

파이썬 프로젝트에서  
데이터 과학을 위한 준비  
numpy

# numpy npsample1.py

The screenshot shows the PyCharm IDE with a project named 'pyc prj01'. The main editor displays the file 'np 샘플코드1.py' with the following Python code:

```

1  import numpy as np
2
3  # Return random floats in the half-open interval [0, 1)
4  # Alias for `random_sample` to ease forward compatibility
5  a = np.random.random((2, 3))
6  print(a)
7  a = np.random.random_sample([2, 4])
8  print(a)

```

The Run window at the bottom shows the command executed: `D:\ve\venv\Scripts\python.exe "D:/pyc prj01/np 샘플코드1.py"`. The output displays two arrays of random floats:

```

[[0.22804363 0.96944722 0.74731405]
 [0.13548272 0.40393197 0.19429434]]
[[0.48850094 0.58055657 0.69788749 0.75680957]
 [0.88506482 0.55466217 0.32290537 0.44633951]]

```

The status bar at the bottom indicates the file encoding is UTF-8, 4 spaces for indentation, and Python 3.8 (venv) is the interpreter.

# numpy npsample1.py 이해

```
import numpy as np
```

```
# Return random floats in the half-open interval  
[0.0, 1.0).
```

```
# Alias for `random_sample` to ease forward-  
porting to the new random API.
```

```
a = np.random.random((2, 3))  
print(a)  
a = np.random.random_sample([2, 4])  
print(a)
```

```
# 정규분포의 난수 생성
```

```
b = np.random.randn(2)  
print(b)  
b = np.random.randn(2, 3)  
print(b)  
b = np.random.randn(3, 4)  
print(b)
```

```
# 값이 모두 1인 텐서
```

```
c = np.ones(3)  
print(c)
```

```
# 값이 모두 0인 텐서
```

```
c = np.zeros((2, 3))  
print(c)  
c = np.zeros([3, 4])  
print(c)
```

```
D:\wenv\Scripts\python.exe "D:/pyc prj01/np 샘플코드1.py"
```

```
[[0.22804363 0.96944722 0.74731405]  
 [0.13548272 0.40393197 0.19429434]]  
[[0.48850094 0.58055657 0.69788749 0.75680957]  
 [0.88506482 0.55466217 0.32290537 0.44633951]]  
[-2.68298825 -0.99579772]  
[[-0.16099173 1.29978199 1.66469465]  
 [ 0.68463138 -1.58433849 -0.06550564]]  
[[ 0.02758157 -0.77454657 0.49862683 0.70299809]  
 [ 1.57338867 -0.71034582 0.06683078 -1.51798292]  
 [-0.20716991 -0.45346911 0.20735756 -0.76737582]]  
[1. 1. 1.]  
[[0. 0. 0.]  
 [0. 0. 0.]  
 [0. 0. 0. 0.]  
 [0. 0. 0. 0.]  
 [0. 0. 0. 0.]  
 [0. 0. 0. 0.]]
```

Process finished with exit code 0

# numpy npsample2.py

The screenshot shows the PyCharm IDE with a project named 'pyc prj01'. The main editor displays the file 'np 샘플코드1.py' with the following code:

```

1  import numpy as np
2
3  # Return random floats in the half-open interval [0, 1)
4  # Alias for `random_sample` to ease forward compatibility
5  a = np.random.random((2, 3))
6  print(a)
7  a = np.random.random_sample([2, 4])
8  print(a)

```

The Run window at the bottom shows the command executed and the output:

```

Run: D:\ve\venv\Scripts\python.exe "D:/pyc prj01/np 샘플코드1.py"
[[0.22804363 0.96944722 0.74731405]
 [0.13548272 0.40393197 0.19429434]]
[[0.48850094 0.58055657 0.69788749 0.75680957]
 [0.88506482 0.55466217 0.32290537 0.44633951]]

```

The status bar at the bottom indicates the file encoding is UTF-8, 4 spaces for indentation, and Python 3.8 (venv) is the interpreter.

# numpy npsample2.py 이해

```
import numpy as np
```

```
# [0, 1) 난수 생성
```

```
a = np.random.random((2, 3))
```

```
print(type(a))
```

```
print(a)
```

```
b = a.reshape(3, 2)
```

```
print(b)
```

```
# 정규분포의 난수 생성
```

```
c = np.random.randn(3, 4)
```

```
print(c.reshape(2, 6))
```

```
# 값이 모두 1인 텐서
```

```
d = np.ones((4, 5))
```

```
print(d.reshape(2, 5, 2))
```

```
# 값이 모두 0인 텐서
```

```
e = np.zeros((3, 4))
```

```
print(e.reshape(2, 3, 2))
```

```
D:\venv_test\Scripts\python.exe "D:/pyc prj02/np 샘플코드2.py"
```

```
<class 'numpy.ndarray'>
```

```
[[0.82253794 0.94805217 0.18727646]
```

```
 [0.93734334 0.64931534 0.34043917]]
```

```
[[0.82253794 0.94805217]
```

```
 [0.18727646 0.93734334]
```

```
 [0.64931534 0.34043917]]
```

```
[[ 0.75831872  0.48717349  0.28570681 -1.52723675 -0.44313711 -0.95670083]
```

```
 [-0.12955399  0.60647143  0.33132981  0.63755303  1.12009118  0.09013185]]
```

```
[[[1. 1.]
```

```
 [1. 1.]
```

```
 [1. 1.]
```

```
 [1. 1.]
```

```
 [1. 1.]]
```

```
[[[1. 1.]
```

```
 [1. 1.]
```

```
 [1. 1.]
```

```
 [1. 1.]
```

```
 [1. 1.]]]
```

```
[[[0. 0.]
```

```
 [0. 0.]
```

```
 [0. 0.]]
```

```
[[[0. 0.]
```

```
 [0. 0.]
```

```
 [0. 0.]]]
```

```
Process finished with exit code 0
```

# numpy npsample3.py

The screenshot shows the PyCharm IDE with a project named 'pyc prj03'. The file 'np 샘플코드3.py' is open in the editor. The code in the file is as follows:

```

1  # http://riseshia.github.io/2017/01/30/numpy-tutorial-with-code.html
2
3  import numpy as np
4
5  a = np.arange(12) # array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11])
6  print(a)
7
8  b = a.reshape(4, 3) # 변환된 행렬을 반환
9  print(b)
10 a.resize((3, 4)) # 자체를 변환함
11 print(a)
12
13 b = a.flatten()
14 print(b)
15 b = a.ravel()
16 print(b)
17
18 b = a.T # 전치 행렬
19 print(a)
20 print(b)
21
22 a.shape = 2, 6 # 파괴적
23 print(a)
24
25 b = a.reshape(3, -1) #=> 변환된 행렬을 반환
26 print(b)
27

```

The left sidebar shows the project structure:

- pyc prj03
  - np 샘플코드3.py
  - External Libraries
    - Python 3.8 (env-Snxa-AB5)
      - Binary Skeletons
      - DLLs library root
      - Extended Definitions
      - Lib
      - Lib library root
      - penv-Snxa-AB5 library root
      - Scripts
        - activate
        - activate.bat
        - activate.ps1
        - activate.xsh
        - activate\_this.py
        - deactivate.bat
        - django-admin.exe
        - django-admin.py
        - easy\_install.exe
        - easy\_install-3.8.exe
        - f2py.exe
        - pip.exe
        - pip3.8.exe
        - pip3.exe
        - python.exe
        - python3.dll
        - python3\_d.dll
        - python38.dll
        - python38\_d.dll
        - python\_d.exe
        - pythonw.exe
        - sqlformat.exe
        - wheel.exe
      - site-packages
      - Typeshed Stubs

The bottom status bar shows 'Python 3.8 (env-Snxa-AB5)'.

A notification bubble in the bottom right corner says: "Looks like you're using NumPy. Would you like to turn scientific mode on? Use scientific mode Keep current layout..."

# numpy npsample3.py 이해

# <http://riseshia.github.io/2017/01/30/numpy-tutorial-with-code.html>

```
import numpy as np

a = np.arange(12) # array([ 0, ... 11])
print(a)

b = a.reshape(4, 3) # 변환된 행렬을 반환
print(b)
a.resize((3, 4)) # 자체를 변환함
print(a)

b = a.flatten()
print(b)
b = a.ravel()
print(b)

b = a.T # 전치 행렬
print(a)
print(b)

a.shape = 2, 6 # 파괴적
print(a)

b = a.reshape(3, -1) # 변환된 행렬을 반환
print(b)
```

```
C:\Users\W217\virtualenvs\Wpenv-Snxa-
AB5\Scripts\python.exe "D:/pyc prj03/np 샘플코드3.py"
[ 0  1  2  3  4  5  6  7  8  9 10 11]
[[ 0  1  2]
 [ 3  4  5]
 [ 6  7  8]
 [ 9 10 11]]
[[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]]
[ 0  1  2  3  4  5  6  7  8  9 10 11]
[ 0  1  2  3  4  5  6  7  8  9 10 11]
[[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]]
[[ 0  4  8]
 [ 1  5  9]
 [ 2  6 10]
 [ 3  7 11]]
[[ 0  1  2  3  4  5]
 [ 6  7  8  9 10 11]]
[[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]]
```

Process finished with exit code 0

# Numpy npsample4.py 이해

```
import numpy as np
```

```
a = np.array(3)
print(type(a))
print(a.ndim, a.shape)
print(a)
```

```
<class 'numpy.ndarray'>
0 ()
3
```

```
b = np.array([1, 2, 3])
print(type(b))
print(b.ndim, b.shape)
print(b)
```

```
<class 'numpy.ndarray'>
1 (3,)
[1 2 3]
```

```
c = np.array([[1, 2], [3, 4]])
print(type(c))
print(c.ndim, c.shape)
print(c)
```

```
<class 'numpy.ndarray'>
2 (2, 2)
[[1 2]
 [3 4]]
```

```
d = np.arange(10)
print(type(d))
print(d.ndim, d.shape)
print(d)
```

```
<class 'numpy.ndarray'>
1 (10,)
[0 1 2 3 4 5 6 7 8 9]
```

```
e = range(10)
print(type(e))
print(e)
print(list(e))
```

```
<class 'range'>
range(0, 10)
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```



# Numpy npsample5.py 이해

- numpy 테스트
  - 브로드캐스팅과 행렬 모양 변환

C:\Windows\System32\cmd.exe - python

```
(venv) D:\#ve>python
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 22:39:24) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy as np
>>> a = np.array([[1, 2]])
>>> a
array([[1, 2]])
>>> b = np.array([[1], [2]])
>>> print(b)
[[1]
 [2]]
>>> c = a + b
>>> print(c)
[[2 3]
 [3 4]]
>>> x = np.array(np.random.random(10))
>>> x
array([0.62964497, 0.20982989, 0.39815468, 0.41462358, 0.41903129,
        0.11562117, 0.54325543, 0.16417797, 0.7962333 , 0.77974707])
>>> y = x.reshape(2, 5)
>>> y
array([[0.62964497, 0.20982989, 0.39815468, 0.41462358, 0.41903129],
       [0.11562117, 0.54325543, 0.16417797, 0.7962333 , 0.77974707]])
>>> print(y)
[[0.62964497 0.20982989 0.39815468 0.41462358 0.41903129]
 [0.11562117 0.54325543 0.16417797 0.7962333 0.77974707]]
>>> -
```

# 일반 리스트 더하기

```
m = [1, 2, 3]
```

```
n = [4, 5, 6]
```

```
print(m + n)
```

```
# print(m - n)
```

import numpy as np

# 브로드캐스팅

```
a = np.array([[1, 2]])
```

```
print(a)
```

```
b = np.array([[1], [2]])
```

```
print(b)
```

```
c = a + b
```

```
print(c)
```

# 행렬 모양 바꾸기

```
x = np.array(np.random.random(10))
```

```
print(x)
```

```
y = x.reshape(2, 5)
```

```
print(y)
```

# 파이썬 프로젝트에서 데이터 과학을 위한 준비 pandas

# Pandas, pdsample1.py

```

1  import numpy as np
2  import pandas as pd
3
4  s = pd.Series([1, 2, 3, 4, 5, 6], index=pd.date_range('20200102', periods=6))
5  print(s)
6
7  s = pd.Series(np.random.randint(0, 7, size=10))
8  print(s)
9  print(s.value_counts())
10
11 s = pd.Series(['A', 'B', 'C', 'Aaba', 'Baca', np.nan, 'CABA', 'dog', 'cat'])
12 print(s)
13 s.str.lower()
14 print(s)
15
16 df = pd.DataFrame(np.random.randn(10, 4))
17 print(df)
18
19

```

Low Memory: The IDE is running low on memory and this might affect performance. Please consider increasing available heap. // Anal... (today 오후 5:00) 9:24 CRLF UTF-8 4 spaces Python 3.8 (venv\_test)

# Pandas, pdsample1.py 이해

```
import numpy as np
import pandas as pd
```

```
s = pd.Series([1, 2, 3, 4, 5, 6], index=pd.date_range('20200102', periods=6))
print(s)
```

```
s = pd.Series(np.random.randint(0, 7, size=10))
print(s)
print(s.value_counts())
```

```
s = pd.Series(['A', 'B', 'C', 'Aaba', 'Baca', np.nan, 'CABA', 'dog', 'cat'])
print(s)
s.str.lower()
print(s)
```

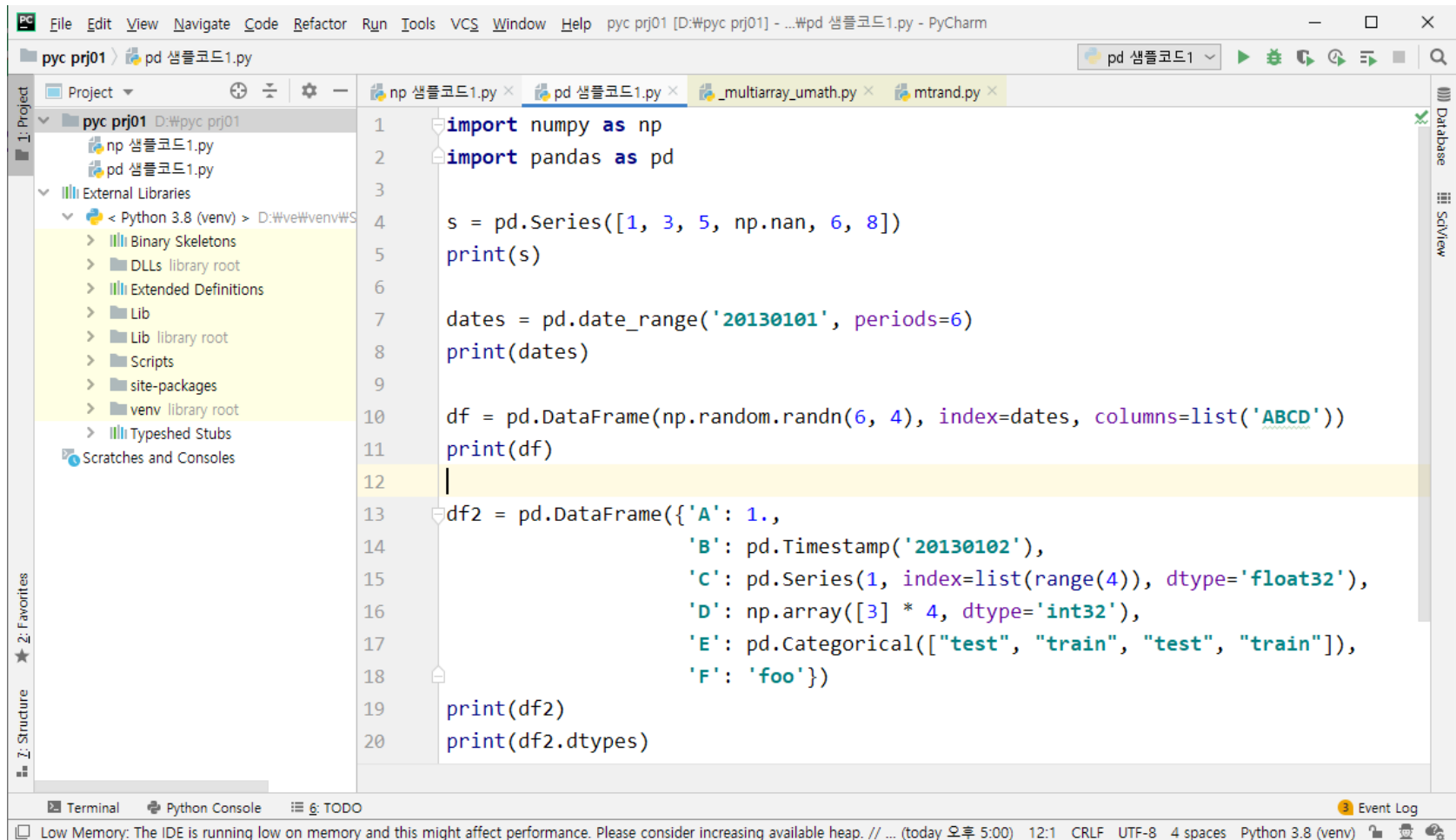
```
df = pd.DataFrame(np.random.randn(10, 4))
print(df)
```

```
D:\WveWvenv_test\Scripts\python.exe
"D:/pyc prj02/pd 샘플코드2.py"
2020-01-02    1
2020-01-03    2
2020-01-04    3
2020-01-05    4
2020-01-06    5
2020-01-07    6
Freq: D, dtype: int64
0    5
1    2
2    0
3    5
4    6
5    1
6    6
7    6
8    2
9    4
dtype: int32
6    3
5    2
2    2
4    1
1    1
0    1
dtype: int64
```

```
0    A
1    B
2    C
3    Aaba
4    Baca
5    NaN
6    CABA
7    dog
8    cat
dtype: object
0    A
1    B
2    C
3    Aaba
4    Baca
5    NaN
6    CABA
7    dog
8    cat
dtype: object
      0      1      2      3
0 -0.827856 -0.591318 -0.446506  1.639843
1 -0.455133  0.652168 -0.542553  0.015321
2 -0.790744  0.088498  0.499716 -0.355695
3  0.252766  0.853125  1.609860 -1.235949
4 -0.778862  0.734792 -0.559469  2.637026
5 -0.066913 -2.701452  0.196265 -1.475756
6 -1.171109 -1.312982 -0.123534 -0.467198
7 -0.560191 -0.025275  0.336903 -0.202051
8 -0.472363  2.441893  2.044766  0.685911
9 -0.899807 -1.176664  0.391078  0.148584
```

Process finished with exit code 0

# Pandas, pdsample2.py



# Pandas, pdsample2.py 이해

```
import numpy as np
import pandas as pd

s = pd.Series([1, 3, 5, np.nan, 6, 8])
print(s)

dates = pd.date_range('20130101', periods=6)
print(dates)

df = pd.DataFrame(np.random.randn(6, 4), index=dates,
                  columns=list('ABCD'))

print(df)

df2 = pd.DataFrame(
    {'A': 1.,
     'B': pd.Timestamp('20130102'),
     'C': pd.Series(1, index=list(range(4)), dtype='float32'),
     'D': np.array([3] * 4, dtype='int32'),
     'E': pd.Categorical(["test", "train", "test", "train"]),
     'F': 'foo'})

print(df2)
print(df2.dtypes)
```

```
D:\wenv\Scripts\python.exe "D:/pyc prj01/pd 샘플코드1.py"
0    1.0
1    3.0
2    5.0
3    NaN
4    6.0
5    8.0
dtype: float64
DatetimeIndex(['2013-01-01', '2013-01-02', '2013-01-03', '2013-01-04',
              '2013-01-05', '2013-01-06'],
              dtype='datetime64[ns]', freq='D')
           A         B         C         D
2013-01-01  0.617621  0.711937  0.555614  1.261003
2013-01-02 -1.378531 -0.325374  0.788234  1.223037
2013-01-03 -1.228281 -0.682720 -0.564663  0.107814
2013-01-04  0.552391 -0.354623 -0.488619 -0.072650
2013-01-05 -1.099271  0.108479 -2.062795  2.163172
2013-01-06 -0.034693  0.705632  0.194938  0.800431
           A         B  C  D     E  F
0  1.0 2013-01-02  1.0  3  test  foo
1  1.0 2013-01-02  1.0  3  train foo
2  1.0 2013-01-02  1.0  3  test  foo
3  1.0 2013-01-02  1.0  3  train foo
A          float64
B  datetime64[ns]
C          float32
D           int32
E          category
F           object
dtype: object
```

Process finished with exit code 0

# Pandas, pdsample3.py

The screenshot shows the PyCharm IDE with a project named 'cond prj'. The file 'pandas 테스트.py' is open in the editor. The code in the file is as follows:

```

1 import numpy as np
2 import pandas as pd
3
4 dates = pd.date_range('20210101', periods=6)
5 print(dates)
6
7 # 정규분포에서 6행 4열의 난수 생성
8 df = pd.DataFrame(np.random.randn(6, 4), index=dates, columns=list('ABCD'))
9
10 print(df)
11 print(df.head()) # 첫 5개 행
12 print(df.tail(3)) # 마지막 3개 행
13 print(df.describe()) # 전체 통계 데이터
14

```

The Run console at the bottom shows the output of the script:

```

Run: pandas 테스트
"D:\Anaconda3\envs\cond prj\python.exe" "D:/cond prj/pandas 테스트.py"
DatetimeIndex(['2021-01-01', '2021-01-02', '2021-01-03', '2021-01-04',
              '2021-01-05', '2021-01-06'],
              dtype='datetime64[ns]', freq='D')

```

# Pandas, pdsample3.py 이해

```
import numpy as np
import pandas as pd

dates = pd.date_range('20210101', periods=6)
print(dates)

# 정규분포에서 6행 4열의 난수 생성
df = pd.DataFrame(np.random.randn(6, 4),
                  index=dates, columns=list('ABCD'))

print(df)
print(df.head()) # 첫 5개 행
print(df.tail(3)) # 마지막 3개 행
print(df.describe()) # 전체 통계 데이터
```

pandas 테스트

```
"D:\Anaconda3\envs\cond_prj\python.exe" "D:/cond_prj/pandas 테스트.py"
```

```
DatetimeIndex(['2021-01-01', '2021-01-02', '2021-01-03', '2021-01-04',
               '2021-01-05', '2021-01-06'],
              dtype='datetime64[ns]', freq='D')
```

	A	B	C	D
2021-01-01	-0.958072	-1.586224	0.035237	-0.232481
2021-01-02	1.575099	0.614962	-1.060176	-1.841971
2021-01-03	0.141609	-1.511011	0.308956	0.467325
2021-01-04	1.394936	1.297629	-1.176749	-0.972958
2021-01-05	-0.197864	1.432869	0.252815	-1.134095
2021-01-06	0.176109	1.741329	1.367940	1.021468

```
count  6.000000  6.000000  6.000000  6.000000
mean    0.355303  0.331592 -0.045329 -0.448785
std     0.967209  1.502458  0.951657  1.070800
min    -0.958072 -1.586224 -1.176749 -1.841971
25%    -0.112996 -0.979518 -0.786323 -1.093810
50%     0.158859  0.956296  0.144026 -0.602720
75%     1.090229  1.399059  0.294920  0.292373
max     1.575099  1.741329  1.367940  1.021468
```

Process finished with exit code 0



# 파이썬 프로젝트에서 데이터 시각화를 위한 준비 matplotlib

# Matplotlib, matplot1.py

- `pip install matplotlib`

> `pip show matplotlib`

설치 유무 확인 방법

The screenshot shows the PyCharm IDE interface. The main editor displays a Python script named `plot 테스트.py` with the following code:

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3
4 x = np.random.random(10)
5 print(x)
6
7 plt.plot(x)
8 plt.show()
9
```

The SciView panel on the right shows a line plot of the generated data. The x-axis ranges from 0 to 5, and the y-axis ranges from 0.2 to 1.0. The plot shows a blue line with several peaks and valleys.

The Terminal panel at the bottom shows the output of the `pip install matplotlib` command:

```
(c) 2018 Microsoft Corporation. All rights reserved.
(venv) D:\numpy prj>pip install matplotlib
Collecting matplotlib
  Using cached https://files.pythonhosted.org/packages/dd/73/d
c25ca27a9960539ef984921b0d42368445b856ae0861c3acba542b9a39c/ma
```

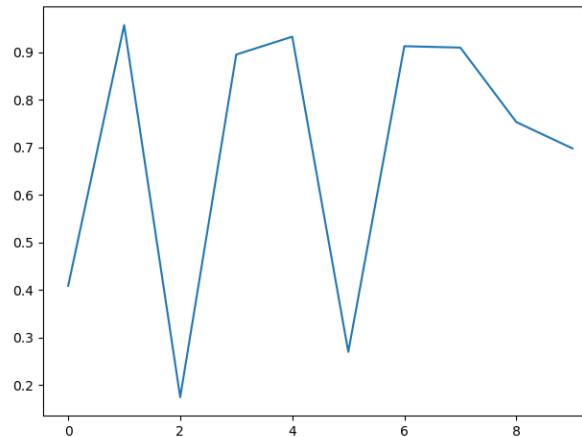
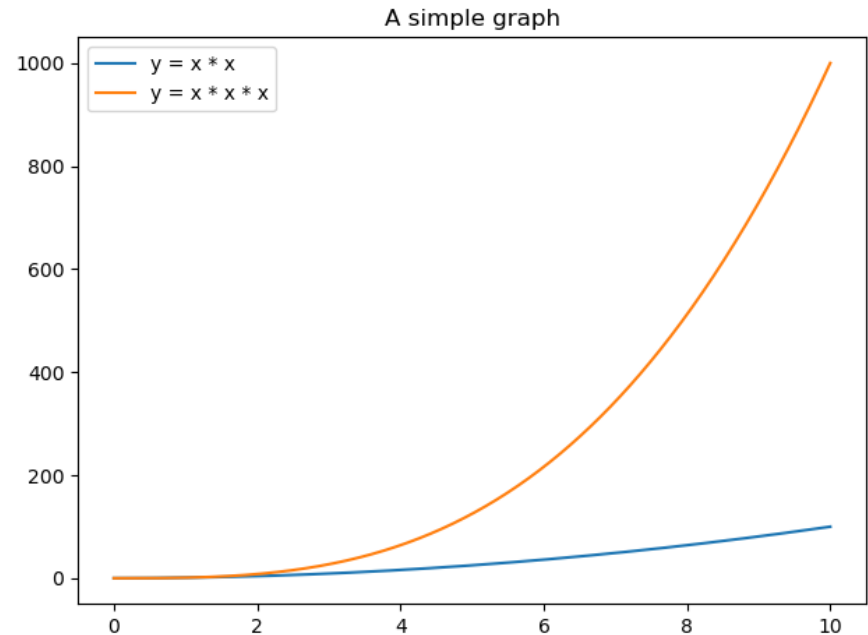
# Matplotlib, matplotlib.pyplot 이해

```
import numpy as np
import matplotlib.pyplot as plt
```

```
x = np.random.random(10)
print(x)
plt.plot(x)
plt.show()
```

```
x = np.linspace(0.0, 10.0, 1000)
```

```
plt.plot(x, x*x)
plt.plot(x, x**3)
plt.title('A simple graph')
plt.legend(['y = x * x', 'y = x * x * x'], loc='upper left')
plt.show()
```



# Matplotlib, matplotlib.pyplot 이해

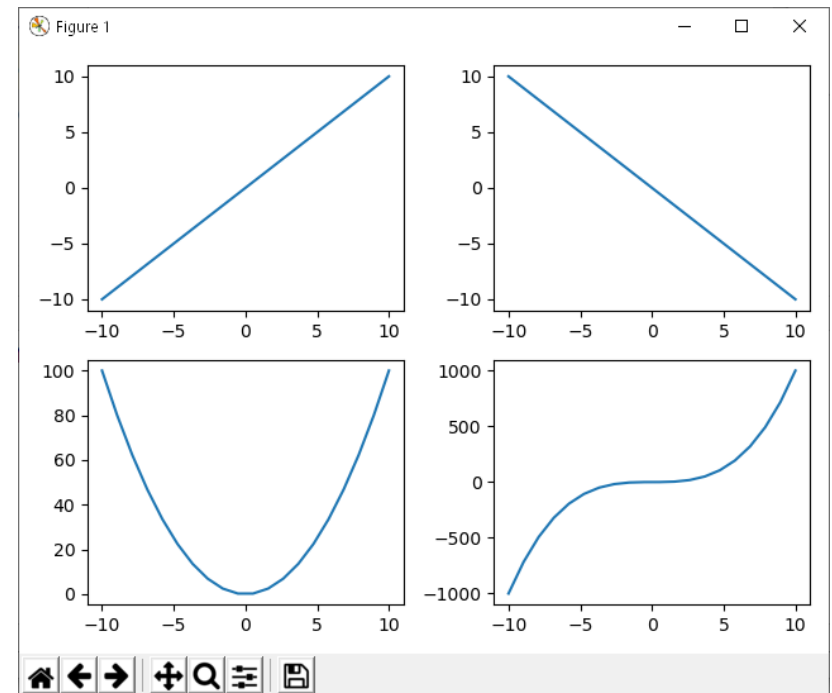
## 여러 그림을 하나의 캔버스에 그리는 방법

```
import numpy as np
import matplotlib.pyplot as plt

# -10에서 10까지 20등분한 자료
x = np.linspace(-10, 10, 20)

# 2행 2열의 부분 그림
plt.subplot(2, 2, 1) # 첫 번째 부분 그림
plt.plot(x, x)
plt.subplot(2, 2, 2) # 두 번째 부분 그림
plt.plot(x, -x)
plt.subplot(2, 2, 3) # 세 번째 부분 그림
plt.plot(x, x*x)
plt.subplot(2, 2, 4) # 네 번째 부분 그림
plt.plot(x, pow(x, 3))

plt.tight_layout() # 적절한 공간 배치
plt.show() # 그리기
```



# Matplotlib, matplotlib2.py 쉘에서 코딩

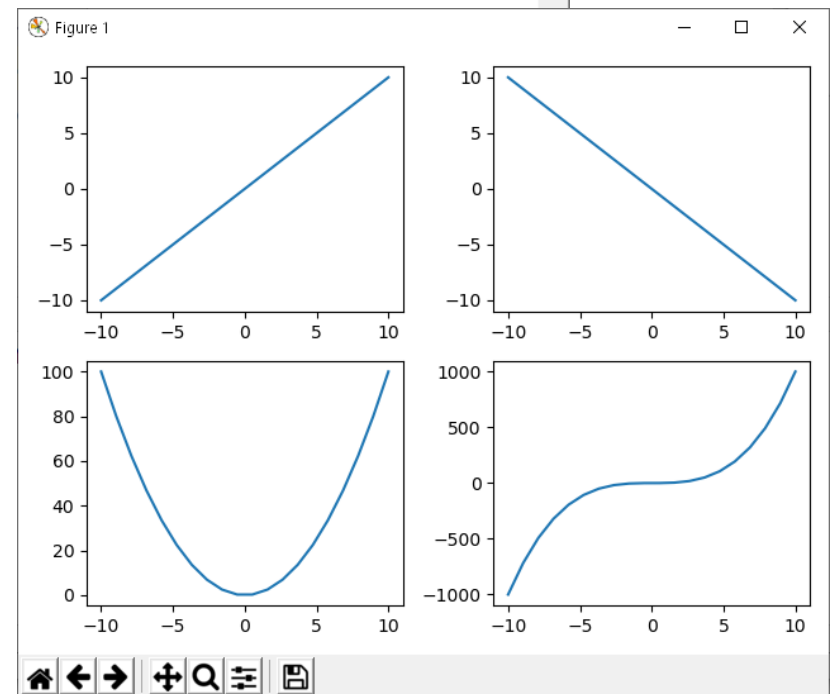
## • Python 실행

```

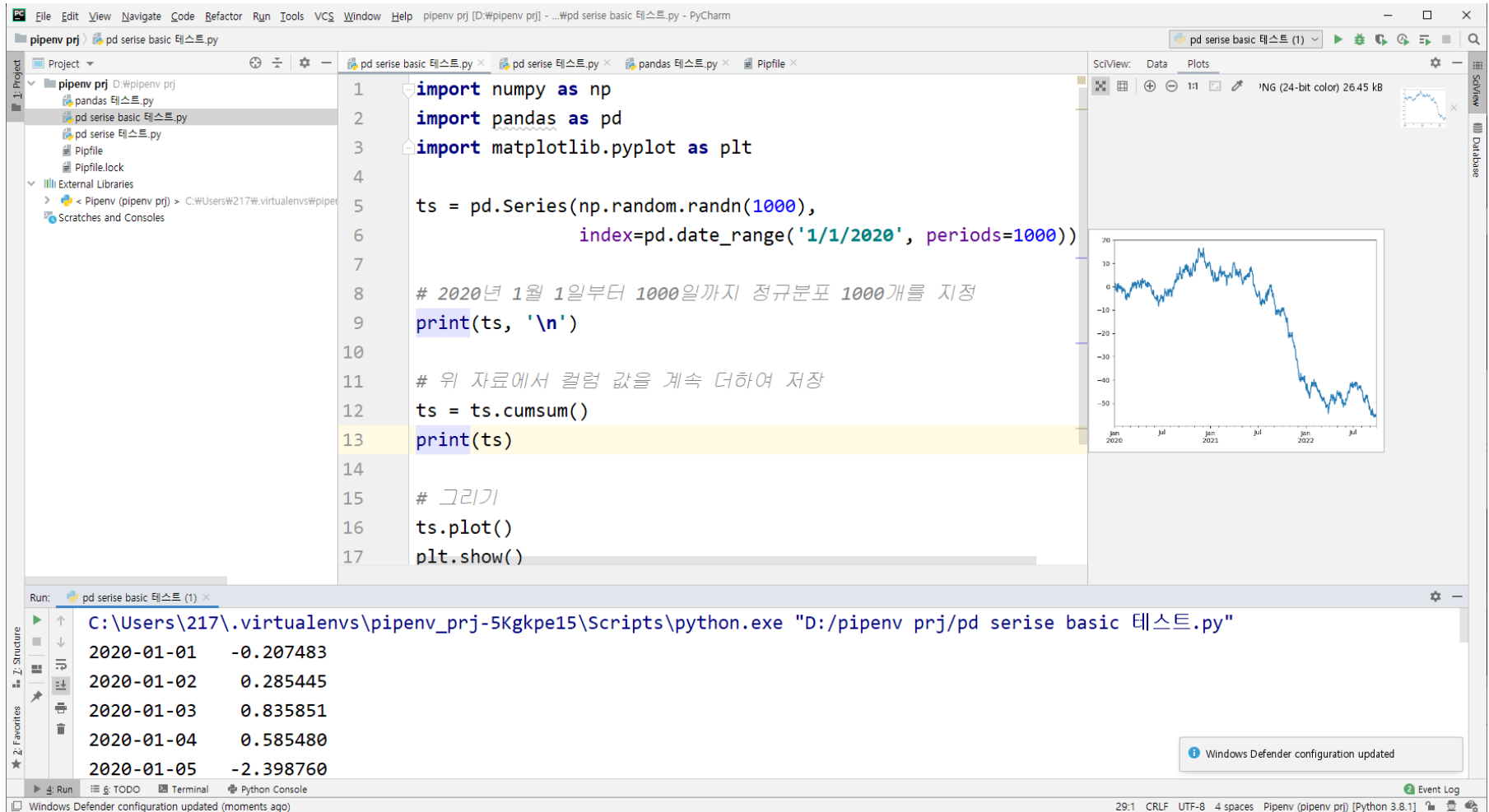
C:\Windows\System32\cmd.exe - pipenv shell - python

(penv-Snxa-AB5) D:\we\#penv>python
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 22:39:24) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy as np
>>> import matplotlib.pyplot as plt
>>> x = np.linspace(-10, 10, 20)
>>> plt.subplot(2, 2, 1)
<matplotlib.axes._subplots.AxesSubplot object at 0x037C3910>
>>> plt.plot(x, x)
[<matplotlib.lines.Line2D object at 0x10815340>]
>>> plt.subplot(2, 2, 2)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'plit' is not defined
>>> plt.subplot(2, 2, 2)
<matplotlib.axes._subplots.AxesSubplot object at 0x10806BB0>
>>> plt.plot(x, -x)
[<matplotlib.lines.Line2D object at 0x00DB9178>]
>>> plt.subplot(2, 2, 3)
<matplotlib.axes._subplots.AxesSubplot object at 0x00DB9AA8>
>>> plt.plot(x, x**2)
[<matplotlib.lines.Line2D object at 0x00DD2700>]
>>> plt.subplot(2, 2, 4)
<matplotlib.axes._subplots.AxesSubplot object at 0x00DD2DD8>
>>> plt.plot(x, pow(x, 3))
[<matplotlib.lines.Line2D object at 0x012D7388>]
>>> plt.show()

```



# Matplotlib, matplotlib3.py



The screenshot shows the PyCharm IDE with a project named 'pipenv prj'. The main editor displays a Python script named 'pd serise basic 테스트.py'. The script imports numpy, pandas, and matplotlib.pyplot. It creates a pandas Series 'ts' with random values over a date range from '1/1/2020' to '1000' periods. The script then calculates the cumulative sum of 'ts' and prints it. Finally, it plots the cumulative sum and shows the plot.

```

1 import numpy as np
2 import pandas as pd
3 import matplotlib.pyplot as plt
4
5 ts = pd.Series(np.random.randn(1000),
6               index=pd.date_range('1/1/2020', periods=1000))
7
8 # 2020년 1월 1일부터 1000일까지 정규분포 1000개를 지정
9 print(ts, '\n')
10
11 # 위 자료에서 컬럼 값을 계속 더하여 저장
12 ts = ts.cumsum()
13 print(ts)
14
15 # 그리기
16 ts.plot()
17 plt.show()

```

The right sidebar shows the 'SciView' panel with a 'Data' tab and a 'Plots' tab. The 'Plots' tab displays a line plot of the cumulative sum of the time series, showing a fluctuating upward trend over time.

The bottom panel shows the 'Run' output for the script 'pd serise basic 테스트 (1)'. The output displays the first five rows of the cumulative sum:

```

C:\Users\217\.virtualenvs\pipenv_prj-5Kgkpe15\Scripts\python.exe "D:/pipenv prj/pd serise basic 테스트.py"
2020-01-01    -0.207483
2020-01-02     0.285445
2020-01-03     0.835851
2020-01-04     0.585480
2020-01-05    -2.398760

```

The status bar at the bottom indicates the file encoding is CRLF, UTF-8, 4 spaces, and the Python version is 3.8.1. A notification for Windows Defender configuration update is also visible.

# Matplotlib, matplotlib.pyplot 이해

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

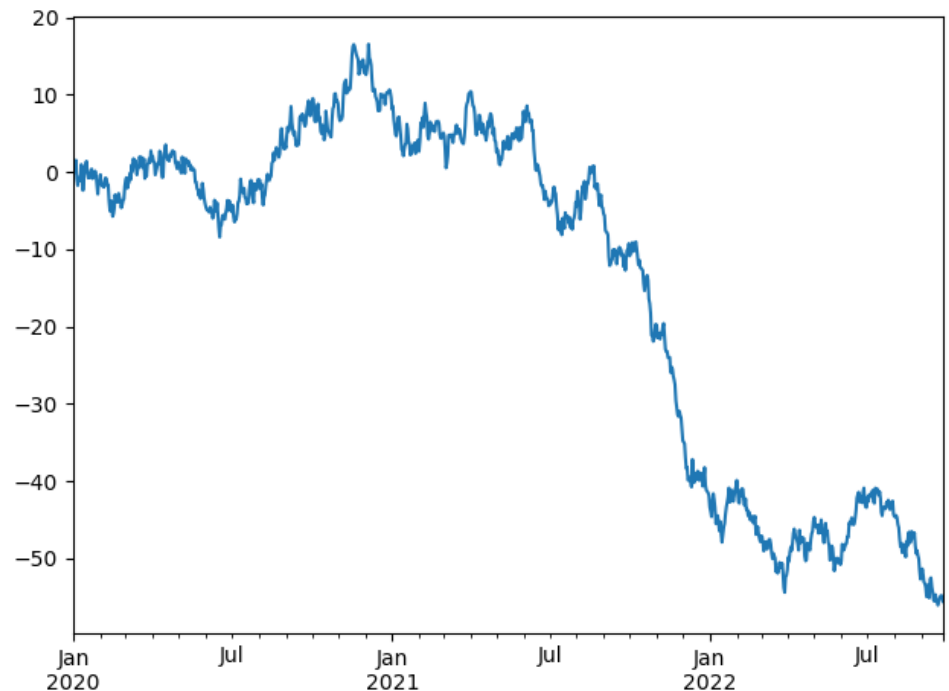
```
# Series는 1차원 데이터를 다루는 데 효과적인 자료구조
# value와 index의 형태를 지니는 Pandas의 자료 구조
# 별도의 인덱스 레이블을 지정하지 않으면 자동적으로
# 0부터 시작되는 디폴트 정수 인덱스를 사용
# 다음은 인덱스를 날짜 1000개로 지정, 자료 값은 난수 1000개
# 즉 2020년 1월 1일부터 1000일까지 정규분포 1000개를 지정
ts = pd.Series(np.random.randn(1000), index=pd.date_range('1/1/2020', periods=1000))
print(ts, '\n')
```

```
# 위 자료에서 컬럼 값을 계속 더하여 저장
ts = ts.cumsum()
print(ts)
```

```
# 그리기
ts.plot()
plt.show()
```

2020-01-01	-0.207483
2020-01-02	0.285445
2020-01-03	0.835851

2020-01-01	-0.207483
2020-01-02	0.077963
2020-01-03	0.913813



# Matplotlib, matplot4.py

The screenshot displays a PyCharm IDE with a project named 'pipenv prj'. The main editor shows a file named 'pd serise 테스트.py' with the following code:

```

1 import numpy as np
2 import pandas as pd
3 import matplotlib.pyplot as plt
4
5 df = pd.DataFrame(np.random.randn(1000, 4), columns=['A', 'B', 'C', 'D'],
6                  index=pd.date_range('1/1/2020', periods=1000))
7
8 df = df.cumsum()
9 print(df.head())
10
11 df.plot()
12 plt.show()

```

The right sidebar shows a 'SciView: Data Plots' panel with a line plot of the DataFrame's cumulative sum. The plot has four series labeled A, B, C, and D, showing a fluctuating upward trend over time from 2020 to 2022.

The bottom panel shows the output of the script, displaying the first 5 rows of the DataFrame:

	A	B	C	D
2020-01-01	-0.643711	-0.436099	-0.117916	0.293648
2020-01-02	-0.727072	0.838606	-0.512600	-0.821388
2020-01-03	-1.346217	0.621275	-0.941971	-1.334309
2020-01-04	0.425535	0.400747	0.695204	-2.376244
2020-01-05	0.022072	-0.085414	1.555287	-4.084275

The output also indicates 'Process finished with exit code 0'.

The bottom status bar shows the file encoding as UTF-8, 4 spaces, and the Python version as 3.8.1.



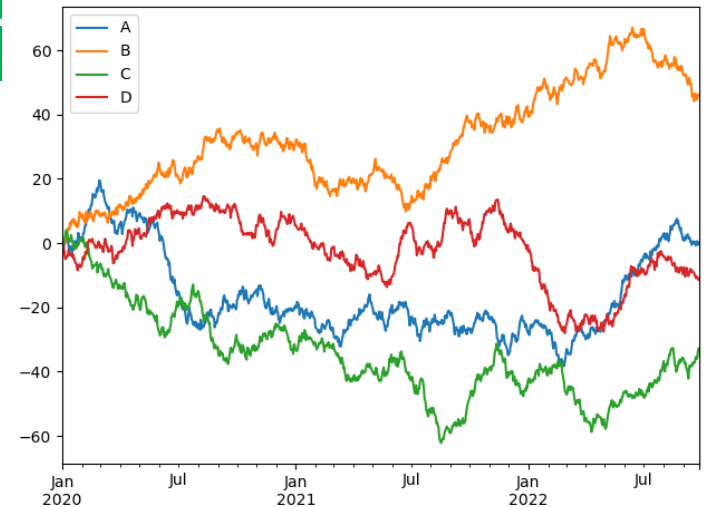
# Matplotlib, matplotlib3.py 0|하

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```
df = pd.DataFrame(np.random.randn(1000, 4), columns=['A', 'B', 'C', 'D'],
                  index=pd.date_range('1/1/2020', periods=1000))
```

```
df = df.cumsum()
print(df.head())
```

```
df.plot()
plt.show()
```



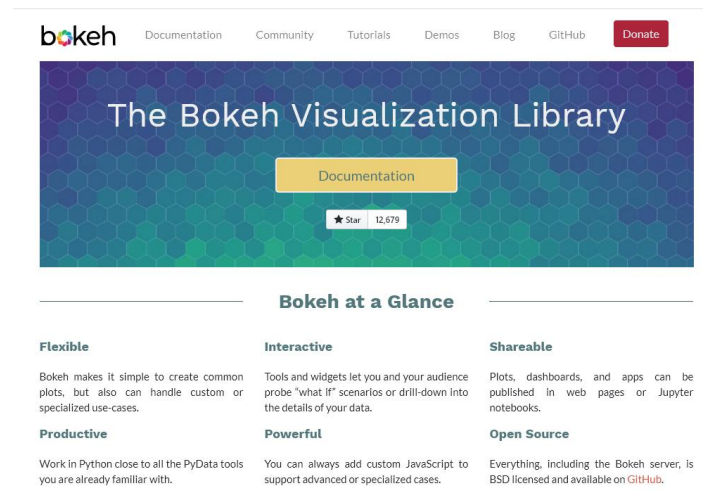
	A	B	C	D
2020-01-01	-0.643711	-0.436099	-0.117916	0.293648
2020-01-02	-0.727072	0.838606	-0.512600	-0.821388
2020-01-03	-1.346217	0.621275	-0.941971	-1.334309
2020-01-04	0.425535	0.400747	0.695204	-2.376244
2020-01-05	0.022072	-0.085414	1.555287	-4.084275

# 파이썬 프로젝트에서 데이터 시각화를 위한 준비 bokeh

# Bokeh 개요

## • Bokeh.org

- 최신 웹 브라우저를 위한 대화 형 시각화 라이브러리
  - 다목적 그래픽의 우아하고 간결한 구성을 제공
  - 대용량 또는 스트리밍 데이터 세트에 대한 고성능 대화식 기능을 제공
- 대화 형 플롯, 대시 보드 및 데이터 응용 프로그램을 빠르고 쉽게 만들고 싶은 사람에게 적합
- Bokeh를 사용하여 시각화를 시작하려면
  - **사용 설명서로 시작**
    - [https://docs.bokeh.org/en/latest/docs/user\\_guide/quickstart.html#userguide-quickstart](https://docs.bokeh.org/en/latest/docs/user_guide/quickstart.html#userguide-quickstart)



# 보케 가이드 페이지

The screenshot shows the Bokeh 1.4.0 documentation page. The browser tabs include 'Data Visualization with Bokeh in ...' and '빠른 시작 — Bokeh 1.4.0 문서'. The address bar shows the URL 'docs.bokeh.org/en/latest/docs/user\_guide/quickstart.html#userguide-quickstart'. The page header includes the Bokeh logo, version '1.4.0', and navigation links like '설치', '사용자 설명서', '갤러리', '참고', '개발자', '자료', '지도 시간', and '커뮤니티'. A search bar is present with the text 'Search the docs ...'. The left sidebar lists various topics under '빠른 시작' (Getting Started), including '핵심 개념 정의', '기본 플리프를 사용한 플로팅', '데이터 제공', '레이아웃 만들기', '범주형 데이터 처리', '네트워크 그래프 시작화', '지리 데이터 매핑', '플롯 도구 구성', '비주얼 속성 스타일링', '주석 추가', '상호작용 만들기', 'Bokeh 서버 실행', '노트북에서 작업', '플롯 내보내기', 'Bokeh 콘텐츠 임베드', '명령 줄 사용', '보케 확장', 'WebGL로 가속화', '다른 도구 살펴보기', and 'JavaScript로 개발'. The main content area has a '노트' (Note) box stating that the pip method does not install examples and that users should check out the examples directory. Below this is the '시작하기' (Getting Started) section, which explains that Bokeh provides many features and that this section covers basic usage and workflow. It mentions that several examples will be shown. The '기본 Python 목록에서 데이터를 줌, 이동, 저장 및 기타 도구를 포함하는 선 그림으로 표시하는 것은 간단하고 간단합니다.' (Getting data from the basic Python list, moving, saving, and other tools, and displaying it as a line plot is simple and simple.) is highlighted. A code block shows the following Python code:

```
from bokeh.plotting import figure, output_file, show

# prepare some data
x = [1, 2, 3, 4, 5]
y = [6, 7, 2, 4, 5]

# output to static HTML file
output_file("lines.html")

# create a new plot with a title and axis labels
p = figure(title="simple line example", x_axis_label='x', y_axis_label='y')

# add a Line renderer with Legend and line thickness
p.line(x, y, legend="Temp.", line_width=2)

# show the results
show(p)
```

Below the code, a plot titled 'simple line example' is shown. The x-axis is labeled 'x' and the y-axis is labeled 'y'. The plot displays a line with data points, and a legend in the bottom right corner indicates the line is 'Temp.'.

# Settings... 에서 모듈 추가하기

- 원 창
  - 오른쪽 +
- 모듈 bokeh 설치
  - 하단 Install Package

Available Packages

Search: bokeh

bokeh

bokeh-catplot

bokeh-metaplot

bokeh-plot

bokeh-wordcloud2

bokeh\_image\_explore

bokehheat

ebokch

Description

Interactive plots and applications in the browser from Python

Version

1.4.0

Author

Bokeh Team

<mailto:info@bokeh.org>

<http://github.com/bokeh/bokeh>

설치가 다소 느린 단점

Specify version 1.4.0

Options

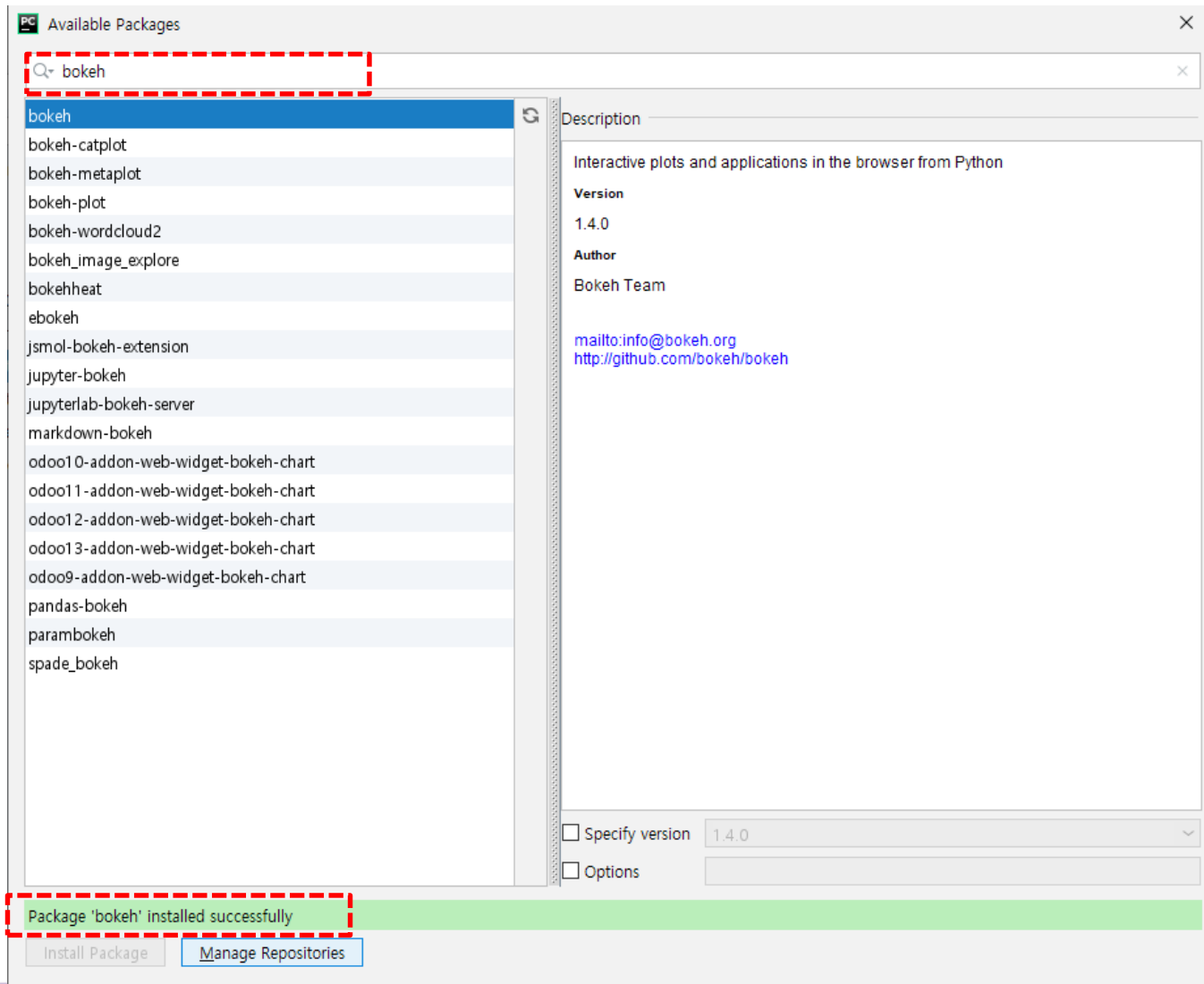
Project: numpy prj > Project Interpreter

Project Interpreter: Python 3.7 (numpy prj) D:\numpy prj\venv\Scripts\python.exe

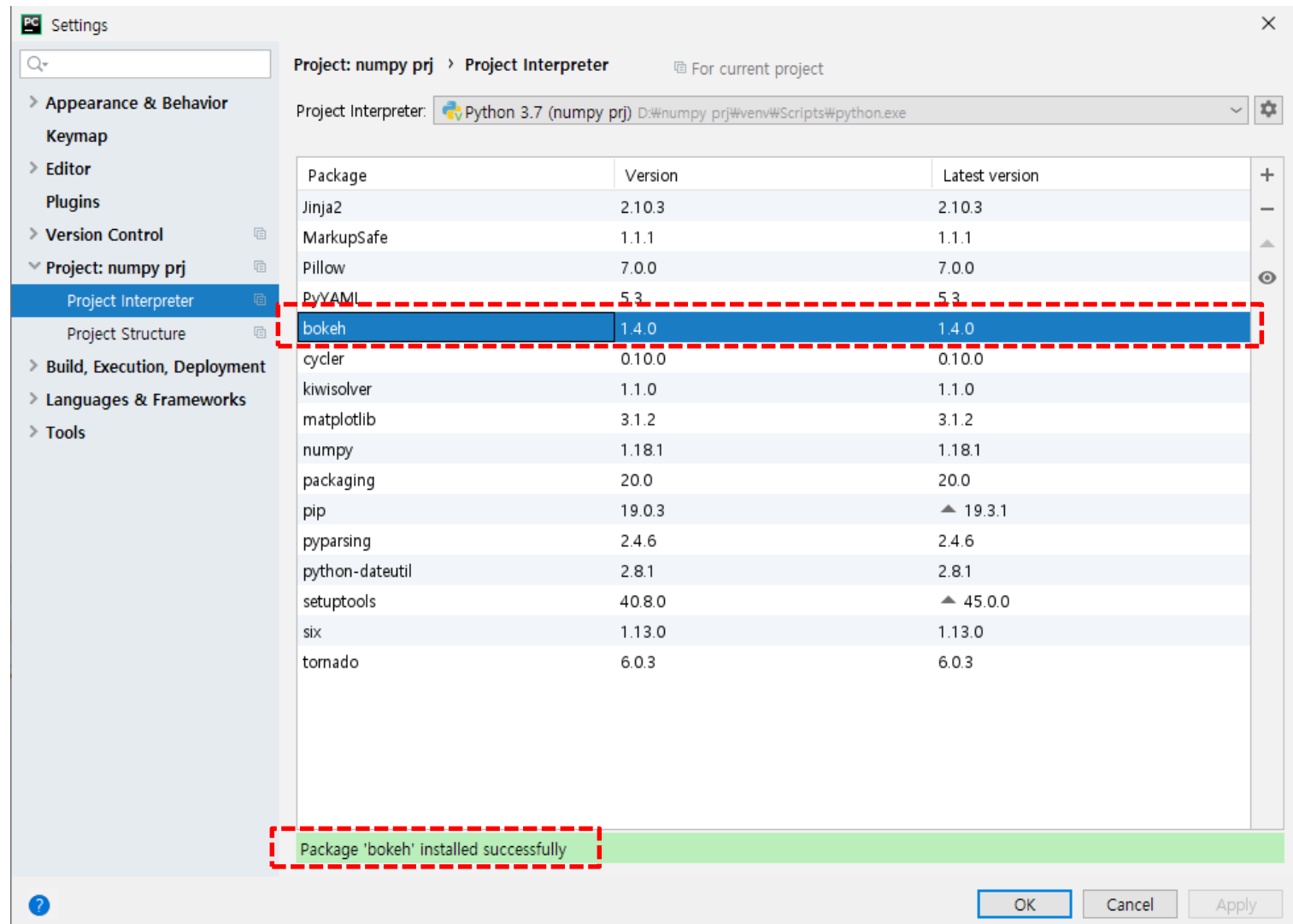
Package	Version	Latest version
cycler	0.10.0	0.10.0
kiwisolver	1.1.0	1.1.0
matplotlib	3.1.2	3.1.2
numpy	1.18.1	1.18.1
pip	19.0.3	▲ 19.3.1
pyparsing	2.4.6	2.4.6
python-dateutil	2.8.1	2.8.1
setuptools	40.8.0	▲ 45.0.0
six	1.13.0	1.13.0

OK Cancel Apply

# 모듈 bokeh 설치 화면



# Settings 화면



# Bokeh 터미널 설치도 물론 가능

- 터미널에서 설치
  - pip install bokeh

- 확인
  - pip list
  - pip show bokeh

```
(venv) D:\numpy prj>pip list
Package           Version
-----
bokeh              1.4.0
cycler             0.10.0
Jinja2             2.10.3
kiwisolver         1.1.0
MarkupSafe         1.1.1
matplotlib         3.1.2
numpy              1.18.1
packaging          20.0
pandas             0.25.3
Pillow             7.0.0
pip                19.0.3
pyparsing          2.4.6
python-dateutil    2.8.1
pytz               2019.3
PyYAML             5.3
setuptools         40.8.0
six                1.13.0
tornado            6.0.3
```



# Bokeh, bksample1.py

- 결과는 html 파일

```

1  from bokeh.plotting import figure, show
2
3  # from bokeh.plotting import output_file
4  # 출력파일 지정, 없으면 소스파일과 동일한 html 파일로 생성
5  # output_file("line.html")
6
7  p = figure()
8  # p = figure(plot_width=400, plot_height=400)
9
10 # add a line renderer
11 p.line([1, 2, 3, 4, 5], [6, 7, 2, 4, 5])
12 # p.line([1, 2, 3, 4, 5], [6, 7, 2, 4, 5], line_width=2)
13
14 show(p)
  
```

# Bokeh, bksample1.py 이해

- 소스 파일과 동일한 폴더에 html 파일로 생성

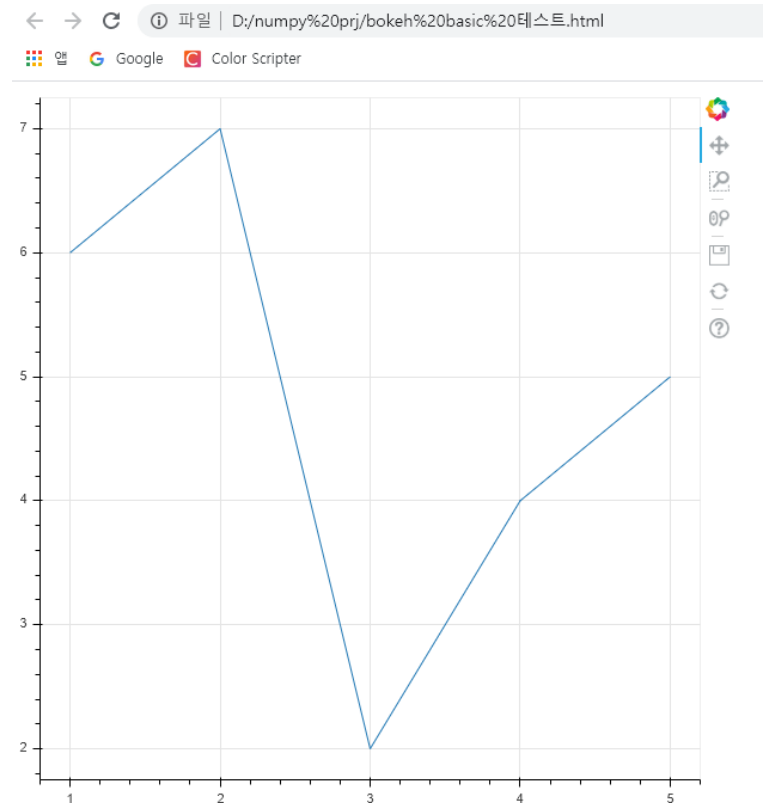
```
from bokeh.plotting import figure, show
```

```
p = figure()
```

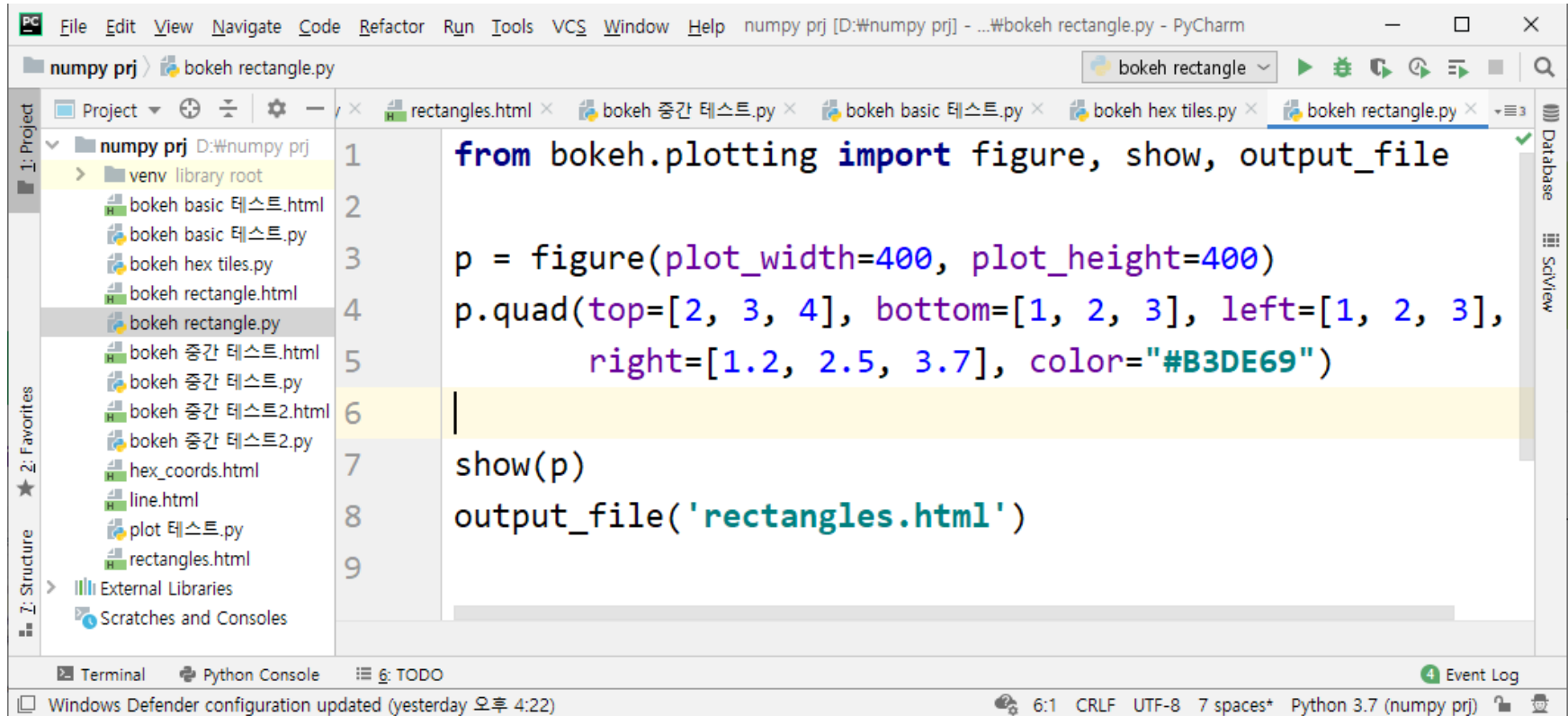
```
# add a line renderer
```

```
p.line([1, 2, 3, 4, 5], [6, 7, 2, 4, 5])
```

```
show(p)
```



# Bokeh, bksample2.py



```
1 from bokeh.plotting import figure, show, output_file
2
3 p = figure(plot_width=400, plot_height=400)
4 p.quad(top=[2, 3, 4], bottom=[1, 2, 3], left=[1, 2, 3],
5        right=[1.2, 2.5, 3.7], color="#B3DE69")
6
7 show(p)
8 output_file('rectangles.html')
9
```

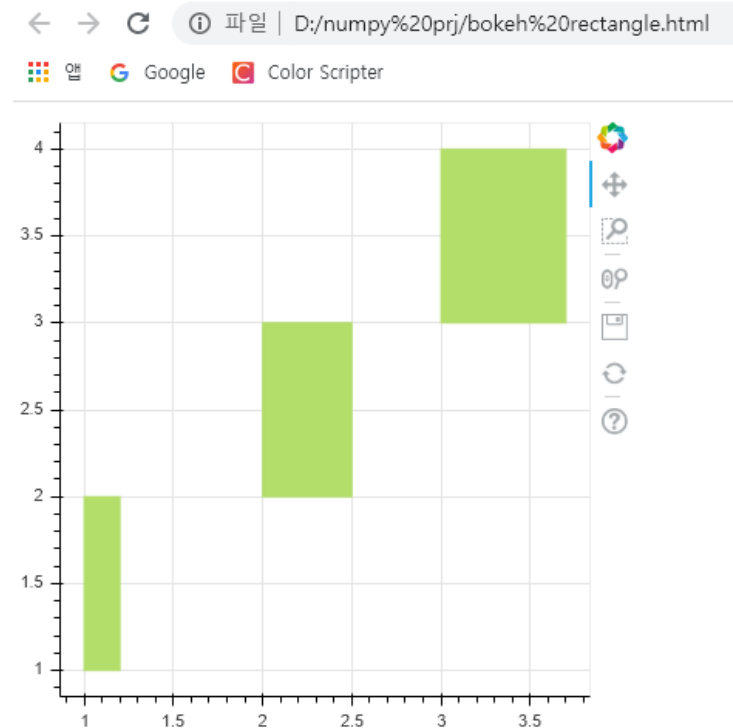
The screenshot shows the PyCharm IDE interface. The top toolbar includes icons for running, debugging, and other development actions. The left sidebar shows the project structure for 'numpy prj', with a 'venv' directory containing various files. The main editor window displays the code for 'bksample2.py'. The code imports 'figure', 'show', and 'output\_file' from 'bokeh.plotting'. It then creates a figure with a width of 400 and a height of 400. A rectangle plot is created using the 'quad' method, with the following parameters: 'top'=[2, 3, 4], 'bottom'=[1, 2, 3], 'left'=[1, 2, 3], 'right'=[1.2, 2.5, 3.7], and 'color'='#B3DE69'. The figure is then displayed using 'show(p)' and the output is saved to a file named 'rectangles.html' using 'output\_file('rectangles.html')'. The bottom status bar shows the file encoding as UTF-8, 7 spaces, and Python 3.7 (numpy prj).

# Bokeh, bksample2.py 이해

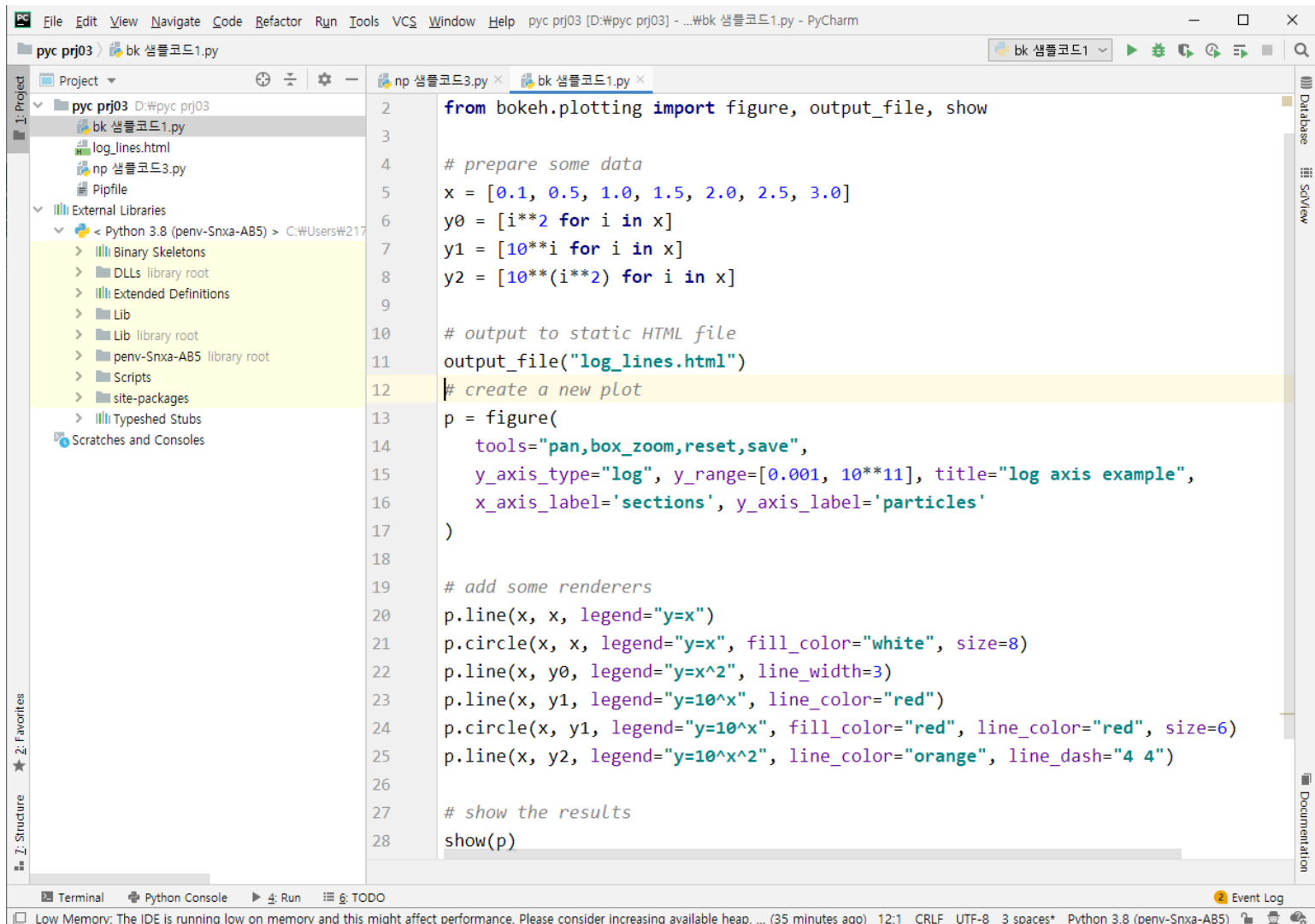
```
from bokeh.plotting import figure, show, output_file
```

```
p = figure(plot_width=400, plot_height=400)
p.quad(top=[2, 3, 4], bottom=[1, 2, 3], left=[1, 2, 3],
        right=[1.2, 2.5, 3.7], color="#B3DE69")
```

```
show(p)
output_file('rectangles.html')
```



# Bokeh, bksample3.py



```

1  from bokeh.plotting import figure, output_file, show
2
3
4  # prepare some data
5  x = [0.1, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0]
6  y0 = [i**2 for i in x]
7  y1 = [10**i for i in x]
8  y2 = [10**(i**2) for i in x]
9
10 # output to static HTML file
11 output_file("log_lines.html")
12 # create a new plot
13 p = figure(
14     tools="pan,box_zoom,reset,save",
15     y_axis_type="log", y_range=[0.001, 10**11], title="log axis example",
16     x_axis_label='sections', y_axis_label='particles'
17 )
18
19 # add some renderers
20 p.line(x, x, legend="y=x")
21 p.circle(x, x, legend="y=x", fill_color="white", size=8)
22 p.line(x, y0, legend="y=x^2", line_width=3)
23 p.line(x, y1, legend="y=10^x", line_color="red")
24 p.circle(x, y1, legend="y=10^x", fill_color="red", line_color="red", size=6)
25 p.line(x, y2, legend="y=10^x^2", line_color="orange", line_dash="4 4")
26
27 # show the results
28 show(p)

```

Low Memory: The IDE is running low on memory and this might affect performance. Please consider increasing available heap. ... (35 minutes ago) 12:1 CRLF UTF-8 3 spaces\* Python 3.8 (penv-Snxa-AB5)

# Bokeh, bksample3.py 이해

```
#
https://docs.bokeh.org/en/latest/docs/user\_guide/quickstart.html#userguide-quickstart
```

```
from bokeh.plotting import figure, output_file, show
```

```
# prepare some data
```

```
x = [0.1, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0]
```

```
y0 = [i**2 for i in x]
```

```
y1 = [10**i for i in x]
```

```
y2 = [10**(i**2) for i in x]
```

```
# output to static HTML file
```

```
output_file("log_lines.html")
```

```
# create a new plot
```

```
p = figure(
    tools="pan,box_zoom,reset,save",
    y_axis_type="log", y_range=[0.001, 10**11],
    title="log axis example",
    x_axis_label='sections', y_axis_label='particles'
)
```

```
# add some renderers
```

```
p.line(x, x, legend="y=x")
```

```
p.circle(x, x, legend="y=x", fill_color="white", size=8)
```

```
p.line(x, y0, legend="y=x^2", line_width=3)
```

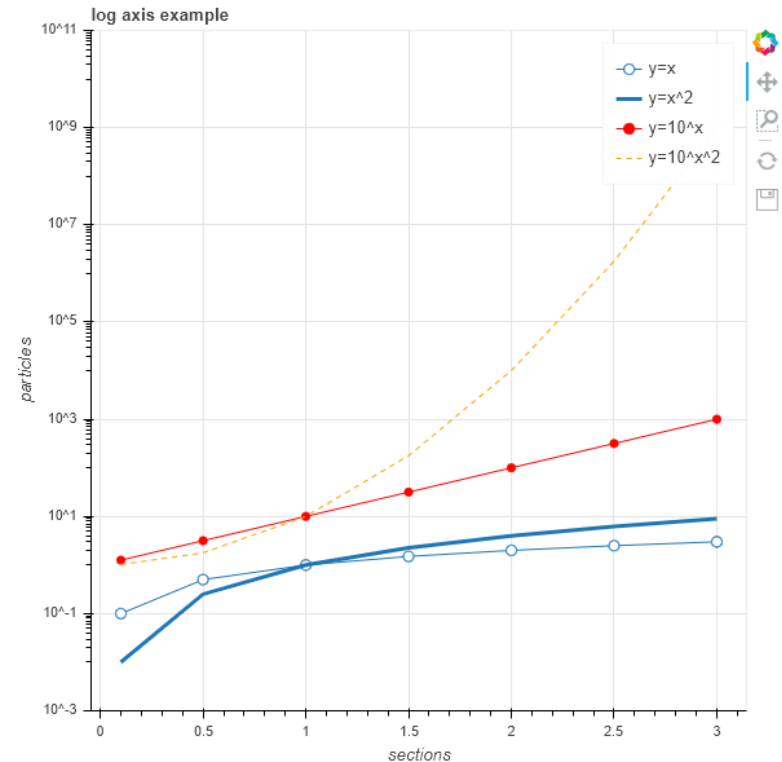
```
p.line(x, y1, legend="y=10^x", line_color="red")
```

```
p.circle(x, y1, legend="y=10^x", fill_color="red", line_color="red", size=6)
```

```
p.line(x, y2, legend="y=10^x^2", line_color="orange", line_dash=[4, 4])
```

```
# show the results
```

```
show(p)
```



# Bokeh, bksample4.py 이해

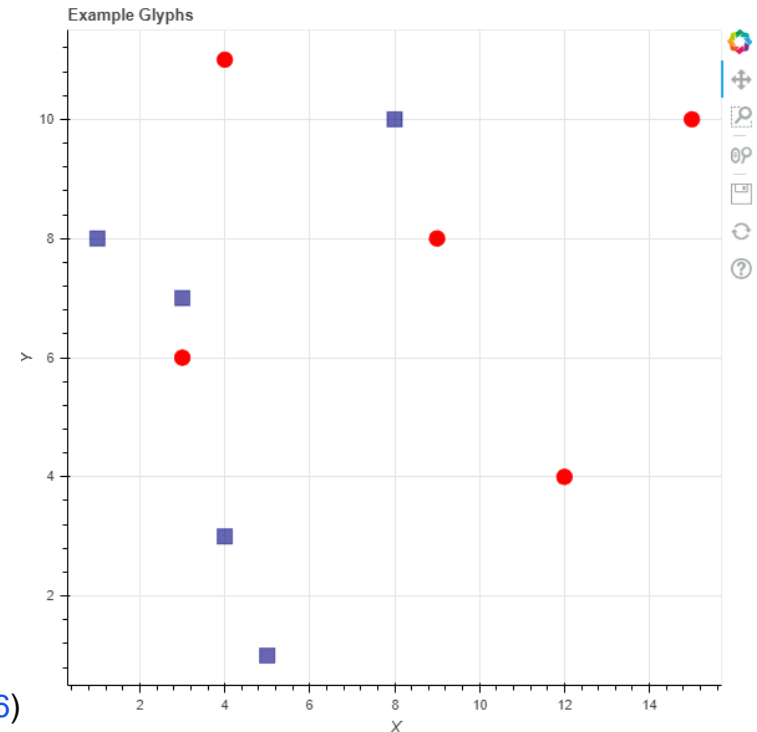
```
# bokeh basics
from bokeh.plotting import figure
from bokeh.io import show
# from bokeh.io import show, output_notebook

# Create a blank figure with labels
p = figure(plot_width = 600, plot_height = 600,
           title = 'Example Glyphs',
           x_axis_label = 'X', y_axis_label = 'Y')

# Example data
squares_x = [1, 3, 4, 5, 8]
squares_y = [8, 7, 3, 1, 10]
circles_x = [9, 12, 4, 3, 15]
circles_y = [8, 4, 11, 6, 10]

# Add squares glyph
p.square(squares_x, squares_y, size = 12, color = 'navy', alpha = 0.6)
# Add circle glyph
p.circle(circles_x, circles_y, size = 12, color = 'red')

# Set to output the plot in the notebook
# output_notebook()
# Show the plot
show(p)
```

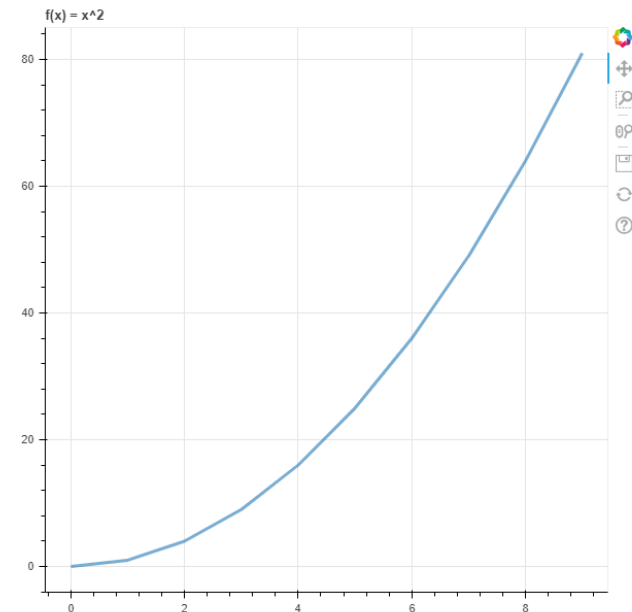


# Bokeh, bksample5.py 이해

```
from bokeh.io import show
from bokeh.plotting import figure
from bokeh.models import ColumnDataSource
```

```
x_values = range(10)
y_values = [x ** 2 for x in x_values]
data_source = ColumnDataSource(data=dict(x=x_values, y=y_values))
```

```
plot = figure(title = 'f(x) = x^2')
plot.line('x', 'y', source = data_source, line_width=3, line_alpha=0.6)
show(plot)
```





# Bokeh, bksample6.py 이해

```
import numpy as np

from bokeh.io import output_file, show
from bokeh.plotting import figure
from bokeh.util.hex import axial_to_cartesian

output_file("hex_coords.html")

q = np.array([0, 0, 0, -1, -1, 1, 1])
r = np.array([0, -1, 1, 0, 1, -1, 0])

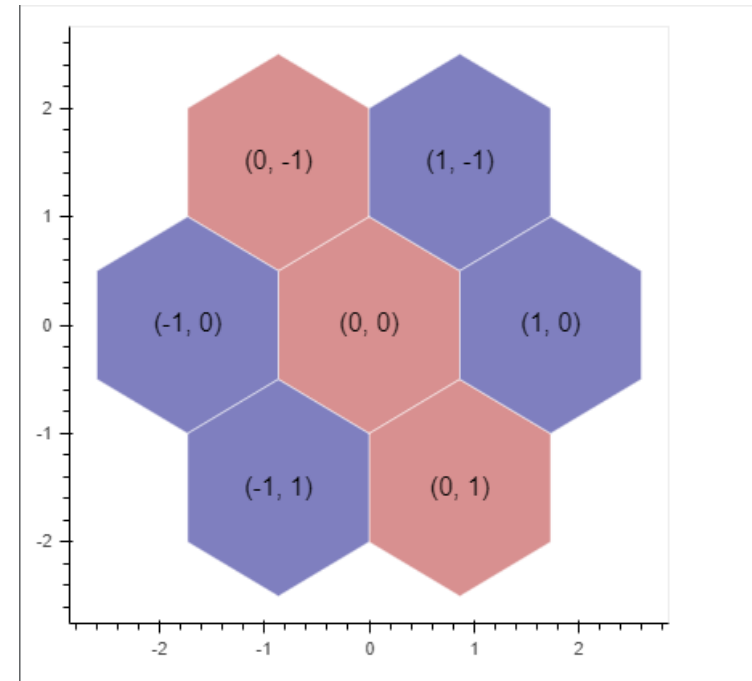
p = figure(plot_width=400, plot_height=400, toolbar_location=None)
p.grid.visible = False

p.hex_tile(q, r, size=1, fill_color=["firebrick"]*3 + ["navy"]*4,
          line_color="white", alpha=0.5)

x, y = axial_to_cartesian(q, r, 1, "pointytop")

p.text(x, y, text=["(%d, %d)" % (q,r) for (q, r) in zip(q, r)],
      text_baseline="middle", text_align="center")

show(p)
```



# Bokeh, bksample7.py 이해

```
import numpy as np

from bokeh.plotting import figure, output_file, show

# prepare some data
N = 4000
x = np.random.random(size=N) * 100
y = np.random.random(size=N) * 100
radii = np.random.random(size=N) * 1.5
colors = [
    "#%02x%02x%02x" % (int(r), int(g), 150) for r, g in zip(50+2*x, 30+2*y)
]

# output to static HTML file (with CDN resources)
output_file("color_scatter.html", title="color_scatter.py example", mode="cdn")

TOOLS = "crosshair,pan,wheel_zoom,box_zoom,reset,box_select,lasso_select"

# create a new plot with the tools above, and explicit ranges
p = figure(tools=TOOLS, x_range=(0, 100), y_range=(0, 100))

# add a circle renderer with vectorized colors and sizes
p.circle(x, y, radius=radii, fill_color=colors, fill_alpha=0.6, line_color=None)

# show the results
show(p)
```

