2장 값을 담아 다루어 보자

이장에서 배울 것들

- 데이터를 저장하기 위해서 사용하는 변수에 대하여 알아 보아요.
- 변수를 사용하면 얻게 되는 이점을 이해해 보아요.
- 다양한 자료형의 차이를 알아 보아요.
- 입력과 출력의 방법을 살펴 보아요.
- 간단한 계산기 프로그램을 작성해 보아요.
- 간단한 퀴즈 프로그램을 작성해 보아요.
- print() 함수의 사용법을 자세히 살펴 보아요.

2.1 데이터를 저장하는 공간 : 변수

- 데이터 과학에서는 많은 데이터들을 처리하여야 한다. 우리는 어디에 데이터를 저장할 수 있을까?
- 파이썬에서는 변수variable에 데이터를 저장한다. 변수는 컴퓨터의 메모리memory 공간에 이름을 붙이는 것으로 우리는 여기에 정수, 실수, 문자열 등의 자료값을 저장할 수 있다.

2.1 데이터를 저장하는 공간 : 변수

- 저장된 것을 꺼내려면 변수의 이름을 적어주면 된다. 변수는 아래 그림과 같이 상자와 같다고 생각하면 된다.
- 상자에 저장된 값은 수시로 바뀔 수 있다. 그래서 변수(변하는 수)라는 이름이 붙은 것이다. 이 때 사용된 '수'는 수치값이라는 의미가 아닌 데이터data로 이해하는 것이 더 정확하다.

2.1 데이터를 저장하는 공간 : 변수

- 예를 들어서 자신의 몸무게를 weight라는 이름의 변수에 저장한다고 하자. 이를 위해서는 다음과 같은 방법을 사용한다.

```
>>> weight = 78.2
```

- 파이썬 내부에 weight라는 이름의 변수가 생성되고 여기에 78.2가 저장된다.
- 위의 코드에 있는 '=' 기호는 '같다'는 등호의 의미가 아니라 '오른쪽의 값을 왼쪽의 weight라는 이름의 변수에 저장하라'는 의미이다. 이 연산자 = 를 할당연산자 혹은 대입연산자라고 한다.

2.1 데이터를 저장하는 공간 : 변수

- 변수는 값을 저장하는 기능이 있으므로 다음과 같이 변수의 값을 출력하여 보자. 우리가 저장하였던 78.2가 출력되는 것을 알 수 있다. 이와 같이 변수에 값을 저장하면 필요할 때마다 불러내어 사용할 수 있다!

```
>>> weight
78.2
```

2.1 데이터를 저장하는 공간 : 변수

- 파이썬의 대화창 콘솔에서는 변수의 이름만 써주어도 변수의 값이 출력되지만 다음과 같은 스크립트 파일(xxx.py 파일을 말함)에서는 반드시 print(weight) 함수를 사용하여야 변수의 값을 출력할 수 있다.
- 실행 결과는 스크립트 내용 아래에 나타나있다.

```
weight = 78.2
print(weight)
```

```
78.2
```

2.2 변수의 내용은 언제든 바꿀 수 있다

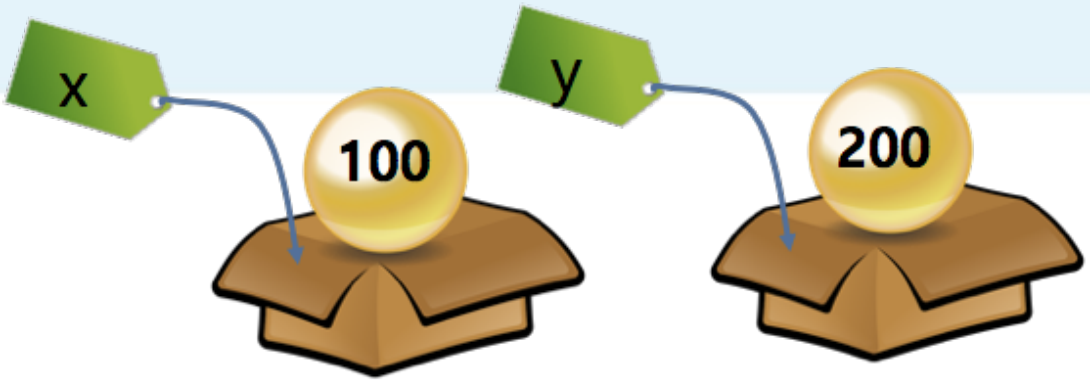
- 우리는 언제든지 변수에 들어 있는 값을 다른 값으로 바꿀 수 있다. 체중이 78.2kg이던 사람이 다이어트를 통하여 76.9kg으로 체중을 줄였다고 가정 하자.
- 이 경우 다음과 같이 체중을 나타내는 변수의 값을 76.9로 바꿀 수 있다.

```
>>> weight = 78.2     # weight 변수의 최초 값
>>> weight = 76.9     # 변수가 가지고 있는 값은 변경할 수 있다
>>> weight
76.9
```

2.2 변수의 내용은 언제든 바꿀 수 있다

- 또한 변수는 필요에 따라서 얼마든지 많이 만들 수 있다. 2개의 변수 x와 y를 생성하고 여기에 100과 200을 저장해보자.
- x, y를 다음과 같이 한 행에 쉼표로 출력하면 (100, 200) 형태의 묶음으로 출력된다.

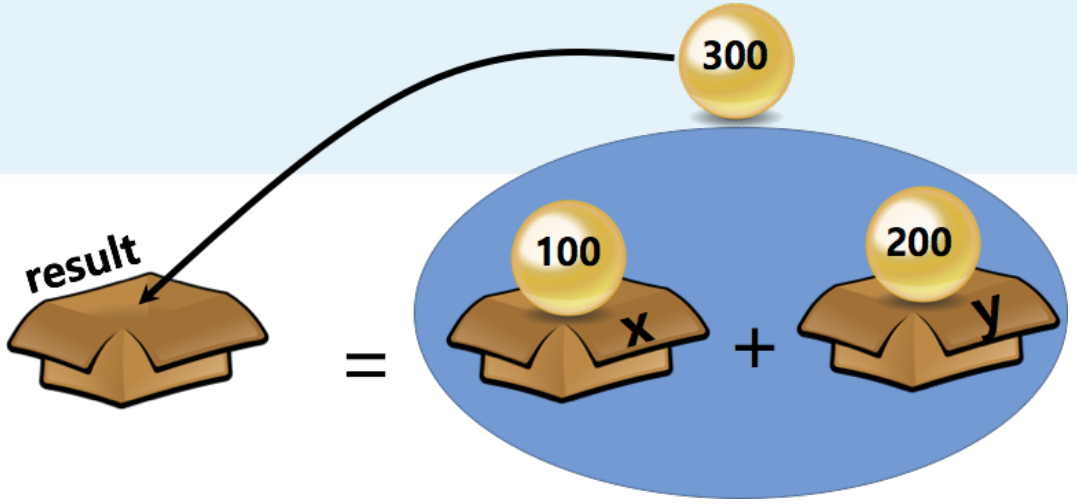
```
>>> x = 100
>>> y = 200
>>> x, y
(100, 200)
```



2.2 변수의 내용은 언제든 바꿀 수 있다

- x, y = 100, 200과 같이 한 줄에 여러 개의 변수를 선언하고 이 변수에 값을 동시에 할당할 수도 있는데, 이를 동시 할당문이라 한다
- x + y 연산을 하여 그 결과 값을 변수 result에 저장하고 이 값을 출력해보자.

```
>>> x, y = 100, 200
>>> result = x + y
>>> result
300
```



잠깐 – 파이썬에서 = 기호는 "같다"라는 뜻이 아니다

입문자들이 가장 많이 오해하는 것 중의 하나가 = 연산자를 '양변이 같다'고 해석하는 것이다. 파이썬에서 = 기호는 "변수에 값을 저장하라"라는 의미이다. = 기호의 오른쪽에 있는 값을 왼쪽의 변수가 가리키는 공간에 담으라는 것이다. 이를 **할당 연산자**assignment operator라고 한다. 따라서 3 = result 같이 **데이터를 담을 수 없는 값이 = 기호의 왼쪽에 오면 문법 오류가 된다**. 파이썬에서 양쪽 값을 비교할 때는 = 기호를 두 개 이어서 사용한 ==를 사용한다.

2.3 변수의 이름은 어떻게 짓나

- 변수의 이름은 프로그래머가 마음대로 지을 수 있지만 몇 가지의 규칙을 지켜야 한다.
- 변수의 이름은 식별자의 일종이다. “홍길동”, “김철수”등의 이름이 사람을 구별하듯이 식별자는 변수들을 구별하는 역할을 한다.

2.3 변수의 이름은 어떻게 짓나

- 식별자는 다음과 같은 규칙에 따라 만들어야 한다.

- 식별자는 문자와 숫자, 밑줄 문자(_)로 이루어지며 밑줄 문자 이외의 특수 문자를 사용할 수 없다.
- 식별자의 첫 글자는 숫자로 시작할 수 없다. 또한 중간에 공백을 가질 수 없다.
- 대문자와 소문자는 구별된다. 따라서 변수 index와 INDEX은 서로 다른 변수이다.
- 파이썬의 예약어(키워드)는 식별자로 사용할 수 없다.

2.3 변수의 이름은 어떻게 짓나

- 우리는 변수의 이름을 마음대로 지을 수 있지만 파이썬이 내부적으로 사용하는 예약어는 변수의 이름으로 사용할 수 없다. 예약어를 사용하여 변수를 생성하면 오류가 발생한다.
- 파이썬의 예약어는 다음과 같다.

False	class	return	is	finally	None
if	for	lambda	continue	True	def
from	while	nonlocal	and	del	global
not	with	as	elif	try	or
yield	assert	else	import	pass	break
except	in	raise			

2.3 변수의 이름은 어떻게 짓나

- 변수의 이름을 지을 때는 변수의 역할을 가장 잘 설명하는 이름을 지어야 한다. 좋은 변수 이름은 전체 프로그램을 읽기 쉽게 만든다. 하지만 반대로 고민없이 즉흥적으로 지은 이름만 사용하게 되면 나중에 프로그램을 다시 읽기가 아주 힘들어진다.
- 예를 들면 체중과 키를 x1, x2라고 하는 것보다 weight와 height라고 하는 편이 이해하기 쉬울 것이다.

체중과 키를 저장하는 나쁜 변수 이름	체중과 키를 저장하는 좋은 변수 이름
`>>> x1 = 78.2` `>>> x2 = 180.0`	`>>> weight = 78.2` `>>> height = 180.0`

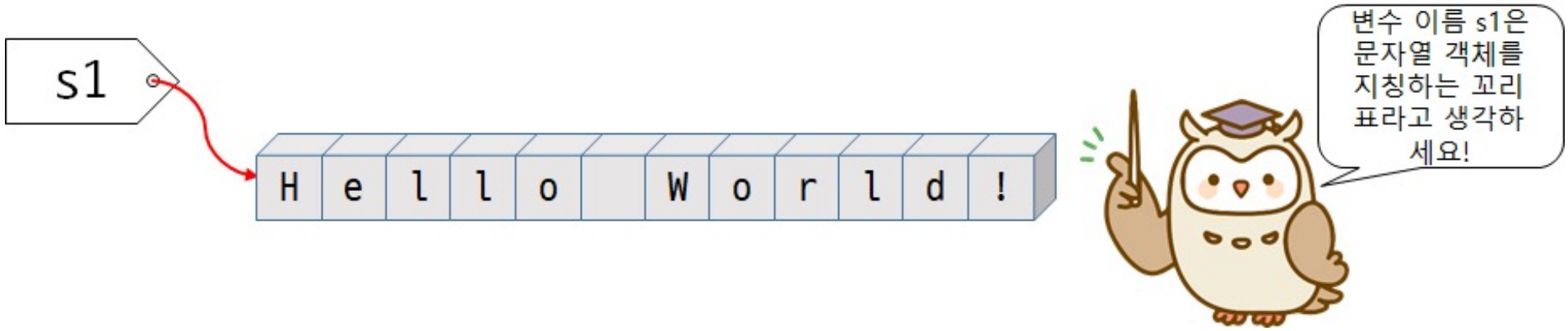
2.4 변수에는 문자열도 저장할 수 있다

- 우리는 지금까지 변수에 정수나 실수만을 저장하였다. 하지만 문자열string: 텍스트 데이터도 변수에 저장할 수 있다. 프로그래밍 세계에서는 텍스트 데이터를 문자열string이라고 부른다.
- 문자열은 큰따옴표("...")나 작은따옴표('...')를 이용하여 만들 수 있다. "Hello World!"도 문자열이고 'Hello World!'도 같은 문자열이다.

```
>>> s1 = 'Hello World!'
>>> s1
'Hello World!'
```

2.4 변수에는 문자열도 저장할 수 있다

- 파이썬에서는 위와 같은 문자열 데이터를 '객체'라는 용어로 부르며, 변수 s1은 이 객체를 지칭하는 꼬리표로 본다.



2.4 변수에는 문자열도 저장할 수 있다

- 파이썬은 개발자들이 편리하게 호출하여 사용할 수 있는 미리 정의된 기능을 함수function라는 형태로 제공하고 있는데, s1 문자열의 길이는 문자열에 len() 함수를 적용하면 알 수 있다.

```
>>> len(s1)    # s1 문자열의 길이를 알려준다
12
```

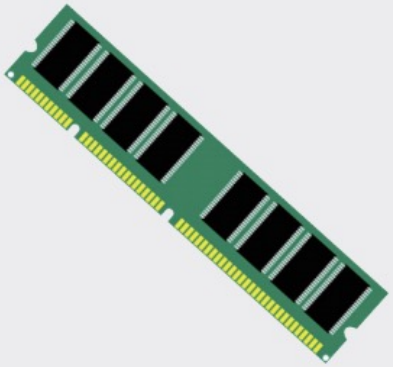
2.4 변수에는 문자열도 저장할 수 있다

- 문자열은 다음과 같이 덧셈 연산자를 사용할 수 있는데, 2개의 문자열을 서로 합하려면 다음과 같이 + 연산을 사용한다.

```
>>> s1 = "Hello"
>>> s2 = "World!"
>>> s1 + s2       # s1 문자열과 s2 문자열을 결합한 결과를 반환
'HelloWorld!'
>>> n1 = '100'
>>> n2 = '200'
>>> n1 + n2       # 문자열에 대한 덧셈 연산자는 문자열 이어 붙이기만 수행
'100200'
```

잠깐 – 오류 메시지를 즐겁게 읽어보자

컴퓨터의 데이터는 오른쪽 그림에 나타난 것과 같은 컴퓨터의 핵심적인 부품인 **메인 메모리**main memory에 저장한다. 이렇게 메모리에 데이터를 저장한 곳의 위치를 **메모리 주소**memory address라고 한다. 이 주소는 보통 0x12FF06이라는 형식의 16진수로 표현한다. 컴퓨터의 입장에서는 0x12FF06라는 주소를 이해하는 것이 전혀 어려운 일이 아니지만, 사람의 입장에서는 그 뜻을 이해하기 힘들고, 다른 주소와 구별도 어렵다. 이 메모리 공간 0x12FF06의 위치에 사람이 인지하기 쉬운 'radius'라는 이름을 붙여보자. 반지름을 의미하는 이 이름을 붙여주면 이 공간에 무엇을 저장해야 하는지, 그리고 읽은 값이 무엇을 의미하는지 다루기가 훨씬 편리하다. 이것이 변수가 사용되는 이유이다.



LAB2-1 신체 질량 지수를 파이썬으로 계산하기

자신의 키와 몸무게를 변수에 저장하고 이들 변수들을 이용하여 BMI를 계산해보자. BMI는 신체 질량 지수로서 신장과 체중을 이용하여 계산할 수도 있다. BMI는 사람의 체지방량과 상관관계가 크다고 증명된 바 있다. BMI 지수는 킬로그램 단위로 측정한 체중을 w, 미터 단위로 측정한 키가 h라고 할 때 다음과 같이 구할 수 있다.

$$BMI = \frac{w}{h^2}$$

원하는 결과

체중(kg), 키(m)를 입력하면 자동으로 BMI 값이 계산되어 변수 bmi에 저장되고, 그 값을 다음과 같이 출력하도록 해보자.

```
>>> bmi
24.1358024691358
```

LAB 2-1 신체 질량 지수를 파이썬으로 계산하기

```
>>> height = 1.80
>>> weight = 78.2
>>> bmi = weight / height**2
>>> bmi
24.1358024691358
```

2.5 변수를 사용해서 좋은 점이 뭘까

- 우리는 체중과 키를 이용하여 bmi를 계산하는 스크립트를 가지고 있다고 가정하자.

```
>>> height = 1.80
>>> weight = 78.2
>>> bmi = weight / height**2
>>> bmi
24.1358024691358
```

- 여러분은 weight 변수의 값만 72.8로 변경하여 스크립트를 다시 실행하면 새로운 BMI 값을 계산할 수 있다.

```
>>> height = 1.80
>>> weight = 72.8
>>> bmi = weight / height**2
>>> bmi
22.469135802469133
```

2.5 변수를 사용해서 좋은 점이 뭘까

- 변수를 사용할 경우에 얻는 좋은 점은 프로그램 코드의 가독성이 높아진다는 것이다.
- 아래의 코드는 무엇인가를 계산하고 있지만, 작성한 사람이 아니라면 무엇을 계산하는지 알기가 어려운 코드일 것이다.

```
>>> print(3.141592 * 10 * 20)
628.3184
```

2.5 변수를 사용해서 좋은 점이 뭘까

- 하지만 다음과 같은 코드는 명확하게 원기둥의 부피를 계산하는 데에 필요한 정보를 이용하여 부피를 계산하는 코드라는 것을 쉽게 파악할 수 있다.

```
>>> pi = 3.141592
>>> radius = 10
>>> height = 20
>>> volume_of_cylinder = pi * radius * height
>>> print(volume_of_cylinder)
628.3184
```

자신의 코드를 나중에 보는 경우가 많으니, 체계적으로 코딩하고, 변수의 이름도 이해하기 쉽게 하세요.

잠깐 – 변수를 사용하면 좋은 이유 2

변수를 사용하면 데이터를 저장하는 곳의 주소를 외우거나 하지 않아도 되는 장점이 있다고 이미 설명했다. 또 한 가지 중요한 장점은 데이터가 변경되어도 일의 방법은 바꿀 필요가 없다는 것이다. 위의 실습에서 몸무게가 바뀌었지만, 신체 질량 지수를 계산하는 방법이 바뀌는 것은 아니다. 따라서 몸무게 변수에 할당되는 값만 바꾸고, 프로그램의 다른 부분은 전혀 손대지 않아도 원하는 결과를 얻을 수 있다.

도전문제 2.1

1. 키가 172cm이고, 몸무게가 78.8kg인 홍길동씨의 BMI를 계산하여 출력하는 bmi_hong.py 스크립트 코드를 작성하고 홍길동씨의 BMI 값을 출력하여라.
2. 자신의 키와 몸무게를 입력하고 BMI를 구한 후 그 값을 출력하여라.

LAB2-2 피자의 면적을 계산해보자

다음과 같이 피자의 면적을 계산하는 스크립트가 있다고 하자. 피자의 반지름은 변수 radius에 저장된다.

```
radius = 10
area = 3.14 * radius**2
print('피자 면적은', area)
```

스크립트를 수행한 결과는 다음과 같이 나타날 것이다.

```
피자 면적은 314.0
```

갑자기 피자의 반지름을 20으로 변경되어서 피자의 면적을 다시 계산하여야 한다면 어떨까? 피자의 반지름이 변수로 표현되었기 때문에 쉽게 구할 수 있을 것이다.

원하는 결과

```
피자 면적은 1256.0
```

LAB2-2 피자의 면적을 계산해보자

```
radius = 20
area = 3.14 * radius**2
print('피자 면적은', area)
```

도전문제 2.2

1. 피자의 반지름을 20으로 하고 면적과 둘레를 동시에 계산해 보자.
2. 피자의 반지름을 30으로 하고 면적과 둘레를 동시에 계산해 보자.

잠깐 – 수학의 변수와 프로그래밍에서의 변수 비교

변수variable는 수학에서 유래한 개념이다. 수학에서 변수는 변할 수 있는 양을 표현하기 위해 사용되는 기호이다. 이 기호는 하나의 특정한 값이 아니라 허용되는 모든 값을 표현하기 위해 사용된다. 이와 상대되는 개념이 상수이다. **상수**constant는 값이 고정되어 있는 수이며, 종종 기호로 표현된다. 예를 들어 원의 넓이를 구하는 잘 알려진 수식 πr^2에서 π는 원주율 3.141592...를 의미하는 상수로 다른 값을 가질 수 없다. 그러나 반지름을 의미하는 r은 특정한 값이 아니라 실수인 모든 수를 표현할 수 있는 변수이다.

프로그래밍에서 사용하는 변수 개념도 수학과 유사한 점이 있다. 그런데 프로그래밍에서 사용하는 변수가 수학의 변수와 다른 점도 분명히 있다. 대표적인 것은 프로그래밍의 변수는 데이터를 저장하는 기억장치에서의 위치를 가진다는 점이다. 이 기억장치에서의 위치와 짝을 이루는 이름을 프로그래밍에서는 변수라고 부른다. 이 변수를 이용하여 해당 메모리 위치에 기록된 값을 읽거나, 새로운 값을 기록할 수 있다.

LAB 2-4 복리 이자 계산하기

다음은 연 3%의 복리 이자를 주는 금융 상품에 1,000만원을 저축하였을 때 5년 후에 얻게 되는 돈을 계산하는 수식이다.

$$10,000,000 \times (1 + 0.03)^5$$

이것을 파이썬 문법에 맞춰 표현하면 다음과 같다.

```
>>> 10000000 * (1.0 + 0.03) ** 5
11592740.743
```

이 수식에 나타나는 값들을 변수로 변경해서 다시 작성해 보자. 그러면 원금과 이율을 바꾸어 가며 복리 계산을 할 수 있을 것이다.

원하는 결과

```
원금:    10000000
이율:    0.03
기간:    5
수령금액:   11592740.743
```

LAB 2-4 복리 이자 계산하기 해답코드

```
principal = 10000000
years = 5
interest_rate = 0.03
money = principal * (1.0 + interest_rate) ** years

print('원금:   ', principal)
print('이율:   ', interest_rate)
print('기간:   ', years)
print('수령금액: ', money)
```

도전문제 2.4

1. 원금을 1,000만원 대신 1억원으로 수정하여 이 프로그램을 실행하고 그 결과를 출력하여라.

2.6 자료형을 알아야 연산을 할 수 있다

- 프로그램에서 사용할 수 있는 데이터의 종류를 자료형data type이라고 한다. 파이썬에서 사용할 수 있는 가장 단순한 기본 자료형은 4가지 종류가 있다.
  - 첫 번째는 1이나 2와 같은 정수integer이다.
  - 두 번째는 3.14와 같은 실수floating-point이다.
  - 세 번째는 'Hello World!'와 같은 문자열string이다.
  - 마지막으로 네 번째는 참과 거짓을 나타내는 부울형bool이다.

자료형	예
정수(int)	…, -3, -2, -1, 0, 1, 2, 3, ...
실수(float)	3.14, 4.28, 0.01, 123.432
문자열(str)	'Hello World', '123', "Hi"
부울형(bool)	True, False

2.6 자료형을 알아야 연산을 할 수 있다

- 파이썬에서는 변수에 어떤 종류의 데이터도 저장할 수 있다. 이것은 프로그래머에게는 축복과도 같다. 왜냐하면 다른 언어에서는 변수를 선언할 때, 반드시 자료형을 지정하여야 한다.
- 심지어 동일한 변수에 정수 데이터를 저장하였다가 나중에 실수 데이터를 저장하여도 된다.

```
>>> x = 10     # 정수 10을 변수 x가 저장하게 됨
>>> x
10
>>> x = 3.14  # 변수 x가 저장하는 값이 3.14로 바뀐다
>>> x
3.14
```

잠깐 – 파이썬의 변수에 어떤 자료형이든 저장할 수 있는 이유

위의 코드에서 x = 10이라고 하면, x는 정수형 데이터를 담게 될 것이다. 그런데 x = 3.14라는 코드가 그 다음에 이어지면, 이제 x는 실수형 변수로 변신하게 된다. 정수와 실수 뿐만 아니라 x = 'hello'라고 하면 순식간에 x는 문자열을 담는 변수로 바뀌게 된다. 어떤 변수가 담을 수 있는 자료형이 번역할 때 고정되어 버리면 **정적 형결정**static typing이라고 하고, 프로그램이 동작하면 변수에 값을 넣을 때 정해지는 것을 **동적 형결정**dynamic typing이라고 한다. **파이썬은 동적 형결정을 사용한다**. 동적 형결정은 미리 변수의 자료형을 정해두고 선언할 필요가 없는 장점이 있지만, 해당 자료형에 허용되지 않는 연산을 할 경우 자료형 오류가 프로그램 동작중에 발생할 수도 있다. 이 경우 프로그램이 이미 시작되었는데 제대로 동작을 마무리하지 못하고 중단되게 된다. 이러한 부분에 주의해야 한다.

2.7 변수의 자료형을 알려면 : type() 함수

- 파이썬에서 변수의 자료형을 알려면 변수의 이름에 type() 함수를 적용하면 된다. 예를 들어서 변수 weight의 타입을 알려면 다음과 같이 한다.

```
>>> weight = 78.2
>>> type(weight)
<class 'float'>
```

- 위의 결과를 보면 'float'라고 나온 것을 알 수 있다. 'float' 타입은 파이썬에서 실수를 의미한다

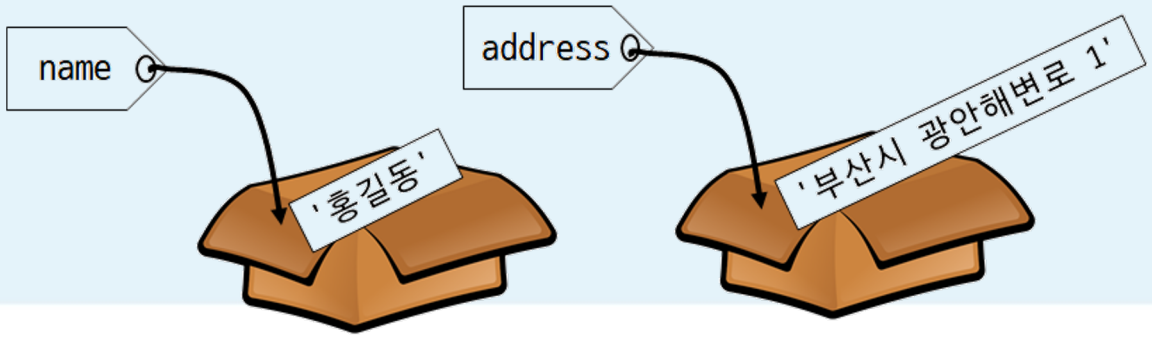
2.7 변수의 자료형을 알려면 : type() 함수

- 이번에는 정수 변수를 만들어서 type()을 호출해보자.

```
>>> salary = 250
>>> type(salary)
<class 'int'>
```

- 이번에는 문자열 변수를 만들어서 type() 함수를 호출해보자.

```
>>> name = '홍길동'
>>> address = '부산시 광안해변로 1'
>>> type(name)
<class 'str'>
>>> type(address)
<class 'str'>
```



2.7 변수의 자료형을 알려면 : type() 함수

- 컴퓨터 프로그래밍에서 매우 중요한 부울bool형이 있다. 부울형은 참이나 거짓을 나타낸다. 부울형은 조건문이나 반복문에서 어떤 조건을 검사할 때 유용하다.

```
>>> b = True
>>> type(b)
<class 'bool'>
```

2.8 왜 자료형에 신경써야 할까 : 자료형과 연산

- 왜 우리는 자료형에 신경 써야 할까? 파이썬에서 자료형에 따라서 각종 연산의 의미가 달라지기 때문이다.
- 즉 변수에 정수가 저장되어 있는 경우와 변수에 문자열에 저장된 경우에 덧셈 연산의 의미가 달라진다.

```
>>> x = 10
>>> y = 10
>>> x + y
20
>>> x = 'good'
>>> y = 'morning!'
>>> x + y
'goodmorning!'
```

x, y는 각각 10, 10을 참조하는 정수형임.

x, y는 각각 'good', 'morning'을 참조하는 문자열형임.
(자료형이 변경되어 +연산이 하는 일이 달라짐)

2.8 왜 자료형에 신경써야 할까 : 자료형과 연산

- 파이썬에서 따옴표가 있으면 문자열이고 따옴표가 없으면 숫자이다. 아래의 코드를 잘 구별하여야 한다. 차이점을 알 수 있는가?

```
>>> '23' + '56'
'2356'
>>> 23 + 56
79
```

- '23'과 '56'은 모두 따옴표가 있기 때문에 파이썬은 문자열로 보았다. 따라서 + 연산자에 의하여 문자열은 서로 합쳐진다.
  하지만 23과 56은 따옴표가 없으므로 23 + 56은 2개의 정수를 합하라는 수식이 되어서 파이썬은 여러분에게 두 정수의 합을 출력한다.

도전문제 2.5

자료형에 따라 + 연산자가 하는 일이 달라진 다는 것을 기억하세요.

무엇이 출력될지 생각해 보고, 그 결과를 파이썬 인터프리터를 통해 확인해보자

(1)
```
>>> x = 7
>>> y = 6
>>> x + y
```
_____

(2)
```
>>> x = '7'
>>> y = '6'
>>> x + y
```
_____

(3)
```
>>> x = '7'
>>> y = '6'
>>> x + y + x
```
_____

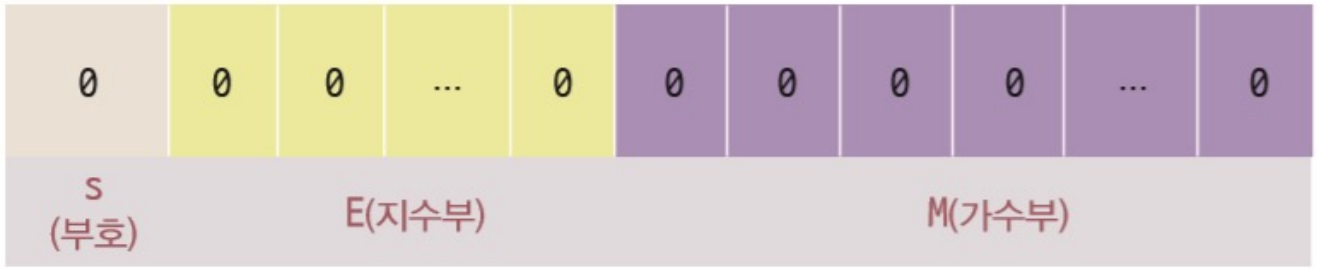
(4)
```
>>> x = 7
>>> y = 6
>>> x + y + x + y
```
_____

2.9 컴퓨터 수치 표현의 한계, 그리고 컴퓨터의 한계

- 컴퓨터를 이용하여 수를 다룰 때 문제가 될 수 있는 수치오류에 대해 알아보도록 하자
- 1/3은 소수로 표현하면 어떻게 표현할 수 있을까? 우리는 쉽게 0.333333.. 으로 시작하는 무한 소수라고 쉽게 대답할 수 있다.
- 그렇다면 이러한 무한히 계속되는 소수를 컴퓨터로는 어떻게 표현할까? 우리가 알다시피 컴퓨터의 저장공간은 유한하므로 컴퓨터는 무한을 표현할 수 없다.

2.9 컴퓨터 수치 표현의 한계, 그리고 컴퓨터의 한계

- 실제 답에 매우 가깝긴 하지만 오차를 가지는 근사치를 표현하는 수밖에 없다.
- 예를 들어 1/3의 답은 무한히 표현할 수는 없으니 0.3333333333333 까지만 표현하는 것이다.
- 컴퓨터는 실수 값을 표현할 때 와 같이 로 표현하고, 부호비트 s, 유효숫자 M, 지수 E를 각각 정수로 저장한다.



2.9 컴퓨터 수치 표현의 한계, 그리고 컴퓨터의 한계

- 따라서 컴퓨터에서 수치오류는 불가피한 것이다. 이러한 불가피한 수치오류는 다음과 같은 상황을 만든다.

```
>>> 0.1 + 0.1 == 0.2
True
>>> 0.1 + 0.1 + 0.1 == 0.3  # 수치오류로 인해 우리가 생각한 True가 나오지 않음
False
```

- 0.1+0.1+0.1이 어떤 값으로 저장되어 있는지를 확인해 보면 더 잘 이해할 수 있을 것이다. 매우 작은 차이지만 0.3과는 다른 값으로 저장되어 있을 것이다.

```
>>> 0.1 + 0.1 + 0.1     # 0.3이 아닌 미세한 수치오류를 가진 근사값
0.30000000000000004
```

2.9 컴퓨터 수치 표현의 한계, 그리고 컴퓨터의 한계

- 실수와 마찬가지로 프로그래밍 언어의 정수 역시 그 범위가 정해져 있다. 그러나 파이썬은 매우 강력한 정수 표현을 제공한다.
- 다음의 실습을 보자. 이처럼 시스템의 메모리가 허용하는 한도 내에서 다음과 같은 규모의 수도 표현이 가능한 것이다. 이는 파이썬의 큰 장점이다.

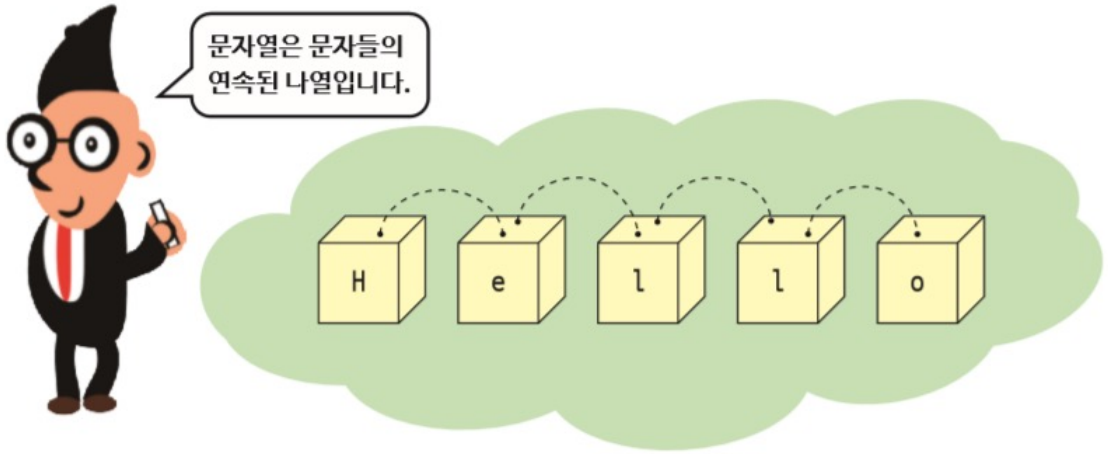
```
>>> 10 ** 100  # 10의 100승
10000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
```

2.10 문자들의 연속된 모음을 문자열이라고 한다

- 컴퓨터에게는 숫자가 중요하지만 인간은 주로 텍스트text를 사용하여 정보를 표현하고 저장하므로 프로그래밍에서 텍스트의 처리도 무척 중요하다.

2.10 문자들의 연속된 모음을 문자열이라고 한다

- 파이썬은 이러한 텍스트를 다루기 위한 자료형으로 문자열을 사용한다. 문자열string은 문자들의 나열이다. 문자열은 우리가 흔히 텍스트Text라고 부르는 것이다. 프로그래밍에서는 전문적인 용어를 사용해야만 혼동이 없기 때문이다.
- 문자열에 대응하는 영어 단어 string은 끈이라는 의미이다. 우리는 글자들이 끈으로 묶여 있는 모습을 상상하면 되겠다.



2.10 문자들의 연속된 모음을 문자열이라고 한다

- 이 책에 있는 모든 글자들도 문자열이 될 수 있다. 우리가 일상적으로 사용하는 단어, 문장들도 문자열이 된다. 우리들의 이름이나 회사의 이름도 문자열이 될 수 있다.
- 주의할 점은 따옴표 내의 공백 문자 역시 문자열의 일부로 간주된다는 점이다. 그리고 화면에 표시되지 않는 탭 문자, 줄바꿈 문자 등도 문자열의 일부이다.

2.11 문자열을 만드는 방법

- 파이썬에서는 텍스트를 큰따옴표("...")나 작은따옴표('...')로 감싸면 문자열이 된다.
- 예를 들어서 다음의 "Hello Python"은 공백문자를 포함한 하나의 문자열이다. 그리고 큰따옴표로 감싼 문자는 특별한 이유가 없을 경우 아래와 같이 작은따옴표로 표기된다.

```
>>> "Hello Python"
'Hello Python'
```

큰따옴표로 감싸더라도 문자열로 인식하므로 표기할 적에는 작은 따옴표로 표기합니다.

2.11 문자열을 만드는 방법

- 문자열은 변수에 저장될 수 있다. 그리고 변수에 저장된 문자열을 print() 함수를 이용하여 출력할 수 있다. 아니면 그냥 변수 이름만 입력하고 엔터키를 눌러도 된다.

```
>>> msg = "Hello"    # 변수를 이용해서 문자열을 저장한다
>>> msg              # 문자열을 출력하면 기본적으로 작은 따옴표내에 문자가 나타남
'Hello'
>>> print(msg)       # print()로 문자열을 출력하면 따옴표는 나타나지 않음
Hello
```

- 문자열을 만들때는 다음과 같이 작은따옴표('...')를 사용해도 된다 (이 방법을 권장한다).

```
>>> msg = 'Hello'
```

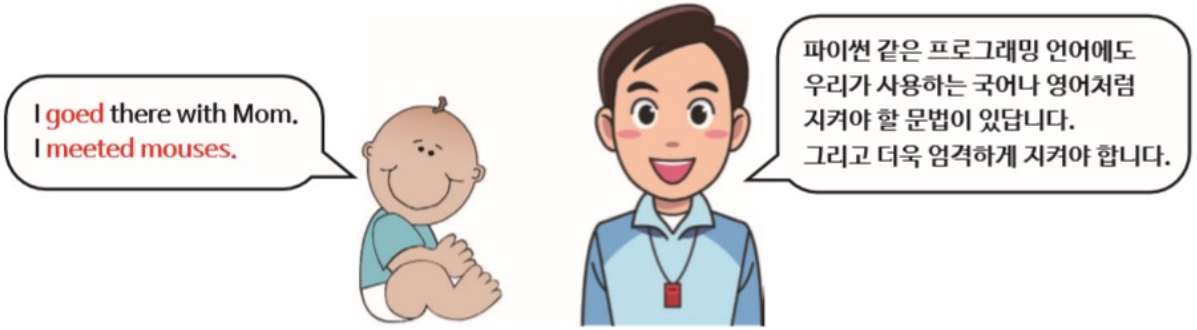
2.11 문자열을 만드는 방법

- 하지만 큰따옴표(")로 시작했다가 작은따옴표(')로 끝내면 문법적인 오류이다.

```
>>> msg = "Hello'
SyntaxError: EOL while scanning string literal
```

- 또 따옴표로 시작했는데 단어의 끝에 따옴표가 없어도 문법적인 오류가 된다.

```
>>> msg = "Hello
SyntaxError: EOL while scanning string literal
```



잠깐 – 문자열을 만들다가 문법오류Syntax Error가 발생하는 경우

문법(syntax)이라는 것은 컴퓨터에서는 **프로그램의 문장을 바르게 구성하기 위한 규칙**을 의미한다. 만약 우리가 영어 문장을 작성하는데 주어와 동사의 수를 일치시키지 않는다면 문법적인 오류이듯이 프로그래밍 언어에서도 문장을 작성하는데 지켜야할 규칙이 있는 것이다.

파이썬에서 문법 오류가 발생하면 "SyntaxError: ..."라는 경고 메시지가 표시된다. 앞선 예시의 오류메시지에서 **EOL이라는 것은 "End Of Line", 즉 줄의 끝을 만났다는 의미**이다. 문자열이기 때문에 문자열을 끝내는 큰따옴표 혹은 작은따옴표가 있을 것으로 기대하였는데 줄의 끝을 만날 때까지 발견하지 못했다는 의미가 된다.

2.12 왜 큰따옴표와 작은따옴표를 동시에 사용할까

- 텍스트 데이터를 처리하기 위해 따옴표를 쓰는데, 따옴표가 없으면 어떻게 될까?

```
>>> msg = Hello
...
NameError: name 'Hello' is not defined
```

- 문법 오류가 발생하였다. Hello라는 이름이 정의되지 않았다고 한다. 파이썬은 따옴표가 없이 주어진 단어는 약속된 단어가 아니라면 변수로 취급한다. Hello라는 변수를 만들어 값을 넣은 일이 없으므로 파이썬은 알지 못 하는 변수를 사용했다고 오류를 발생시키는 것이다.

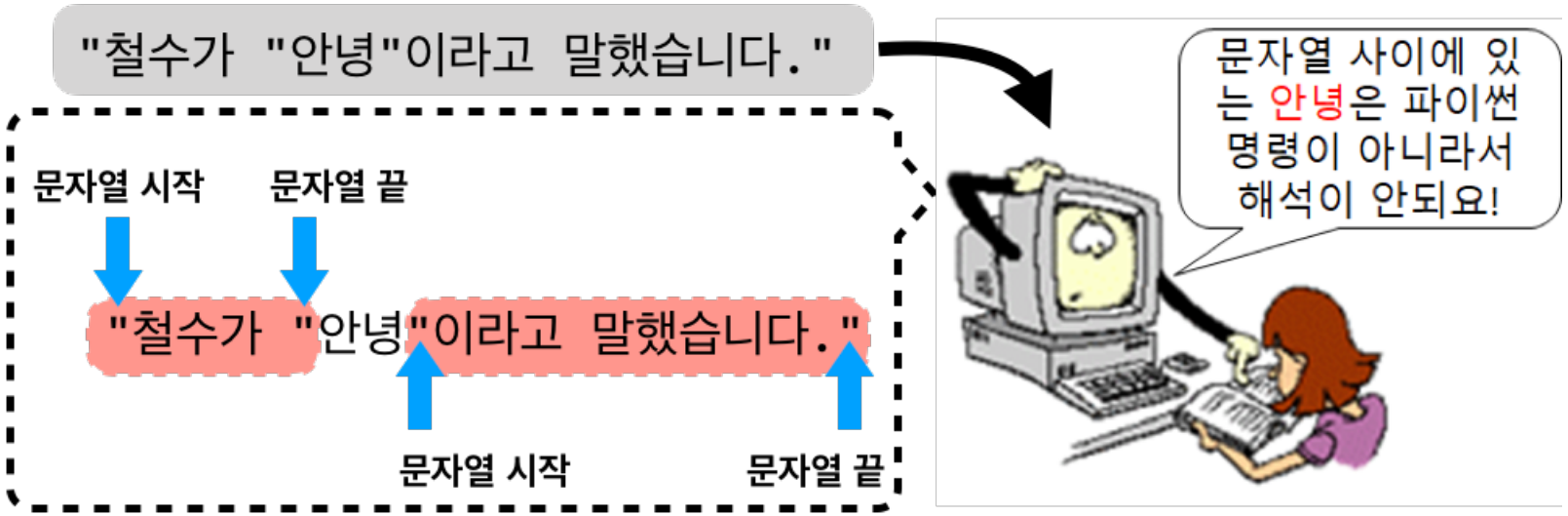
2.12 왜 큰따옴표와 작은따옴표를 동시에 사용할까

- 그러면 왜 파이썬에서는 문자열을 나타내는데 큰따옴표와 작은따옴표를 동시에 사용할까?
- 이것에도 이유가 있다. 문자열 안에 따옴표가 들어가는 경우를 처리하기 위해서이다. 예를 들어서 "철수가 "안녕"이라고 말했습니다." 문장을 문자열 안에 저장할 수 있을까?
- "..." 형태의 문자열 안에 "..." 형태의 문자열을 포함해 파이썬 인터프리터에게 넘겨주면 아래와 같이 문법 오류가 발생할 것이다.

```
>>> message = "철수가 "안녕"이라고 말했습니다."
SyntaxError: invalid syntax
```

2.12 왜 큰따옴표와 작은따옴표를 동시에 사용할까

- 파이썬은 따옴표가 시작된 뒤에 따옴표를 다시 만나면 문자열이 끝난 것으로 파악한다.
- 따라서 아래 그림과 같이 빨간색으로 표시된 두 개의 영역이 문자열로 해석하고 그 사이는 알 수 없는 명령어나 변수로 해석한다.



2.12 왜 큰따옴표와 작은따옴표를 동시에 사용할까

- 이제 문자열을 작은따옴표로 시작해 보자. 그러면 작은따옴표를 다시 만날 때까지 문자열이 이어진다. 그러면 "안녕"은 과 같이 큰 따옴표는 문자열 내에 포함된 문자들로 취급될 것이다.

```
>>> message = '철수가 "안녕"이라고 말했습니다.'
>>> message
'철수가 "안녕"이라고 말했습니다.'
>>> print(message)
철수가 "안녕"이라고 말했습니다.
```

2.12 왜 큰따옴표와 작은따옴표를 동시에 사용할까

- 문자열을 시작한 따옴표와 같은 모양의 따옴표를 출력해야 한다면 다음과 같이 \'을 이용하여 작은 따옴표안에 작은 따옴표를 써 줄 수도 있는데 역슬래시는 이와 특수한 문자를 표기하는데 사용된다. \n은 줄바꿈, \t는 탭 문자를 표시한다. 역슬래시를 출력하고자 한다면 역슬래시를 2개 연속하여 사용하면 된다.

```
>>> print('철수가 \'안녕\'이라고 말했습니다.')
철수가 '안녕'이라고 말했습니다.
>>> print('안녕\n우리 다시 만나~~')
안녕
우리 다시 만나~~
>>> print('안녕\t우리 다시 만나~~')
안녕	우리 다시 만나~~
```

이와 같이 특수한 목적으로 사용되는 역슬래시를 이스케이프 문자라는 특별한 이름으로 부른다.

2.13 왜 오류가 발생할까 : 자료형의 변환

- 우리는 100 뒤에 문자열 '원'을 붙여서 출력하려고 다음과 같이 파이썬 인터프리터에 입력하려고 할 수 있다.
- 하지만 이렇게 입력하면 오류가 발생한다. 왜 다음과 같은 코드에서는 오류가 발생할까?

```
>>> 100 + '원'
Traceback (most recent call last):
File "<ipython-input-3-1b3ddc5be148>", line 1
100+"원"
TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

정수(int)와 문자열(str) 사이에는 덧셈연산을 지원하지 않는다는 의미.

2.13 왜 오류가 발생할까 : 자료형의 변환

- 이 오류는 자료형과 깊은 관련이 있다. 100의 자료형은 정수이다. '원'의 자료형은 문자열이다.
- 만약 정수+정수라면 2개의 정수를 덧셈하면 된다. 만약 문자열+문자열이라면 2개의 문자열을 연결하면 된다.
- 하지만 위의 코드에서는 정수+문자열이기 때문에 파이썬은 무엇을 해야할 지 모르는 것이다.

2.13 왜 오류가 발생할까 : 자료형의 변환

- 이때는 자료형을 통일시켜야 한다. 즉 우리가 하고자 하는 작업에서 100을 문자열 '100'으로 변환하고 이 문자열에 '원'과 같은 문자열을 붙이는 것이었다.
- 따라서 우리는 우선 100을 문자열 '100'으로 변환하여야 하는데, 파이썬에서는 str() 함수가 이 일을 한다. 함수에 대해서는 6장에서 상세히 다룰 것이다.

```
>>> str(100) + '원'    # str() 함수는 100을 문자열 '100'으로 변환시킴
'100원'
```

2.13 왜 오류가 발생할까 : 자료형의 변환

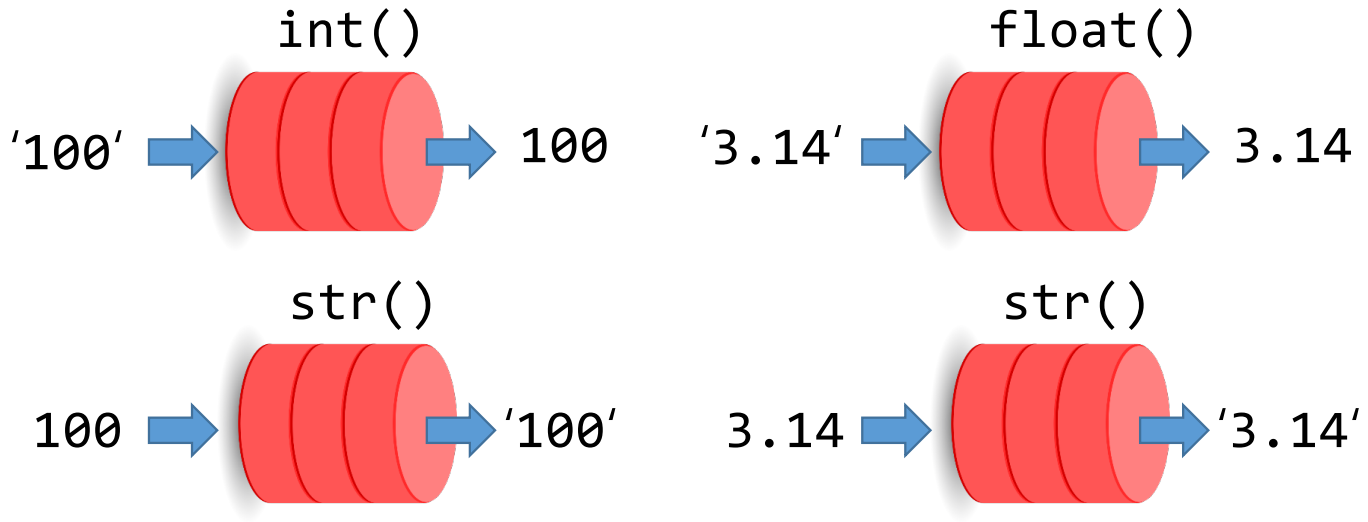
- 이것을 좀 더 자세히 살펴보자. 정수를 문자열로 변환하려면 str()을 사용한다. 반대로 문자열을 정수로 변환하려면 int()를 사용한다.

```
>>> str(100)
'100'
>>> int("100")
100
```

- 마찬가지로 실수 데이터를 문자열로 변환할 때에도 str()을 사용한다. 반대로 문자열을 실수로 변환하려면 float()를 사용한다.

```
>>> str(3.14)
'3.14'
>>> float("3.14")
3.14
```

2.13 왜 오류가 발생할까 : 자료형의 변환



정수와 문자열 사이의 변환

실수와 문자열 사이의 변환

2.14 사용자로부터 문자열 입력받기 : input() 함수

- 우리는 앞에서 변수에 들어 있는 값들을 이용하여 BMI를 계산하거나 정수의 합을 계산한 바 있다. 하지만 이 값들을 사용자로부터 직접 받을 수 있다면 더 유용할 것이다.
- 파이썬에서 사용자로부터 무엇인가를 입력받으려면 input() 함수를 사용한다. input() 함수는 사용자의 입력을 무조건 문자열 형태로 우리에게 반환한다.

```
input_string = input(output_string)
```

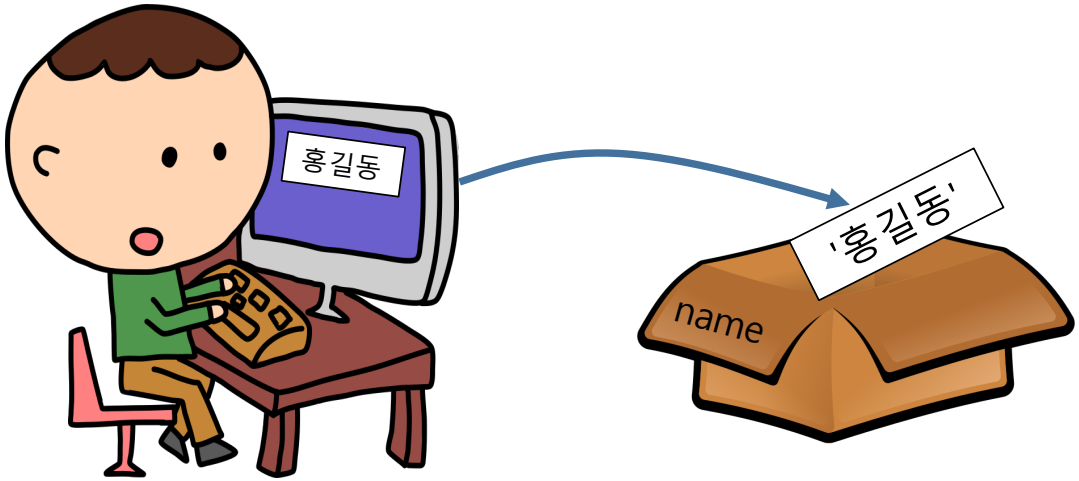
2.14 사용자로부터 문자열 입력받기 : input() 함수

- 코드가 사용되면 화면에 output_string에 담긴 문자열이 출력된다. 그리고 사용자의 입력을 기다리는 상태가 된다.
- 사용자가 키보드를 이용하여 입력을 하고 엔터키를 누르면 input() 함수는 그때까지 입력한 문자들을 모두 하나로 묶어 문자열을 만들어 반환한다. 따라서 입력의 내용은 input_string에 저장된다.
- input() 함수를 사용하는 실제 예를 통해 사용법을 익혀보자. 사용자의 이름을 입력받는 코드는 다음과 같이 구현할 수 있다. 음영으로 처리된 홍길동 부분이 사용자가 입력한 부분이다.

```
>>> name = input("이름을 입력하시오: ")
이름을 입력하시오: 홍길동
```

2.14 사용자로부터 문자열 입력받기 : input() 함수

- 위의 코드가 실행되면 파이썬 내부에는 name이라는 변수가 생성되었고 여기에는 '홍길동'이라는 문자열이 저장되었다.
- 파이썬의 변수는 이와 같이 컴퓨터에서 사용할 수 있는 자료값을 저장할 수 있다.



2.14 사용자로부터 문자열 입력받기 : input() 함수

- 변수 name에는 사용자의 이름이 저장되어 있다. 이 변수를 이용하여 다음과 같이 출력하는 코드를 작성해보자.

```
이름을 입력하시오: 홍길동
홍길동 씨, 안녕하세요?
파이썬 프로그래밍의 세계에 오신 것을 환영합니다.
```

```
name = input("이름을 입력하시오: ")
print(name, "씨, 안녕하세요?")
print("파이썬 프로그래밍의 세계에 오신 것을 환영합니다.")
```

2.15 사용자로부터 정수 입력받기

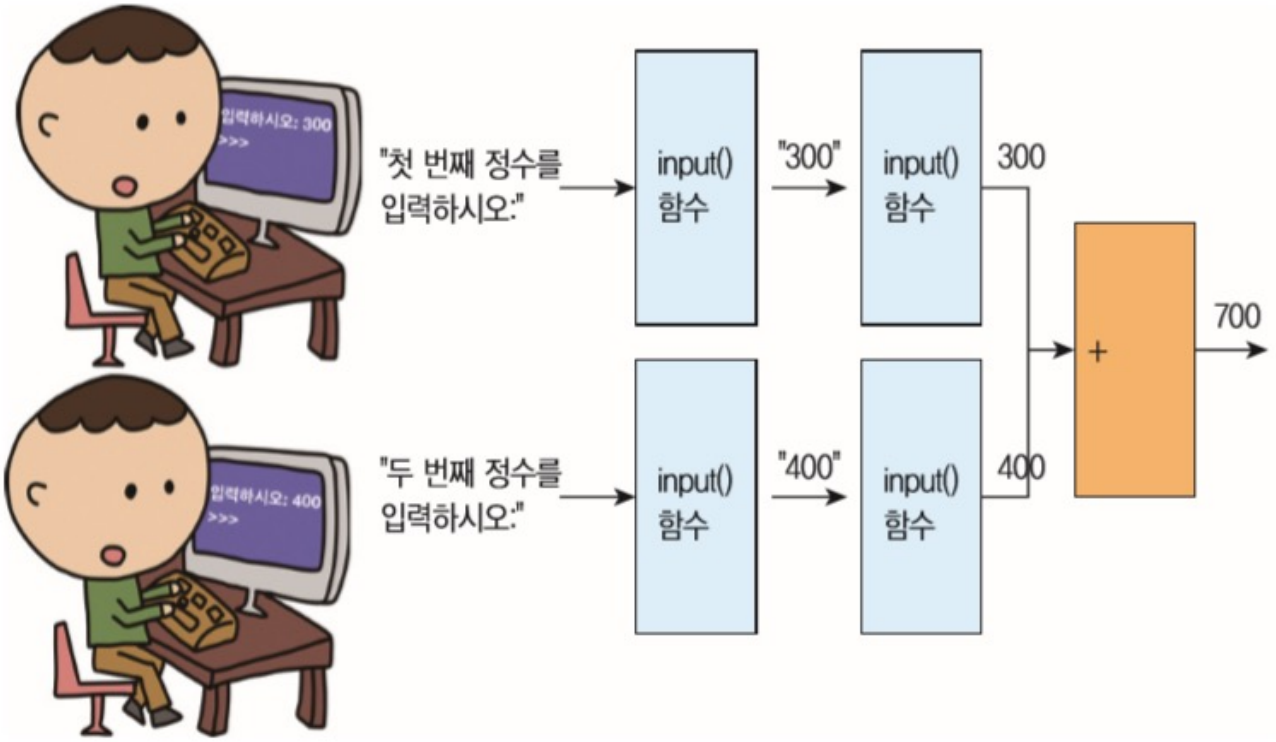
- 앞에서 우리는 덧셈 프로그램을 작성한 적이 있는데 항상 100+200의 결과만을 출력한다.
- 만약 우리가 사용자로부터 정수 2개를 받아서 덧셈을 한 후에 결과를 출력한다면 좀 더 유용한 프로그램이 될 것이다.

```
첫 번째 정수를 입력하시오: 300
두 번째 정수를 입력하시오: 400
300 과 400 의 합은 700 입니다.
```

2.15 사용자로부터 정수 입력받기

- 우리는 input() 함수를 이용하여 정수를 받을 수 있다. 하지만 input() 함수는 사용자의 입력을 무조건 문자열 형태로 우리에게 반환한다.
- 따라서 "300"이라는 문자열이 우리에게 반환된다. 이것을 정수 300으로 변환하려면 input() 함수의 반환값을 int()로 감싸야 한다. int() 함수는 문자열을 정수로 변환한다.

```
x = int(input("첫 번째 정수를 입력하시오: "))
y = int(input("두 번째 정수를 입력하시오: "))
s = x + y
print(x, "과", y, "의 합은", s, "입니다.")
```



도전문제 2.6

(1) 사용자로부터 체중과 신장을 입력하도록 하여, 입력된 값에 따라 BMI를 계산하는 스크립트를 작성하라. 앞에서 이미 만들어 본 스크립트를 참조할 수 있을 것이다.
(2) 사용자로부터 두 수를 입력받아 두 수의 합과 평균을 출력하는 프로그램을 작성하여라.
(3) 사용자로부터 세 수를 입력받아 세 수의 합과 평균을 출력하는 프로그램을 작성하여라.

LAB2-5 로봇 기자가 야구기사를 쓰다

야구 기사를 보면 거의 비슷한 기사가 되풀이 된다. 이긴 팀이나 진 팀, 점수, 경기장, 우수 선수 등의 핵심 요소를 제외한 나머지 부분은 크게 바뀌지 않는다. 기사의 틀을 만들어두고 핵심 요소는 변수로 만들면 자동으로 기사를 작성할 수 있다. 이것을 프로그램을 만들어보자. 사용자에게 경기장, 점수, 이긴 팀과 진 팀, 우수 선수를 질문하고 변수에 저장한다. 이들 문자열에 문장을 붙여서 기사를 작성한다.

원하는 결과

```
경기장은 어디입니까? 강릉
이긴 팀은 어디입니까? 드림즈
진 팀은 어디입니까? 비룡스
우수선수는 누구입니까? 강두기
스코어는 몇대몇입니까? 9:7

==========================================
오늘 강릉 에서 야구 경기가 열렸습니다.
드림즈 과 비룡스 은 치열한 공방전을 펼쳤습니다.
강두기 의 맹활약으로  드림즈 가 비룡스 를  9:7 로 이겼습니다.
==========================================
```

LAB2-5 로봇 기자가 야구기사를 쓰다

```
# 사용자의 대답을 변수에 저장한다.
stadium = input("경기장은 어디입니까? ")
winner = input("이긴 팀은 어디입니까? ")
loser = input("진 팀은 어디입니까? ")
vip = input("우수선수는 누구입니까? ")
score = input("스코어는 몇대몇입니까? ")

# 사용자의 입력을 바탕으로 기사를 작성한다.
print("")
print("==========================================")
print("오늘", stadium, "에서 야구 경기가 열렸습니다.")
print(winner, "과", loser, "은 치열한 공방전을 펼쳤습니다.")
print(vip, "의 맹활약으로 ", winner,"가", loser,"를 ", score,"로 이겼습니다.")
print("==========================================")
```

도전문제 2.7

축구나 테니스 등 자신이 좋아하는 경기로 바꾸어 로봇 기자가 다양한 기사를 작성할 수 있도록 해 보라. 스포츠 경기 뿐만 아니라 주식 시황을 알려주는 기사도 작성해 보라.

LAB2-6 부동산 광고 만들기에 도전하자

이번 실습에서는 부동산에 관한 여러 가지 사항을 변수에 저장한다. 예를 들어서 부동산의 주소는 문자열 형태로 street 변수에 저장될 수 있다. 부동산의 가격은 정수 형태의 변수 price에 저장될 수 있다. 방의 수도 number_of_rooms라는 변수에 저장될 수 있다. 부동산 타입도 문자열 변수 st에 저장될 수 있다.

```
street = "서울시 종로구"
st = "아파트"
number_of_rooms = 3
price = 100000000
```

이들 변수들을 사용하여 다음과 같은 부동산 광고를 화면에 출력하여 보자.

원하는 결과

```
################################
#                              #
# 부동산 매물 광고             #
#                              #
################################

서울시 종로구 에 위치한 아주 좋은 아파트 가 매물로 나왔습니다. 이 아파트 는 3 개의 방을 가지고 있으며 가격은 100000000 입니다.
```

LAB2-6 부동산 광고 만들기에 도전하자

```
street = "서울시 종로구"
st = "아파트"
number_of_rooms = 3
price = 100000000

print("##################################")
print("#                                #")
print("# 부동산 매물 광고               #")
print("#                                #")
print("##################################")
print("")
print(street, "에 위치한 아주 좋은 ", st, "가 매물로 나왔습니다. 이 ",
st, "는 ", number_of_rooms, "개의 방을 가지고 있으며 가격은",
price, "입니다.")
```

summary

핵심 정리

- 컴퓨터에서는 변수를 사용하여 어떤 것을 컴퓨터 메모리 안에 저장할 수 있다.
- 변수들은 이름을 가지고 있다.
- 변수들은 숫자뿐만 아니라 문자열도 저장할 수 있다. 사실은 어떤 자료형이든지 저장이 가능하다.
- `input()` 함수를 사용해서 사용자로부터 문자열을 입력받을 수 있다.
- `int()` 함수를 사용해서 문자열을 정수값으로 변환할 수 있다.

Questions?