

2장. 변수와 계산

변수 생성

- 변수를 만들려면?

전체적인 구조



변수이름 = 값

파이썬에서 변수는 값이 할당되는 순간에 생성됩니다.

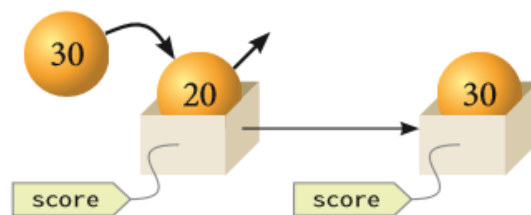


```
>>> score = 20
>>> score
20
>>> print(score)
20
```

변수의 사용

- 생성된 변수에는 얼마든지 다른 값을 저장할 수 있다.

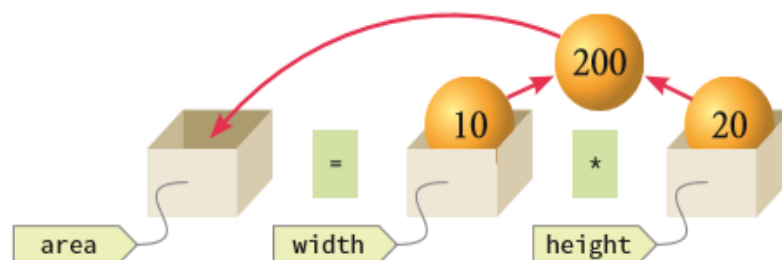
```
>>> score = 20  
>>> score = 30  
>>> score  
30
```



변수의 사용

- 변수에는 다른 변수의 값도 저장할 수 있다.

```
>>> width = 10  
>>> height = 20  
>>> area = width * height  
>>> print(area)  
200
```



변수의 사용

- 파이썬의 변수에는 정수뿐만 아니라 문자열도 저장할 수 있다.

```
>>> s = '안녕하세요?'
```

```
>>> print(s)
```

```
안녕하세요?
```

```
>>> pi = 3.141592
```

```
>>> print(pi)
```

```
3.141592
```

Lab: 파티 준비

- 참석자에 맞추어서 치킨(1인당 1마리), 맥주(1인당 2캔), 케익(1인당 4개)를 출력하는 프로그램을 작성해보자.

참석자의 수를 입력하시오: 25

치킨의 수: 25

맥주의 수: 50

케익의 수: 100



Solution

```
number = int(input("참석자의 수를 입력하시오:"))  
chickens = number  
beers = number*2  
cakes = number*4  
print("치킨의 수: ", chickens)  
print("맥주의 수: ", beers)  
print("케익의 수: ", cakes)
```

변수가 저장하는 것

- 파이썬에서 변수는 어떤 데이터든지 저장할 수 있다.

```
value = 3  
value = 3.14  
value = "hello"
```

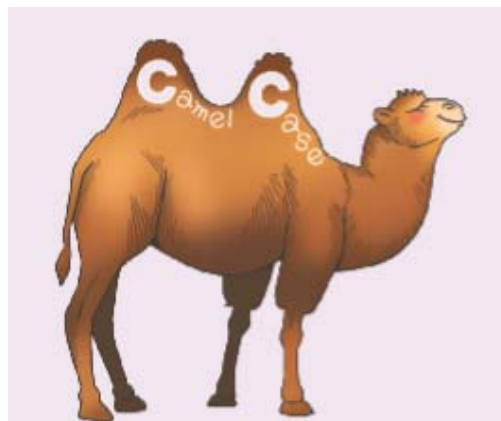
변수의 이름

- 의미 있는 이름을 사용
- 소문자와 대문자는 서로 다르게 취급된다.
- 변수의 이름은 영문자와 숫자, 밑줄(_)로 이루어진다.
- 변수의 이름 중간에 공백이 들어가면 안 된다. 단어를 구분하려면 밑줄(_)을 사용 한다.



낙타체

- 낙타체는 변수의 첫 글자는 소문자로, 나머지 단어의 첫 글자는 대문자로 적는 방법이다.
- 예를 들면, **myNewCar**처럼 첫 'm'은 소문자로, 나머지 단어들의 첫 글자는 대문자로 표기한다



상수

- 상수(constant)는 한번 값이 결정되면 절대로 변경되지 않는 변수.

```
TAX_RATE = 0.35
```

```
PI = 3.141592
```

```
MAX_SIZE = 100
```

주석

- 주석(comment)은 소스 코드에 붙이는 설명글

```
# 사각형의 가로 길이  
width = 10
```

```
# 사각형의 세로 길이  
height = 20
```

```
# 사각형의 면적 계산  
area = width * height
```

주석의 또 다른 용도

```
##
```

```
# 이 프로그램은 사용자로부터 2개의 정수를 받아서  
# 합을 계산한다.
```

```
x = int(input("첫 번째 정수: "))
```

```
y = int(input("두 번째 정수: "))
```

```
sum = x + y
```

```
#diff = x - y
```

```
print("합은 ", sum)
```

코드 중 실행하고 싶지 않은 문장이 있을 때

첫 번째 정수: 10

두 번째 정수: 20

합은 30

수식과 연산자

```
>>> 3 + 4
```

```
7
```

```
>>> 3.14 * 5.0 * 5.0
```

```
78.5
```

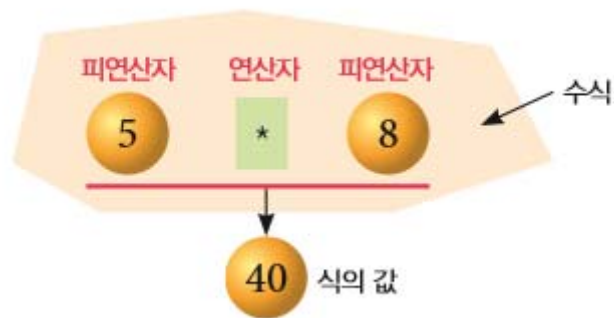


파이썬을 이용하면 계산을
할 수 있습니다.



연산자와 피연산자

- 수식(expression): 피연산자들과 연산자의 조합
- 연산자(operator): 연산을 나타내는 기호
- 피연산자(operand): 연산의 대상이 되는 것



산술 연산자

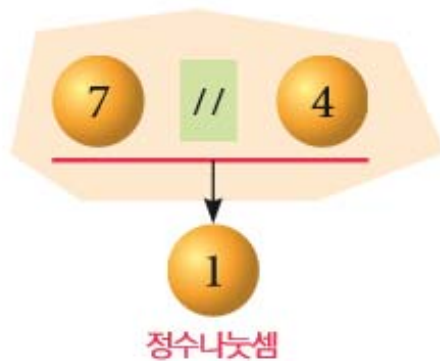
- 덧셈, 뺄셈, 곱셈, 나눗셈, 나머지 연산

연산자	기호	사용예	결과값
덧셈	+	$7 + 4$	11
뺄셈	-	$7 - 4$	3
곱셈	*	$7 * 4$	28
나눗셈	//	$7 // 4$	1
나눗셈	/	$7 / 4$	1.75
나머지	%	$7 \% 4$	3

나눗셈

```
>>> 7 / 4  
1.75
```

```
>>> 7 // 4  
1
```



지수 계산

- 지수(power)를 계산하려면 `**` 연산자를 사용한다.

```
>>> 2 ** 7  
128
```

- 원리금 계산

```
>>> a = 1000  
>>> r = 0.05  
>>> n = 10  
>>> a*(1+r)**n  
1628.894626777442
```

나머지 계산

```
>>> 7 % 4  
3
```

- 예제로 초 단위의 시간을 받아서 몇 분 몇 초인지를 계산하여 보자.

```
>>> sec = 1000  
>>> min = 1000 // 60  
>>> remainder = 1000 % 60  
>>> print(min, remainder)  
16 40
```

Lab: 변수 값 교환

- 예를 들어서 파이썬을 사용하여서 2차 함수 $y = 3x^2 + 7x + 9$ 에서 $x=2$ 일 때, 함수의 값을 계산하여 보자.

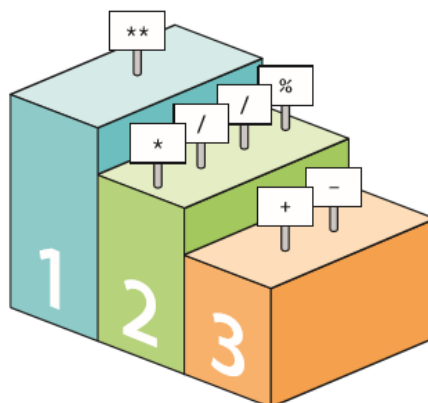
```
y = 3.0 * x**2 + 7.0 * x + 9.0
```

Solution

```
>>> x = 2.0  
>>> y = 3.0 * x**2 + 7.0 * x + 9.0  
>>> print(y)  
35.0
```

연산자의 우선 순위

```
>>> 1 + 2 * 3  
7  
>>> 4 - 40 - 3  
-39
```



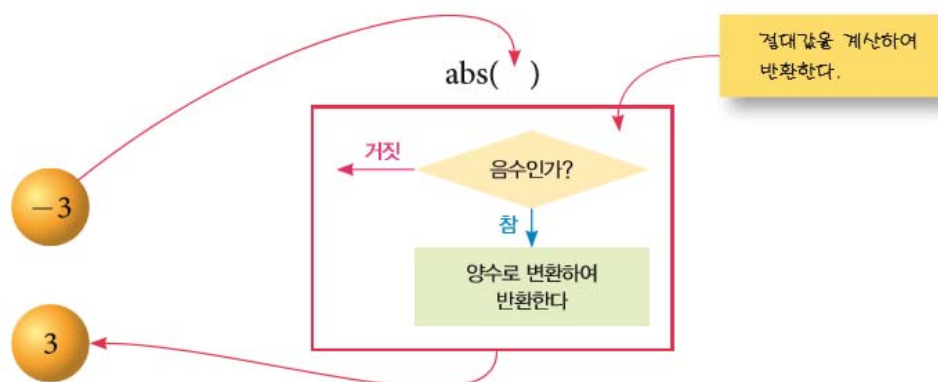
괄호의 사용

```
>>> 10 + 20 / 2  
20.0
```

```
>>> (10 + 20) / 2  
15.0
```

함수 호출

- 함수(function)란 특별한 작업을 담당하는 명령어들의 모임이다.
- 파이썬이 기본으로 제공하는 내장 함수는 상당히 많다.



내장 함수

```
>>> value = abs(-3)
>>> value
3

>>> round(1.2345)
1
>>> round(1.9876)
2

>>> max(10, 20)
20

>>> min(10, 20, 30, 40, 50)
10
```

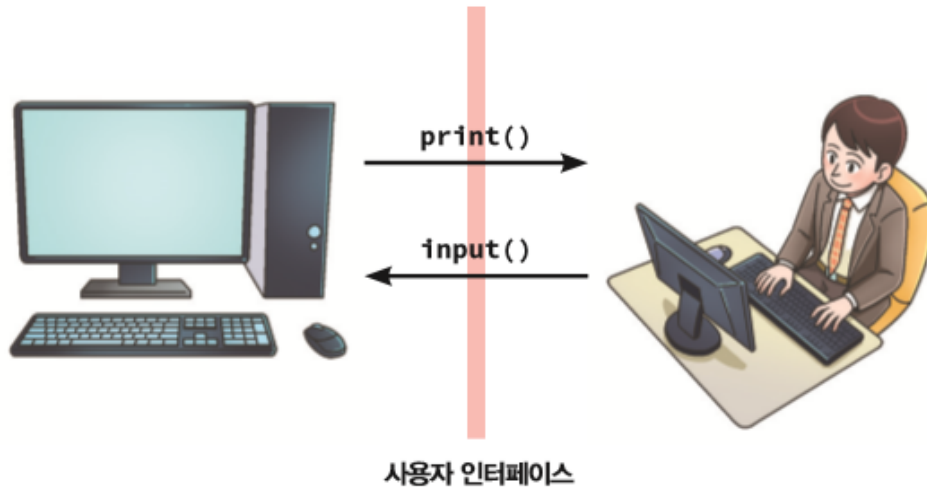
내장 함수

```
>>> from math import *           # 한번만 하면 된다.
>>> sqrt(4.0)
2.0

>>> x=2.0
>>> y=3.0
>>> sqrt(x**2+y**2)
3.605551275463989
```

input() 함수

- 사용자와의 상호작용



input() 함수

전체적인 구조



```
변수 = input( "프롬프트 문자열" )
```

프롬프트 문자열이
출력되고 사용자의 입력이
변수에 저장됩니다.



문자열 입력

```
name = input("이름이 무엇인가요? ")
print("만나서 반갑습니다. " + name + "씨!")
age = input("나이는요? ")
print("네, 그러면 당신은 이미 " + age + " 살이시군요, " + name + "씨!")
```

이름이 무엇인가요? 홍길동
만나서 반갑습니다. 홍길동씨!
나이는요? 21
네, 그러면 당신은 이미 21 살이시군요, 홍길동씨!

숫자 입력

```
x = input("첫 번째 정수: ")
y = input("두 번째 정수: ")
sum = x + y
print("합은 ", sum)
```

첫 번째 정수: 10
두 번째 정수: 20
합은 1020

문자열로 간주하여 서로 합침!

숫자 입력

```
x = int(input("첫 번째 정수: "))  
y = int(input("두 번째 정수: "))  
sum = x + y  
print("합은 ", sum)
```

첫 번째 정수: 10
두 번째 정수: 20
합은 30

자료형

- 정수(integer), 실수(floating-point), 문자열(string)

자료형	예
정수	..., -2, -1, 0, 1, 2, ...
실수	3.2, 3.14, 0.12
문자열	'Hello World!', "123"



자료형을 알고 싶으면?

```
>>> type("Hello World!")  
<class 'str'>  
>>> type(3.2)  
<class 'float'>  
>>> type(17)  
<class 'int'>
```

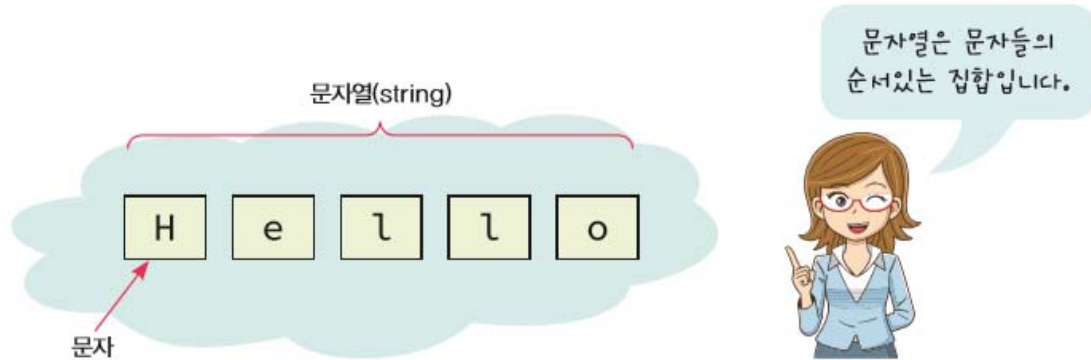
문자열

- 컴퓨터에게는 숫자가 중요하지만 인간은 주로 문자열 (string)를 사용하여 정보를 표현하고 저장하므로 문자열의 처리도 무척 중요하다.



문자열이란?

- 문자열(string)은 문자들의 순서 있는 집합(sequence of characters)



큰따옴표 사용

```
>>> greeting="Merry Christmas!"
```

```
>>> greeting
```

```
'Merry Christmas!'
```

```
>>> print(greeting)
```

```
Merry Christmas!
```

작은 따옴표 사용

```
>>> greeting = 'Happy Holiday!'
>>> print(greeting)
Happy Holiday!

>>> greeting="Happy Holiday'
SyntaxError: EOL while scanning string literal
>>>
>>> greeting="Happy Holiday
SyntaxError: EOL while scanning string literal
>>>
```

큰 따옴표 안의 작은 따옴표 사용

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

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

여러 줄의 문자열

```
>>> greeting = """지난 한해 저에게 보여주신 보살핌과  
사랑에  
깊은 감사를 드립니다.  
새해에도 하시고자 하는 일  
모두 성취하시기를 바랍니다."""
```

```
>>> print(greeting)  
지난 한해 저에게 보여주신 보살핌과 사랑에  
깊은 감사를 드립니다.  
새해에도 하시고자 하는 일  
모두 성취하시기를 바랍니다.
```

특수 문자열

- 문자 앞에 `\`가 붙으면 문자의 특수한 의미를 잃어버린다.

```
>>> message= 'doesn\'t'      # \를 사용하여 작은따옴표를 출력  
>>> print(message)  
doesn't  
>>>  
>>> message="\\"Yes,\" he said."  
>>> print(message)  
"Yes,\" he said.  
>>>
```

문자열의 연결

```
>>> 'Py' 'thon'
'Python'

>>> 'Harry ' + 'Porter'
'Harry Porter'

>>> first_name="길동"
>>> last_name="홍"
>>> name = last_name + first_name
>>> print(name)
홍길동
```

문자열과 정수 간의 변환

```
>>> "Student"+26
...
TypeError: Can't convert 'int' object to str implicitly

>>> "Student" + str(26)
'Student26'

>>> price = int("259000")
>>> height = float("290.54")
```

문자열의 반복

```
>>> line = "=" * 50
```

```
>>> print(line)
```

```
=====
```

```
>>> message = "Congratulations! "
```

```
>>> print(message * 3)
```

```
Congratulations! Congratulations! Congratulations!
```

문자열의 출력

```
>>> price = 10000
```

```
>>> print("상품의 가격은 %s원입니다." % price)
```

```
상품의 가격은 10000원입니다.
```

```
>>> message = "현재 시간은 %s입니다."
```

```
>>> time = "12:00pm"
```

```
>>> print(message % time)
```

```
현재 시간은 12:00pm입니다.
```

인덱싱

- 인덱싱(Indexing)이란 문자열에 [과]을 붙여서 문자를 추출하는 것이다.

P	y	t	h	o	n
0	1	2	3	4	5

인덱스는 문자에 매겨진
번호입니다. 0부터 시작해요!



```
>>> word = 'Python'
>>> word[0]
'P'
>>> word[5]
'n'
```

Lab: 약어 출력

- 사용자에게 단어 3개를 입력 받아서 약자(acronym: 몇 개 단어의 머리글자로 된 말)를 만들어 보자. 예를 들어서 'OST'도 Original Sound Track의 약자이다. 이 예제는 소스 파일로 작성하여 실행해보자.

첫 번째 단어를 입력해주세요: *Original*

두 번째 단어를 입력해주세요: *Sound*

세 번째 단어를 입력해주세요: *Track*

OST

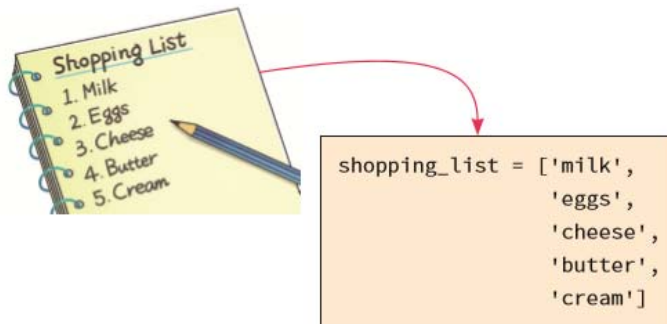
Solution

```
word1 = input('첫 번째 단어를 입력해주세요: ')
word2 = input('두 번째 단어를 입력해주세요: ')
word3 = input('세 번째 단어를 입력해주세요: ')

acronym = word1[0] + word2[0] + word3[0]
print(acronym)
```


리스트

- 파이썬은 여러 개의 값을 모아서 하나의 변수에 저장할 수 있다.
- **리스트**는 `[]` 안에 값을 나열하고 값과 값 사이에 코마를 찍으면 된다.




```
>>> shopping_list = ['milk', 'eggs', 'cheese', 'butter', 'cream']
>>> print(shopping_list)
['milk', 'eggs', 'cheese', 'butter', 'cream']
>>>
```

인덱싱

shopping_list = ['milk', 'eggs', 'cheese', 'butter', 'cream']

항목 #0 항목 #1 항목 #2 항목 #3 항목 #4



0	milk
1	eggs
2	cheese
3	butter
4	cream

리스트의 인덱스는 항상 0부터 시작합니다. 인덱스를 시작위치에서의 오프셋으로 생각하세요!



```
>>> print(shopping_list[2])
cheese

>>> shopping_list[2]='apple'
>>> print(shopping_list)
['milk', 'eggs', 'apple', 'butter', 'cream']
>>>
```

핵심 정리

- 변수의 개념을 소개하였다. 변수는 값을 저장하는 상자와 같은 것으로 변수에 저장된 값을 나중에 유용하게 사용될 수 있다.
- 다양한 산술 계산 연산자에 대하여 학습하였다. 연산자들은 우선 순위를 가지고 있지만 우리는 괄호를 사용하여 연산자의 우선 순위를 변경할 수 있었다. 지수를 계산하는 연산자는 **이다.
- 문자열은 큰따옴표나 작은따옴표를 이용하여 표현한다. `input()` 함수를 이용하여 사용자로부터 문자열을 받을 수 있다. 인덱싱 연산자 `[]`을 이용하여 각각의 문자를 추출할 수 있다.