

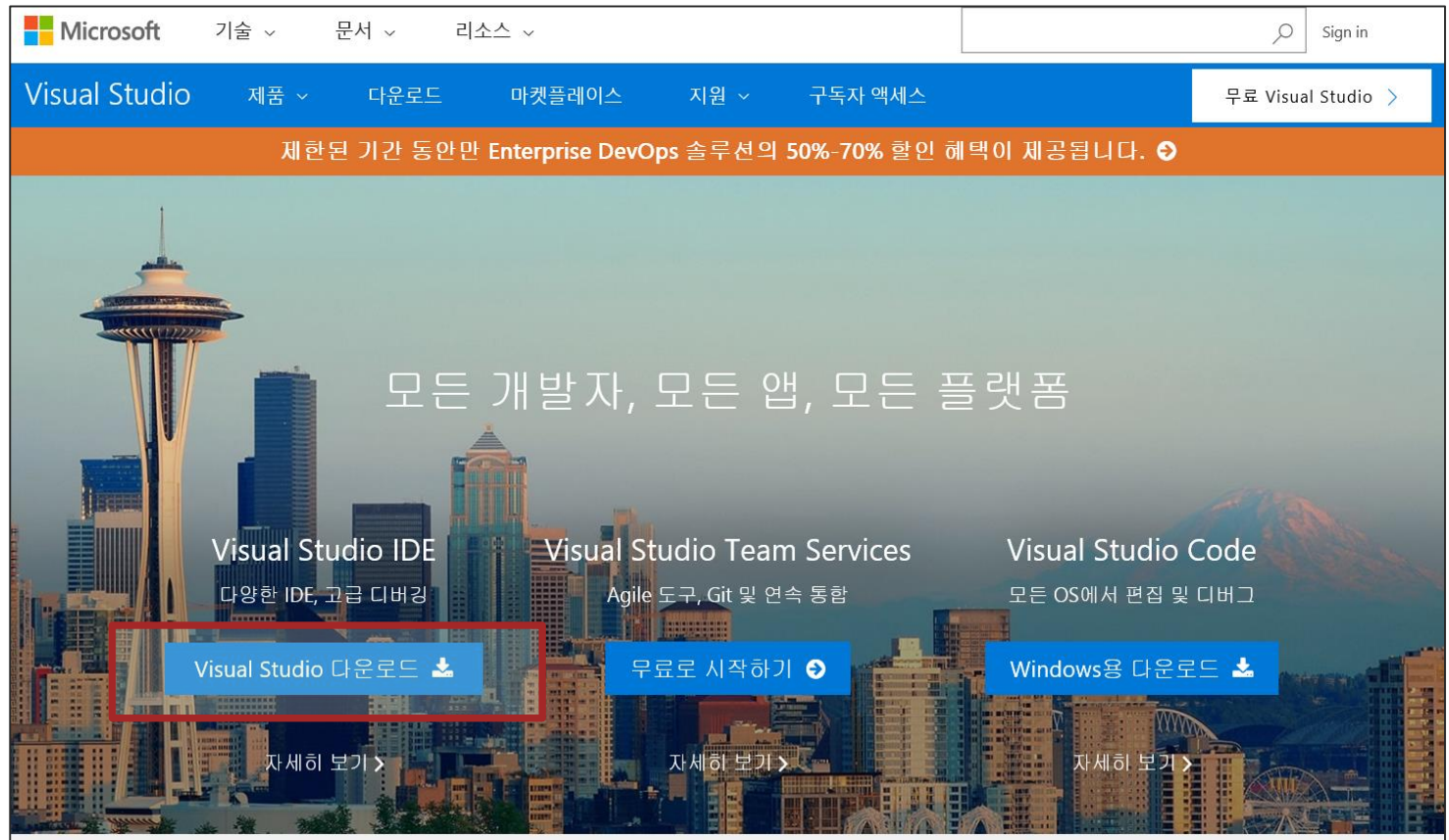


Visual Studio + TensorFlow 실습 환경 구성

박기태
(sina2010@naver.com)

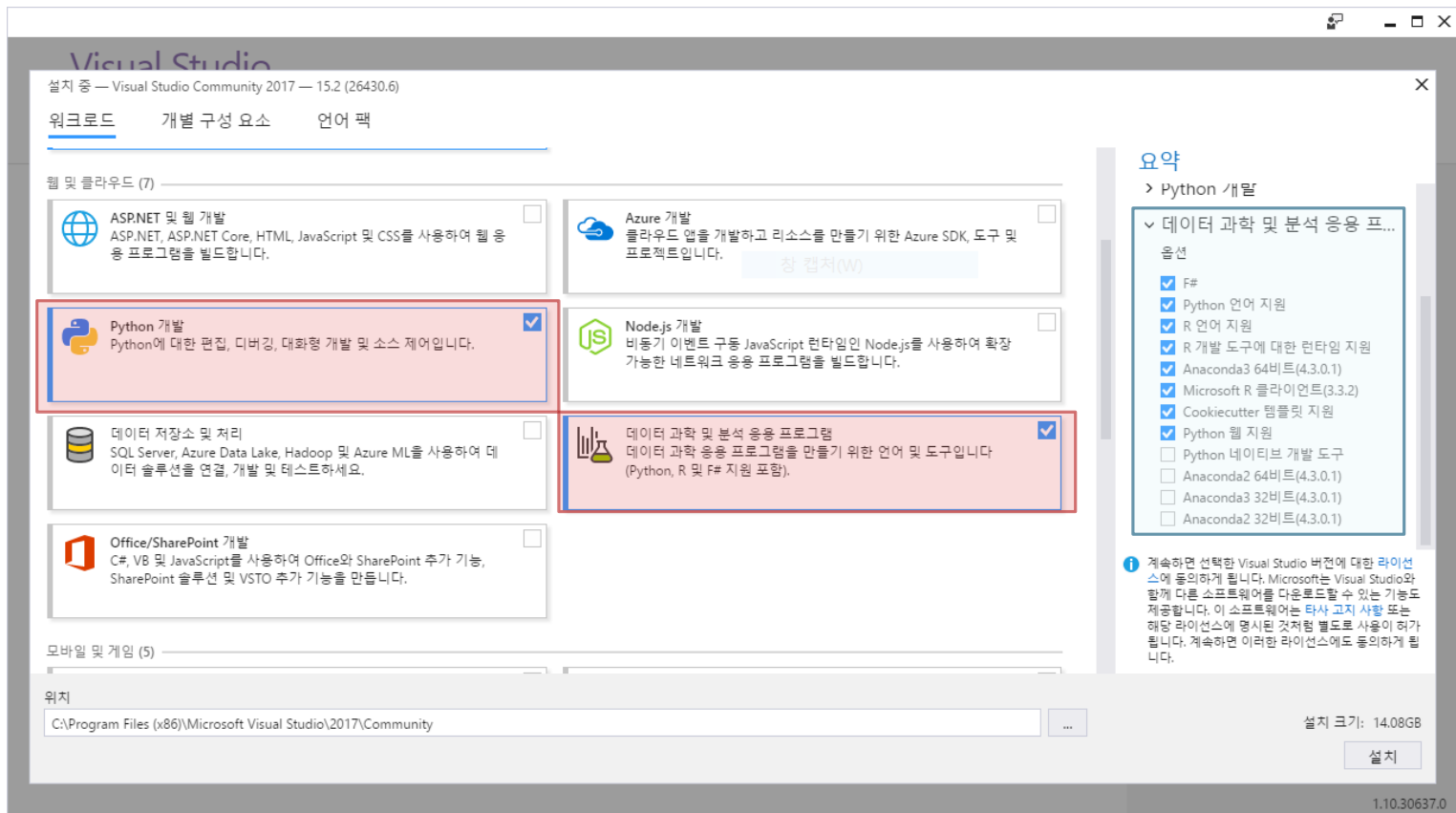
Visual Studio 2017 Download

1. Visual Studio 2017 설치를 위해
<https://www.visualstudio.com/ko/> 로 접속하여 설치 파일을 다운 받아 실행 합니다.
(Community 2017)



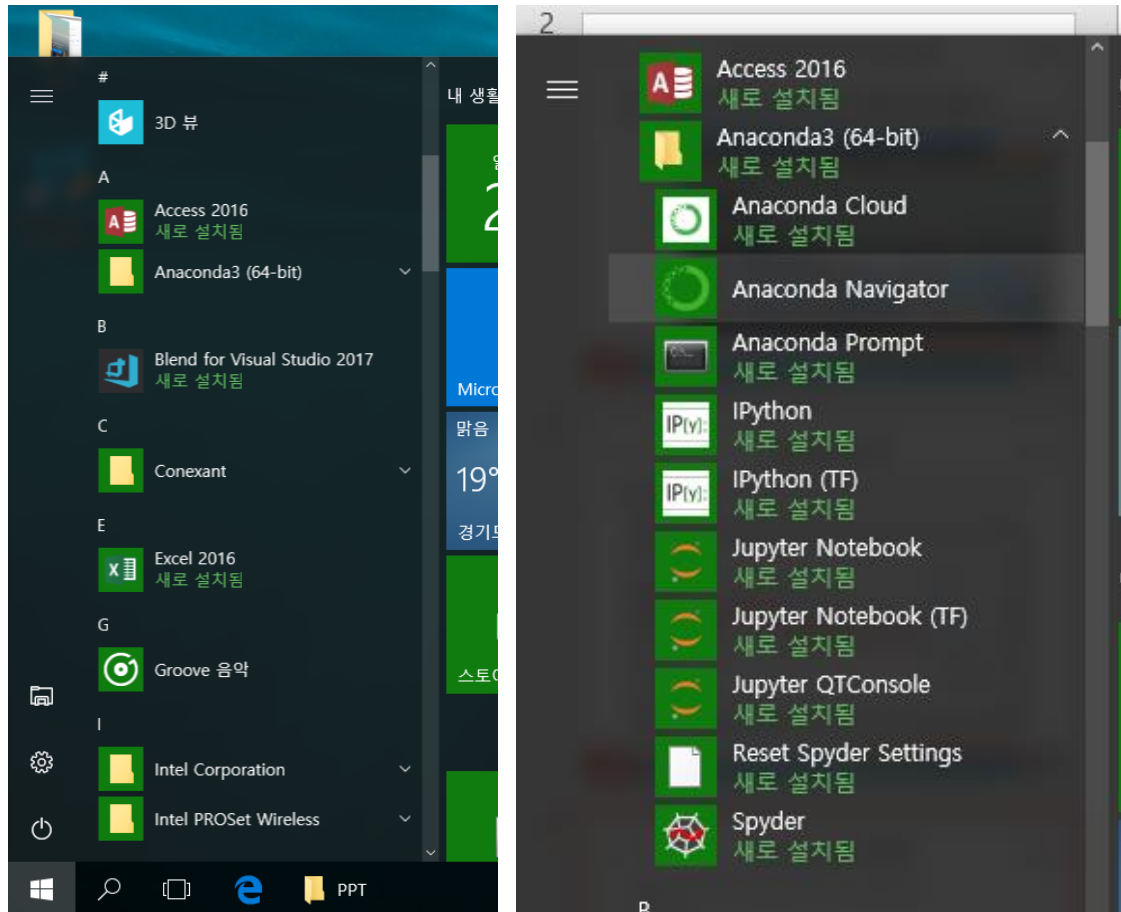
설치 항목 선택

2. 여러 설치 항목 중 python 및 anaconda3를 포함하여 설치하기 위해 아래 두 항목을 선택합니다.



Anaconda3 설치 확인

3. Visual Studio가 설치되면 Anaconda3가 설치 된 것을 확인 할 수 있습니다.
4. Visual Studio에서 사용할 환경을 구성하기 위해 "Anaconda Navigator"를 실행합니다



새 Environment 추가.

5. Navigator가 실행되면, [Environments] → [Create] 버튼을 클릭하여, Python 3.5를 기준으로 새로운 환경을 생성합니다.
(tensorflow 설치시 3.6이 지원되지 않아, 3.5로 생성합니다.)

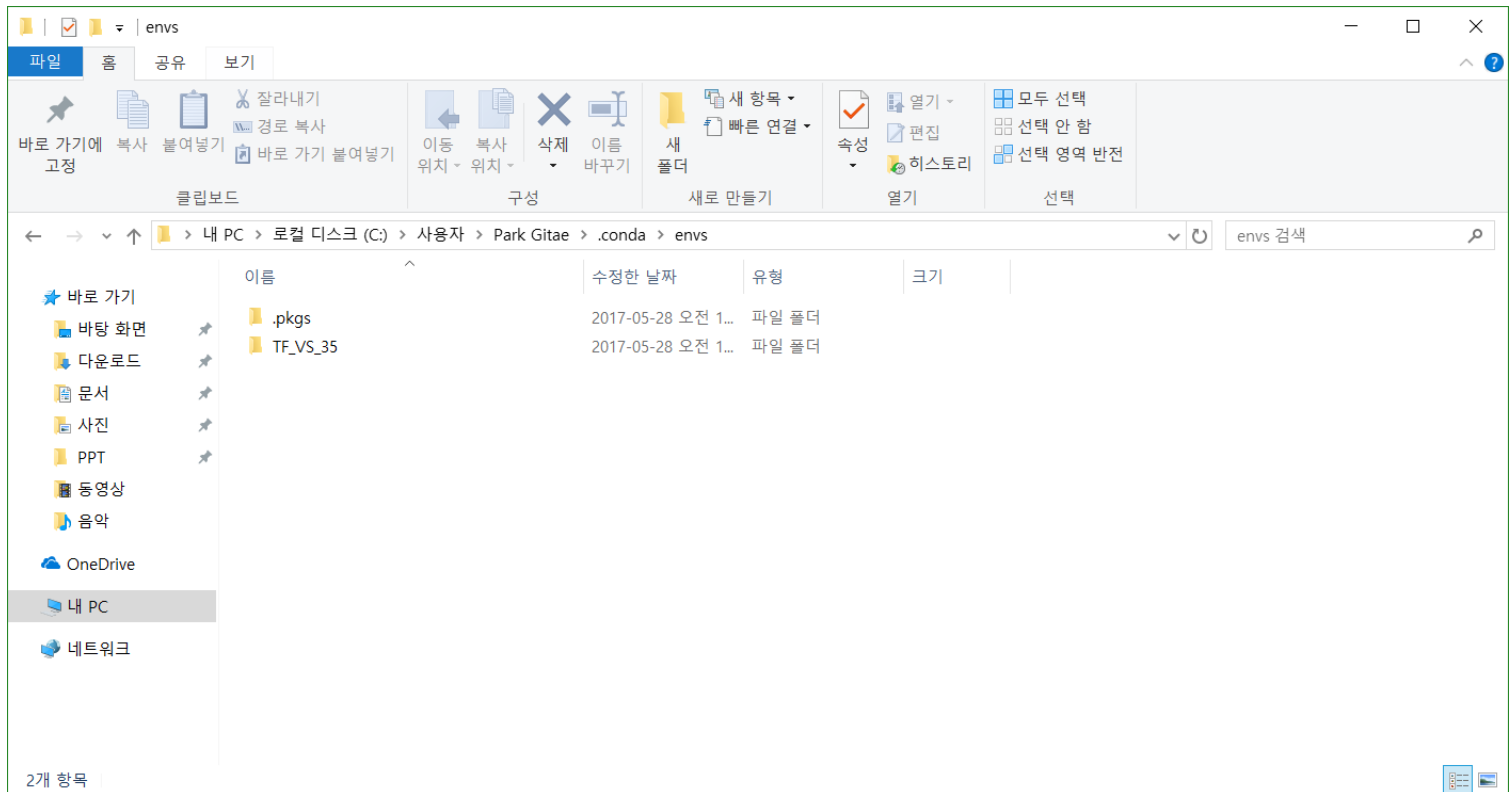
The screenshot shows the Anaconda Navigator interface. On the left sidebar, the 'Environments' tab is selected, indicated by a red circle with the number 1. The main panel displays a list of installed packages. A 'Create new environment' dialog box is open in the center, showing the 'Environment name' field with the text 'TF_VS_35' and the 'Python version' dropdown set to '3.5'. The 'Python' checkbox is checked. The 'Create' button is highlighted with a red circle and the number 3. A larger, zoomed-in view of the dialog box is shown on the right, with the 'Create' button also highlighted with a red circle and the number 3.

Name	T	Description	Version
✓ _license	○		1.1
✓ alabaster	○	Configurable, python 2+3 compatible sphinx theme	0.7.9
✓ anaconda	○		4.3.0
✓ anaconda-client	○	Anaconda.org command line client library	1.6.0
✓ astroid	○	Abstract syntax tree	1.4.9
✓ astropy	○	Community	1.3
✓ babel	○	Utilities to	2.3.4
✓ backports	○		
✓ beautifulsoup4	○	Python libra	
✓ bitarray	○		
✓ blaze	○	Numpy and pandas interface to big data	
✓ bokeh	○	Python interactive visualization library for modern web browsers	
✓ boto	○	Amazon web services library	
✓ bottleneck	○	Fast numpy array functions written in cython.	
✓ bzip2	○	High-quality data compressor	
✓ cffi	○	C foreign function interface for python	
✓ chardet	○	Universal character encoding detector	
✓ chest	○		
✓ click	○	Command line interface creation kit	

167 packages available (root)

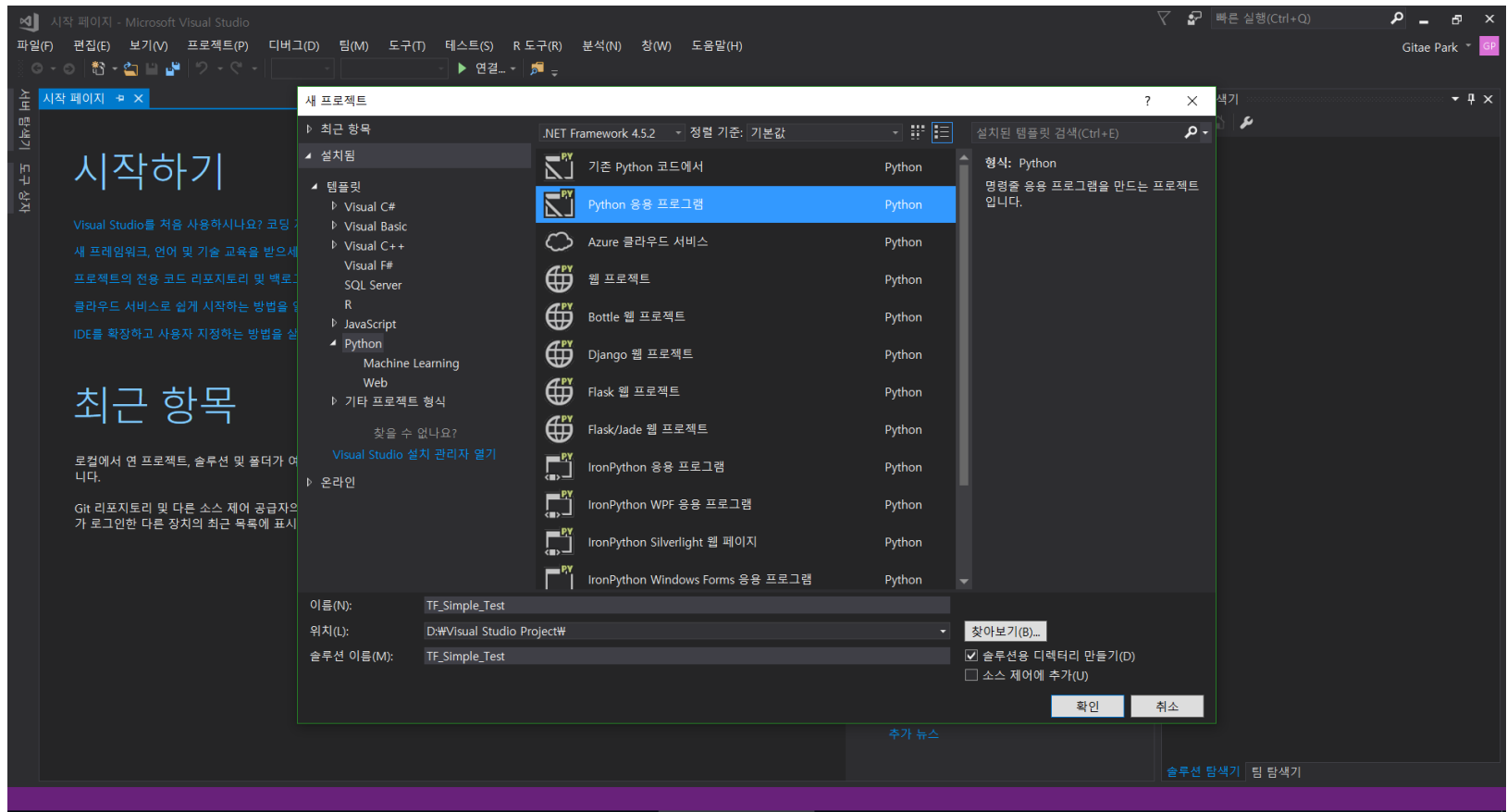
추가된 Environment 위치

6. 방금 전 추가한 anaconda의 환경은 다음 위치에 추가 되는 것 같습니다. (윈도우 10 기준)
[C:\Users\사용자명\conda\envs]
➔ Visual Studio에서 환경 구성시 디렉토리 위치 정보를 추가해 주어야 하기 때문에 어디에 있는지 알아야 합니다..



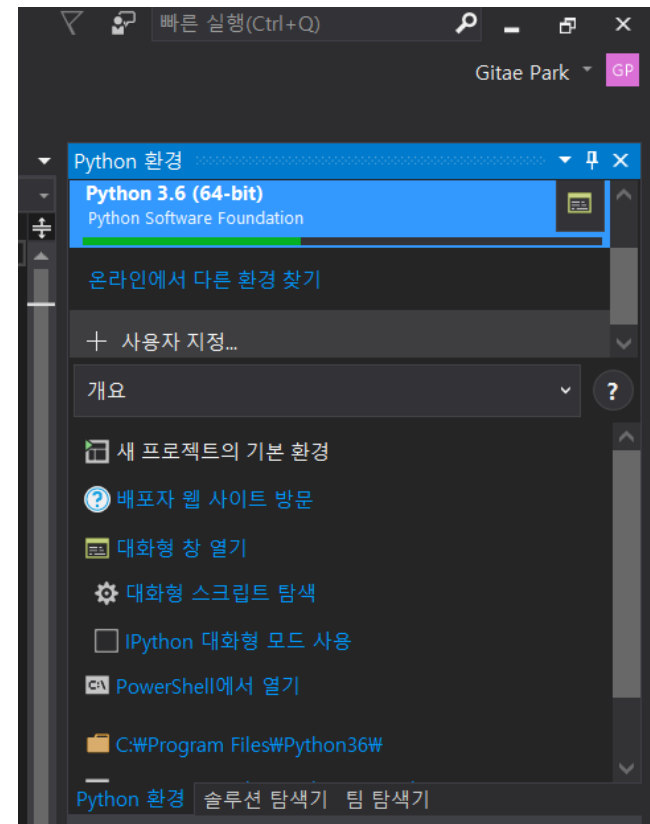
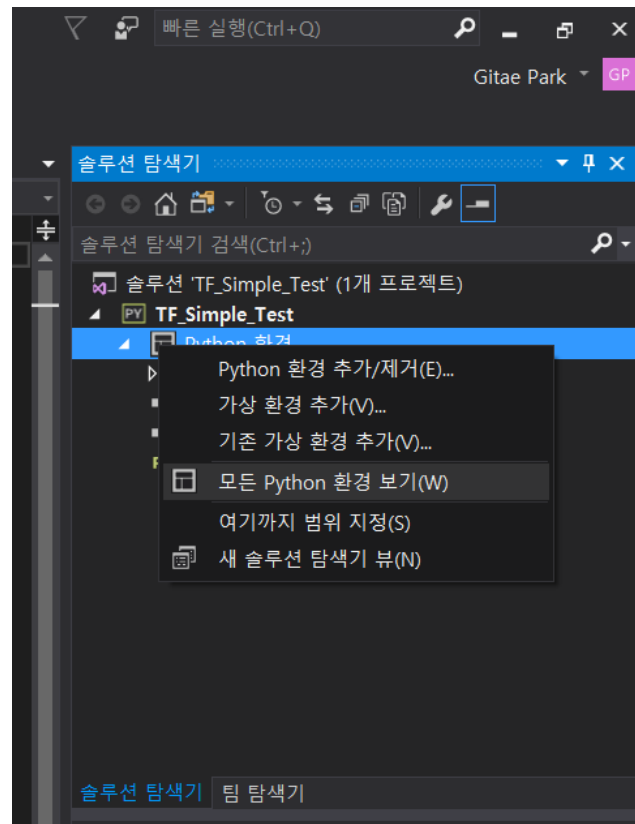
프로젝트 생성

7. 이제 Visual Studio를 실행 하고, 새로운 python 프로젝트를 생성합니다.
이때 [Python 응용 프로그램] 으로 프로젝트 유형을 선택합니다.
(다른 프로젝트 유형도 상관없는데는 잘 모르겠습니다.)

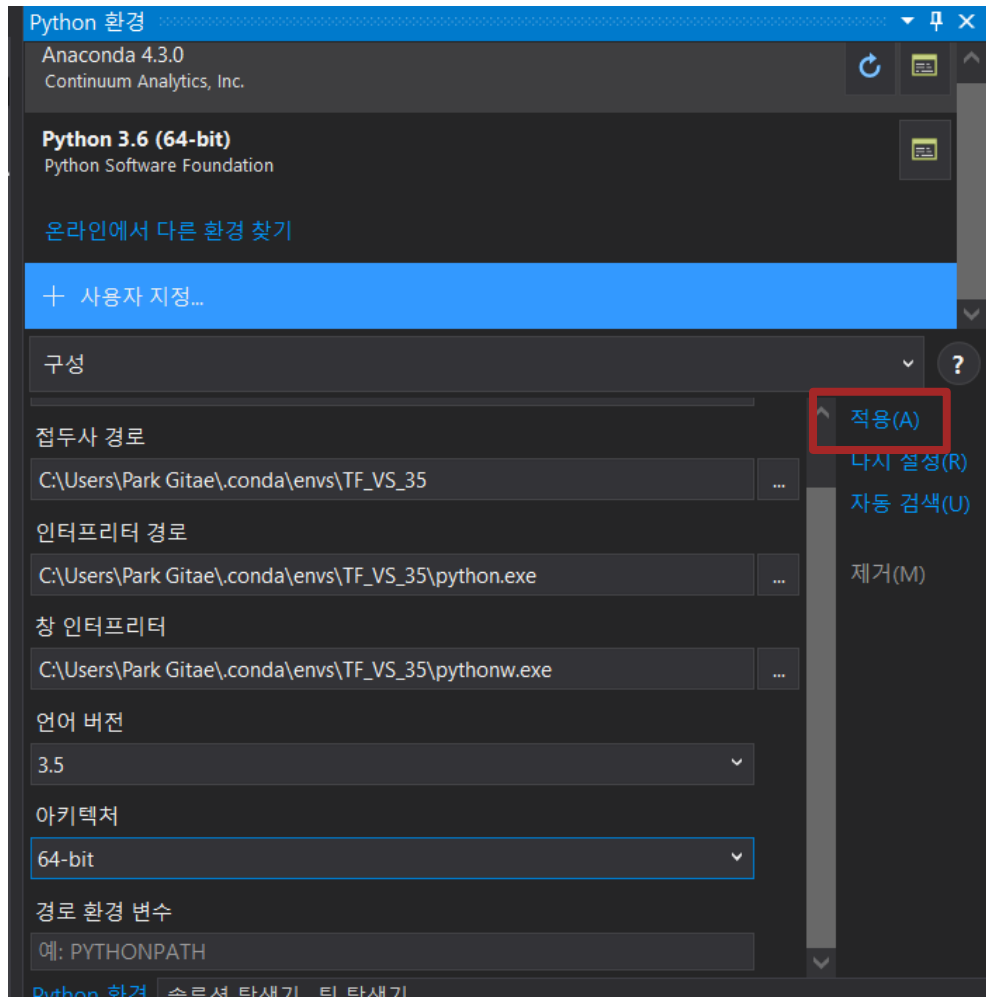


새 가상환경 추가

- 프로젝트가 생성되면, 아래 왼쪽 그림과 같이, 솔루션 탐색기에서, [Python 환경]에 마우스 오른쪽 버튼을 클릭하여, **[모든 Python 환경 보기]**를 클릭합니다.
- 그 후 **[+ 사용자 지정...]**을 클릭하여 방금 전 Anaconda Navigator에서 추가했던 가상환경을 Visual Studio에 추가합니다.



새 가상환경 추가

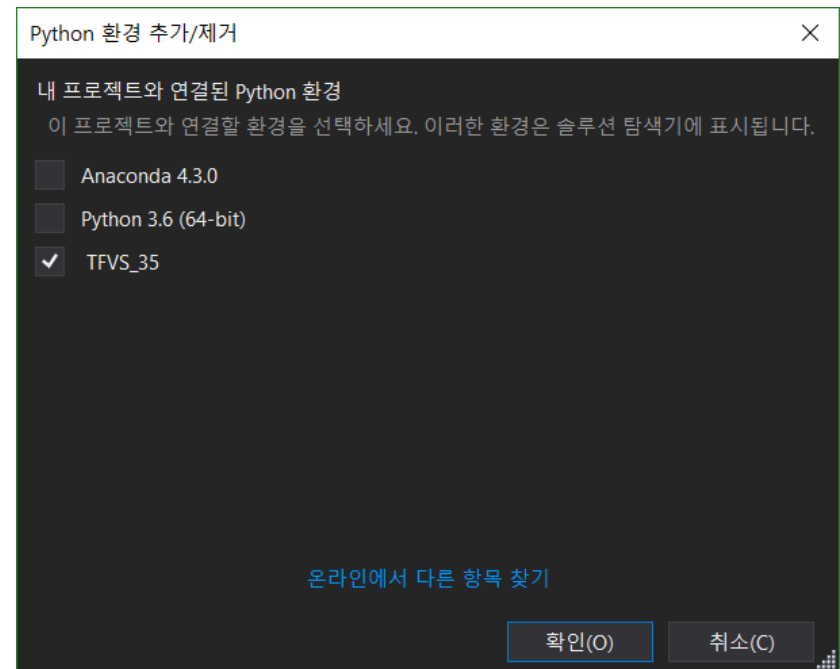
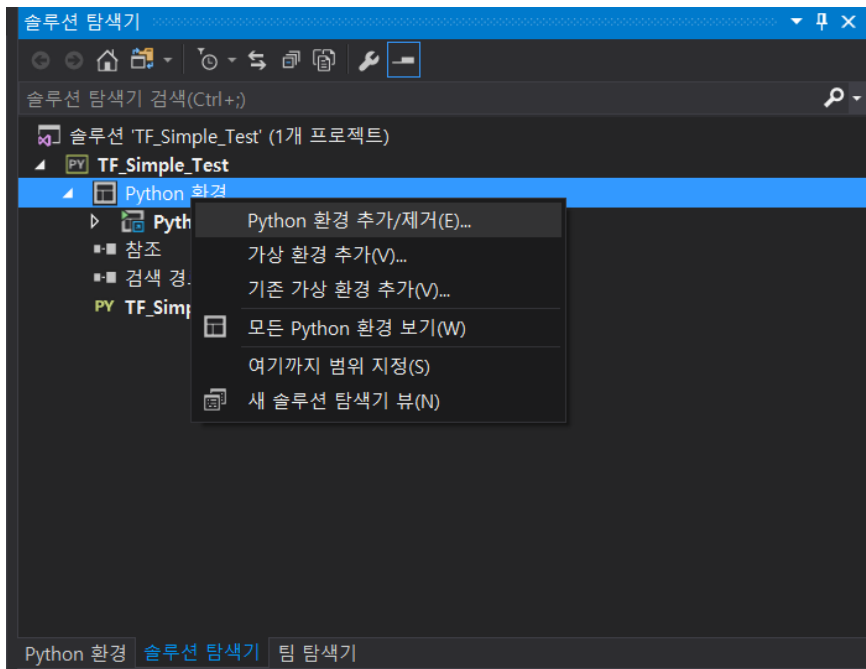


10. 설정화면이 뜨면 왼쪽과 같이, 디렉토리 위치와, Python 인터프리터 프로그램 등의 경로를 입력하고, [적용]을 눌러 Visual Studio에 추가합니다.

솔루션에 환경 추가.

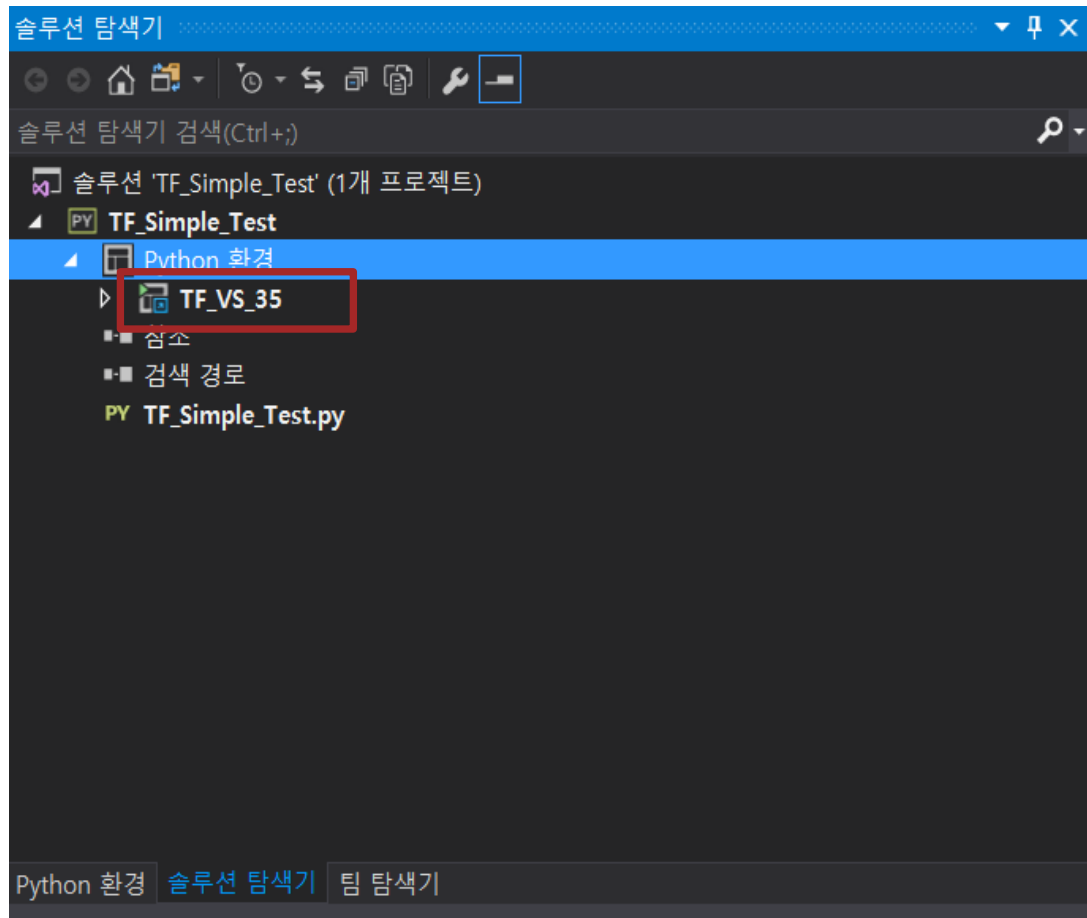
11. 다시 솔루션 탐색기로 돌아와서, 방금 Visual Studio에 추가한 가상환경을, 현재 프로젝트의 가상환경이 될 수 있게 변경합니다.

[Python 환경] → 마우스 오른쪽 버튼 클릭 → [Python 환경 추가/제거] → 기존 항목 체크 해제 후, 방금 추가한 환경만 체크 → [확인]



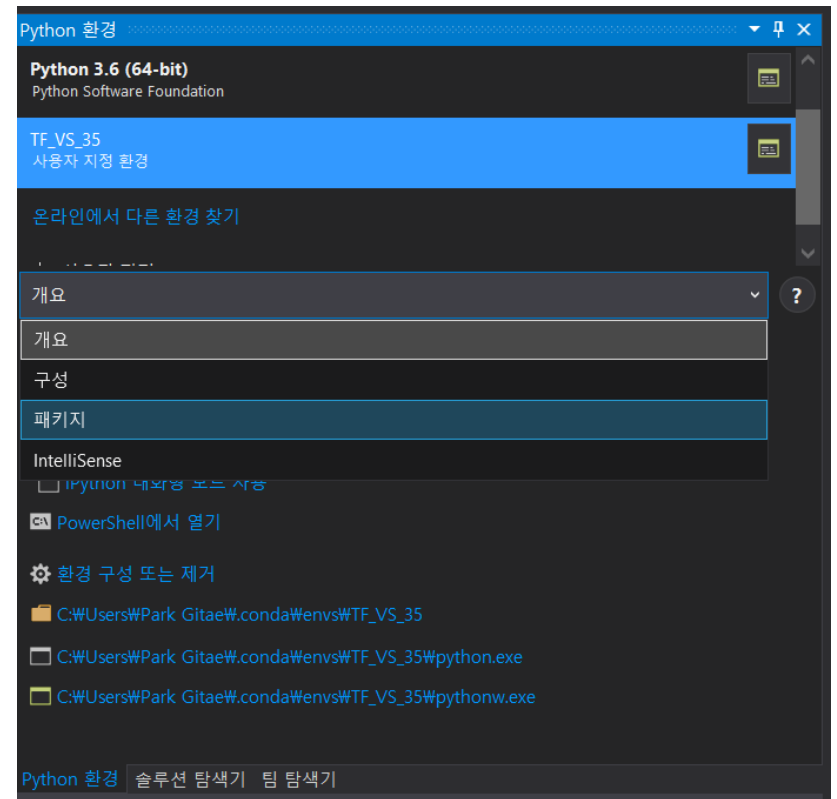
결과

12. 정상적으로 추가 되었다면, 아래와 같이 자신이 추가한 가상 환경 이름으로 변경 되었을 것입니다.





- 

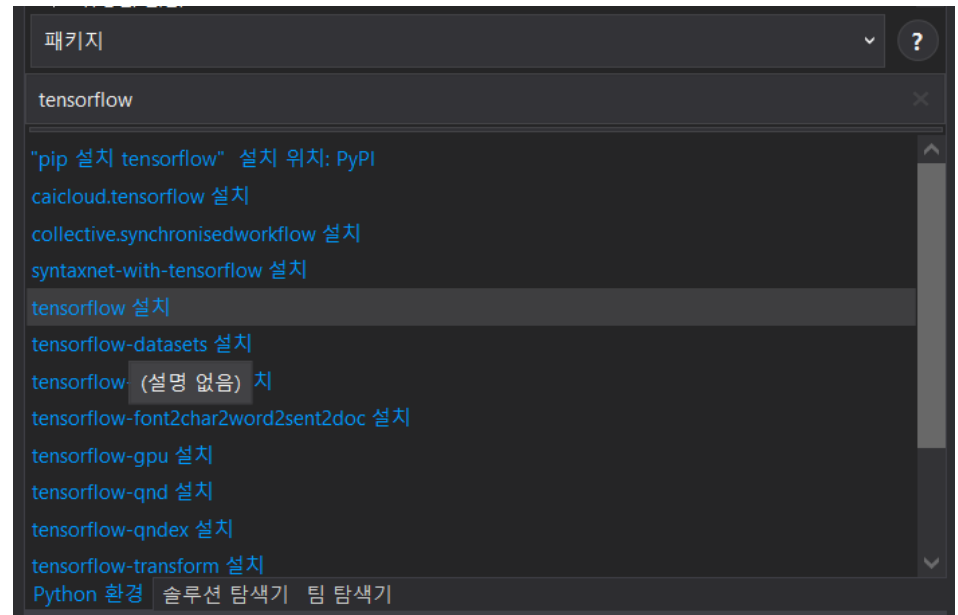
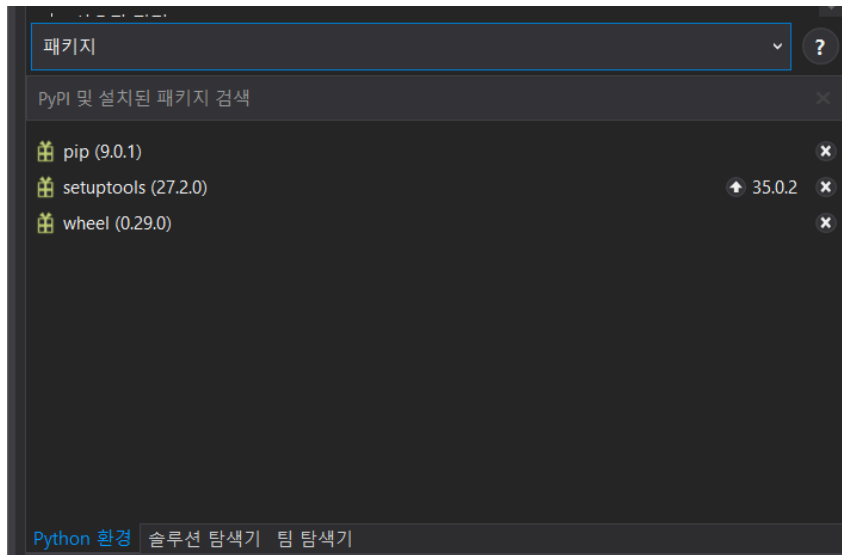




Tensorflow 설치



14. 아래와 같이 검색창이 뜨면 "tensorflow"를 검색합니다.
15. 해당 항목이 뜨면 [Tensorflow 설치]를 클릭합니다.





Tensorflow 설치 결과

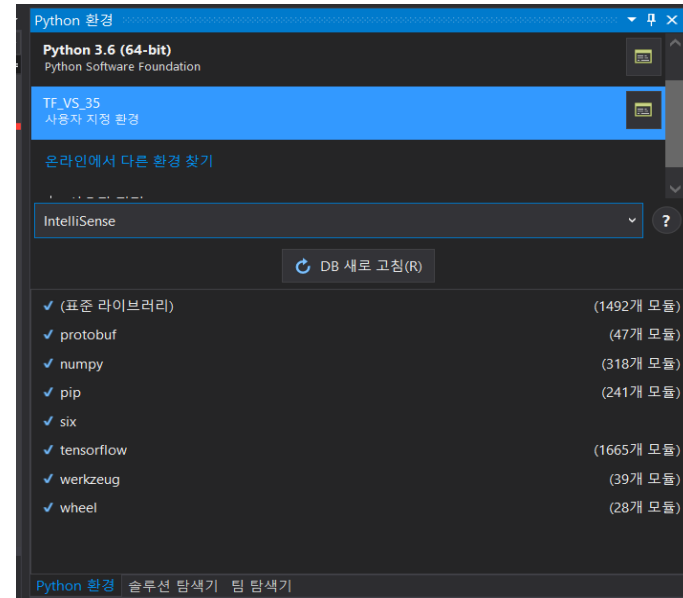
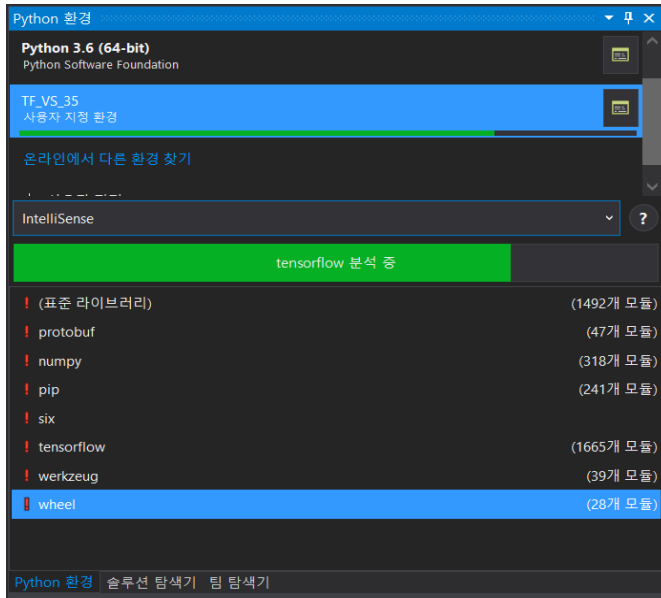


16. 출력 창을 통해 설치 결과를 확인합니다.

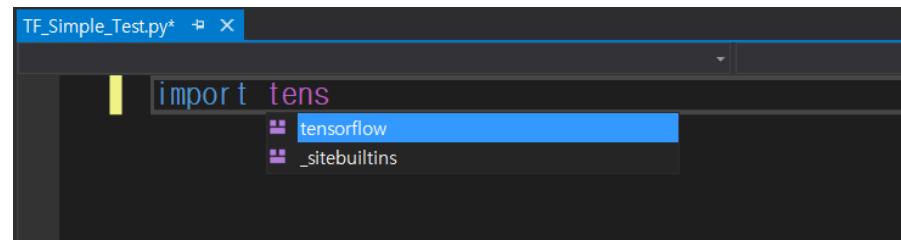
```
출력
출력 보기 선택(S): General
----- 'tensorflow' 설치 중 -----
Collecting tensorflow
  Downloading tensorflow-1.1.0-cp35-cp35m-win_amd64.whl (19.4MB)
Collecting werkzeug>=0.11.10 (from tensorflow)
  Downloading Werkzeug-0.12.2-py2.py3-none-any.whl (312kB)
Collecting protobuf>=3.2.0 (from tensorflow)
  Downloading protobuf-3.3.0.tar.gz (271kB)
Requirement already satisfied: wheel>=0.26 in c:\Users\park gitae\conda\envs\tf_vs_35\lib\site-packages (from tensorflow)
Collecting six>=1.10.0 (from tensorflow)
  Downloading six-1.10.0-py2.py3-none-any.whl
Collecting numpy>=1.11.0 (from tensorflow)
  Downloading numpy-1.12.1-cp35-none-win_amd64.whl (7.7MB)
Requirement already satisfied: setuptools in c:\Users\park gitae\conda\envs\tf_vs_35\lib\site-packages\setuptools-27.2.0-py3.5.egg (from proto
Building wheels for collected packages: protobuf
  Running setup.py bdist_wheel for protobuf: started
  Running setup.py bdist_wheel for protobuf: finished with status 'done'
  Stored in directory: C:\Users\Park Gitae\AppData\Local\pip\Cache\wheels\1b\42\4a\04c7343df5b629ec9c75655468dce7652b28026896b0209ba55
Successfully built protobuf
Installing collected packages: werkzeug, six, protobuf, numpy, tensorflow
Successfully installed numpy-1.12.1 protobuf-3.3.0 six-1.10.0 tensorflow-1.1.0 werkzeug-0.12.2
----- 'tensorflow' 설치됨 -----
```

Tensorflow 설치 결과

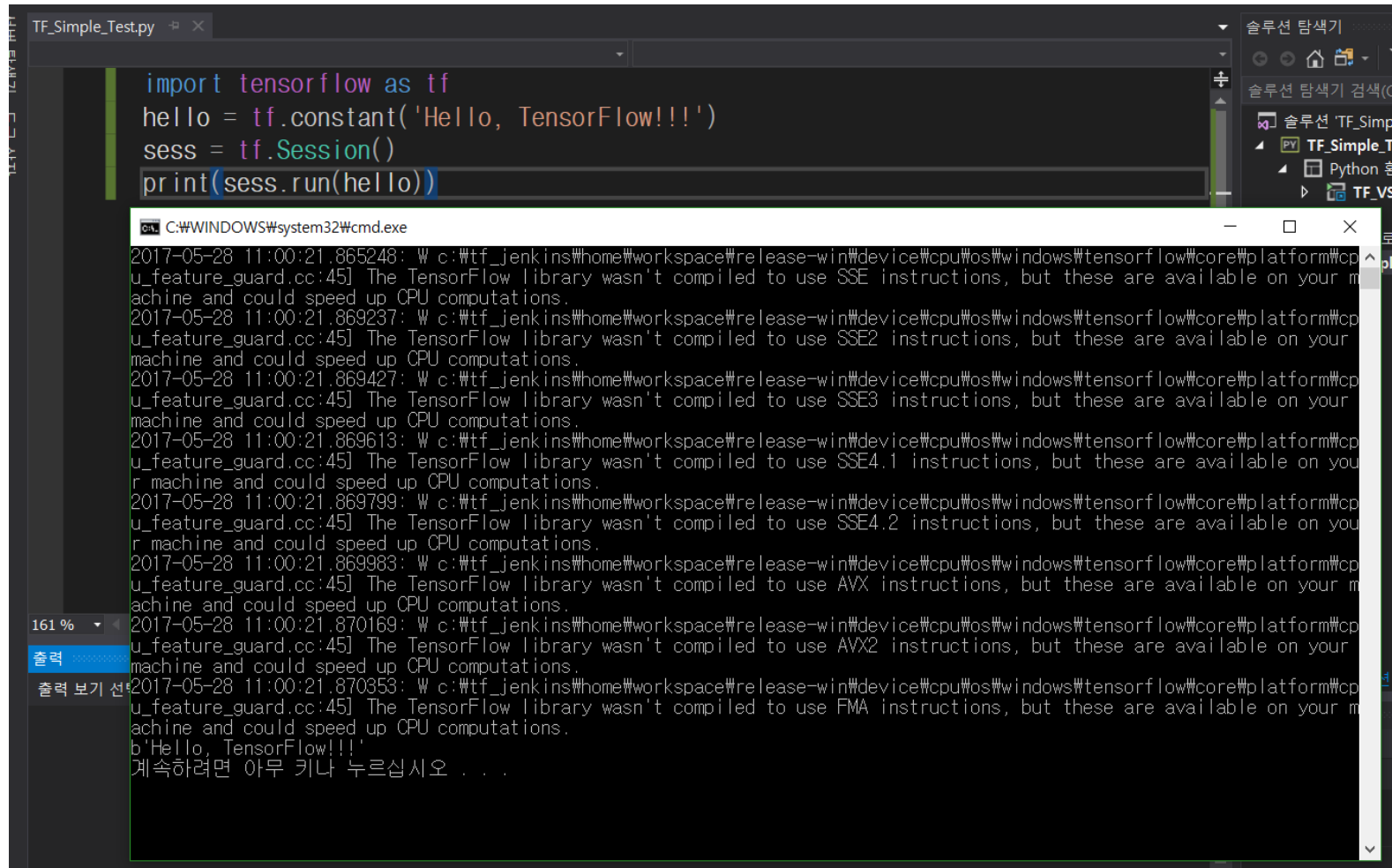
17. 설치가 완료 된 후.. 분석(?) 과정이 끝나기를 기다립니다.



18. 위의 과정이 끝나면, 다음과 같이 tensorflow를 import할 수 있고 자동완성 기능도 가능한 것을 볼 수 있습니다.



코드 실행 결과 #1



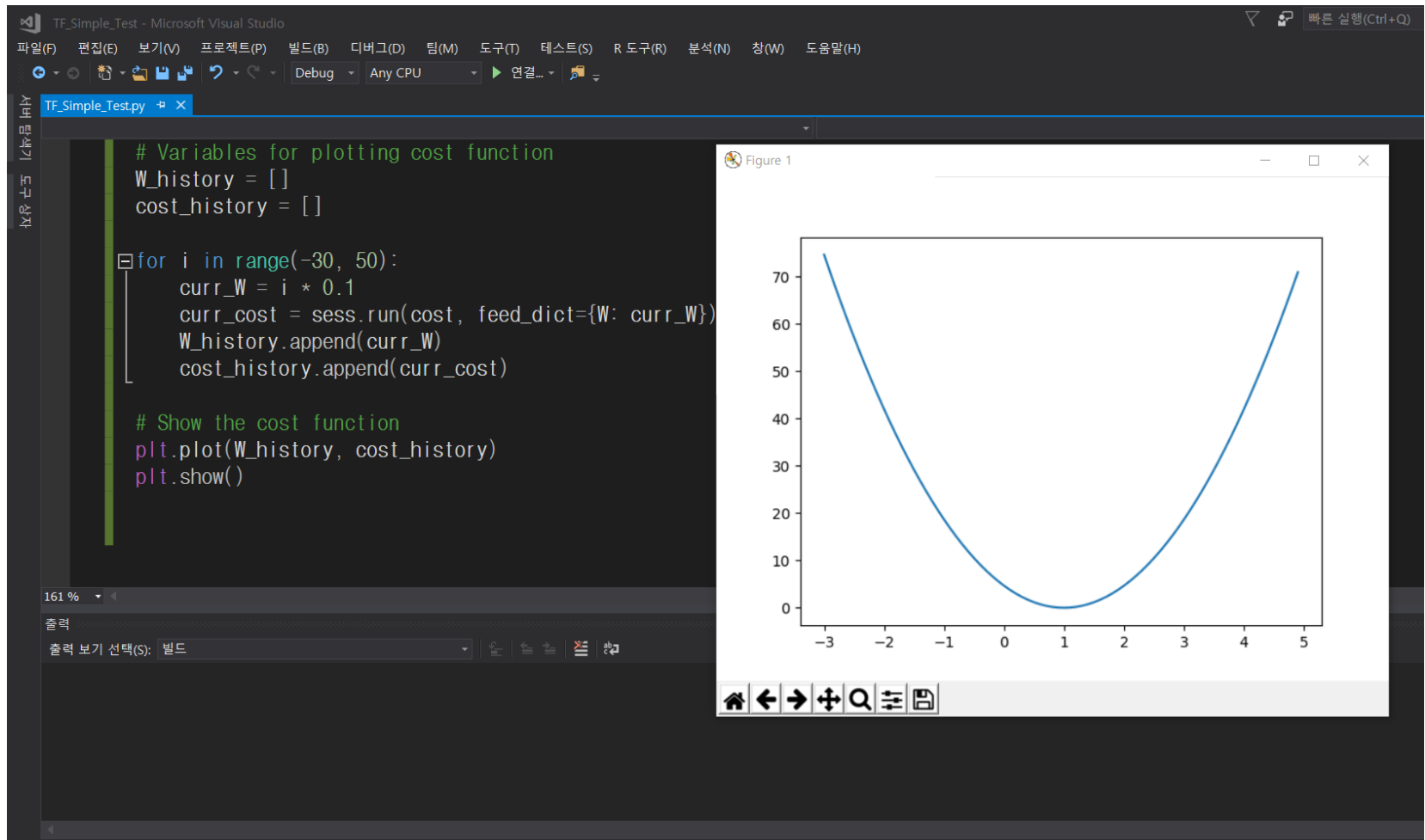
The screenshot shows a code editor with a file named `TF_Simple_Test.py`. The code in the editor is:

```
import tensorflow as tf
hello = tf.constant('Hello, TensorFlow!!!')
sess = tf.Session()
print(sess.run(hello))
```

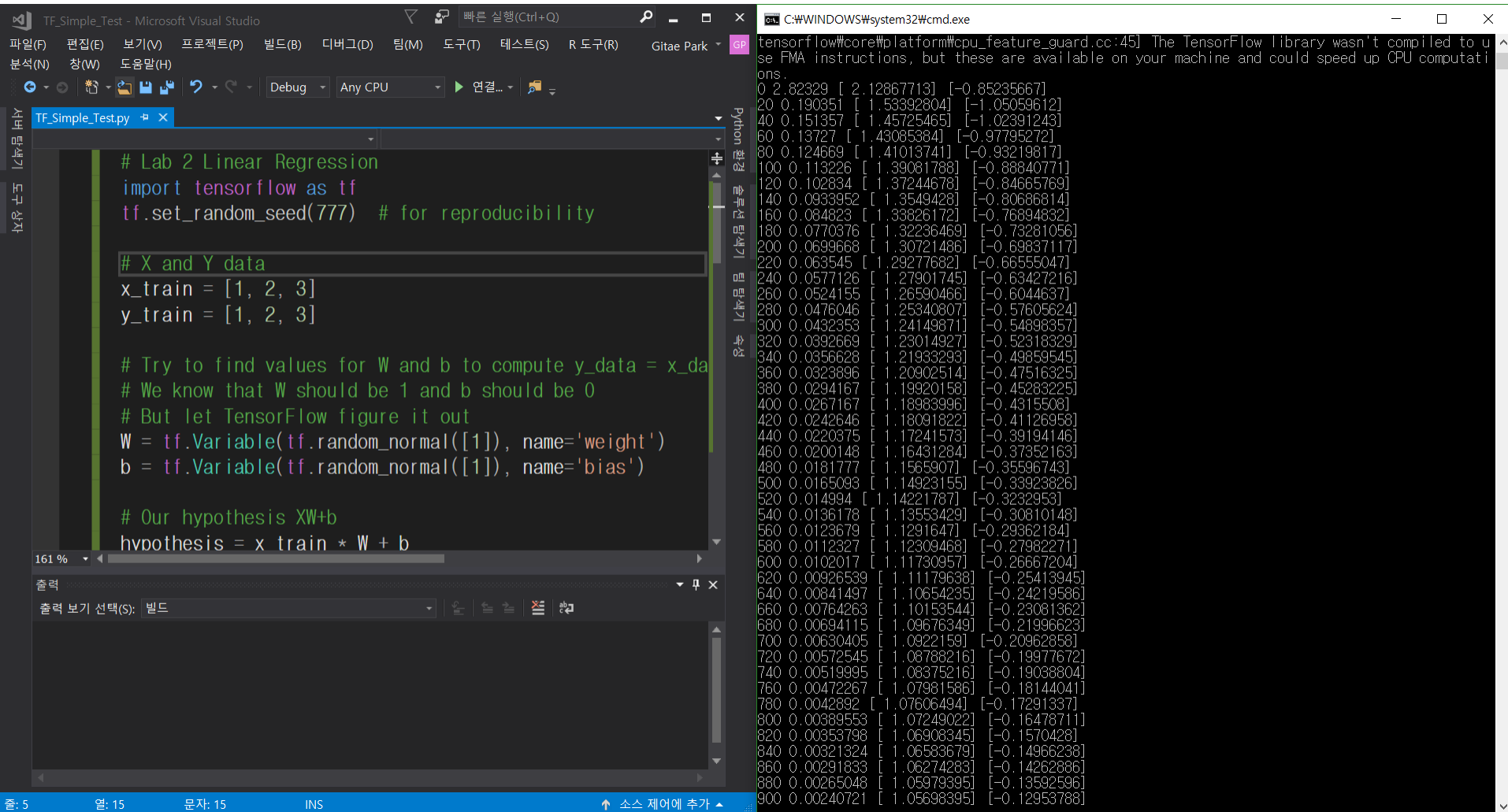
Below the code editor, a terminal window titled `C:\WINDOWS\system32\cmd.exe` displays the execution output. The output consists of several warning messages from TensorFlow about hardware acceleration (SSE, SSE2, SSE3, SSE4.1, SSE4.2, AVX, AVX2, FMA) and the final output of the program:

```
2017-05-28 11:00:21.865248: W c:\tf_jenkins\home\workspace\release-win\device\cpu\os\windows\tensorflow\core\platform\cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use SSE instructions, but these are available on your machine and could speed up CPU computations.
2017-05-28 11:00:21.869237: W c:\tf_jenkins\home\workspace\release-win\device\cpu\os\windows\tensorflow\core\platform\cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use SSE2 instructions, but these are available on your machine and could speed up CPU computations.
2017-05-28 11:00:21.869427: W c:\tf_jenkins\home\workspace\release-win\device\cpu\os\windows\tensorflow\core\platform\cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use SSE3 instructions, but these are available on your machine and could speed up CPU computations.
2017-05-28 11:00:21.869613: W c:\tf_jenkins\home\workspace\release-win\device\cpu\os\windows\tensorflow\core\platform\cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use SSE4.1 instructions, but these are available on your machine and could speed up CPU computations.
2017-05-28 11:00:21.869799: W c:\tf_jenkins\home\workspace\release-win\device\cpu\os\windows\tensorflow\core\platform\cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use SSE4.2 instructions, but these are available on your machine and could speed up CPU computations.
2017-05-28 11:00:21.869983: W c:\tf_jenkins\home\workspace\release-win\device\cpu\os\windows\tensorflow\core\platform\cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use AVX instructions, but these are available on your machine and could speed up CPU computations.
2017-05-28 11:00:21.870169: W c:\tf_jenkins\home\workspace\release-win\device\cpu\os\windows\tensorflow\core\platform\cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use AVX2 instructions, but these are available on your machine and could speed up CPU computations.
2017-05-28 11:00:21.870353: W c:\tf_jenkins\home\workspace\release-win\device\cpu\os\windows\tensorflow\core\platform\cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use FMA instructions, but these are available on your machine and could speed up CPU computations.
b'Hello, TensorFlow!!!'
계속하려면 아무 키나 누르십시오 . . .
```

코드 실행 결과 #2



코드 실행 결과 #3



The image shows a screenshot of a code editor (Visual Studio) and a terminal window. The code editor displays a Python script for linear regression using TensorFlow. The terminal window shows the output of the script, which is a list of 1000 rows of data, each containing three values: a scalar, a vector, and another scalar.

```
# Lab 2 Linear Regression
import tensorflow as tf
tf.set_random_seed(777) # for reproducibility

# X and Y data
x_train = [1, 2, 3]
y_train = [1, 2, 3]

# Try to find values for W and b to compute y_data = x_data * W + b
# We know that W should be 1 and b should be 0
# But let TensorFlow figure it out
W = tf.Variable(tf.random_normal([1]), name='weight')
b = tf.Variable(tf.random_normal([1]), name='bias')

# Our hypothesis XW+b
hypothesis = x_train * W + b
```

terminal C:\WINDOWS\system32\cmd.exe

```
tensorflow\cc\core\platform\cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use FMA instructions, but these are available on your machine and could speed up CPU computations.
0 2.82329 [ 2.12867713] [-0.85235667]
20 0.190351 [ 1.53392804] [-1.05059612]
40 0.151357 [ 1.45725465] [-1.02391243]
60 0.13727 [ 1.43085384] [-0.97795272]
80 0.124669 [ 1.41013741] [-0.93219817]
100 0.113226 [ 1.39081788] [-0.88840771]
120 0.102834 [ 1.37244678] [-0.84665769]
140 0.0933952 [ 1.3549428] [-0.80686814]
160 0.084823 [ 1.33826172] [-0.76894832]
180 0.0770376 [ 1.32236469] [-0.73281056]
200 0.0699668 [ 1.30721486] [-0.69837117]
220 0.063545 [ 1.29277682] [-0.66555047]
240 0.0577126 [ 1.27901745] [-0.63427216]
260 0.0524155 [ 1.26590466] [-0.60446371]
280 0.0476046 [ 1.25340807] [-0.57605624]
300 0.0432353 [ 1.24149871] [-0.54898357]
320 0.0392669 [ 1.23014927] [-0.52318329]
340 0.0356628 [ 1.21933293] [-0.49859545]
360 0.0323896 [ 1.20902514] [-0.47516325]
380 0.0294167 [ 1.19920158] [-0.45283225]
400 0.0267167 [ 1.18983996] [-0.4315508]
420 0.0242646 [ 1.18091822] [-0.41126958]
440 0.0220375 [ 1.17241573] [-0.39194146]
460 0.0200148 [ 1.16431284] [-0.37352163]
480 0.0181777 [ 1.1565907] [-0.35596743]
500 0.0165093 [ 1.14923155] [-0.33923826]
520 0.014994 [ 1.14221787] [-0.3232953]
540 0.0136178 [ 1.13553429] [-0.30810148]
560 0.0123679 [ 1.1291647] [-0.29362184]
580 0.0112327 [ 1.12309468] [-0.27982271]
600 0.0102017 [ 1.11730957] [-0.26667204]
620 0.00926539 [ 1.11179638] [-0.25413945]
640 0.00841497 [ 1.10654235] [-0.24219586]
660 0.00764263 [ 1.10153544] [-0.23081362]
680 0.00694115 [ 1.09676349] [-0.21996623]
700 0.00630405 [ 1.0922159] [-0.20962858]
720 0.00572545 [ 1.08788216] [-0.19977672]
740 0.00519995 [ 1.08375216] [-0.19038804]
760 0.00472267 [ 1.07981586] [-0.18144041]
780 0.0042892 [ 1.07606494] [-0.17291337]
800 0.00389553 [ 1.07249022] [-0.16478711]
820 0.00353798 [ 1.06908345] [-0.1570428]
840 0.00321324 [ 1.06583679] [-0.14966238]
860 0.00291833 [ 1.06274283] [-0.14262886]
880 0.00265048 [ 1.05979395] [-0.13592596]
900 0.00240721 [ 1.05698395] [-0.12953788]
```

줄: 5 열: 15 문자: 15 INS 소스 제어에 추가