Python 시작하기

백석대학교 강윤희

(문제풀이) Python 언어 특징 이해하기

- Python은 네덜란드의 귀도 반 로섬(Guido Van Rossum)이 개발한 해석 기반의 (　　) 언어임
- Python 은 온라인 사진 공유 서비스인 인스타그램(Instagram), 파일 동기화 서비스인 (　　)등의 개발언어임
- Python 은 실행시점에서 자료형이 결정되는 (　　)을 지원함
- Python 은 가독성이 높다. 대표적인 예는 무엇인가

Python 다운로드 및 설치

Should I use Python 2 or Python 3 for my development activity?

What are the differences?

Short version: Python 2.x is legacy, Python 3.x is the present and future of the language

Python 3.0 was released in 2008. The final 2.x version 2.7 release came out in mid-2010, with a statement of extended support for this end-of-life release. The 2.x branch will see no new major releases after that. 3.x is under active development and has already seen over five years of stable releases, including version 3.3 in 2012 and 3.4 in 2014. This means that all recent standard library improvements, for example, are only available by default in Python 3.x.

Guido van Rossum (the original creator of the Python language) decided to clean up Python 2.x properly, with less regard for backwards compatibility than is the case for new releases in the 2.x range. The most drastic improvement is the better Unicode support (with all text strings being Unicode by default) as well as saner bytes/Unicode separation.

Python 3.3.4	2014-02-9	Download	Release Notes
Python 3.3.3	2013-11-17	Download	Release Notes
Python 2.7.6	2013-11-10	Download	Release Notes

2.X버전과 3.X버전 호환 안됨!!

https://www.python.org/downloads/

버전 차이 예

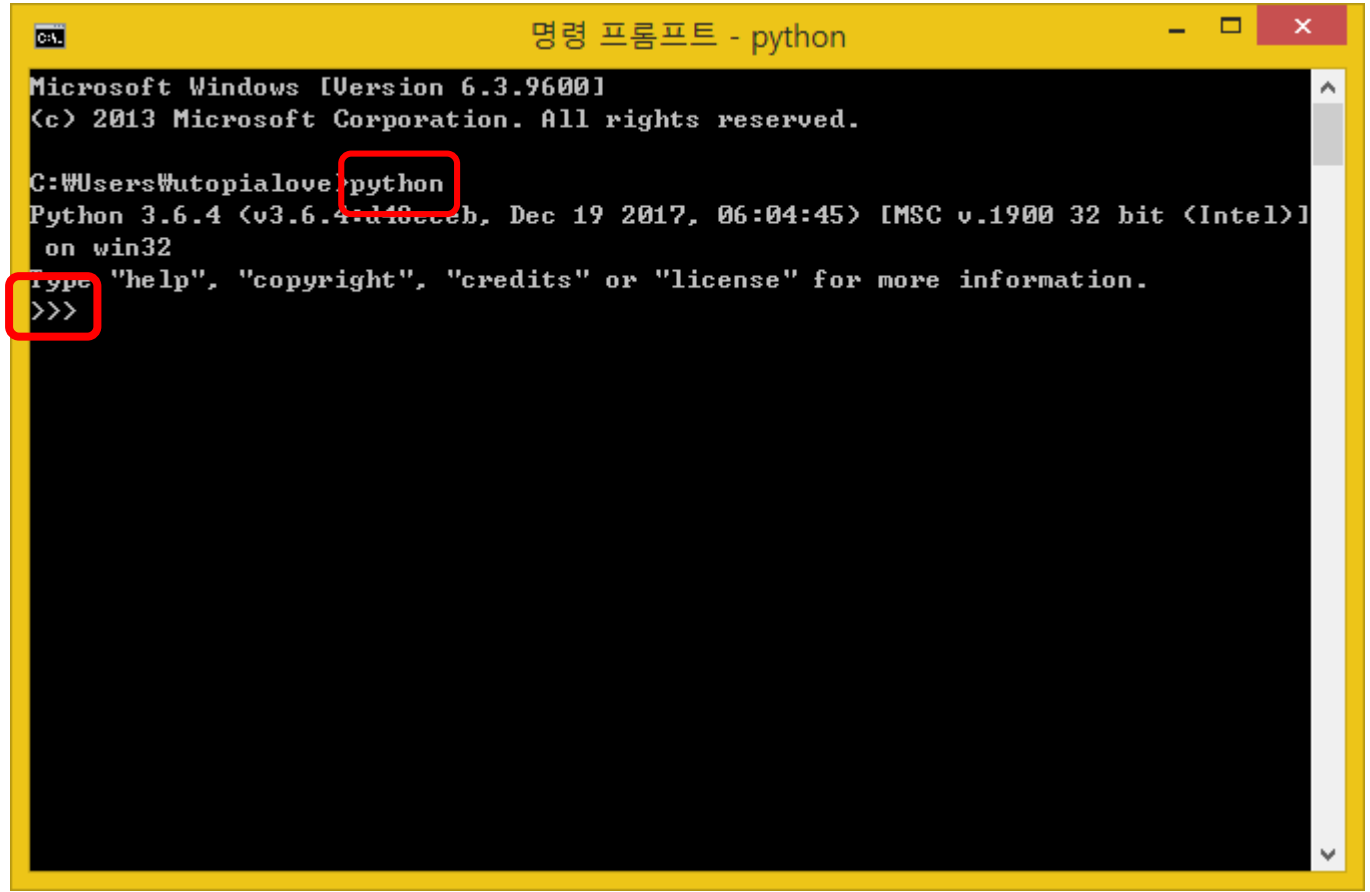
- Python3:
  >>> 10 / 3
  3.3333333333333335
- Python 2.x:
  >>> 10 / 3
  3

- Python3:
  >>> print (“hello Python”)
- Python2:
  >>> print “hello Python”

Python3에서는 long 형 자료형이 int형으로 통일됨

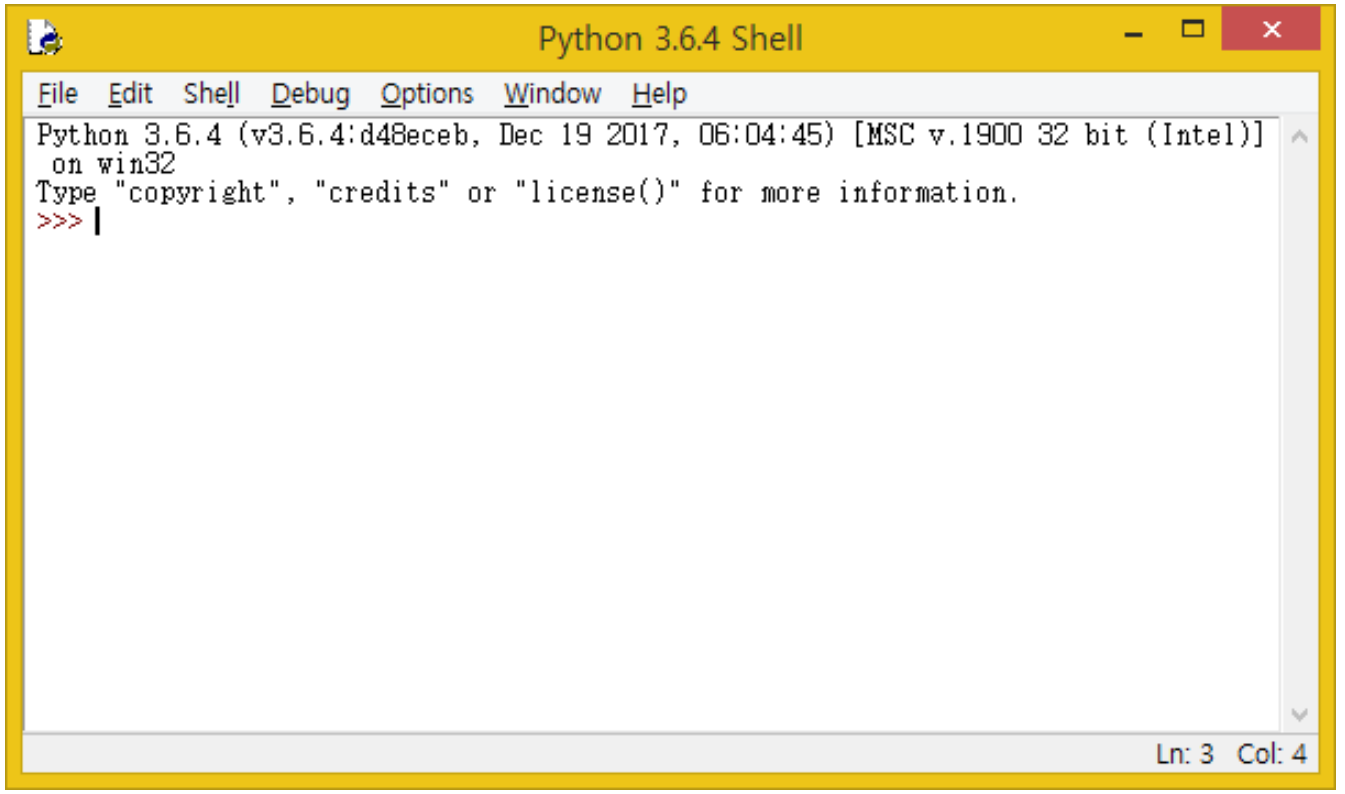
파이썬 시작하기

- DOS 명령 프롬프트에서 "python"이라고 입력



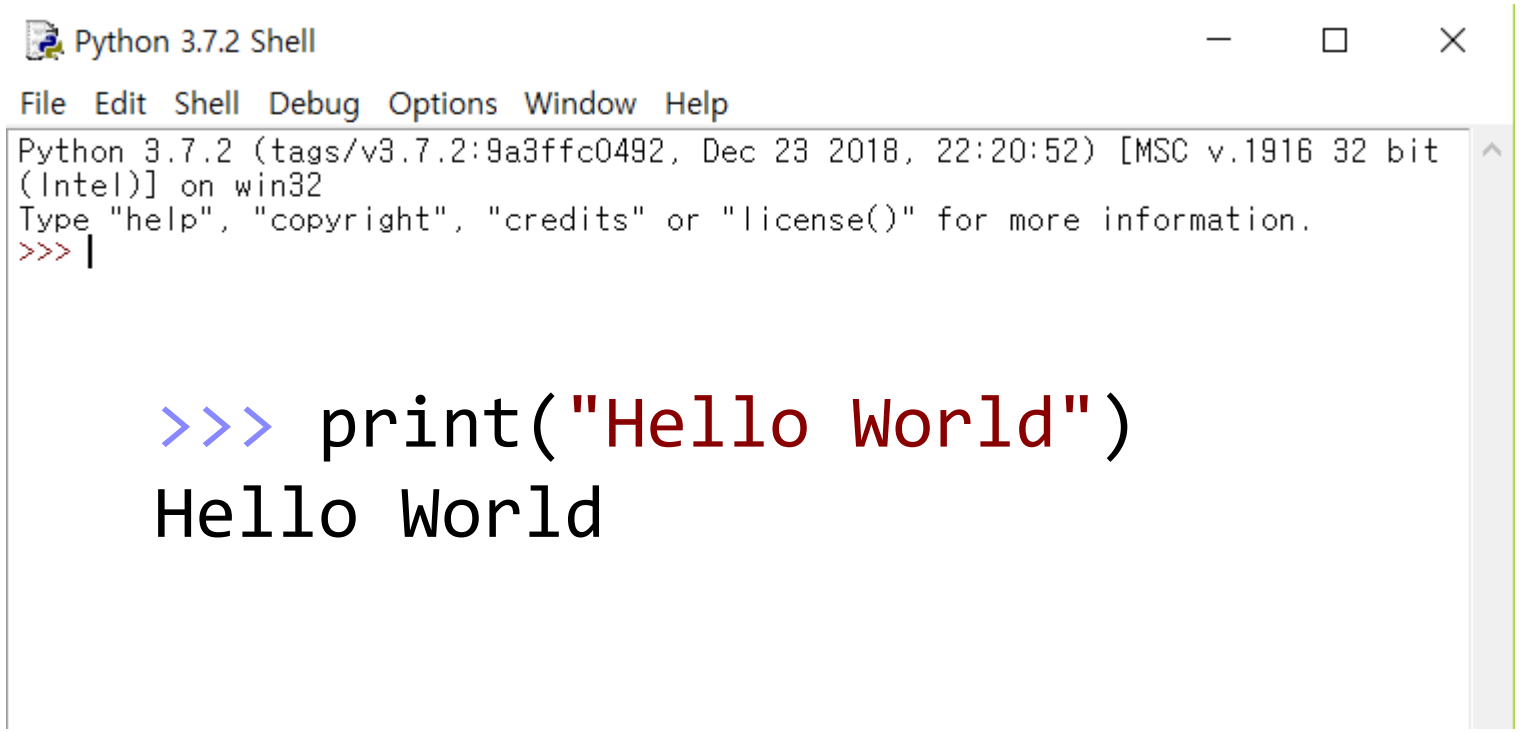
파이썬 시작하기

- 윈도우의 시작 메뉴에서 "IDLE" 프로그램을 찾아서 실행



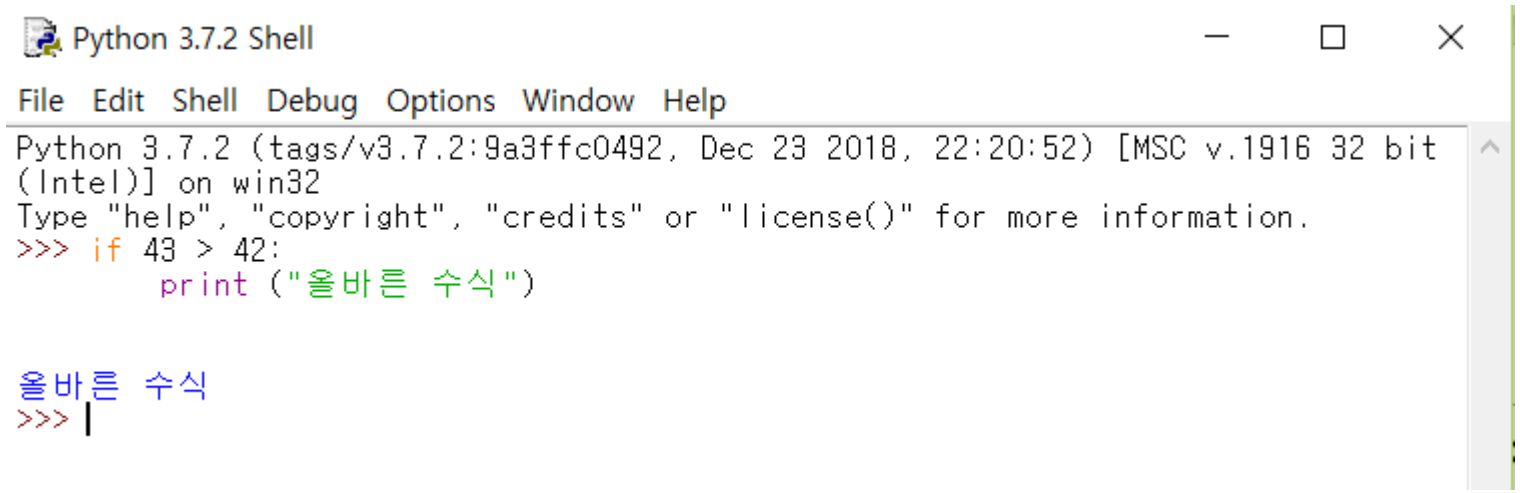
IDLE(Integrated DeveLopment Environment)

- 통합 개발 환경
- 세개의 닫는 꺽쇠(>>>) 프롬프트에 코드 입력



IDLE 사용하기

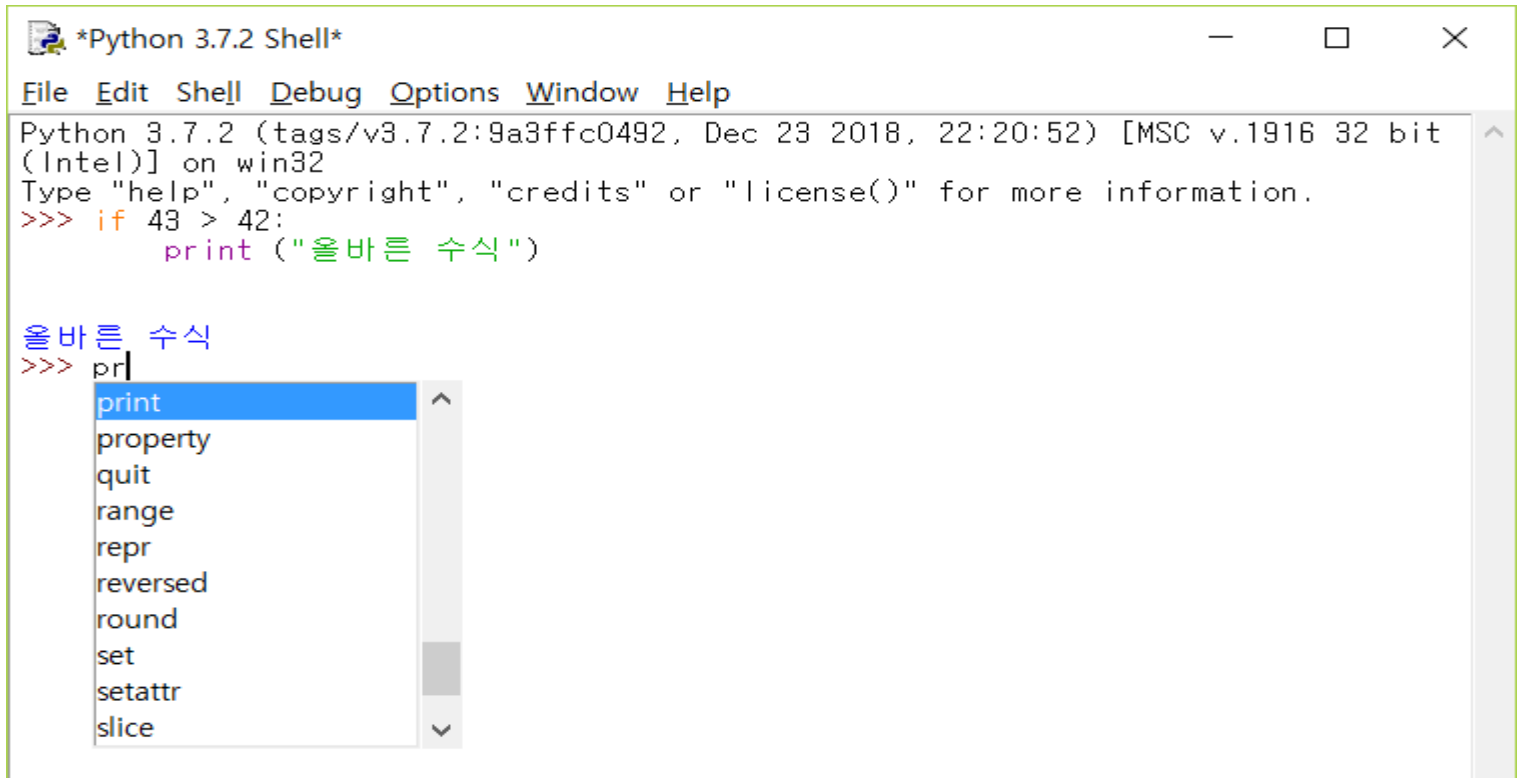
- Python 코드를 입력 후 수행 후 결과를 출력함



- IDLE을 사용 구문 및 들여쓰기 규칙을 이해함

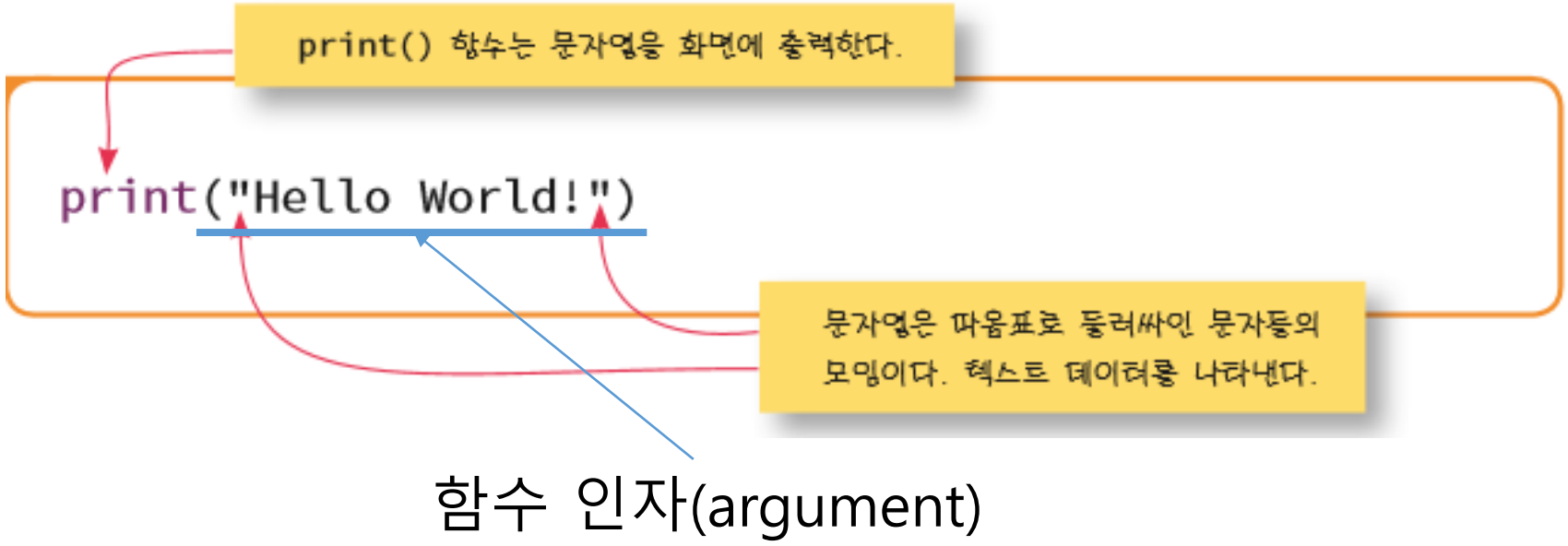
IDLE 사용하기

- 탭(tab)을 사용하여 자동완성 활용하기



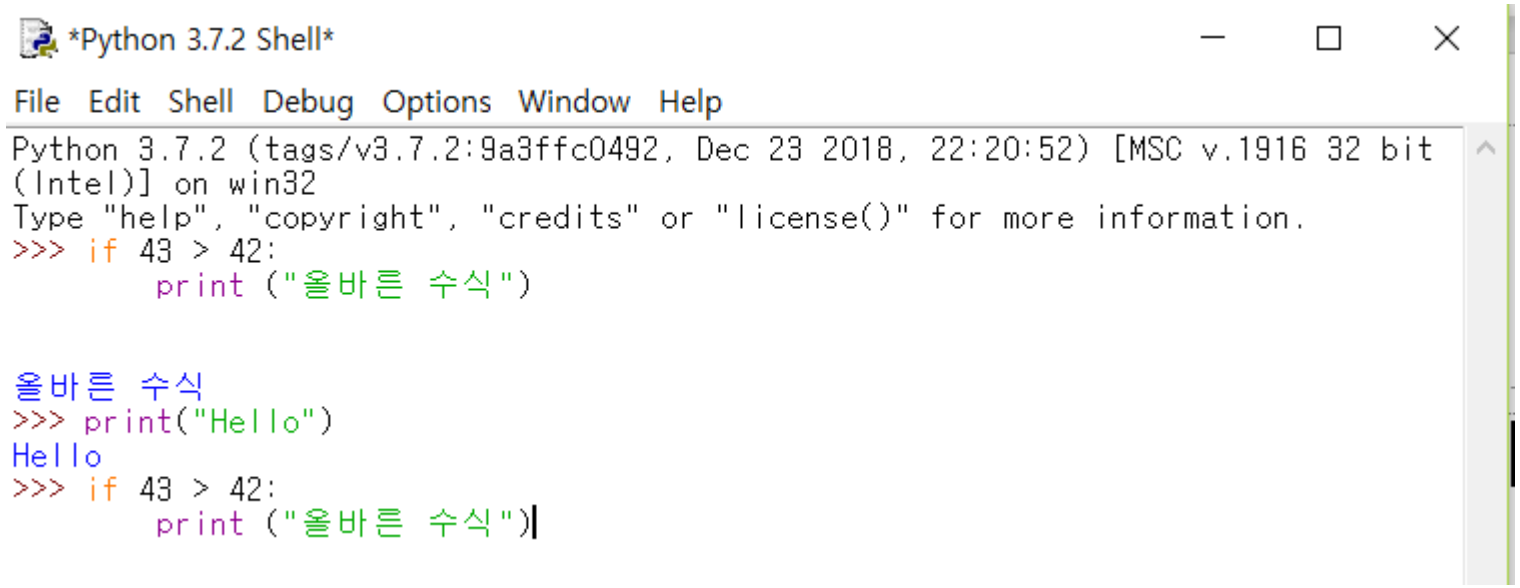
문장

- 프로그램은 여러 줄의 명령어로 구성
  - 한 줄의 명령어를 **문장(statement)**이라 함
  - 문장은 파이썬 **인터프리터**에 의하여 실행



IDLE 사용하기

- 코드 문장 불러오기 (Alt-N, Alt-P)



IDLE 사용하기

- 대화형 모드 프로그래밍
  - 작은 명령들이 모여 실행 가능한 파일을 구성

```
$ python

>>> print ("Hello world!")
Hello world!
```

- 스크립트 모드 프로그래밍
  - 소스코드가 기록된 파일을 구성 (확장자 .py)

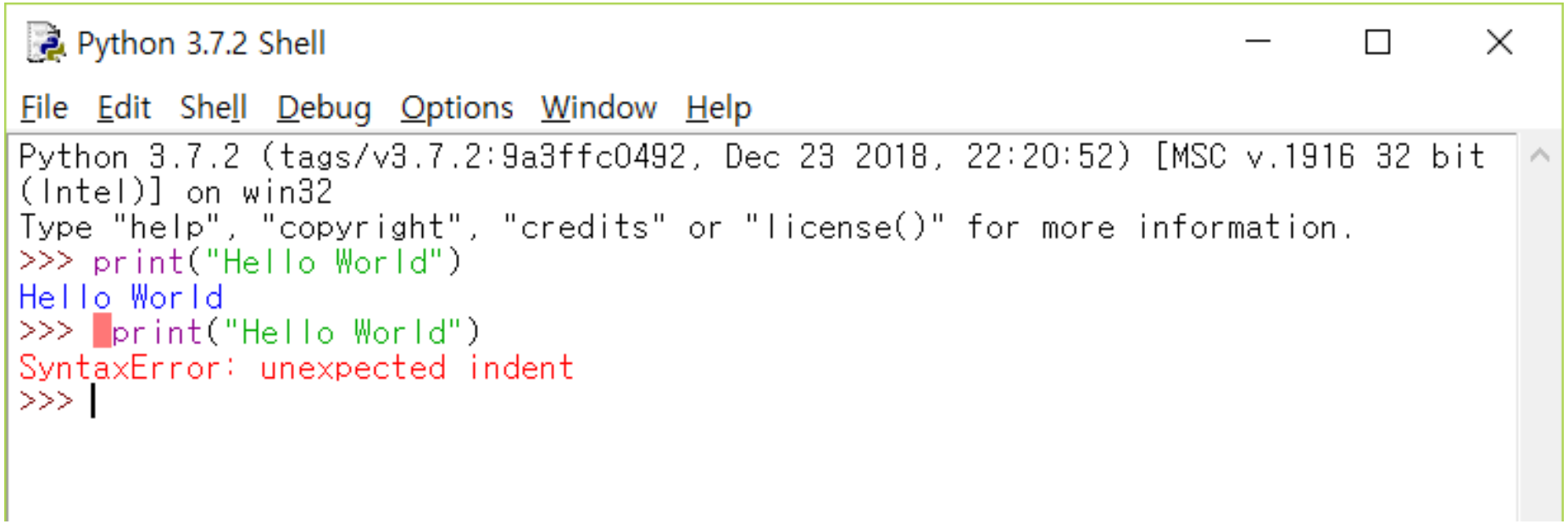
```
$ python hello.py
Hello world!
```

IDLE 사용하기

```
>>> 1 + 2
3
>>> name = "Sarah"
>>> "Hello " + name
'Hello Sarah'
```

들여쓰기

- Python은 들여쓰기(indent)에 민감함



파이썬 들여쓰기 오류의 예

들여쓰기

- 파이션은 { } 를 사용하지 않음
- 중첩되는 문장은 들여쓰기를 사용하여 구분함 (PEP 4칸 들여쓰기, https://pep8.org/)
- 출력 내용을 예상 하시오

```
for i in range(10):
     print("Hello")

for i in range(2):
    print("A")
    print("B")

for i in range(2):
     print("A")
print("B")
```

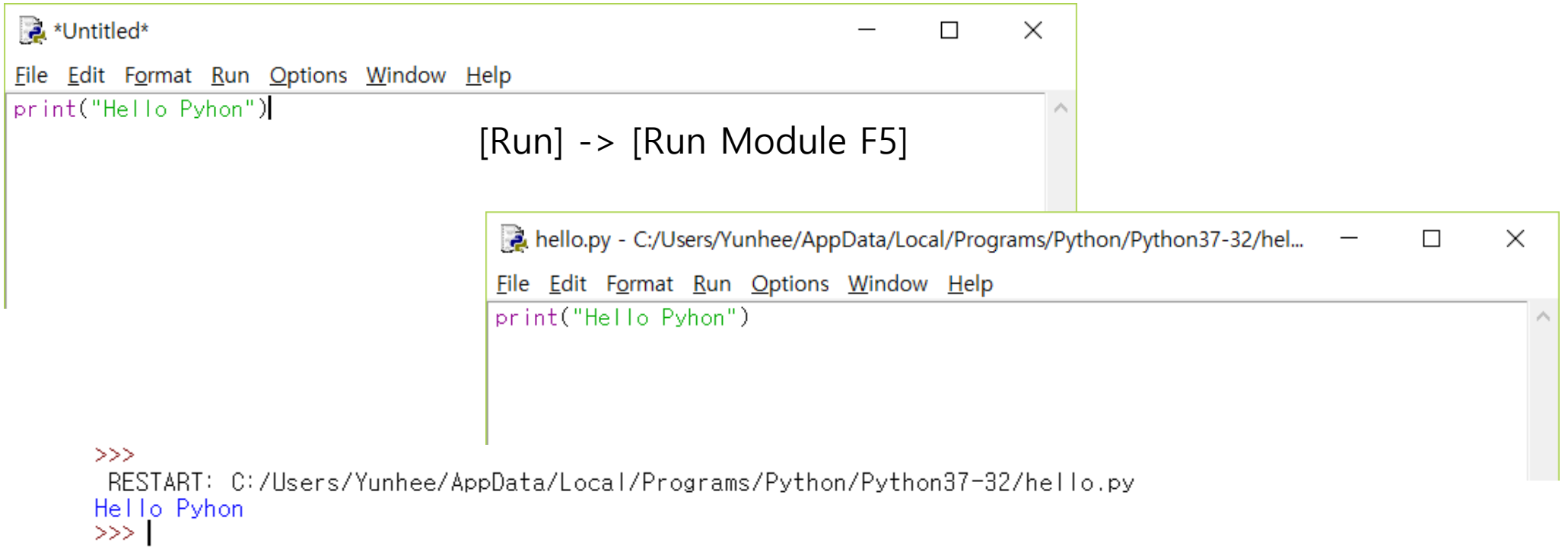
(참고) 들여쓰기 규칙

```
# Aligned with opening delimiter.
foo = long_function_name(var_one, var_two,
                         var_three, var_four)

# More indentation included to distinguish this from the rest.
def long_function_name(
        var_one, var_two, var_three,
        var_four):
    print(var_one)

# Hanging indents should add a level.
foo = long_function_name(
    var_one, var_two,
    var_three, var_four)
```

```
# Arguments on first line forbidden when not using vertical alignment.
foo = long_function_name(var_one, var_two,
    var_three, var_four)

# Further indentation required as indentation is not distinguishable.
def long_function_name(
    var_one, var_two, var_three,
    var_four):
    print(var_one)
```

(실습) IDLE 사용

- IDLE을 실행하여 “Hello Python”을 출력하고 hello.py에 저장함



(실습)연습문제

- 다음 다섯 가지 파이썬 표현 중 정상적으로 화면에 값이 출력되는 것을 고르시오.

1) print("I love 'you'")

2) print("I like you')

3) print('Korea')

4) print{Hello}

5) print[Hello]

```
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 22:20:52) [MSC v.1916 32 bit
(Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print("I love 'you'")
I love 'you'
>>> print("I like you')

SyntaxError: EOL while scanning string literal
>>> print('Korea')

Korea
>>> print{Hello}

SyntaxError: invalid syntax
>>> print[Hello]

Traceback (most recent call last):
  File "<pyshell#4>", line 1, in <module>
    print[Hello]
NameError: name 'Hello' is not defined
>>>
```