Q1. Mean Income

Problem Statement:

Given a dataframe **df** having the income details for different individuals.

Return a Series that contains the **gender-wise average income**.

Input Format: A DataFrame

Output Format: A DataFrame

Sample Input:

	name	gender	income
0	Elon	M	53000.0
1	Jeff	F	28000.0
2	Bill	M	25000.0
3	Falguni	F	44000.0

Sample Output:

```
          income
gender
F        36000.0
M        39000.0
```

Note:

- Recall **group by** in pandas.
- We are expecting the function to return a **Series**.

Use the code below in order to create the given dataframe:

import pandas as pd

data = {

'name': ['Elon', 'Jeff', 'Bill', 'Falguni'],

'gender': ['M', 'F', 'M', 'F'],

'income': [53000.0, 28000.0, 25000.0, 44000.0]

}

df = pd.DataFrame(data)

```
import pandas as pd
def calc_income(df):
    '''
    INPUT: df - > dataframe

    OUTPUT: result -> dataframe
    '''
    result = _____

    return result
```

Q2. Groupby and drop

	school_code	class	name	age	height	weight
S1	s001	V	Alberto Franco	12	173	35
S2	s002	V	Gino Mcneill	12	192	32
S3	s003	VI	Ryan Parkes	13	186	33
S4	s001	VI	Eesha Hinton	13	167	30
S5	s002	V	Gino Mcneill	14	151	31
S6	s004	VI	David Parkes	12	159	32
S7	s001	IX	Jane	15	140	40
S8	s003	X	Martin	20	160	50

Student_data

The code for above dataframe can be downloaded from **here**.

```
#Block a
grouped_data = student_data.groupby(['school_code', 'class'])
print((grouped_data.size()))


#Block b
n = 1
rows_to_drop=student_data.groupby(['school_code']).tail(n).index
result1 = student_data.drop(rows_to_drop, axis=0)
print(result1)
```

For the above dataframe **student_data,** which of the mentioned statements are correct wrt code in block **a** and **b**?

The statements are:

1. Code in block **a**, prints the number of observations in all the unique 'school_code' and 'class' combinations.

2. Code in block **a**, prints the number of observations in all the unique 'class' values.

3. Code in block **b**, prints all the observations of each 'school_code' except the first one observation in each unique school_code.

4. Code in block **b**, prints all the observations of each 'school_code' except the last one observation in each unique school_code.

Options:

- **A. 1 and 4.**
- **B. 1 and 3.**
- **C. 2 and 3.**
- **D. 2 and 4.**

Q3. On your marks

Problem Statement: Given a dataframe containing student's marks for multiple subjects, return a dataframe which contains records of **subject-wise highest marks** along with their **roll number**.

Input Format: A DataFrame

Output Format: A DataFrame

Sample Input:

	roll_no	subject	marks
0	1	NN	97
1	2	DL	63
2	1	ML	63
3	3	Prob	71
4	1	DL	64
5	3	ML	90
6	3	DL	66
7	3	NN	46
8	2	NN	74
9	2	Prob	62
10	1	Prob	94
11	2	ML	67

Sample Output:

	subject	marks	roll_no
0	DL	66	3
1	ML	90	3
2	NN	97	1
3	Prob	94	1

Note: Recall how we **group** and **merge** data in Pandas.

Use the code below in order to create the given dataframe:

```
import pandas as pd

data = {
   'roll_no': [1, 2, 1, 3, 1, 3, 3, 3, 2, 2, 1, 2],
   'subject': ['NN', 'DL', 'ML', 'Prob', 'DL', 'ML', 'DL', 'NN', 'NN', 'Prob', 'Prob', 'ML'],
   'marks': [97, 63, 63, 71, 64, 90, 66, 46, 74, 62, 94, 67]
}
df = pd.DataFrame(data)
```

```python
import pandas as pd

def get_marks(df):
    '''
    INPUT: df -> dataframe

    OUTPUT: result -> dataframe
    '''

    ### STEP 1: group the df using subject and get max marks

    max_marks = ____

    ### STEP 2: reset the index

    max_marks = _____


    ### STEP 3: merge with original df to get roll id

    result = ______
```

Q4. Group the Data

Problem Description:

Given a dataset about salesperson and customer orders, the task is to group the data as per salesman_id and customer_id, and return a new dataframe with the ord_no count for each salesman_id and customer_id pair.

- Refer to sample input and output for better understanding.

Input Format: The first line of the input contains the dataframe, in the form of a dictionary.

Output Format: The new dataframe

Sample Input:

	ord_no	purch_amt	ord_date	customer_id	salesman_id
0	70001	150.50	2012-10-05	3005	5002
1	70009	270.65	2012-09-10	3001	5005
2	70002	65.26	2012-10-05	3002	5001
3	70004	110.50	2012-08-17	3009	5003
4	70007	948.50	2012-09-10	3005	5002
5	70005	2400.60	2012-07-27	3007	5001
6	70008	5760.00	2012-09-10	3002	5001
7	70010	1983.43	2012-10-10	3004	5006
8	70003	2480.40	2012-10-10	3009	5003
9	70012	250.45	2012-06-27	3008	5002
10	70011	75.29	2012-08-17	3003	5007
11	70013	3045.60	2012-04-25	3002	5001

Sample Output:

	salesman_id	customer_id	ord_no
0	5001	3002	3
1	5001	3007	1
2	5002	3005	2
3	5002	3008	1
4	5003	3009	2
5	5005	3001	1
6	5006	3004	1
7	5007	3003	1

Sample Explanation:

The data is grouped by salesman_id and customer_id, with the count of each (as in salesman_id 5001 with customer_id 3002 appears 3 times, and so on).

Use the code below in order to create the given dataframe:

import pandas as pd

data = {

'ord_no': [70001, 70009, 70002, 70004, 70007, 70005, 70008, 70010, 70003, 70012, 70011, 70013],

'purch_amt': [150.50, 270.65, 65.26, 110.50, 948.50, 2400.60, 5760.00, 1983.43, 2480.40, 250.45, 75.29, 3045.60],

'ord_date': ['2012-10-05', '2012-09-10', '2012-10-05', '2012-08-17', '2012-09-10', '2012-07-27', '2012-09-10', '2012-10-10', '2012-10-10', '2012-06-27', '2012-08-17', '2012-04-25'],

'customer_id': [3005, 3001, 3002, 3009, 3005, 3007, 3002, 3004, 3009, 3008, 3003, 3002],

'salesman_id': [5002, 5005, 5001, 5003, 5002, 5001, 5001, 5006, 5003, 5002, 5007, 5001]

}

df = pd.DataFrame(data)

```
import pandas as pd

def group_the_data(df):
    '''
    input:
    df -> the dataframe provided to the function

    output:
    result -> the grouped by data as per required in the question
    '''
    result = None

    # Your Code Starts here

    # Your Code ends here
    return result
```

Q5. Super Valuable Customers

Problem Description:

We have access to Amazon's sales data.

Our objective is to filter out user users who have purchased a minimum of '**x**' quantity of items and ensure that the average price of the items they've bought is at least '**y**'.

Values of **x** and **y** will be provided to the function.

Input Desription: A dataframe, an integer x and a float y

Output Description: A dataframe of the filtered values, with only the "Name" and "Purchase Address" columns.

Sample Input:

	Name	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
0	Julie Dsouza	Apple Airpods Headphones	1	150.00	01/22/19 21:20	868 Willow St, Los Angeles, CA 90001
1	Kelly Sebastian	Apple Airpods Headphones	1	150.00	01/24/19 8:13	442 Cedar St, Portland, OR 97035
2	Daniel Fernandez	27in 4K Gaming Monitor	1	389.99	01/26/19 12:16	741 10th St, Los Angeles, CA 90001
3	Julie Dsouza	Bose SoundSport Headphones	1	99.99	01/01/19 10:30	867 Willow St, Los Angeles, CA 90001
4	John Jacob	Wired Headphones	2	11.99	01/17/19 13:33	946 Walnut St, Boston, MA 02215

x = 2

y = 100

Sample Output:

	Name	Purchase Address
0	Julie Dsouza	868 Willow St, Los Angeles, CA 90001
3	Julie Dsouza	867 Willow St, Los Angeles, CA 90001

Sample Explanation:

- From the given dataframe, people who have bought atleast 2 items are Julie Dsouza (2 different headphones) and John Jacob (2 Wired headphones).
- Then from these, the average price of products of Julie is **124.5** ((150+99.99) /2) while of John is **11.99**.
- Hence, the result are the name and address of the user "**Julie Dsouza**".

Use the code below in order to create the given dataframe:

import pandas as pd

data = {

'Name': ['Julie Dsouza', 'Kelly Sebastian', 'Daniel Fernandez', 'Julie Dsouza', 'John Jacob'],

'Product': ['Apple Airpods Headphones', 'Apple Airpods Headphones', '27in 4K Gaming Monitor', 'Bose SoundSport Headphones', 'Wired Headphones'],

'Quantity Ordered': [1, 1, 1, 1, 2],

'Price Each': [150.00, 150.00, 389.99, 99.99, 11.99],

'Order Date': ['01/22/19 21:20', '01/24/19 8:13', '01/26/19 12:16', '01/01/19 10:30', '01/17/19 13:33'],

'Purchase Address': ['868 Willow St, Los Angeles, CA 90001', '442 Cedar St, Portland, OR 97035', '741 10th St, Los Angeles, CA 90001', '867 Willow St, Los Angeles, CA 90001', '946 Walnut St, Boston, MA 02215']

}

df = pd.DataFrame(data)

```
import pandas as pd
def filter_the_customers(df, x, y):
    '''
    Input:
    df -> The input sales data
    x -> The minimum quantity required
    y -> The average price required

    Output:
    Return the filtered dataframe
    '''
    # Your Code Starts here


    return df
```

Q6. Road runner

Problem Statement:

Given a dataframe containing the distance ran by track runner (in 15 sec) for various attempts.

We want to find

- for each runner, count of attempts in which the distance covered by runner was greater than the his/her average distance.

Input Format:

A DataFrame

Output Format:

A Series containing the required count of attempts.

Sample Input:

	runner	distance
0	runner1	82
1	runner2	101
2	runner3	84
3	runner4	106
4	runner1	93
5	runner2	86
6	runner3	87
7	runner4	82
8	runner1	92
9	runner2	86
10	runner3	98
11	runner4	84

Sample Output:

```
runner1    2
runner2    1
runner4    1
runner3    1
Name: runner, dtype: int64
```

Sample Explanation:

Runner 1 took 3 attempts i.e. 82, 93, 92.

The average of 3 attempts will be 89.

Runner 1 has 2 attempts for which the distance covered is more than his/her average.

Similarly, the count for each runner is returned.

Note: Recall group by transform

```python
import pandas as pd

def get_count(df):
    '''
    INPUT: df -> dataframe

    OUTPUT: result -> series
    '''
    ### CODE STARTS HERE



    return result
```

Q7. Categorise ages

You have a data frame that consists of the names and ages of a few candidates. You need to convert age into categories according to the following conditions:
1. If age<18: **kid**
2. age >=18 and <=50: **adult**
3. age>50 **senior**
Write a python program to do so using apply function.

Input Format:

A dataframe

Output Format:

A dataframe

Sample Input:

{'name':["Ram","Shyam","Mukesh","Suresh"],'age':[10,18,60,50]}

Sample Output:

	name	age
0	Ram	kid
1	Shyam	adult
2	Mukesh	senior
3	Suresh	adult

```
import pandas as pd
def replace_ages(a):
    '''a is a dataframe with columns ['name', 'age']
       Output A dataframe with same columns but with changed records as per required is expected to be
returned'''

    # YOUR CODE GOES HERE
```

Q8. Haven't followed the instructions?

df

	name	username	userid
0	Jim	itsjimhere	20
1	Clarke	clark002	10
2	Kent	itskentment	86
3	Mark	markyoumustknow	21

In the above-given dataframe, **usernames** are the display names of the learner's accounts shown on a video-conferencing app when taking classes.

The instructor clearly said to the learners that usernames must contain the real name (**name**) of the learners.

We tried to print the **userid** of the learners who haven't followed the instruction:

```
def check(name,username):
   return name.lower() in username

df[~(df.apply(lambda x:check(x['name'],x['username']), axis=0))]['userid']
```

Note: 'username' will be in lowercase.

We found that we are **not** getting the required results using the code, what can be the **reason** here?

Use the code below in order to create the given dataframe:

```
import pandas as pd
data = {
   'name': ['Jim', 'Clarke', 'Kent', 'Mark'],
   'username': ['itsjimhere', 'clark002', 'itskentment', 'markyoumustknow'],
   'userid': [20, 10, 86, 21]
}
df = pd.DataFrame(data)
```

A. **Equate the axis to 1.**
B. **Remove the column ['userid'] from the end.**
C. **The .lower() function is not correct here.**
D. **~ (tilde), remove this symbol, it is the issue.**