OLYMPIC HERO Project

Problem Statement

The Olympic Games, considered to be the world's foremost sports competition has more than 200 nations participating across the Summer and Winter Games alternating by occurring every four years but two years apart.

Throughout this project, we will explore the Olympics dataset(scraped from https://en.wikipedia.org/wiki/All-time_Olympic_Games_medal_table), look at some interesting statistics, and then try to find out which country is the King of the Olympic Games.

About the dataset

The snapshot of the data, you will be working on:

Country_Name	# Summer	Gold_Summer	Silver_Summer	Bronze_Summer	Total_Summer	# Winter	Gold_Winter	Silver_Winter	Bronze_Winter	Total_Winter	# Games	Gold_Total	Silver_Total	Bronze_Total	Total
Afghanistan	13	0	0	2	2	0	0	0	0	0	13	0	0	2	2
Algeria	12	5	2	8	15	3	0	0	0	0	15	5	2	8	15
Argentina	23	18	24	28	70	18	0	0	0	0	41	18	24	28	70
Armenia	5	1	2	9	12	6	0	0	0	0	11	1	2	9	12
Australasia	2	3	4	5	12	0	0	0	0	0	2	3	4	5	12
Australia	25	139	152	177	468	18	5	3	4	12	43	144	155	181	480
Austria	26	18	33	35	86	22	59	78	81	218	48	77	111	116	304
Azerbaijan	5	6	5	15	26	5	0	0	0	0	10	6	5	15	26
Bahamas	15	5	2	5	12	0	0	0	0	0	15	5	2	5	12
Bahrain	8	0	0	1	1	0	0	0	0	0	8	0	0	1	1
Barbados	11	0	0	1	1	0	0	0	0	0	11	0	0	1	1
Belarus	5	12	24	39	75	6	6	4	5	15	11	18	28	44	90

The dataset has details of 146 countries with the following 16 features

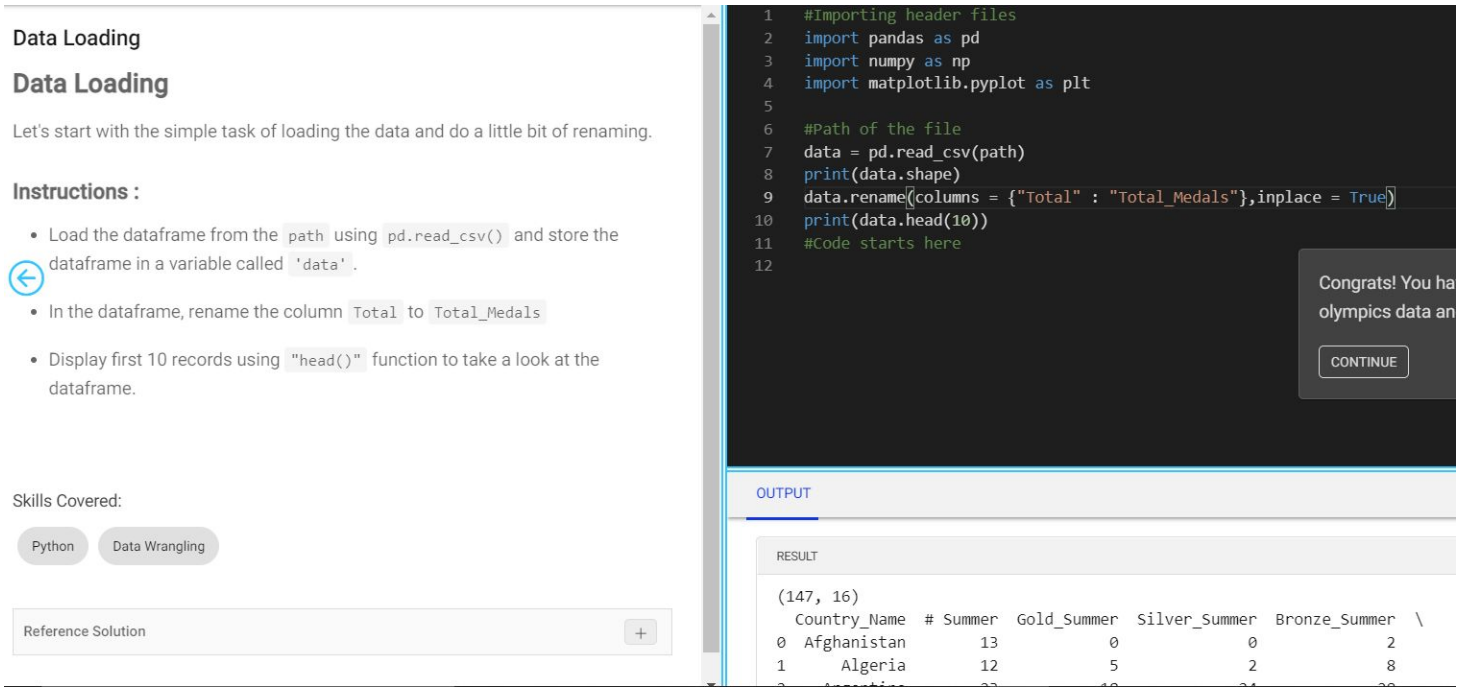
Feature	Description
Country_Name	Name of the country
# Summer	No. of games played in Summer Olympics
Gold_Summer	No. of gold medals won in Summer Olympics
Silver_Summer	No. of silver medals won in Summer Olympics
Bronze_Summer	No. of bronze medals won in Summer Olympics

Total_Summer	Total no. of all the medals won in Summer Olympics
# Winter	No. of games played in Winter Olympics
Gold_Winter	No. of gold medals won in Winter Olympics
Silver_Winter	No. of silver medals won in Winter Olympics
Bronze_Winter	No. of bronze medals won in Winter Olympics
Total_Winter	Total no. of all the medals won in Winter Olympics
# Games	Total no. of games played in both Summer and Winter Olympics
Gold_Total	Total no. of gold medals won in both Summer and Winter Olympics
Silver_Total	Total no. of silver medals won in both Summer and Winter Olympics
Bronze_Total	Total no. of bronze medals won in both Summer and Winter Olympics
Total	Total no. of all the medals won in both Summer and Winter Olympics

Why solve this project?

After completing this project, you will have a better understanding of data handling with python(pandas). In this project, you will be applying the following concepts :

1. Dataframe operations
2. Conditional statement and loops
3. List operations
4. Bar Plotting
5. Mathematical operations



Summer or Winter

Some Countries love Summer, some Winter. We think it has to do something with their Olympic performance.

For this task we will try to figure out which olympic event does a country perform better in.

Instructions :

- Create a new column `Better_Event` that stores `'Summer'`,`'Winter'` or `'Both'` based on the comparision between the total medals won in Summer event and Winter event (i.e. comparision between the `Total_Summer` and `Total_Winter` columns) using `"np.where()"`function.

```
        Example of np.where() function:


data = {'name': ['A', 'B', 'C', 'D', 'E'],
        'age': [12, 66, 22, 80, 7],
        'gender': ['M', 'F', 'F', 'M', 'M'],
        }
df = pd.DataFrame(data, columns = ['name', 'age', 'gender'])

print("dataframe before: \n",df)

"""
Creating a new column called senior_citizen where the value is yes
if df.age is greater than 60 and no if not
"""
```

```
df['senior_citizen'] = np.where(df['age']>=60, 'yes', 'no')
print("dataframe after:\n",df)
```

Output

```
dataframe before:

   name  age gender
0     A   12      M
1     B   66      F
2     C   22      F
3     D   80      M
4     E    7      M

dataframe after:
   name  age gender senior_citizen
0     A   12      M            no
1     B   66      F           yes
2     C   22      F            no
3     D   80      M           yes
4     E    7      M            no
```

- Find out which has been a better event with respect to all the performing countries by using `value_counts()` function and store it in a new variable called 'better_event'.

```
#Code starts here

#Creating new column 'Better_Event'
data['Better_Event'] = np.where(data['Total_Summer'] > data['Total_Winter'] ,
'Summer', 'Winter')
data['Better_Event'] = np.where(data['Total_Summer'] == data['Total_Winter'] , 'Both',
data['Better_Event'])

#Finding the value with max count in 'Better_Event' column
better_event=data['Better_Event'].value_counts().index.values[0]

#Printing the better event
print('Better_Event=', better_event)

#Code ends here
```

Top 10

So we figured out which is a better event for each country. Let's move on to finding out the best performing countries across all events

In this task we will try to find

- Which are the top 10 performing teams at summer event (with respect to total medals), winter event and overall?
- How many teams are present in all of the three lists above?

Instructions :

- Create a new dataframe subset called `'top_countries'` with the columns `['Country_Name','Total_Summer', 'Total_Winter','Total_Medals']` only
- Drop the last row from `'top_countries'`(The last row contains the sum of the medals)
- Create a function called `'top_ten'` that:
    - Takes the dataframe `'top_countries'` and a column name as parameters.
    - Creates a new empty list called 'country_list'
    - Find the top 10 values for that particular column(for e.g. `'Total_Summer'`) using `"nlargest()"` function
    - From the dataframe returned by nlargest function, slices the `Country_Name` column and stores it in the `'country_list'` list
    - Returns the `'country_list'`

Example of 'nlargest()' function :

```
df = pd.DataFrame({'ID': [1, 2, 3, 4, 5],
                   'Score': [33, 92, 26, 75, 80]})

print("The dataframe:\n",df)
# Filtering the 3 largest scores and getting the IDs associated with it
top_3=df.nlargest(3, 'Score')
print("df having top 3 scores:")
print(top_3)
print("IDs associated to top 3:")
print(list(top_3['ID']))
```

Output

```
The dataframe:
    ID  Score
0   1      33
1   2      92
2   3      26
3   4      75
4   5      80
df having top 3 scores:
   ID  Score
1   2      92
4   5      80
3   4      75
IDs associated to top 3:
[2, 5, 4]
```

Parameters :

parameter	dtype	Argument Type	default value	description
variable1	pandas.DataFrame	compulsory		dataframe to be loaded
variable2	string	compulsory		column name

Returns:

returns	dtype	description
variable1	list	list containing countries names

- Call the `'top_ten()'` function for the three columns :`Total_Summer`,`Total_Winter` and `Total_Medals` and store their respective results in lists called `'top_10_summer'`, `'top_10_winter'` and `'top_10'`
- Create a new list `'common'` that stores the common elements between the three lists(`'top_10_summer'`, `'top_10_winter'` and `'top_10'`)

```
#Code starts here

#Subsetting the dataframe
top_countries=data[['Country_Name','Total_Summer', 'Total_Winter','Total_Medals']]

#Dropping the last row
top_countries=top_countries[:-1]

#Function for top 10
def top_ten(data, col):

   #Creating a new list
   country_list=[]

   #Finding the top 10 values of 'col' column
   country_list= list((data.nlargest(10,col)['Country_Name']))

   #Returning the top 10 list
   return country_list
```

```
#Calling the function for Top 10 in Summer
top_10_summer=top_ten(top_countries,'Total_Summer')
print("Top 10 Summer:\n",top_10_summer, "\n")

#Calling the function for Top 10 in Winter
top_10_winter=top_ten(top_countries,'Total_Winter')
print("Top 10 Winter:\n",top_10_winter, "\n")

#Calling the function for Top 10 in both the events
top_10=top_ten(top_countries,'Total_Medals')
print("Top 10:\n",top_10, "\n")

#Extracting common country names from all three lists
common=list(set(top_10_summer) & set(top_10_winter) & set(top_10))

print('Common Countries :\n', common, "\n")

#Code ends here
```

Plotting Top 10

From the lists that you have created from the previous task, let's plot the medal count of the top 10 countries for better visualisation

Instructions :

- Take the three previously created lists(top_10_summer, top_10_winter, top_10)
- Subset the dataframe `'data'` based on the country names present in the list `top_10_summer` using `"isin()"` function on the column `Country_Name`. Store the new subsetted dataframes in `'summer_df'`. Do the similar operation using `top_10_winter` and `top_10` and store the subset dataframes in `'winter_df'` & `'top_df'` respectively.

Example of isin() function:

```
df = pd.DataFrame({'A': [1, 2, 3, 4, 5], 'B': ['Alpha', 'Beta', 'Gamma','Delta','Epsilon'],
'C':[1, 4, 9, 16, 25]})
#List
List= ['Beta','Epsilon']

#Usage of isin() function
subset_df=df[df['B'].isin(List)]
```

```
print(subset_df)
```

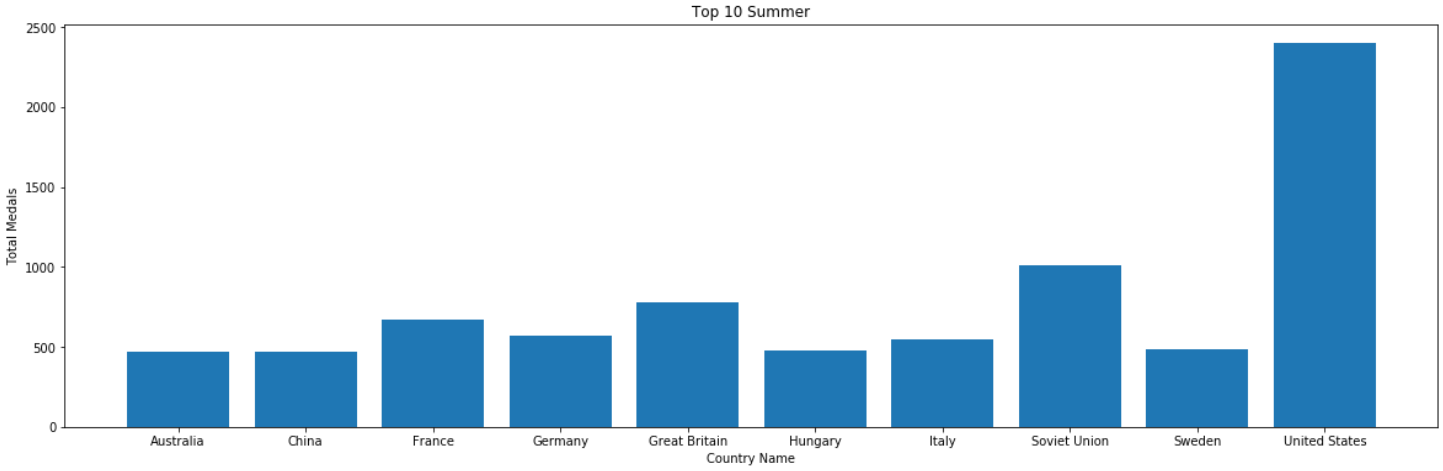
Output

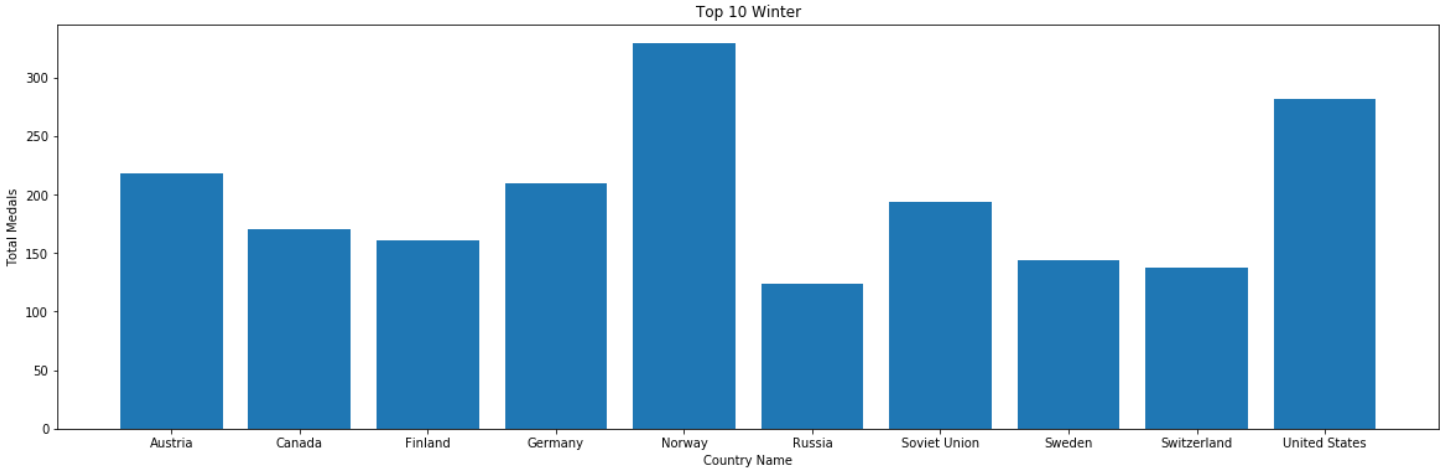
```
   A        B   C
1  2     Beta   4
4  5  Epsilon  25
```

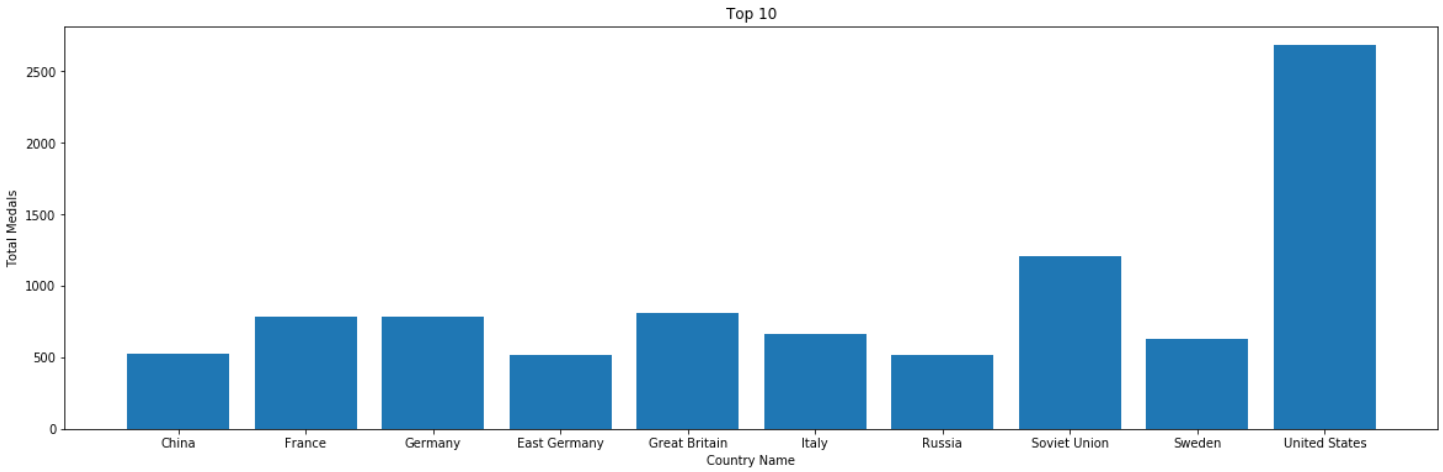
- Take each subsetted dataframe and plot a bar graph between the country name and total medal count according to the event (For e.g. for `'summer_df'` plot a bar graph between `Country_Name` and `Total_Summer`)
- Modify the axes info accordingly.

```
#Code starts here

#For Summer

#Creating the dataframe for Summer event
summer_df= data[data['Country_Name'].isin(top_10_summer)]

#Plotting the bar graph
plt.figure(figsize=(20, 6))
plt.bar(summer_df['Country_Name'], summer_df['Total_Summer'])

#Changing the graph title
plt.title('Top 10 Summer')

#Changing the x-axis label
plt.xlabel('Country Name')

#Changing the y-axis label
plt.ylabel('Total Medals')


#For Winter

#Creating the dataframe for Winter event
winter_df=data[data['Country_Name'].isin(top_10_winter)]

#Plotting the bar graph
plt.figure(figsize=(20, 6))
plt.bar(winter_df['Country_Name'], winter_df['Total_Winter'])
```

```
#Changing the graph title
plt.title('Top 10 Winter')

#Changing the x-axis label
plt.xlabel('Country Name')

#Changing the y-axis label
plt.ylabel('Total Medals')


#For both the events

#Creating the dataframe for both the events
top_df=data[data['Country_Name'].isin(top_10)]

#Plotting the bar graph
plt.figure(figsize=(20, 6))
plt.bar(top_df['Country_Name'], top_df['Total_Medals'])

#Changing the graph title
plt.title('Top 10')

#Changing the x-axis label
plt.xlabel('Country Name')

#Changing the y-axis label
plt.ylabel('Total Medals')
```







Top-performing country(Gold)

Winning silver or bronze medals is a big achievement but winning gold is bigger.

Using the above created dataframe subsets, in this task let's find out which country has had the best performance with respect to the ratio between gold medals won and total medals won.

Instructions :

- In the dataframe `'summer_df'`(created in the previous function) , create a new column `Golden_Ratio` which is the quotient after dividing the two columns `Gold_Summer` and `Total_Summer`.
- Find the max value of `Golden_Ratio` and the country associated with it and store them in `summer_max_ratio` and `summer_country_gold` respectively.
- In the dataframe `'winter_df'`(created in the previous function) , create a new column `Golden_Ratio` which is the quotient after dividing the two columns `Gold_Winter` and `Total_Winter`.
- Find the max value of `Golden_Ratio` and the country associated with it and store them in `'winter_max_ratio'` and `'winter_country_gold'` respectively.
- In the dataframe `top_df'`(created in the previous function) , create a new column `Golden_Ratio` which is the quotient after dividing the two columns `Gold_Total` and `Total_Medals`.

- Find the max value of `Golden_Ratio` and the country associated with it and store them in `top_max_ratio'` and `'top_country_gold'` respectively.

```
#Code starts here

#For Summer List

#Creating new column 'Golden_Ratio'
summer_df['Golden_Ratio']=summer_df['Gold_Summer']/summer_df['Total_Summer']


#Finding the max value of 'Golden_Ratio' column
summer_max_ratio=max(summer_df['Golden_Ratio'])

#Finding the country assosciated with the max value of 'Golden_Ratio' column
summer_country_gold=summer_df.loc[summer_df['Golden_Ratio'].idxmax(),'Country_Name']

print("Top Summer Country:", summer_country_gold, " with a ratio of %.2f"
%summer_max_ratio )


#For Winter List

#Creating new column 'Golden_Ratio'
winter_df['Golden_Ratio']=winter_df['Gold_Winter']/winter_df['Total_Winter']


#Finding the max value of 'Golden_Ratio' column
winter_max_ratio=max(winter_df['Golden_Ratio'])

#Finding the country assosciated with the max value of 'Golden_Ratio' column
winter_country_gold=winter_df.loc[winter_df['Golden_Ratio'].idxmax(),'Country_Name']

print("Top Winter Country:", winter_country_gold, " with a ratio of %.2f"
%winter_max_ratio )


#For Overall List

#Creating new column 'Golden_Ratio'
top_df['Golden_Ratio']=top_df['Gold_Total']/top_df['Total_Medals']

#Finding the max value of 'Golden_Ratio' column
top_max_ratio=max(top_df['Golden_Ratio'])

#Finding the country assosciated with the max value of 'Golden_Ratio' column
```

```
top_country_gold=top_df.loc[top_df['Golden_Ratio'].idxmax(),'Country_Name']

print("Top Country:", top_country_gold, " with a ratio of %.2f" %top_max_ratio )


#Code ends here
```

```
Top Summer Country: China  with a ratio of 0.42
Top Winter Country: Soviet Union  with a ratio of 0.40
Top Country: China  with a ratio of 0.40
```

Best in the world

Winning Gold is great but is winning most gold equivalent to being the best overall perfomer? Let's find out.

Instructions :

- Drop the last row from the dataframe(The last row contains the total of all the values calculated vertically) and save the result in `'data_1'`
- Update the dataframe `'data_1'` to include a new column called `Total_Points` which is a weighted value where each gold medal counts for 3 points, silver medals for 2 points, and bronze medals for 1 point.(i.e. You need to take weighted value of `Gold_Total`, `Silver_Total` and `Bronze_Total`)
- Find the max value of `Total_Points` in `'data_1'` and the country assosciated with it and store it in variables `'most_points'` and `'best_country'` respectively.

```
#Code starts here

#Removing the last column of the dataframe
data_1=data[:-1]

#Creating a new column 'Total_Points'
data_1['Total_Points']= data_1['Gold_Total']*3 + data_1['Silver_Total']*2 +
data_1['Bronze_Total']*1
# Use of position index to handle the ambiguity of having same name columns

#Finding the maximum value of 'Total_Points' column
most_points=max(data_1['Total_Points'])

#Finding the country assosciated with the max value of 'Total_Column' column
best_country=data_1.loc[data_1['Total_Points'].idxmax(),'Country_Name']
print('The maximum points achieved is ', most_points, ' by ', best_country )

#Code ends here
```

```
Total Points scored by Best Country is: 5684
Top Country: United States  with total points  0.4049429657794677
```

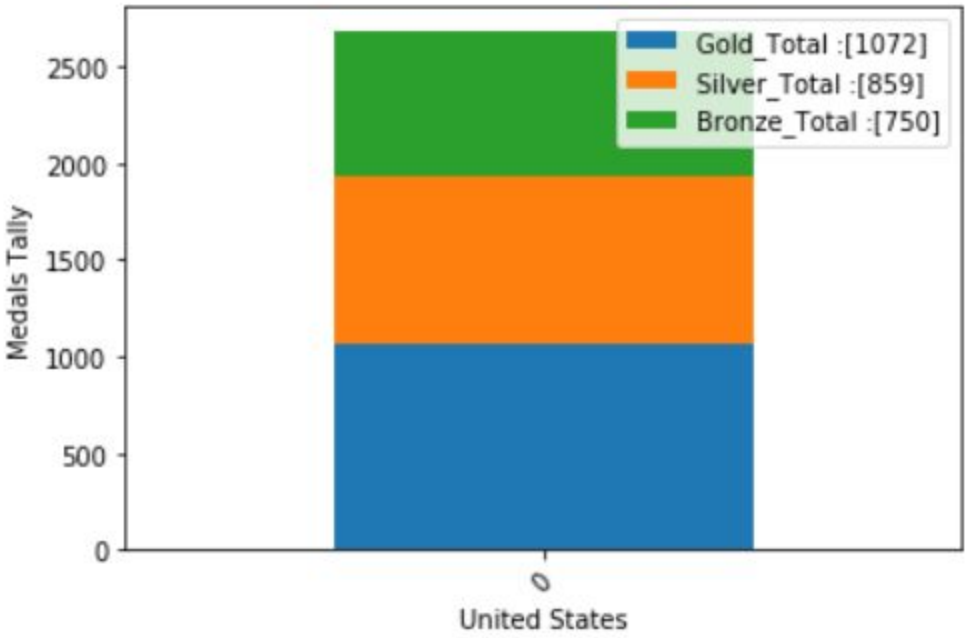
Plot for the best

We know which country is best when it comes to winning the most points in Olympic Games. Let's plot the medal count to visualise their success better.

Instructions

- Create a single row dataframe called `'best'` from `'data'` where value of column `Country_Name` is equal to `'best_country'`(The variable you created in the previous task)
- Subset `'best'` even further by only including the columns : `['Gold_Total','Silver_Total','Bronze_Total']`
- Create a stacked bar plot of `'best'` using `"DataFrame.plot.bar()"` function
- Name the x-axis as `United States` using `"plt.xlabel()"`
- Name the y-axis as `Medals Tally` using `"plt.ylabel()"`
- Rotate the labels of x-axis by 45o using `"plt.xticks()"`

```
#Code starts here

#Subsetting the dataframe
best=data[data['Country_Name']==best_country]
best.reset_index(drop = True, inplace = True)
best=best[['Gold_Total','Silver_Total','Bronze_Total']]

#Plotting bar plot
best.plot.bar(stacked=True)

#Changing the x-axis label
plt.xlabel('United States')

#Changing the y-axis label
plt.ylabel('Medals Tally')

#Rotating the ticks of X-axis
plt.xticks(rotation=45)

#Updating the graph legend
l=plt.legend()
l.get_texts()[0].set_text('Gold_Total :' + str(best['Gold_Total'].values))
l.get_texts()[1].set_text('Silver_Total :' + str(best['Silver_Total'].values))
l.get_texts()[2].set_text('Bronze_Total :' + str(best['Bronze_Total'].values))

#Code ends here
```


Gold_Total :[1072]
Silver_Total :[859]
Bronze_Total :[750]
Medals Tally
2500
2000
1500
1000
500
0
0
United States