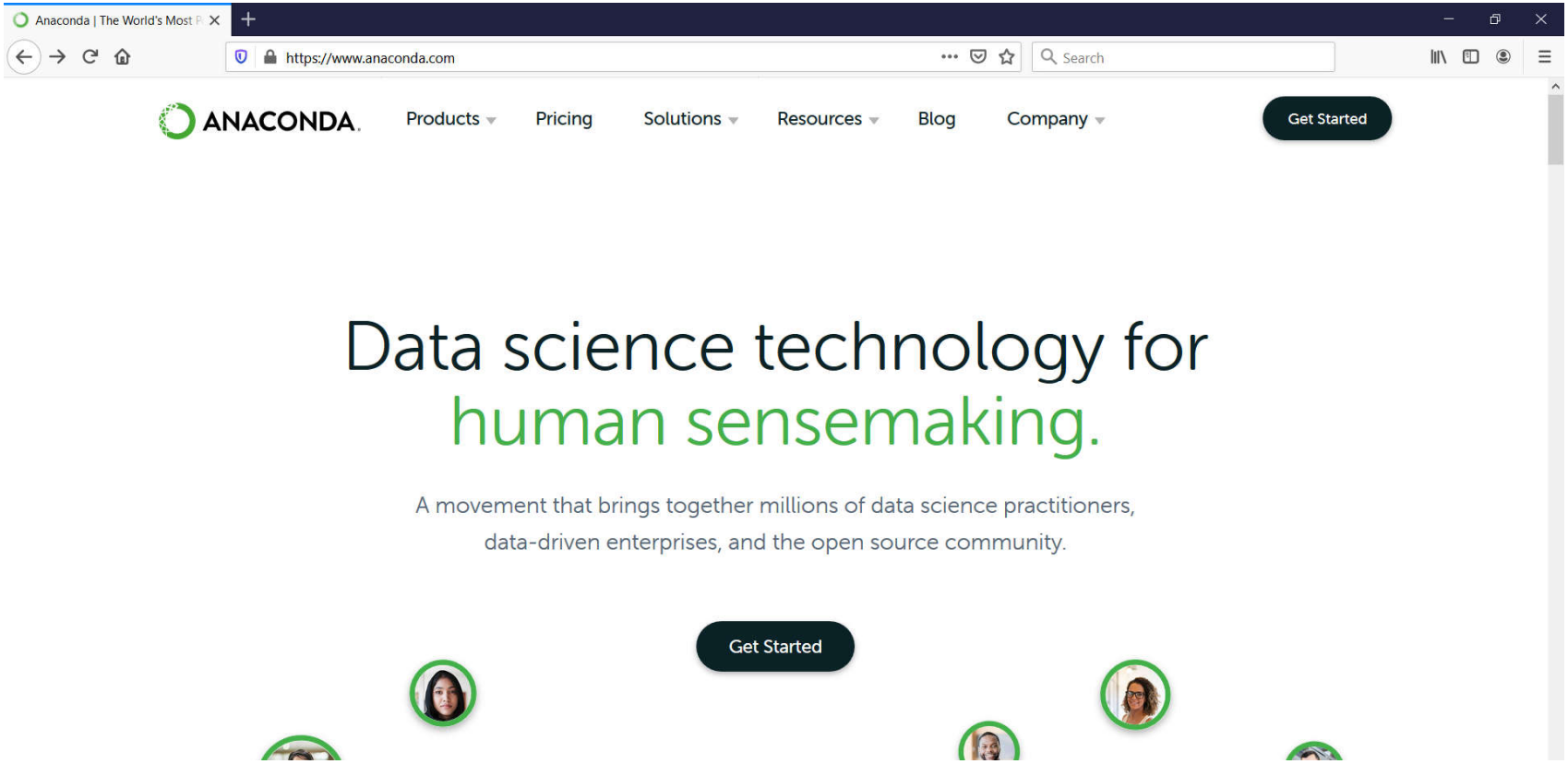
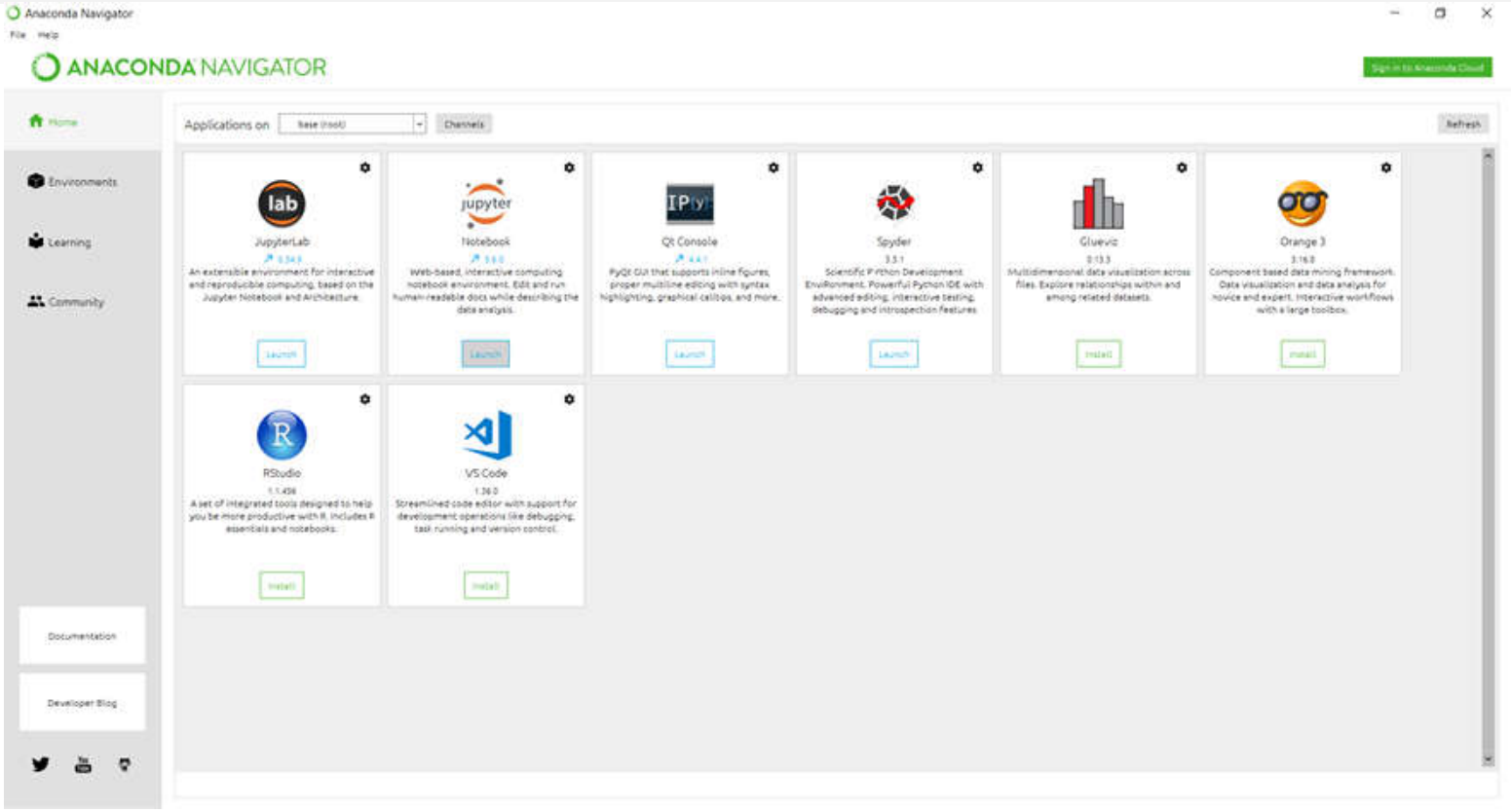
Pandas

Install Anaconda



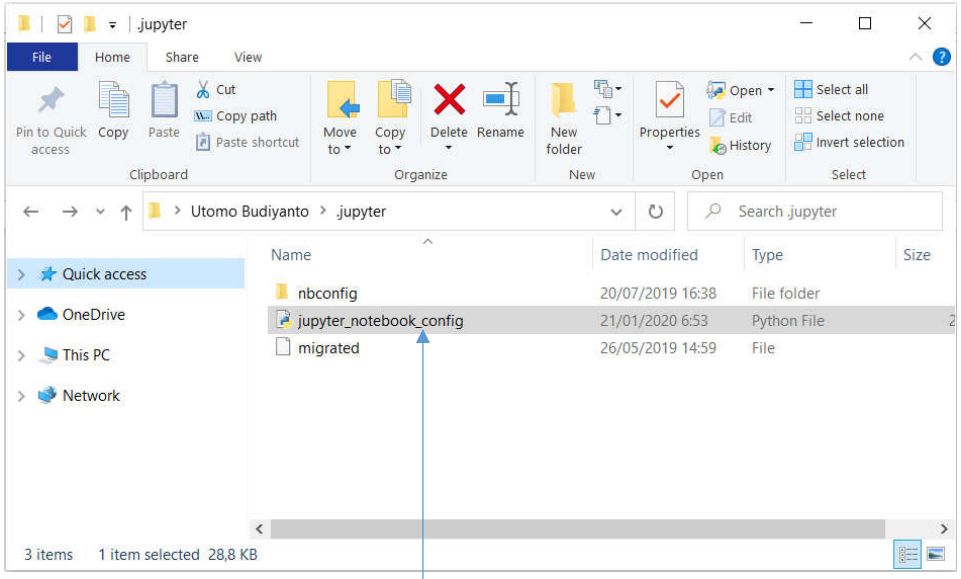
Running Anaconda

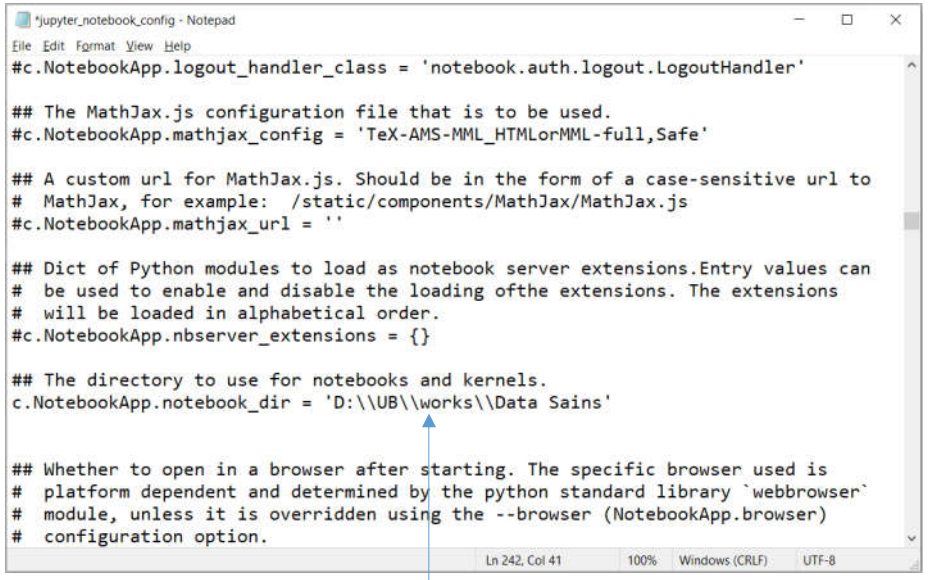


Running Jupyter Notebook

Untuk mengubah default folder: (optional)

- Folder (.jupyter) terletak di dalam folder Users
- misal: C:\Users\Utomo Budiyanto\.jupyter



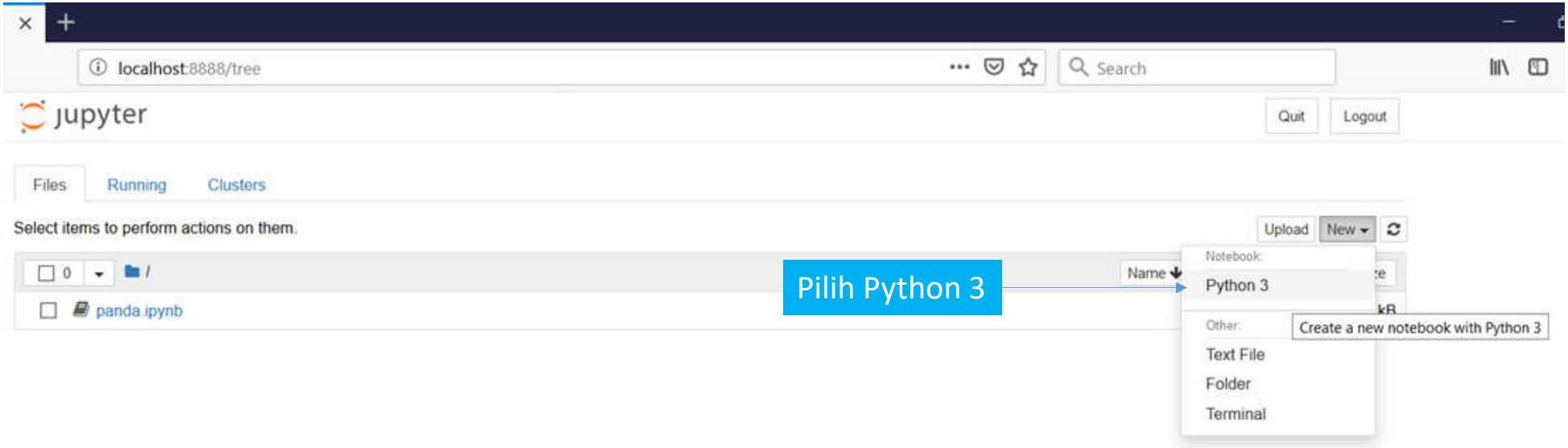


Nama file: jupyter_notebook_config

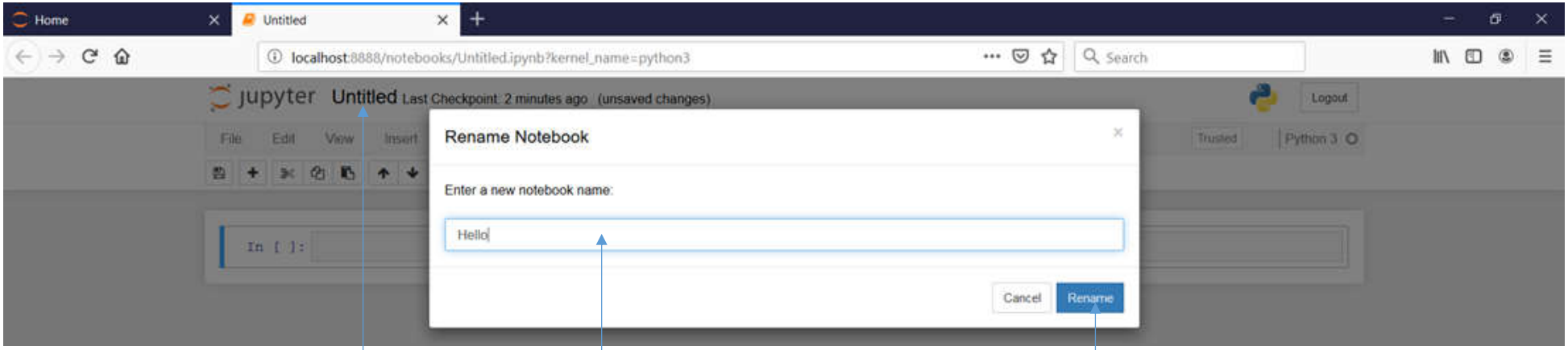
Ganti Folder untuk menyimpan file Jupyter Notebook (.ipynb)

Running Jupyter Notebook

Untuk mengubah default folder:
- ada di file jupyter_notebook_config
- Ubah



Rename Notebook

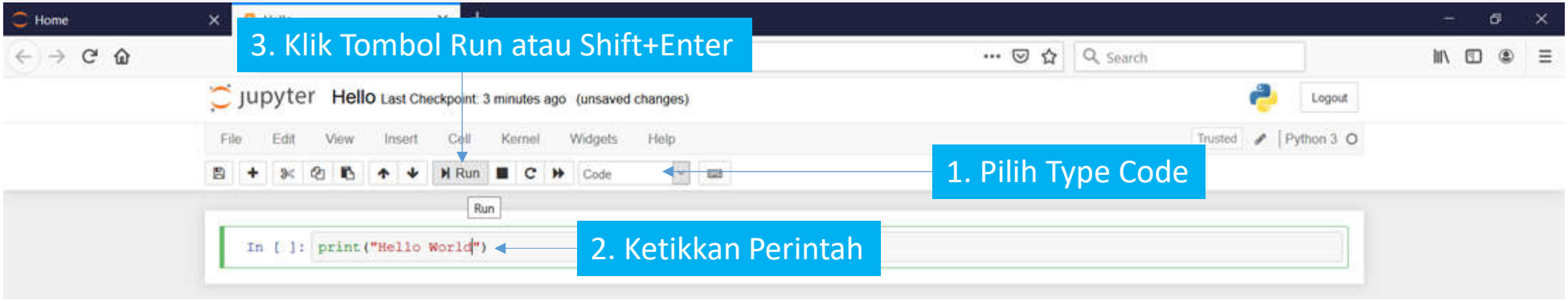


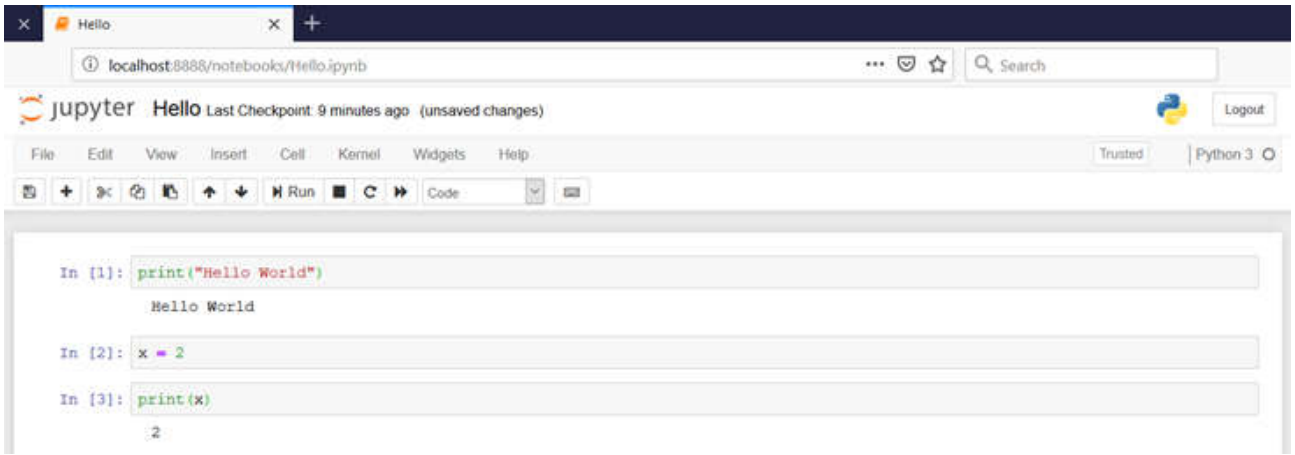
1. Double Klik Untitled

2. Input nama file

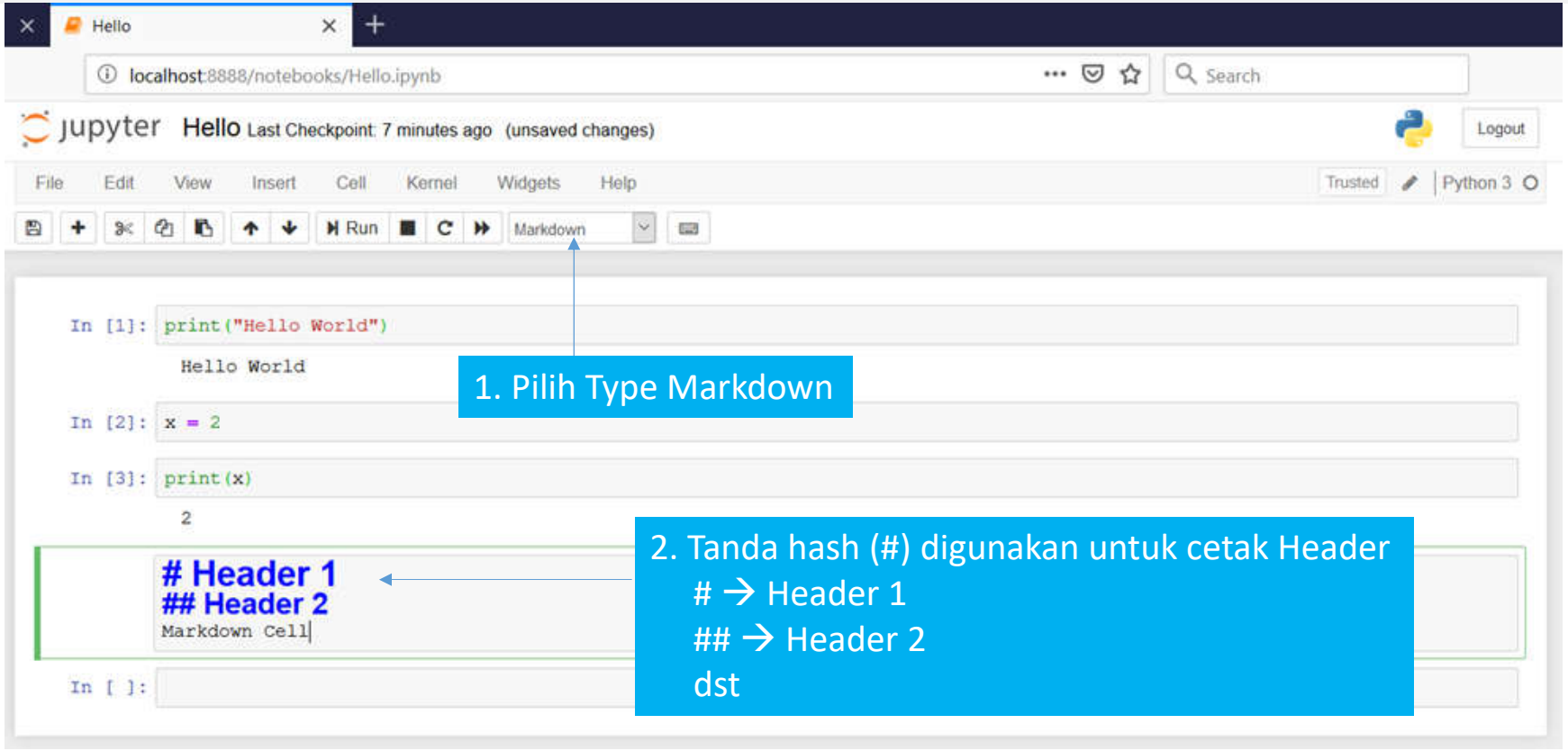
3. Klik Rename

Code Cell

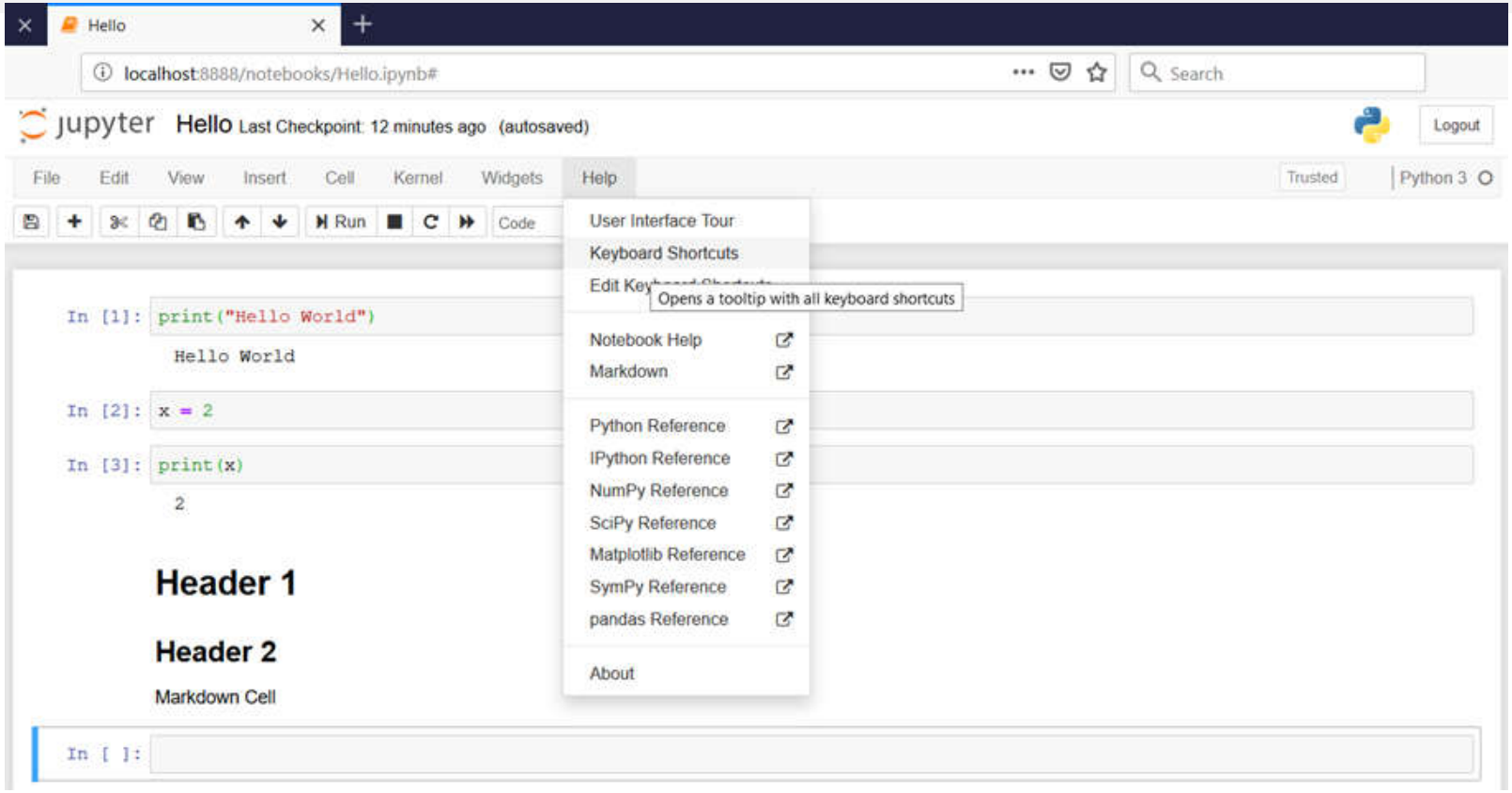

3. Klik Tombol Run atau Shift+Enter
1. Pilih Type Code
2. Ketikkan Perintah
jupyter Hello Last Checkpoint: 3 minutes ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted Python 3
Run
Code
In [ ]: print("Hello World")


Hello
localhost:8888/notebooks/Hello.ipynb
jupyter Hello Last Checkpoint: 9 minutes ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted Python 3
Run
Code
In [1]: print("Hello World")
Hello World
In [2]: x = 2
In [3]: print(x)
2

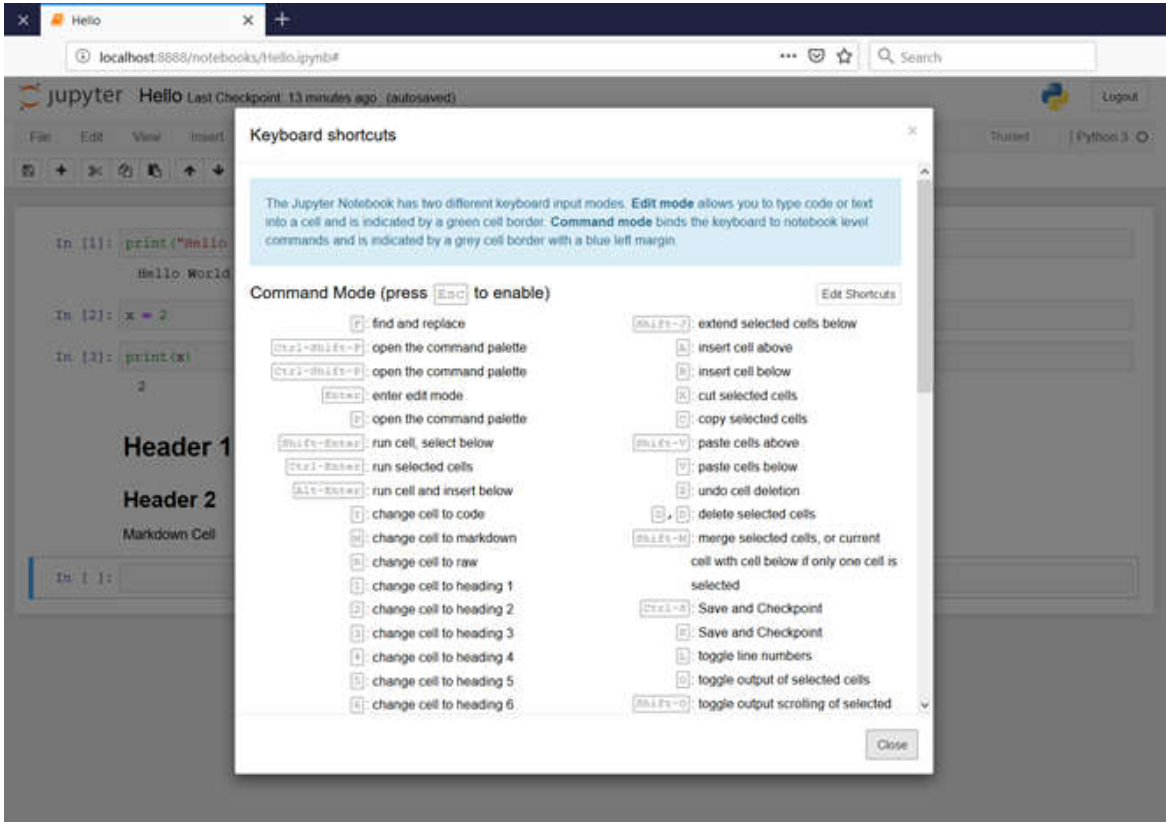
Markdown Cell



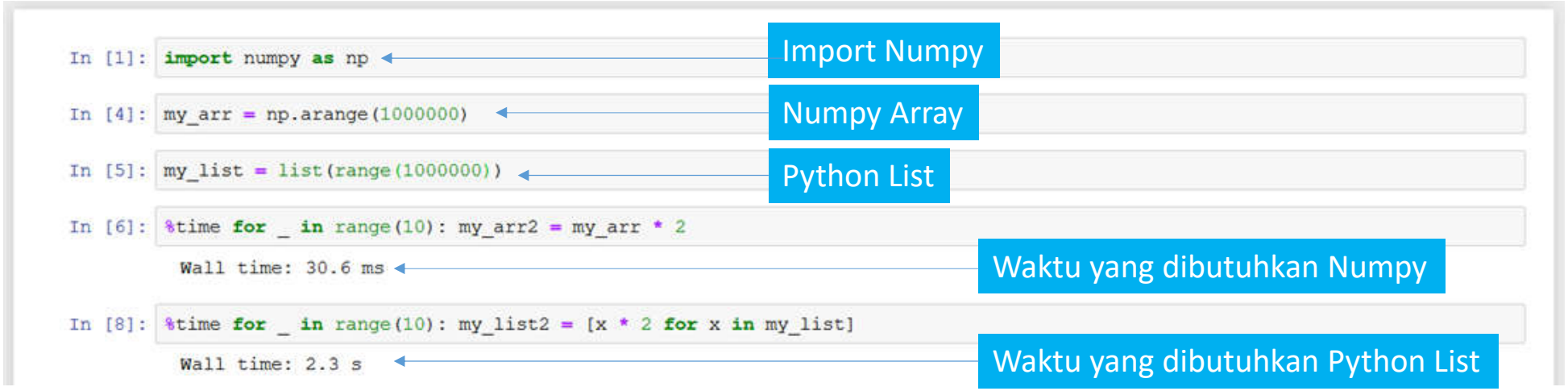
Menampilkan Keyboard Shortcuts



Jendela Bantuan Keyboard Shortcuts



Numpy Array vs Python List



Panda Introduction

- Contains data structures and data manipulation tools designed to make data cleaning and analysis fast and easy in Python
- Pandas is often used in tandem with numerical computing tools like NumPy and SciPy, analytical libraries like statsmodels and scikit-learn, and data visualization libraries like matplotlib.
- Pandas adopts significant parts of NumPy's idiomatic style of array-based computing, especially array-based functions and a preference for data processing without for loops.
- While pandas adopts many coding idioms from NumPy, the biggest difference is that pandas is designed for working with **tabular or heterogeneous** data. NumPy, by contrast, is best suited for working with **homogeneous** numerical array data.

Pandas Data Structure

- Series
  - A Series is a one-dimensional array-like object containing a sequence of values (of similar types to NumPy types) and an associated array of data labels, called its index.
- DataFrame
  - A DataFrame represents a rectangular table of data and contains an ordered collection of columns, each of which can be a different value type (numeric, string, boolean, etc.).
  - The DataFrame has both a row and column index; it can be thought of as a dict of Series all sharing the same index.
  - Under the hood, the data is stored as one or more two-dimensional blocks rather than a list, dict, or some other collection of one-dimensional arrays.

Series

Import Panda

```
In [1]: import pandas as pd
```

Series Data Structure

```
In [2]: x = pd.Series([10, -4, 72, 8])

In [3]: x

Out[3]: 0    10
        1    -4
        2    72
        3     8
        dtype: int64
```

Values

```
In [4]: x.values

Out[4]: array([10, -4, 72,  8], dtype=int64)
```

Index

```
In [5]: x.index

Out[5]: RangeIndex(start=0, stop=4, step=1)
```

```
In [6]: y = pd.Series([10, -4, 72, 8], index=['d', 'b', 'a', 'c'])

In [7]: y

Out[7]: d    10
        b    -4
        a    72
        c     8
        dtype: int64

In [8]: y.index

Out[8]: Index(['d', 'b', 'a', 'c'], dtype='object')

In [9]: y['a']

Out[9]: 72

In [10]: y[y > 0]

Out[10]: d    10
         a    72
         c     8
         dtype: int64

In [11]: y * 2

Out[11]: d     20
         b     -8
         a    144
         c     16
         dtype: int64
```

```
In [12]: 'b' in y
Out[12]: True

In [13]: 'e' in y
Out[13]: False
```

Create from Python Dict

```
In [14]: sdata = {'Jakarta': 35000, 'Bandung': 71000, 'Surabaya': 16000, 'Yogyakarta': 5000}

In [15]: z = pd.Series(sdata)

In [16]: z
Out[16]: Jakarta       35000
         Bandung       71000
         Surabaya      16000
         Yogyakarta     5000
         dtype: int64

In [17]: z['Jakarta']
Out[17]: 35000
```

DataFrame

DataFrame

In [18]:
```
data = {'state': ['Ohio', 'Ohio', 'Ohio', 'Nevada', 'Nevada', 'Nevada'],
        'year': [2000, 2001, 2002, 2001, 2002, 2003],
        'pop': [1.5, 1.7, 3.6, 2.4, 2.9, 3.2]}
frame = pd.DataFrame(data)
```

In [19]:
```
frame
```

Out[19]:

	state	year	pop
0	Ohio	2000	1.5
1	Ohio	2001	1.7
2	Ohio	2002	3.6
3	Nevada	2001	2.4
4	Nevada	2002	2.9
5	Nevada	2003	3.2

Display Five First Rows

In [22]:
```
frame.head()
```

Out[22]:

	state	year	pop
0	Ohio	2000	1.5
1	Ohio	2001	1.7
2	Ohio	2002	3.6
3	Nevada	2001	2.4
4	Nevada	2002	2.9

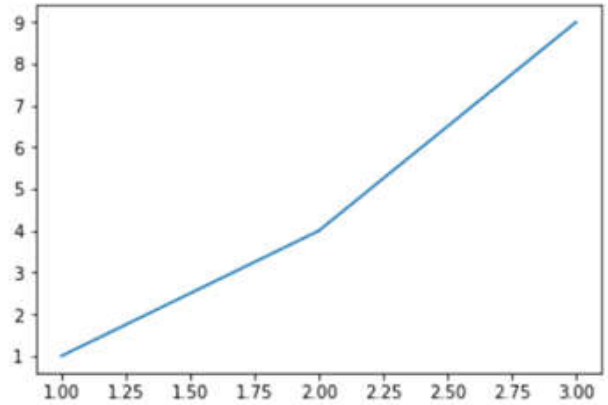
Matplotlib

Import Matplotlib

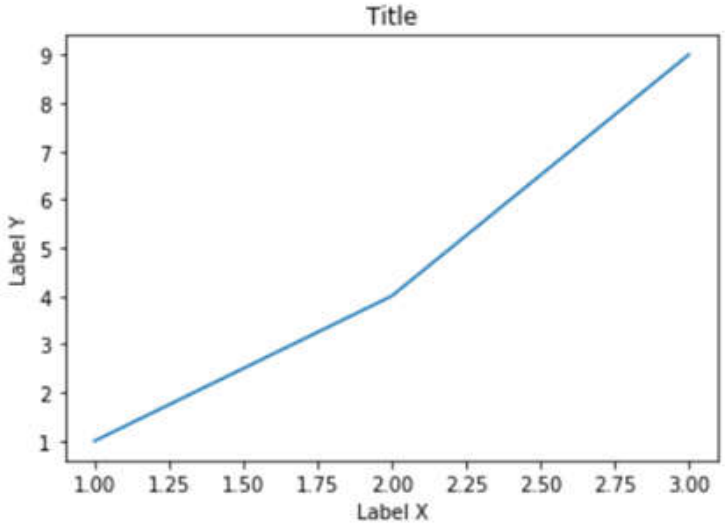
Pyplot merupakan simple interface untuk matplotlib

In [2]:
```
from matplotlib import pyplot as plt
```

In [3]:
```
plt.plot([1,2,3], [1,4,9])
plt.show()
```

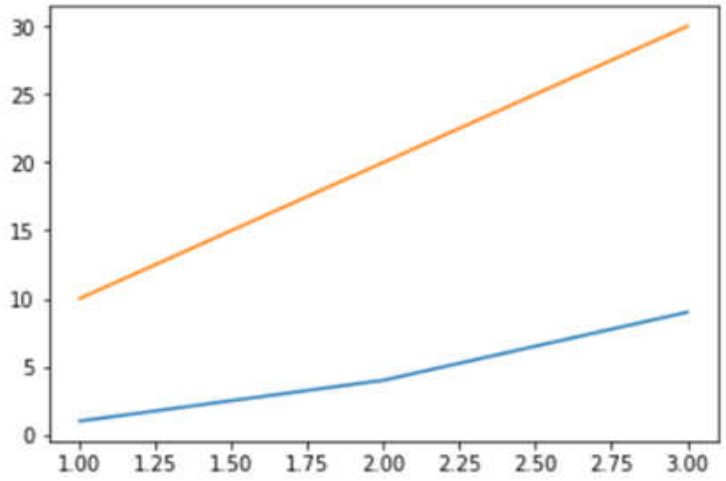


In [4]:
```
plt.plot([1,2,3], [1,4,9])
plt.xlabel('Label X')
plt.ylabel('Label Y')
plt.title('Title')
plt.show()
```

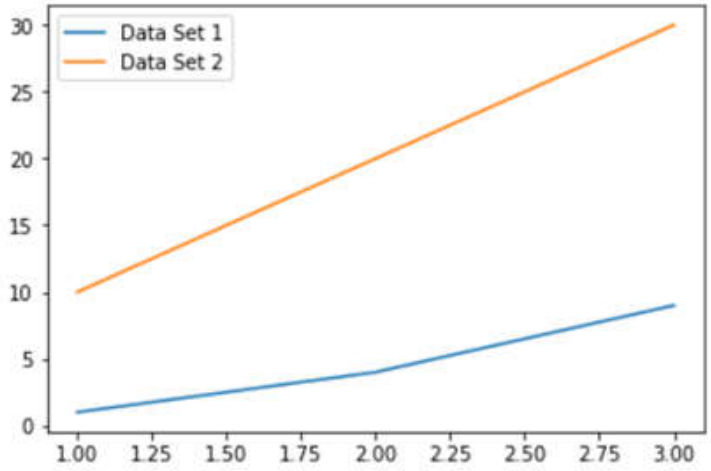


Data Set

In [5]:
```
plt.plot([1,2,3], [1,4,9])
plt.plot([1,2,3], [10,20,30])
plt.show()
```

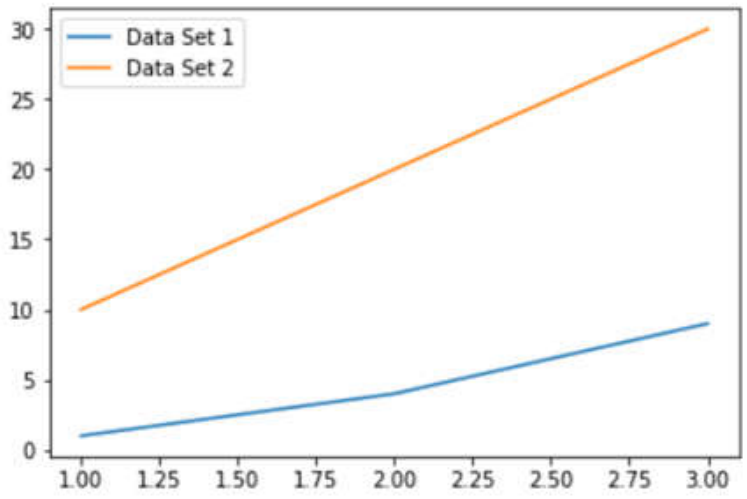


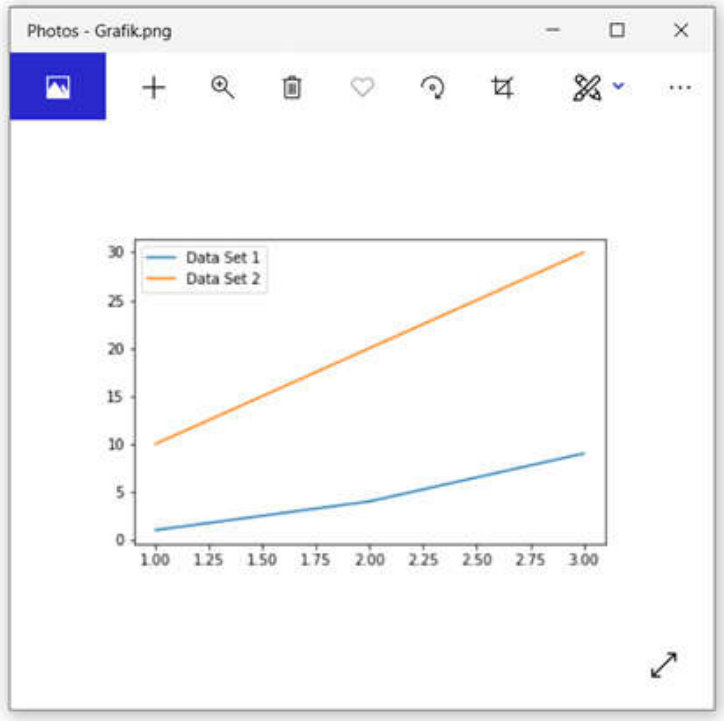
In [9]:
```
plt.plot([1,2,3], [1,4,9])
plt.plot([1,2,3], [10,20,30])
plt.legend(['Data Set 1','Data Set 2'])
plt.show()
```



Save Image

```
In [10]: plt.plot([1,2,3], [1,4,9])
         plt.plot([1,2,3], [10,20,30])
         plt.legend(['Data Set 1','Data Set 2'])
         plt.savefig('Grafik')
```





Pandas

In [2]:
```
from matplotlib import pyplot as plt
import pandas as pd
```

In [3]:
```
data = {'year': [2010, 2014, 2018],
        'attendees': [112, 321, 729],
        'average age': [20, 35, 28]
       }
df = pd.DataFrame(data)
```

In [4]:
```
df
```

Out[4]:

	year	attendees	average age
0	2010	112	20
1	2014	321	35
2	2018	729	28

In [5]:
```
df['year']
```

Out[5]:
```
0    2010
1    2014
2    2018
Name: year, dtype: int64
```

In [6]:
```
type(df['year'])
```

Out[6]:
```
pandas.core.series.Series
```

In [7]:
```
df['year'] < 2015
```

Out[7]:
```
0     True
1     True
2    False
Name: year, dtype: bool
```

In [8]:
```
before_2015 = df['year'] < 2015
```

In [9]:
```
df[before_2015]
```

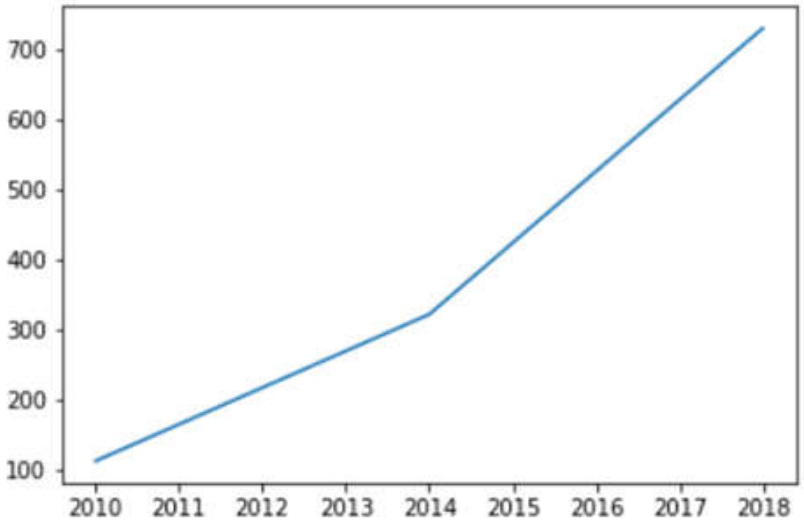
Out[9]:

	year	attendees	average age
0	2010	112	20
1	2014	321	35

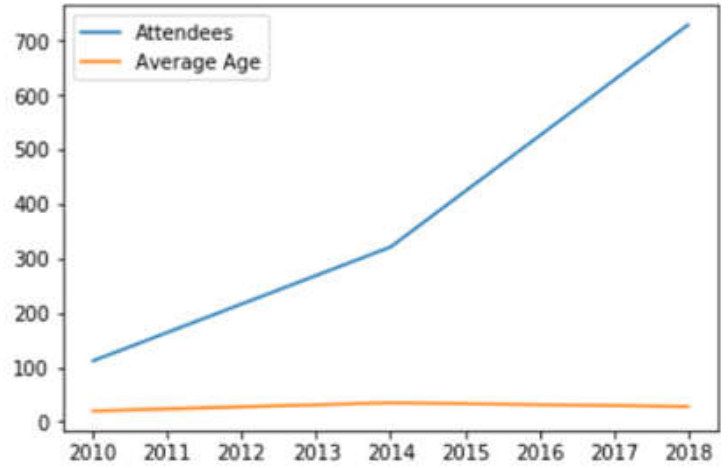
In [10]:
```
before_2015
```

Out[10]:
```
0     True
1     True
2    False
Name: year, dtype: bool
```

In [11]:
```
plt.plot(df['year'], df['attendees'])
plt.show()
```



In [12]:
```
plt.plot(df['year'], df['attendees'])
plt.plot(df['year'], df['average age'])
plt.legend(['Attendees', 'Average Age'])
plt.show()
```



Importing Data

```
In [1]: import pandas as pd
        from matplotlib import pyplot as plt
```

```
In [3]: data = pd.read_csv('countries.csv')
        data
```

Out[3]:

	country	year	population
0	Afghanistan	1952	8425333
1	Afghanistan	1957	9240934
2	Afghanistan	1962	10267083
3	Afghanistan	1967	11537966
4	Afghanistan	1972	13079460
5	Afghanistan	1977	14880372
6	Afghanistan	1982	12881816
7	Afghanistan	1987	13867957
8	Afghanistan	1992	16317921

```
In [4]: data = pd.read_csv('countries.csv')
        data.head()
```

Out[4]:

	country	year	population
0	Afghanistan	1952	8425333
1	Afghanistan	1957	9240934
2	Afghanistan	1962	10267083
3	Afghanistan	1967	11537966
4	Afghanistan	1972	13079460

```
In [5]: data = pd.read_csv('countries.csv')
        data.tail()
```

Out[5]:

	country	year	population
1699	Zimbabwe	1987	9216418
1700	Zimbabwe	1992	10704340
1701	Zimbabwe	1997	11404948
1702	Zimbabwe	2002	11926563
1703	Zimbabwe	2007	12311143

```
In [6]: data['country']
```

```
Out[6]: 0       Afghanistan
        1       Afghanistan
        2       Afghanistan
        3       Afghanistan
        4       Afghanistan
        5       Afghanistan
        6       Afghanistan
        7       Afghanistan
        8       Afghanistan
        9       Afghanistan
```

```
In [7]: data.country
```

```
Out[7]: 0       Afghanistan
        1       Afghanistan
        2       Afghanistan
        3       Afghanistan
        4       Afghanistan
        5       Afghanistan
        6       Afghanistan
        7       Afghanistan
        8       Afghanistan
```

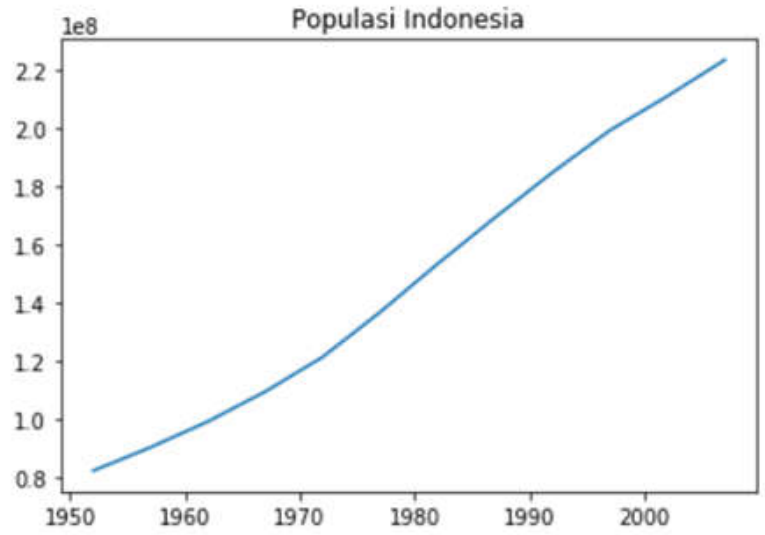
```
In [9]: data[data.country == 'Indonesia']
```

Out[9]:

	country	year	population
708	Indonesia	1952	82052000
709	Indonesia	1957	90124000
710	Indonesia	1962	99028000
711	Indonesia	1967	109343000
712	Indonesia	1972	121282000
713	Indonesia	1977	136725000
714	Indonesia	1982	153343000
715	Indonesia	1987	169276000
716	Indonesia	1992	184816000
717	Indonesia	1997	199278000
718	Indonesia	2002	211060000
719	Indonesia	2007	223547000

In [10]:
```
indonesia = data[data.country == 'Indonesia']
```

In [12]:
```
plt.plot(indonesia.year, indonesia.population)
plt.title('Populasi Indonesia')
plt.show()
```



In [19]:
```
# Compare the population growth in the US and China
```

In [23]:
```
data[data.country == 'United States']
```

Out[23]:

	country	year	population
1608	United States	1952	157553000
1609	United States	1957	171984000
1610	United States	1962	186538000
1611	United States	1967	198712000
1612	United States	1972	209896000
1613	United States	1977	220239000
1614	United States	1982	232187835
1615	United States	1987	242803533
1616	United States	1992	256894189
1617	United States	1997	272911760
1618	United States	2002	287675526
1619	United States	2007	301139947

In [20]:
```
us = data[data.country == 'United States']
```

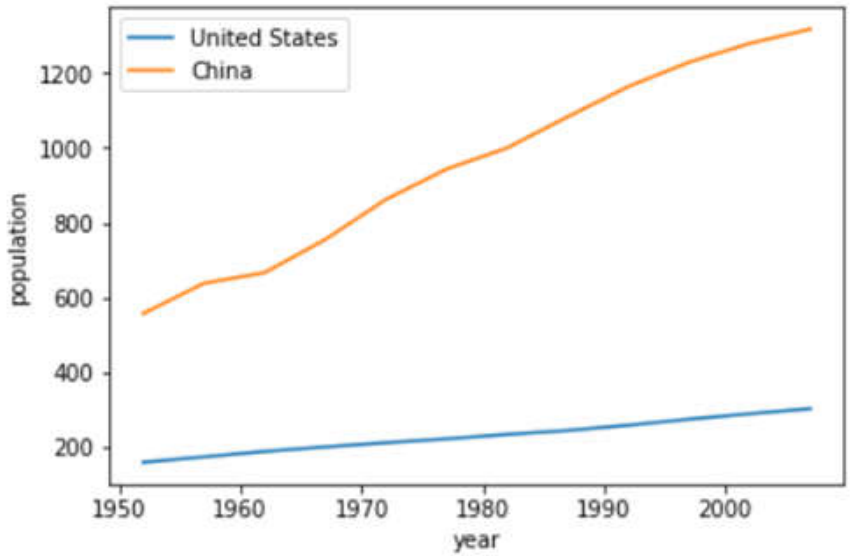
In [24]:
```
china = data[data.country == 'China']
```

In [25]:
```
china
```

Out[25]:

	country	year	population
288	China	1952	556263527
289	China	1957	637408000
290	China	1962	665770000
291	China	1967	754550000
292	China	1972	862030000
293	China	1977	943455000
294	China	1982	1000281000
295	China	1987	1084035000
296	China	1992	1164970000
297	China	1997	1230075000
298	China	2002	1280400000
299	China	2007	1318683096

In [29]:
```
plt.plot(us.year, us.population / 10**6)
plt.plot(china.year, china.population / 10**6)
plt.legend(['United States', 'China'])
plt.xlabel('year')
plt.ylabel('population')
plt.show()
```



In [24]:
```
df = pd.read_csv('nations.csv')
df
```

Out[24]:

	iso2c	iso3c	country	year	gdp_percap	life_expect	population	birth_rate	neonat_mortal_rate	region	income
0	AD	AND	Andorra	1996	NaN	NaN	64291.0	10.900	2.8	Europe & Central Asia	High income
1	AD	AND	Andorra	1994	NaN	NaN	62707.0	10.900	3.2	Europe & Central Asia	High income
2	AD	AND	Andorra	2003	NaN	NaN	74783.0	10.300	2.0	Europe & Central Asia	High income
3	AD	AND	Andorra	1990	NaN	NaN	54511.0	11.900	4.3	Europe & Central Asia	High income
4	AD	AND	Andorra	2009	NaN	NaN	85474.0	9.900	1.7	Europe & Central Asia	High income
5	AD	AND	Andorra	2011	NaN	NaN	82326.0	NaN	1.6	Europe & Central Asia	High income
6	AD	AND	Andorra	2004	NaN	NaN	78337.0	10.900	2.0	Europe & Central Asia	High income
7	AD	AND	Andorra	2010	NaN	NaN	84419.0	9.800	1.7	Europe & Central Asia	High income
8	AD	AND	Andorra	2001	NaN	NaN	67770.0	11.800	2.1	Europe & Central Asia	High income
9	AD	AND	Andorra	2002	NaN	NaN	71046.0	11.200	2.1	Europe & Central Asia	High income
10	AD	AND	Andorra	1997	NaN	NaN	64147.0	11.200	2.6	Europe & Central Asia	High income
11	AD	AND	Andorra	1993	NaN	NaN	61003.0	11.400	3.4	Europe & Central Asia	High income
12	AD	AND	Andorra	2008	NaN	NaN	85616.0	10.400	1.8	Europe & Central Asia	High income

In [45]:
```
set(df.region)
```

Out[45]:
```
{'East Asia & Pacific',
 'Europe & Central Asia',
 'Latin America & Caribbean',
 'Middle East & North Africa',
 'North America',
 'South Asia',
 'Sub-Saharan Africa'}
```

In [49]:
```
data_2007 =  df[df.year == 2007]
```

In [50]:
```
south_asia_2007 = data_2007[data_2007.region == 'South Asia']
north_america_2007 = data_2007[data_2007.region == 'North America']
```

In [51]:
```
south_asia_2007.head()
```

Out[51]:

	iso2c	iso3c	country	year	gdp_percap	life_expect	population	birth_rate	neonat_mortal_rate	region	income
70	AF	AFG	Afghanistan	2007	1245.059223	57.833829	2.587754e+07	42.779	40.4	South Asia	Low income
386	BD	BGD	Bangladesh	2007	2031.778522	68.859976	1.465927e+08	22.858	32.8	South Asia	Low income
695	BT	BTN	Bhutan	2007	5172.726954	66.293098	6.814710e+05	21.991	25.5	South Asia	Lower middle income
2156	IN	IND	India	2007	3484.756463	65.300439	1.179686e+09	23.144	36.0	South Asia	Lower middle income
2726	LK	LKA	Sri Lanka	2007	6964.986319	74.194122	1.966800e+07	18.415	7.7	South Asia	Lower middle income

In [53]:
```
set(south_asia_2007.country)
```

Out[53]:
```
{'Afghanistan',
 'Bangladesh',
 'Bhutan',
 'India',
 'Maldives',
 'Nepal',
 'Pakistan',
 'Sri Lanka'}
```

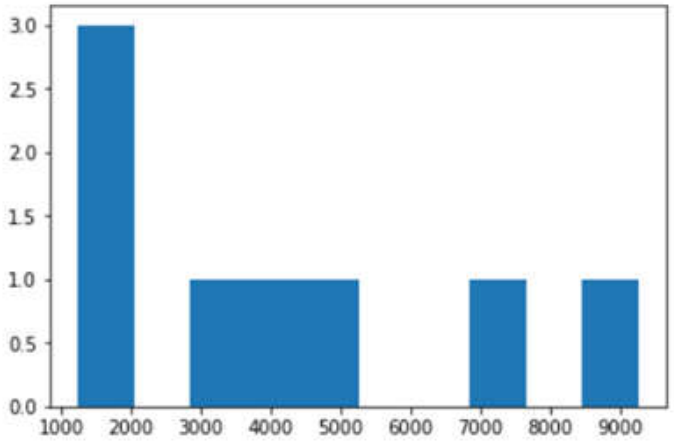
In [54]:
```
print(len(set(south_asia_2007.country)))
print(len(set(north_america_2007.country)))
```

```
8
3
```

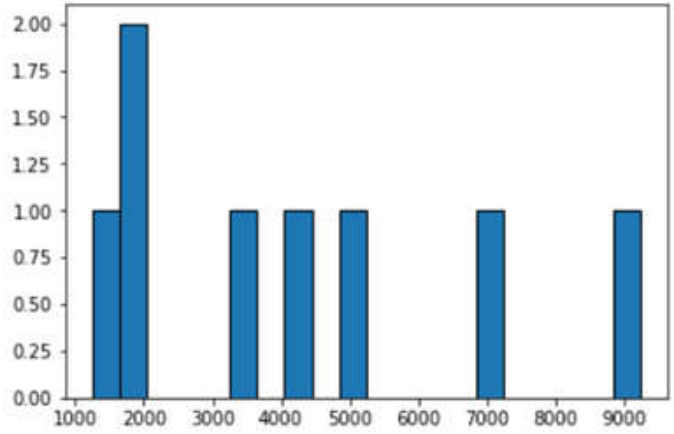
In [56]:
```
print('Mean GDP in South Asia')
print(south_asia_2007.gdp_percap.mean())
print('Mean GDP in North America')
print(north_america_2007.gdp_percap.mean())
print('Median GDP in South Asia')
print(south_asia_2007.gdp_percap.median())
print('Median GDP in North America')
print(north_america_2007.gdp_percap.median())
```

```
Mean GDP in South Asia
4234.135677604255
Mean GDP in North America
48040.01954450427
Median GDP in South Asia
3769.6991305677348
Median GDP in North America
48061.5376613353
```

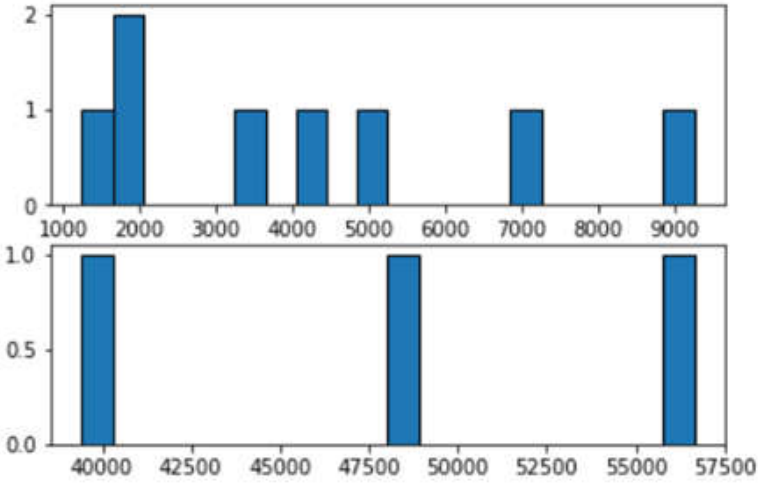
In [57]:
```
plt.hist(south_asia_2007.gdp_percap)
plt.show()
```



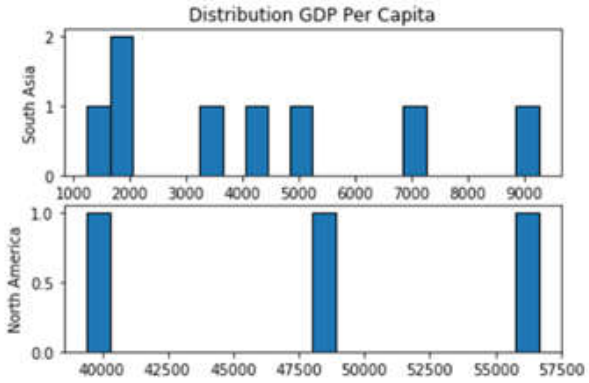
In [58]:
```
plt.hist(south_asia_2007.gdp_percap, 20, edgecolor='black')
plt.show()
```



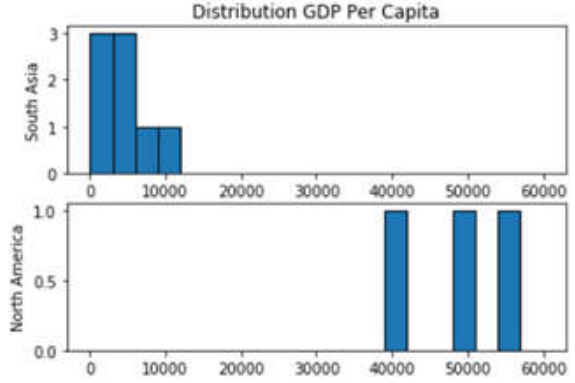
In [59]:
```
plt.subplot(2,1,1) #subplot(211)
plt.hist(south_asia_2007.gdp_percap, 20, edgecolor='black')

plt.subplot(2,1,2)
plt.hist(north_america_2007.gdp_percap, 20, edgecolor='black')

plt.show()
```



In [60]:
```
plt.subplot(2,1,1) #subplot(211)
plt.title('Distribution GDP Per Capita')
plt.hist(south_asia_2007.gdp_percap, 20, edgecolor='black')
plt.ylabel('South Asia')

plt.subplot(2,1,2)
plt.hist(north_america_2007.gdp_percap, 20, edgecolor='black')
plt.ylabel('North America')
plt.show()
```



In [61]:
```
plt.subplot(2,1,1) #subplot(211)
plt.title('Distribution GDP Per Capita')
plt.hist(south_asia_2007.gdp_percap, 20, range=(0, 60000), edgecolor='black')
plt.ylabel('South Asia')

plt.subplot(2,1,2)
plt.hist(north_america_2007.gdp_percap, 20, range=(0, 60000), edgecolor='black')
plt.ylabel('North America')
plt.show()
```



Latihan

- Compare Europe & Central Asia and Latin America & Caribbean’s Life Expectancy in 2000 using Histogram