

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](#) 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 derived from the Wikipedia entry on [All Time Olympic Games Medals](#), 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 [1]:

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]=='N':
        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 ID (take fir
st 3 characters from that)

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

Out[1]:

	# Summer	Gold	Silver	Bronze	Total	# Winter	Gold.1	Silver.1	Bronze.1	Total.1	# Games	Gold.2	Silver.2	Bronze
Afghanistan	13	0	0	2	2	0	0	0	0	0	13	0	0	
Algeria	12	5	2	8	15	3	0	0	0	0	15	5	2	
Argentina	23	18	24	28	70	18	0	0	0	0	41	18	24	2
Armenia	5	1	2	9	12	6	0	0	0	0	11	1	2	
Australasia	2	3	4	5	12	0	0	0	0	0	2	3	4	

### Question 0 (Example)

What is the first country in df?

This function should return a Series

In [2]:

```
# 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[2]:

```
# 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 [3]:

```
def answer_one():
    #1 = list((df.index[df['Gold'] == df['Gold'].max()]))[0]
    return df['Gold'].idxmax()
```

In [4]:

```
answer_one()
```

Out[4]:

```
'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 [5]:

```
def answer_two():
    diff = abs(df['Gold'] - df['Gold.1'])
    #return (diff.index[diff == max(diff)])[0]
    return diff.idxmax()
```

```
In [6]:
```

```
answer_two()
```

```
Out[6]:
```

```
'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?

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

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

*This function should return a single string value.*

```
In [7]:
```

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

```
In [8]:
```

```
answer_three()
```

```
Out[8]:
```

```
'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, with the country names as indices.

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

```
In [9]:
```

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

```
In [10]:
```

```
answer_four()
```

```
Out[10]:
```

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

Name: Points, Length: 146, dtype: int64

## Part 2

For the next set of questions, we will be using census data from the [United States Census Bureau](#). 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](#) 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 [11]:

```
import pandas as pd
```

```
census_df = pd.read_csv('census.csv')
census_df.head()
```

Out[11]:

	SUMLEV	REGION	DIVISION	STATE	COUNTY	STNAME	CTYNAME	CENSUS2010POP	ESTIMATESBASE2010	POPESTIM/
0	40	3	6	1	0	Alabama	Alabama	4779736	4780127	
1	50	3	6	1	1	Alabama	Autauga County	54571	54571	
2	50	3	6	1	3	Alabama	Baldwin County	182265	182265	
3	50	3	6	1	5	Alabama	Barbour County	27457	27457	
4	50	3	6	1	7	Alabama	Bibb County	22915	22919	

5 rows x 100 columns



In [12]:

```
def answer_five():
    county = census_df[census_df['SUMLEV'] == 50]
    group = census_df.groupby('STNAME').count()['CTYNAME']
    return group.idxmax()
```

In [13]:

```
answer_five()
```

Out[13]:

'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 [14]:

```
census_df['CENSUS2010POP']
```

Out[14]:

0 4779736
1 54571
2 182265
3 27457
4 22915
5 57322
6 10914
7 20947
8 118572
9 34215
10 25989
11 43643
12 13859
13 25833
14 13932
15 14972
16 49948
17 54428
18 13228
19 11539
20 27765

```
20      37765
21      13906
22      80406
23      50251
24      43820
25      71109
26      79303
27      38319
28     104430
29      17241
```

```
...
```

```
3163     131887
3164     389891
3165       52410
3166     24496
3167     166994
3168       74749
3169     563626
3170       36299
3171       11668
3172      46133
3173      15885
3174      13833
3175        7083
3176      40123
3177      13249
3178        4812
3179        8569
3180      91738
3181      18106
3182      75450
3183        2484
3184      28205
3185        8667
3186      29116
3187      10247
3188      43806
3189      21294
3190      21118
3191        8533
3192        7208
```

Name: CENSUS2010POP, Length: 3193, dtype: int64

In [15]:

```
def answer_six():
    census = census_df[census_df['SUMLEV'] == 50]
    census = census.groupby('STNAME')['CENSUS2010POP'].nlargest(3).reset_index()
    return list(census.groupby('STNAME').sum()['CENSUS2010POP'].nlargest(3).index)
```

In [16]:

```
answer_six()
```

Out[16]:

```
['California', 'Texas', 'Illinois']
```

## 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 [17]:

```
def answer_seven():
    census = census_df[census_df['SUMLEV'] == 50].copy()
    columns = ['POPESTIMATE2010', 'POPESTIMATE2011', 'POPESTIMATE2012', 'POPESTIMATE2013', 'POPESTIMATE2014', 'POPESTIMATE2015']
    census['change'] = abs(census[columns].max(axis=1) - census[columns].min(axis=1))
    return census['CTYNAME'].loc[census['change'].idxmax()]
```

In [18]:

```
answer_seven()
```

Out[18]:

```
'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 [19]:

```
def answer_eight():
    census = census_df[census_df['SUMLEV'] == 50]
    return census[((census['REGION'] == 1) | (census['REGION'] == 2)) & (census['CTYNAME'].str.startswith('Washington')) & (census['POPESTIMATE2015'] > census['POPESTIMATE2014'])].loc[:, ['STNAME', 'CTYNAME']]
```

In [20]:

```
answer_eight()
```

Out[20]:

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