

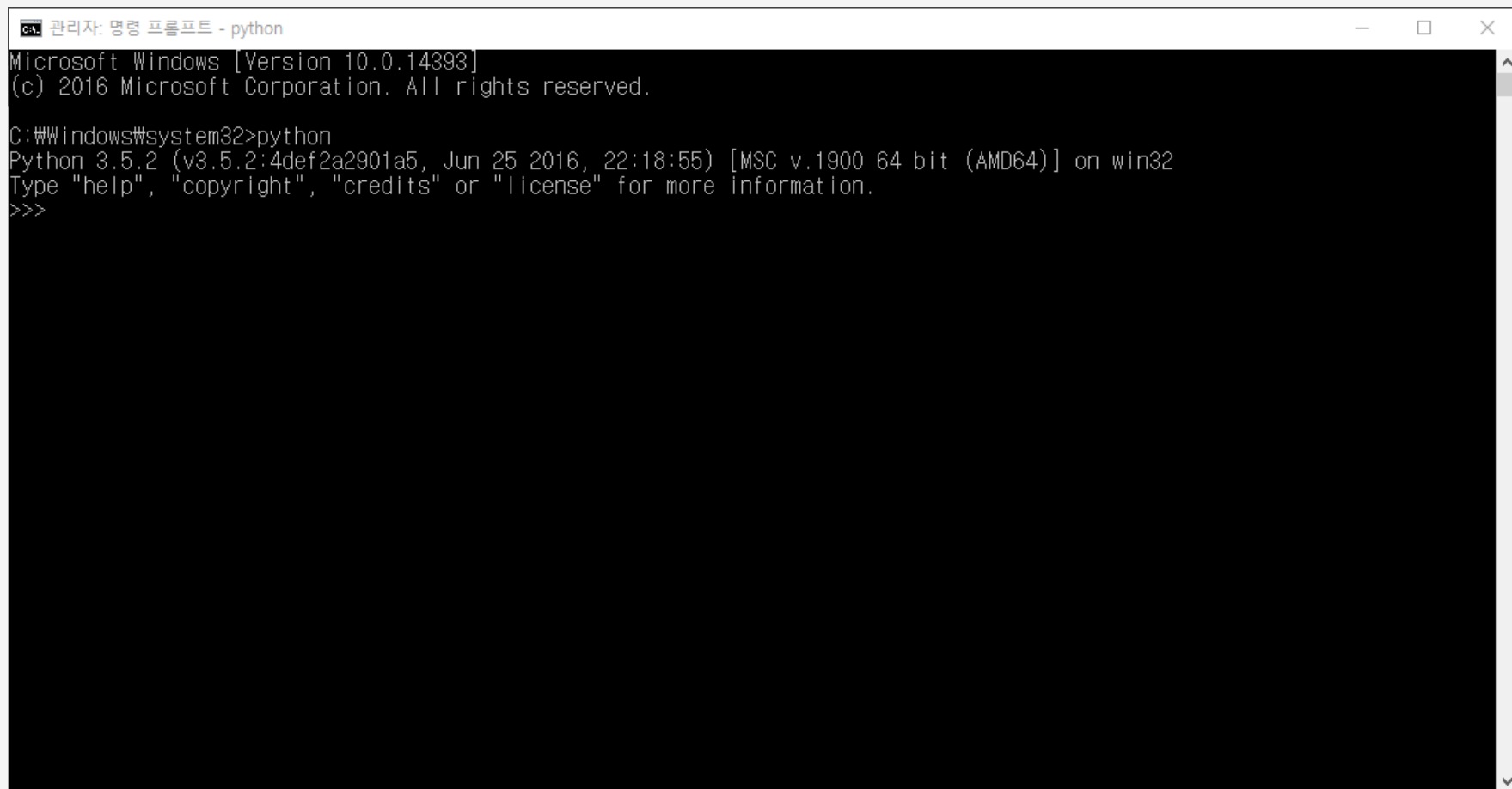
Python 기초

Python 기본 사용 법

인터프리터 사용

Windows의 명령 프롬프트 입력 창에 **python** 명령어를 입력하여 인터프리터 엔진을 동작시킨다.

인터프리터 엔진 종료는 **exit()** 사용

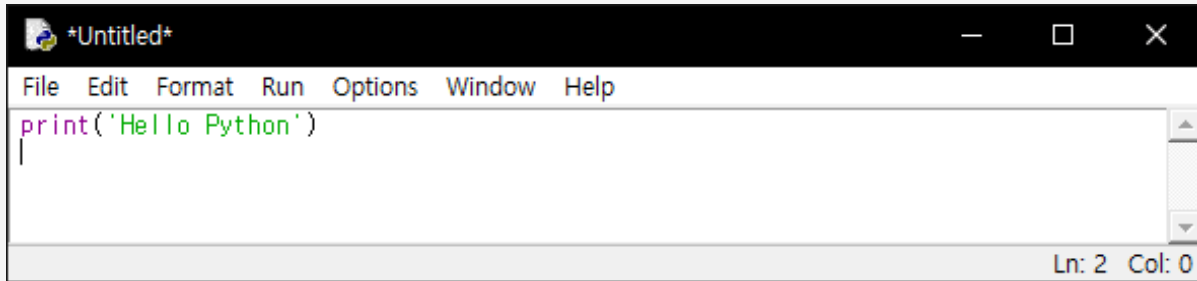


```
관리자: 명령 프롬프트 - python
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Windows\system32>python
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:18:55) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

스크립트 파일 사용

Python IDLE 실행 -> File -> New File



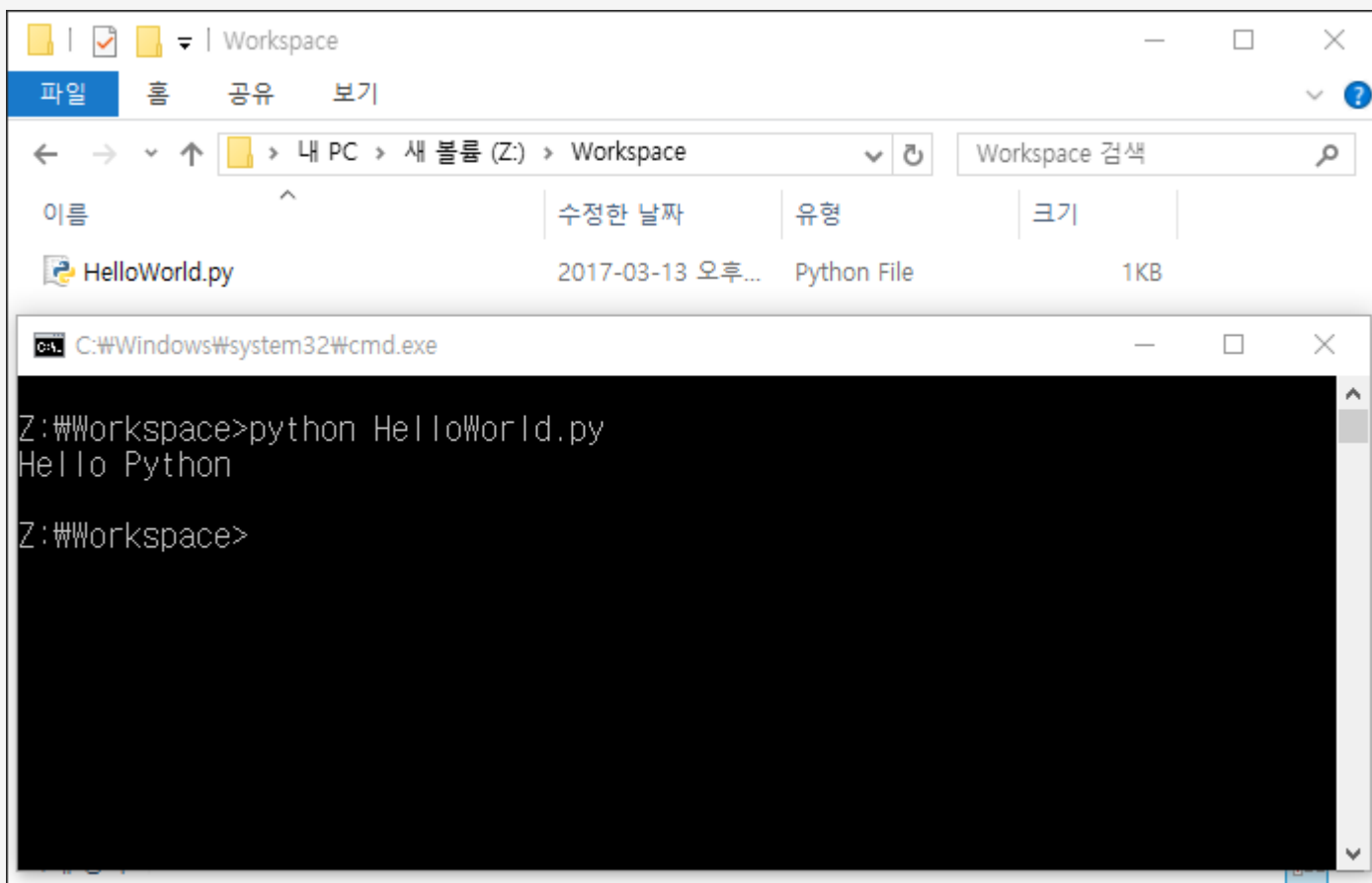
File -> Save 또는 단축키 Ctrl + S 로 파일 저장

Python IDLE에서 단축키 F5 눌러서 스크립트 실행

스크립트 파일 사용

명령 프롬프트 상에서 Python 스크립트 파일이 존재하는 곳으로 이동하여 실행

윈도우에서는 탐색기를 이용하여 이동 후 Shift Key + Right Mouse Click -> 여기서 명령 창 열기 클릭



문자열 출력

따옴표를 사용하여 문자열 출력

```
print('Hello Python')
```

```
print("Hello Python")
```

연산 결과 출력

연산 결과를 출력 하기 위한 방법

```
print(10 + 20)
```

```
print(10 * 20)
```

혼합 출력

쉼표(,)로 문자열 및 연산 결과 등을 구분하여 출력

```
print('10 더하기 20 은', 10 + 20)
```

```
print('10 곱하기 20 은', 10 * 20)
```


혼합 출력

퀴즈

$12 + 54 = 66$ 입니다.

$268 - 42 = 226$ 입니다.

$2 * 23 = 46$ 입니다.

$120 / 3 = 40.0$ 입니다.

이스케이프 문자

이스케이프 문자	기능
\n	다음 줄로 이동
\r	해당 줄의 처음으로 이동
\t	8 칸 공백
\'	' 문자
\"	" 문자
\\	\ 문자

이스케이프 문자는 문자열을 출력하기 위해서 사용 되는 기능
외의 부가적인 기능을 사용하기 위해서 쓰인다.

2줄 이상의 문자열 출력

이스케이프 문자를 사용하여 출력 문자열을 다음 라인으로 이동 할 수 있다.

```
print(' 다음 라인으로\n이동')
```

Comment (주석)

프로그램 코드 내에 설명문을 넣기 위해 사용 된다.

한 줄 주석

'''Multi-Line

주석'''

Python 진법

2진법 8진법 16진법

진법은 수를 표현 하기 위한 문자의 묶음에 따라 2진법, 8진법, 10진법, 16진법이라 한다.

진법	표현 문자	표현 식
2	0, 1	0b
8	0, 1, 2, 3, 4, 5, 6, 7	0o
10	0, 1, 2, 3, 4, 5, 6, 7, 8, 9	
16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F	0x

진법 사용 하기

Python에 특정 진법의 값을 입력 할 때 표현식을 사용

```
print(0b100)
```

```
print(0o100)
```

```
print(100)
```

```
print(0x100)
```

진법 변환 하기

서로 다른 진법 간에 변환을 수행하는 함수이다.

함수	설명
bin()	2진수 값으로 변환
oct()	8진수 값으로 변환
hex()	16진수 값으로 변환

진법 변환 하기

진법 변환 함수 안에는 변환 시키고자하는 값을 입력 한다.

```
print(bin(100))
```

```
print(oct(100))
```

```
print(hex(100))
```

Python 서식 문자

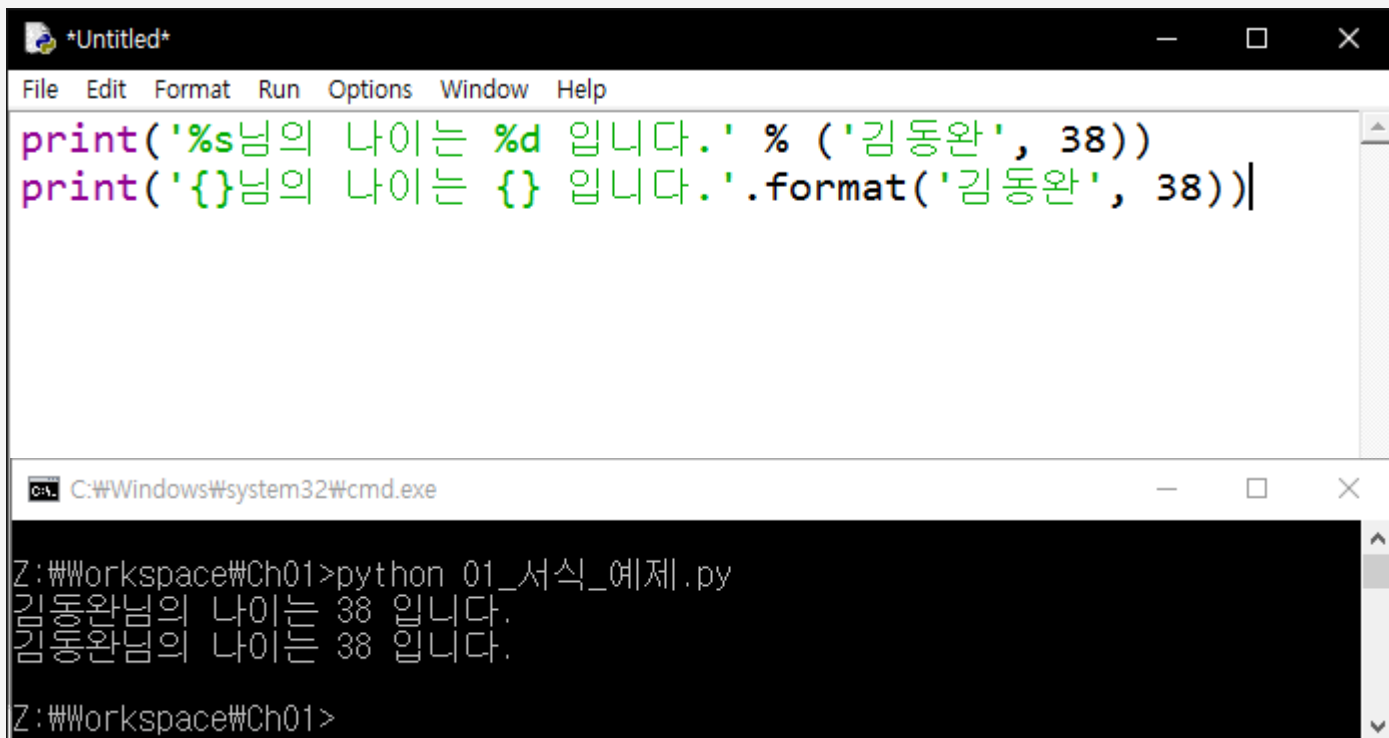
서식 문자 (기본)

출력 되는 문자열에 대해 특정 서식을 지정하고 출력하기 위해서 사용 된다.

C 스타일	Python 3	설명
%s	{}	문자열 출력
%d	{}	정수 출력
	{:b}	표현식 없는 2진수 값 출력
%o	{:o}	표현식 없는 8진수 값 출력
%x	{:x}	표현식 없는 16진수 값 출력
%f	{:f}	실수 출력
%.2f	{:.2f}	소수점 2자리 까지의 실수 출력
%6d	{:6}	6자리 고정 출력

서식 문자 (예제)

문자열과 정수 출력 예제



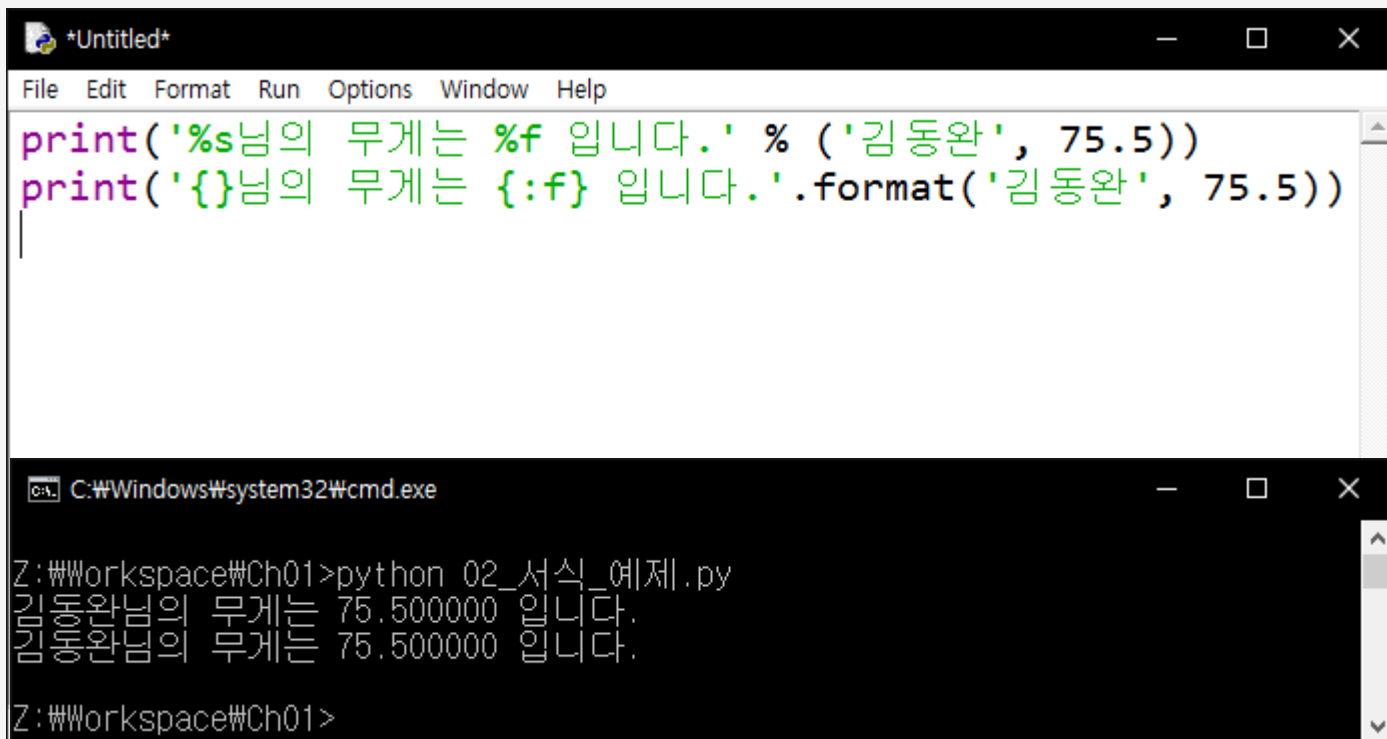
The image shows two windows. The top window is a Python IDE titled '*Untitled*' with a menu bar (File, Edit, Format, Run, Options, Window, Help). It contains two lines of Python code: `print('%s님의 나이는 %d 입니다.' % ('김동완', 38))` and `print('{}님의 나이는 {} 입니다.'.format('김동완', 38))`. The bottom window is a command prompt titled 'C:\Windows\system32\cmd.exe' showing the execution of the Python script. The prompt shows the command `python 01_서식_예제.py` being run, followed by the output `김동완님의 나이는 38 입니다.` appearing twice.

```
*Untitled*
File Edit Format Run Options Window Help
print('%s님의 나이는 %d 입니다.' % ('김동완', 38))
print('{}님의 나이는 {} 입니다.'.format('김동완', 38))

C:\Windows\system32\cmd.exe
Z:\workspace\Ch01>python 01_서식_예제.py
김동완님의 나이는 38 입니다.
김동완님의 나이는 38 입니다.
Z:\workspace\Ch01>
```

서식 문자 (예제)

문자열과 실수 출력 예제



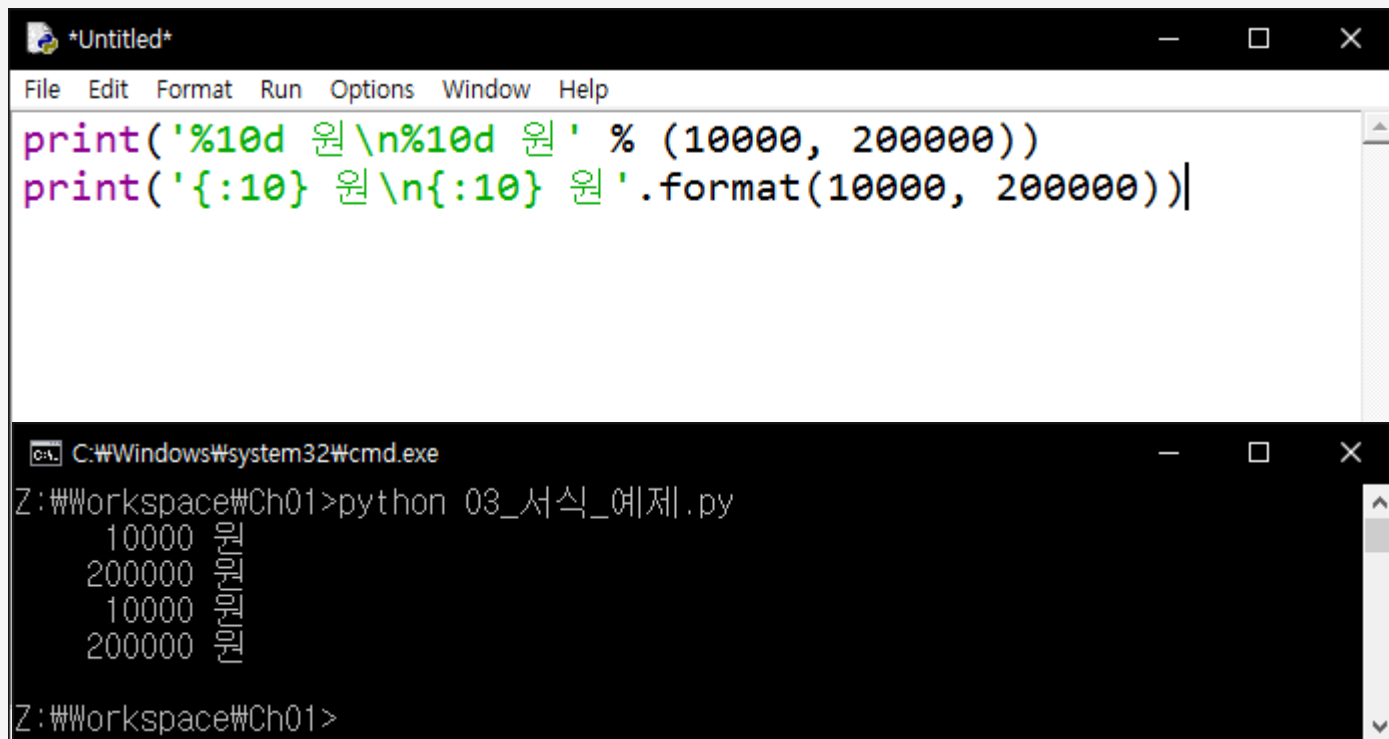
The image shows a Python IDE window titled '*Untitled*' with a menu bar (File, Edit, Format, Run, Options, Window, Help) and a code editor containing two lines of Python code. Below the IDE is a Windows command prompt window titled 'C:\Windows\system32\cmd.exe' showing the execution of the Python script. The code in the IDE uses both the % operator and the .format() method to format a string with a name and a weight. The command prompt shows the output of the script, which is the same string formatted with the given values.

```
*Untitled*
File Edit Format Run Options Window Help
print('%s님의 무게는 %f 입니다.' % ('김동완', 75.5))
print('{}님의 무게는 {:.f} 입니다.'.format('김동완', 75.5))

C:\Windows\system32\cmd.exe
Z:\workspace\Ch01>python 02_서식_예제.py
김동완님의 무게는 75.500000 입니다.
김동완님의 무게는 75.500000 입니다.
Z:\workspace\Ch01>
```

서식 문자 (예제)

고정 길이 출력



The image shows a Python IDE window titled '*Untitled*' and a Windows command prompt window. The IDE contains two lines of Python code that demonstrate string formatting with fixed widths. The command prompt shows the output of running a Python script, displaying the formatted numbers.

```
*Untitled*
File Edit Format Run Options Window Help
print('%10d 원\n%10d 원' % (10000, 200000))
print('{:10} 원\n{:10} 원'.format(10000, 200000))

C:\Windows\system32\cmd.exe
Z:\Workspace\Ch01>python 03_서식_예제.py
      10000 원
     200000 원
      10000 원
     200000 원

Z:\Workspace\Ch01>
```

서식 문자 (정렬)

정렬 하여 출력



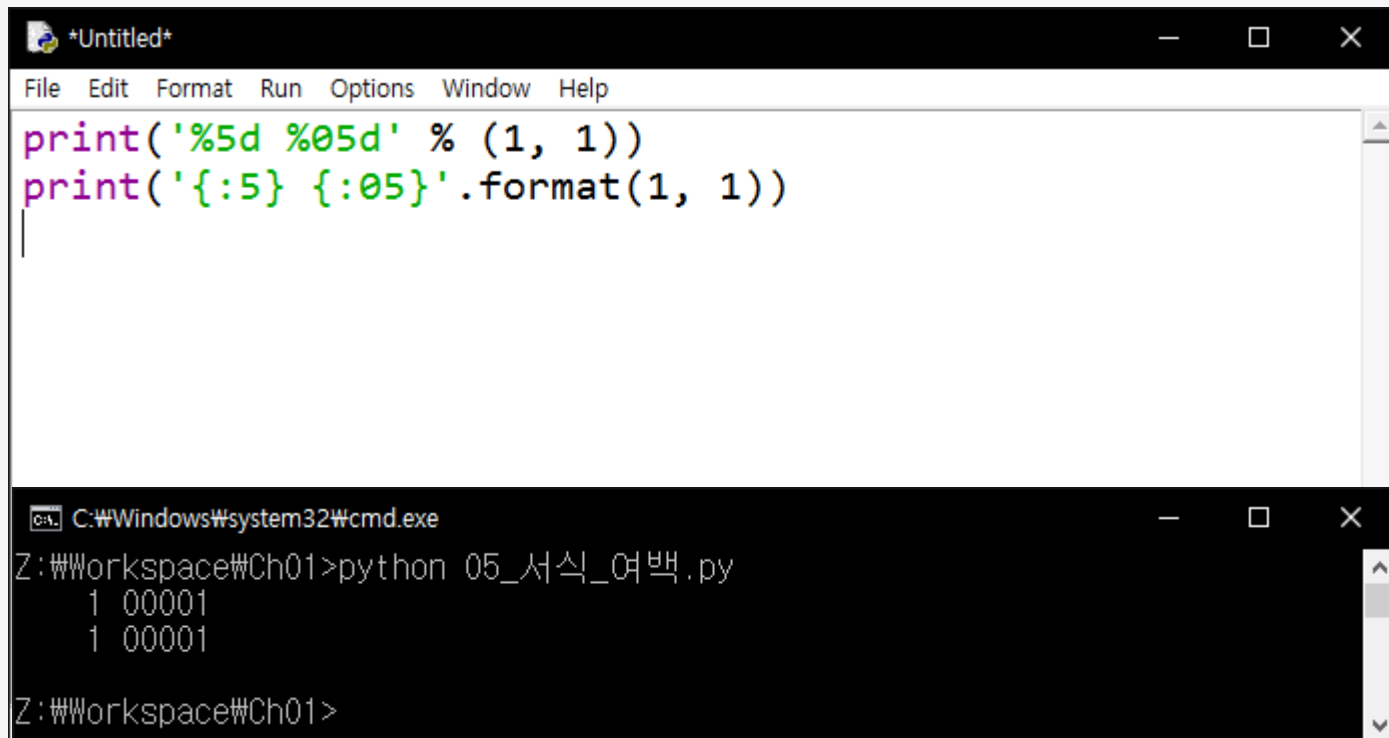
The image shows two windows. The top window is a Python IDE titled '*Untitled*' with a menu bar (File, Edit, Format, Run, Options, Window, Help). It contains three lines of Python code using string formatting to align text. The bottom window is a command prompt titled 'C:\Windows\system32\cmd.exe' showing the execution of a Python script. The output of the script is displayed in the command prompt, showing the text '오른쪽 , 왼쪽' aligned to the right and left respectively, and '가운데' aligned to the center.

```
*Untitled*
File Edit Format Run Options Window Help
print('%10s , %-10s' % ('오른쪽', '왼쪽'))
print('{:>10} , {:<10}'.format('오른쪽', '왼쪽'))
print('{:^10}'.format('가운데'))

C:\Windows\system32\cmd.exe
Z:\Workspace\Ch01>python 04_서식_정렬.py
      오른쪽 ,  왼쪽
      오른쪽 ,  왼쪽
    가운데
Z:\Workspace\Ch01>
```

서식 문자 (여백 채우기)

빈 여백을 특정 문자로 채워서 출력



The image shows a Python IDE window titled '*Untitled*' with a menu bar (File, Edit, Format, Run, Options, Window, Help) and a code editor containing two lines of Python code:

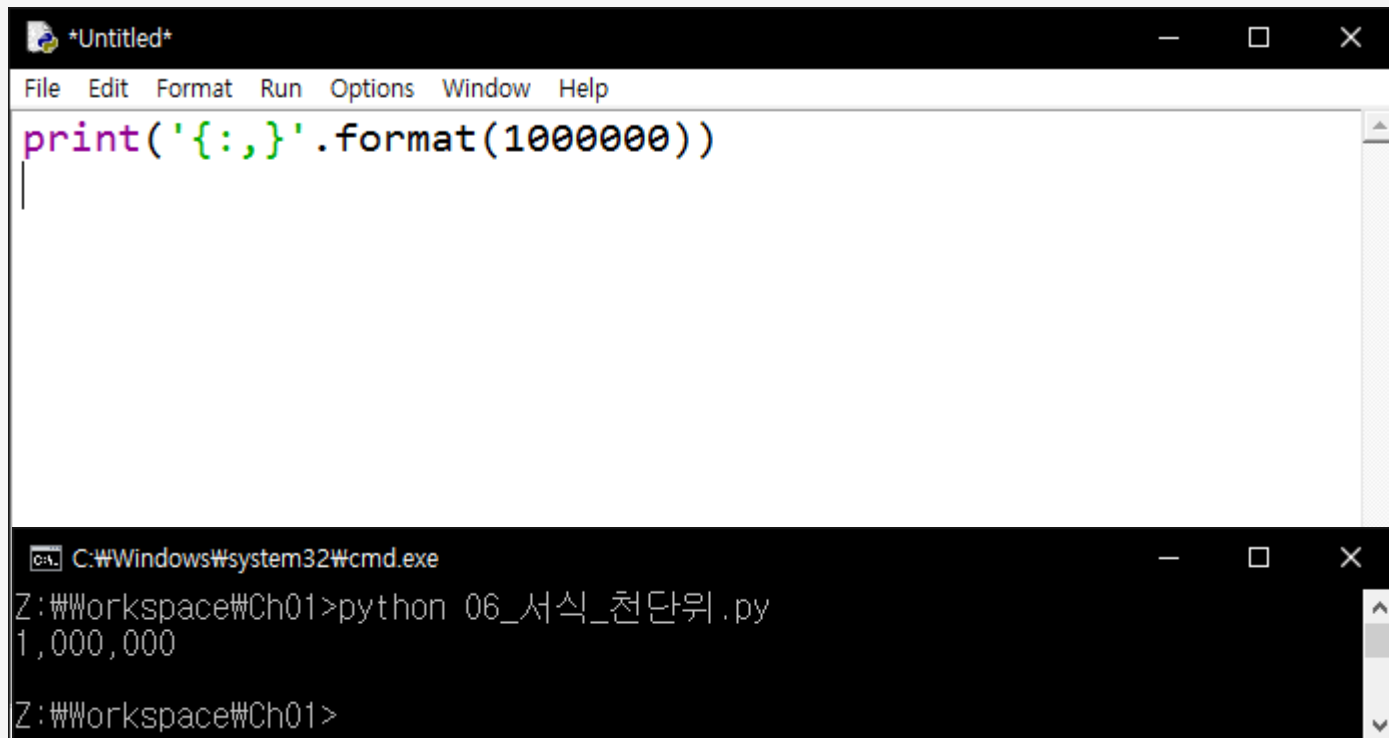
```
print('%5d %05d' % (1, 1))  
print('{:5} {:05}'.format(1, 1))
```

Below the IDE is a Windows command prompt window titled 'C:\Windows\system32\cmd.exe'. It shows the execution of the Python script:

```
Z:\workspace\Ch01>python 05_서식_여백.py  
1 00001  
1 00001  
Z:\workspace\Ch01>
```


서식 문자 (천 단위 구분)

천 단위 구분 출력



The image shows a Python IDE window titled '*Untitled*' with a menu bar (File, Edit, Format, Run, Options, Window, Help) and a code editor containing the line `print('{:,}'.format(1000000))`. Below the IDE is a Windows command prompt window titled 'C:\Windows\system32\cmd.exe' showing the execution of the command `python 06_서식_천단위.py` and its output `1,000,000`.

```
*Untitled*
File Edit Format Run Options Window Help
print('{:,}'.format(1000000))

C:\Windows\system32\cmd.exe
Z:\workspace\Ch01>python 06_서식_천단위.py
1,000,000
Z:\workspace\Ch01>
```

Python 변수

변수 기본 사용

변수명 = 값

변수명1, 변수명2 = 값1, 값2

변수명1 = 변수명2 = 값

변수 작성 규칙

1. 알파벳, 숫자, 언더스코어(_)로 구성
2. 알파벳은 대/소문자 구분
3. 한글 사용 가능
4. 변수명의 시작은 숫자로 할 수 없음
5. 공백이나 특수 기호는 포함 할 수 없음
6. Python 예약어는 사용하면 안된다.

변수 자료형 종류

1. 정수 : 0과 음수, 양수 값을 포함하는 숫자 값
2. 실수 : 소수점을 사용하는 숫자 값
3. 문자열 : 따옴표로 묶여 있는 값
4. 리스트 : 정수, 실수 및 문자열 등 자료들의 집합 (값의 집합)
5. 튜플 : 정수, 실수 및 문자열 등 자료들의 집합 (값의 집합)
6. 사전 : 정수, 실수, 및 문자열 등 자료들의 집합 (키와 값이 쌍으로 존재)

변수

변수에 데이터 저장

```
num = 100
```

```
print("정수형 변수 : %d" % num)
```

```
print("정수형 변수 :", num)
```

```
print("정수형 변수 : {}".format(num))
```

변수

변수에 데이터 저장

```
num = 100
```

```
print("변경 전 : %d" % num)
```

```
num = num + 100
```

```
print("변경 후 : {}".format(num))
```

변수

변수에 데이터 저장

```
num = 100
```

```
print("변경 전 : %d" % num)
```

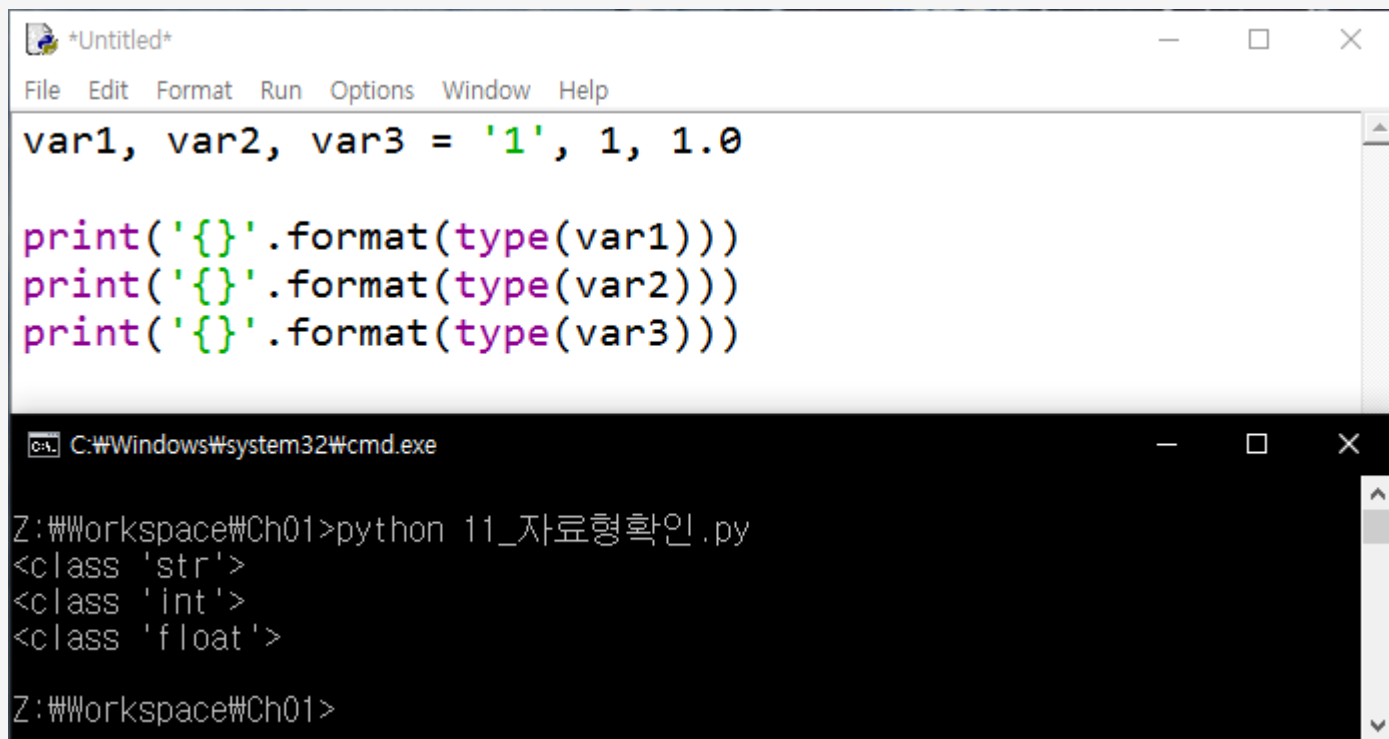
```
num = num + 100
```

```
print("변경 후 : {}".format(num))
```


변수 자료형 확인

변수에 담긴 값의 자료형을 확인하기 위한 함수

type(변수명)



The image shows a Python IDE window titled '*Untitled*' with a menu bar (File, Edit, Format, Run, Options, Window, Help) and a code editor containing the following Python code:

```
var1, var2, var3 = '1', 1, 1.0

print('{}'.format(type(var1)))
print('{}'.format(type(var2)))
print('{}'.format(type(var3)))
```

Below the IDE is a Windows command prompt window titled 'C:\Windows\system32\cmd.exe'. It shows the execution of the Python script:

```
Z:\workspace\Ch01>python 11_자료형확인.py
<class 'str'>
<class 'int'>
<class 'float'>

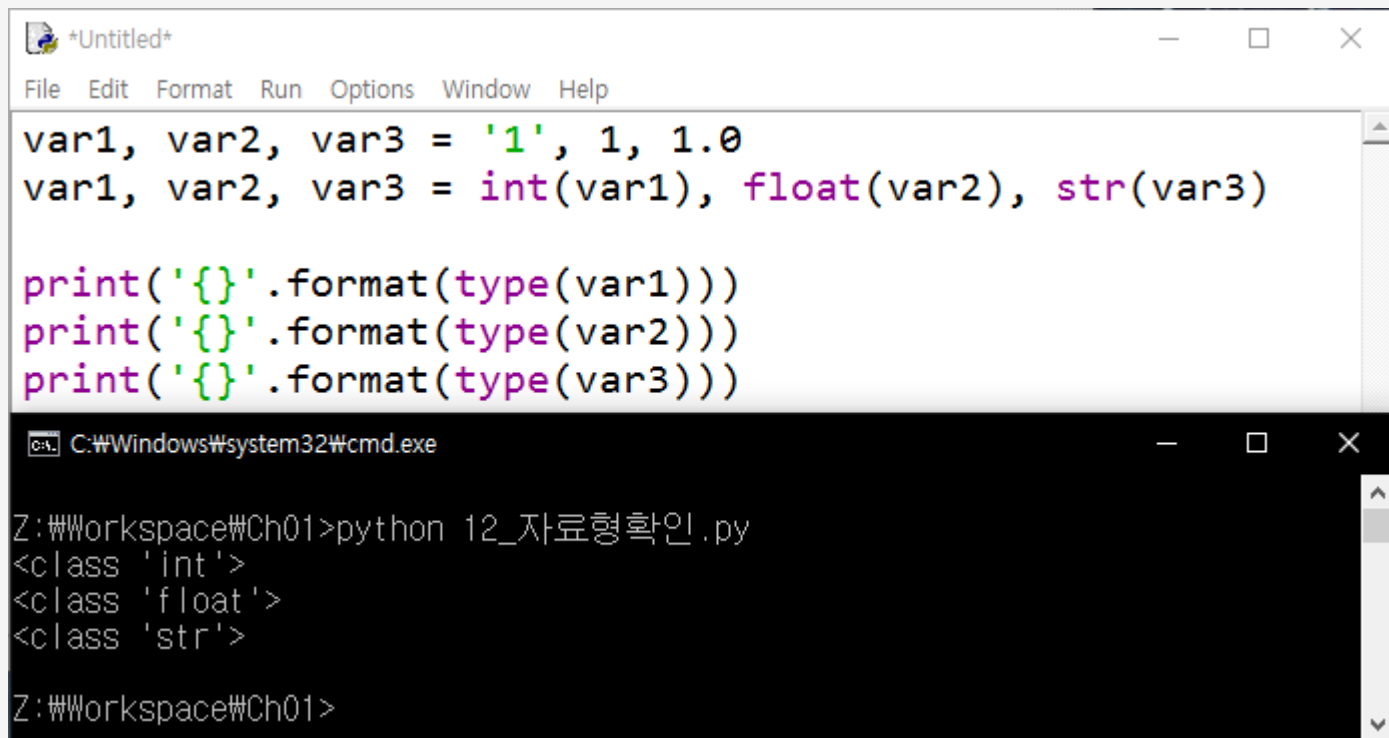
Z:\workspace\Ch01>
```

변수 자료형 변환

자료형을 변환 시키기 위한 함수

함수	설명
int()	정수형 자료로 변환
str()	문자열 자료로 변환
float()	실수형 자료로 변환

변수 자료형 변환 (예제)



The image shows a Python IDE window titled '*Untitled*' with a menu bar (File, Edit, Format, Run, Options, Window, Help) and a code editor. The code in the editor performs variable assignment and type conversion, followed by printing the types. Below the IDE is a Windows command prompt window titled 'C:\Windows\system32\cmd.exe' showing the execution of the Python script and its output.

```
var1, var2, var3 = '1', 1, 1.0
var1, var2, var3 = int(var1), float(var2), str(var3)

print('{}'.format(type(var1)))
print('{}'.format(type(var2)))
print('{}'.format(type(var3)))
```

```
C:\Windows\system32\cmd.exe

Z:\Workspace\Ch01>python 12_자료형확인.py
<class 'int'>
<class 'float'>
<class 'str'>

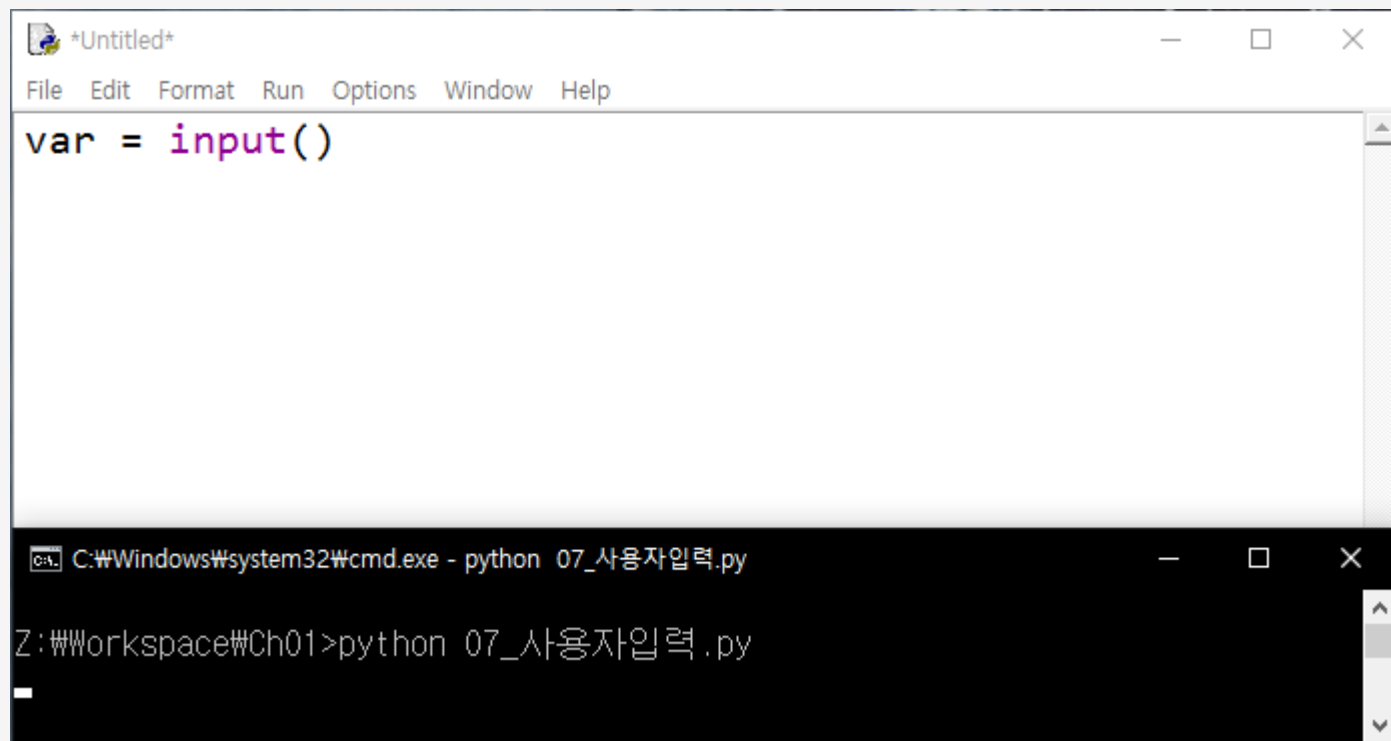
Z:\Workspace\Ch01>
```

Python 사용자 입력

input() 함수

사용자로부터 값을 입력 받아 코드 안에서 사용하기 위한 함수

변수명 = **input()**



The image shows a screenshot of a Python IDE window titled '*Untitled*' with a menu bar (File, Edit, Format, Run, Options, Window, Help). The code editor contains the line `var = input()`. Below the IDE is a black command prompt window. The title bar of the command prompt reads 'C:\Windows\system32\cmd.exe - python 07_사용자입력.py'. The command prompt shows the command `Z:\workspace\Ch01>python 07_사용자입력.py` being executed, with a cursor on the line below.

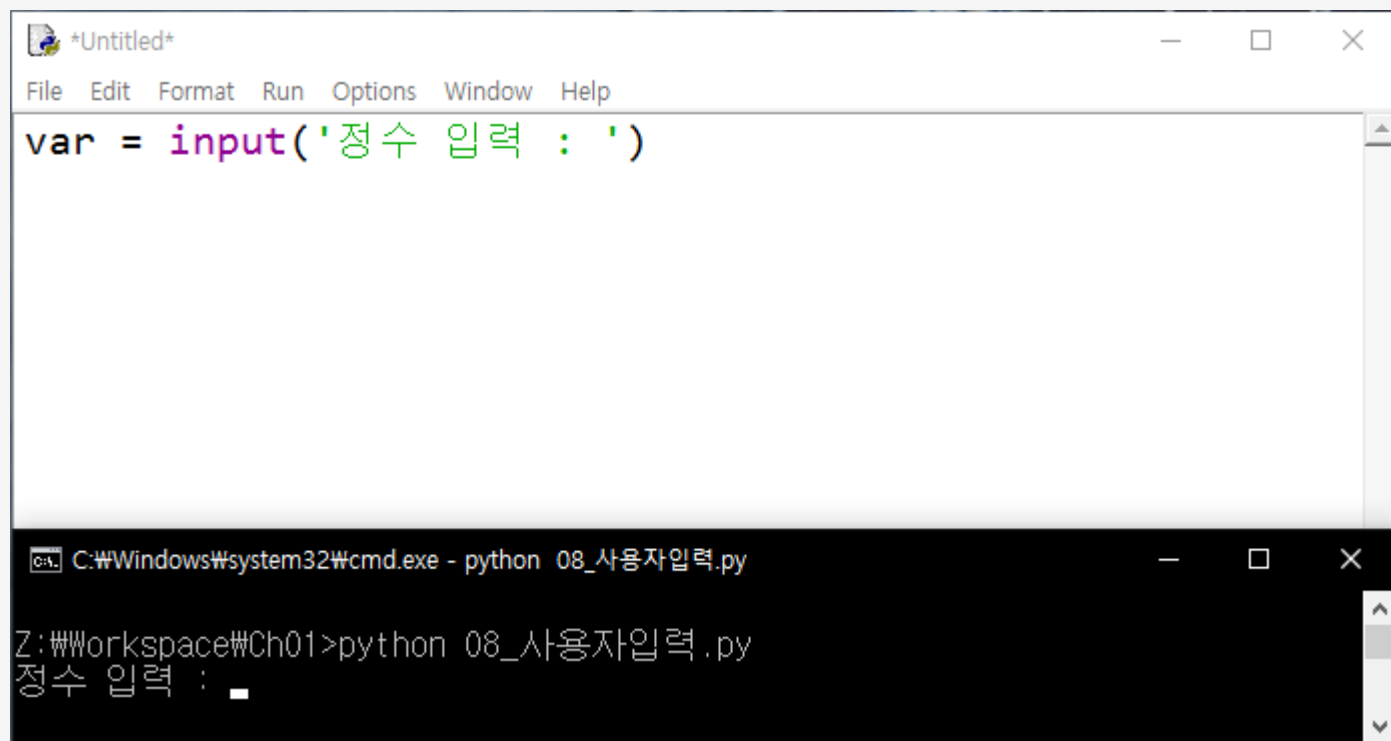
```
*Untitled*
File Edit Format Run Options Window Help
var = input()

C:\Windows\system32\cmd.exe - python 07_사용자입력.py
Z:\workspace\Ch01>python 07_사용자입력.py
_
```

input() 함수

사용자에게 입력해야 하는 값의 종류를 알려주기 위한 문자열

변수명 = `input('문자열')`

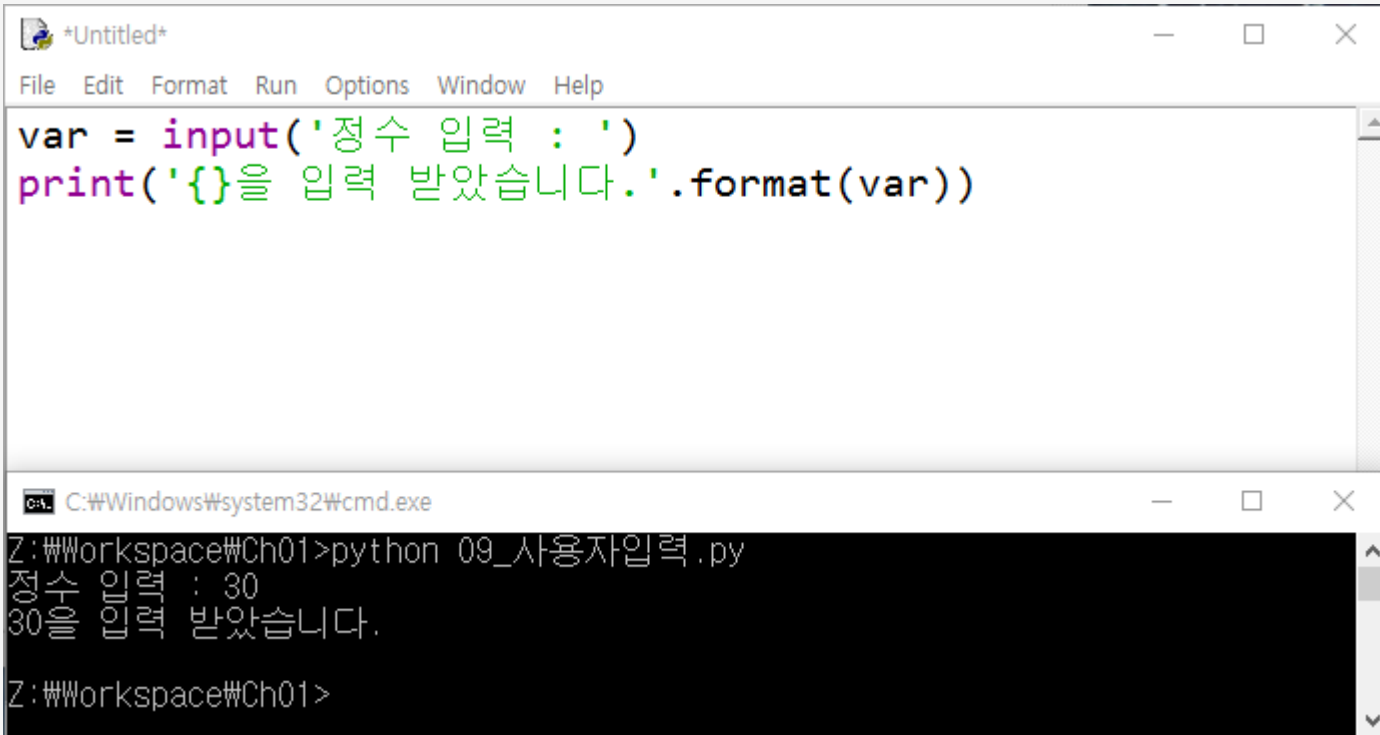


The screenshot displays two windows. The top window is a Python IDE titled '*Untitled*' with a menu bar (File, Edit, Format, Run, Options, Window, Help). The code editor contains the line `var = input('정수 입력 : ')`. The bottom window is a command prompt titled 'C:\Windows\system32\cmd.exe - python 08_사용자입력.py'. It shows the command `Z:\workspace\Ch01>python 08_사용자입력.py` being executed, followed by the prompt `정수 입력 :` with a cursor.

```
*Untitled*
File Edit Format Run Options Window Help
var = input('정수 입력 : ')

C:\Windows\system32\cmd.exe - python 08_사용자입력.py
Z:\workspace\Ch01>python 08_사용자입력.py
정수 입력 : 
```

input() 함수 (예제)



The image shows two windows. The top window is a Python IDE titled '*Untitled*' with a menu bar (File, Edit, Format, Run, Options, Window, Help). It contains the following Python code:

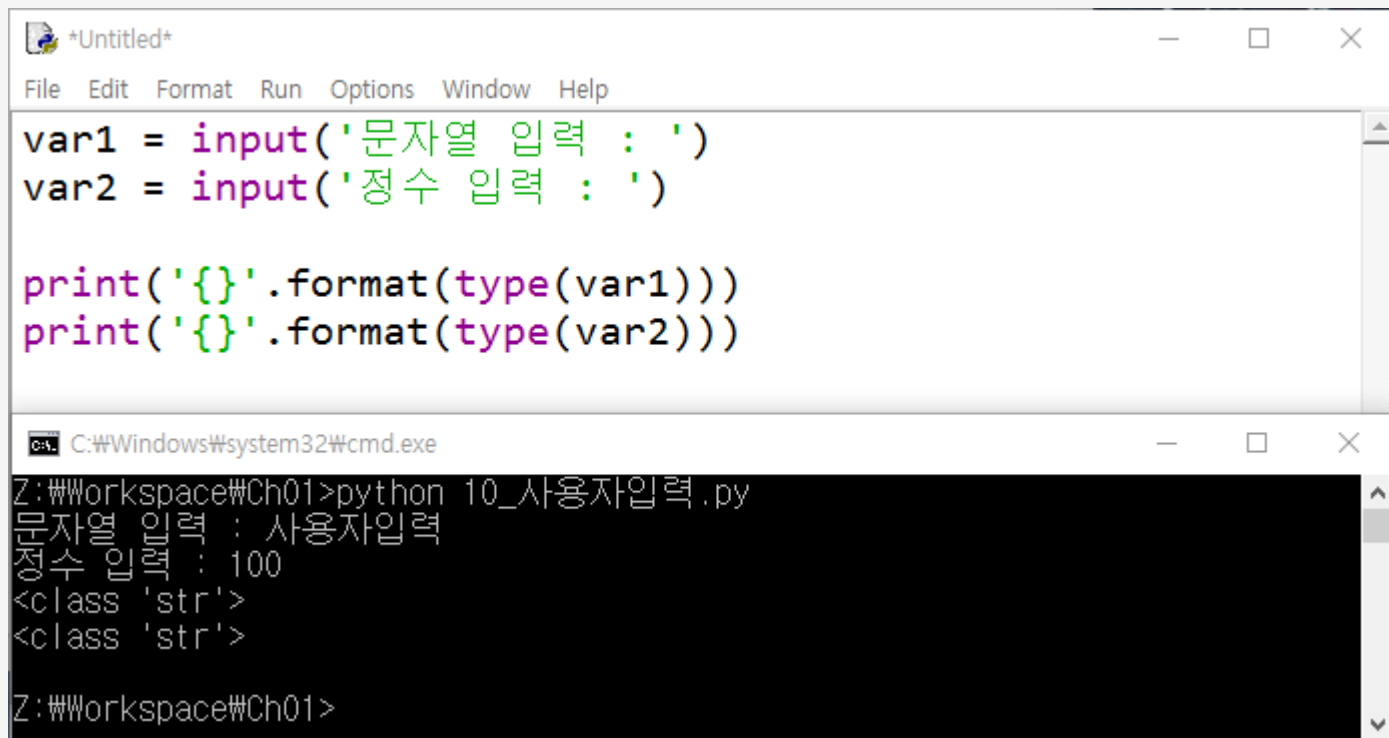
```
var = input('정수 입력 : ')
print('{}을 입력 받았습니다.'.format(var))
```

The bottom window is a command prompt titled 'C:\Windows\system32\cmd.exe'. It shows the execution of the Python script:

```
Z:\workspace\Ch01>python 09_사용자입력.py
정수 입력 : 30
30을 입력 받았습니다.
Z:\workspace\Ch01>
```

input() 함수 (예제)

사용자로부터 입력 받은 값은 항상 문자열이다.



```
*Untitled*
File Edit Format Run Options Window Help
var1 = input('문자열 입력 : ')
var2 = input('정수 입력 : ')

print('{}'.format(type(var1)))
print('{}'.format(type(var2)))

C:\Windows\system32\cmd.exe
Z:\Workspace\Ch01>python 10_사용자입력.py
문자열 입력 : 사용자입력
정수 입력 : 100
<class 'str'>
<class 'str'>

Z:\Workspace\Ch01>
```


Python 연산자

산술 연산자

연산자	예제	설명
+	$3 + 2$	두 값을 더한 결과를 반환
-	$3 - 2$	두 값을 뺀 결과를 반환
*	$3 * 2$	두 값을 곱한 결과를 반환
/	$3 / 2$	두 값을 나눈 결과를 반환(실수 값)
//	$3 // 2$	두 값을 나눈 결과의 몫 반환(정수 값)
%	$3 \% 2$	두 값을 나눈 결과의 나머지 반환
**	$3 ** 2$	거듭 제곱의 결과 반환

비교 연산자

연산자	예제	설명
==	3 == 3	두 피 연산자 값을 비교하여 동일하면 True, 동일하지 않으면 False
!=	3 != 2	두 피 연산자 값을 비교하여 동일하면 False, 동일하지 않으면 True
>	3 > 2	두 피 연산자 값을 비교하여 왼쪽의 값이 크면 True, 그렇지 않으면 False
<	2 < 3	두 피 연산자 값을 비교하여 왼쪽의 값이 작으면 True, 그렇지 않으면 False
>=	3 >= 2	두 피 연산자 값을 비교하여 왼쪽의 값이 크거나 같으면 True, 그렇지 않으면 False
<=	3 <= 3	두 피 연산자 값을 비교하여 왼쪽의 값이 작거나 같으면 True, 그렇지 않으면 False

논리 연산자

연산자	예제	설명
and	True and True True and False	두 피 연산자가 전부 True인 경우에만 True (논리곱)
or	True or True True or False	두 피 연산자가 전부 False인 경우에만 False (논리합)
not	not True not False	오른쪽 피 연산자에 대한 부정

멤버 연산자

연산자	예제	설명
in	1 in (1, 2, 3)	왼쪽 피 연산자의 값이 오른쪽 피 연산자 멤버 중 일치하는 값이 존재 하면 True
not in	1 not in (1, 2, 3)	왼쪽 피 연산자의 값이 오른쪽 피 연산자 멤버 중 일치하는 값이 존재 하지 않으면 True

식별 연산자

연산자	예제	설명
is	<code>type(1) is int</code>	두 피 연산자의 식별 값을 비교하였을 때 동일한 객체이면 True
is not	<code>type('1') is not int</code>	두 피 연산자의 식별 값을 비교하였을 때 동일한 객체이면 False

비트 연산자

연산자	예제	설명
&	10 & 5	두 피 연산자의 and 비트 연산을 수행 한다.
	10 5	두 피 연산자의 or 비트 연산을 수행 한다.
^	10 ^ 5	두 피 연산자의 xor 비트 연산을 수행 한다.
~	~10	피연산자에서 1비트를 더한 음수 값
<<	10 << 2	왼쪽 피 연산자의 비트를 왼쪽으로 2개 비트 이동
>>	10 >> 2	왼쪽 피 연산자의 비트를 오른쪽으로 2개 비트 이동