 입력 받을 경우에는 int(input()), 실수로 입력 받을 경우에는 float(input())을 사용한다. 다음 명령문을 직접 실행시켜본 후 결과를 비교해보자.

```
>>> a = int(input("정수 입력: " ))
정수 입력:50
>>> print( " 자료형: " ,type(a))
자료형: <class   ' int'>
>>> print(a+100)
>>> 150
```

```
>>> a = float(input("실수 입력: " ))
실수 입력:50.5
>>> print( " 자료형: " ,type(a))
자료형: <class 'float'>
>>> print(a+100)
>>> 150.5
```

5. 출력문을 위한 함수

2.1 print()_출력문을 위한 함수

▪입력을 위한 input()함수가 있다면 출력을 위한 print() 함수가 있다. 출력이란 컴퓨터가 계산과정에서 얻은 결과를 보여주는 것인데 print명령어는 3가지 형식으로 나눌 수 있는데 아래와 같다.

print('문자열')	**'문자열'을 화면에 출력해준다.**
print(변수이름)	변수(변수이름)에 해당하는 값을 화면에 출력해준다.
print('문자열',변수이름)	'문자열'과 변수(변수이름)에 해당하는 값을 연속해서 화면에 출력한다.

5. 출력문을 위한 함수

- print('문자열')

```
>>> print('True False')
True False
>>>
```

- print(변수이름)

```
>>> number1 = 36
>>> number2 = 42
>>> sum = number1 + number2
>>> print(sum)
78
>>>
```

- print('문자열',변수이름)

```
>>> math = 74
>>> english = 87
>>> science = 80
>>> print('수학, 영어, 과학 점수 : ', math, english, science)
수학, 영어, 과학 점수 : 74 87 80
```

5. 출력문을 위한 함수

2.1 print() 함수를 이용한 양식 문자

▪print() 함수를 이용하여 문자열, 변수, 숫자로 이루어진 구조의 내용을 출력할 때 양식 문자를 이용하면 좀 더 단순한 형태의 구조로 표현하고 출력할 수 있다.

양식 문자	표현 내용	비고
%d	정수(십진수)	Decimal (0~9)
%f	실수(소수점)	Floating point number
%g	정수 혹은 실수	소수점의 여부에 따라 정수, 실수 자동표시
%s	문자열	String
%c	문자	Character
%o	8진수	Octal number (0~7)
%x	16진수	Hexa number (0~9, A~F)

5. 출력문을 위한 함수

예를 들어 아래와 같이 양식 문자를 사용하지 않은 경우와 사용한 경우를 비교해 보자.

```
name = "홍길동"
age = 21
weight = 58.7
print("내 이름은", name, "입니다.")
print("나는 ", age, "살 입니다.")
print("나의 몸무게는", weight, "kg입니다.")
```

```
name = "홍길동"
age = 21
weight = 58.7
print("내 이름은 %s입니다."%name)
print("나는 %d살 입니다."%age)
print("나의 몸무게는 %f kg입니다."%weight)
```

5. 출력문을 위한 함수

양식 문자 2개 이상 활용하는 경우를 살펴보자.

```
kor = 90
eng = 80
print("국어는 %d점이고 영어는 %d점입니다."%(kor, eng))
국어는 90점이고 영어는 80점입니다.
```

```
x = 7
y = 8
print("%d와 %d를 곱한 값은 %d점입니다."%(x, y, x*y))
7과 8을 곱한 값은 56입니다.
```

연습 문제

1. int형인 숫자들이 나눗셈 연산을 하면 결과의 자료형은 무엇인가?

- 예 : >>> 96 / 4

① int

② float

③ str

④ list

연습 문제

2. 다음의 설명에서 맞는 것에는 O, 틀린 것에는 X를 표시하시오.

1) 변수의 이름은 문자, 숫자 그리고 _(Underscore), 특수문자로만 이루어진다. ( )

2) 변수명은 문자 또는 Underscore로만 시작해야 하며 숫자로는 시작할 수 없다. ( )

3) 변수 Number는 number로 사용해도 된다. ( )

4) num1, num2 = 100 입력 시 두 변수 num1, num2에 각각 100이 할당 된다. ( )

연습 문제

3. ( ) 안에 들어갈 단어를 채우시오.

- '//' 연산자는 나눗셈 연산의 ( ) 만을 결과로 나타내고,
- '%' 연산자는 나눗셈 연산의 ( ) 만을 결과로 나타낸다.

연습 문제

4. 다음 실행문의 잘못 된 부분을 찾으시오.

- >>> b = a + 1

연습 문제

5. a,b에 각각 10, 20 을 할당하고 곱셈을 한 값을 변수 result에 저장하여 출력하는 프로그램을 작성하라.

감사합니다.