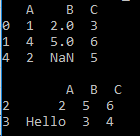
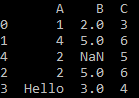
**Data Science: Tools and Techniques  
Lab Exercise 2 (Week 4)**

**Prepared By  
Dr.Muhammad Atif Tahir and Dr. Abdul Aziz (Fall 2020)**

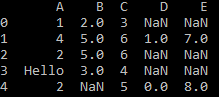
1. Do all examples provided in   
   <http://www.slideshare.net/AndrewHenshaw1/pandas-22984889>
2. In this lab, we will focus on taking input from different sources. Data may be incomplete and there may be missing values. The objective is to combine data and save it as one   
   dataframe object. Do the following steps
   1. Read data1.csv and data2.csv as df1 and df2 dataFrames objects respectively. Use Pandas I/O libraries. Use column 0 as index for df1 and column 1 as index for df2. Print df1 and df2



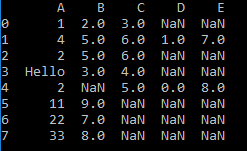
* 1. Modify (a) to concatenate df1 and df2 and assign it to df3. Print df3



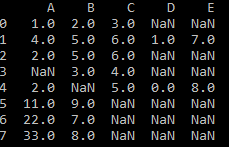
* 1. Read data3.csv as df4 dataframe object and print df4 (not shown below). There are 2 new column ‘D’ and ‘E’ in this file. Merge df4 with df3 so that new dataframe (df5) has total 5 columns (A, B, C, D, E)



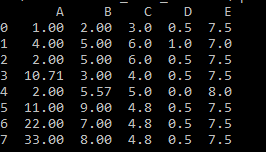
* 1. Read data.json as df6 and concatenate with df5. Use df7 as name of dataframe



* 1. Replace Hello with NaN. Make it as general as possible so that all strings in dataframe df7 automatically becomes NaN



* 1. Replace NaN with mean values of the columns. Save the final dataframe as “newdata.csv”



1. Finish Ten Minutes exercise,

<https://pandas.pydata.org/pandas-docs/stable/getting_started/10min.html#min>

1. Finish Exploratory Data Analysis with Pandas from the following link

<https://www.kaggle.com/kashnitsky/topic-1-exploratory-data-analysis-with-pandas>  
<https://www.kaggle.com/ekami66/detailed-exploratory-data-analysis-with-python>

1. **Complete the following program**

import pandas as pd

data = {'cities' : ['lahore','karachi',], 'provinces' : ['punjab','sindh']}

# store data as DataFrame object. Assign object name as frame1

frame1 = \_pd.DataFrame(data)\_\_\_\_

# print frame

\_print(frame1)\_\_\_\_\_\_\_\_\_\_\_\_\_\_

data2 = {"cities": ["islamabad","karachi","peshawar","quetta"],

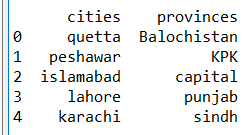
"provinces": ["capital","sindh", "KPK","Balochistan"]}

# store data as DataFrame object. Assign object name as frame2

\_\_frame2 = pd.DataFrame(data2)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# combine both objects frame1 and frame2; without any duplicate rows and re-arrange all indexes

frame3 = …pd.concat([frame1,frame2]) ……………………………………..# combine frame1 and frame2  
frame3 = …frame3.drop\_duplicates()…………..……………………………..# remove duplicates rows   
frame3 = ……frame3.sort\_values(by='provinces' , inplace=True)…# sort based on provinces  
frame3 = ……frame3.reindex(range(2), method='ffill')……..………...# re-arrange all indexes  
…print(frame3)…………………..……………………………………………………….# print frame3

 **Figure1: Screen shot of Final Output for Q3.**

1. Consider the following table

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Field** | **Age** | **Marks** |
|  | C |  | -90 |
| Ali | E |  | 60 |
| Ahmed | E |  | -10 |
| Nida | C |  | 70 |
|  | C |  | 75 |

 Perform following data cleansing operation on the given data.

1. Drop column **Age** as it does not contain any value

A: frame.drop([‘Age’],axis=1)

1. All empty strings in the **Name** column should be replaced by “---“

A: frame[‘Age’].fillna(“---”, inplace=True)

1. In the **Field** column replace “C” with 0 and “E” with 1. The column must contain only numeric values after this operation

A: frame['female'].map({'C': '0', 'E': '1'})

1. Negative values are not permitted in **Marks** column. The invalid value in **Marks** column should be replaced with the average of all valid values in the same column

A: df\_mean\_marks = df7[“Marks”].mean()

frame = np.where(frame ["Marks"]>=0, frame["Marks"], df\_mean\_marks)