DS ASSIGNMENT-2

NAME:T.SRIPOORNIMA

CLASS:III CSE-A

ROLLNO:19BCS027

1). Exploratory Data Analysis: College data set: College.csv. It contains a number of variables for 777 different universities and colleges in the US. Do all the exercises in Python:

a). Read the csv file with pandas

b).Fix the first row as row headers

c).1. produce a numerical summary of the variables in the data set.

2. produce a scatterplot matrix of the first ten columns or variables of the data.

3. produce side-by-side boxplots of Outstate versus Private

4. Create a new qualitative variable, called Elite, by binning the Top10perc variable and divide universities into two groups based on whether or not the proportion of students coming from the top 10 % of their high school classes exceeds 50 %

5. Produce some histograms with differing numbers of bins for a few of the quantitative variables: Room.Board','Books', 'Personal', 'Expend'

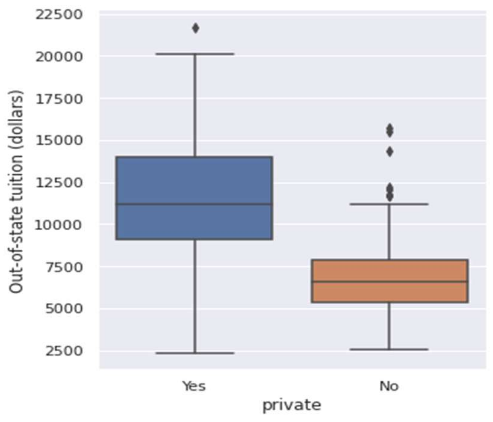
Solution:

a).
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
sns.set()
college = pd.read_csv('data.csv')
```

b).
```
college.rename_axis(index = "College", inplace = True)
college.head()
```

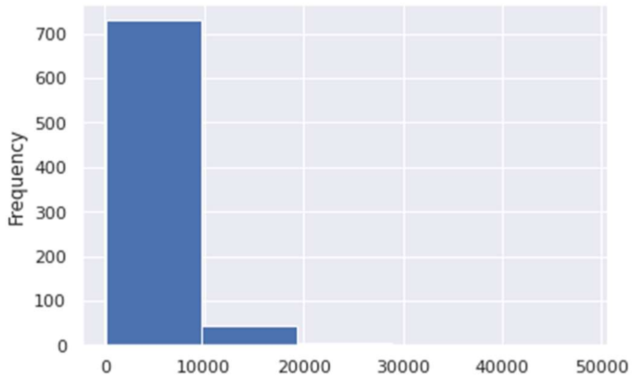
```
[{"index":0,"private":"Yes","apps":1660,"accept":1232,"enroll":721,"top10perc":23,"top25pe
rc":52,"f_undergrad":2885,"p_undergrad":537,"outstate":7440,"room_board":3300,"books":4
50,"personal":2200,"phd":70,"terminal":78,"s_f_ratio":18.1,"perc_alumni":12,"expend":7041
}]
```

c).
```
college.describe()
sns.pairplot(college.iloc[:, 1:11])
 ax = sns.catplot(x = "private", y = "outstate", kind = "box", order = ["Yes", "No"], data =
college)
ax.set(ylabel = "Out-of-state tuition (dollars)")
plt.show()
```

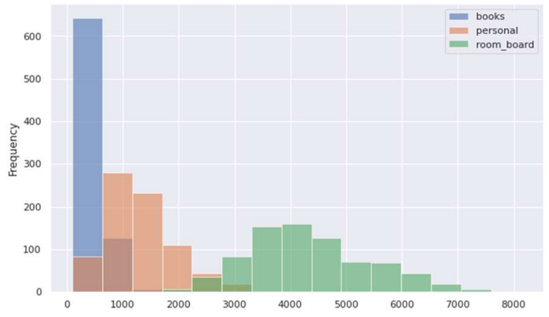


```
d). college["Elite"] = "No"
college.loc[college["top10perc"] > 50, "Elite"] = "Yes"
```

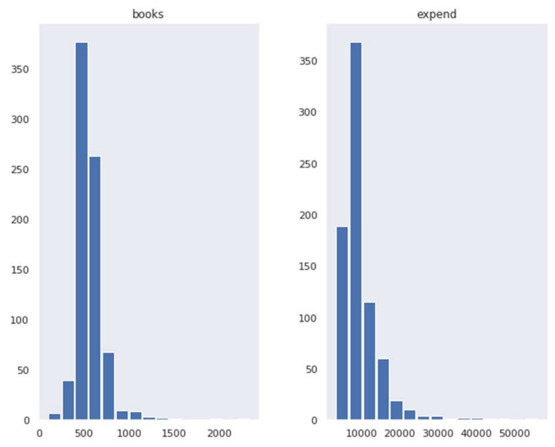
```
e). college['apps'].plot(kind='hist',bins=5)
```



```
college[['books', 'personal', 'room_board']].plot.hist(bins = 15, figsize=(10,6), alpha = 0.6)
plt.show()
```



```
college[['books', 'expend']].hist(
    bins=15,
    figsize=(10, 8),
    grid = False,
    rwidth = 0.9,
)
plt.show()
```



2). Load the energy data from the file Energy Indicators.xls, which is a list of indicators of energy supply and renewable electricity production from the United Nations for the year 2013, and should be put into a DataFrame with the variable name of energy**.**

a). Keep in mind that this is an Excel file, and not a comma separated values file. Also, make sure to exclude the footer and header information from the datafile. The first two columns are unneccessary, so you should get rid of them, and you should change the column labels so that the columns are:['Country', 'Energy Supply', 'Energy Supply per Capita', '% Renewable']

b). Convert Energy Supply to gigajoules (there are 1,000,000 gigajoules in a petajoule). For all countries which have missing data (e.g. data with "...") make sure this is reflected as np.NaN values.

c). Rename the following list of countries (for use in later questions):"Republic of Korea": "South Korea", "United States of America": "United States", "United Kingdom of Great Britain and Northern Ireland": "United Kingdom", "China, Hong Kong Special Administrative Region": "Hong Kong"

d). Next, load the GDP data from the file *world_bank.csv*, which is a csv containing countries' GDP from 1960 to 2015 from World Bank. Call this DataFrame GDP.

Solution:

a). import pandas as pd
import numpy as np

```
def get_energy():
  energy = pd.read_excel('Energy Indicators.xls', header = 15, skip_footer = 38, skiprows = 1)
  energy = energy.drop(energy.columns[0:2], axis = 1)
  energy.columns = ['Country', 'Energy Supply', 'Energy Supply per Capita', '% Renewable']
  energy['Energy Supply'] = energy['Energy Supply'] * 1000000
  energy = energy.replace("...", np.NaN)
  energy.Country.replace(r'\s\([^()]*\)', '', regex = True, inplace = True)
energy.Country.replace(r'\d+', '', regex = True, inplace = True)
country_name = {"Republic of Korea": "South Korea",
                "United States of America": "United States",
                "United Kingdom of Great Britain and Northern Ireland": "United Kingdom",
                "China, Hong Kong Special Administrative Region": "Hong Kong"}
    energy.Country = energy.Country.replace(country_name)
    return energy

    b). def get_GDP():
    GDP = pd.read_csv('world_bank.csv', header = 4)
    country_name ={"Korea, Rep.": "South Korea","Iran, Islamic Rep.": "Iran", "Hong Kong
SAR, China": "Hong Kong"}
    GDP['Country Name'] = GDP['Country Name'].replace(country_name)
    GDP.drop(GDP.columns[1:50], axis=1, inplace = True)
    return GDP

    def one():
    energy = get_energy()

    GDP = get_GDP()

    ScimEn = pd.read_excel('scimagojr-3.xlsx')

    df = pd.merge(ScimEn, energy, how = 'outer')
    df = pd.merge(df, GDP, how = 'outer', left_on = 'Country', right_on = 'Country Name')
    df = df[:15]
    df.drop(['Country Name'], axis = 1, inplace = True)
    df.set_index('Country', inplace = True)

    return df

one()

c). df = pd.merge(ScimEn, energy, how = 'outer')
df = pd.merge(df, GDP, how = 'outer', left_on = 'Country', right_on = 'Country Name')
df = df[:15]
df.drop(['Country Name'], axis = 1, inplace = True)
df.set_index('Country', inplace = True)
df


d). GDP = pd.read_csv('world_bank.csv', header = 4)
country_name ={"Korea, Rep.": "South Korea","Iran, Islamic Rep.": "Iran", "Hong Kong
SAR, China": "Hong Kong"}
GDP['Country Name'] = GDP['Country Name'].replace(country_name)
GDP.drop(GDP.columns[1:50], axis=1, inplace = True)
GDP
```

3). The previous question joined three datasets then reduced this to just the top 15 entries. When you joined the datasets, but before you reduced this to the top 15 items, how many entries did you lose?

Solution:

```
def two():
    union = pd.merge(pd.merge(ScimEn, energy, how ='outer'),
                     GDP, left_on = 'Country', right_on = 'Country Name', how ='outer')
    intersect = pd.merge(pd.merge(ScimEn, energy, how ='inner'),
                         GDP, left_on = 'Country', right_on = 'Country Name', how ='inner')
    return len(union)-len(intersect)
two()
```

4). What is the average GDP over the last 10 years for each country? (exclude missing values from this calculation.)

Solution:

```
def three():
    Top15 = answer_one()
    avg = Top15[['2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013', '2014',
'2015']].mean(axis = 1)
    avg.sort_values(ascending = False, inplace=True)
    return avg

three()
```

Output:

```
Country
United States        1.536434e+13
China                6.348609e+12
Japan                5.542208e+12
Germany              3.493025e+12
France               2.681725e+12
United Kingdom       2.487907e+12
Brazil               2.189794e+12
```

5). By how much had the GDP changed over the 10 year span for the country with the 6th largest average GDP?

Solution:

```
def four():
    Top15 = answer_one()
    avg = answer_three()
    Top15.reset_index(inplace = True)
    return (Top15[Top15.Country == avg.index[5]]['2015'] - Top15[Top15.Country ==
avg.index[5]]['2006']).values[0]
```

```
four()
```

Output:

```
246702696075.3999
```

6). Use the following dictionary to group the Countries by Continent, then create a dateframe that displays the sample size (the number of countries in each continent bin), and the sum, mean, and std deviation for the estimated population of each country.

```
ContinentDict  = {'China':'Asia',
              'United States':'North America',
              'Japan':'Asia'}
```

Solution:

```
def five():
   Top15 = answer_one()
   ContinentDict  = {'China':'Asia',
              'United States':'North America',
              'Japan':'Asia',
              'United Kingdom':'Europe',
              'Russian Federation':'Europe',
              'Canada':'North America',
              'Germany':'Europe',
              'India':'Asia',
              'France':'Europe',
              'South Korea':'Asia',
              'Italy':'Europe',
              'Spain':'Europe',
              'Iran':'Asia',
              'Australia':'Australia',
              'Brazil':'South America'}
   Top15['Population'] = Top15['Energy Supply'] / Top15['Energy Supply per Capita']
   Top15['Population'] = np.float64(Top15['Population'])
   Top15.rename(index = ContinentDict, inplace=True)
   Top15.reset_index(inplace = True)
   Top15.rename(columns = {'Country' : 'ContinentDict'}, inplace = True)
   functions = ['size', 'sum', 'mean', 'std']
   return
Top15[['ContinentDict','Population']].groupby('ContinentDict')['Population'].agg(functions)
five()
```

Output:

```
               size   sum            mean           std
ContinentDict
Asia           5      2.898666e+09   5.797333e+08   6.790979e+08
Australia      1      2.331602e+07   2.331602e+07   NaN
Europe         6      4.579297e+08   7.632161e+07   3.464767e+07
North America  2      3.528552e+08   1.764276e+08   1.996696e+08
South America  1      2.059153e+08   2.059153e+08   NaN
```