

02. 파이썬 변수와 문자열

이번 장에서는 파이썬을 계산기처럼 사용해 보면서 파이썬에 좀 더 익숙해진 후, 모든 프로그래밍 언어에서 사용하는 기본 개념 중 하나인 변수(variable)에 대해 배우겠습니다. 그리고 파이썬 프로그래밍의 기본 데이터 타입인 정수, 실수, 문자열(string)에 대해 설명하겠습니다.

1) 파이썬으로 하는 계산

네이버(Naver) 주식 10 주를 가지고 있다고 가정해 봅시다. 참고로 2015 년 6 월 12 일 장 종료를 기준으로 네이버 주가는 한 주당 601,000 원이었습니다. 10 주를 모두 주당 601,000 원에 매도했다면 매도 금액은 얼마가 될까요?

실시간 증권정보



그림 2.1 네이버 주가(출처: 다음증권)

계산이 빠르신 분들은 아마 6,010,000 원이라고 바로 대답하신 분도 있겠지만 산수에 약하신 분들은 그림 2.2 와 같이 윈도우에서 제공하는 계산기를 사용하신 분도 있을 것입니다.

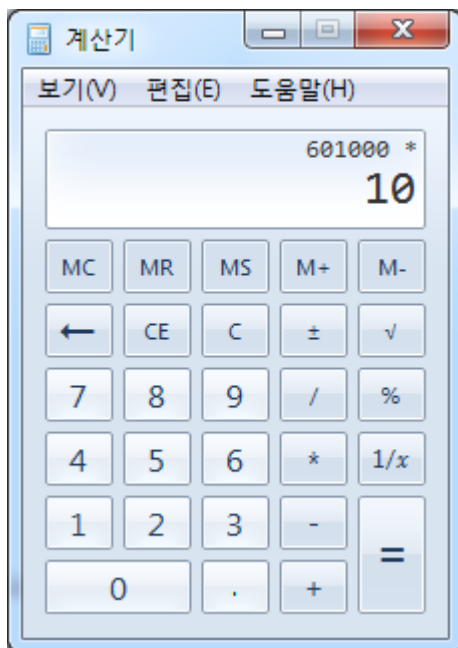


그림 2.2 계산기를 이용한 계산

1 장에서 배운 파이썬 IDLE 을 이용하면 윈도우 계산기를 이용하는 것보다 더 쉽게 위의 계산을 수행할 수 있습니다. 윈도우 계산기에서 벗어나 파이썬 IDLE 에서 계산을 하는 순간, 이미 여러분은 프로그래밍을 하고 있는 것입니다. 먼저 파이썬 IDLE 를 실행한 후 파이썬 프롬프트에 "601000*10"을 입력한 후 엔터(Enter) 키를 눌러봅시다.

```
>>> 601000*10
```

```
6010000
```

```
>>>
```

윈도우 계산기를 이용했던 것과 마찬가지로 주당 601000 을 입력하고(이때 쉼표를 넣지 않아야 합니다), 곱하기 기호를 누른 후 10 을 입력했습니다. 그런 다음 엔터키를 누르면 여러분이 예상했던 것처럼 6,010,000 으로 정확히 계산됩니다. '프로그래밍을 공부하다가 갑자기 웬 계산이냐?'라고 생각하는 분도 있겠지만 주식에서는 더하기, 빼기, 곱하기, 나누기를 잘해야 돈을 벌 수 있습니다.

예를 들어, 매수 금액 대비 현재가가 3% 이하로 하락하면 보유 주식을 모두 매도하는 프로그램을 만든다고 가정해 봅시다. 참고로 이러한 기능을 '스탑로스'라고 하며, 웬만한 증권사의 HTS 는 이러한 기능을 기본적으로 제공합니다. 이러한 프로그램을 직접 만들려면 먼저 여러분이 매수한 주식의 주가가 매수 가격 대비 3% 떨어졌을 때의 가격을 알아야 합니다. 가령 네이버 주식 10 주를 601,000 원에 매수했다고 가정하면 3% 하락 시의 주당 가격은 " $601,000 * 0.97$ "이 됩니다.

IDLE 프롬프트에 계산할 때 숫자와 곱하기 기호(*) 사이에는 공백이 있어도 되고, 없어도 됩니다. 즉, $601000*0.97$ 이나 $601000 * 0.97$ 이나 같은 결과가 나옵니다. 다만 프로그램도 규모가 커지다 보면 여러 사람이 나눠서 작성하게 되는데, 이 경우 여러분이 작성한 프로그램의 소스코드(source code)를 다른 누군가도 보고 이해해야 합니다. 또는 여러분이 지금 작성한 코드를 한 달 후에 다시 여러분이 수정해야 하는 경우도 자주 발생합니다. 따라서 소스코드를 작성할 때 항상 가독성을 생각해서 적당한 공백(blank)을 넣는 것이 좋습니다. 아래의 코드와 같이 곱하기 기호 좌우에 공백을 넣는 것이 훨씬 가독성이 좋지 않나요?

```
>>> 601000 * 0.97
```

```
582970.0
```

파이썬 IDLE 에 나타난 결과값을 확인하면 매수 가격(601,000 원)에서 3% 하락한 금액은 582,970 원입니다. 즉, 여러분이 만든 프로그램은 주식시장이 열려 있는 동안 주기적으로 현재가를 얻어온 후 현재가가 582,970 보다 낮으면 바로 매도 주문을 넣으면 됩니다. 이를 글로 써보면 다음과 같습니다.

```
만약 현재가가 582970 보다 낮으면
```

```
    보유 주식을 전량 매도
```

```
그렇지 않으면
```

```
    보유
```

물론 1장에서 배웠듯이 파이썬 프로그래밍은 영어로 된 키워드(keyword)를 사용하기 때문에 위와 같이 한글로는 명령을 내릴 수 없습니다. 다만, 앞으로 파이썬 프로그래밍을 배워서 작성할 코드의 구성은 앞의 예제와 정확히 일치할 것입니다.

지금까지는 곱셈 연산만 해봤지만 파이썬은 덧셈, 뺄셈, 곱셈, 나눗셈, 즉 사칙연산을 지원합니다. 즉, 다음과 같은 코드도 얼마든지 작성할 수 있습니다.

```
>>> 601000 - 587000
```

```
14000
```

```
>>> 180 * 10 + 10000
```

```
11800
```

```
>>>
```

위 코드에서 첫 번째 구문은 네이버 주식의 매수 가격(601,000 원)과 매도 가격(587,000 원)을 뺀 것으로, 주당 손실액이 14,000 원임을 알 수 있습니다. 두 번째 구문은 네이버와 관계없이 저자가 단순히 파이썬 IDLE 를 이용해 "180 * 10+10000"을 계산한 것입니다.

여기서 한 가지 주의할 점이 있는데 산수와 마찬가지로 프로그래밍의 사칙 연산에도 연산자의 우선순위가 존재한다는 것입니다. 즉, 덧셈, 뺄셈, 곱셈, 나눗셈이 동시에 사용되는 경우 곱셈, 나눗셈이 먼저 실행되고, 그다음이 덧셈, 뺄셈 순입니다. 물론 괄호가 있는 경우에는 괄호가 가장 먼저 실행됩니다. 따라서 "180*10"이 먼저 실행되고 그 결과값에 10000 이 더해지므로 최종값은 11800 이 됩니다.

사칙 연산의 순서를 기억하기 어렵다면 파이썬으로 코드를 작성할 때 다음과 같이 괄호를 사용해 연산자의 우선순위를 명시적으로 나타내는 것이 좋습니다.

```
>>> (180 * 10) + 10000
```

```
11800
```

```
>>>
```

2) 변수

2.1 절에서는 파이썬 IDLE 을 이용해 간단한 사칙 연산을 해봤습니다. 이번 절에서는 변수(variable)라는 것을 배울 텐데, 보통 프로그래밍 분야에서 변수라는 용어는 어떤 값을 저장하는 공간을 의미합니다. 변수는 프로그래밍에서 가장 기초적인 내용이기 때문에 변수의 개념을 잘 이해하는 것은 매우 중요합니다. 그렇다면 왜 변수라는 것이 필요할까요?

2.1 절에서 네이버의 시가가 601,000 원이라고 가정했습니다. 그런데 자꾸 계산할 때마다 이 값을 기억하고 있자니 참으로 귀찮고 어렵기도 합니다. 우리는 일상생활에서 어떤 것을 기억해야 할 때 종종 수첩에 기록해두고 수첩을 보면서 해당 내용을 떠올립니다. 이처럼 프로그래밍을 할 때도 어떤 값 자체를 기억하는 것이 아니라 어떤 이름을 통해 해당 값을 가리키게 해두고 싶습니다. 이 같은 상황에서 사용하는 것이 바로 '변수'입니다.

먼저 파이썬 IDLE 를 실행한 후 프롬프트에 다음과 타이핑하고 엔터키를 눌러 봅시다.

```
>>> naver = 601000
```

위 코드의 의미는 그림 2.3 처럼 'naver 라는 이름(변수)이 601000 이라는 값을 가리키게 하라'는 것입니다. 산수에서는 등호 표시(=)가 '같음'을 의미하지만 파이썬에서는 바인딩(binding)을 의미합니다. 참고로 바인딩이라는 단어는 '가리키는 것'을 의미합니다.

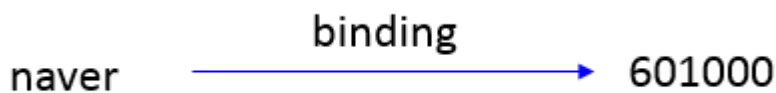


그림 2.3 파이썬 바인딩의 의미

컴퓨터에서 처리되는 데이터는 메모리의 어딘가에 실제로 존재하게 됩니다. 즉, 위의 예에서 601000 이라는 값 역시 메모리의 어딘가에 위치하는데, 그 위치를 naver 라는 변수가 가리키고 있는 것입니다.

저자가 변수의 개념을 우리의 일상생활과 관련 지어 좀 더 쉽게 설명해보겠습니다. 여러분이 친구 집으로 편지를 보내는 상황을 생각해봅시다. 편지를 보낼 때 '받는 주소' 부분에 친구의 주소를 기록해 두면 집배원은 그 주소를 통해 친구의 집을 정확히 찾아갈 수 있습니다.

파이썬의 변수도 마찬가지입니다. 어떤 데이터가 있을 때 그 데이터가 메모리 상에 위치하는 주소를 변수라는 곳에 저장해두고, 나중에 변수에 저장된 메모리 상의 주소에 가서 실제 값을 읽을 수 있는 것입니다.

아직도 변수가 잘 이해되지 않는 분들도 있겠지만 일단 우리에게 중요한 것은 이제 당분간은 네이버의 시가인 601,000 원을 잊고 살아도 된다는 사실입니다. 해당 값을 알고 싶으면 프롬프트 (>>>)에 naver 라고 입력하면 그림 2.4 와 같이 값이 리턴(return)되기 때문입니다.

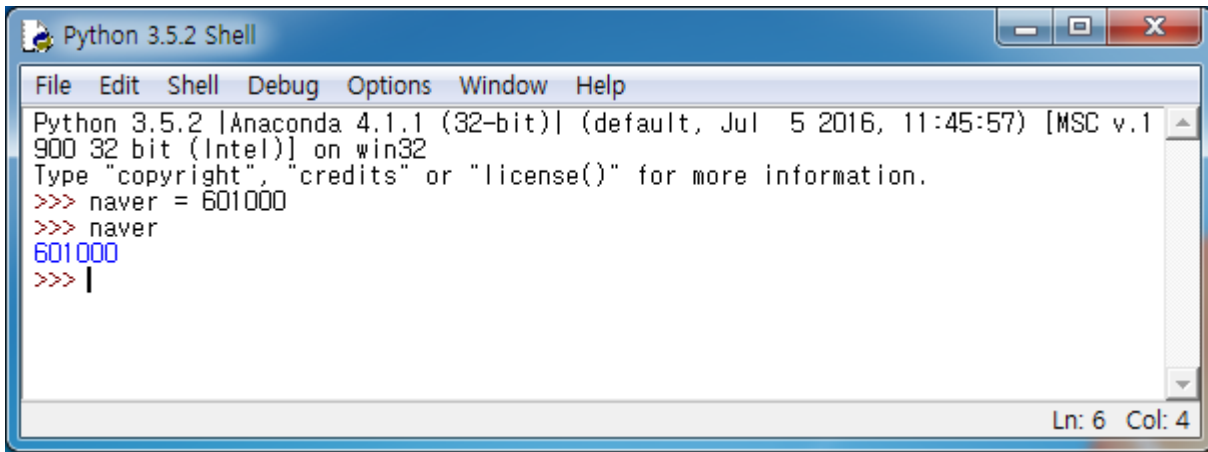


그림 2.4 변수를 통한 값 확인

아직도 변수의 편리함을 모르겠다는 분들은 파이썬 IDLE의 프롬프트에 다음과 같이 입력한 후 엔터키를 눌러 보기 바랍니다. 이 코드는 네이버 주식 10주를 601,000 원에 모두 매도했을 때의 매도 총액을 계산해줍니다. 어떨까요? 601,000 원이라는 값을 직접 사용하는 것보다 변수를 사용하니 값 자체를 기억할 필요도 없고 매우 편리하죠?

```
>>> naver * 10
```

```
6010000
```

```
>>>
```

파이썬에서 변수를 사용할 때 한 가지 중요한 점은 변수의 이름을 잘 정해야 한다는 것입니다. 파이썬에서 변수란 어떤 값을 대신 기억하기 위해 값에 이름표를 붙여둔 것과 비슷한데, 값과 상관없는 이름표를 붙여두면 더 불편하기 때문입니다. 예를 들어, '콜라'에 '간장'이라는 이름표를 붙여 두고, '간장'에 '콜라'라는 이름을 붙여두면 나중에 둘을 구분하기가 매우 어려워지겠죠?

파이썬에서 변수의 이름을 정할 때는 몇 가지 규칙이 있습니다. 예를 들어, 변수의 첫 글자는 영어 또는 언더스코어(_)로 시작해야 하고, 숫자로 시작할 수 없습니다. 즉, 그림 2.5와 같이 변수를 선언하면 오류가 발생하고 파이썬 코드가 실행되지 않게 됩니다.

```
>>> 1naver = 601000
SyntaxError: invalid syntax
>>> |
```

그림 2.5 변수 선언 오류 예

참고로 앞에서 영어 또는 언더스코어(_)로 시작해야 한다고 했지만 사실 한글로도 변수명을 만들 수는 있습니다. 다만 일반적으로 프로그래밍할 때는 한글 변수명보다는 영어 소문자와 언더스코어(_)를 조합해서 변수명을 만드는 것이 좋습니다.

```
>>> 네이버 = 601000
```

```
>>> 네이버
```

```
601000
```

```
>>> naver_juga = 601000
```

```
>>> naver_juga
```

```
601000
```

```
>>>
```

2-1) 파이썬 변수와 객체

보통 프로그래밍 언어를 배울 때 변수라는 개념을 가장 처음으로 배우는데, 프로그래밍에 입문하는 많은 분들이 이 단계에서 쉽게 좌절하곤 합니다. 왜냐하면 변수라는 개념 자체도 이해되지 않고 왜 사용하는지를 모르기 때문에 프로그래밍에 대한 흥미를 잃어버리곤 합니다.

저자는 변수의 개념을 다시 한 번 강조하기 위해 간단한 예제를 통해 여러분과 같이 프로그래밍해보겠습니다. 네이버의 월요일 주가가 10,000 원에서 시작했는데, 월요일과 화요일에 연속해서 하한가 (-30%)를 기록했을 때 화요일의 종가를 계산해 봅시다. 단 화요일의 '시작가'가 월요일의 종가와 같다고 가정합니다.

저자가 위 문제를 풀어본다면 아마 다음과 같이 월요일의 종가를 먼저 어딘가에 계산한 후 해당 가격을 이용해 다시 화요일에 하한가를 적용해 최종적으로 화요일의 종가를 계산할 것입니다.

```
월요일 종가 = 10,000 - (10,000 * 0.3) = 7,000
```

```
화요일 종가 = 7,000 - (7,000 * 0.3) = 4,900
```

일상생활에서 어떤 복잡한 값을 계산할 때 보통은 위와 같이 중간 과정의 결과값을 어딘가에 기록해 두고, 조금씩 문제를 해결해 나갑니다. 마찬가지로 프로그램을 작성할 때도 의미가 있는 값을 저장해두거나 해당 값들을 가리키는 역할을 하는 것이 바로 변수입니다.

위에서 간단히 손으로 계산했던 것을 파이썬 코드로 옮기면 다음 코드와 같습니다. 해당 코드를 보면 저자가 변수의 이름을 정할 때 고심한 흔적인 보이나요? 이처럼 변수명이 조금 길더라도 나중에라도 해당 변수가 가리키는(바인딩하는) 값이 무엇인지 쉽게 알 수 있게 만드는 것이 매우 중요합니다.

```
>>> monday_end_price = 10000 - (10000 * 0.3)
```

```
>>> tuesday_end_price = monday_end_price - (monday_end_price * 0.3)
```

```
>>> tuesday_end_price
```

```
4900.0
```

```
>>>
```

이제 변수의 필요성을 어느 정도 느끼셨을 테니 파이썬의 변수와 바인딩에 대해 좀 더 자세히 알아보겠습니다. 다음 코드를 파이썬 IDLE 에서 실행시켜 보기 바랍니다.

```
>>> x = 100
```

```
>>>
```


파이썬 IDLE 에서 위 코드를 실행했을 때 실제로는 어떤 일이 벌어질까요? 먼저 '100'이라는 값이 메모리의 어딘가에 할당되는데, 파이썬에서는 이를 객체(object)라고 부릅니다. 변수 'x' 역시 메모리의 어딘가에 할당되는 공간인데, 그 공간에는 '100'이라는 객체의 메모리 주솟값이 저장돼 있습니다. 이를 다르게 표현하자면 변수 x는 100이라는 값의 객체를 가리키고 있는 것입니다.

변수와 객체의 관계를 일상생활과 연관 지어 설명해보면 여러분이 살고 계시는 집이 바로 객체입니다. 실제로 존재하는 것이죠. 이와 달리 주소 자체는 집은 아니지만 다른 누군가가 여러분 집의 주소를 통해 여러분이 실제로 사는 집에 갈 수 있습니다. 파이썬도 마찬가지입니다. 변수를 통해 실제 객체에 접근할 수 있는 것입니다.

위 코드에서 변수 'x'와 객체 '100'은 그림 2.6 과 같은 관계를 맺습니다. 그림 2.6 에서 객체는 네모 박스로 표시했고 변수는 동그라미로 표시했습니다. 변수는 "집 주소"로 생각하면 되고, 객체는 실제 여러분이 살고 있는 집으로 생각하면 좀 더 쉽게 이해할 수 있습니다.

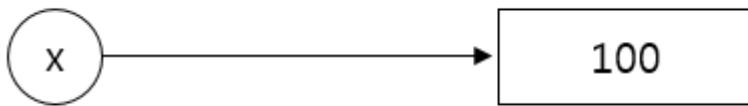


그림 2.6 변수와 객체의 바인딩 예

그렇다면 파이썬에서 메모리에 할당된 객체의 주소(정확히 말하면 주소는 아니지만 의미상으로는 주소로 생각해도 됩니다)는 어떻게 확인할 수 있을까요? 다음과 같이 id()라는 함수를 사용하면 됩니다.

```
>>> x = 100
>>> id(x)
1773787088
>>>
```

여기까지 이해하신 분들은 다음 코드에서 id(x)와 id(y) 값이 같을지 또는 다를지 생각해보기 바랍니다.

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

예제 2.11 을 보면 먼저 메모리에 100이라는 값이 할당되고, 그 주소를 x가 가리킵니다. 그리고 다음 줄에 보면 또 100이 나오는데, 이때 다음과 같이 두 가지 방식이 있을 것입니다.

1) 메모리에 조금 전의 100 과 다른 위치에 100 을 할당하고, 그 주솟값을 y 가 가리킨다(메모리에 두 개의 '100'이 존재).

2) 메모리가 아까우니 조금 전의 100 의 주소를 y 가 가리키게 한다(메모리에 100 은 실제로 하나만 존재).

위 코드를 실행 결과값을 보면 id 값이 같습니다. 즉, 두 변수가 서로 같은 객체를 가리키고 있음을 확인할 수 있습니다. 이를 그림으로 나타내면 그림 2.7 과 같습니다. 파이썬은 위 코드에 대해 2 번과 같은 방식으로 동작했던 것입니다.

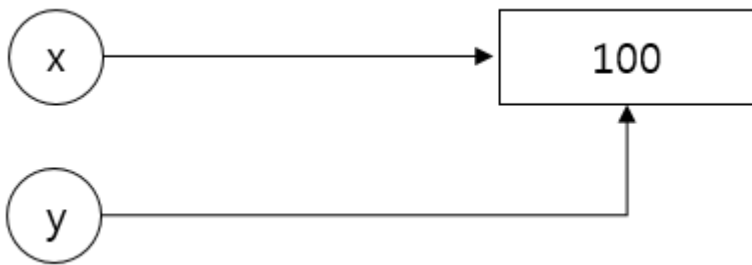


그림 2.7 변수와 객체의 바인딩 예(1)

단, 파이썬이 항상 2 번 방식으로 동작하는 것은 아닙니다. 다음 코드를 실행시켜보면 id 값이 서로 다르게 나오는 것을 확인할 수 있습니다.

```
>>> x = 10000
```

```
>>> y = 10000
```

```
>>> id(x), id(y)
```

```
(44212048, 48072352)
```

```
>>>
```

위 코드에서 변수와 객체의 관계를 그림으로 나타내면 그림 2.8 과 같습니다. 변수 x와 변수 y 는 서로 다른 객체를 가리키고 있습니다. 그림 2.8 에서 파란색으로 나타낸 부분은 객체의 id 값을 의미합니다. 참고로 위 코드를 실행했을 때 반환되는 id 값은 실행할 때마다 달라질 수 있으므로 저자와 여러분의 값이 서로 다를 수 있습니다.

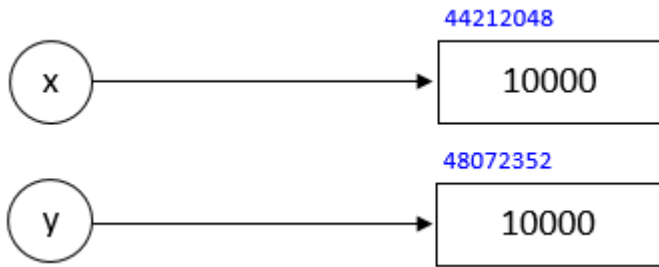


그림 2.8 변수와 객체의 바인딩 예(2)

참고로 파이썬이 이렇게 동작하는 이유는 프로그램을 작성할 때 사용하는 정숫값(Integer) 중 자주 사용할 것 같은 범위의 정숫값은 메모리에 한 번만 올려두고 이를 여러 변수가 가리키게 함으로써 메모리를 효과적으로 사용하기 위해서입니다. 심심하신 분들은 파이썬 IDLE 를 통해 위와 같은 방식으로 숫자에 대해 x, y 라는 변수로 바인딩해보면 256 까지는 id 값이 같지만 257 부터는 서로 다른 객체가 생성되는 것을 확인할 수 있습니다.

3) 파이썬 문자열

2.2 절에서 변수를 이용해 메모리에 할당된 어떤 정숫값을 가리키는 것(바인딩)을 배웠습니다. 이번 절에서는 변수로 문자열을 가리키는 것에 대해 알아보도록 하겠습니다.

C/C++과 같은 프로그래밍 언어에서는 'a', 'b', 'c'와 같은 알파벳 글자 하나를 '문자'라고 부르며, 'house'와 같이 두 개 이상의 문자로 구성된 것을 '문자열(String)'이라고 합니다. 다른 프로그래밍 언어와 달리 파이썬은 문자와 문자열을 구분하지 않고 작은따옴표(')나 큰따옴표(")로 묶인 문자의 모음을 문자열이라고 부릅니다.

예를 들어, 다음의 세 개의 구문은 모두 변수가 파이썬 문자열을 바인딩하는 예 입니다. 앞서 설명한 것처럼 파이썬은 문자와 문자열을 구분하지 않고 모두 문자열로 취급합니다. 이때 한 가지 주의해야 할 점은 작은따옴표로 문자열을 시작한 경우는 반드시 작은따옴표로 끝나야 하고, 큰따옴표로 시작한 경우 반드시 큰따옴표로 끝나야 한다는 것입니다. 즉, 문자열의 시작과 끝에 서로 다른 종류의 따옴표를 사용할 수 없습니다.

```
>>> mystring = 'hello world'
```

```
>>> mystring1 = 'a'
```

```
>>> mystring2 = "a"
```

```
>>> mystring3 = "abc mart"
```

위 코드처럼 변수가 문자열을 가리키고 있으면 변수명을 통해 해당 문자열을 참조 할 수 있습니다. 일단 mystring 이라는 변수가 문자열 객체를 잘 가리키고 있는지 확인하기 위해 파이썬 프롬프트에 mystring 을 타이핑하고 엔터 키를 눌러봅시다.

```
>>> mystring
```

```
'hello world'
```

```
>>>
```

'hello world'라고 정상적으로 값이 나오는 것을 확인할 수 있습니다. 이번에는 화면에 값을 출력하는 함수인 print 를 통해 mystring 이 가리키고 있는 값을 출력해 보겠습니다

```
>>> print(mystring)
```

```
hello world
```

```
>>>
```

3-1) 문자열 인덱싱 및 슬라이싱

파이썬은 다른 프로그래밍 언어에 비해 문자열을 매우 쉽게 다룰 수 있습니다. 먼저 'hello world'라는 문자열에 대해 총 몇 글자가 있는지 알아보겠습니다.

다음 코드를 실행하면 '11'이라는 값이 반환되는데 이는 'hello world'라는 문자열이 총 11 개의 글자로 구성되어 있다는 의미입니다. 참고로 len() 함수는 length 의 줄임말입니다. 그런데 왜 10 자가 아니라 11 자일까요? 그것은 'hello'와 'world'라는 단어 사이에 있는 공백도 하나의 문자로 간주하기 때문입니다.

```
>>> mystring = 'hello world'
```

```
>>> len(mystring)
```

```
11
```

```
>>>
```

'hello world'라는 문자열 중에서 'hello'라는 글자만 구하고 싶다면 어떻게 하면 될까요? 파이썬에서는 '슬라이싱'이라는 기능을 제공하는데 다음과 같이 가져오고 싶은 문자열의 범위를 지정하면 됩니다. 참고로 slice 라는 영어 단어는 '자르다', '일부', '한 조각'과 같은 의미가 있습니다.

```
>>> mystring[0:5]
```

```
'hello'
```

```
>>>
```

위 코드를 보면 [0:5]라는 표현을 사용했습니다. []는 슬라이싱할 범위를 지정할 때 사용하는 기호이며, 0 은 시작 위치, 5 는 끝 위치를 의미합니다. 시작과 끝을 구분하기 위해 그 사이에 콜론(:)을 사용합니다.

참고로 프로그래밍 언어에서는 우리의 일상생활과 달리 범위가 1 부터 시작하는 것이 아니라 0 부터 시작합니다. 그래서 위의 예제에서 시작 값이 1 이 아니라 0 으로 돼 있는 것입니다. 그림 2.9 는 'hello world'라는 문자열의 각 글자 사이에 인덱스를 붙여본 것입니다.

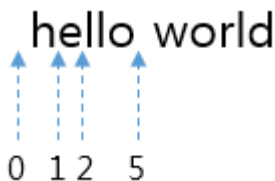


그림 2.9 문자열 인덱싱 예

파이썬 문자열의 인덱싱을 조금 더 응용해 보겠습니다. 이번에는 mystring 이라는 변수를 통해 'world'만 가져오려면 어떻게 해야 할까요? 정답은 다음과 같습니다.

```
>>> mystring[6:11]
```

```
'world'
```

```
>>>
```

위 코드의 문제점은 맨 마지막 글자의 위치를 세기가 참으로 귀찮다는 것입니다. 다행히도 파이썬의 슬라이싱에서는 시작값 또는 끝값을 생략하면 알아서 해당 문자열의 시작과 끝을 의미합니다. 즉, 다음과 같이 작성할 수 있습니다.

```
>>> mystring[6:]
```

```
'world'
```

```
>>>
```

hello 를 구하는 코드도 시작값을 생략하여 다음과 같이 작성할 수 있습니다.


```
>>> mystring[:5]
```

```
'hello'
```

```
>>>
```

파이썬의 문자열 인덱싱에서 인덱스 값으로 양수만 사용할 수 있는 것은 아니고 그림 2.10 에서처럼 음수를 사용할 수도 있습니다. 인덱스 값이 음수인 경우에는 문자열의 뒤쪽부터 역순으로 글자를 셉니다.

hello world



-2 -1

그림 2.10 음수 인덱싱 예

음수 인덱싱을 응용하면 다음 코드처럼 문자열에 접근할 수 있습니다. 그런데 아쉽게도 'world'에서 'd'가 빠져버렸네요.

```
>>> mystring[6:-1]
```

```
'worl'
```

```
>>>
```

3-2) 문자열 자르기

저자가 몇 년 전에 네이버와 다음 주식을 가지고 있었습니다. 그래서 my_jusik 이라는 변수가 "naver daum"이라는 문자열 객체를 가리키게 해 봤습니다(중간에 공백이 하나 있습니다).

```
>>> my_jusik = "naver daum"
```

```
>>>
```

파이썬 IDLE 에 my_jusik 이라고 입력한 후 엔터 키를 눌렀더니 'naver daum'이라고 문자열이 잘 반환됩니다. 여기서 한 가지 팁을 드리면 파이썬 IDLE 에서 my_jusik 을 모두 입력하지 않고 'my_' 정도만 입력한 상태에서 키보드의 탭(Tab) 키를 누르면 자동으로 변수명이 완성됩니다. 이러한 자동 완성 기능은 변수명이 긴 경우에 유용하게 활용할 수 있습니다.

```
>>> my_jusik
```

```
'naver daum'
```

```
>>>
```

my_jusik 이라는 변수가 가리키는 문자열 객체에서 'naver'만 출력하려면 어떻게 해야 할까요? 즉, naver 와 daum 이 현재는 하나의 문자열인데 이를 분리하고 싶은 것입니다. 2.3.1 절에서 배운 문자열 슬라이싱으로도 문자열을 분리할 수 있지만 파이썬에서는 문자열을 쉽게 분리하기 위해 split 라는 메서드를 지원합니다. 참고로 split 라는 단어는 '나누다', '쪼개다'라는 뜻이 있습니다.

```
>>> my_jusik.split(' ')
```

```
['naver', 'daum']
```

```
>>>
```

파이썬에서 문자열을 분리하고 싶을 때는 위 코드와 같이 split 메서드를 사용하면 됩니다. 이때 메서드의 인자(argument)로 어떤 문자를 기준으로 문자열을 나눌지 알려줘야 합니다. 'naver'와 'daum'이라는 문자열 사이에는 공백(blank)이 존재하므로 공백을 기준으로 문자열을 나눠주면 됩니다. 이를 위해 split(' ')와 같이 작은따옴표 사이에 공백을 하나 넣어줬습니다.

그러나 위 코드에서 반환값을 보면 여전히 우리가 원하는 값은 아닙니다. 반환값을 보면 'naver'와 'daum'이라는 두 개의 문자열이 '[' 와 ']'로 둘러싸여 있음을 확인할 수 있습니다. 파이썬에서는 이것을 리스트(list)라고 부릅니다(리스트에 대해서는 3 장에서 다룹니다).

이번 장에서는 리스트 내의 값에 접근할 때 [0], [1], [2]와 같이 인덱스 값을 주면 된다는 것만 기억하면 됩니다. 리스트의 인덱싱 기능을 이용하여 다음과 같이 코드를 수정해 보겠습니다. split(' ')의 반환값이 리스트이므로 [0]을 사용해서 리스트의 첫 번째 값을 가져왔습니다.

```
>>> my_jusik.split(' ')[0]
```

```
'naver'
```

```
>>>
```

화면에 출력하는 함수는 print 였죠? 이번에는 출력 함수까지 꼭 붙여서 써보겠습니다.

```
>>> print(my_jusik.split(' ')[0])
```

```
naver
```

```
>>>
```

위 코드에서는 한 번에 붙여서 썼더니 가독성이 조금 떨어진 것 같습니다. 다음과 같이 변수를 이용해 중간값을 바인딩해서 좀 더 이해하기 쉽게 코드를 작성해 봅시다.

```
>>> split_jusik = my_jusik.split(' ')
```

```
>>> split_jusik
```

```
['naver', 'daum']
```

```
>>> print(split_jusik[0])
```

```
naver
```

```
>>>
```


3-3) 문자열 합치기

2014 년 6 월 카카오(Kakao)와 다음(Daum)이 합병했습니다. 그 당시 합병된 회사의 이름은 여러분도 잘 아시다시피 'Daum KaKao' 였습니다. 이를 파이썬 문자열로는 어떻게 표현할 수 있을까요?

먼저 daum 이라는 변수가 'Daum'이라는 문자열 객체를 가리키게 해봅시다. 그리고 kakao 라는 변수는 'KAKAO'라는 문자열 객체를 가리키게 해봅시다.

```
>>> daum = "Daum"
```

```
>>> kakao = "KAKAO"
```

```
>>>
```

이제 두 변수를 더해봅시다. 정수를 더할 때 사용하는 '+' 기호를 그대로 사용해 다음과 같이 작성하면 됩니다.

```
>>> daum + kakao
```

```
'DaumKAKAO'
```

```
>>>
```

'DaumKAKAO'라는 이름 사이에 공백이 없으니 뭔가 허전합니다. 읽기도 어렵고 짝퉁 회사가 된 것 같습니다. 문자열을 더할 때 두 문자열 사이에 공백을 하나 넣어보겠습니다. 문자열을 합치는 것은 그냥 숫자를 더하듯이 여러 문자열을 그냥 + 기호로 더하면 됩니다. 공백을 나타내고 싶으면 작은따옴표 사이에 공백을 넣어서 표현하면 됩니다.

```
>>> daum + ' ' + kakao
```

```
'Daum KAKAO'
```

```
>>>
```

이름을 변수로 가리켜 두지 않으면 'Daum KAKAO'라는 문자열 객체를 사용할 때마다 두 문자열을 합치는 귀찮은 작업을 해야 합니다. 다음과 같이 합쳐진 문자열을 다른 변수가 가리키게 해줍니다.

```
>>> daum_kakao = daum + ' ' + kakao
```

```
>>> daum_kakao
```

'Daum KAKAO'

> > >

이제 앞으로는 'Daum KAKAO'라는 이름을 출력할 때 'daum_kakao'라는 변수를 사용하면 됩니다.

4) 파이썬 기본 데이터 타입

지금까지 정수, 실수, 문자열에 대해 배웠습니다. 이러한 정수, 실수, 문자열은 파이썬 프로그래밍에서 자주 사용되는데, 이를 기본 데이터 타입이라고 합니다.

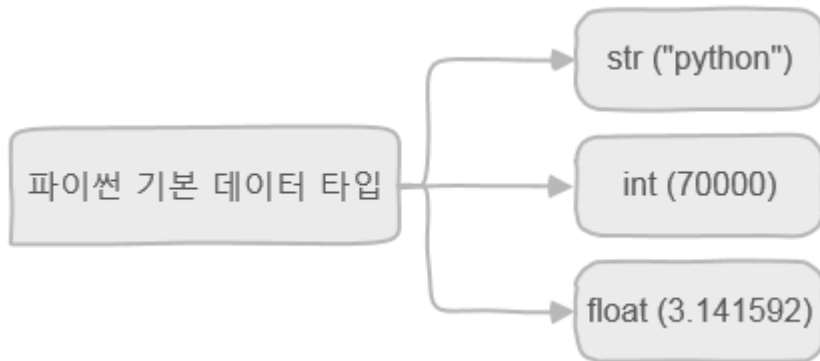


그림 2.11 파이썬 기본 데이터 타입

파이썬에서 어떤 값(객체)에 대한 데이터 타입을 확인하려면 `type()`이라는 내장 함수를 사용하면 됩니다. 다음 코드를 파이썬 IDLE 을 통해 실행해보기 바랍니다.

```
>>> type(70000)
<class 'int'>
>>>
```

위 코드의 실행 결과를 살펴보면 `type()` 함수의 반환값이 임을 알 수 있습니다. 여기서 `class` 라는 단어는 잠시 잊고, `int` 에 주목하기 바랍니다. `int` 는 `integer` 의 약자로서 정수를 의미합니다. 참고로 산수가 약하신 분들을 위해 설명해 드리면 1, 2, 3, ... 등을 정수라고 하고, 1.234, 1.432 등을 실수라고 합니다. 즉, 우리가 알고 있는 정수 70000 에 대해 `type()` 함수를 이용해 파이썬에게 타입을 물어봤더니 파이썬이 'int'라고 답한 것입니다.

파이썬에는 정숫값 말고도 실숫값도 있으니 이 경우에도 `type` 함수가 잘 동작하는지 확인해 보겠습니다. 3.141592 라는 값에 대한 `type()` 함수의 반환값을 보니 'float'입니다. `float` 의 사전적 의미는 '뜨는 것', '둥둥 떠다니는 것'을 의미합니다. 프로그래밍에서 `float` 는 부동 소수점(실수)를 의미합니다.

```
>>> type(3.141592)
<class 'float'>
>>>
```

파이썬의 기본 데이터 타입으로 정수와 실수 외에도 문자열이 있죠? 문자열에 대해서도 타입을 확인해 보겠습니다.

```
>>> type('python')
```

```
<class 'str'>
```

```
>>> type('jusik')
```

```
<class 'str'>
```

```
>>>
```

앞에서 배운 것처럼 작은따옴표나 큰따옴표로 둘러싸인 값에 대해 'str'이라는 값이 반환되는 것을 볼 수 있습니다. 참고로 str은 영어 단어에서 문자열을 의미하는 'string'에서 앞 세 글자를 딴 것입니다.

1) 연습문제

문제 2-1 다음(Daum)의 주가가 89,000 원이고 네이버(Naver)의 주가가 751,000 이라고 가정하고, 어떤 사람이 다음 주식 100 주와 네이버 주식 20 주를 가지고 있을 때 그 사람이 가지고 있는 주식의 총액을 계산하는 프로그램을 작성하세요.

문제 2-2 문제 2-1 에서 구한 주식 총액에서 다음과 네이버의 주가가 각각 5%, 10% 하락한 경우에 손실액을 구하는 프로그램을 작성하세요.

문제 2-3 우리나라는 섭씨 온도를 사용하는 반면 미국과 유럽은 화씨 온도를 주로 사용합니다. 화씨 온도(F)를 섭씨 온도(C)로 변환할 때는 다음과 같은 공식을 사용합니다. 이 공식을 사용해 화씨 온도가 50 일 때의 섭씨 온도를 계산해 보세요.

$$C = (F - 32) / 1.8$$

문제 2-4 화면에 "pizza"를 10 번 출력하는 프로그램을 작성하세요.

문제 2-5 월요일에 네이버의 주가가 100 만 원으로 시작해 3 일 연속으로 하한가(-30%)를 기록했을 때 수요일의 증가를 계산해 보세요.

문제 2-6 다음 형식과 같이 이름, 생년월일, 주민등록번호를 출력하는 프로그램을 작성해 보세요. 이름: 파이썬 생년월일: 2014 년 12 월 12 일 주민등록번호: 20141212-1623210

문제 2-7 s 라는 변수에 'Daum KaKao'라는 문자열이 바인딩돼 있다고 했을 때 문자열의 슬라이싱 기능과 연결하기를 이용해 s 의 값을 'KaKao Daum'으로 변경해 보세요.

문제 2-8 a 라는 변수에 'hello world'라는 문자열이 바인딩돼 있다고 했을 때 a 의 값을 'hi world'로 변경해 보세요.

문제 2-9 x 라는 변수에 'abcdef'라는 문자열이 바인딩돼 있다고 했을 때 x 의 값을 'bcdefa'로 변경해 보세요.