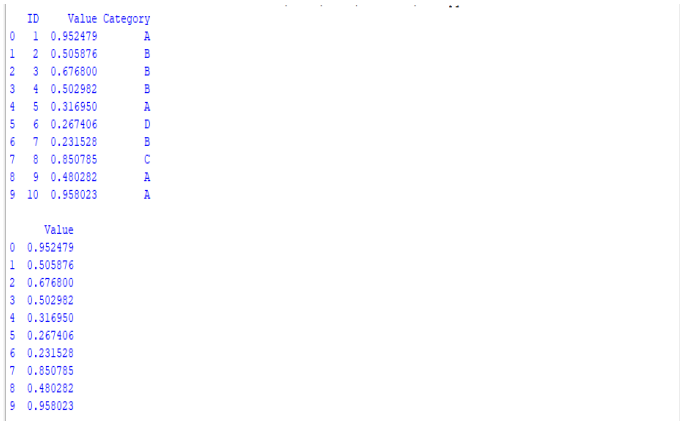
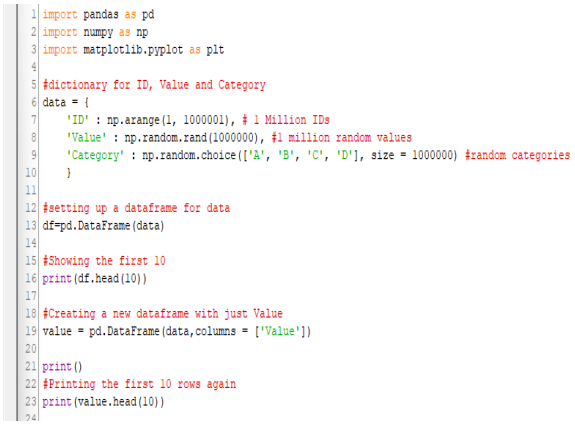
Imports and output for first couple of problems



```
   ID     Value Category
0   1  0.952479        A
1   2  0.505876        B
2   3  0.676800        B
3   4  0.502982        B
4   5  0.316950        A
5   6  0.267406        D
6   7  0.231528        B
7   8  0.850785        C
8   9  0.480282        A
9  10  0.958023        A

      Value
0  0.952479
1  0.505876
2  0.676800
3  0.502982
4  0.316950
5  0.267406
6  0.231528
7  0.850785
8  0.480282
9  0.958023
```



```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

#dictionary for ID, Value and Category
data = {
    'ID' : np.arange(1, 1000001), # 1 Million IDs
    'Value' : np.random.rand(1000000), #1 million random values
    'Category' : np.random.choice(['A', 'B', 'C', 'D'], size = 1000000) #random categories
    }

#setting up a dataframe for data
df=pd.DataFrame(data)

#Showing the first 10
print(df.head(10))

#Creating a new dataframe with just Value
value = pd.DataFrame(data,columns = ['Value'])

print()
#Printing the first 10 rows again
print(value.head(10))
```

Code for student data

```python
#Code as given in the assignment
pd.set_option('display.max_rows', None)
#pd.set_option('display.max_columns', None)
student_data = pd.DataFrame({
    'school_code': ['s001','s002','s003','s001', 's002','s004'],
    'class': ['V','V','VI','VI','V','VI'],
    'name':['Alberto Franco', 'Gino Mcneill', 'Ryan Parkes', 'Eesha Hinton', 'Gino Mcneill', 'David Parkes'],
    'date_of_birth': ['15/05/2002','17/05/2002', '16/02/1999','25/09/1998','11/05/2002','15/09/1997'],
    'age':[12,12,13,13,14,12],
    'height':[173,192,186,167,151,159],
    'weight':[35,32,33,30,31,32],
    'address': ['street1','street2','street3','street1','street2','street4']},
    #Had to include index in the dataframe
    index = ['S1','S2','S3','S4','S5','S6'])

#printing out the origininal dataframe
print("Original DataFrame:")
print(student_data)

#splitting it up based on school code and class
print('\nSplit the said data on school_code, class wise:')
result = student_data.groupby(['school_code','class'])
for name,group in result:
    #loop that will print out each group and their members
    print("\nGroup:")
    print(name)
    print(group)
```

Output for student data

```
Original DataFrame:
   school_code class            name  ... height  weight  address
S1        s001     V  Alberto Franco  ...    173      35  street1
S2        s002     V    Gino Mcneill  ...    192      32  street2
S3        s003    VI     Ryan Parkes  ...    186      33  street3
S4        s001    VI    Eesha Hinton  ...    167      30  street1
S5        s002     V    Gino Mcneill  ...    151      31  street2
S6        s004    VI    David Parkes  ...    159      32  street4

[6 rows x 8 columns]

Split the said data on school_code, class wise:

Group:
('s001', 'V')
   school_code class            name  ... height  weight  address
S1        s001     V  Alberto Franco  ...    173      35  street1

[1 rows x 8 columns]

Group:
('s001', 'VI')
   school_code class          name date_of_birth  age  height  weight  address
S4        s001    VI  Eesha Hinton    25/09/1998   13     167      30  street1

Group:
('s002', 'V')
   school_code class          name date_of_birth  age  height  weight  address
S2        s002     V  Gino Mcneill    17/05/2002   12     192      32  street2
S5        s002     V  Gino Mcneill    11/05/2002   14     151      31  street2

Group:
('s003', 'VI')
   school_code class         name date_of_birth  age  height  weight  address
S3        s003    VI  Ryan Parkes    16/02/1999   13     186      33  street3

Group:
('s004', 'VI')
   school_code class          name date_of_birth  age  height  weight  address
S6        s004    VI  David Parkes    15/09/1997   12     159      32  street4
```

Code for the csv file

```
#reading the csv file provided in the assignment
data = pd.read_csv("data.csv")
print("Statistical Description")
#describing the data
print(data.describe())

#checking for null values
print("\nChecking Null Values: ")
print(data.isnull().sum())

#filling in all null vallues with the mean
data.fillna(data.mean(), inplace=True)

#printing out the null values again to show that it was changed
print("\nChecking Null Values Again: ")
print(data.isnull().sum())

#choosing pulse and calories to aggregate
columns = ['Pulse', 'Calories']
aggregation = data[columns].agg(['min', 'max', 'count', 'mean'])
print("\nAggregated data for Pulse and Calories:")
print(aggregation)

#filtering all rows to get only rows that have between 500 and 1000 calories
cal_filter = data[(data['Calories'] >= 500) & (data['Calories'] <= 1000)]
print("\nRows where calories are between 500 and 1000:")
print(cal_filter)

#Another filter but this time it wants more than 500 calories and less than 100 pulse
cal_filter = data[(data['Calories'] > 500) & (data['Pulse'] < 100)]
print("\nRows where calories are above 500 and pulse is below 100:")
print(cal_filter)

#modified dataframe without Maxpulse
df_modified = data.drop(columns=['Maxpulse'])
print("\nModified DataFrame (without 'Maxpulse'):")
print(df_modified)

#removing maxpulse from the main data
data.drop(columns=['Maxpulse'], inplace=True)
print("\nDataFrame after deleting 'Maxpulse':")
print(data)

#Coverting calories to an int type
data['Calories'] = data['Calories'].astype(int)
print("\nData types after converting 'Calories' to int:")
print(data.dtypes)

#creating a scatter plot of duration vs calories
data.plot(kind='scatter', x='Duration', y='Calories', color='blue', title='Scatter plot of Duration vs Calories
plt.xlabel('Duration')
plt.ylabel('Calories')
plt.show()
```

Output for the csv file

```
Statistical Description
         Duration       Pulse    Maxpulse     Calories
count  169.000000  169.000000  169.000000   164.000000
mean    63.846154  107.461538  134.047337   375.790244
std     42.299949   14.510259   16.450434   266.379919
min     15.000000   80.000000  100.000000    50.300000
25%     45.000000  100.000000  124.000000   250.925000
50%     60.000000  105.000000  131.000000   318.600000
75%     60.000000  111.000000  141.000000   387.600000
max    300.000000  159.000000  184.000000  1860.400000

Checking Null Values:
Duration    0
Pulse       0
Maxpulse    0
Calories    5
dtype: int64

Checking Null Values Again:
Duration    0
Pulse       0
Maxpulse    0
Calories    0
dtype: int64

Aggregated data for Pulse and Calories:
            Pulse     Calories
min     80.000000    50.300000
max    159.000000  1860.400000
count  169.000000   169.000000
mean   107.461538   375.790244

Rows where calories are between 500 and 1000:
     Duration  Pulse  Maxpulse  Calories
51         80    123       146     643.1
62        160    109       135     853.0
65        180     90       130     800.4
66        150    105       135     873.4
67        150    107       130     816.0
72         90    100       127     700.0
73        150     97       127     953.2
75         90     98       125     563.2
78        120    100       130     500.4
83        120    100       130     500.0
90        180    101       127     600.1
99         90     93       124     604.1
101        90     90       110     500.0
102        90     90       100     500.0
103        90     90       100     500.4
106       180     90       120     800.3
108        90     90       120     500.3
```

More output for the csv file

```
Rows where calories are above 500 and pulse is below 100:
     Duration  Pulse  Maxpulse  Calories
65        180     90       130     800.4
70        150     97       129    1115.0
73        150     97       127     953.2
75         90     98       125     563.2
99         90     93       124     604.1
103        90     90       100     500.4
106       180     90       120     800.3
108        90     90       120     500.3

Modified DataFrame (without 'Maxpulse'):
```

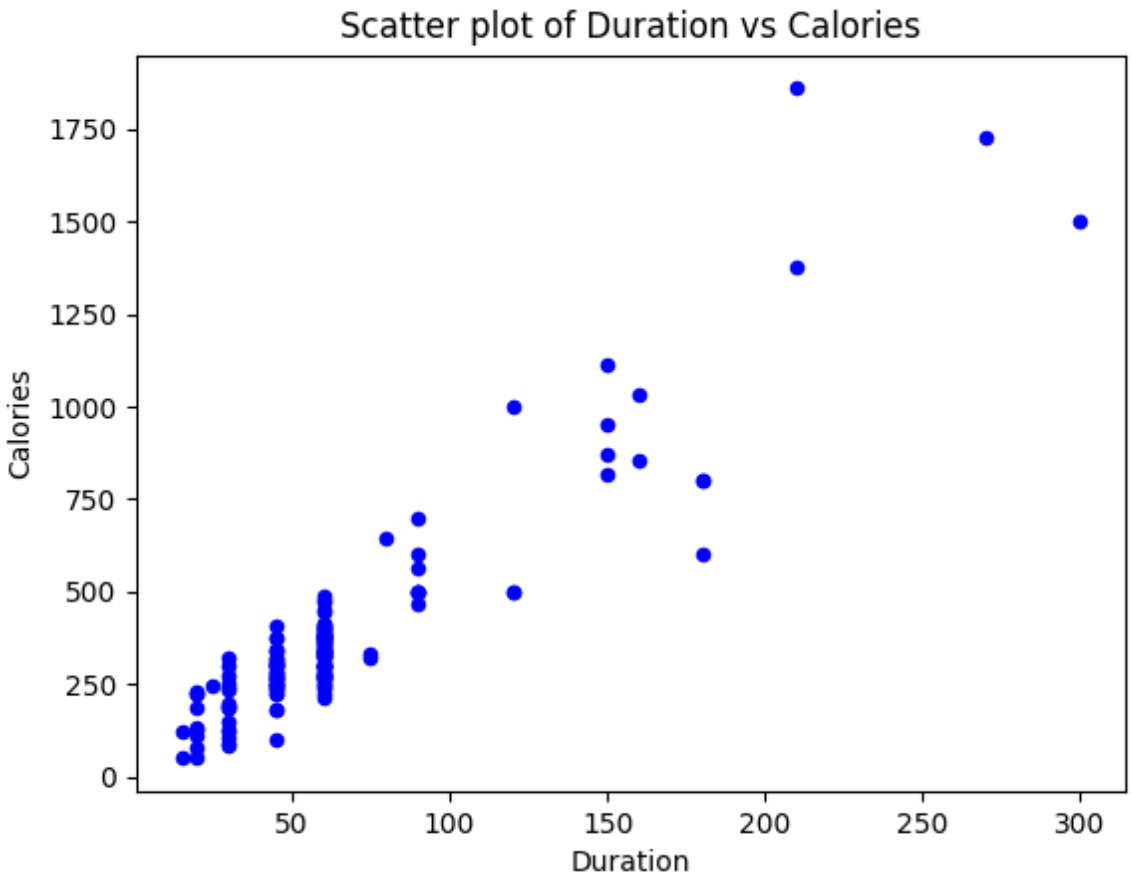
Squeezed text (170 lines).

```
DataFrame after deleting 'Maxpulse':
```

Squeezed text (170 lines).

```
Data types after converting 'Calories' to int:
Duration     int64
Pulse        int64
Calories     int64
dtype: object
```

Scatter Plot for the csv file


Scatter plot of Duration vs Calories
Calories
Duration
0
250
500
750
1000
1250
1500
1750
50
100
150
200
250
300