

Markdown

This is a markdown cell where we write instructions and math formulas. **Double click** to edit and **press Run** to see the formatted texts.

Load and Save Notebook

To load a note book, just go to "File" and navigate to "Open...". Use keyboard shortcut **Cmd/Ctrl+s** to save the file. You will be saving a checkpoint as well if you use this keyboard shortcut. You can click on "File" on the menu bar and select an option to revert to a certain checkpoint. Jupyter Notebook will also autosave your file every 120s, but it's still a good practice to press **Cmd/Ctrl+s** often when you write programs.

Code Blocks

Below will be an example of a Python code block. Hit **Run** to run the block.

```
In [1]: print("Hello world!")
```

Hello world!

You can also hit **Run All** in "Cell" menu. To run every blocks.

```
In [2]: print("Hello world, again!")
```

Hello world, again!

Python Programming

We have finished Jupyter Notebook introduction. Below we will start Python Programming.

Importing Libraries

For the lab, we will be using different libraries. This is how you do it in Python.

```
In [3]: import pandas as pd
import numpy as np
```

By importing pandas as pd, you can use the shorthand `pd` when you need to use pandas. The same method applies to `np`.

NumPy Tutorial

Below we show you a few ways to create arrays.

```
In [4]: # Create a basic array
x = np.array([0, 1, 2, 3])
print("x:\n", x)

# Create a 2D array
y = np.array([[0, 1, 2, 3],
              [4, 5, 6, 7]])
print("y:\n", y)

# Create a 3 by 4 empty array
z = np.zeros((3, 4))
print("z:\n", z)

# Create a 5 by 5 array filled with 1s
a = np.ones((5, 5))
print("a:\n", a)

# Create a sequence of numbers, note the array does not include 20
b = np.arange(0, 20, 2)
print("b:\n", b)

# Use reshape to change the dimension of the array
c = np.arange(0, 20, 2, dtype=float).reshape(2, 5)
print("c:\n", c)

# Create a random 6 by 3 array with value between [0, 1).
d = np.random.rand(6, 3)
print("d:\n", d)

# This creates random values ranging between [1, 4).
d = d*3 + 1
print("New d:\n", d)

# Operations
e = d + 10
print("e:\n", e)
```

```

x:
[0 1 2 3]
y:
[[0 1 2 3]
 [4 5 6 7]]
z:
[[0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]]
a:
[[1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1.]]
b:
[ 0  2  4  6  8 10 12 14 16 18]
c:
[[ 0.  2.  4.  6.  8.]
 [10. 12. 14. 16. 18.]]
d:
[[0.19131215 0.19806847 0.6063701 ]
 [0.40219466 0.46790151 0.41932415]
 [0.17440604 0.55262483 0.0535973 ]
 [0.57842371 0.66882642 0.801927  ]
 [0.19263839 0.54796216 0.02273444]
 [0.66330064 0.13471529 0.82358127]]
New d:
[[1.57393645 1.59420542 2.81911031]
 [2.20658398 2.40370454 2.25797246]
 [1.52321811 2.65787448 1.16079189]
 [2.73527112 3.00647927 3.40578101]
 [1.57791517 2.64388649 1.06820332]
 [2.98990192 1.40414588 3.4707438 ]]
e:
[[11.57393645 11.59420542 12.81911031]
 [12.20658398 12.40370454 12.25797246]
 [11.52321811 12.65787448 11.16079189]
 [12.73527112 13.00647927 13.40578101]
 [11.57791517 12.64388649 11.06820332]
 [12.98990192 11.40414588 13.4707438 ]]

```

Now it's your turn.

To-do:

1. Create and print out an empty array with dimensions of 15 by 29.
2. From the 0 array you created, make it to an ones array without using np.ones. Print out the array.
3. Create an 8 by 9 array with values between [5, 11). Note that 11 is excluded. Print out the array.

Insert code below

```
# Q3
# Create a random 8 by 9 array with value between [0, 1).
arr1 = np.random.rand(8, 9)
# Use operations to create random values ranging between [5, 11).
arr1 = arr1*6 + 5
print("3. 8 by 9 array 'arr1' with values between [5,11):\n", arr1)
```

[illegible]

```

1. 1. 1. 1. 1.]
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1.]
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1.]
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1.]
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1.]
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1.]
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1.]
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1.]
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1.]
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1.]
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1.]
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1.]]
3. 8 by 9 array 'arr1' with values between [5,11):
[[ 5.85061351  6.19063994  6.0600434  5.83013236  7.73595169  5.4994962
  9.20944035  6.2625936  9.25203564]
 [ 6.03540038  8.64944006 10.91648485  5.87040845  7.82654437  6.88838566
  8.14939402  9.84679248 10.56217391]
 [ 9.05872406  8.81532207  9.20215909  7.92698469 10.06597824 10.87191459
  9.95199901  8.56550523  6.6483805 ]
 [ 8.89888245  9.80866821 10.02159614 10.31429067  7.31272914  5.67996929
 10.39667492  7.63996932  7.28746727]
 [ 7.42491813  8.93888157 10.30668006  5.39917229  8.45387572  7.83843523
  5.2047312  7.97720869  8.5016959 ]
 [ 8.36828449  8.14097908  5.66675277  6.30416647  5.23821278  5.5436112
  8.88411069  8.14898046  9.35247849]
 [ 7.31726618  7.30203927  8.403802  5.456208  9.12352906  5.30924929
  5.94698032 10.17724717  7.77567709]
 [ 9.49766449  7.74347651 10.63295518  5.47768264  9.27326721 10.02869831
  8.27153649 10.67303962  9.75846156]]

```

Pandas Tutorial

Create Data Frame

Pandas is a data frame that will be used to handel our data.

In [6]:

```
# Create a data frame from numpy array
df_1 = pd.DataFrame(y, columns=['col 1', 'col 2', 'col 3', 'col 4'])
print("df_1:\n", df_1, '\n')
print("-----") # for readability only

# You can also use display() to make the dataframe looks nicer at the output
print("df_1 using display:")
display(df_1)
print("-----")

# Create a data frame with strings
data_2 = {'Aminal': ['Dog', 'Cat'], 'Color': ['Yellow', 'Pink'], 'Age': [1, 3]}
df_2 = pd.DataFrame(data=data_2)
print("df_2:\n", df_2, '\n')
print("-----")

# Print the type of the data. Notice that Age is int
df_2.dtypes
```

```
df_1:
   col 1  col 2  col 3  col 4
0      0      1      2      3
1      4      5      6      7
```

df_1 using display:

	col 1	col 2	col 3	col 4
0	0	1	2	3
1	4	5	6	7

df_2:

	Aminal	Color	Age
0	Dog	Yellow	1
1	Cat	Pink	3

Out[6]: Aminal object
Color object
Age int64
dtype: object

Now it's your turn.

To-do:

1. Create a data frame containing demographics of you and your 2 friends. The data frame should be 3 by 4. The columns will be 'Name', 'Age', 'Height', 'Hobby'. You can make up data if you like. Print or display the result.

In [7]:

```
### Insert your code below ###

# Create a dictionary with demographics for three people
friends = {'Name': ['Madhu', 'Rachel', 'Ross'], 'Age': [21,24,26], 'Height':
           'Hobby': ['Tennis', 'Shopping', 'Museums']}

# Create a data frame using "friends" dictionary
demographics = pd.DataFrame(data=friends)

# Display dataframe
print("Dataframe (3 by 4) containing demographics of three individuals:")
display(demographics)
```

Dataframe (3 by 4) containing demographics of three individuals:

	Name	Age	Height	Hobby
0	Madhu	21	5'4"	Tennis
1	Rachel	24	5'5"	Shopping
2	Ross	26	6'1"	Museums

Modify Data Frame

Notice that when you use `=`, you are not copying data frame. You are just saying that `df_3` is now referring to the same data frame as `df_1`. If you change values in `df_3` you will change the values in `df_1` too, since both are referring to the same dataframe. Also notice that every time you click **Run** in this block, the values in `'col 1'` changes.

In [8]:

```
# Add a new column
df_2['Weight'] = [89, 60]
print("New df_2:")
display(df_2)
print("-----")

# Special thing to take notice
df_3 = df_1
df_3['col 1'] = df_3['col 1'] - 1
print("df_3:")
display(df_3)
print("df_1:")
display(df_1)
```

New df_2:

	Aminal	Color	Age	Weight
0	Dog	Yellow	1	89
1	Cat	Pink	3	60

df_3:

	col 1	col 2	col 3	col 4
0	-1	1	2	3
1	3	5	6	7

df_1:

	col 1	col 2	col 3	col 4
0	-1	1	2	3
1	3	5	6	7

Print Specific Data

Here we use `.loc` , `.at` to obtain the cell values by providing the **labels** (e.g. 'Age', 'Weight'). For rows, since we do not create labels for them the defaults will be 0, 1, 2, 3...etc. We use `.iloc` , `.iat` to obtain the cell values by providing the indices (positions) of the rows and columns.

In [9]:

```
# Few ways to view the values

# Create a data frame based on data_2; the values are copied
df_4 = pd.DataFrame(data=data_2)
print("df_4:")
display(df_4)
print("-----")

# Selection by Label
    # Getting the scalar value
dog_age = df_4.loc[0, 'Age']
print("Age of Dog:", dog_age, '\n')
print("-----")

    # Getting the whole column
animals_age = df_4.loc[:, ['Age']]
print("Age Column:")
display(animals_age)
print("-----")

    # Faster way to get a scalar
cat_age = df_4.at[1, 'Age']
print("Age of Cat:", cat_age, '\n')
print("-----")

# Selection by position
    # Selecting Row based on row number
dog = df_4.iloc[0]
print("Dog row:")
print(dog, '\n')
print("-----")

    # Selecting Col based on col number
animals_age_p = df_4.iloc[:, 2]
print("Animals' age:")
print(animals_age_p, '\n')
print("-----")

    # Selecting cell based on col number
cat_age_p = df_4.iat[1, 2]
print("Age of Cat: ", cat_age_p)
```

```
df_4:
```

	Aminal	Color	Age
0	Dog	Yellow	1
1	Cat	Pink	3

```
-----  
Age of Dog: 1
```

```
-----  
Age Column:
```

	Age
0	1
1	3

```
-----  
Age of Cat: 3
```

```
-----  
Dog row:
```

```
Aminal      Dog  
Color      Yellow  
Age         1  
Name: 0, dtype: object
```

```
-----  
Animals' age:
```

```
0      1  
1      3  
Name: Age, dtype: int64
```

```
-----  
Age of Cat: 3
```

Now it's your turn.

To-do:

1. Add a new **row** to your **demographics** data frame. The new row will contain information of another friend. *Note that we haven't taught you how to do so but you should be able to find resources online easily.* Print/display the data frame.
2. Print all the information (whole row) about you using `.loc`.
3. Print the 'Name' of your second friend using `.iat`.

In [10]:

```
### Insert your code below ###

# Dictionary containing new information of another friend
new_friend = {'Name': 'Monica', 'Age': 24, 'Height': "5'4\"", 'Hobby': 'Cooki

# Add new row (information from dictionary new_friend) to demographics datafr
demographics = demographics.append(new_friend, ignore_index = True)

print("Demographics data frame with new row with information of another frien
display(demographics)
print("-----\n")

# Select entire row for label of the index (0) using .loc
my_information = demographics.loc[0]
print("My information (whole row) using .loc:")
print(my_information, '\n')

print("-----\n")

# Selecting cell based on row/column pair (row=2, column=0)
name_friend2 = demographics.iat[2, 0]
print("Name of Second Friend using .iat: ", name_friend2, "\n")
print("-----")
```

Demographics data frame with new row with information of another friend append
ed:

	Name	Age	Height	Hobby
0	Madhu	21	5'4"	Tennis
1	Rachel	24	5'5"	Shopping
2	Ross	26	6'1"	Museums
3	Monica	24	5'4"	Cooking

My information (whole row) using .loc:

Name Madhu
Age 21
Height 5'4"
Hobby Tennis
Name: 0, dtype: object

Name of Second Friend using .iat: Ross

Congradulation on finishing the tutorial! Now you can move on to the next step of the lab.