Topic	Mean,Median, Mode	
Class Description	Student learns to about the descriptive statistics .mean , median and mode Student also learns the usage of some python inbuilt function such as Counter, items() and values() Student builds a small python program/tool to find mean , median and mode	
Class	C104	
Class time	45 mins	
Goal	● Learn about descriptive statistics. ● Learn about finding mean, median and mode from given data. ● Build python programs to get mean , median and mode from given csv file	

Resources Required	● Teacher Resources
	  ○ Visual Code studio
	  ○ Laptop with internet connectivity
	  ○ Earphones with mic
	  ○ Notebook and pen
	
	● Student Resources
	  ○ Visual Code studio
	  ○ Laptop with internet connectivity
	  ○ Earphones with mic
	  ○ Notebook and pen	

Class structure	**Warm Up** **Teacher-led Activity** **Student-led Activity** **Wrap up**	**5 mins** **15 min** **15 min** **5 min**

CONTEXT
● **Talk about using Descriptive data**

Class Steps	Teacher Action	Student Action

Step 1: Warm Up (5 mins)	Hi Do you remember what we learned in the last class?	ESR: We learned about using ploty express . We plotted line graph, scatter graph and bar graph using plotly express.
	Great! so we saw how data can be shown using graphs. Let's say when you get your report card you get a mark and percentage. So what does that percentage and marks denotes?	ESR: Marks denote how good we did in the subject and where we need improvement. And percentage denotes how good we did overall.
	So this kind of data is called descriptive data which gives us information . Sometimes we also find a central tendency. Central tendency is a value that tries to describe a data by identifying the central position within the given data. This is also classed as summary statistics. An example of central tendency is finding an average.	-
	Let's learn more about central tendencies in our class. Are you ready?	ESR: Yes.

<table>
<tr><td colspan="3" align="center">Teacher Initiates Screen Share</td></tr>
<tr><td colspan="3" align="center"><u>CHALLENGE</u>
● Get csv data and convert it to list
● Use Counter , items(), values() to find mean ,median and mode.</td></tr>
</table>

Step 2: Teacher-led Activity	What meaning comes to your mind when you hear the word "statistics"?	ESR: Statistics means a collection of data.

(15 min)	And what comes to your mind when you hear the word "descriptive statistics"?	Descriptive statistics means data which describes something or gives information about something.
	Yes! So descriptive statistics is a summary statistics that gives information or tells about the features of the data. Mean ,Median and Mode are a part of descriptive statistics.	-
	We also know Mean by another name that is "Average" . Can you tell me how we calculate the average?	ESR: to calculate average we divide the sum of values by the number of values.
	Perfect!! Same way we calculate mean. For example we have a set of numbers 23,25,26,30,35,23,46,55 to find mean:- mean = sum of values / no of values so 23+25+26+30+35+23+46+55 /8 263/8 = 32.875 So our mean is 32.875	Student asks questions about finding mean value

```
>>> a =23+25+26+30+35+23+46+55
>>> print(a)
263
>>> a/8
32.875
>>>
```

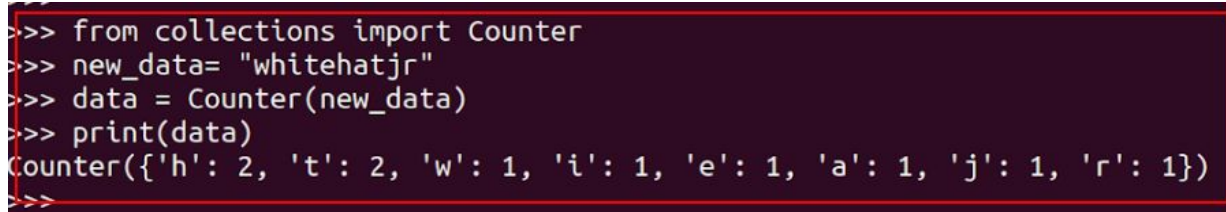|	Now let's see what is median. <Teacher googles the technical meaning of median and reads it for the student> So as the meaning says it means situated in the middle or the middle number. To find Median we arrange all the numbers from small to greater and then pick the number in the middle or if there are two middle numbers then we take mean to those two numbers. For example we have our set of numbers 25,30,35,23,46,55,23,26. First we have to sort it from lower to higher. 23,23,25,26,30,35,46,55. Now we find two middle numbers which are 26 and 30. let's find the mean of it which is (26+30) /2 = 28. So the median is 28.	Student asks questions about finding median value

```
>>>
>>> 26+30
56
>>> 56/2
28.0
>>>
```

	The final is mode . Mode is used to find the most occuring value of a data set. To find mode we just have to find the most occuring element of the data. For example 23,25,26,30,35,23,46,55 in this data set which is the most occuring number? Yes 23. so that's our mode.	ESR: 23. <Student asks questions about mode>
	Most of our data is going to be in the csv files. So we need to read it , get it in a list and from there and use it to find mean median and mode. Can you tell me how we open a file and read it ?	ESR: We use with open("filename", "r") as f: to read the file where r stands for read method on the file.
	yes! Here it's a little different let's see how. first we import csv The CSV module is one of the modules in Python which provides classes for reading and writing tabular information in CSV file format. Then we open our csv file **with open('Internet Users.csv', newline='') as f:** if we have csv as a text file object then we open it with newline to avoid pythons own line handling module . CSV provides us a csv.reader method which reads and returns the current row and then moves on to the next. **file_data = list(reader)** adds the data to the list.	Student asks questions about using with and newline and csv.reader method

| | full code:-
import csv
with open('height-weight.csv',
newline='') as f:
    reader = csv.reader(f)
    file_data = list(reader) | |

```
import csv
with open('height-weight.csv', newline='') as f:
    reader = csv.reader(f)
    file_data = list(reader)
```

| | Python also has a collection module which has a counter method which is basically a container that keeps track of how many times the same values are added/repeated.
To use this method you simply have to import it and pass the file data to the counter method.
\<teacher runs the following code on python shell and shows output\>
code:-
from collections import Counter
new_data= "whitehatjr"
data = Counter(new_data)
print(data) | Student observes and asks questions about counter |

```
>>> from collections import Counter
>>> new_data= "whitehatjr"
>>> data = Counter(new_data)
>>> print(data)
Counter({'h': 2, 't': 2, 'w': 1, 'i': 1, 'e': 1, 'a': 1, 'j': 1, 'r': 1})
>>>
```

	Python has a **dictionary.items()** method which used to return the list with all dictionary keys with values. It returns a tuple of key value pairs. code: **dictionary.items()** Teacher runs the following code and shows output code:- **new_list = data.items()** **print(new_list)**	Student asks questions about dictionary items

```
>>>
>>> new_list = data.items()
>>> print(new_list)
dict_items([('w', 1), ('h', 2), ('i', 1), ('t', 2), ('e', 1), ('a', 1), ('j', 1)
, ('r', 1)])
>>>
```

	dictionary.values() returns the list of all the values in the dictionary.	-

```
>>>
>>> value = data.values()
>>> print(value)
dict_values([1, 2, 1, 2, 1, 1, 1, 1])
>>>
```

	Now that you know how to find mean , median and mode , I have a challenge for you to write code to find mean , median and mode. Can you do that?	ESR: YES!! <Student takes up the challenge>

<div align="center" style="color:red;font-weight:bold">Teacher Stops Screen Share</div>

	Now it's your turn. Please share your screen with me.	

- **Ask Student to press ESC key to come back to panel**
- **Guide Student to start Screen Share**

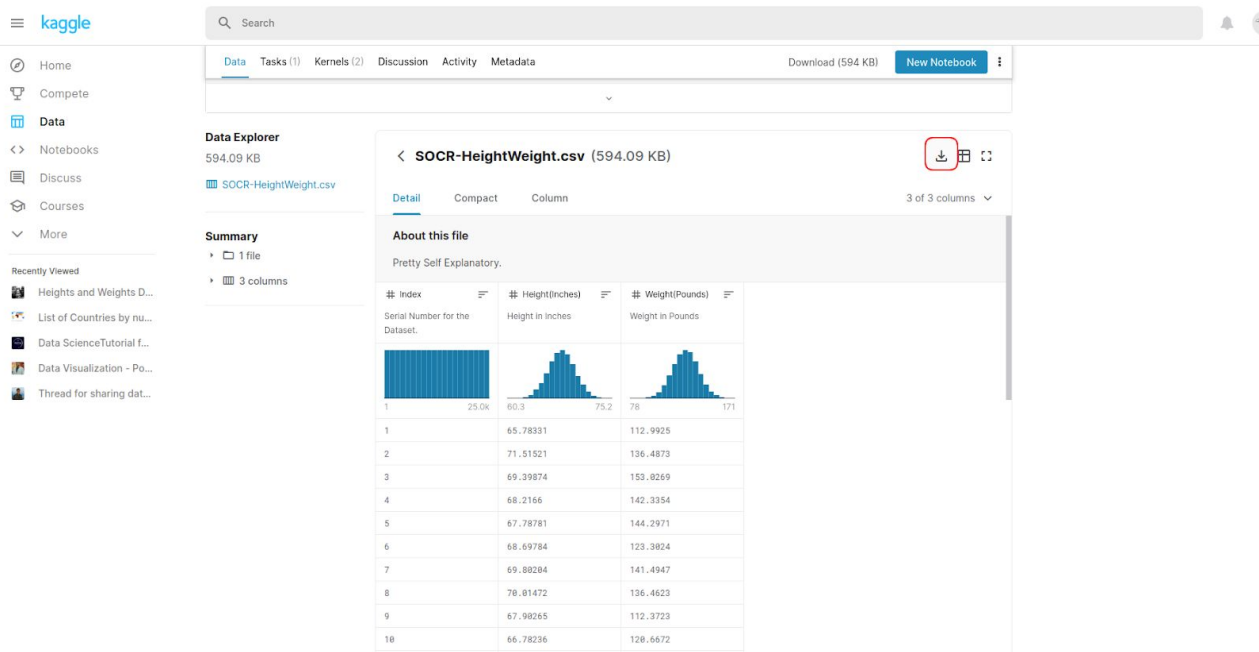| ● **Teacher gets into Fullscreen** |
|---|

ACTIVITY
● **Build a python program to find mean, median and mode from given csv file data.**

Step 3: Student-Led Activity (15 min)	Teacher helps students to create a file and download the csv file.	Student opens the editor and creates a mean.py file and downloads the csv file from **student activity 1**

	Teacher helps student to write code for reading the csv and storing data in a list	<Student writes code to read csv file and store data in a list.>

```
# list of elements to calculate mean
import csv
with open('height-weight.csv', newline='') as f:
    reader = csv.reader(f)
    file_data = list(reader)
```

	Now we need to sort data from **file_data** to get the list of height from it.	<student codes to sort data from the given data removes the first list using pop() .
	First we remove the title list from the data using **pop()** .	The create the empty list named new_data.
	Then create a empty list named new_data .	uses for loop on file_data to get the element inside the nested lists and append them to the new_data list.
	Then use a for loop on file_data to get the elements inside the nested lists and append them to the **new_data** list.	
	<Teacher helps student with the code>	

```
4    # list of elements to calculate mean
5    import csv
6    with open('height-weight.csv', newline='') as f:
7        reader = csv.reader(f)
8        file_data = list(reader)
9
0    file_data.pop(0)
1    # print(file_data)
2    # sorting data  to get the height of people.
3    new_data=[]
4    for i in range(len(file_data)):
5        n_num = file_data[i][1]
6        new_data.append(float(n_num))
7
8
```

	How do we find Mean? Prefect. <Teacher helps student find mean>	ESR: We get the sum of the values and then divide by the number of values' Student codes to find the mean and then print it. Then runs the code and tests it.

```
4    # list of elements to calculate mean
5    import csv
6    with open('height-weight.csv', newline='') as f:
7        reader = csv.reader(f)
8        file_data = list(reader)
9
0    file_data.pop(0)
1    # print(file_data)
2    # sorting data  to get the height of people.
3    new_data=[]
4    for i in range(len(file_data)):
5        n_num = file_data[i][1]
6        new_data.append(float(n_num))
7
8
9    # #getting the mean
0    n = len(new_data)
1    total =0
2    for x in new_data:
3        total += x
4
5    mean = total / n
6    #
7    print("Mean / Average is: " + str(mean))
8
```

```
$ python3 mean.py
Mean / Average is: 67.99311359679979
```

	Now we'll code to find the **Median**. First we need to read the csv and sort the data to get a list of heights. <Teacher helps student create a new file and code to read the csv file and get data specifically of heights >	student create a new file and code to read the csv file and get data specifically of heights>

```python
# Median of elements
import csv

with open('height-weight.csv', newline='') as f:
    reader = csv.reader(f)
    file_data = list(reader)
# removing the list of titles using pop
file_data.pop(0)

new_data=[]
for i in range(len(file_data)):
    n_num = file_data[i][1]
    new_data.append(n_num)
```

	Now we need this data in ascending order of numbers. To do that we'll use a sort() function , this function sorts the data in ascending order for us. And then get the length of the data and store it in n variable <teacher helps student to use sort function to get data in ascending order and get the length of data and store it in n variable.>	Student codes to use sort() function to sort the data and store it's length in a variable.

```
3    import csv
4
5    with open('height-weight.csv', newline='') as f:
6        reader = csv.reader(f)
7        file_data = list(reader)
8    # removing the list of titles using pop
9    file_data.pop(0)
10
11   new_data=[]
12   for i in range(len(file_data)):
13       n_num = file_data[i][1]
14       new_data.append(n_num)
15
16   n = len(new_data)
17   new_data.sort()
18
19
```

	The length we took before is to check if the number of values in the data are even or not. Here we use the **floor** division to get the whole number . we also convert the string values to float using the float **function**. If the length is an even number then there will be 2 values as medians and we'll have to find the mean of those two values. If the length is an odd number then we don't need to find the mean . As the number we find will be the median. <Teacher helps student to write a if condition to check if the length of the data is even . If the length is even then we find the mean of the two numbers and the result is the median.	<student writes a if condition to check if the length of the data is even . If the length is even then we find the mean of the two numbers and the result is the median. If the length is a odd number then get the middle number is median>

	If the length is a odd number then get the middle number is median>	

```
11    new_data=[]
12    for i in range(len(file_data)):
13        n_num = file_data[i][1]
14        new_data.append(n_num)
15
16    n = len(new_data)
17    new_data.sort()
18
19
20    #using floor division to get the nearest number whole number
21    # floor division is shown by //
22    if n % 2 == 0:
23        #getting the first number
24        median1 = float(new_data[n//2])
25        #getting the second number
26        median2 = float(new_data[n//2 - 1])
27        #getting mean of those numbers
28        median = (median1 + median2)/2
29    else:
30        median = new_data[n//2]
31        print(n)
32    print("Median is: " + str(median))
33
```

	Now lets run and test the code.	Student runs and tests the code.

```
$ python3 median.py
Median is: 67.9957
ashura@zeros:~/Documents/c104$
```

	We wrote code for mean and median now all that is left is mode. As we know mode is used to find the most frequent occurrence of a value. What will be our first step?	ESR: To get the data from csv and store it in a list. <Student codes to read csv file and store the data in a list

```
 3   from collections import Counter
 4   import csv
 5
 6   with open('height-weight.csv', newline='') as f:
 7       reader = csv.reader(f)
 8       file_data = list(reader)
 9
10   file_data.pop(0)
11
12   new_data=[]
13   for i in range(len(file_data)):
14       n_num = file_data[i][1]
15       new_data.append(n_num)
16
```

	Then from collection import Counter and pass the **new_data** to Counter method and store it in a variable called data. Then we create a dictionary with range of height as keys and the occurrences of the heights as values which at start is 0.	<Student codes to import Counter form collection, pass **new_data** to the Counter method and store it in a variable called data. Then create a dictionary with range of height as keys and the occurrences of the heights .>

```
18
19     #Calculating Mode
20     data = Counter(new_data)
21     mode_data_for_range = {
22                            "50-60": 0,
23                            "60-70": 0,
24                            "70-80": 0
25                          }
```

| | Using for loop on data.item(), using if and elif condition we'll check if the height lies between 50-60 ,60-70.70-80. We also need to convert the string to float using float If the height lies in the range the occurrence count will increase.

<Teacher helps student with the code>
code:-
for height, occurrence in data.items():
    if 50 < float(height) < 60:
       mode_data_for_range["50-60"] += occurrence
    elif 60 < float(height) < 70:
       mode_data_for_range["60-70"] += occurrence
    elif 70 < float(height) < 80:
       mode_data_for_range["70-80"] += occurrence | <student codes to create 3 empty lists mode,mode1 and mode2. Using for loop on get_mode.item(), using if and elif condition we'll check if the height lies between 50-60 ,60-70.70-80. If the height lies in the range the occurrence count will increase> |

```
#Calculating Mode
data = Counter(new_data)
mode_data_for_range = {
                        "50-60": 0,
                        "60-70": 0,
                        "70-80": 0
                      }
for height, occurrence in data.items():
    if 50 < float(height) < 60:
        mode_data_for_range["50-60"] += occurrence
    elif 60 < float(height) < 70:
        mode_data_for_range["60-70"] += occurrence
    elif 70 < float(height) < 80:
        mode_data_for_range["70-80"] += occurrence
```

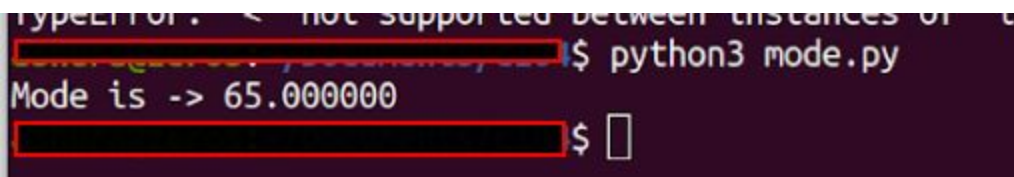|		
	Then declare **mode_range and mode_occurrence** and set it's value to 0. Using for loop using on **mode_data_for_range.items().** If occurrence is greater than mode_occurrence then using split() on range we get the first and second number mode. Then find the mean of the two modes and print it <Teacher helps student with the code>	<student codes to declare **mode_range and mode_occurrence** and set it's value to 0. Using for loop using on **mode_data_for_range.ite ms().** If occurrence is greater than mode_occurrence then using split() on range we get the first and second number mode. Then find the mean of the two modes and print it>

```
for height, occurrence in data.items():
    if 50 < float(height) < 60:
        mode_data_for_range["50-60"] += occurrence
    elif 60 < float(height) < 70:
        mode_data_for_range["60-70"] += occurrence
    elif 70 < float(height) < 80:
        mode_data_for_range["70-80"] += occurrence

mode_range, mode_occurrence = 0, 0
for range, occurrence in mode_data_for_range.items():
    if occurrence > mode_occurrence:
        mode_range, mode_occurrence = [int(range.split("-")[0]), int(range.split("-")[1])], occ
mode = float((mode_range[0] + mode_range[1]) / 2)
print(f"Mode is -> {mode:2f}")
```

	Let's test the code	<Student runs and tests the code>

```
TypeError: < not supported between instances of ...
$ python3 mode.py
Mode is -> 65.000000
$
```

Teacher Guides Student to Stop Screen Share

FEEDBACK
● **Appreciate the student for their class**
● **Get them to play around with different data to find mean , median and mode.**

| **Step 4:**
Wrap-Up
(5 min) | Let's quickly wrap up today's class. What did we learn? | ESR:
We learned about descriptive statistics.
-We learned to use inbuilt |

		functions like Counter, items(), values(). -We leaned to find central tendencies such as mean, mode and median
	Now that you know how to find mean,mode and median using code you can experiment with different data sets. In next class we'll learn about the correlation of the two data sets. Are you excited?	Yes!
	Looking forward to the next class then	
	Teacher Clicks ✖ End Class	

Additional Activities	Encourage the student to write reflection notes in their reflection journal using markdown.
	
	Use these as guiding questions:
	
	● What happened today?
	  - Describe what happened
	  - Code I wrote
	● How did I feel after the class?
	● What have I learned about programming and developing games?
		The student uses the markdown editor to write her/his reflection in a reflection journal.

	● What aspects of the class helped me? What did I find difficult?	

Activity	Activity Name	Links
Teacher Activity 1	Reference code	https://github.com/whitehatjr/MeanMedianMode
Student Activity 1	height weight data set	https://www.kaggle.com/burnoutminer/heights-and-weights-dataset