

## 2장 변수와 수식

# 학습 목표

- 변수와 상수를 정의하고 사용할 수 있다.
- 주석의 개념을 이해한다.
- 산술 연산자와 할당 연산자에 대하여 이해한다.
- 연산자의 우선순위 개념을 이해한다.
- 사용자로부터 입력을 받고 출력을 하는 프로그램을 작성할 수 있다.
- 문자열의 기초 연산을 이해한다.



# 이번 장에서 만들 프로그램

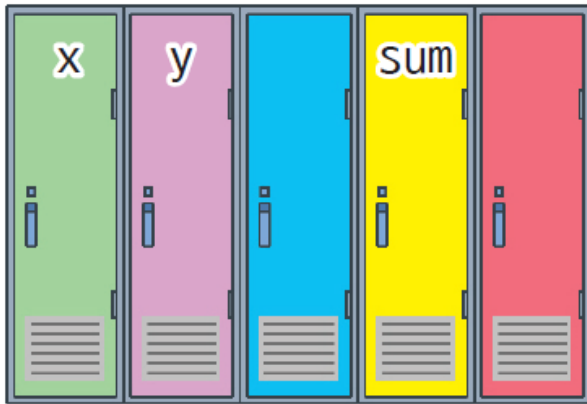
첫 번째 정수를 입력하시오: 10  
두 번째 정수를 입력하시오: 3  
10 의 3 승은 1000 입니다.

몸무게를 kg 단위로 입력하시오: 85.0  
키를 미터 단위로 입력하시오: 1.83  
당신의 BMI= 25.381468541909282

물건값을 입력하시오: 750  
1000원 지폐개수: 1  
500원 동전개수: 0  
100원 동전개수: 0  
500원= 0 100원= 2 10원= 5 1원= 0

# 변수

- 변수(variable)는 컴퓨터의 메모리 공간에 이름을 붙이는 것으로 우리는 여기에 값을 저장할 수 있다.



변수는 이름 붙인 메모리 공간으로 우리는 여기에 값을 저장할 수 있습니다.

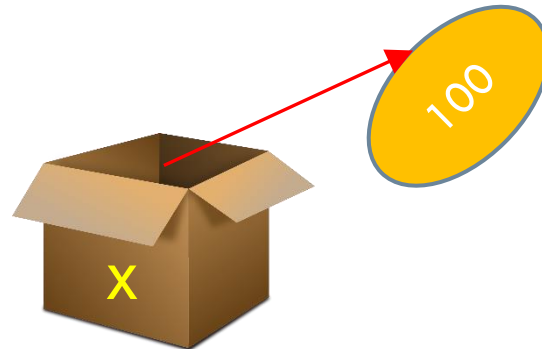
# 변수 정의하기

- 파이썬에서는 변수에 값을 저장하면 변수가 자동으로 생성된다.

Syntax: 변수정의

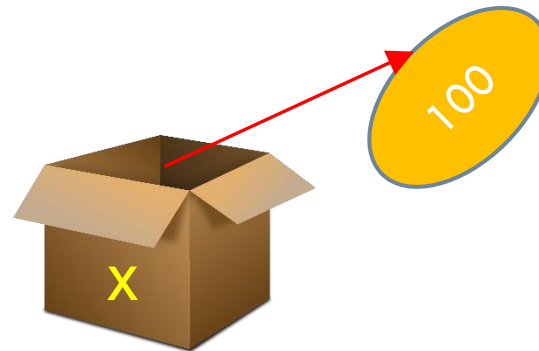
**형식**    변수이름 = 값

**예**    `x = 100`  
          ↑        ↑  
      변수이름    값



# 파이썬에서의 변수

다른 언어에서의 변수	파이썬에서의 변수
변수는 상자와 같고, 상자 안에 값이 저장된다.	데이터가 객체 형태로 메모리에 저장되고 변수에는 객체를 참조할 수 있는 값이 저장된다.

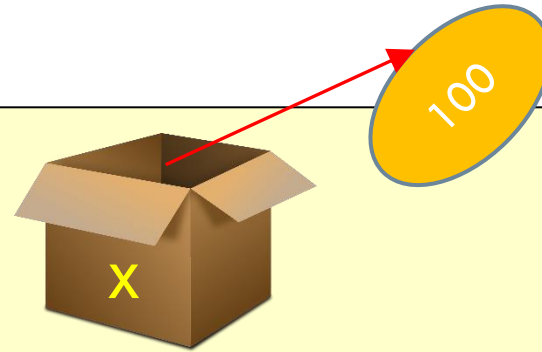


# 변수 정의하기

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

```
>>> x
```

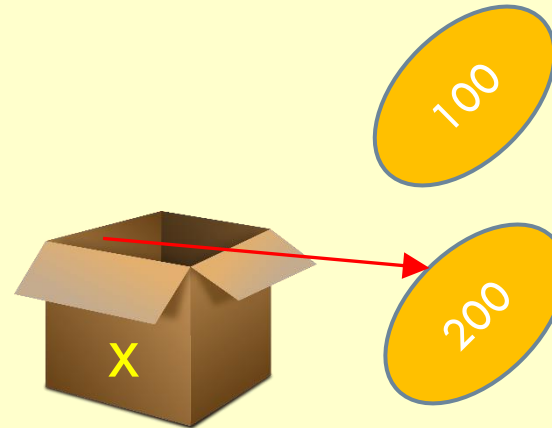
```
100
```



```
>>> x = 200
```

```
>>> x
```

```
200
```

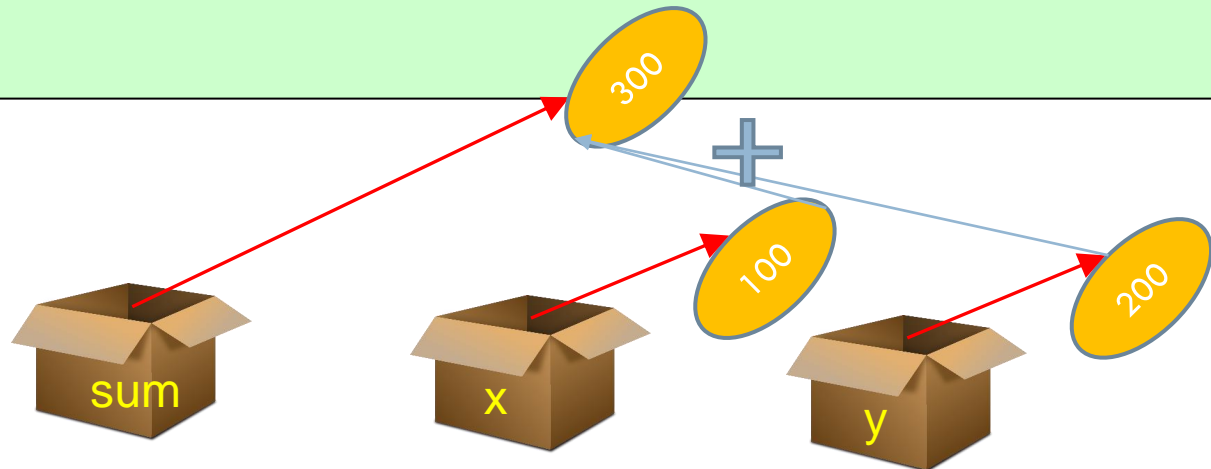


# 예제: 두 수의 합 계산하기

```
x = 100  
y = 200  
sum = x + y  
print("합은", sum)
```

```
# 변수 x를 생성하고 100을 저장한다.  
# 변수 y를 생성하고 200을 저장한다.  
# 변수 sum을 생성하고 x+y를 저장한다.
```

합은 300

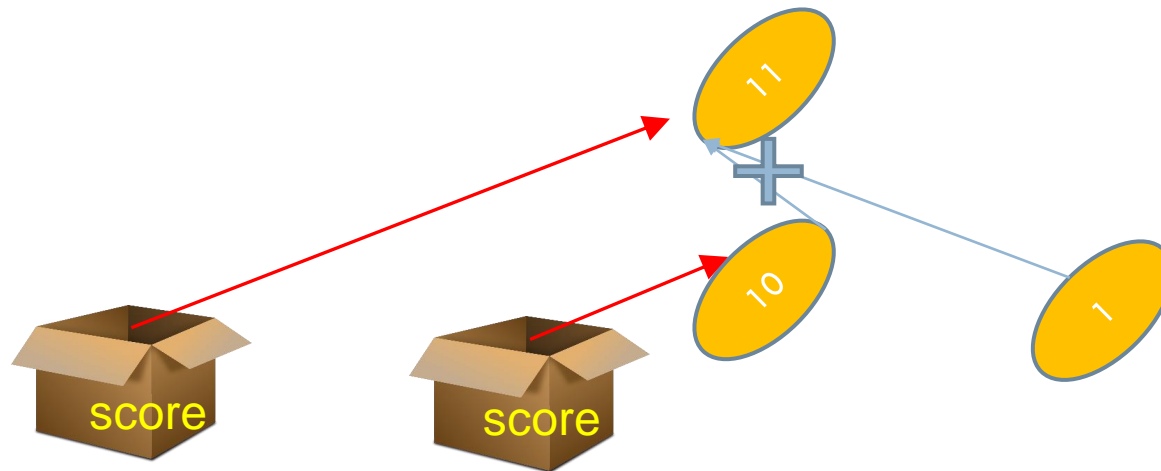




# 이런 것도 가능하다!

score = 10

score = score + 1



# 예제: 원의 면적 계산하기

- 빈칸을 채워본다.

반지름 20인 원의 면적= 1256.0

## 알고리즘

STEP #1. 사용자로부터 원의 반지름을 입력받는다.

STEP #2. 공식을 적용하여 면적을 계산한다.

$$\text{area} = \text{radius} * \text{radius} * p$$

STEP #3. 면적을 화면에 출력한다.

## 코드

```
# 변수 radius에 값을 저장한다.
```

```
radius = 10
```

```
# 공식을 적용하여 면적을 계산한다
```

```
area = 3.14 * radius * radius
```

```
# 면적을 화면에 출력한다.
```

```
print("반지름", radius, "인 원의 면적=", area)
```

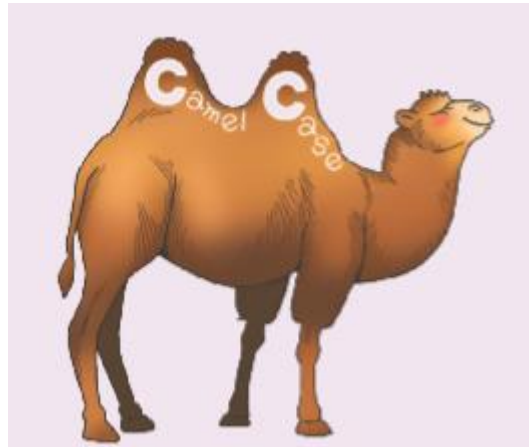
# 변수의 이름

- 소문자와 대문자는 서로 다르게 취급된다.
- 영문자와 숫자, 밑줄(\_)로 이루어진다. 숫자는 첫 글자로 불가.
- 의미 있는 이름을 사용. 예약어는 불가
- 단어를 구분하려면 밑줄(\_)을 사용 한다.

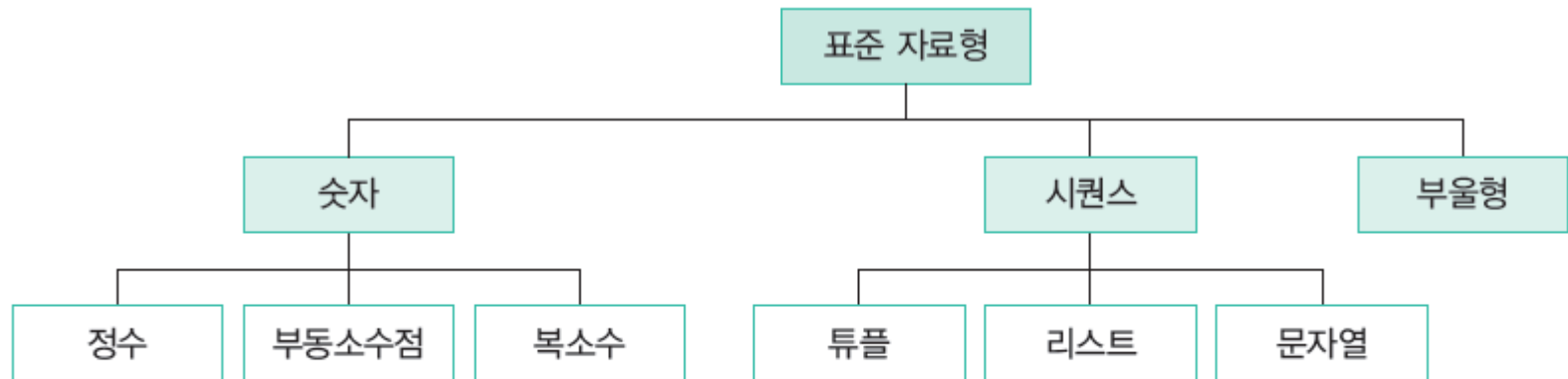
변수 이름	설명
size	가능하다.
cloud9	가능하다. 변수는 영문자, 숫자, _로 이루어진다.
max_size	가능하다. 변수의 중간에 _가 있어도 된다.
_count	가능하다. _가 앞에 붙으면 클래스 내부에서만 사용하는 변수라는 의미도 있다.
6pack	올바르지 않다! 숫자가 앞에 오면 안된다.
mid score	올바르지 않다! 중간에 공백이 있으면 안된다.
class	올바르지 않다! 예약어를 변수의 이름으로 사용할 수 없다.
money#	올바르지 않다! 기호를 변수의 이름으로 사용하면 안 된다.

# 낙타체

- 낙타체는 변수의 첫 글자는 소문자로, 나머지 단어 의 첫 글자는 대문자로 적는 방법이다.
  - ▣ myNewCar처럼 첫 'm'은 소문자로, 나머지 단어들의 첫 글 자는 대문자로 표기한다



# 자료형



자료형	예
정수(int)	..., -2, -1, 0, 1, 2 ...
부동소수점수(float)	3.2, 3.14, 0.12
문자열(str)	'Hello World!', "123"

# 변수에 어떤 자료형도 저장가능

```
>>> radius = 10
```

```
>>> radius = 10.003
```

```
>>> radius = "Unknown"
```

하지만 바람직하지는 않음!



# 자료형을 알려면?

Syntax: 변수의 자료형

**형식**    `type(수식)`

**예**    `>>> type(1234)`  
         `<class 'int'>`

```
>>> type(12.30)          # float 형
<class 'float'>

>>> type("hello")       # str(문자열) 형
<class 'str'>
```

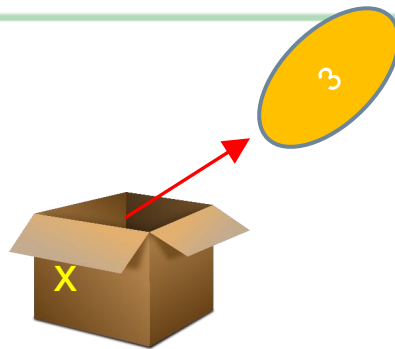
# 변수의 세부 구현 사항

- 변수에 저장되는 것은 실제 값이 아니고 객체의 참조값(주소)이다.

Syntax: 변수의 참조값

**형식** id(변수)

**예**  
`>>> x = 3`  
`>>> id(x)`  
`140721955779472`





# 변수를 복사할 때

```
>>> x = 3
```

```
>>> y = x
```

# 변수 y에 변수 x의 참조값이 복사된다.

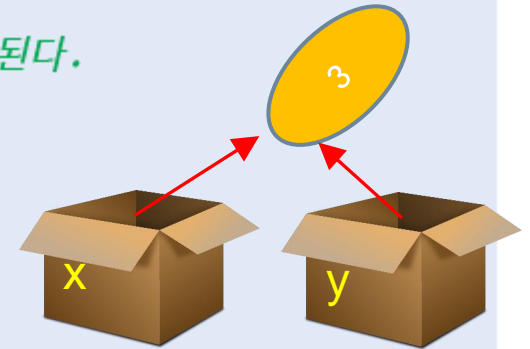
```
>>> id(x)
```

```
140721955779472
```

```
>>> id(y)
```

```
140721955779472
```

# 같은 주소를 가리킨다.

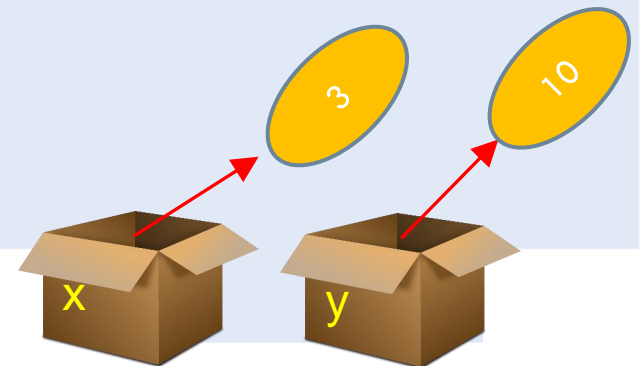


```
>>> y = 10
```

```
>>> id(y)
```

```
140721955779696
```

# 변수 y에 새로운 값이 할당되면 주소가 달라진다.



# 주석

tempconv.py

```
##  
# 이 프로그램은 화씨 온도를 받아서 섭씨 온도로 변환한다.  
#  
ftemp = 100          # 화씨 온도 100를 변수에 저장한다.  
  
ctemp = (ftemp-32.0)*5.0/9.0  # 화씨온도->섭씨온도  
print("섭씨온도:", ctemp)    # 섭씨온도를 화면에 출력한다.
```

주석으로 컴파일러에게 무시되지만 프로그램에 대한 설명이나 메모를 붙이는 것이다.

# 주석의 2번째 용도

```
##  
# 이 프로그램은 정수들의 합을 계산한다.  
#  
x = 100  
y = 200  
sum = x + y  
#diff = x - y  
print("합은 ", sum)
```

이 문장은 실행되지 않는다.

## 실행결과

합은 300

# 상수

- 변수의 이름을 대문자로 하여서 일반적인 변수와 구분

```
INCOME = 1000
```

```
TAX_RATE = 0.35
```

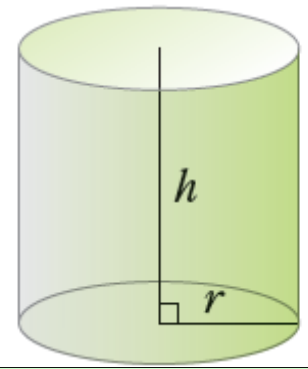
```
tax = INCOME * TAX_RATE
```

```
net_income = INCOME - tax
```

```
...
```

# Lab: 원기둥의 부피 계산

$$V = \pi r^2 h$$



반지름= 5 높이= 10 원기둥의 부피= 785.0

```
##  
#           이 프로그램은 원기둥의 부피를 계산한다.  
#  
# 원주율을 나타내는 상수를 정의한다.  
PI = 3.14  
  
# 변수를 정의한다.  
radius = 5  
height = 10  
  
# 부피를 계산한다.  
volume = PI * radius * radius * height  
  
# 결과를 출력한다.  
print("반지름=", radius, "높이=", height, "원기둥의 부피=", volume)
```

# 산술 연산

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

# 몫과 나머지 연산

```
p = 7  
q = 4  
print("나눗셈의 몫=", p // q)  
print("나눗셈의 나머지=", p % q)
```

나눗셈의 몫= 1  
나눗셈의 나머지= 3

```
today = 0  
print( (today + 10) % 7 )    # 오늘부터 10일 후는 무슨 요일일까?
```

3

# 할당 연산

$x = y = z = 0$

$x, y, z = 10, 20, 30$

# 한번에 여러 개의 변수 초기화

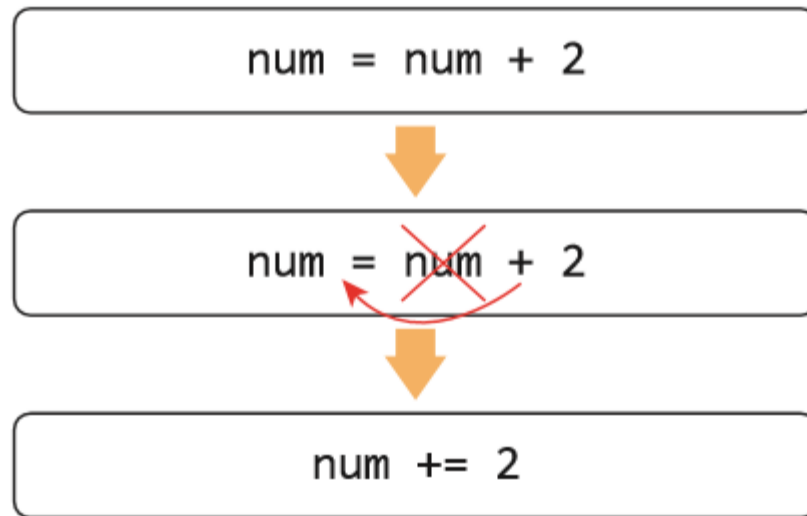
$x, y = y, x$

#  $x$ 와  $y$ 의 값을 서로 교환한다.



# 복합 연산자

- 복합 연산자(**compound operator**)란 +=처럼 대입 연산자와 다른 연산자를 합쳐 놓은 연산자이다.



# 복합 연산자

복합 연산자	의미
$x += y$	$x = x + y$
$x -= y$	$x = x - y$
$x *= y$	$x = x * y$
$x /= y$	$x = x / y$
$x \% = y$	$x = x \% y$

```
x = 1000
print("초깃값 x=", x)
x += 2;
print("x += 2 후의 x=", x)
x -= 2;
print("x -= 2 후의 x=", x)
```

```
초깃값 x= 1000
x += 2 후의 x= 1002
x -= 2 후의 x= 1000
```

# 지수 계산

- 지수(power)를 계산하려면 \*\* 연산자를 사용한다.


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


- 원리금 계산

```
a = 1000          # 원금  
r = 0.05          # 이자율  
n = 10            # 기간  
result = a*(1+r)**n  # 원리금 합계  
  
print("원리금 합계=", result)
```

```
원리금 합계= 1628.894626777442
```

# 연산자의 우선 순위

$$x + y * z$$


$$(x + y) * z$$


## 괄호로 우선 순위 변경

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

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

# 우선 순위표

연산자	설명
**	지수 연산자
~, +, -	단항 연산자
*, /, %, //	곱셈, 나눗셈, 나머지 연산자
+, -	덧셈, 뺄셈
>>, <<	비트 이동 연산자
&	비트 AND 연산자
^,	비트 XOR 연산자, 비트 OR 연산자
<=, <, >, >=	비교 연산자
<>, ==, !=	동등 연산자
=, %=, /=, //, -=, +=, *=, **=	대입, 복합 연산자
is, is not	동등 연산자
in, not in	소속 연산자
not, or, and	논리 연산자

# Lab: 복리 계산

- 1626년에 아메리카 인디언들이 뉴욕의 맨하탄섬을 단돈 60길더(약 24달러)에 탐험가 Peter Minuit에게 팔았다고 한다. 382년 정도 경과한 맨하탄 땅값은 약 600억달러라고 한다.
- 하지만 만약 인디언이 24달러를 은행의 정기예금에 입금해두었다면 어떻게 되었을까? 예금 금리는 복리로 6%라고 가정하자. 그리고 382년이 지난 후에는 원리금을 계산하여 보자.

## Solution

```
init_money = 24
interest = 0.06
years = 382
print(init_money*(1+interest)**years)
```

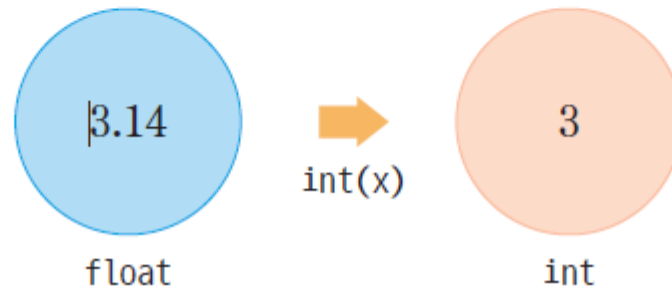
```
111442737812.28842
```

# 타입 변환

Syntax: 타입변환

**형식** 새로운타입(변수)

**예** `x = 3.14`  
`y = int(x)`



# 반올림

- 물건 값의 7.5%가 부가세라고 하자. 물건값이 12345원일 때, 부가세를 소수점 2번째 자리까지 계산하는 프로그램

```
price = 12345  
tax = price * 0.075  
tax = round(tax, 2)  
print(tax)
```

```
925.88
```



# 문자열

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



문자열은 문자들의 나열입니다.

# 큰따옴표 사용/작은 따옴표 사용

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

```
>>> msg = 'Hello'  
>>> msg  
'Hello'  
>>> print(msg)  
Hello
```

## 큰따옴표 속에 작은 따옴표 사용

```
>>> message="철수가 "안녕"이라고 말했습니다."  
SyntaxError: invalid syntax  
  
>>> message="철수가 '안녕'이라고 말했습니다."  
>>> print(message)  
철수가 '안녕'이라고 말했습니다.
```

# 세 따옴표

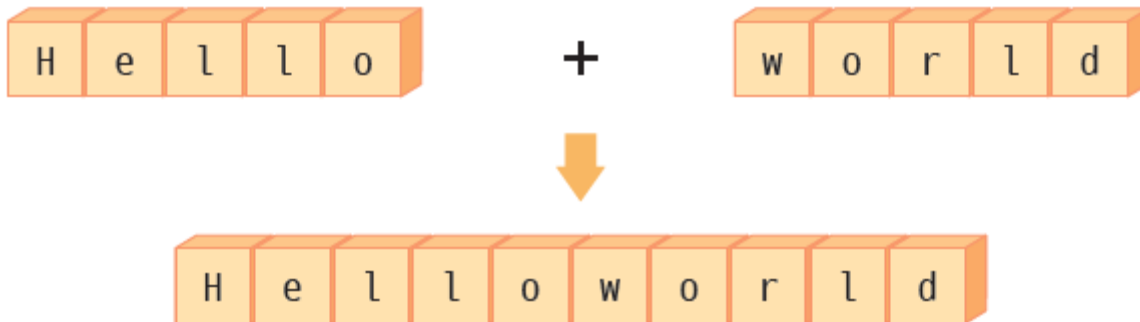
```
a = """TWINKLE, twinkle, little star,  
How I wonder what you are!  
Up above the world so high,  
Like a diamond in the sky."""
```

```
print(a)
```

```
TWINKLE, twinkle, little star,  
How I wonder what you are!  
Up above the world so high,  
Like a diamond in the sky.
```

# 문자열의 결합

```
>>> "Hello" + "World!"  
'HelloWorld!'
```



# 문자열의 반복

```
>>> lines = "-" * 30
```

```
>>> lines
```

```
'-----'
```

```
>>> song = "뚜 루루 뚜루 " * 5
```

```
>>> song
```

```
'뚜 루루 뚜루 뚜 루루 뚜루 뚜 루루 뚜루 뚜 루루 뚜루 ' '
```

# 숫자와 문자열의 구별



```
>>> print(100+200 )
300
>>> print("100"+"200")
100200
```

# 숫자 <-> 문자열

```
>>> movie = "Terminator" + 3  
TypeError: can only concatenate str (not "int") to str
```

```
>>> movie = "Terminator" + str(3)  
>>> movie  
'Terminator3'
```

```
>>> price = int("100")           # price = 100  
>>> PI = float("3.14")          # PI = 3.14
```

# 특수 문자열

특수 문자열	의미
\n	줄바꿈 문자
\t	탭문자
\\	역슬래시 자체
\"	큰따옴표 자체
\'	작은따옴표 자체

```
>>> print("말 한마디로\n천냥빚을 갚는다")  
말 한마디로  
천냥빚을 갚는다
```



# 문자와 문자열

0	1	2	3	4	5	6	7	8	9	10
H	e	l	l	o		w	o	r	l	d
-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
>>> s = "Hello World"
```

```
>>> s[0]
```

```
'H'
```

```
>>> s = "Hello World"
```

```
>>> s[-1]
```

```
'd'
```

# 예제

```
a = "Kim"  
b = "Park"  
acronym = a[0] + "과" + b[0]  
print(acronym)
```

K과P



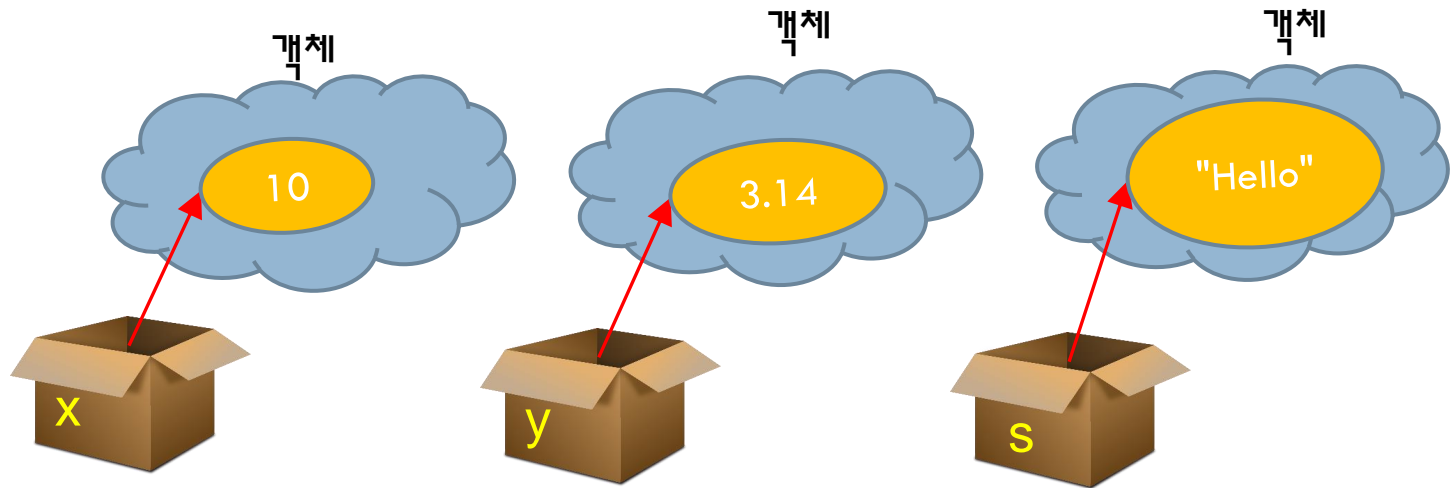
# 문자열은 객체

- 객체(object)란 프로그래밍에서 관련 있는 변수와 함수(아직 학습하지 않았지만 어떤 동작이라고 생각하자)를 하나로 묶은 것이다.



# 사실 파이썬에서는 모든 것이 객체이다.

- 파이썬에서는 정수나 실수도 객체로 저장된다.



# 문자열의 함수

```
name = "Harry Parter"
lower_name = name.lower()           # 'harry parter'
```

```
name = "Harry Parter"
new_name = name.replace("Parter", "Porter")   # Harry Porter
```

# Lab: 로봇 기자 만들기



- 사용자에게 경기장, 점수, 이긴 팀, 진 팀, 우수 선수를 질문하고 변수에 저장한다. 이들 문자열에 문장을 붙여서 기사를 작성한다.

경기장은 어디입니까?서울  
이긴팀은 어디입니까삼성  
진팀은 어디입니까?LG  
우수선수는 누구입니까?홍길동  
스코어는 몇대몇입니까?8:7

=====

오늘 서울 에서 야구 경기가 열렸습니다.  
삼성 과 **LG** 은 치열한 공방전을 펼쳤습니다.  
홍길동 이 맹활약을 하였습니다.  
결국 삼성 가 **LG** 를 8:7 로 이겼습니다.

=====

# Solution

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

# 변수와 문자열을 연결하여 기사를 작성한다.
print("")
print("=====")
print("오늘", stadium, "에서 야구 경기가 열렸습니다.")
print(winner, "과", loser, "은 치열한 공방전을 펼쳤습니다.")
print(vip, "이 맹활약을 하였습니다.")
print("결국", winner,"가", loser,"를 ", score,"로 이겼습니다.")
print("=====
```

# 입력 input() 함수

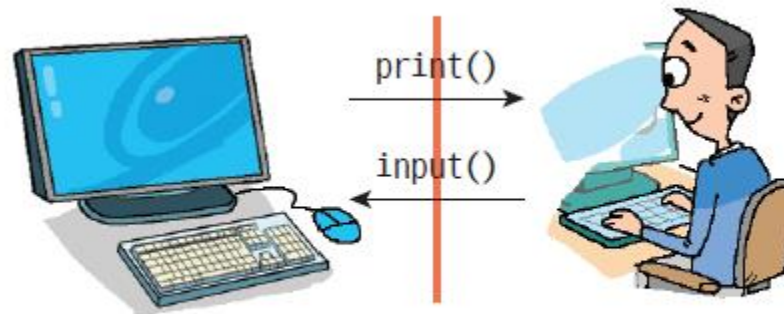
Syntax: input() 함수

**형식** 변수 = input(안내메시지)

**예** x = input("이름을 입력하시오: ")

변수

안내 메시지를 출력하고 사용자가 입력한 값을 문자열 형태로 반환한다.



사용자 인터페이스



# input() 함수

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

이름을 입력하시오: 홍길동

```
>>> print(name)
```

홍길동

```
name = input("이름을 입력하시오: ")
```

```
print(name, "씨, 안녕하세요?")
```

```
print("파이썬에 오신 것을 환영합니다.")
```

이름을 입력하시오: 홍길동

홍길동 씨, 안녕하세요?

파이썬에 오신 것을 환영합니다.

# 정수 입력

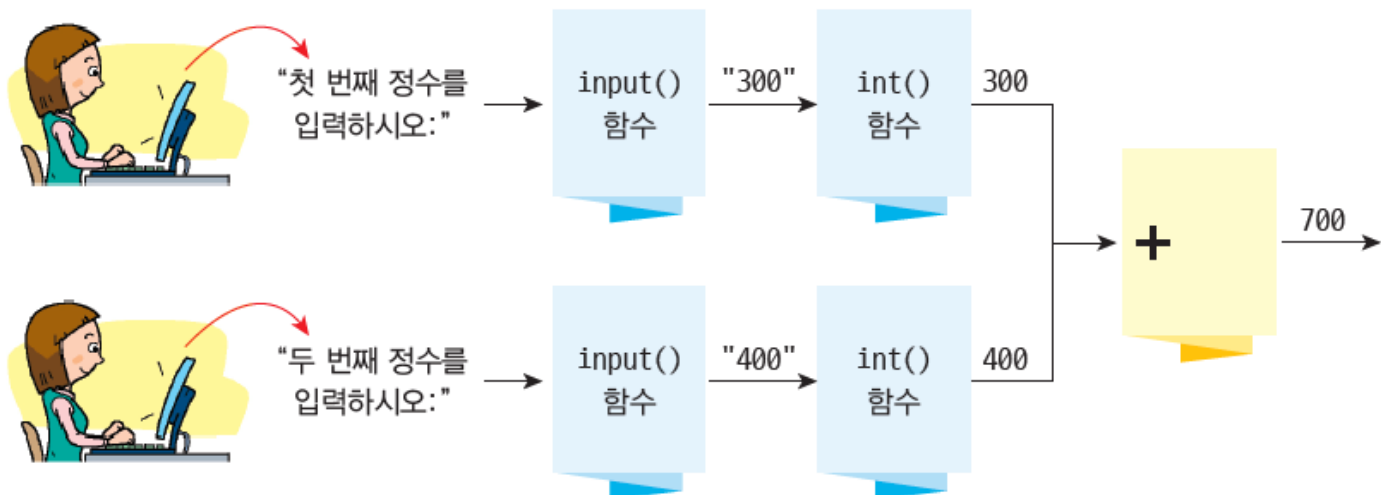
```
x = input("첫 번째 정수를 입력하시오:")  
y = input("두 번째 정수를 입력하시오:")  
sum = x + y  
print("합은 ", sum)
```

```
첫 번째 정수를 입력하시오: 300  
첫 번째 정수를 입력하시오: 400  
합은 300400
```

# 정수 입력

```
s1 = input("첫 번째 정수를 입력하시오:")  
x = int(s1)                                # 문자열을 정수로 변환한다.  
s2 = input("두 번째 정수를 입력하시오:")  
y = int(s2)                                # 문자열을 정수로 변환한다.  
sum = x + y  
print("합은 ", sum)
```

첫 번째 정수를 입력하시오: 300  
첫 번째 정수를 입력하시오: 400  
합은 700



# 부동소수점수 입력

```
SQMETER_PER_P = 3.3
```

```
area = float(input("면적(제곱미터):"))  
py = area / SQMETER_PER_P  
print(py, "평")
```

```
면적(제곱미터):25.6  
7.757575757575759 평
```

# 변수와 문자열을 동시에 출력할때

```
x = 100  
y = 200  
print(x, "와 ", y, "의 합=", x+y)
```

100 와 200 의 합= 300

```
x = 100  
y = 200  
print(f"{x}와 {y}의 합={x+y}")
```

100와 200의 합=300

# 형식화된 출력

Syntax: print() 함수

**형식**    형식문자열 % (값1, 값2, ..., 값n)

**예**    `p = 7.76`  
      `print("%10.2f" % p)`

```
SQMETER_PER_P = 3.3
```

```
area = eval(input("면적(제곱미터):"))  
py = area / SQMETER_PER_P  
print("%.2f평" % py)                    # 출력 7.76평
```

```
면적(제곱미터):25.6  
7.76평
```

# Lab: 대화하는 프로그램 만들기

- 변수를 사용하여 사용자의 이름과 나이를 문자열 형태로 기억했다가 출력할 때 사용하는 프로그램을 작성해보자.

안녕하세요?

이름이 어떻게 되시나요? 홍길동  
만나서 반갑습니다. 홍길동씨  
이름의 길이는 다음과 같군요: 3

나이가 어떻게 되나요? 21  
내년이면 22이 되시는군요.

# Solution

```
##  
#           이 프로그램은 사용자와 친근하게 대화한다.  
#  
print("안녕하세요?")  
name = input("이름이 어떻게 되시나요? ")  
print("만나서 반갑습니다. " + name + "씨")  
  
print("이름의 길이는 다음과 같군요:", len(name))  
  
age = int(input("나이가 어떻게 되나요? "))  
print("내년이면 " + str(age+1) + "이 되시는군요.")
```



# Lab: 구의 부피 계산하기

$$V = \frac{4}{3}\pi r^3$$

- 반지름이 5m인 구의 부피를 계산하는 프로그램을 작성해보자.

반지름을 입력하시오: 5.0  
구의 부피= 523.5986666666666

```
##  
#           이 프로그램은 구의 부피를 계산한다.  
#  
  
# 사용자에게 구의 반지름을 입력하도록 한다. 구의 반지름을 문자열에서 실수로  
# 변환한다.  
r = float(input("반지름을 입력하시오: "))  
  
# 구의 부피를 공식을 이용하여 계산한다.  
volume = (4.0/3.0) * 3.141592 * r**3  
  
# 구의 부피를 화면에 출력한다.  
print("구의 부피=",volume)
```

# Lab: 자동판매기 프로그램

- 자동 판매기를 시뮬레이션하는 프로그램을 작성하여 보자.

물건값을 입력하시오: 750

1000원 지폐개수: 1

500원 동전개수: 0

100원 동전개수: 0

500원 = 0 100원 = 2 10원 = 5 1원 = 0



```
##
```

```
# 이 프로그램은 자판기에서 거스름돈을 계산한다.
```

```
#
```

```
itemPrice = int(input("물건값을 입력하시오: "))
```

```
note = int(input("1000원 지폐개수: "))
```

```
coin500 = int(input("500원 동전개수: "))
```

```
coin100 = int(input("100원 동전개수: "))
```

```
change = note*1000 + coin500*500 + coin100*100 - itemPrice
```

```
# 거스름돈(500원 동전 개수)을 계산한다.
```

```
nCoin500 = change//500
```

```
change = change%500
```

```
# 거스름돈(100원 동전 개수)을 계산한다.
```

```
nCoin100 = change//100
```

```
change = change%100
```

```
# 거스름돈(10원 동전 개수)을 계산한다.
```

```
nCoin10 = change//10
```

```
change = change%10
```

```
# 거스름돈(1원 동전 개수)을 계산한다.
```

```
nCoin1 = change
```

```
print("500원=", nCoin500, "100원=", nCoin100, "10원=", nCoin10, "1원=", nCoin1)
```