DATA Science LAB#2

Roll no: 20K-0409

Screen Shots:

Task#1

```
[11]  1 import pandas as pd
      2 import numpy as np
```

```
[12]  1 df1 = pd.read_csv('data1.csv', index_col=0)
      2 df2 = pd.read_csv('data2.csv', index_col=1)
```

```
[13]  1 print("data 1: \n", df1)
      2 print("\ndata 2: \n", df2)
```

```
data 1:
    A    B  C
0   1  2.0  3
1   4  5.0  6
4   2  NaN  5

data 2:
         A  B  C
2        2  5  6
3  Hello  3  4
```

Concat

b. Modify (a) to concatenate df1 and df2 and assign it to df3. Print df3

```
[51]  1
      2 df3 = pd.concat([df1, df2], axis = 0, join='outer')
      3 print("Concated data in df 3 : \n", df3)
      4
```

```
Concated data in df 3 :
         A    B  C
0        1  2.0  3
1        4  5.0  6
4        2  NaN  5
2        2  5.0  6
3  Hello  3.0  4
```

Read data 4 as df4, merging data

c. Read data3.csv as df4 dataframe object and print df4

```
[52]  1 df4 = pd.read_csv('data3.csv', index_col=0)
      2
      3 df5 = pd.concat([df3, df4], axis=1, join='outer')
      4 print(df5)
```

```
         A    B  C    D    E
0        1  2.0  3  NaN  NaN
1        4  5.0  6  1.0  7.0
4        2  NaN  5  0.0  8.0
2        2  5.0  6  NaN  NaN
3  Hello  3.0  4  NaN  NaN
```

d. Read data.json as df6 and concatenate with df5.

```
1
2 df6 = pd.read_json('data.json')
3 print(df6)
4
5
6 df7 = pd.concat([df5, df6], axis=0, ignore_index=True)
7 print(df7)
```

```
    A  B
0  11  9
1  22  7
2  33  8
       A    B    C    D    E
0      1  2.0  3.0  NaN  NaN
1      4  5.0  6.0  1.0  7.0
2      2  NaN  5.0  0.0  8.0
3      2  5.0  6.0  NaN  NaN
4  Hello  3.0  4.0  NaN  NaN
5     11  9.0  NaN  NaN  NaN
6     22  7.0  NaN  NaN  NaN
7     33  8.0  NaN  NaN  NaN
```

e. Replace Hello with NaN.

```
[62]  1 df7 = df7.replace('Hello', value=np.nan)
      2 print(df7)
```

```
     A    B    C    D    E
0    1  2.0  3.0  NaN  NaN
1    4  5.0  6.0  1.0  7.0
2    2  NaN  5.0  0.0  8.0
3    2  5.0  6.0  NaN  NaN
4  NaN  3.0  4.0  NaN  NaN
5   11  9.0  NaN  NaN  NaN
6   22  7.0  NaN  NaN  NaN
7   33  8.0  NaN  NaN  NaN
```

f. Replace NaN with mean values of the columns.

```
[63]  1 df7.fillna(df7.mean(), inplace=True)
      2
      3 df7.to_csv("newdata.csv")
      4 print(df7)
```

```
     A         B    C    D    E
0    1  2.000000  3.0  0.5  7.5
1    4  5.000000  6.0  1.0  7.0
2    2  5.571429  5.0  0.0  8.0
3    2  5.000000  6.0  0.5  7.5
4  NaN  3.000000  4.0  0.5  7.5
5   11  9.000000  4.8  0.5  7.5
6   22  7.000000  4.8  0.5  7.5
7   33  8.000000  4.8  0.5  7.5
```

TASK # 2

TASK 2

```
[20]  1 import pandas as pd
      2 import numpy as np
      3
      4 df = pd.read_csv('BL-Flickr-Images-Book.csv')
      5
      6 df.head(5)
      7 df.columns
```

```
Index(['Identifier', 'Edition Statement', 'Place of Publication',
       'Date of Publication', 'Publisher', 'Title', 'Author', 'Contributors',
       'Corporate Author', 'Corporate Contributors', 'Former owner',
       'Engraver', 'Issuance type', 'Flickr URL', 'Shelfmarks'],
      dtype='object')
```

```
1
2 # Dropping columns
3
4 to_drop = ['Edition Statement',
5             'Corporate Author',
6             'Corporate Contributors',
7             'Former owner',
8             'Engraver',
9             'Contributors',
10            'Issuance type',
11            'Shelfmarks']
12
13 df.drop(to_drop, inplace=True, axis=1)
14 df.head()
15
16
17 # Changing the index of a DataFrame
18
19 df = df.set_index('Identifier')
20 df.head()
21
22 # Accessing records using loc[]:
23 df.loc[206]
24
```

```
Place of Publication                                          London
Date of Publication                                       1879 [1878]
Publisher                                              S. Tinsley & Co.
Title                                  Walter Forbes. [A novel.] By A. A
Author                                                        A. A.
Flickr URL              http://www.flickr.com/photos/britishlibrary/ta...
Name: 206, dtype: object
```

```
2
3 data = pd.read_csv('BL-Flickr-Images-Book.csv')
4
5 # Cleaning the 'Date of Publication' column:
6
7 extr = data['Date of Publication'].str.extract(r'^(\\d{4})', expand=False)
8 extr.head()
9 data['Date of Publication'] = pd.to_numeric(extr)
10 data['Date of Publication'].dtype
11 data['Date of Publication'].isnull().sum() / len(df)
12
13 # Cleaning 'Place of Publication' column:
14
15 pub = data['Place of Publication']
16 london = pub.str.contains('London')
17 london[:5]
18 oxford = pub.str.contains('Oxford')
19 data['Place of Publication'] = np.where(london, 'London',
20                                         np.where(oxford, 'Oxford',
21                                                  pub.str.replace('-', ' ')))
22 data['Place of Publication'].head()
23 #  cleaned DataFrame:
24
25 data.head(5)
```

	Identifier	Edition Statement	Place of Publication	Date of Publication	Publisher	Title	Author	Contributors	Corporate Author	Corporate Contributors	Former owner	Engraver	Issuance type	
0	206	NaN	London	NaN	S. Tinsley & Co.	Walter Forbes. [A novel.] By A. A	A. A.	FORBES, Walter.	NaN	NaN	NaN	NaN	monographic	http://www.flickr.com/phot
1	216	NaN	London	NaN	Virtue & Co.	All for Greed. [A novel. The dedication signed...	A., A. A.	BLAZE DE BURY, Marie Pauline Rose - Baroness	NaN	NaN	NaN	NaN	monographic	http://www.flickr.com/phot
2	218	NaN	London	NaN	Bradbury, Evans & Co.	Love the Avenger. By the author of "All for Gr...	A., A. A.	BLAZE DE BURY, Marie Pauline Rose - Baroness	NaN	NaN	NaN	NaN	monographic	http://www.flickr.com/phot
3	472	NaN	London	NaN	James Darling	Welsh Sketches, chiefly ecclesiastical, to the...	A., E. S.	Appleyard, Ernest Silvanus.	NaN	NaN	NaN	NaN	monographic	http://www.flickr.com/phot
4	480	A new edition, revised, etc.	London	NaN	Wertheim & Macintosh	[The World in which I live, and my place in it...	A., E. S.	BROOME, John Henry.	NaN	NaN	NaN	NaN	monographic	http://www.flickr.com/phot

✓ 0s completed at 1:19 AM

```
olympics_df = pd.read_csv('olympics.csv')
olympics_df.head()


new_names =  {'Unnamed: 0': 'Country', '? Summer': 'Summer Olympics', '01 !': 'Gold', '02 !': 'Silver','03 !': 'Bronze', '? Winter': 'Winter Olympics','01 !.1': 'Gold.1',
              '02 !.1': 'Silver.1', '03 !.1': 'Bronze.1', '? Games': '# Games', '01 !.2': 'Gold.2', '02 !.2': 'Silver.2', '03 !.2': 'Bronze.2'}

olympics_df.rename(columns=new_names, inplace=True)
olympics_df.head()
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	NaN	? Summer	01 !	02 !	03 !	Total	? Winter	01 !	02 !	03 !	Total	? Games	01 !	02 !	03 !	Combined total
1	Afghanistan (AFG)	13	0	0	2	2	0	0	0	0	0	13	0	0	2	2
2	Algeria (ALG)	12	5	2	8	15	3	0	0	0	0	15	5	2	8	15
3	Argentina (ARG)	23	18	24	28	70	18	0	0	0	0	41	18	24	28	70
4	Armenia (ARM)	5	1	2	9	12	6	0	0	0	0	11	1	2	9	12