

Chapter 1

What Is Computational thinking?

Contents

- Python Programming Language
- Python Features
- Python Installation
- Python IDLE
- Data Output
- Formatting Output
- Escape Sequence
- Displaying Korea Population Statistics

Python Programming Language

- Guido van Rossum이 1991년에 발표한 프로그래밍 언어
- Guido가 즐겨보던 코메디 쇼인 Monty Python's Flying Circus에서 언어의 이름을 따옴
- 전세계 대학 및 교육기관을 비롯하여 여러 기업과 단체에서 이용
 - 기업: Youtube, Google, Yahoo, NASA, 다음카카오 등
 - 어플리케이션: Torrent, 문명IV, 카카오톡 등
- Python 3 버전이 2008년 발표 됨
 - 본 수업에서 사용할 버전



Python Features

■ 간단한 문법 구조

- 초보자도 쉽게 이해할 수 있는 명확하고 간단한 문법
- cf) C, C++, JAVA

■ 함축적이고 강력한 프로그래밍 특징

- 소수의 명령문으로 효율적인 프로그램 작성이 가능

■ 다양한 모듈(기능) 지원

- 여러가지 필요한 기능을 추가하여 활용할 수 있음

■ 들여쓰기 문법

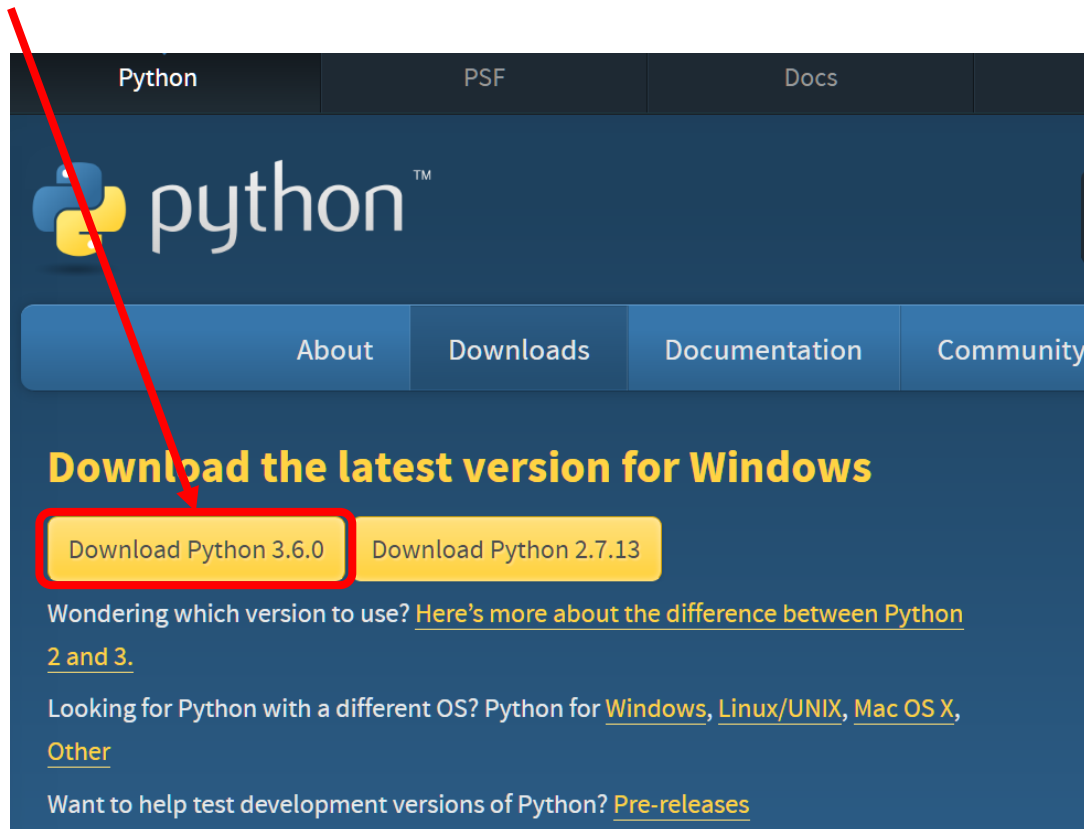
- 초보자가 올바른 프로그래밍 작성 스타일을 익힘

■ 단점

- C/C++ 언어에 비해 실행 속도가 느림
- 최근에는 하드웨어의 처리속도가 매우 빠르고, 또한 Python의 실행 속도를 개선할 수 있는 기술적인 해결 방법이 있음

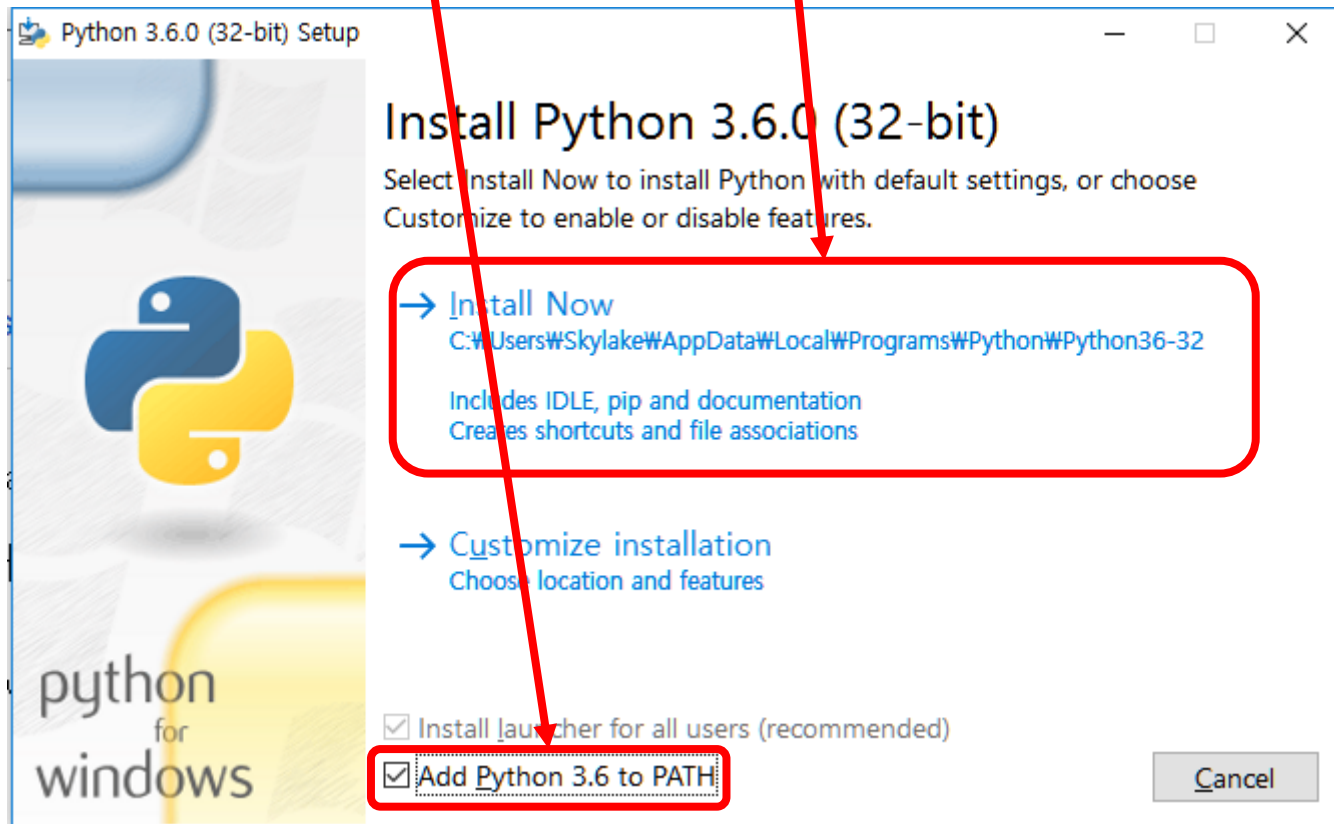
Python Installation

- Python download and install
 - <https://www.python.org/downloads/>
 - Python 최신 버전을 다운로드



Python Installation

- Python download and install
 - <https://www.python.org/downloads/>
 - Add Python 3.6 to PATH 및 Install Now



Python IDLE

■ IDLE란?

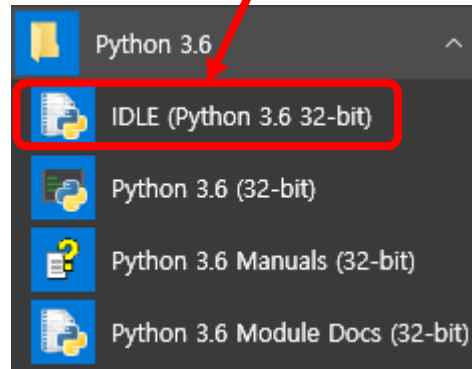
- Integrated DeveLopment Environment
- 프로그램 개발을 위한 여러 도구들의 집합

■ IDLE의 구성 요소

- Text editor
 - ✓ Python 코드를 작성하기 위한 텍스트 편집 기능
- Python shell
 - ✓ Python 코드를 직접 실행해 볼 수 있는 명령창
- Debugger
 - ✓ Python 코드의 오류를 찾기 위한 기능

Python IDLE

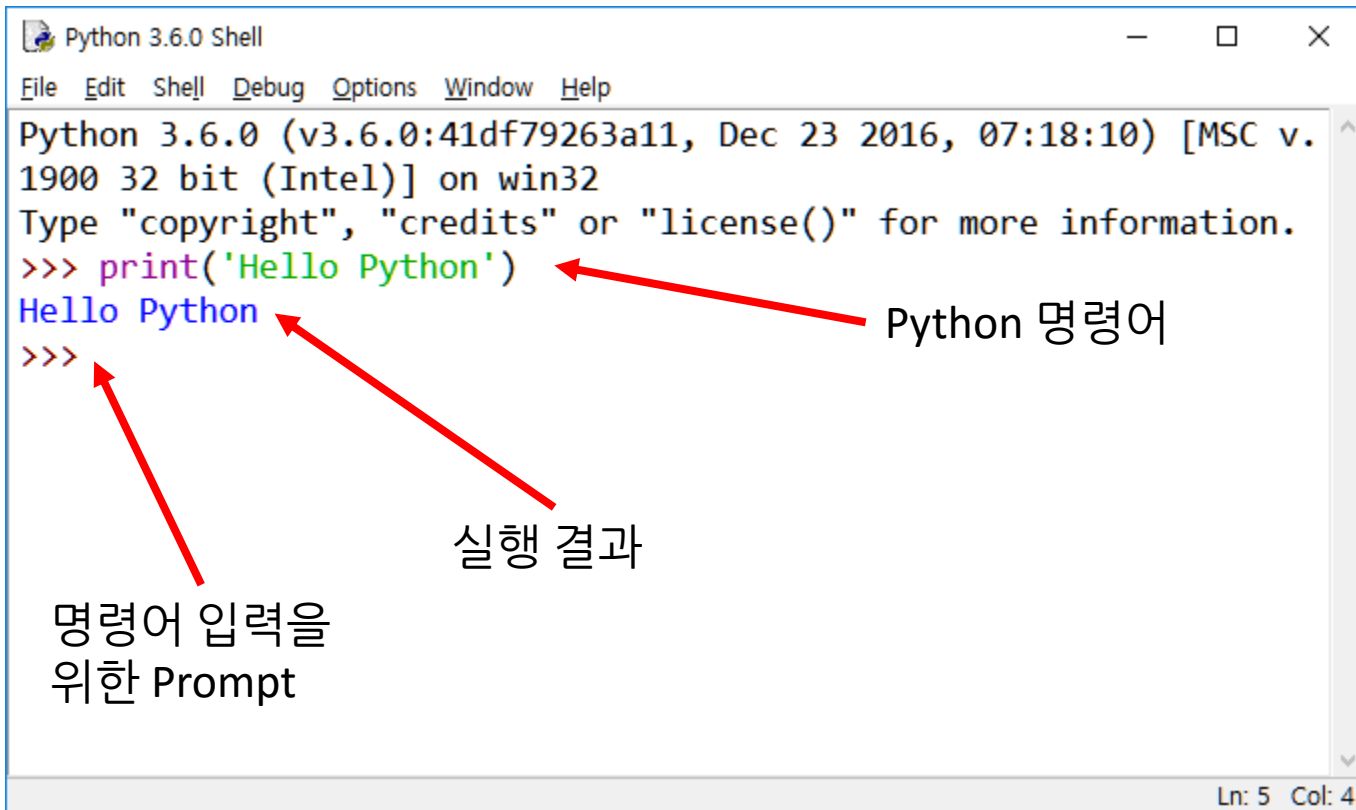
- Python IDLE의 실행
 - Python 3.6 프로그램 그룹에서 IDLE를 실행



Python IDLE

■ Python Shell

- IDLE 구성 요소 중 하나로 Python 명령어를 입력하고 즉시 실행 결과를 확인할 수 있음
- >>>(Prompt)에서 Python 명령을 입력 후 Enter
- 명령 실행 결과가 화면에 출력 됨



The screenshot shows the Python 3.6.0 Shell window. The title bar reads "Python 3.6.0 Shell". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main text area displays the following content:

```
Python 3.6.0 (v3.6.0:41df79263a11, Dec 23 2016, 07:18:10) [MSC v. 1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> print('Hello Python')
Hello Python
>>>
```

Three red arrows point from Korean text labels to the shell content:

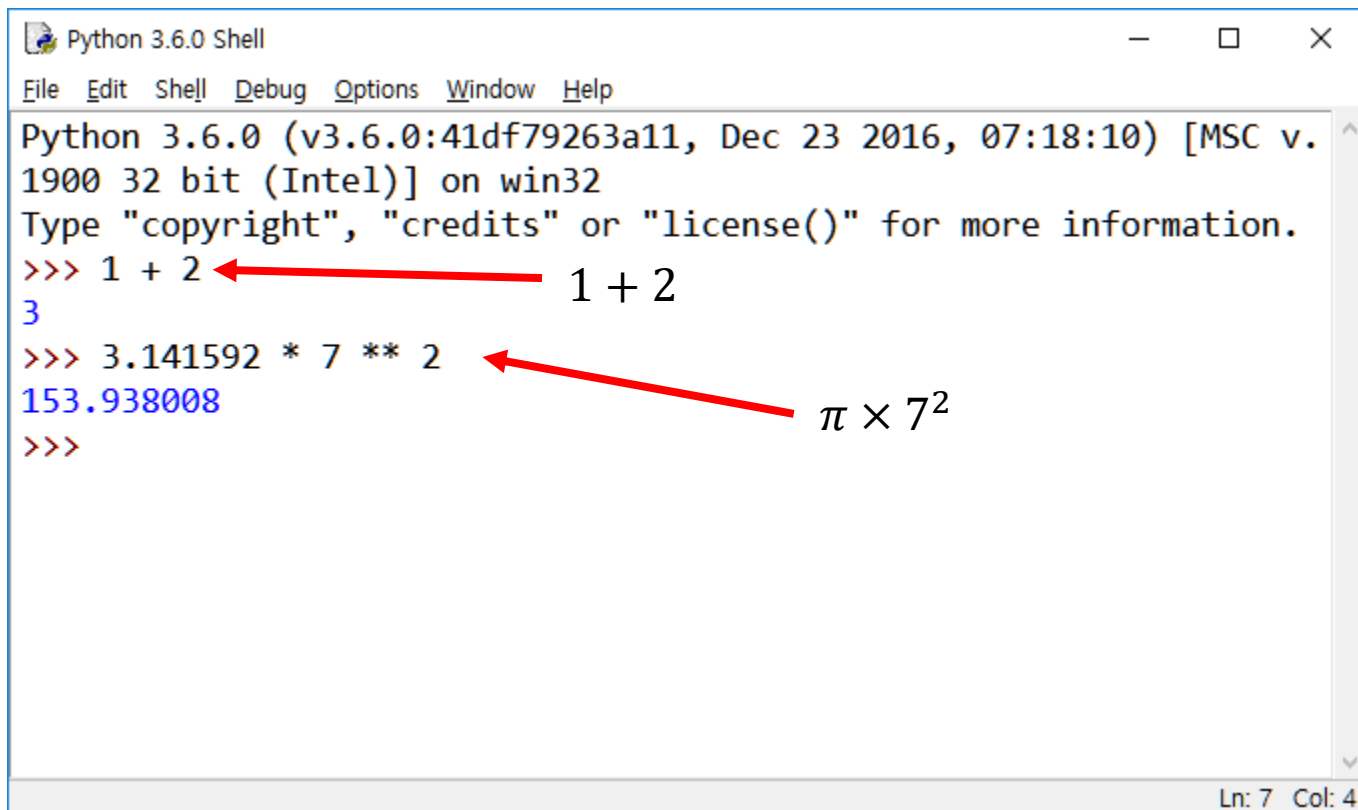
- An arrow points from "Python 명령어" (Python command) to the line `print('Hello Python')`.
- An arrow points from "실행 결과" (Execution result) to the output `Hello Python`.
- An arrow points from "명령어 입력을 위한 Prompt" (Prompt for command input) to the prompt `>>>`.

The status bar at the bottom right shows "Ln: 5 Col: 4".

Python IDLE

■ Python Shell

- IDLE 구성 요소 중 하나로 Python 명령어를 입력하고 즉시 실행 결과를 확인할 수 있음
- Prompt에서 Python 명령을 입력 후 Enter
- 명령 실행 결과가 Shell에 출력 됨



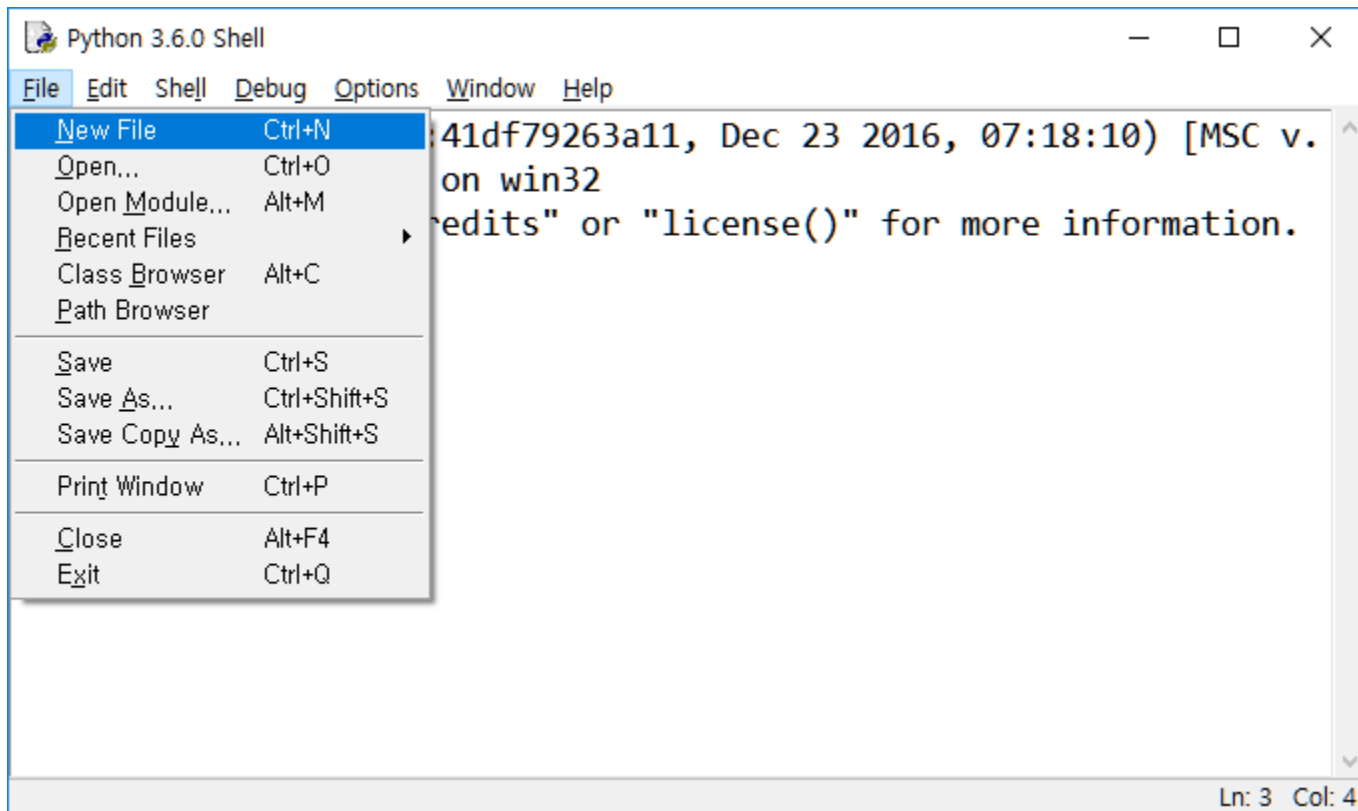
```
Python 3.6.0 Shell
File Edit Shell Debug Options Window Help
Python 3.6.0 (v3.6.0:41df79263a11, Dec 23 2016, 07:18:10) [MSC v.
1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 1 + 2 1 + 2
3
>>> 3.141592 * 7 ** 2 153.938008
>>>
```

The screenshot shows the Python 3.6.0 Shell window. It displays the Python version and environment information. Two calculations are shown: `1 + 2` resulting in `3`, and `3.141592 * 7 ** 2` resulting in `153.938008`. Red arrows point from the mathematical expressions $1 + 2$ and $\pi \times 7^2$ to their respective results in the shell output.

Python IDLE

■ Python Editor

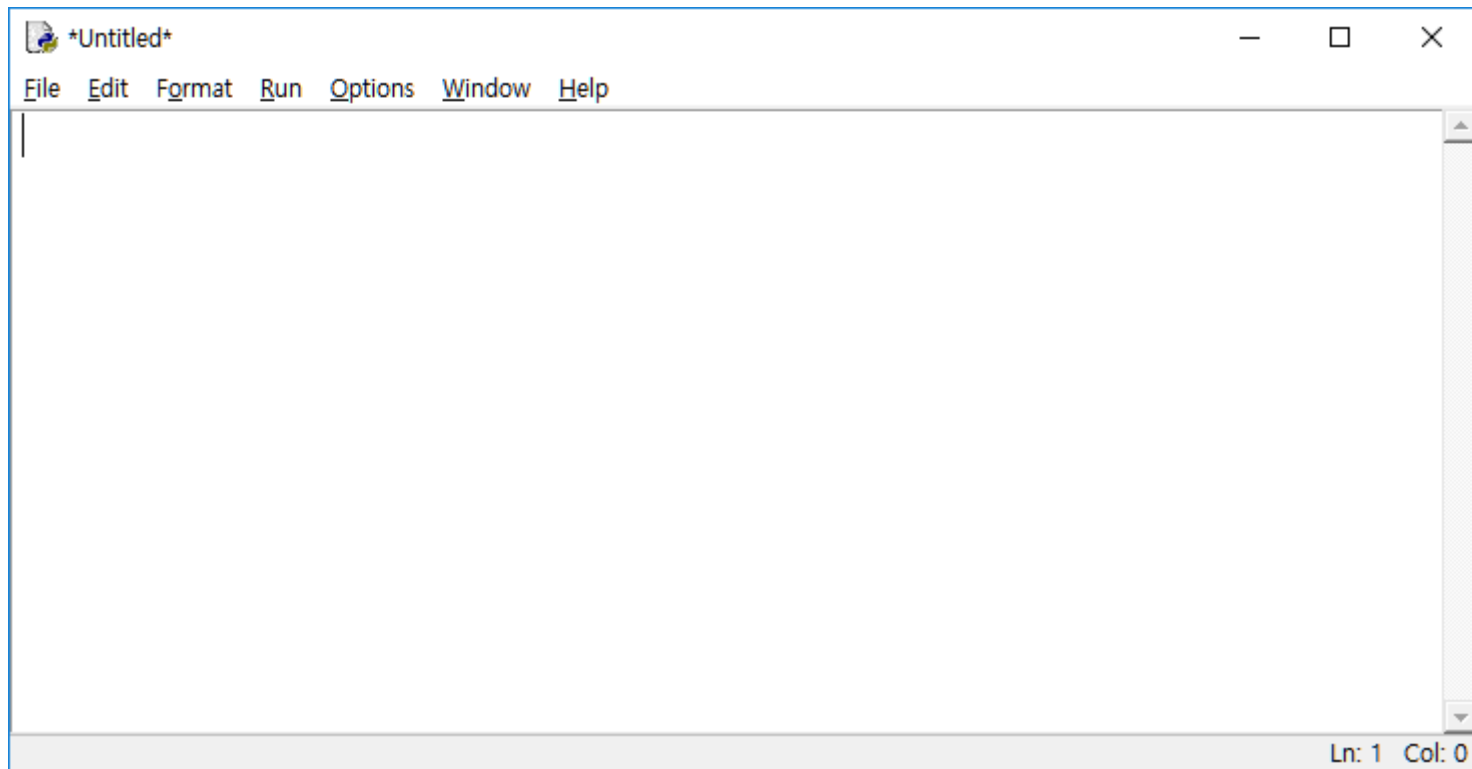
- Python 소스 코드를 작성하기 위한 텍스트 편집기
- Python 명령문 작성 후 실행 키(F5)를 입력하면 실행 결과가 Python Shell에 출력 됨
- Editor 실행: File – New File 혹은 Ctrl-N 입력



Python IDLE

■ Python Editor

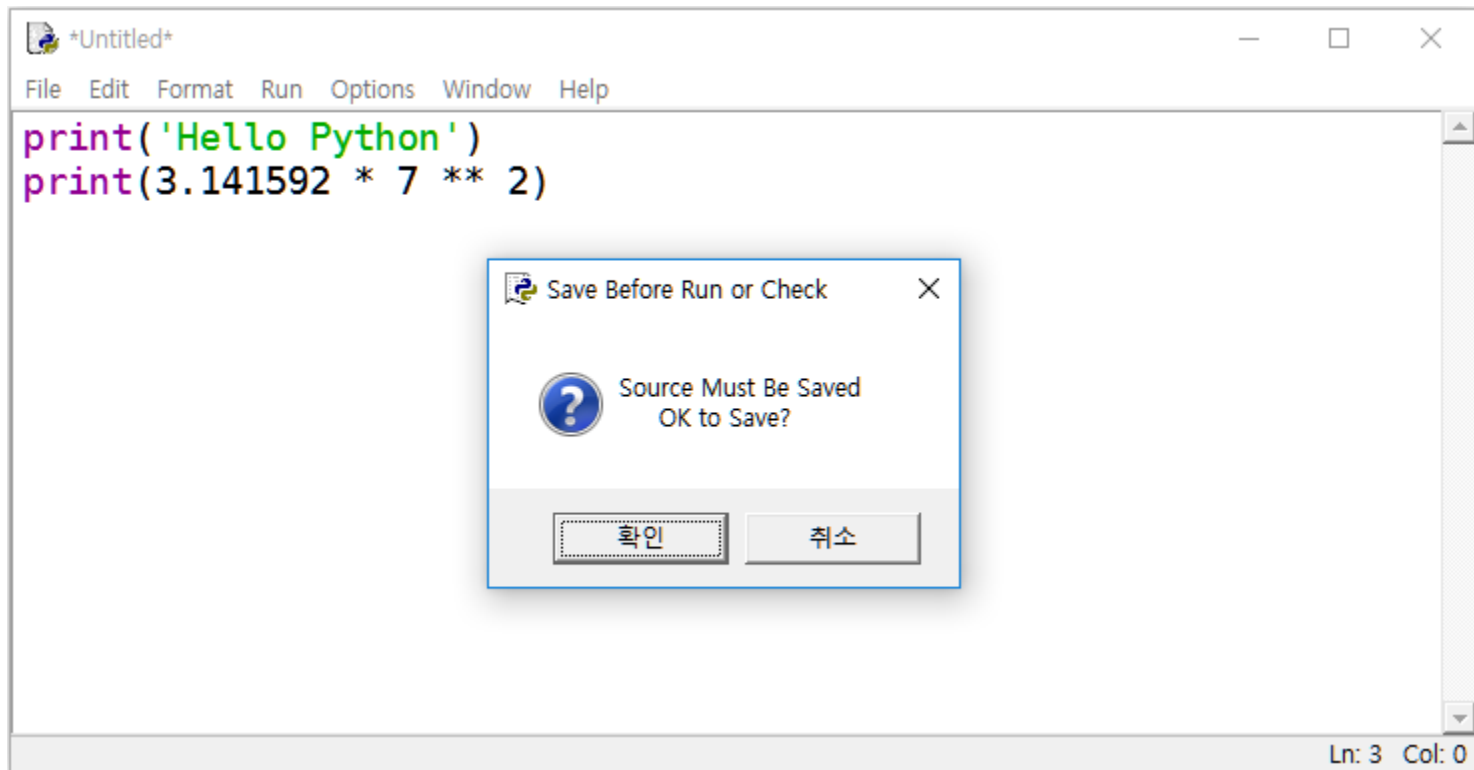
1. Editor에서 Python 명령문 작성
2. 프로그램 실행: **F5**



Python IDLE

■ Python Editor

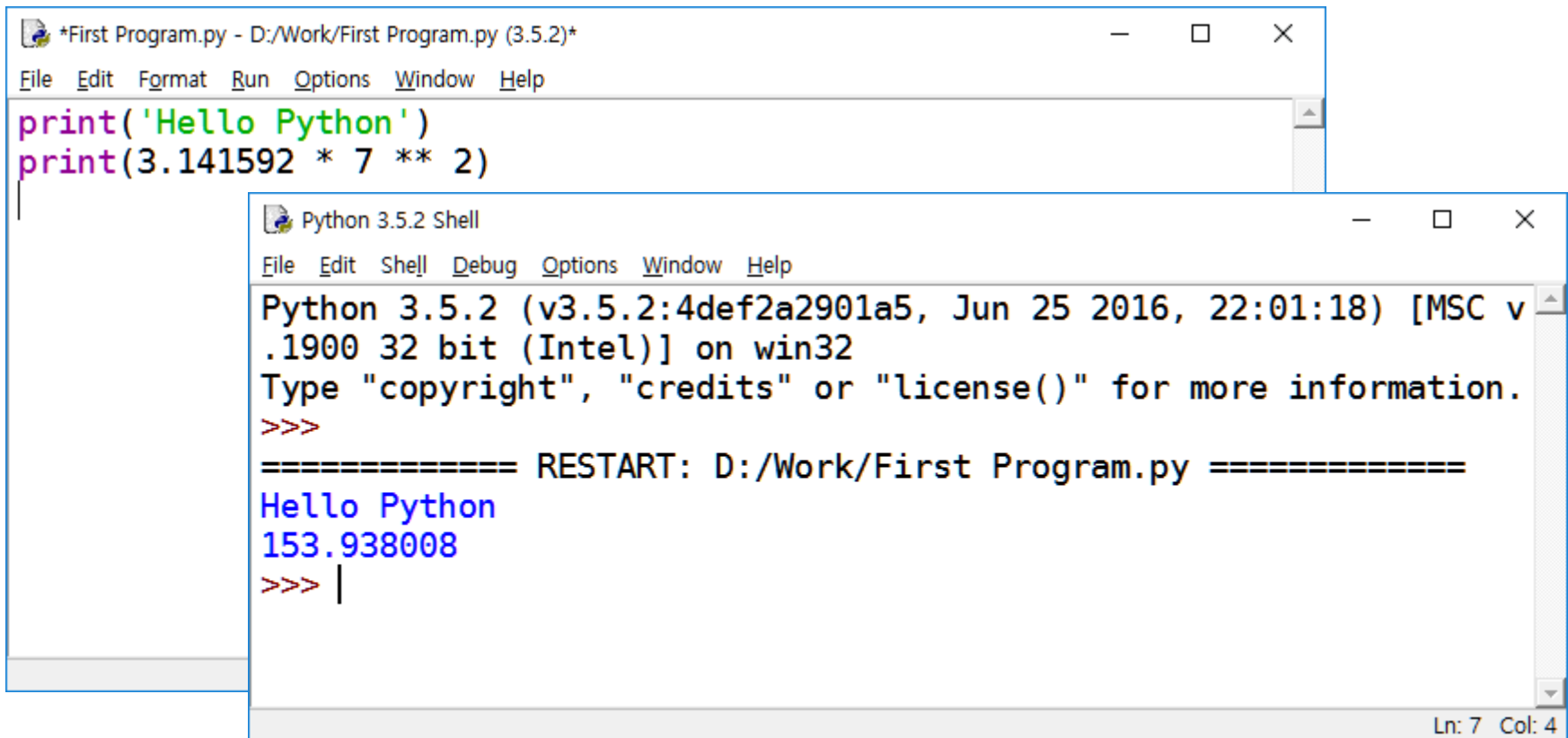
- Python 명령문은 실행 전 파일로 저장되어야 함
- Ctrl - S 입력 혹은 첫 실행 시 팝업 창에서 확인을 선택하여 파일을 저장



Python IDLE

■ Python Editor

- Python 명령문은 실행 전 파일로 저장되어야 함
- Ctrl - S 입력 혹은 첫 실행 시 팝업 창에서 확인을 선택하여 파일을 저장



The screenshot displays two windows from the Python IDLE environment. The top window, titled '*First Program.py - D:/Work/First Program.py (3.5.2)*', contains the following Python code:

```
print('Hello Python')
print(3.141592 * 7 ** 2)
```

The bottom window, titled 'Python 3.5.2 Shell', shows the execution output. It includes the Python version information and a restart message for the current file:

```
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:01:18) [MSC v
.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: D:/Work/First Program.py =====
Hello Python
153.938008
>>> |
```

The status bar at the bottom right of the shell window indicates 'Ln: 7 Col: 4'.

Data Output

■ 데이터 출력

- `print(data)`: 화면에 *data*를 출력하는 함수(function)

```
>>> print('Hello')  
Hello
```

출력할 데이터(문자열)



```
>>> print(10)  
10
```

출력할 데이터(정수)



Data Output

- 쉼표(comma)를 이용하여 여러 개의 데이터를 한 줄에 출력

```
>>> print('Welcome to', 'Python!')  
Welcome to Python!
```

쉼표로 여러 개의 데이터를
공백으로 연결하여 한
줄에 출력

정수	문자열	수식	문자열
↓	↓	↓	↓
>>> print(5, 'inch is', 2.54 * 5, 'cm')			
5 inch is 12.7 cm			

Data Output

■ Options

- `sep=""`: 데이터 연결 시 공백을 제거

```
>>> print(2.54 * 2, 'cm', sep='')  
5.08cm
```

- `end=""`: 줄 바꿈 문자를 제거

```
print('This is ', end='')  
print('a single line')
```

This is a single line

Formatting Output

■ 숫자 및 문자열의 형식 지정

`format(value, format_specifier)`

value: 형식을 지정할 값

format_specifier: 값에 적용할 형식

■ 소수점 이하 유효 숫자를 2로 지정

```
>>> format(3.141592, '.2f')  
'3.14'
```

■ 문자열의 우측 정렬

```
>>> format('Hello', '>10')  
'      Hello'
```

`format()`은 형식이 적용된 문자열을 반환(출력 함수가 아님에 유의)

Formatting Output

Format Specifier := [align][width][.precision][type]

문자열 형식 적용

폭 10칸 및 우측 정렬

```
>>> format('Hello', '>10')  
'      Hello'
```

폭 10칸 및 중앙 정렬

```
>>> format('Hello', '^10')  
'   Hello   '
```

폭 10칸 및 좌측 정렬(문자열의 정렬 기본값: '<')

```
>>> format('Hello', '<10')  
'Hello      '
```

```
>>> format('Hello', '10')  
'Hello      '
```

Formatting Output

Format Specifier := [align][width][.precision][type]

숫자 형식 적용

실수 출력('f' = float)

```
>>> format(5/9, 'f')  
'0.555556'
```

소수점 이하 유효자리 2자리

```
>>> format(5/9, '.2f')  
'0.56'
```

폭 10칸 및 우측 정렬(숫자의 정렬 기본값: '>')

```
>>> format(5/9, '10.2f')  
'      0.56'
```

Formatting Output

Format Specifier := [align][width][.precision][type]

그 외 출력 형식

1,000단위 콤마 넣기

```
>>> format(1000000, ',')  
'1,000,000'
```

16진수 및 2진수 형식 변환

```
>>> format(255, 'X')  
'FF'
```

```
>>> format(255, 'b')  
'11111111'
```

Escape Sequence

■ Escape sequence

- 문자로 표현할 수 없는 특수 문자를 backslash로 표현하는 방식
- Tab, new line, single quote, double quote 등
- single quote를 문자열에 포함 시켰을 경우:

```
>>> print('It's good')
```

SyntaxError: invalid syntax

- 'It's good' → 'It' 까지를 문자열로 인식

- Escape sequence를 이용한 해결

```
>>> print('It\'s good')
```

It's good

Escape Sequence

자주 사용되는 Escape Sequence 기호

Symbol	Meaning
\'	Single quote
\"	Double quote
\t	Tab
\n	New line
\\	Backslash

Displaying Korea Population Statistics

The Problem

좌측의 Korea Population Statics 표의 데이터를 그대로 이용하여 Python의 print()와 format()으로 우측 결과와 동일한 형식으로 출력하도록 프로그램을 구현

Korea Population Statistics		
Year	AP	NC
2013	50428893	3.37501
2014	50746659	3.305498
2015	51014947	3.185831
*"World's Population Prospects"		



```
Korea Population Statistics
Year |          AP          |  NC
2013 |    50,428,893    |  3.38
2014 |    50,746,659    |  3.31
2015 |    51,014,947    |  3.19

*"World's Population Prospects"
```

AP: Average Population

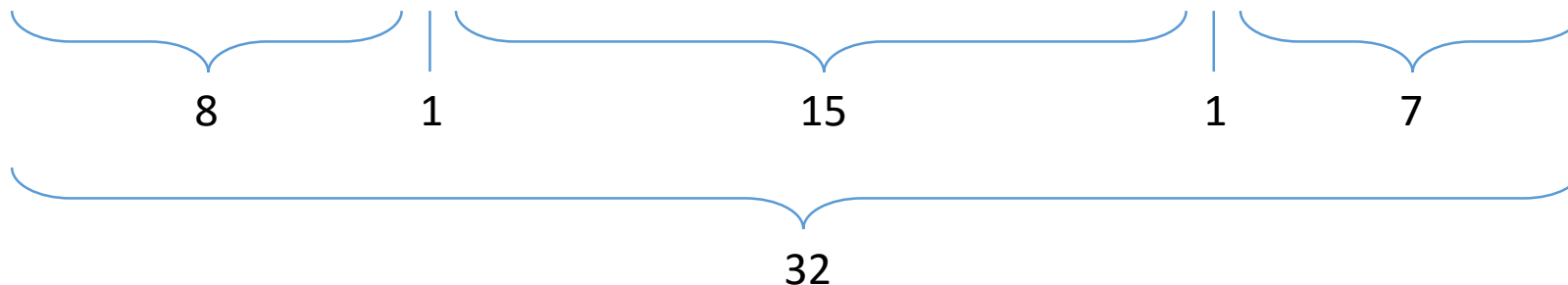
NC: Natural Change (per 1000)

Displaying Korea Population Statistics

Problem Analysis

출력 결과의 형식(format) 분석

		K	o	r	e	a		P	o	p	u	l	a	t	i	o	n		S	t	a	t	i	s	t	i	c	s				
		Y	e	a	r									A	P												N	C				
		2	0	1	3									5	0	,	4	2	8	,	8	9	3					3	.	3	8	
		2	0	1	4									5	0	,	7	4	6	,	6	5	9					3	.	3	1	
		2	0	1	5									5	1	,	0	1	4	,	9	4	7					3	.	1	9	
*	"	W	o	r	l	d	'	s		P	o	p	u	l	a	t	i	o	n		P	r	o	s	p	e	c	t	s	"		



Displaying Korea Population Statistics

Problem Analysis

center aligned

출력 결과의 형식(format) 분석

		K	o	r	e	a		P	o	p	u	l	a	t	i	o	n		S	t	a	t	i	s	t	i	c	s			
		Y	e	a	r									A	P												N	C			
		2	0	1	3									5	0	,	4	2	8	,	8	9	3					3	.	3	8
		2	0	1	4									5	0	,	7	4	6	,	6	5	9					3	.	3	1
		2	0	1	5									5	1	,	0	1	4	,	9	4	7					3	.	1	9
*	"	W	o	r	l	d	'	s		P	o	p	u	l	a	t	i	o	n		P	r	o	s	p	e	c	t	s	"	

left aligned
containing quotes

right aligned
thousand separators

right aligned
two decimal places

Displaying Korea Population Statistics

Program Design

Requirements

- 주어진 수치 데이터를 그대로 사용
- `format()`과 `print()`를 이용하여 출력 결과를 구성

Displaying Korea Population Statistics

Program Implementation

print() 명령문의 반복 사용, 끝!

```
print(format('Korea Population Statistics', '^32'))
print(format('Year', '^8'), '|', format('AP', '^15'), '|', format('NC', '^7'), sep='')
print(format(2013, '^8'), '|', format(50428893, '15, '), '|', format(3.37500964, '7.2f'), sep='')
print(format(2014, '^8'), '|', format(50746659, '15, '), '|', format(3.305498397, '7.2f'), sep='')
print(format(2015, '^8'), '|', format(51014947, '15, '), '|', format(3.185831008, '7.2f'), sep='')
print('\n*"World\'s Population Prospects"')
```

Caution: nested parenthesis and quotes

Displaying Korea Population Statistics

Program Execution

```
Korea Population Statistics
Year |          AP          | NC
2013 | 50,428,893 | 3.38
2014 | 50,746,659 | 3.31
2015 | 51,014,947 | 3.19
```

```
*"World's Population Prospects"
```

```
>>>
```