Assignment 2

April 12, 2020

---

*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.*

---

1 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, 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 [65]: 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\(')
         df.index = names_ids.str[0]

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

```
Out[65]:              # Summer  Gold  Silver  Bronze  Total  # Winter  Gold.1  \
         Afghanistan        13     0       0       2      2         0       0
```

```
Algeria             12     5       2       8     15         3       0
Argentina           23    18      24      28     70        18       0
Armenia              5     1       2       9     12         6       0
Australasia          2     3       4       5     12         0       0

             Silver.1  Bronze.1  Total.1  # Games  Gold.2  Silver.2  Bronze.2  \
Afghanistan         0         0        0       13       0         0         2
Algeria             0         0        0       15       5         2         8
Argentina           0         0        0       41      18        24        28
Armenia             0         0        0       11       1         2         9
Australasia         0         0        0        2       3         4         5

             Combined total
Afghanistan               2
Algeria                  15
Argentina                70
Armenia                  12
Australasia              12
```

1.0.1 Question 0 (Example)

What is the first country in df?
This function should return a Series.

```
In [66]: # You should write your whole answer within the function provided. The autograder will
         # 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 assi
             # 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 que
         # about the assignment formats, check out the discussion forums for any FAQs
         answer_zero().name

Out[66]: 'Afghanistan'
```

1.0.2 Question 1

Which country has won the most gold medals in summer games?
This function should return a single string value.

```
In [67]: def answer_one():
             return df['Gold'].argmax()
         answer_one()

Out[67]: 'United States'
```

1.0.3 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 [68]: def answer_two():
             return (df['Gold'] - df['Gold.1']).idxmax()
         answer_two()
```

```
Out[68]: 'United States'
```

1.0.4 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 [69]: def answer_three():
             only_gold = df.where((df['Gold'] > 0) & (df['Gold.1'] > 0))
             only_gold = only_gold.dropna()
             return (abs((only_gold['Gold'] - only_gold['Gold.1']) / only_gold['Gold.2'])).idxma

         answer_three()
```

```
Out[69]: 'Bulgaria'
```

1.0.5 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 [70]: def answer_four():
             df['Points'] = (df['Gold.2'] * 3 + df['Silver.2'] * 2 + df['Bronze.2'] * 1)
             return df['Points']

         answer_four()
```

```
Out[70]: 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, dtype: int64
```

1.1 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.

1.1.1 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 [35]: census_df = pd.read_csv('census.csv')
         census_df.head()
```

```
Out[35]:    SUMLEV  REGION  DIVISION  STATE  COUNTY   STNAME         CTYNAME  \
         0      40       3         6      1       0  Alabama         Alabama
         1      50       3         6      1       1  Alabama  Autauga County
         2      50       3         6      1       3  Alabama  Baldwin County
         3      50       3         6      1       5  Alabama  Barbour County
         4      50       3         6      1       7  Alabama     Bibb County

            CENSUS2010POP  ESTIMATESBASE2010  POPESTIMATE2010     ...       \
         0        4779736            4780127          4785161     ...
         1          54571              54571            54660     ...
         2         182265             182265           183193     ...
         3          27457              27457            27341     ...
         4          22915              22919            22861     ...

            RDOMESTICMIG2011  RDOMESTICMIG2012  RDOMESTICMIG2013  RDOMESTICMIG2014  \
         0          0.002295         -0.193196          0.381066          0.582002
         1          7.242091         -2.915927         -3.012349          2.265971
         2         14.832960         17.647293         21.845705         19.243287
         3         -4.728132         -2.500690         -7.056824         -3.904217
         4         -5.527043         -5.068871         -6.201001         -0.177537
```

```
   RDOMESTICMIG2015  RNETMIG2011  RNETMIG2012  RNETMIG2013  RNETMIG2014  \
0         -0.467369     1.030015     0.826644     1.383282     1.724718
1         -2.530799     7.606016    -2.626146    -2.722002     2.592270
2         17.197872    15.844176    18.559627    22.727626    20.317142
3        -10.543299    -4.874741    -2.758113    -7.167664    -3.978583
4          0.177258    -5.088389    -4.363636    -5.403729     0.754533

   RNETMIG2015
0     0.712594
1    -2.187333
2    18.293499
3   -10.543299
4     1.107861

[5 rows x 100 columns]
```

```
In [71]: def answer_five():
             new_df = census_df[census_df['SUMLEV'] == 50]
             return new_df.groupby('STNAME').count()['SUMLEV'].idxmax()

         answer_five()
```

```
Out[71]: 'Texas'
```

1.1.2 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 [72]: def answer_six():
             new_df = census_df[census_df['SUMLEV'] == 50]
             most_populous_counties = new_df.sort_values('CENSUS2010POP', ascending=False).group
             return most_populous_counties.groupby('STNAME').sum().sort_values('CENSUS2010POP',

         answer_six()
```

```
Out[72]: ['California', 'Texas', 'Illinois']
```

1.1.3 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 [73]: def answer_seven():
             new_df1 = census_df[census_df['SUMLEV'] == 50][[6, 9, 10, 11, 12, 13, 14]]
             new_df1["MaxDiff"] = abs(new_df1.max(axis=1) - new_df1.min(axis=1))
             most_change = new_df1.sort_values(by=["MaxDiff"], ascending = False)
             return most_change.iloc[0][0]

         answer_seven()

Out[73]: 'Harris County'
```

1.1.4 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 [11]: def answer_eight():
             return "YOUR ANSWER HERE"
```