*You are currently looking at **version 1.2** of this notebook. To download notebooks and datafiles, as well as get help on Jupyter notebooks in the Coursera platform, visit the Jupyter Notebook FAQ (https://www.coursera.org/learn/python-data-analysis/resources/0dhYG) course resource.*

Assignment 2 - Pandas Introduction

All questions are weighted the same in this assignment.

Part 1

The following code loads the olympics dataset (olympics.csv), which was derrived from the Wikipedia entry on All Time Olympic Games Medals (https://en.wikipedia.org/wiki/All-time_Olympic_Games_medal_table), and does some basic data cleaning.

The columns are organized as # of Summer games, Summer medals, # of Winter games, Winter medals, total # number of games, total # of medals. Use this dataset to answer the questions below.

In [2]:
```
import pandas as pd

df = pd.read_csv('olympics.csv', index_col=0, skiprows=1)

for col in df.columns:
    if col[:2]=='01':
        df.rename(columns={col:'Gold'+col[4:]}, inplace=True)
    if col[:2]=='02':
        df.rename(columns={col:'Silver'+col[4:]}, inplace=True)
    if col[:2]=='03':
        df.rename(columns={col:'Bronze'+col[4:]}, inplace=True)
    if col[:1]=='№':
        df.rename(columns={col:'#'+col[1:]}, inplace=True)

names_ids = df.index.str.split('\s\(') # split the index by '('

df.index = names_ids.str[0] # the [0] element is the country name (new index)
df['ID'] = names_ids.str[1].str[:3] # the [1] element is the abbreviation or I
D (take first 3 characters from that)

df = df.drop('Totals')
df.head()
```

Out[2]:

	# Summer	Gold	Silver	Bronze	Total	# Winter	Gold.1	Silver.1	Bronze.1	To
Afghanistan	13	0	0	2	2	0	0	0	0	0
Algeria	12	5	2	8	15	3	0	0	0	0
Argentina	23	18	24	28	70	18	0	0	0	0
Armenia	5	1	2	9	12	6	0	0	0	0
Australasia	2	3	4	5	12	0	0	0	0	0

Question 0 (Example)

What is the first country in df?

This function should return a Series.

In [3]:
```
# You should write your whole answer within the function provided. The autograder will call
# this function and compare the return value against the correct solution value
def answer_zero():
    # This function returns the row for Afghanistan, which is a Series object. The assignment
    # question description will tell you the general format the autograder is expecting
    return df.iloc[0]

# You can examine what your function returns by calling it in the cell. If you have questions
# about the assignment formats, check out the discussion forums for any FAQs
answer_zero()
```

Out[3]:
```
# Summer           13
Gold                0
Silver              0
Bronze              2
Total               2
# Winter            0
Gold.1              0
Silver.1            0
Bronze.1            0
Total.1             0
# Games            13
Gold.2              0
Silver.2            0
Bronze.2            2
Combined total      2
ID                AFG
Name: Afghanistan, dtype: object
```

Question 1

Which country has won the most gold medals in summer games?

This function should return a single string value.

In [6]:
```
def answer_one():
    y = max(df['Gold'])
    ans = df[df['Gold'] == y].index.tolist()
    return ans[0]

answer_one()
```

Out[6]:
```
'United States'
```

Question 2

Which country had the biggest difference between their summer and winter gold medal counts?

This function should return a single string value.

In [7]:
```
def answer_two():
    x = max(df['Gold'] - df['Gold.1'])
    ans = df[(df['Gold'] - df['Gold.1']) == x].index.tolist()
    return ans[0]

answer_two()
```

Out[7]: 'United States'

Question 3

Which country has the biggest difference between their summer gold medal counts and winter gold medal counts relative to their total gold medal count?

$$\frac{Summer\ Gold - Winter\ Gold}{Total\ Gold}$$

Only include countries that have won at least 1 gold in both summer and winter.

This function should return a single string value.

In [12]:
```
def answer_three():
    df_gold = df[(df['Gold']>0) & (df['Gold.1']>0)]
    df_max_diff = (abs(df_gold['Gold'] - df_gold['Gold.1'])/df_gold['Gold.2'])
    return df_max_diff.idxmax()

answer_three()
```

Out[12]: 'Bulgaria'

Question 4

Write a function that creates a Series called "Points" which is a weighted value where each gold medal (`Gold.2`) counts for 3 points, silver medals (`Silver.2`) for 2 points, and bronze medals (`Bronze.2`) for 1 point. The function should return only the column (a Series object) which you created.

This function should return a Series named `Points` of length 146

In [13]:
```
def answer_four():
    points = 3*df['Gold.2'] + 2*df['Silver.2'] + 1*df['Bronze.2']
    return points

answer_four()
```

Out[13]:

Afghanistan	2
Algeria	27
Argentina	130
Armenia	16
Australasia	22
Australia	923
Austria	569
Azerbaijan	43
Bahamas	24
Bahrain	1
Barbados	1
Belarus	154
Belgium	276
Bermuda	1
Bohemia	5
Botswana	2
Brazil	184
British West Indies	2
Bulgaria	411
Burundi	3
Cameroon	12
Canada	846
Chile	24
China	1120
Colombia	29
Costa Rica	7
Ivory Coast	2
Croatia	67
Cuba	420
Cyprus	2
	...
Spain	268
Sri Lanka	4
Sudan	2
Suriname	4
Sweden	1217
Switzerland	630
Syria	6
Chinese Taipei	32
Tajikistan	4
Tanzania	4
Thailand	44
Togo	1
Tonga	2
Trinidad and Tobago	27
Tunisia	19
Turkey	191
Uganda	14
Ukraine	220
United Arab Emirates	3
United States	5684
Uruguay	16
Uzbekistan	38
Venezuela	18
Vietnam	4
Virgin Islands	2
Yugoslavia	171

```
Independent Olympic Participants       4
Zambia                                 3
Zimbabwe                              18
Mixed team                            38
dtype: int64
```

Part 2

For the next set of questions, we will be using census data from the United States Census Bureau (http://www.census.gov/popest/data/counties/totals/2015/CO-EST2015-alldata.html). Counties are political and geographic subdivisions of states in the United States. This dataset contains population data for counties and states in the US from 2010 to 2015. See this document (http://www.census.gov/popest/data/counties/totals/2015/files/CO-EST2015-alldata.pdf) for a description of the variable names.

The census dataset (census.csv) should be loaded as census_df. Answer questions using this as appropriate.

Question 5

Which state has the most counties in it? (hint: consider the sumlevel key carefully! You'll need this for future questions too...)

This function should return a single string value.

In [14]:
```
census_df = pd.read_csv('census.csv')
census_df.head()
```

Out[14]:

	SUMLEV	REGION	DIVISION	STATE	COUNTY	STNAME	CTYNAME	CENSUS2010POP
0	40	3	6	1	0	Alabama	Alabama	4779736
1	50	3	6	1	1	Alabama	Autauga County	54571
2	50	3	6	1	3	Alabama	Baldwin County	182265
3	50	3	6	1	5	Alabama	Barbour County	27457
4	50	3	6	1	7	Alabama	Bibb County	22915

5 rows × 100 columns

In [16]:
```
def answer_five():
    counties_df = census_df[census_df['SUMLEV'] == 50]
    x = counties_df.groupby('STNAME').count()['SUMLEV']
    ans = x.idxmax()
    return ans

answer_five()
```

Out[16]: 'Texas'

Question 6

Only looking at the three most populous counties for each state, what are the three most populous states (in order of highest population to lowest population)? Use `CENSUS2010POP`.

This function should return a list of string values.

In [19]:
```
def answer_six():
    counties_df = census_df[census_df['SUMLEV'] == 50]
    top_counties_df = counties_df.sort_values(by=['STNAME','CENSUS2010POP'],as
cending=False).groupby('STNAME').head(3)
    ans = top_counties_df.groupby('STNAME').sum().sort_values(by='CENSUS2010PO
P').head(3).index.tolist()
    return ans

answer_six()
```

Out[19]: ['Wyoming', 'Vermont', 'North Dakota']

Question 7

Which county has had the largest absolute change in population within the period 2010-2015? (Hint: population values are stored in columns POPESTIMATE2010 through POPESTIMATE2015, you need to consider all six columns.)

e.g. If County Population in the 5 year period is 100, 120, 80, 105, 100, 130, then its largest change in the period would be |130-80| = 50.

This function should return a single string value.

In [20]:
```
def answer_seven():
    counties_df = census_df[census_df['SUMLEV'] == 50]
    counties_df['pop_change'] = abs(counties_df['POPESTIMATE2015'] - counties_
df['POPESTIMATE2014'])+abs(counties_df['POPESTIMATE2014'] - counties_df['POPES
TIMATE2013'])+abs(counties_df['POPESTIMATE2013'] - counties_df['POPESTIMATE201
2'])+abs(counties_df['POPESTIMATE2012'] -
counties_df['POPESTIMATE2011'])+abs(counties_df['POPESTIMATE2011'] - counties_
df['POPESTIMATE2010'])
    a = max(counties_df['pop_change'])
    ans = counties_df['CTYNAME'][counties_df['pop_change']==a].tolist()
    return ans[0]


answer_seven()
```

Out[20]: 'Harris County'

Question 8

In this datafile, the United States is broken up into four regions using the "REGION" column.

Create a query that finds the counties that belong to regions 1 or 2, whose name starts with 'Washington', and whose POPESTIMATE2015 was greater than their POPESTIMATE 2014.

This function should return a 5x2 DataFrame with the columns = ['STNAME', 'CTYNAME'] and the same index ID as the census_df (sorted ascending by index).

In [21]:
```
def answer_eight():
    counties_df = census_df[census_df['SUMLEV'] == 50]
    ans = counties_df[((counties_df['REGION']==1)|(counties_df['REGION']==2))&
(counties_df['CTYNAME']=='Washington County')&
(counties_df['POPESTIMATE2015']>counties_df['POPESTIMATE2014'])][['STNAME','CT
YNAME']]
    return ans

answer_eight()
```

Out[21]:

	STNAME	CTYNAME
896	Iowa	Washington County
1419	Minnesota	Washington County
2345	Pennsylvania	Washington County
2355	Rhode Island	Washington County
3163	Wisconsin	Washington County

In [ ]:
```

```