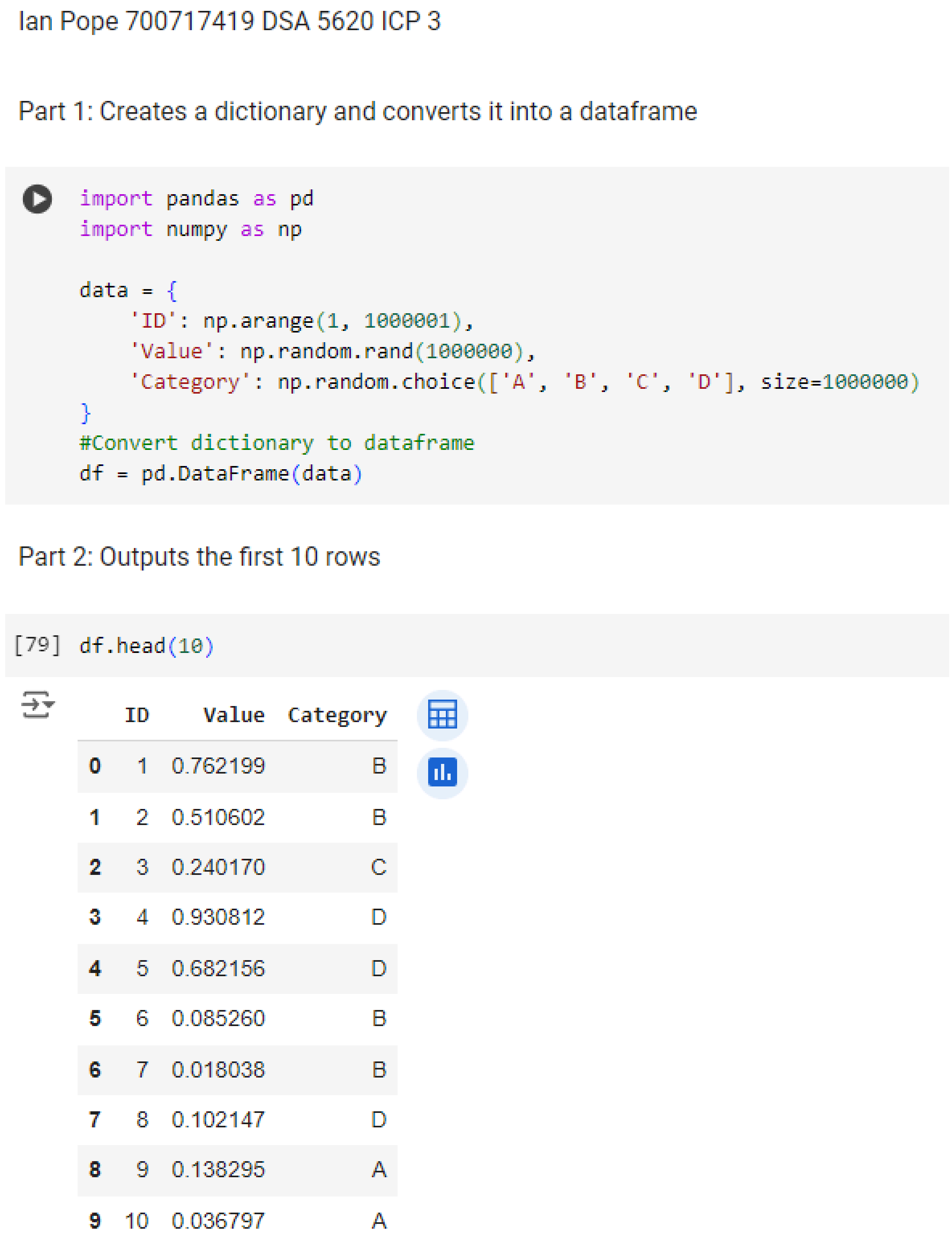
Ian Pope 700717419
DSA 5620 ICP 3



Ian Pope 700717419 DSA 5620 ICP 3

Part 1: Creates a dictionary and converts it into a dataframe

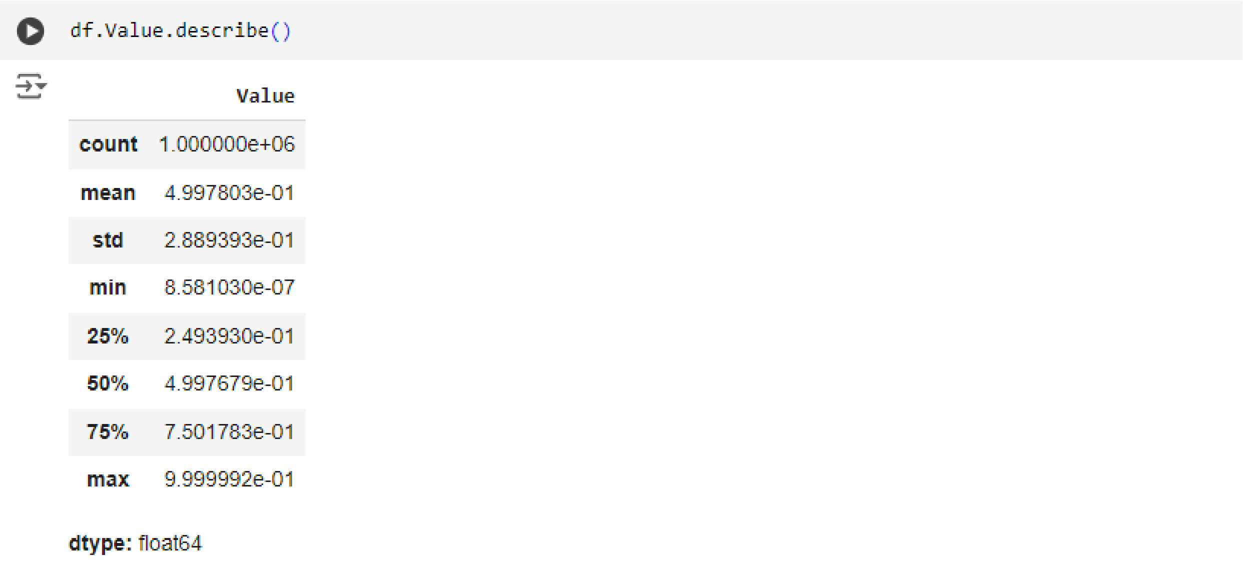
```
import pandas as pd
import numpy as np

data = {
    'ID': np.arange(1, 1000001),
    'Value': np.random.rand(1000000),
    'Category': np.random.choice(['A', 'B', 'C', 'D'], size=1000000)
}
#Convert dictionary to dataframe
df = pd.DataFrame(data)
```

Part 2: Outputs the first 10 rows

```
[79] df.head(10)
```

	ID	Value	Category
0	1	0.762199	B
1	2	0.510602	B
2	3	0.240170	C
3	4	0.930812	D
4	5	0.682156	D
5	6	0.085260	B
6	7	0.018038	B
7	8	0.102147	D
8	9	0.138295	A
9	10	0.036797	A

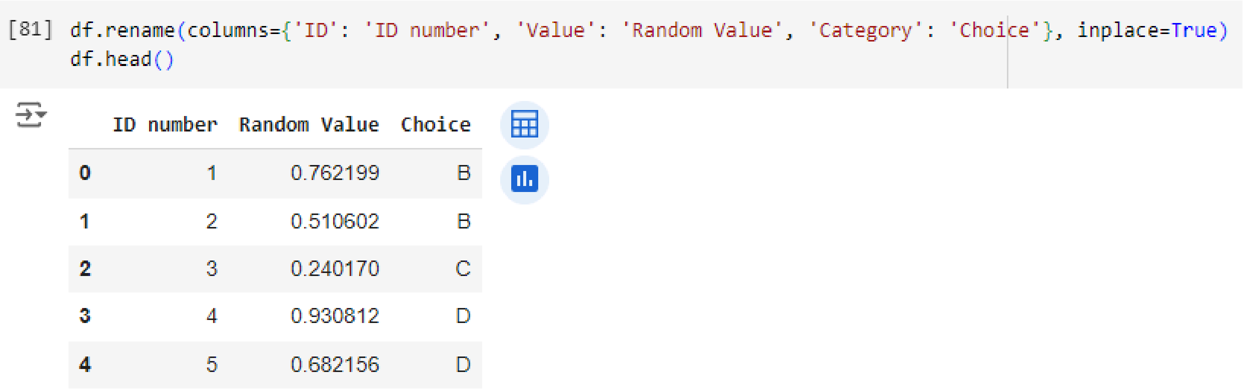
Part 3: Accesses a column 'Value' and describes it



```
df.Value.describe()
```

```
                Value
count   1.000000e+06
 mean   4.997803e-01
  std   2.889393e-01
  min   8.581030e-07
  25%   2.493930e-01
  50%   4.997679e-01
  75%   7.501783e-01
  max   9.999992e-01

dtype: float64
```

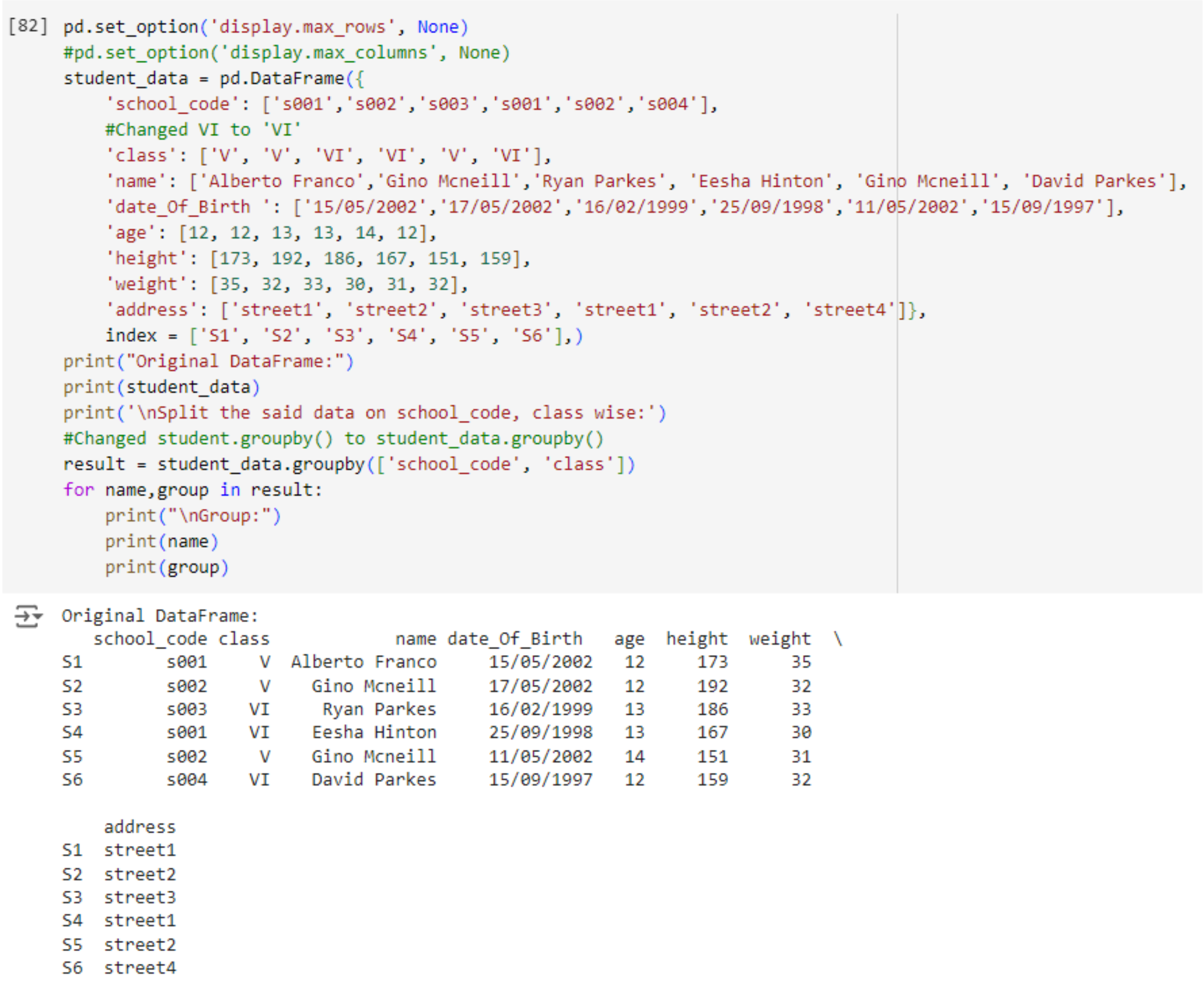
Part 4: Renames columns and outputs first five rows



```
[81] df.rename(columns={'ID': 'ID number', 'Value': 'Random Value', 'Category': 'Choice'}, inplace=True)
     df.head()
```

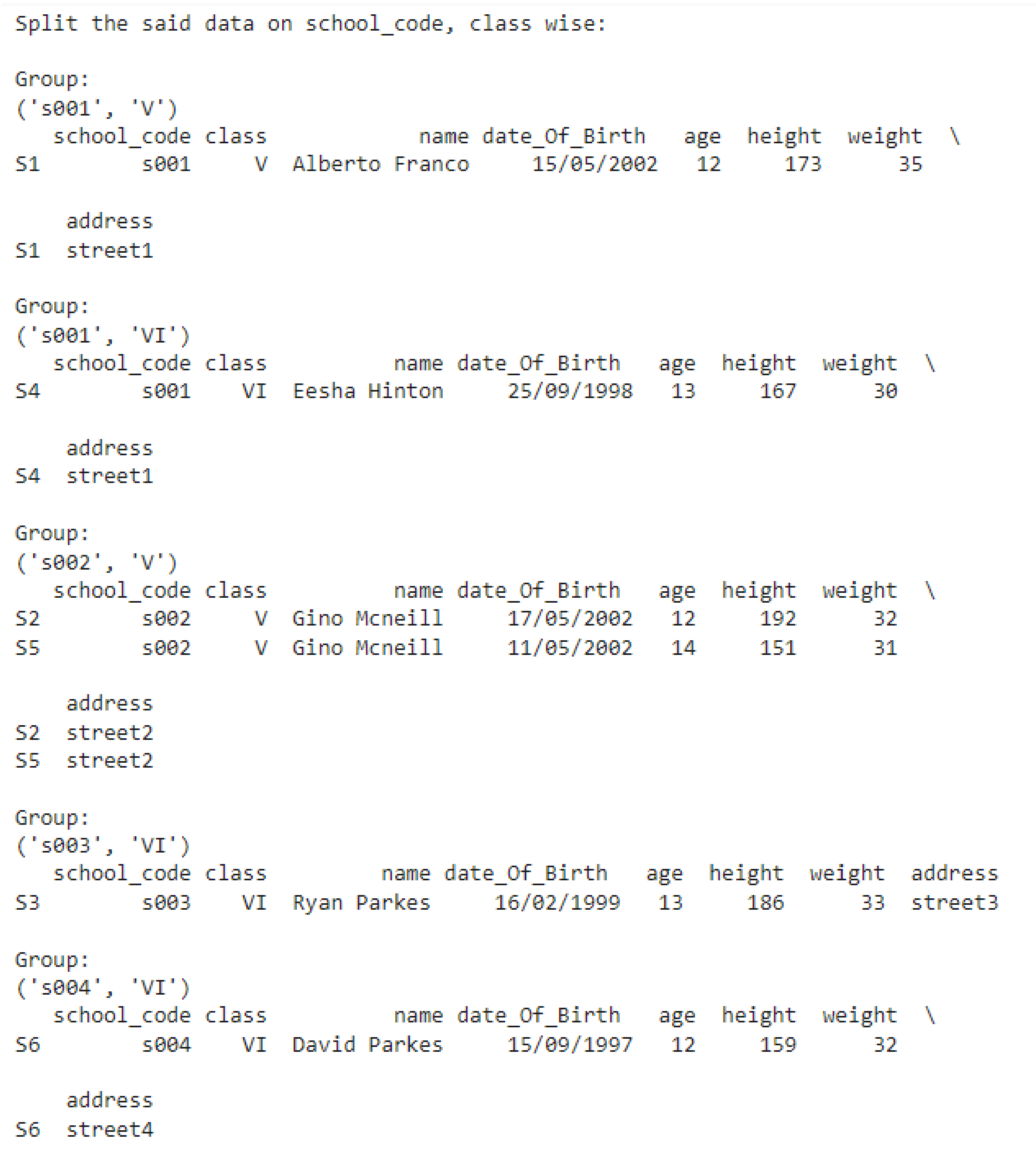
```
   ID number  Random Value  Choice
0          1      0.762199       B
1          2      0.510602       B
2          3      0.240170       C
3          4      0.930812       D
4          5      0.682156       D
```

Part 5: Fixes bugs in the given code to allow it to run

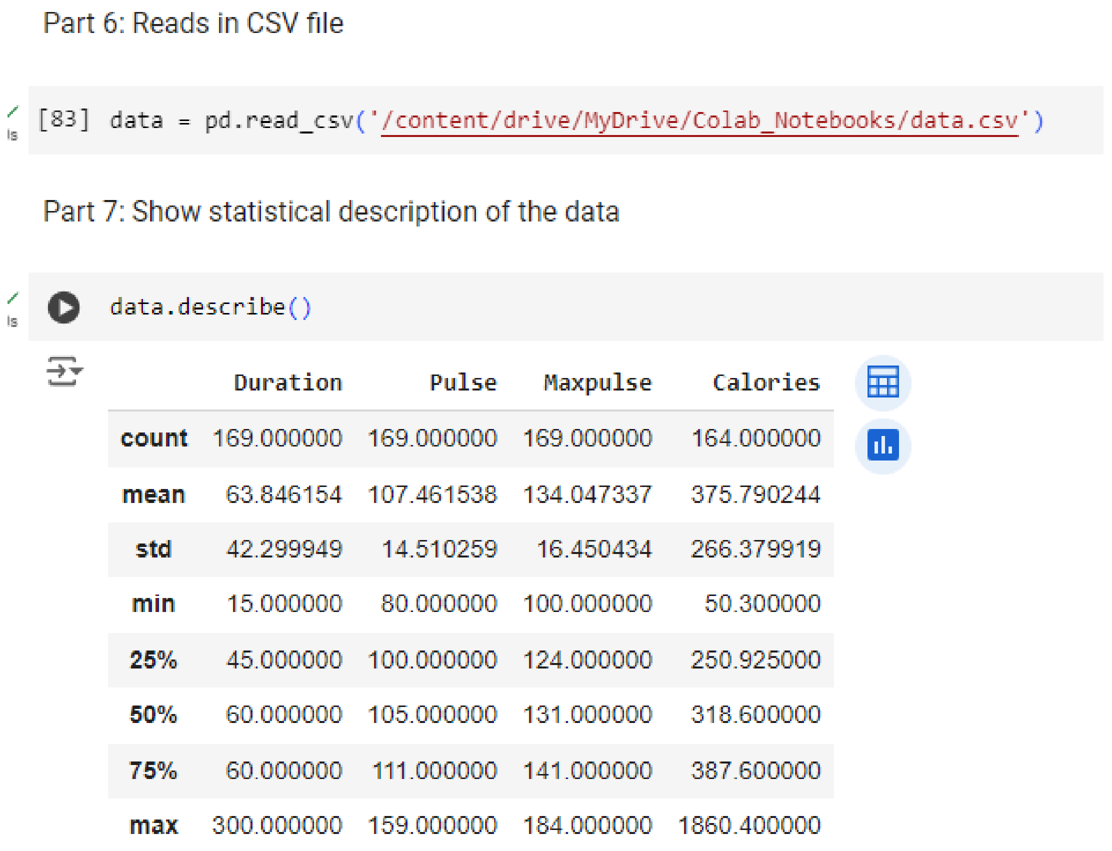


```
[82] pd.set_option('display.max_rows', None)
     #pd.set_option('display.max_columns', None)
     student_data = pd.DataFrame({
         'school_code': ['s001','s002','s003','s001','s002','s004'],
         #Changed VI to 'VI'
         'class': ['V', 'V', 'VI', 'VI', 'V', 'VI'],
         'name': ['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']},
         index = ['S1', 'S2', 'S3', 'S4', 'S5', 'S6'],)
     print("Original DataFrame:")
     print(student_data)
     print('\nSplit the said data on school_code, class wise:')
     #Changed student.groupby() to student_data.groupby()
     result = student_data.groupby(['school_code', 'class'])
     for name,group in result:
         print("\nGroup:")
         print(name)
         print(group)
```

```
Original DataFrame:
   school_code class            name date_Of_Birth   age  height  weight  \
S1        s001     V  Alberto Franco     15/05/2002   12     173      35
S2        s002     V    Gino Mcneill     17/05/2002   12     192      32
S3        s003    VI     Ryan Parkes     16/02/1999   13     186      33
S4        s001    VI    Eesha Hinton     25/09/1998   13     167      30
S5        s002     V    Gino Mcneill     11/05/2002   14     151      31
S6        s004    VI    David Parkes     15/09/1997   12     159      32

    address
S1  street1
S2  street2
S3  street3
S4  street1
S5  street2
S6  street4
```



```
Split the said data on school_code, class wise:

Group:
('s001', 'V')
   school_code class            name date_Of_Birth   age  height  weight  \
S1        s001     V  Alberto Franco     15/05/2002   12     173      35

    address
S1  street1

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

    address
S4  street1

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

    address
S2  street2
S5  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  \
S6        s004    VI  David Parkes     15/09/1997   12     159      32

    address
S6  street4
```



Part 6: Reads in CSV file

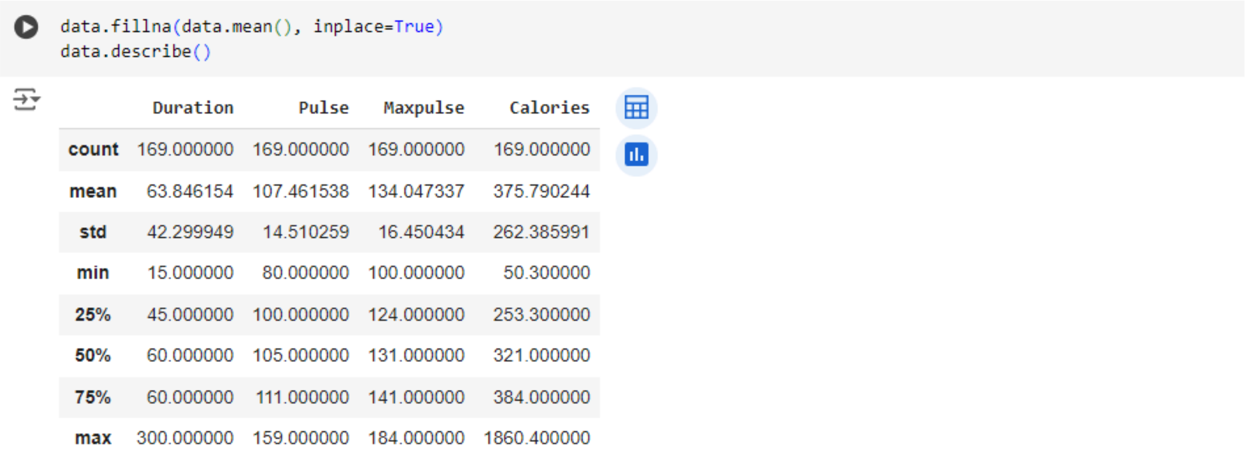
```
[83] data = pd.read_csv('/content/drive/MyDrive/Colab_Notebooks/data.csv')
```

Part 7: Show statistical description of the data

```
data.describe()
```

	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

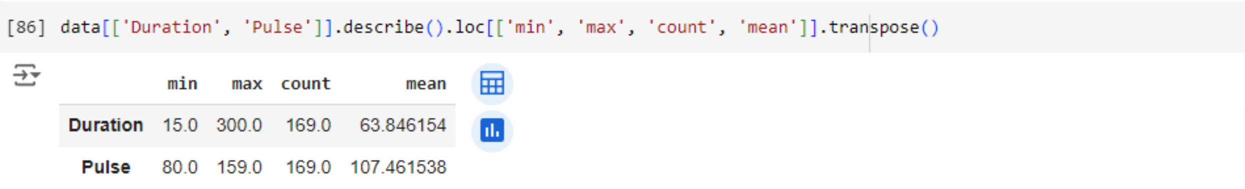
Part 8: Check data for null values and replace with mean. We can tell it was modified because the 50% marking for Calories changed from what was printed above.



```
data.fillna(data.mean(), inplace=True)
data.describe()
```

	Duration	Pulse	Maxpulse	Calories
count	169.000000	169.000000	169.000000	169.000000
mean	63.846154	107.461538	134.047337	375.790244
std	42.299949	14.510259	16.450434	262.385991
min	15.000000	80.000000	100.000000	50.300000
25%	45.000000	100.000000	124.000000	253.300000
50%	60.000000	105.000000	131.000000	321.000000
75%	60.000000	111.000000	141.000000	384.000000
max	300.000000	159.000000	184.000000	1860.400000

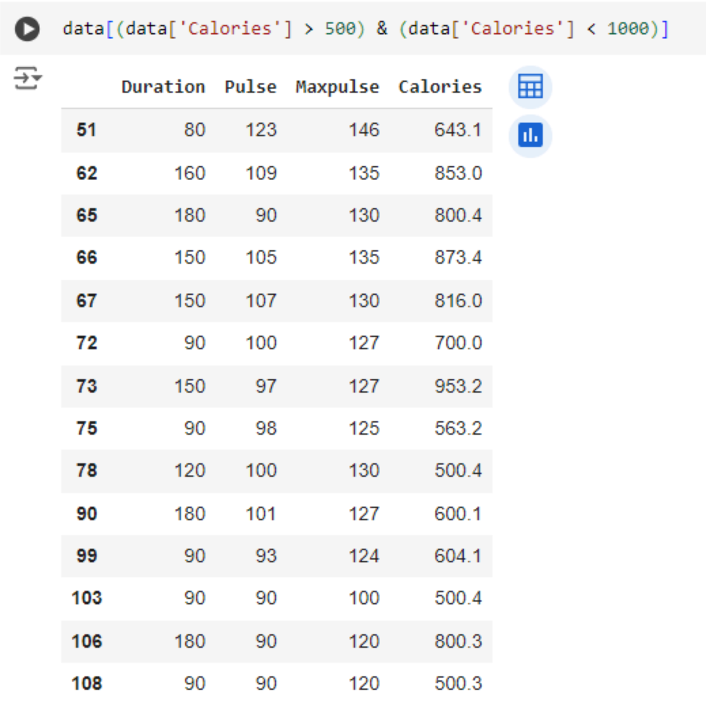
Part 9: Get the min, max, count, and mean of two columns



```
[86] data[['Duration', 'Pulse']].describe().loc[['min', 'max', 'count', 'mean']].transpose()
```

	min	max	count	mean
Duration	15.0	300.0	169.0	63.846154
Pulse	80.0	159.0	169.0	107.461538

Part 10: Filter data to select rows with calories between 500 and 1000



```
data[(data['Calories'] > 500) & (data['Calories'] < 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
90	180	101	127	600.1
99	90	93	124	604.1
103	90	90	100	500.4
106	180	90	120	800.3
108	90	90	120	500.3

Part 11: Filter dataframe to get rows with calories > 500 and a pulse < 100

```
data[(data['Calories'] > 500) & (data['Pulse'] < 100)]
```

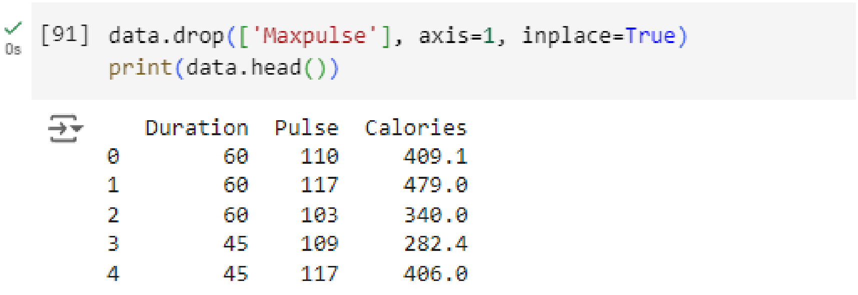
	Duration	Pulse	Maxpulse	Calories
65	180	90	130	800.4
70	150	97	129	1115.0
73	150	97	127	953.2
75	90	98	125	563.2
99	90	93	124	604.1
103	90	90	100	500.4
106	180	90	120	800.3
108	90	90	120	500.3

Part 12: Create a new dataframe without the maxpulse column. We can see that the original dataframe remains unaffected.

```
[90] df_modified = data.drop(['Maxpulse'], axis=1)
     print(df_modified.head())
     print(data.head())
```

```
   Duration  Pulse  Calories
0        60    110     409.1
1        60    117     479.0
2        60    103     340.0
3        45    109     282.4
4        45    117     406.0
   Duration  Pulse  Maxpulse  Calories
0        60    110       130     409.1
1        60    117       145     479.0
2        60    103       135     340.0
3        45    109       175     282.4
4        45    117       148     406.0
```

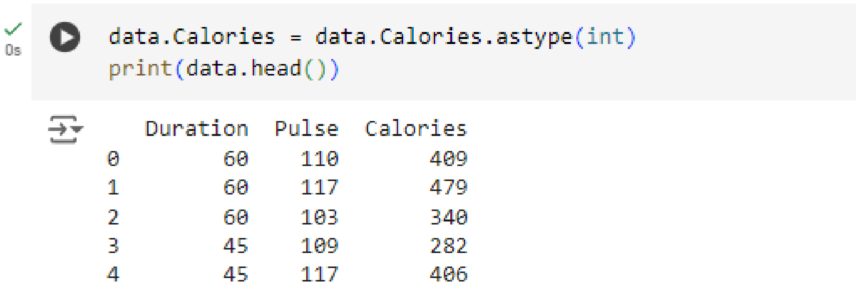
Part 13: Remove the maxpulse column from the original dataframe



```
data.drop(['Maxpulse'], axis=1, inplace=True)
print(data.head())
```

```
   Duration  Pulse  Calories
0        60    110     409.1
1        60    117     479.0
2        60    103     340.0
3        45    109     282.4
4        45    117     406.0
```

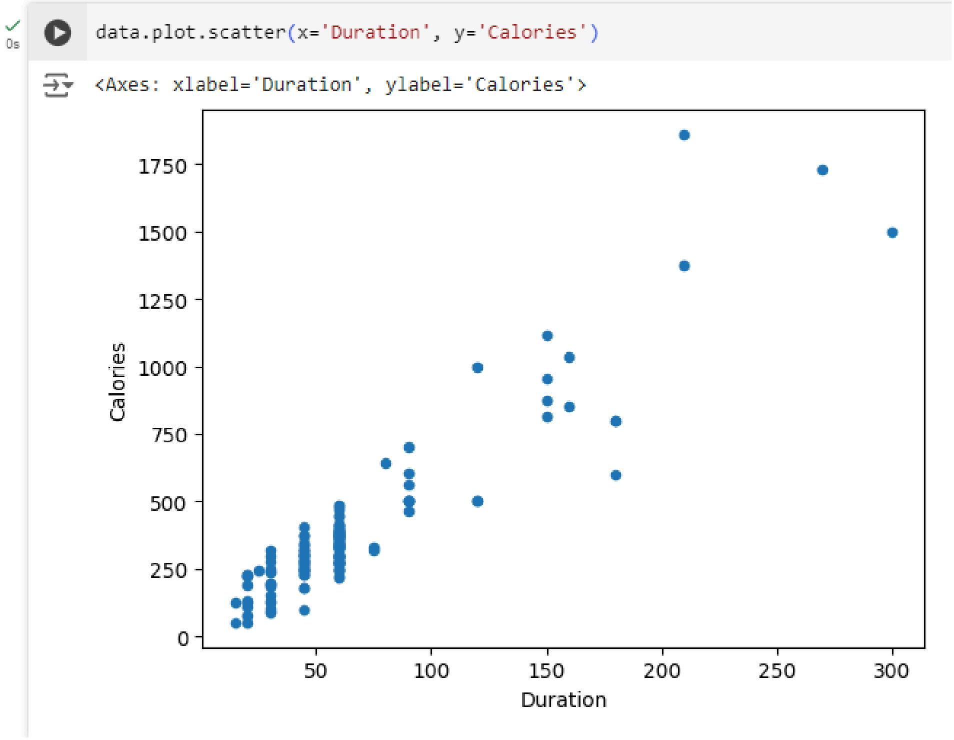
Part 14: Convert calories from a float to an int



```
data.Calories = data.Calories.astype(int)
print(data.head())
```

```
   Duration  Pulse  Calories
0        60    110       409
1        60    117       479
2        60    103       340
3        45    109       282
4        45    117       406
```

Part 15: Create a scatter plot for duration and calories



```
data.plot.scatter(x='Duration', y='Calories')
```

```
<Axes: xlabel='Duration', ylabel='Calories'>
```