

01-1 파이썬 시작하기

파이썬이란?

파이썬(Python)은 1990 년 암스테르담의 귀도 반 로섬(Guido Van Rossum)이 개발한 인터프리터 ¹ 언어이다. 귀도는 파이썬이라는 이름을 자신이 좋아하는 코미디 쇼인 "몬티 파이썬의 날아다니는 서커스(Monty Python's Flying Circus)"에서 따왔다고 한다. 파이썬의 사전적인 의미는 고대 신화에 나오는 파르나소스 산의 동굴에 살던 큰 뱀을 뜻하며, 아폴로 신이 델파이에서 파이썬을 퇴치했다는 이야기가 전해지고 있다. 대부분의 파이썬 책 표지와 아이콘이 뱀 모양으로 그려져 있는 이유가 여기에 있다.



파이썬은 우리나라에서는 아직 대중적으로 사용되고 있지 않지만 외국에서는 교육 목적뿐 아니라 실무에서도 많이 사용되고 있다. 그 대표적인 예가 바로 구글이다. 필자는 구글에서 만들어진 소프트웨어의 50% 이상이 파이썬으로 만들어졌다는 이야기를 들은 적도 있다. 이외에도 많이 알려진 예를 몇 가지 들자면 파일 동기화 서비스인 드롭박스(Dropbox), 쉽고 빠르게 웹 개발을 할 수 있도록 도와주는 프레임워크인 장고(Django) 등이 있다.

또한 파이썬 프로그램은 공동 작업과 유지 보수가 매우 쉽고 편하다. 그 때문에 이미 다른 언어로 작성된 많은 프로그램과 모듈들이 파이썬으로 재구성되고 있다. 국내에서도 그 가치를 인정받아 사용자층이 더욱 넓어지고 있고, 파이썬을 이용해 프로그램을 개발하는 기업체들 또한 늘어 가고 있는 추세이다.

1. 인터프리터 언어란 한 줄씩 소스 코드를 해석해서 그때그때 실행해 결과를 바로 확인할 수 있는 언어이다. [↗](#)

01-2 파이썬의 특징

필자는 파이썬을 무척 좋아한다. 모든 프로그래밍 언어에는 각기 장점이 있지만 파이썬에는 다른 언어들에서는 쉽게 찾아볼 수 없는 파이썬만의 독특한 매력이 있다. 파이썬의 특징을 알면 왜 파이썬을 공부해야 하는지, 과연 시간을 투자할 만한 가치가 있는지 분명하게 판단할 수 있을 것이다.

파이썬은 인간다운 언어이다

프로그래밍이란 인간이 생각하는 것을 컴퓨터에 지시하는 행위라고 할 수 있다. 앞으로 살펴볼 파이썬 문법에서도 보게 되겠지만 파이썬은 사람이 생각하는 방식을 그대로 표현할 수 있는 언어이다. 따라서 프로그래머는 굳이 컴퓨터의 사고 체계에 맞추어서 프로그래밍을 하려고 애쓸 필요가 없다. 이제 곧 어떤 프로그램을 구상하자마자 머릿속에서 생각한 대로 술술 써 내려가는 여러분의 모습에 놀라게 될 것이다.

아래 소스 코드를 보면 이 말이 쉽게 이해될 것이다.

```
if 4 in [1,2,3,4]: print("4 가 있습니다")
```

위의 예제는 다음처럼 읽을 수 있다:

"만약 4 가 1,2,3,4 중에 있으면 "4 가 있습니다"를 출력한다."

프로그램을 모르더라도 직관적으로 무엇을 뜻하는지 알 수 있지 않는가? 마치 영어 문장을 읽는 듯한 착각에 빠져든다.

파이썬은 문법이 쉬워 빠르게 배울 수 있다

어려운 문법과 수많은 규칙에 둘러싸인 언어에서 탈피하고 싶지 않은가? 파이썬은 문법 자체가 아주 쉽고 간결하며 사람의 사고 체계와 매우 닮아 있다. 배우기 쉬운 언어, 활용하기 쉬운 언어가 가장 좋은 언어가 아닐까? 유명한 프로그래머인 에릭 레이먼드(Eric Raymond)는 파이썬을 공부한 지 단 하루 만에 자신이 원하는 프로그램을 작성할 수 있었다고 한다.

※ 프로그래밍 경험이 조금이라도 있는 사람이라면 파이썬의 자료형, 함수, 클래스 만드는 법, 라이브러리 및 내장 함수 사용 방법 등을 익히는데 1 주일이면 충분하리라 생각한다.

파이썬은 무료이지만 강력하다

오픈 소스¹인 파이썬은 당연히 무료이다. 사용료 걱정없이 언제 어디서든 파이썬을 다운로드하여 사용할 수 있다.

또한 프로그래머는 만들고자 하는 프로그램의 대부분을 파이썬으로 만들 수 있다. 물론 시스템 프로그래밍이나 하드웨어 제어와 같은 매우 복잡하고 반복 연산이 많은 프로그램은 파이썬과 어울리지

않는다. 하지만 파이썬은 이러한 약점을 극복할 수 있게끔 다른 언어로 만든 프로그램을 파이썬 프로그램에 포함시킬 수 있다.

파이썬과 C는 찰떡궁합이란 말이 있다. 즉, 프로그램의 전반적인 뼈대는 파이썬으로 만들고, 빠른 실행 속도를 필요로 하는 부분은 C로 만들어서 파이썬 프로그램 안에 포함시키는 것이다(정말 놀라우리만치 영악한 언어가 아닌가). 사실 파이썬 라이브러리²들 중에는 순수 파이썬만으로 제작된 것도 많지만 C로 만들어진 것도 많다. C로 만들어진 것들은 대부분 속도가 빠르다.

파이썬은 간결하다

귀도는 파이썬을 의도적으로 간결하게 만들었다. 만약 펄(Perl)과 같은 프로그래밍 언어가 100 가지 방법으로 하나의 일을 처리할 수 있다면 파이썬은 가장 좋은 방법 1 가지만 이용하는 것을 선호한다. 이 간결함의 철학은 파이썬 문법에도 그대로 적용되어 파이썬 프로그래밍을하는 사람들은 잘 정리되어 있는 소스 코드를 볼 수 있다. 다른 사람이 작업한 소스 코드도 한눈에 들어와 이해하기 쉽기 때문에 공동 작업과 유지 보수가 아주 쉽고 편하다.

다음은 파이썬 프로그램의 예제이다. 이 프로그램 소스 코드를 굳이 이해하려 하지 않아도 된다. 이것을 이해할 수 있다면 여러분은 이미 파이썬에 중독된 사람일 것이다. 그냥 한번 구경해 보도록 하자.

```
# simple.py
```

```
languages = ['python', 'perl', 'c', 'java']
```

```
for lang in languages:
```

```
    if lang in ['python', 'perl']:
```

```
        print("%6s need interpreter" % lang)
```

```
    elif lang in ['c', 'java']:
```

```
        print("%6s need compiler" % lang)
```

```
    else:
```

```
        print("should not reach here")
```

이 예제는 프로그래밍 언어를 판별하여 그에 맞는 문장을 출력하는 파이썬 프로그램 예제이다. 다른 언어들에서 늘 보게 되는 단락을 구분하는 괄호({ }) 문자가 보이지 않는 것을 확인할 수 있다. 또한 줄을 참 잘 맞춘 코드라는 것도 알 수 있다. 파이썬 프로그램은 줄을 맞추지 않으면 실행이 되지 않는다. 코드를 예쁘게 작성하려고 줄을 맞추는 것이 아니라 실행이 되게 하려면 꼭 줄을 맞추어야 하는 것이다. 이렇듯 줄을 맞추어 코드를 작성하는 행위³는 가독성에 크게 도움이 된다.

파이썬은 프로그래밍을 즐기게 해준다

이 부분이 가장 강조하고 싶은 부분이다. 파이썬만큼 필자에게 프로그래밍을 즐기게 해준 언어는 없었다. 파이썬은 다른 것에 신경 쓸 필요 없이 내가 하고자 하는 부분에만 집중할 수 있게 해준다. 파이썬을 배우고 나면 다른 언어로 프로그래밍하는 것에 지루함을 느끼게 될지도 모른다. 조심하자!

파이썬은 개발 속도가 빠르다

마지막으로, 재미있는 다음 문장으로 파이썬의 특징을 마무리하려 한다.

"Life is too short, You need python." (인생은 너무 짧으니 파이썬이 필요해.)

파이썬의 엄청나게 빠른 개발 속도를 두고 유행처럼 퍼진 말이다. 이 위트 있는 문장은 이 책에서 계속 예제로 사용될 것이다.

1. 오픈 소스(Open Source)란 저작권자가 소스 코드를 공개하여 누구나 별다른 제한 없이 자유롭게 사용, 복제, 배포, 수정할 수 있는 소프트웨어이다. [↗](#)
2. 파이썬 라이브러리는 파이썬 프로그램 작성시 불러와 사용할 수 있는 미리 만들어진 파이썬파일들의 모음이다. [↗](#)
3. 이렇게 코드의 줄을 맞추는 것을 "들여쓰기" 라고 부른다. 파이썬에서 들여쓰기를 하지 않으면 프로그램이 실행되지 않는다. [↗](#)

01-3 파이썬으로 무엇을 할 수 있을까?

프로그래밍 언어를 좋은 언어와 나쁜 언어로 구분할 수 있을까? 사실 현실에서 이런 구분은 무의미하다. 어떤 언어든지 강점과 약점이 존재하기 때문이다. 그러므로 어떤 프로그래밍 언어가 어떤 일에 효율적인지를 안다는 것은 프로그래머의 생산성을 크게 높일수 있는 힘이 된다. 그렇다면 파이썬으로 하기에 적당한 일과 적당하지 않은 일은 무엇일까? 이에 대해서 알아보는 것은 매우 가치 있는 일이 될 것이다.

파이썬으로 할 수 있는 일

파이썬으로 할 수 있는 일은 아주 많다. 대부분의 프로그래밍 언어가 하는 일을 파이썬은 쉽고 깔끔하게 처리한다. 파이썬으로 할 수 있는 일들을 나열하자면 끝도 없겠지만 대표적인 몇 가지 예를 들어 보겠다.

시스템 유틸리티 제작

파이썬은 운영체제(윈도우, 리눅스 등)의 시스템 명령어들을 이용할 수 있는 각종 도구를 갖추고 있기 때문에 이를 바탕으로 갖가지 시스템 유틸리티¹를 만드는 데 유리하다. 실제로 여러분은 시스템에서 사용 중인 서로 다른 유틸리티성 프로그램들을 하나로 묶어서 큰 힘을 발휘하게 하는 프로그램들을 무수히 만들어낼 수 있다.

GUI 프로그래밍

GUI(Graphic User Interface) 프로그래밍이란 쉽게 말해 윈도우 창처럼 화면을 보며 마우스나 키보드로 조작할 수 있는 프로그램을 만드는 것이다. 파이썬으로 GUI 프로그램을 만드는 것은 다른 언어를 이용해 만드는 것보다 훨씬 쉽다. 대표적인 예로 파이썬 프로그램을 설치할때 함께 설치되는 기본 모듈인 Tkinter(티케이인터)를 이용해 만드는 GUI 프로그램을 들 수 있다. 실제로 Tkinter 를 이용한 파이썬 GUI 프로그램의 소스 코드는 매우 간단하다. Tkinter 를 이용하면 단 5 줄의 소스 코드만으로도 윈도우 창을 띄울 수 있다. 놀랍지 않은가!

※ 파이썬에는 wxPython, PyQt, PyGTK 등과 같이 Tkinter 보다 빠른 속도와 보기 좋은 인터페이스를 자랑하는 것들도 있다.

C/C++와의 결합

파이썬은 접착(glue) 언어라고도 부르는데, 그 이유는 다른 언어들과 잘 어울려 다른 언어와 결합해서 사용할 수 있기 때문이다. C 나 C++로 만든 프로그램을 파이썬에서 사용할 수 있으며, 파이썬으로 만든 프로그램 역시 C 나 C++에서 사용할 수 있다.

웹 프로그래밍

일반적으로 익스플로러나 크롬, 파이어폭스와 같은 브라우저를 이용해 인터넷을 사용하는데, 누구나 한 번쯤 웹 서핑을 하면서 게시판이나 방명록에 글을 남겨 본 적이 있을 것이다. 그러한 게시판이나 방명록을 바로 웹 프로그램이라고 한다. 파이썬은 웹 프로그램을 만들기엔 매우 적합한 도구이며 실제로 파이썬으로 제작된 웹사이트는 셀 수 없을 정도로 많다.

수치 연산 프로그래밍

사실 파이썬은 수치 연산 프로그래밍에 적합한 언어는 아니다. 수치가 복잡하고 연산이 많다면 C 같은 언어로 하는 것이 더 빠르기 때문이다. 하지만 파이썬에는 Numeric Python이라는 수치 연산 모듈이 제공된다. 이 모듈은 C로 작성되었기 때문에 파이썬에서도 수치 연산을 빠르게 할 수 있다.

데이터베이스 프로그래밍

파이썬은 사이베이스(Sybase), 인포믹스(Infomix), 오라클(Oracle), 마이에스큐엘(MySQL), 포스트그레스큐엘(PostgreSQL) 등의 데이터베이스에 접근할 수 있게 해주는 도구들을 제공한다.

또한 이런 굴직한 데이터베이스를 직접 이용하는 것 외에도 파이썬에는 재미있는 모듈이 하나 더 있다. 바로 피클(pickle)이라는 모듈이다. 피클은 파이썬에서 사용되는 자료들을 변형없이 그대로 파일에 저장하고 불러오는 일들을 맡아 한다. 이 책에서는 외장 함수에서 피클을 어떻게 사용하고 활용하는지에 대해서 알아본다.

데이터 분석, 사물 인터넷

파이썬으로 만들어진 판다스(Pandas)라는 모듈을 이용하면 데이터 분석을 더 쉽고 효과적으로 할 수 있다. 데이터 분석을 할 때 아직까지는 데이터 분석에 특화된 "R"이라는 언어를 많이 사용하고 있지만, 판다스가 등장한 이후로 파이썬을 이용하는 경우가 점점 증가하고 있다. 사물 인터넷 분야에서도 파이썬은 활용도가 높다. 한 예로 라즈베리파이(Raspberry Pi)는 리눅스 기반의 아주 작은 컴퓨터이다. 라즈베리파이를 이용하면 홈시어터나 아주 작은 게임기 등 여러 가지 재미있는 것들을 만들 수 있는데 파이썬은 이 라즈베리파이를 제어하는 도구로 사용된다. 예를 들어 라즈베리파이에 연결된 모터를 작동시키거나 램프에 불이 들어오게 하는 일들을 파이썬으로 할 수 있다.

파이썬으로 할 수 없는 일

시스템과 밀접한 프로그래밍 영역

파이썬으로 도스나 리눅스 같은 운영체제, 엄청난 횟수의 반복과 연산을 필요로 하는 프로그램 또는 데이터 압축 알고리즘 개발 프로그램 등을 만드는 것은 어렵다. 즉, 대단히 빠른 속도를 요구하거나 하드웨어를 직접 건드려야 하는 프로그램에는 어울리지 않는다.

파이썬은 구글이 가장 많이 애용하는 언어이지만 파이썬으로 안드로이드 앱(App)을 개발하는 것은 아직 어렵다. 안드로이드에서 파이썬으로 만든 프로그램들이 실행되도록 지원하긴 하지만 이것만으로 앱을 만들기에는 아직 역부족이다. 아이폰 앱을 개발하는 것 역시 파이썬으로는 할 수 없다.

1. 유틸리티란 컴퓨터 이용에 도움이 되는 여러소프트웨어를 말한다. ↩

01-4 파이썬 설치하기

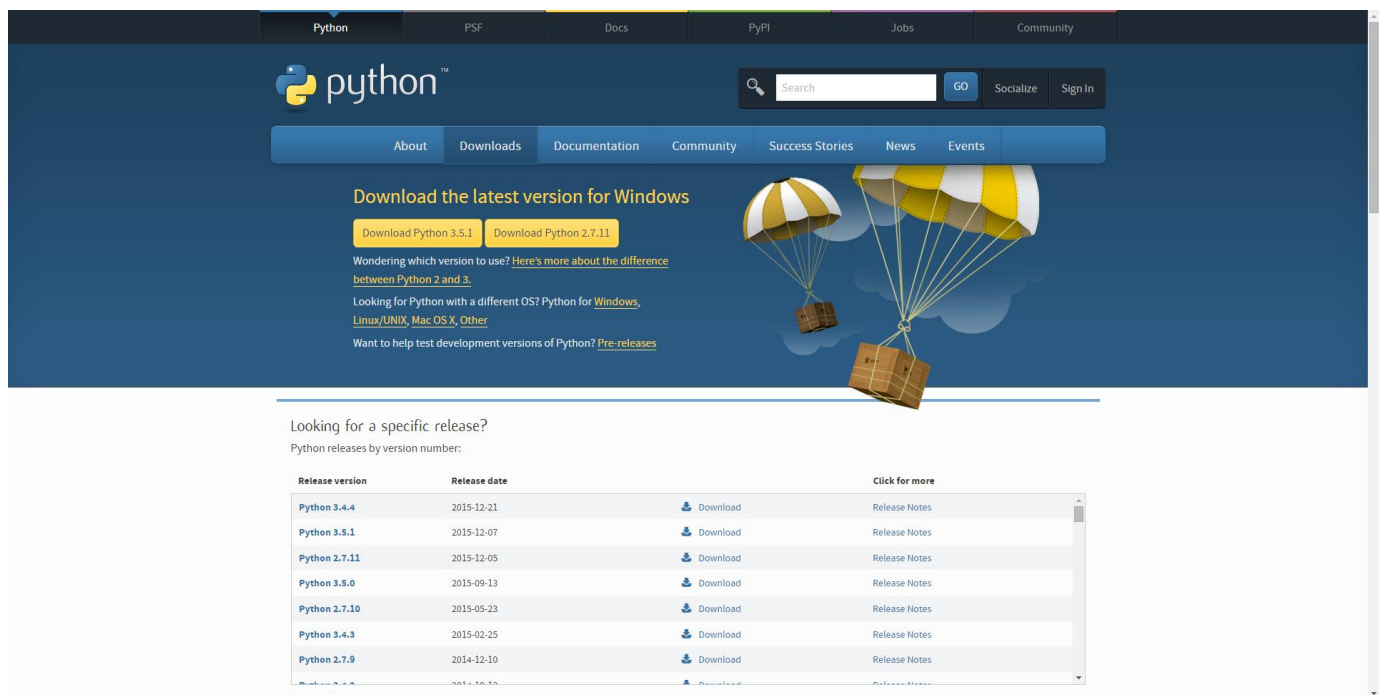
이제 실습을 위해 컴퓨터에 파이썬을 설치해 보자. 이 책에서는 윈도우와 리눅스에서 설치하는 방법만 다룬다. 다른 시스템을 이용할 경우 파이썬 홈페이지(<http://www.python.org>)의 설명을 참고하도록 하자.

윈도우에서 파이썬 설치하기

윈도우의 경우에는 설치가 정말 쉽다.

1. 우선 파이썬 공식 홈페이지의 다운로드 페이지(<http://www.python.org/downloads>)에서 윈도우용 파이썬 언어 패키지를 다운로드한다. 다음 화면에서 Python 3.x 로 시작하는 버전 중 가장 최근의 윈도우 인스톨러를 다운로드하도록 하자(이 글을 작성하는 시점의 최신 버전은 3.5.1 이다.).

※ 만약 파이썬 2.7 버전을 설치할 경우에는 Python 2.7 용 인스톨러 파일을 받아서 설치하면 된다.



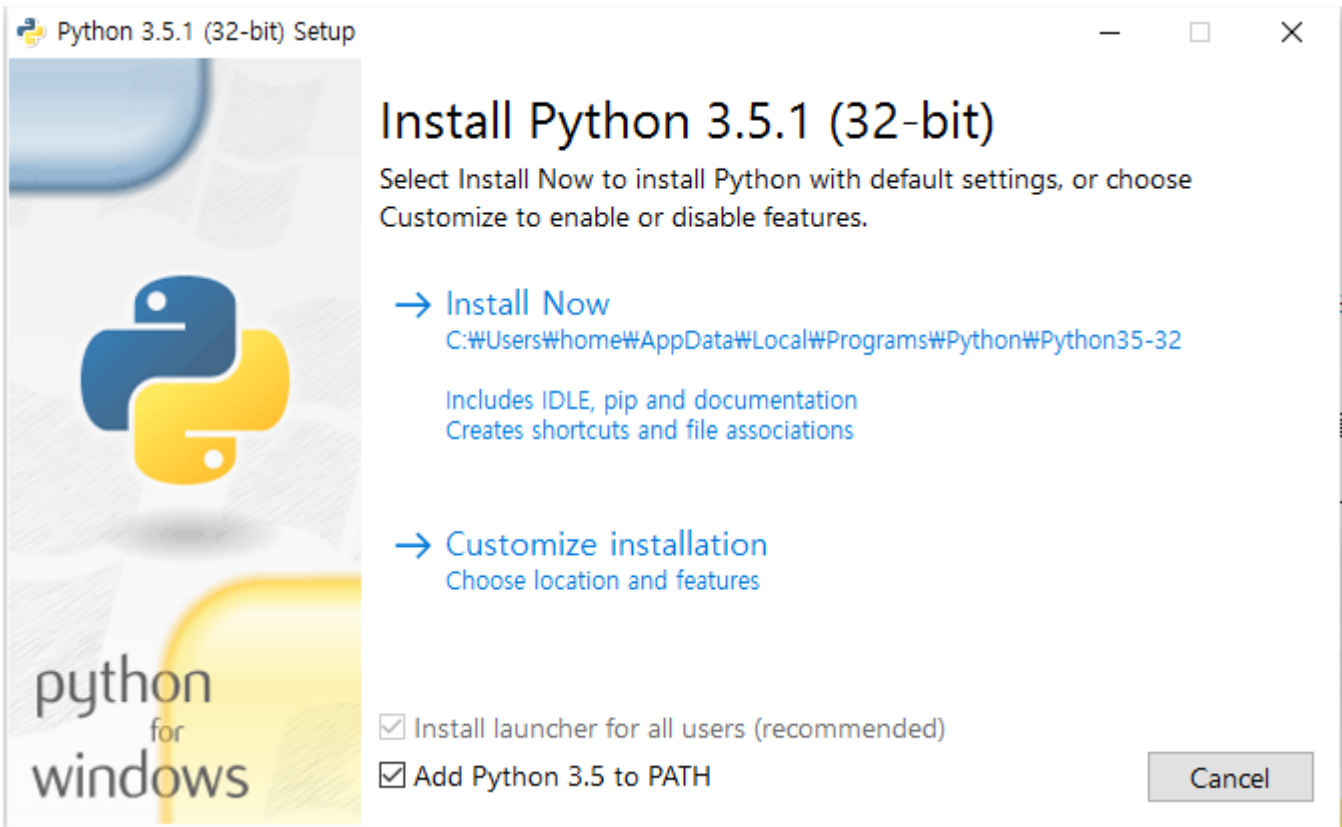
Looking for a specific release?

Python releases by version number:

Release version	Release date	Download	Click for more
Python 3.4.4	2015-12-21	Download	Release Notes
Python 3.5.1	2015-12-07	Download	Release Notes
Python 2.7.11	2015-12-05	Download	Release Notes
Python 3.5.0	2015-09-13	Download	Release Notes
Python 2.7.10	2015-05-23	Download	Release Notes
Python 3.4.3	2015-02-25	Download	Release Notes
Python 2.7.9	2014-12-10	Download	Release Notes

2. 인스톨러를 실행한 후에 "Install Now"를 선택하면 바로 설치가 진행된다.

파이썬이 어느 곳에서든지 실행될 수 있도록 "Add Python 3.5 to PATH" 옵션을 선택하도록 하자.

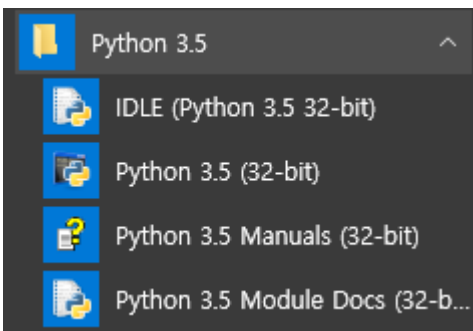


※ "Add Python 3.5 to PATH" 옵션을 누락할 경우 이후 설명되는 예제에서 오류가 발생할 수 있다. 만약 python 이 설치되는 경로와 PATH 에 대한 사전지식이 있는 사용자라면 이 옵션을 생략해도 된다.

3. 설치가 완료되면 [close]를 클릭하여 종료한다.

파이썬이 정상적으로 설치되었다면 오른쪽 그림과 같이 프로그램 메뉴에서 확인할 수 있을 것이다.

[시작 → 모든 프로그램(모든 앱) → Python 3.5]



리눅스에서 파이썬 설치

리눅스 사용자라면 기본적으로 파이썬이 설치되어 있을 것이다.

```
$ python -V
```

리눅스 셸에서 위의 명령어를 입력하면 파이썬 버전을 확인할 수 있다.

만약 파이썬이 기본으로 설치되어 있지 않다면 소스를 컴파일하여 설치한다. 리눅스의 경우 배포본별로 여러 가지 설치 방법이 있을 수 있는데, 소스를 컴파일하여 설치하는 것이 모든 배포본에서 사용할 수 있는 가장 일반적인 방법이다.

파이썬 공식 홈페이지의 다운로드 페이지(<http://www.python.org/download>)에 접속해 "Python-3.X.X.tgz" 를 다운로드한다. 이 책에서는 Python-3.5.1 버전을 사용한다.

먼저 터미널에 다음 문장을 입력하여 다운로드한 파일의 압축을 푼다.

```
$ tar xvzf Python-3.5.1.tgz
```

그런 다음 해당 디렉터리로 이동한다.

```
$ cd Python-3.5.1
```

Makefile 파일을 만들기 위해서 configure 를 실행한다.

```
$ ./configure
```

파이썬 소스를 컴파일한다.

```
$ make
```

루트 계정으로 설치한다.

```
$ su -
```

```
$ make install
```

파이썬 2.7 버전을 설치할 경우 Python-3.5.1.tgz 가 아닌 Python-2.7.tgz 파일을 다운로드해 동일한 방법으로 설치하면 된다.

01-5 파이썬 둘러보기

도대체 파이썬이라는 언어는 어떻게 생겼는지, 간단한 코드를 작성하며 알아보자. 파이썬을 자세하게 탐구하기 전에 전체적인 모습을 죽 훑어보는 것은 매우 유익한 일이 될 것이다.

"백문이 불여일견, 백견이 불여일타"라고 했다. 직접 따라 해보자.

파이썬 기초 실습 준비하기

파이썬 프로그래밍 실습을 시작하기 전에 기초적인 것을 준비해 보자.

[시작] 메뉴에서 [프로그램 → Python 3.X → Python 3.X(XX-bit)]을 선택하면 다음과 같은 화면이 나타난다.

```
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AM...
```

```
Type "help", "copyright", "credits" or "license" for more information.
```

```
>>>
```

위와 같은 것을 대화형 인터프리터라고 하는데, 앞으로 이 책에서는 이 인터프리터로 파이썬 프로그래밍의 기초적인 사항들에 대해 설명할 것이다.

※ 대화형 인터프리터는 파이썬 셸(Python shell)이라고도 한다. 3 개의 꺾은 괄호(>>>)는 프롬프트(prompt)라고 한다.

대화형 인터프리터를 종료할 때는 **Ctrl+Z** 를 누른다 (유닉스 계열에서는 **Ctrl+D**). 또는 다음의 예와 같이 `sys` 모듈을 사용하여 종료할 수도 있다.

```
>>> import sys
```

```
>>> sys.exit()
```

파이썬 기초 문법 따라 해보기

여기서 소개하는 내용들은 나중에 다시 자세하게 다룰 것이니 이해가 되지 않는다고 절망하거나 너무 고심하지 말도록 하자.

파이썬 인터프리터를 실행하여 다음을 직접 입력해 보자.

사칙연산

1 더하기(+) 2 는 3 이라는 값을 출력해 보자. 보통 계산기 사용하듯 더하기 기호만 넣어 주면 된다.

```
>>> 1 + 2
```

```
3
```

나눗셈(/)과 곱셈(*) 역시 예상한 대로 결과값을 보여준다.

```
>>> 3 / 2.4
```

```
1.25
```

```
>>> 3 * 9
```

```
27
```

우리가 일반적으로 알고 있는 ÷ 기호나 × 기호가 아닌 것에 주의하자.

변수에 숫자 대입하고 계산하기

```
>>> a = 1
```

```
>>> b = 2
```

```
>>> a + b
```

```
3
```

a 에 1 을, b 에 2 를 대입한 다음 a 와 b 를 더하면 3 이라는 결과값을 보여 준다.

변수에 문자 대입하고 출력하기

```
>>> a = "Python"
```

```
>>> print(a)
```

```
Python
```

a 라는 변수에 Python 이라는 값을 대입한 다음 print(a) 라고 작성하면 a 의 값을 출력한다.

※ 파이썬은 대소문자를 구분한다. print 를 PRINT 로 쓰면 정의되지 않았다는 에러 메시지가 나온다.

[변수에 복소수도 넣을 수 있을까?]

파이썬은 복소수도 지원한다

```
>>> a = 2 + 3j
```

```
>>> b = 3
```

```
>>> a * b
```

```
(6+9j)
```

변수 a 에 $2+3j$ 라는 값을 대입하고, 변수 b 에 3 을 대입하였다. 여기서 $2+3j$ 란 복소수를 의미한다. 보통 우리는 고등학교 때 복소수를 표시할 때 알파벳 i 를 이용해서 $2 + 3i$ 처럼 사용했지만 파이썬에서는 j 를 사용한다. 위의 예는 $2 + 3j$ 와 3 을 곱하는 방법이다. 당연히 결과값으로 $6+9j$ 를 출력한다.

조건문 if

다음은 간단한 조건문 if 를 이용한 예제이다.

```
>>> a = 3
```

```
>>> if a > 1:
```

```
...     print("a is greater than 1")
```

```
...
```

```
a is greater than 1
```

※ print 문 앞의 '...'은 아직 문장이 끝나지 않았음을 의미한다.

위 예제는 a 가 1 보다 크면 "a is greater than 1"이라는 문장을 출력(print)하라는 뜻이다. 위 예제에서 a 는 3 이므로 1 보다 크다. 따라서 두 번째 "..." 이후에 Enter 키를 입력하면 if 문이 종료되고 "a is greater than 1"이라는 문장이 출력된다.

if a > 1: 다음 문장은 Tab 키 또는 Spacebar 키 4 개를 이용해 반드시 들여쓰기 한 후에 print("a is greater than 1")이라고 작성해야 한다. 들여쓰기 규칙에 대해서는 05 장 제어문에서 자세하게 알아볼 것이다. 바로 뒤에 이어지는 반복문 for, while 예제도 마찬가지로 들여쓰기가 필요하다.

반복문 for

다음은 for 를 이용해서 [1, 2, 3]안의 값들을 하나씩 출력해 주는 것을 보여주는 예이다.

```
>>> for a in [1, 2, 3]:
```

```
...     print(a)
```

```
...
1
2
3
```

for 문을 이용하면 실행해야 할 문장을 여러 번 반복해서 실행시킬 수 있다. 위의 예는 대괄호([]) 사이에 있는 값들을 하나씩 출력한다. 위 코드의 의미는 "[1, 2, 3]이라는 리스트의 앞에서부터 하나씩 꺼내어 a 라는 변수에 대입한 후 `print(a)`를 수행하라"이다. 당연히 a 에 차례로 1, 2, 3 이라는 값이 대입되며 `print(a)`에 의해서 그 값이 차례대로 출력된다.

반복문 while

다음은 **while** 을 이용하는 예이다.

```
>>> i = 0
>>> while i < 3:
...     i=i+1
...     print(i)
...
1
2
3
```

while 이라는 영어 단어는 "~인 동안"이란 뜻이다. for 문과 마찬가지로 반복해서 문장을 수행할 수 있도록 해준다. 위의 예제는 i 값이 3 보다 작은 동안 `i=i+1` 과 `print(i)`를 수행하라는 말이다. `i=i+1` 이라는 문장은 i 의 값을 1 씩 더하게 한다. i 값이 3 보다 커지게 되면 while 문을 빠져나가게 된다.

함수

파이썬의 함수는 다음과 같은 형태이다.

```
>>> def sum(a, b):
```

```
... return a+b
...
>>> print(sum(3,4))
7
```

파이썬에서 **def** 는 함수를 만들 때 사용하는 예약어이다. 위의 예제는 sum 이라는 함수를 만들고 그 함수를 어떻게 사용하는지를 보여준다. sum(a, b)에서 a, b 는 입력값이고, a+b 는 결과값이다. 즉 3, 4 가 입력으로 들어오면 3+4 를 수행하고 그 결과값인 7 을 돌려 준다.

이렇게 해서 기초적인 파이썬 문법에 대해서 간략하게 알아보았다.

파이썬 프로그램을 작성할 수 있는 여러 가지 에디터

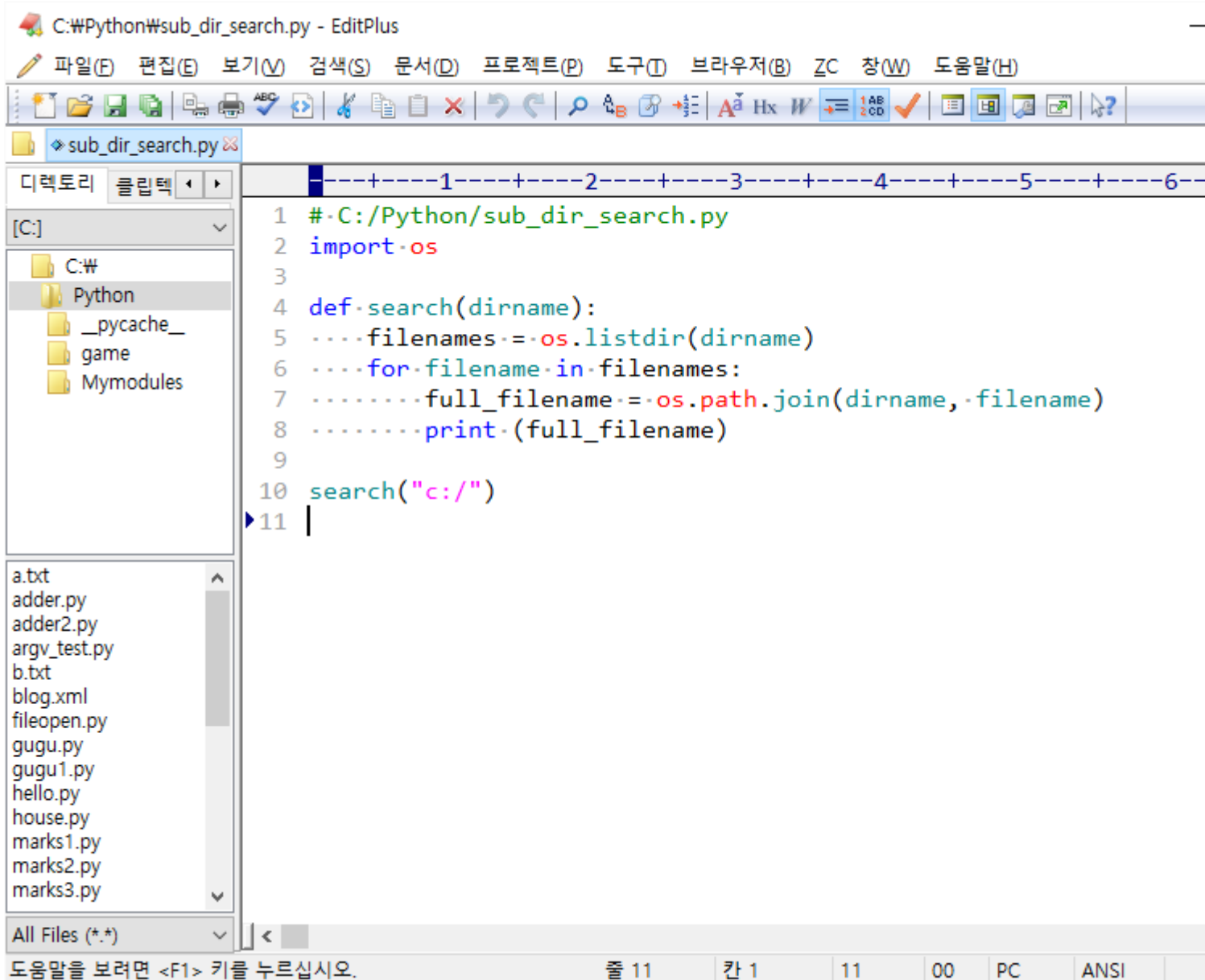
이미 눈치챘을지 모르지만 파이썬 프로그램을 편하게 작성하기 위해서는 인터프리터보다는 에디터를 이용해서 작성하는 것이 좋다. 에디터란 문서를 편집할 수 있는 프로그래밍 툴을 말한다.

에디터는 자신에게 익숙한 것을 사용하면 되는데, 아직 즐겨 사용하는 에디터가 마땅히 없는 독자에게 몇 가지 추천하고 싶은 에디터가 있다. 윈도우 사용자라면 에디트 플러스나 파이참(PyCharm), 노트패드++ 또는 서브라임 텍스트 3, 리눅스 사용자라면 당연히 vi 에디터를 추천한다. 물론 리눅스에는 이맥스라는 좋은 에디터가 있지만 초보자가 다루기는 쉽지 않다.

다음에 여러 가지 에디터를 소개해 두었으니 읽어 보고 자신에게 맞는 에디터를 선택하자. 에디터를 선택하기 어렵다면 파이썬 프로그래밍을 처음 시작하기에 좋은 에디트 플러스를 추천한다.

에디트 플러스

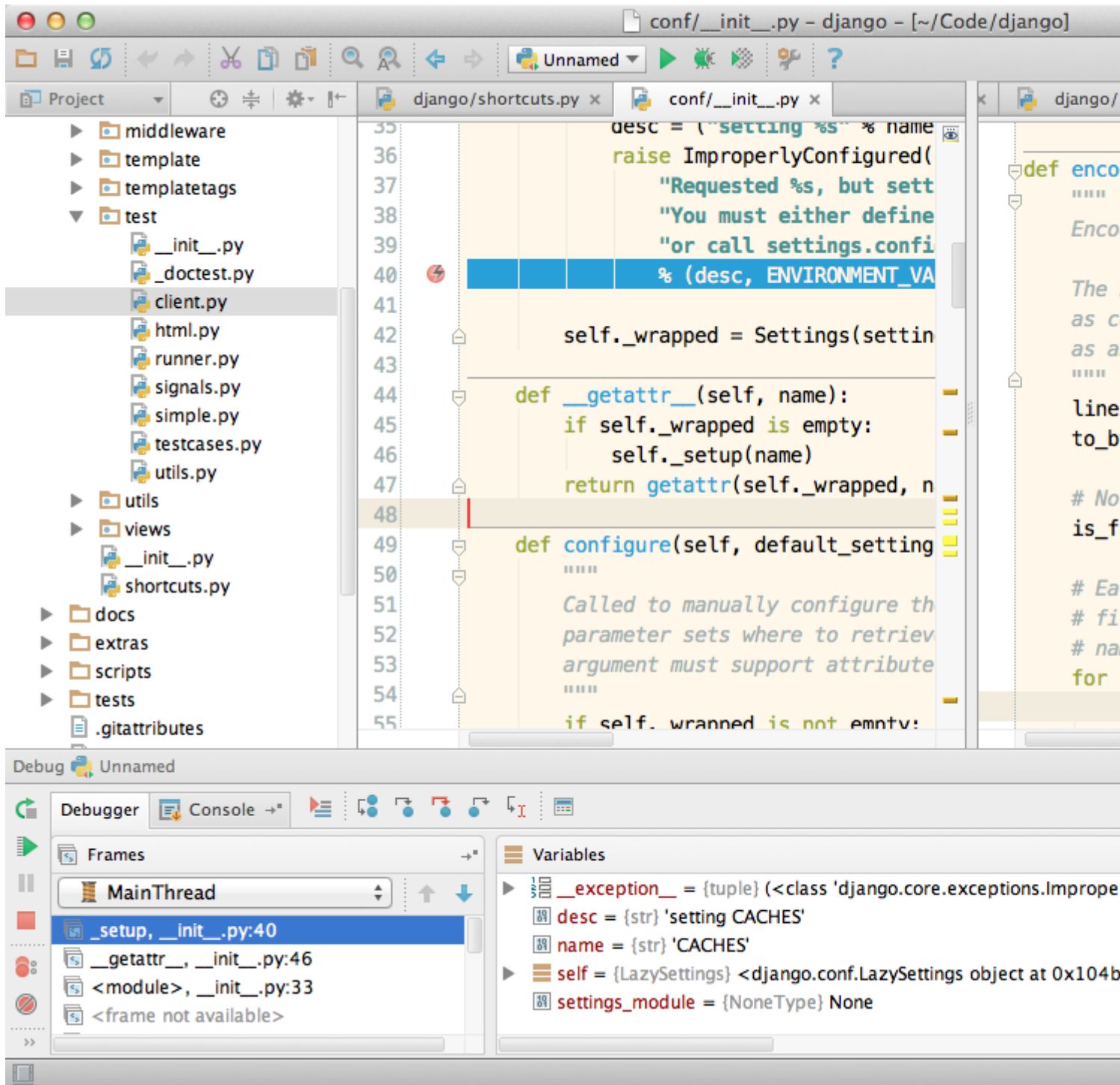
이 프로그램은 무료 소프트웨어가 아니기 때문에 평가판을 이용해야 한다. 다운로드 한 후부터 한 달간 사용할 수 있다. 에디트 플러스 공식 사이트 (<http://www.editplus.com/kr>) 에서 파일을 다운로드해 설치하자.



파이참

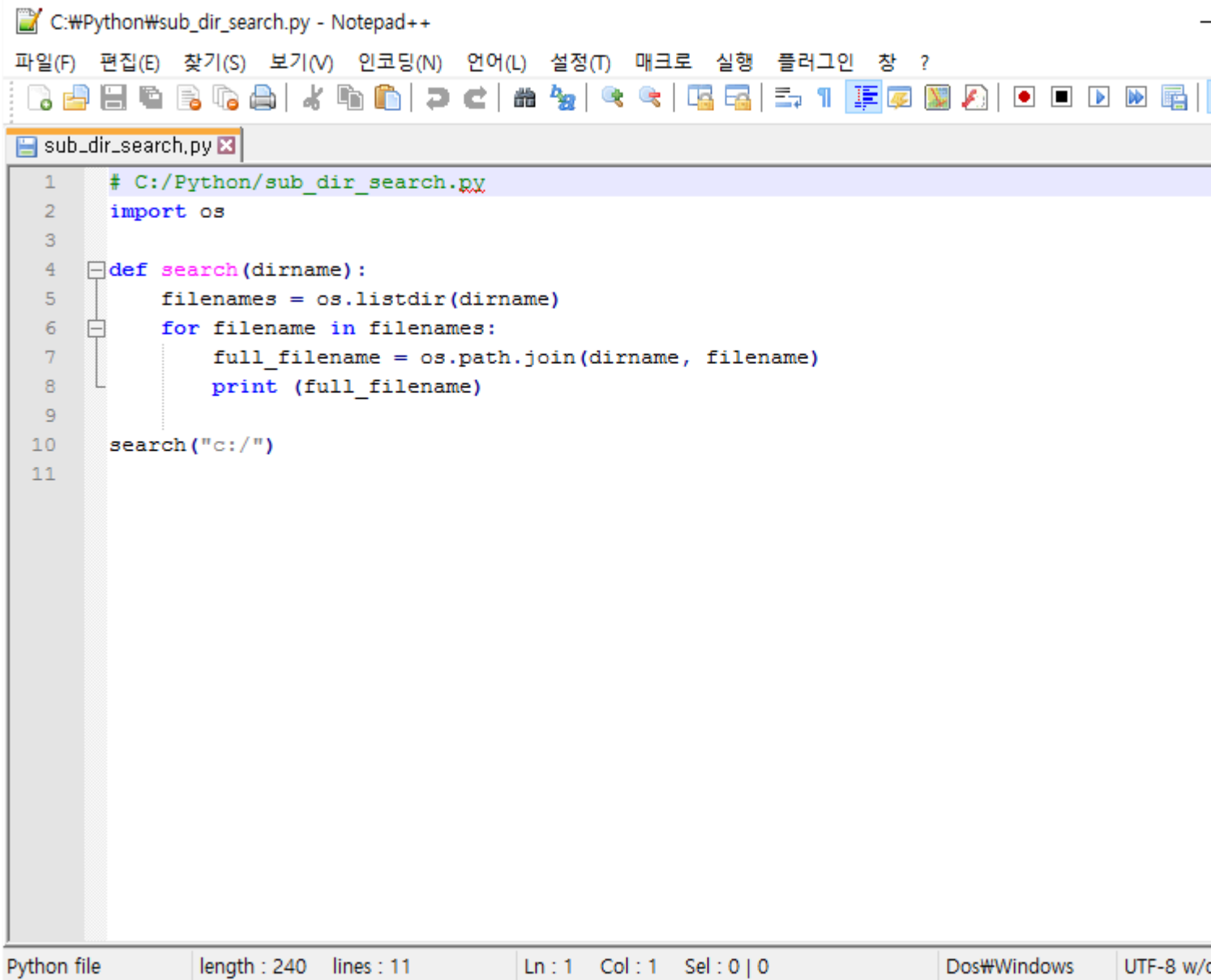
파이썬에 어느 정도 익숙해졌다면 파이참(PyCharm)을 사용해 보기를 적극 추천한다. 파이참은 가장 유명한 파이썬 에디터 중 하나로 코드 작성 시 자동 완성, 문법 체크 등 편리한 기능들을 많이 제공한다.

이 에디터는 파이참 공식 다운로드 사이트 (<http://www.jetbrains.com/pycharm/download>) 에서 다운로드할 수 있다.



노트패드++

노트패드++도 많은 사람들이 추천하는 윈도우용 파이썬 에디터 중의 하나이다. 이 에디터는 노트패드++ 공식 다운로드 사이트 (<https://notepad-plus-plus.org>) 에서 다운로드할 수 있다.



The screenshot shows a Notepad++ window titled "C:\Python\sub_dir_search.py - Notepad++". The menu bar includes "파일(F)", "편집(E)", "찾기(S)", "보기(V)", "인코딩(N)", "언어(L)", "설정(T)", "매크로", "실행", "플러그인", and "창 ?". The toolbar contains various icons for file operations and editing. The active tab is "sub_dir_search.py". The code is as follows:

```
1  # C:/Python/sub_dir_search.py
2  import os
3
4  def search(dirname):
5      filenames = os.listdir(dirname)
6      for filename in filenames:
7          full_filename = os.path.join(dirname, filename)
8          print (full_filename)
9
10 search("c:/")
11
```

The status bar at the bottom displays: "Python file", "length : 240", "lines : 11", "Ln : 1", "Col : 1", "Sel : 0 | 0", "Dos#Windows", and "UTF-8 w/o BOM".

서브라임 텍스트 3

서브라임 텍스트 3 역시 파이썬 사용자에게 사랑받는 에디터 중의 하나로 심플하면서 세련된 사용자 인터페이스를 자랑한다. 이 에디터는 서브라임 텍스트 3 공식 다운로드 사이트 (<http://www.sublimetext.com/3>) 에서 다운로드할 수 있다.



The screenshot shows a Sublime Text editor window titled "C:\Python\sub_dir_search.py - Sublime Text (UNREGISTERED)". The menu bar includes File, Edit, Selection, Find, View, Goto, Tools, Project, Preferences, and Help. The editor has a tab for "sub_dir_search.py". The code is as follows:

```
1 # C:/Python/sub_dir_search.py
2 import os
3
4 def search(dirname):
5     filenames = os.listdir(dirname)
6     for filename in filenames:
7         full_filename = os.path.join(dirname, filename)
8         print (full_filename)
9
10 search("c:/")
11
```

The status bar at the bottom indicates "Line 11, Column 1".

에디터로 파이썬 프로그램 작성하기

다음과 같은 프로그램을 에디터로 직접 작성해 보자.

```
# hello.py

print("Hello world")
```

위의 파일에서 `# hello.py` 라는 문장은 주석이다. `#`으로 시작하는 문장은 `#`부터 시작해서 그 줄 끝까지 프로그램 수행에 전혀 영향을 주지 않는다. 주석은 프로그래머를 위한 것으로, 프로그램 소스에 설명문을 달 때 사용한다.

[여러줄짜리 주석문]

주석문이 여러 줄인 경우 다음의 방법을 사용하면 편리하다.

```
"""
Author: EungYong Park

Date : 2016-01-01

이 프로그램은 Hello World 를 출력하는 프로그램이다.
"""
```

여러 줄로 이루어진 주석을 작성하려면 큰따옴표 세 개를 연속으로 사용한 `"""` 기호 사이에 주석문을 작성하면 된다. 큰따옴표 대신 작은따옴표 세 개를(`'''`)를 사용해도 된다.

앞에서와 같이 작성한 파일을 `hello.py` 라는 이름으로 `C:\Python` 디렉터리에 저장하자. 에디터로 파이썬 프로그램을 작성한 후 저장할 때는 파일 이름의 확장자명을 항상 `py` 로 해야한다. `py` 는 파이썬 파일임을 알려주는 관례적인 확장자명이다.

이제 이 `hello.py` 라는 프로그램을 실행시키기 위해 [`윈도우+R` -> `cmd` 입력 -> `Enter`]를 눌러 도스 창을 연다. `hello.py` 라는 파일이 저장된 곳으로 이동한 후 다음과 같이 입력한다. 이 책에서는 `hello.py` 파일을 `C:\Python` 디렉터리에 저장했다.

```
C:\Users\Whome>cd C:\Python
C:\Python>python hello.py
Hello World
```

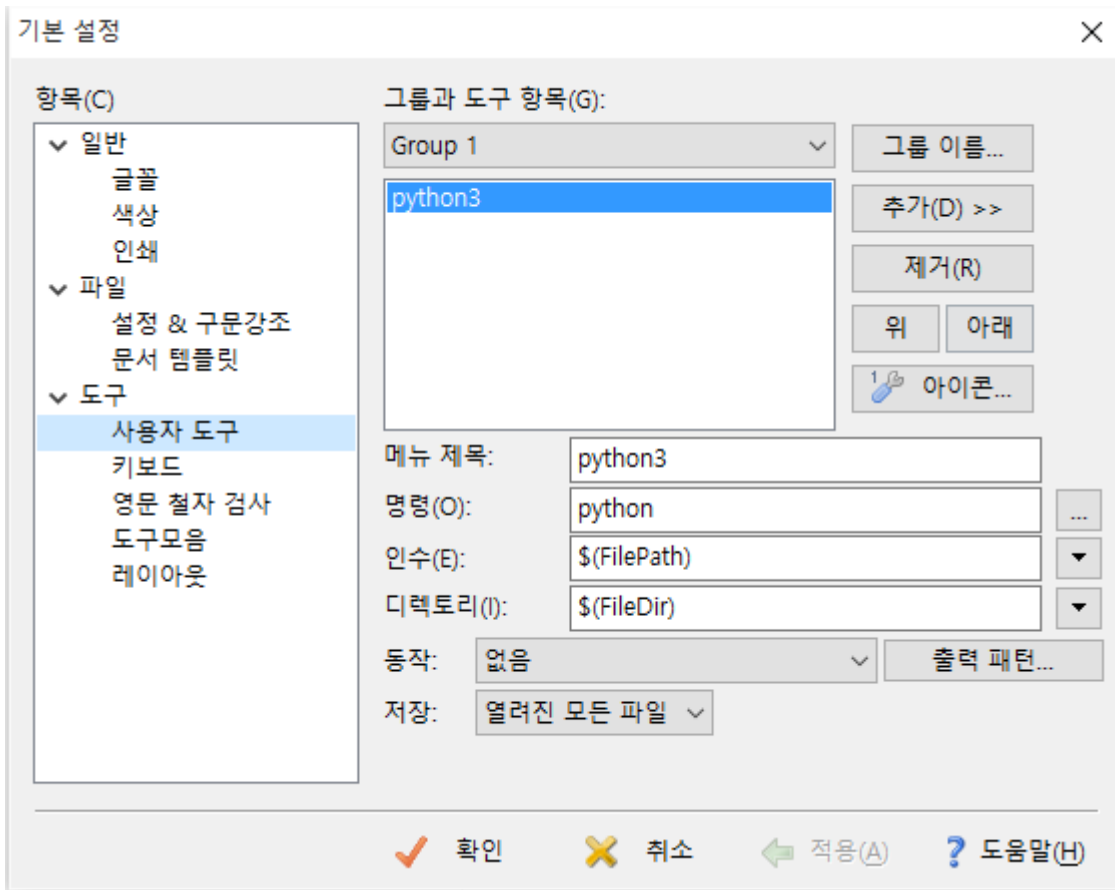
위와 같은 결과값을 볼 수 있을 것이다. 결과값이 위와 같지 않다면 `hello.py` 파일이 `C:\Python` 디렉터리에 존재하는지 다시 한 번 살펴보도록 하자.

이번 예제에서는 Hello world 라는 문장을 출력하는 단순한 프로그램을 에디터로 작성했지만 보통 에디터로 작성하는 프로그램은 꽤 여러 줄로 이루어진다. 여기서 중요한 사실은 에디터로 만든 프로그램은 파일로 존재한다는 점이다. 대화형 인터프리터에서 만든 프로그램은 인터프리터를 종료함과 동시에 사라지지만 에디터로 만든 프로그램은 파일로 존재하기 때문에 언제든지 다시 사용할 수 있다.

왜 대부분이 에디터를 이용해서 파이썬 프로그램을 작성하는지 이제 이해가 될 것이다.

[에디트 플러스로 파이썬 프로그램 쉽게 실행하기]

에디트 플러스를 실행한 후 메뉴바에서 [`도구` → `기본 설정`]를 선택한 다음 [`도구` → `사용자 도구`]를 선택하면 다음과 같은 화면이 나타난다.



위 화면처럼 내용이 채워지도록 다음과 같이 수정하자.

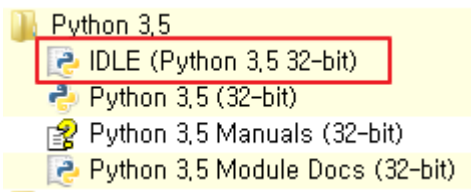
1. [추가 → 프로그램]을 선택한다.
 2. 메뉴 제목 : `python3` 라고 입력한다.
 3. 명령 : `python` 이라고 입력한다.
 4. 인수 : 오른쪽 버튼을 누르고 첫 번째 항목인 "파일 경로"를 선택한다. 입력창에 `$(FilePath)`가 자동으로 입력된다.
 5. 디렉토리 : 오른쪽 버튼을 누르고 첫 번째 항목인 "파일 디렉토리"를 선택한다. 입력창에 `$(FileDir)`이 자동으로 입력된다.
 6. 하단의 [적용] 버튼을 클릭하여 설정을 저장한다.
- 이제 에디트 플러스로 `hello.py` 와 같은 프로그램을 작성하고 저장한 후 `Ctrl+1` 을 누르면 `hello.py` 프로그램이 자동으로 실행되는 것을 확인할 수 있다.

파이썬 IDLE

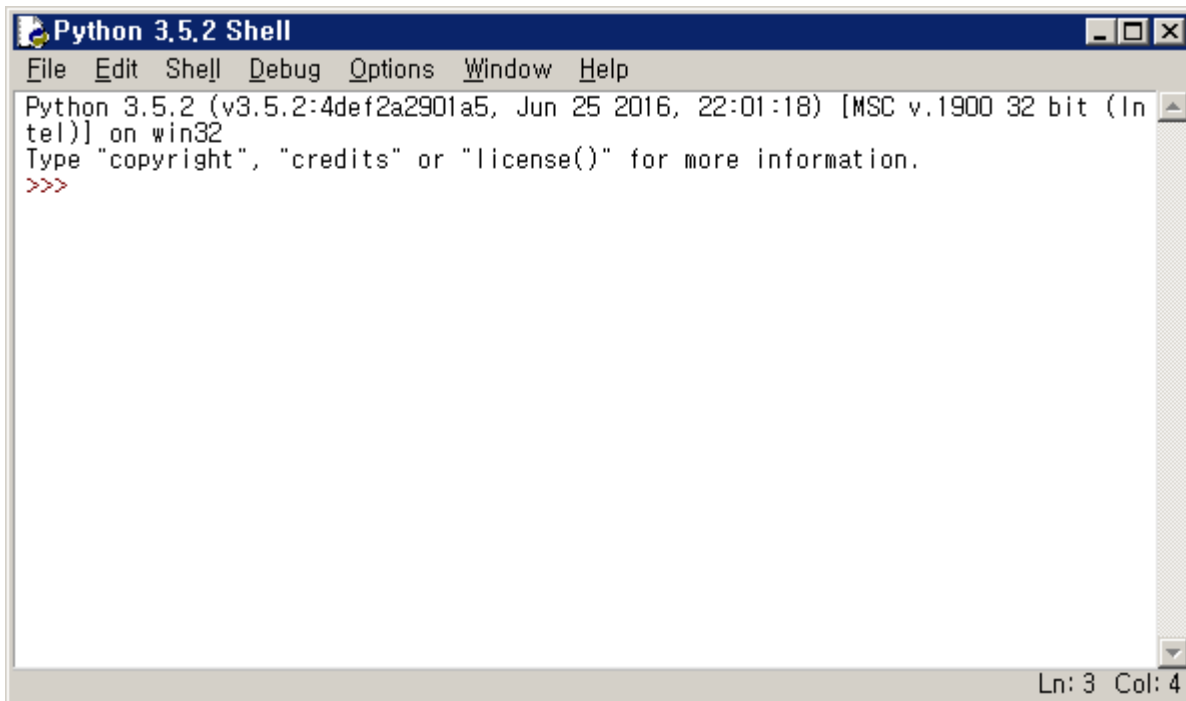
파이썬 IDLE(Integrated Development and Learning Environment)는 파이썬 프로그램 작성을 도와주는 통합 개발환경으로 파이썬 설치시 기본으로 설치되는 프로그램이다. IDLE 를 가지고 전문적인 파이썬 프로그램을 만들기에는 좀 부족하지만 파이썬 공부가 목적이라면 더할 나위없이 좋은 도구가 될 것이다.

파이썬 IDLE 를 실행 해 보자.

[시작 -> 모든 프로그램 -> Python 3.5 -> IDLE 선택]



그러면 다음과 같은 IDLE 셸(Shell) 창이 나타난다.



IDLE 는 크게 두가지 창으로 구성된다.

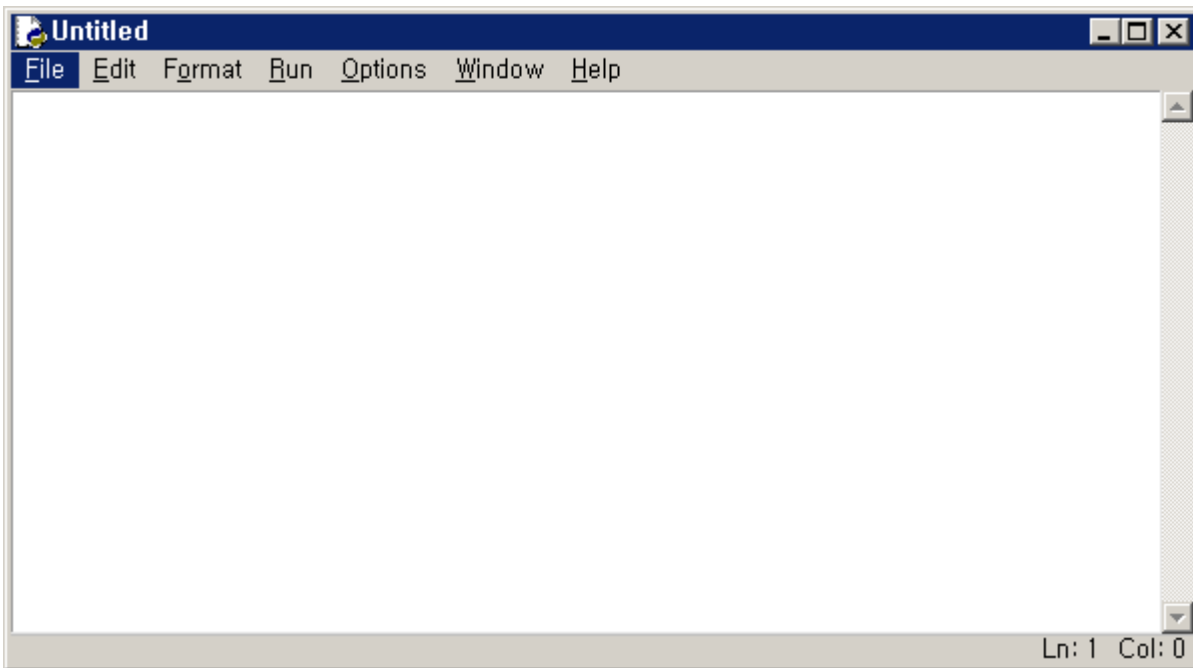
- 셸 창(Shell Window) - 파이썬 셸(Python Shell)이 실행되는 창
- 에디터 창(Editor Window) - 파이썬 에디터(Editor)가 실행되는 창

IDLE 실행 시 가장 먼저 나타나는 창은 셸 창이다. 이 곳에서 파이썬 명령들을 수행하고 테스트 해 볼 수 있다.

이번에는 IDLE 에디터(Editor)를 실행 해 보자.

셸 창 메뉴에서 [File -> New File]을 선택하자.

다음 그림과 같은 IDLE 에디터가 나타날 것이다.



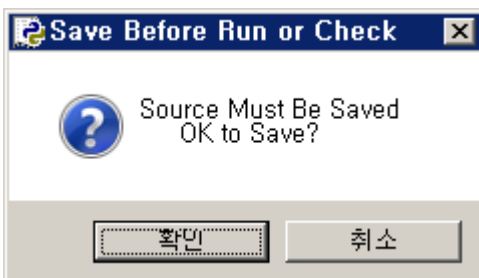
이제 다음과 같은 프로그램을 IDLE 에디터에서 직접 작성해 보자.

```
# hello_idle.py  
  
print("Hello IDLE")
```

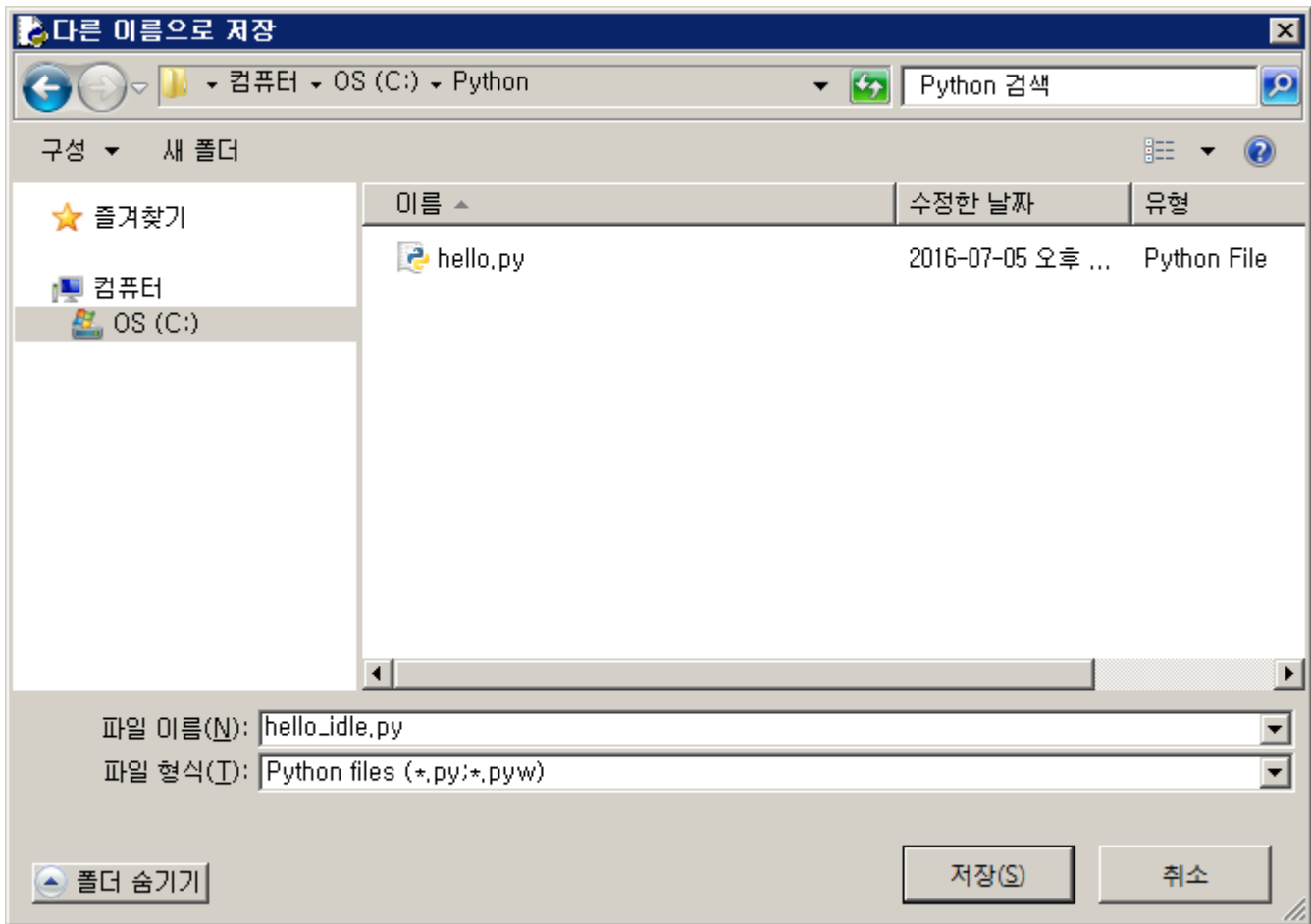
이제 작성한 프로그램을 실행 해 보자.

메뉴에서 [Run -> Run Module]을 선택하자. (단축키: F5)

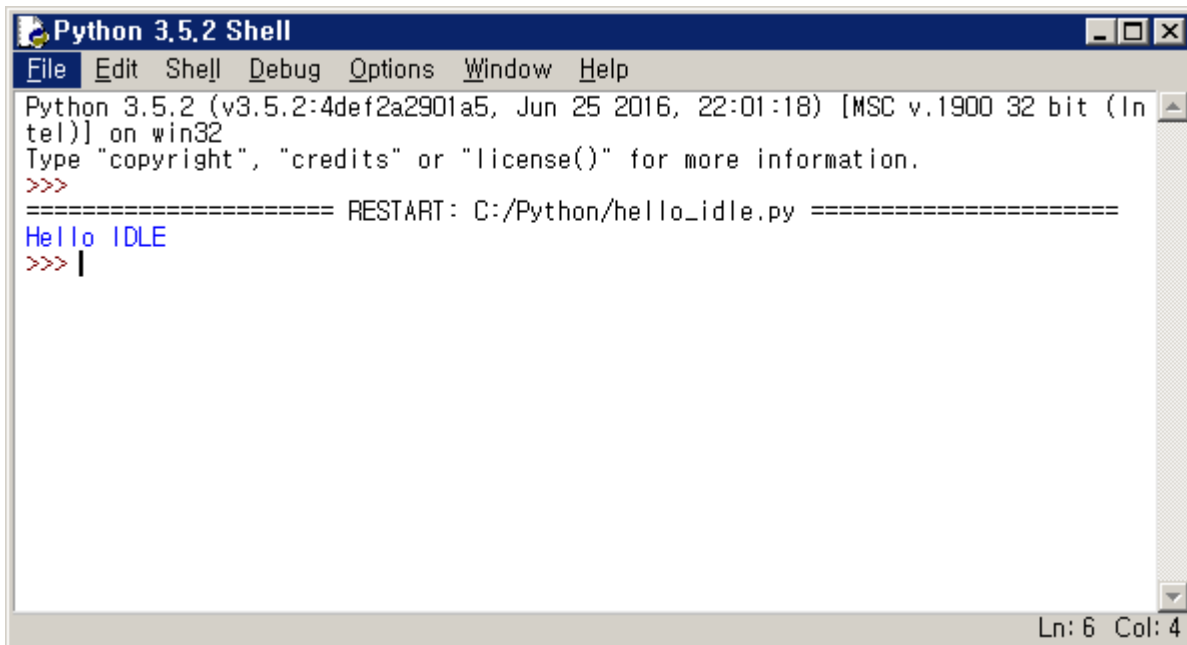
실행 해 보면 파일을 먼저 저장하라는 다음과 같은 다이얼로그가 나올것이다.



"확인"을 선택하고 `C:\Python` 이라는 디렉터리에 `hello_idle.py` 라는 이름으로 저장을 하도록 하자.



파일을 저장하면 자동으로 파이썬 프로그램이 실행 된다. 실행 결과는 다음과 같이 IDLE 셸 창에 표시된다.



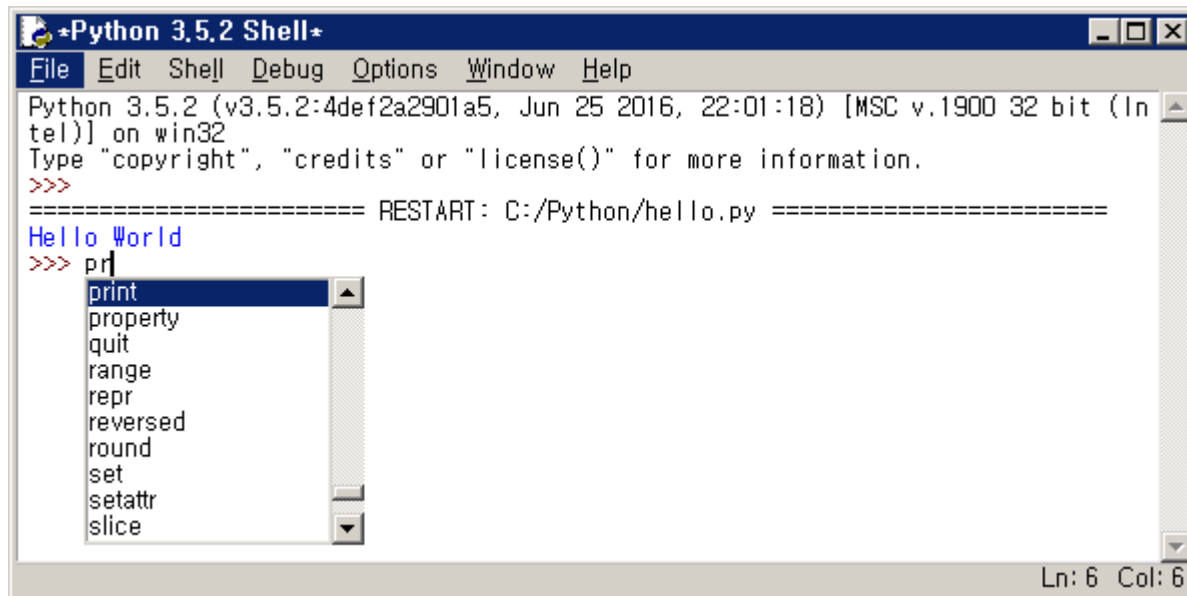
프로그램을 재 실행하려면 에디터 창에서 F5 키로 다시한번 실행하면 된다.

IDLE 를 사용하면 자동완성(auto completion) 기능을 사용할 수 있다.

IDLE 셸 창이나 에디터 창에서 다음과 같이 입력 해 보자.

pr + [Tab]

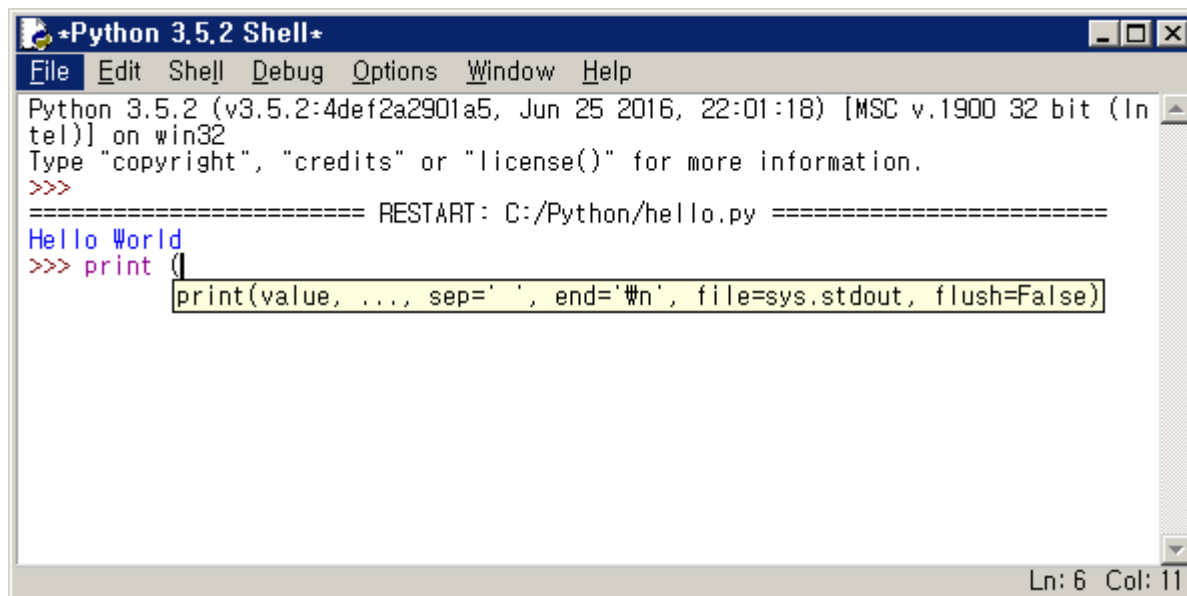
pr 로 시작하는 파이썬 키워드가 콤보박스 가장 상단에 표시될 것이다.



print 를 선택하고 (선택할 때는 스페이스바로 선택한다.) 다음과 같이 왼쪽 괄호만 입력 해 보자.

```
print(
```

그러면 다음과 같이 print 라는 함수의 도움말이 표시(call tips)되는 것을 볼 수 있다.



파이썬 IDLE 는 이 외에도 자동 들여쓰기, 코드를 컬러로 표시하기 등의 여러 유용한 기능들을 제공한다. 나머지 IDLE 의 메뉴나 기타 기능들에 대해서는 IDLE 를 사용하면서 차츰 익혀나가도록 하자.