Name: Akanksh Rao S R
Email: akankshrao2212@gmail.com

What is Pandas?

python library for data manipulation and analysis

```
import pandas as pd
data_frame = pd.read_csv('data/friend_list.csv')
```

```
data_frame
```

	name	age	job
0	John	20	student
1	Jenny	30	developer
2	Nate	30	teacher
3	Julia	40	dentist
4	Brian	45	manager
5	Chris	25	intern

What is DataFrame?

dataframe is a 2-dimensional labeled data structure with columns

```
data_frame.head()
```

	name	age	job
0	John	20	student
1	Jenny	30	developer
2	Nate	30	teacher
3	Julia	40	dentist
4	Brian	45	manager

What is Series?

Every single column in dataframe is series

```
type(data_frame.job)
```

```
pandas.core.series.Series
```

```
data_frame.job = data_frame.job.str.upper()
data_frame.head()
```

	name	age	job
0	John	20	STUDENT
1	Jenny	30	DEVELOPER
2	Nate	30	TEACHER
3	Julia	40	DENTIST
4	Brian	45	MANAGER

Series is just wrapper for python list

```
s1 = pd.core.series.Series(['one', 'two', 'three'])
s2 = pd.core.series.Series([1, 2, 3])
pd.DataFrame(data=dict(word=s1, num=s2))
```

	num	word
0	1	one
1	2	two
2	3	three

Why Pandas?

Very similar to Excel spreadsheet view,
support various functions for data manipulation and analysis.
Fast based on Numpy.
Easy to manipulate data for your purpose

Read File to DataFrame

A **Data frame** is a two-dimensional data structure, i.e., data is aligned in a tabular fashion in rows and columns.

by default, pandas support csv format

```
df = pd.read_csv('data/friend_list.csv')
```

```
df
```

	name	age	job
0	John	20	student
1	Jenny	30	developer
2	Nate	30	teacher
3	Julia	40	dentist
4	Brian	45	manager
5	Chris	25	intern

you can read txt file like below, if the txt file data are comma separated

```
df = pd.read_csv('data/friend_list.txt')
```

```
df.head()
```

	name	age	job
0	John	20	student
1	Jenny	30	developer
2	Nate	30	teacher
3	Julia	40	dentist
4	Brian	45	manager

if txt file delimiter is not comma, you can use define delimiter using keyword argument

```
df = pd.read_csv('data/friend_list_tab.txt', delimiter = "\t")
```

```
df.head()
```

	name	age	job
0	John	20	student
1	Jenny	30	developer
2	Nate	30	teacher
3	Julia	40	dentist

if data file doesn't have header,
Use header = None like below, so first column not to be your column header

```
df = pd.read_csv('data/friend_list_no_head.csv', header = None)
```

```
df.head()
```

	0	1	2
0	John	20	student
1	Jenny	30	developer
2	Nate	30	teacher
3	Julia	40	dentist
4	Brian	45	manager

you can add column header after you create dataframe

```
df.columns = ['name', 'age', 'job']
```

```
df.head()
```

	name	age	job
0	John	20	student
1	Jenny	30	developer
2	Nate	30	teacher
3	Julia	40	dentist
4	Brian	45	manager

you can create column header for no header data at once

```
df = pd.read_csv('data/friend_list_no_head.csv', header = None, names=['name', 'age', 'job'])
```

```
df.head()
```

	name	age	job
0	John	20	student
1	Jenny	30	developer
2	Nate	30	teacher
3	Julia	40	dentist
4	Brian	45	manager

Create DataFrame

when you want to create dataframe from your python code

[ ] ↳ *15 cells hidden*

Write DataFrame to File

[ ] ↳ *24 cells hidden*

Select Row

[ ] ↳ *16 cells hidden*

Filter Column

[ ] ↳ *14 cells hidden*

Drop rows

[ ] ↳ *23 cells hidden*

Drop column

Add Column / Update Column

[ ] ↳ *48 cells hidden*

Add Row

[ ] ↳ *4 cells hidden*

Group by

group by command helps to get more information from given data

[ ] ↳ *11 cells hidden*

Drop Duplicate

sometimes you need to drop duplicate rows and here is elegant way to to it

[ ] ↳ *9 cells hidden*

how to manage None value?

[ ] ↳ *9 cells hidden*

Unique

[ ] ↳ *5 cells hidden*

Concatenate two dataframe

[ ] ↳ *10 cells hidden*

Concatenate two list as a dataframe

[ ] ↳ *1 cell hidden*