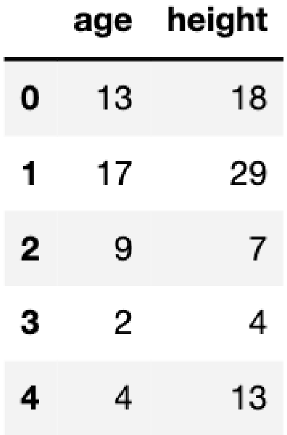
Data 100, Spring 2023

Exam Prep Section #1 Solutions

Pandas

1. Suppose we have the following DataFrame about pandas (the animal) shown below. We will calculate various statistics about pandas using the `pandas` library. In the end, we will hopefully understand pandas and `pandas` better!

   The DataFrame `panda` contains two columns: `age` and `height` for 10,000 pandas across the world.



	age	height
0	13	18
1	17	29
2	9	7
3	2	4
4	4	13

   (a) We first want to know if there is a relationship between a panda's age and height. Write a one-line expression to calculate the average ratio between a panda's height and age for all pandas in our DataFrame.

   Solution:

   ```
   (panda['height'] / panda['age']).mean()
   ```

   (b) Instead, suppose we decide to calculate the median height for each age. Write a one-line `pandas` expression to accomplish this.

   Solution:

   ```
   panda.groupby('age')['height'].median()
   ```

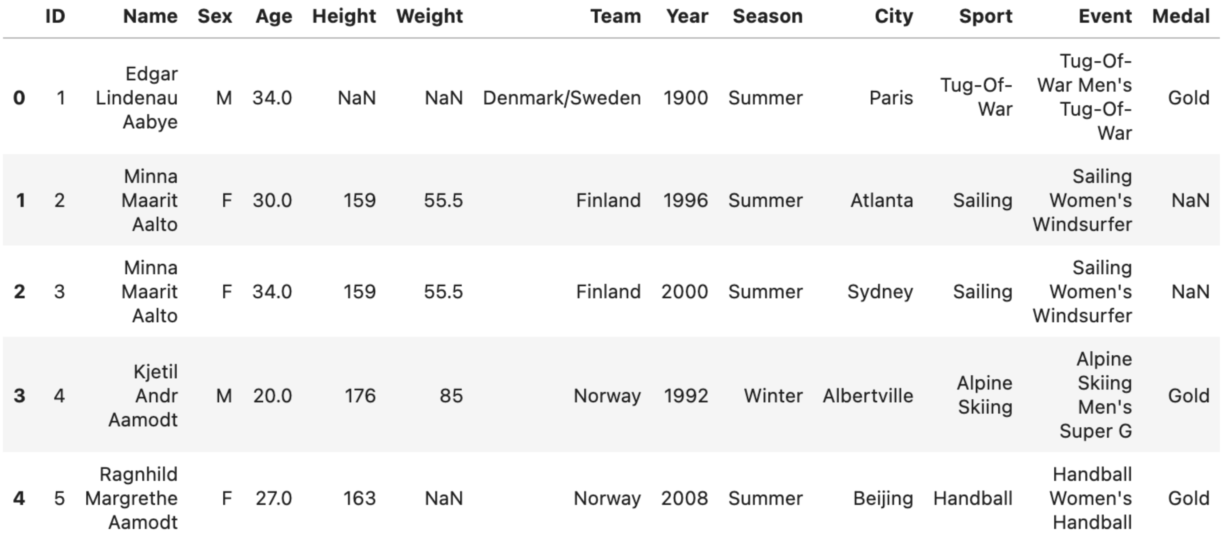
   (c) Expanding on the expression from the previous section, calculate the median ratio between a panda's height and age for each age. Write `pandas` code to accomplish this.

Solution:

```
panda['height/age'] = panda['height'] / panda['age']
panda.groupby('age')['height/age'].agg('median')
```

Pandas: Olympics

2. We will work with an Olympics dataset containing the names of all athletes who participated in the Olympic Games, including all the Games from Athens 1896 to Tokyo 2020. We refer to this dataframe as `ath`. The first 5 lines of the table are shown below. You may assume that the ID column is unique to each athlete and that the only column with null values are height, weight, and medal.



	ID	Name	Sex	Age	Height	Weight	Team	Year	Season	City	Sport	Event	Medal
0	1	Edgar Lindenau Aabye	M	34.0	NaN	NaN	Denmark/Sweden	1900	Summer	Paris	Tug-Of-War	Tug-Of-War Men's Tug-Of-War	Gold
1	2	Minna Maarit Aalto	F	30.0	159	55.5	Finland	1996	Summer	Atlanta	Sailing	Sailing Women's Windsurfer	NaN
2	3	Minna Maarit Aalto	F	34.0	159	55.5	Finland	2000	Summer	Sydney	Sailing	Sailing Women's Windsurfer	NaN
3	4	Kjetil Andr Aamodt	M	20.0	176	85	Norway	1992	Winter	Albertville	Alpine Skiing	Alpine Skiing Men's Super G	Gold
4	5	Ragnhild Margrethe Aamodt	F	27.0	163	NaN	Norway	2008	Summer	Beijing	Handball	Handball Women's Handball	Gold

(a) Write `pandas` code to returns the 10 most common middle names among gold medal winners as `pandas` Series. The column `Name` contains the first name, the middle name, and the last name of each athlete that are separated by space. You may assume that all athletes in the table have a middle name.

Solution:

```
ath.loc[ath[’Medal’] == ’Gold’, ’Name’] \
    .str.split().str[1] \
    .value_counts().index[:10]
```

(b) Which of the following lines of Pandas code will output the the most number of medals won by a single athlete?

□ A. `ath[’ID’].groupby(’ID’).count().iloc[0]`

□ B. `ath[[’ID’]].groupby(’ID’).count().iloc[0]`

□ C. `ath[[’Name’, ’Medal’]].groupby(’Name’).count().sort_values(’Medal’, ascending = False).iloc[0]`

- □ D. `ath[['ID', 'Medal']].groupby('ID').size().iloc[0]`
- □ E. `ath[['Name', 'Medal']].groupby('Name').sum().iloc[0]`
- □ F. `ath[['Name', 'Medal']].groupby('Name').count().iloc[0].sort_values( 'Medal', ascending = False)`

Staff Notes:

You can assume all athlete's names are unique.

(c) What are the oldest athletes to participate in each sport along with the corresponding year in which they participated?

Solution:

```
ath.sort_values('Age', ascending=False).groupby('Sport') \
.agg({'Year':'first','Name':'first'})
```

Staff Notes:

For this question, the 'corresponding year' referred to the final year that the athlete participated.

(d) Fill in the blanks below to return the names of all the athletes who won a medal after a gap of 10 years or more of not winning any Olympics medals. You may assume that each individual's name is unique to them.

```
def filter_func(subframe):
    return ____________________________________________________

ath.sort_values(_____________________) \
   [_____________________] \
   .groupby(_________) \
   .filter(filter_func)['Name'] \
   .unique()
```

Solution:

```
def filter_func(subframe):
    return (subframe['Year'].diff() >= 10).any()
    # note that pd.diff is not required!
    return (subframe['Year'].iloc[:-1].values + 10 <=
            subframe['Year'].iloc[1:].values).any()

ath.sort_values('Year') \
   [~ath['Medal'].isna()] \
   .groupby('Name') \
   .filter(filter_func)['Name'] \
   .unique()
```

Staff Notes:

While the question's format doesn't allow it, the best practice is to always do the boolean filtering first. Here, doing sort_values() first produces a warning.