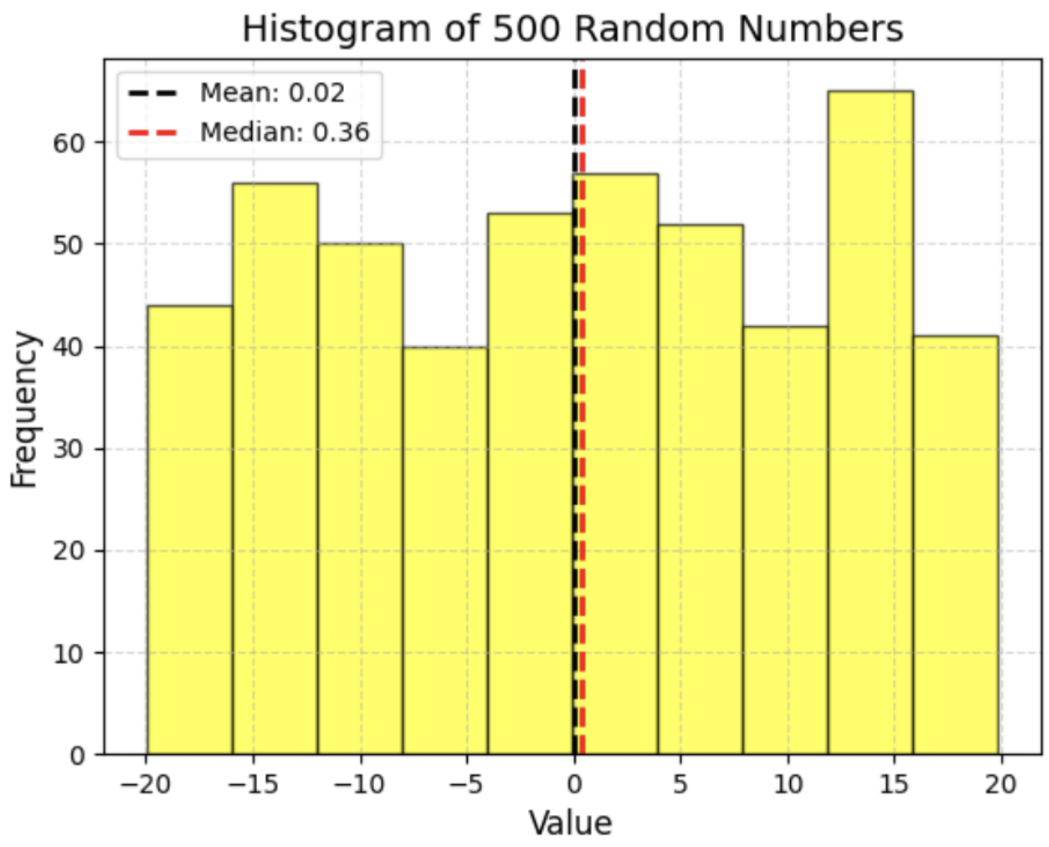
NAME - JASPREET SINGH
SUBMITTED TO - PROF. ALEX YANG
SUBJECT - PROBABILITY AND STATISTICS
2023 FALL HOMEWORK#1

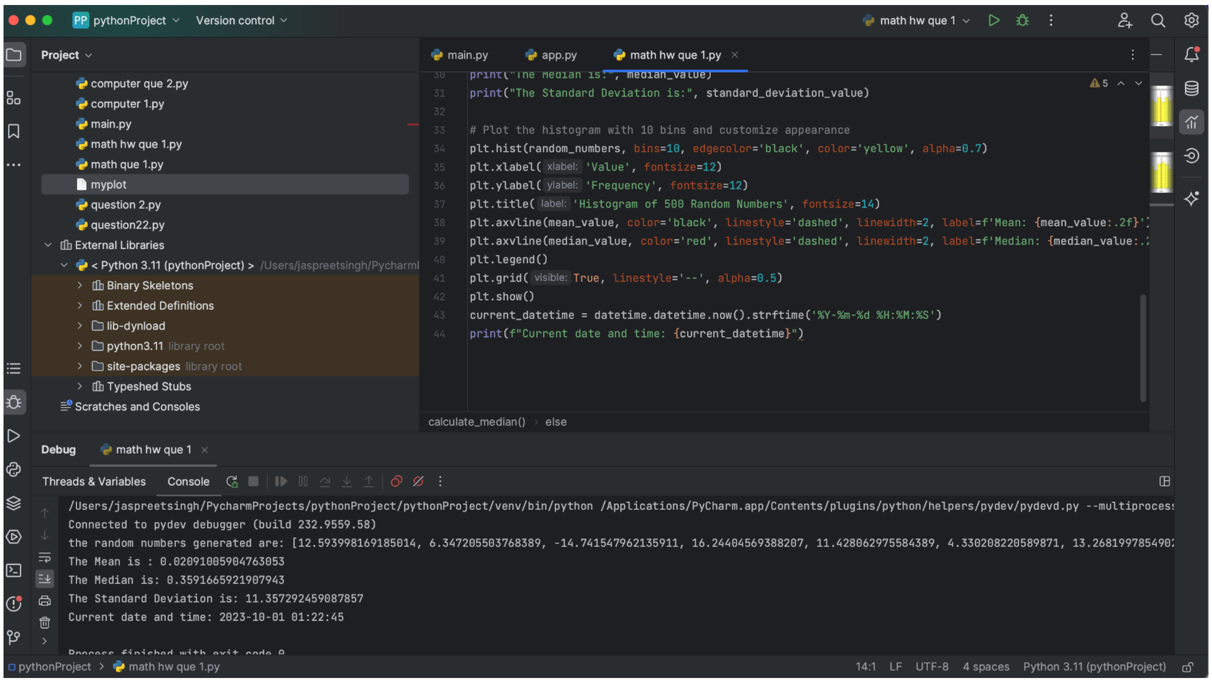
1. Write the program in any computer language, Python preferred to create 500 Random numbers from -20 to +20 in uniform distribution and find the mean, median And standard deviation. After that, plot the histogram with 10 bins. Notice that the only user defined function can be used to calculate the mean, median and standard deviation, don't directly call existing function from Python library.

CODE:

```python
import datetime
import random
import matplotlib.pyplot as plt

def calculate_mean(numbers):
   return sum(numbers) / len(numbers)

def calculate_median(numbers):
   sorted_numbers = sorted(numbers)
   n = len(sorted_numbers)
   if n % 2 == 0:
       return (sorted_numbers[n // 2 - 1] + sorted_numbers[n // 2]) / 2
   else:
       return sorted_numbers[n // 2]

def calculate_standard_deviation(numbers, mean):
   variance = sum((x - mean) ** 2 for x in numbers) / len(numbers)
   return variance ** 0.5

# Generate 500 random numbers in the range [-20, 20]
random_numbers = [random.uniform(-20, 20) for _ in range(500)]

# Calculate mean, median, and standard deviation
mean_value = calculate_mean(random_numbers)
median_value = calculate_median(random_numbers)
standard_deviation_value = calculate_standard_deviation(random_numbers, mean_value)

print("the random numbers generated are:", random_numbers)
print("The Mean is :", mean_value)
print("The Median is:", median_value)
print("The Standard Deviation is:", standard_deviation_value)

# Plot the histogram with 10 bins and customize appearance
plt.hist(random_numbers, bins=10, edgecolor='black', color='yellow', alpha=0.7)
plt.xlabel('Value', fontsize=12)
plt.ylabel('Frequency', fontsize=12)
plt.title('Histogram of 500 Random Numbers', fontsize=14)
plt.axvline(mean_value, color='black', linestyle='dashed', linewidth=2, label=f'Mean: {mea
plt.axvline(median_value, color='red', linestyle='dashed', linewidth=2, label=f'Median: {m
plt.legend()
plt.grid(True, linestyle='--', alpha=0.5)
plt.show()
current_datetime = datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S')
print(f"Current date and time: {current_datetime}")
```

OUTPUT:





```
print("The Median is:", median_value)
print("The Standard Deviation is:", standard_deviation_value)

# Plot the histogram with 10 bins and customize appearance
plt.hist(random_numbers, bins=10, edgecolor='black', color='yellow', alpha=0.7)
plt.xlabel( xlabel: 'Value', fontsize=12)
plt.ylabel( ylabel: 'Frequency', fontsize=12)
plt.title( label: 'Histogram of 500 Random Numbers', fontsize=14)
plt.axvline(mean_value, color='black', linestyle='dashed', linewidth=2, label=f'Mean: {mean_value:.2f}')
plt.axvline(median_value, color='red', linestyle='dashed', linewidth=2, label=f'Median: {median_value:.
plt.legend()
plt.grid( visible: True, linestyle='--', alpha=0.5)
plt.show()
current_datetime = datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S')
print(f"Current date and time: {current_datetime}")
```

```
/Users/jaspreetsingh/PycharmProjects/pythonProject/pythonProject/venv/bin/python /Applications/PyCharm.app/Contents/plugins/python/helpers/pydev/pydevd.py --multiproces
Connected to pydev debugger (build 232.9559.58)
the random numbers generated are: [12.593998169185014, 6.347205503768389, -14.741547962135911, 16.24404569388207, 11.428062975584389, 4.330208220589871, 13.268199785490
The Mean is : 0.02091005904763053
The Median is: 0.3591665921907943
The Standard Deviation is: 11.357292459087857
Current date and time: 2023-10-01 01:22:45

Process finished with exit code 0
```

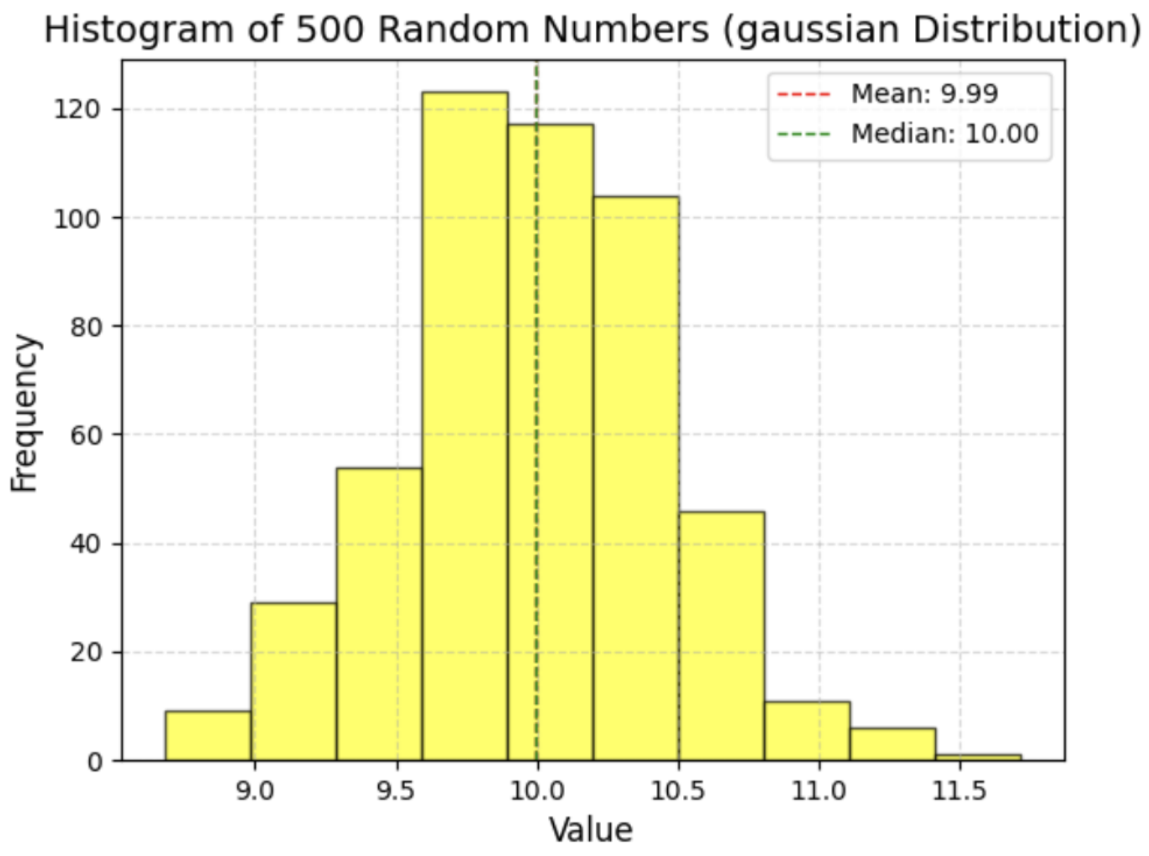
GITHUB LINK :

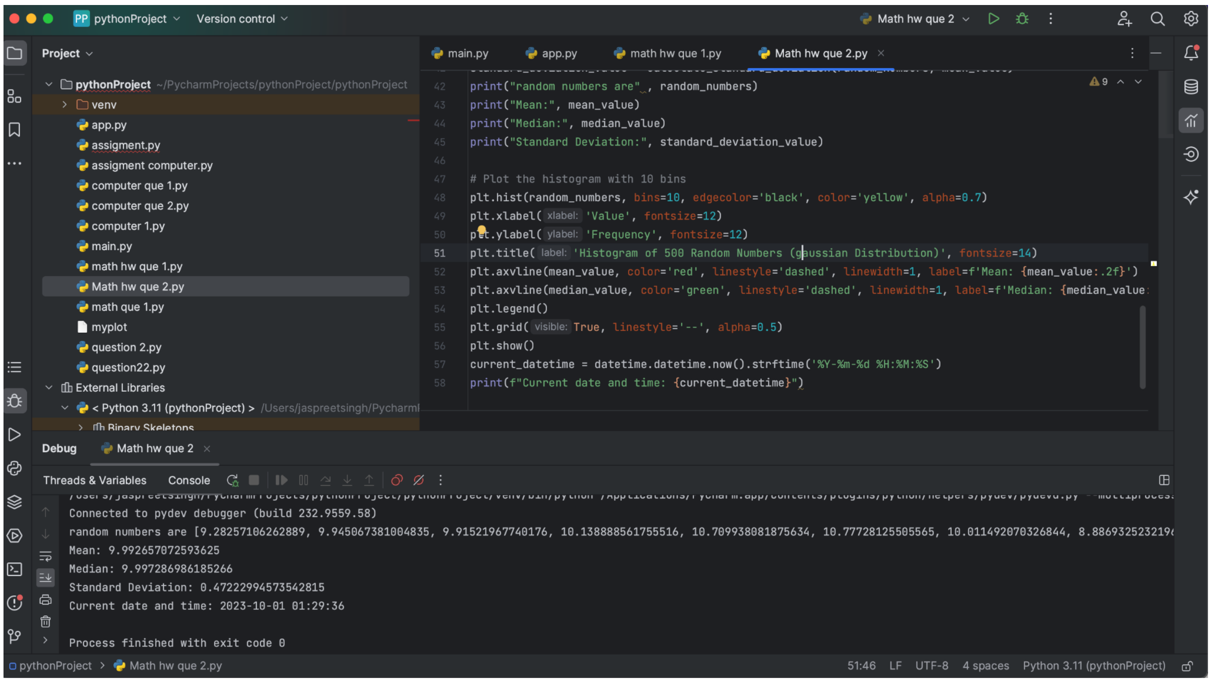
Q2.Similar to the above, write the program to create 500 random numbers with mean = 10 and standard deviation = 0.5 in Gaussian distribution and find the mean, median and standard deviation. After that, plot the histogram with 10 bins. Notice that the only user defined function can be used to calculate the mean, median and standard deviation, don't directly call existing function from Python library.

CODE:

```
import datetime
import random
import math
import matplotlib.pyplot as plt

def calculate_mean(numbers):
    return sum(numbers) / len(numbers)

def calculate_median(numbers):
   sorted_numbers = sorted(numbers)
   n = len(sorted_numbers)
   if n % 2 == 0:
       return (sorted_numbers[n // 2 - 1] + sorted_numbers[n // 2]) / 2
   else:
       return sorted_numbers[n // 2]

def calculate_standard_deviation(numbers, mean):
   variance = sum((x - mean) ** 2 for x in numbers) / len(numbers)
   return variance ** 0.5

def generate_gaussian_distribution(mean, std_dev, n):
   random_numbers = []
   for _ in range(n):
       u1 = random.uniform(0, 1)
       u2 = random.uniform(0, 1)
       z = math.sqrt(-2 * math.log(u1)) * math.cos(2 * math.pi * u2)
       random_numbers.append(mean + std_dev * z)
   return random_numbers

# Parameters for the normal distribution
desired_mean = 10
desired_std_dev = 0.5
num_samples = 500

# Generate random numbers with the desired mean and standard deviation
random_numbers = generate_gaussian_distribution(desired_mean, desired_std_dev, num_samples

# Calculate mean, median, and standard deviation
mean_value = calculate_mean(random_numbers)
median_value = calculate_median(random_numbers)
standard_deviation_value = calculate_standard_deviation(random_numbers, mean_value)
print("random numbers are" , random_numbers)
print("Mean:", mean_value)
print("Median:", median_value)
print("Standard Deviation:", standard_deviation_value)

# Plot the histogram with 10 bins
plt.hist(random_numbers, bins=10, edgecolor='black', color='yellow', alpha=0.7)
plt.xlabel('Value', fontsize=12)
plt.ylabel('Frequency', fontsize=12)
plt.title('Histogram of 500 Random Numbers (gaussian Distribution)', fontsize=14)
plt.axvline(mean_value, color='red', linestyle='dashed', linewidth=1, label=f'Mean: {mean_
plt.axvline(median_value, color='green', linestyle='dashed', linewidth=1, label=f'Median:
plt.legend()
plt.grid(True, linestyle='--', alpha=0.5)
plt.show()
current_datetime = datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S')
print(f"Current date and time: {current_datetime}")
```

OUTPUT:



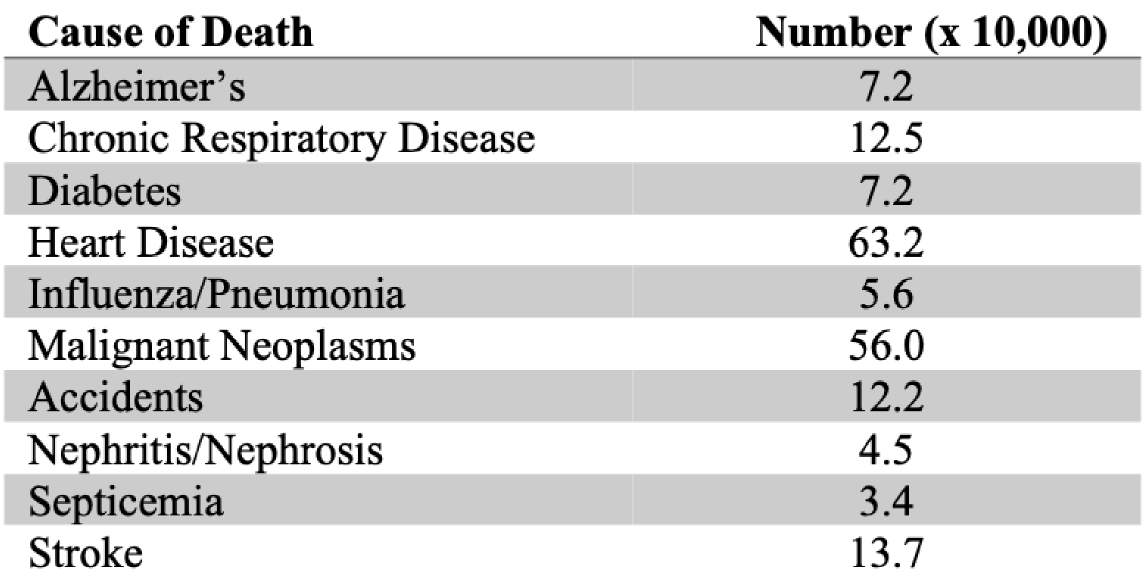


```
print("random numbers are", random_numbers)
print("Mean:", mean_value)
print("Median:", median_value)
print("Standard Deviation:", standard_deviation_value)

# Plot the histogram with 10 bins
plt.hist(random_numbers, bins=10, edgecolor='black', color='yellow', alpha=0.7)
plt.xlabel( xlabel: 'Value', fontsize=12)
plt.ylabel( ylabel: 'Frequency', fontsize=12)
plt.title( label: 'Histogram of 500 Random Numbers (gaussian Distribution)', fontsize=14)
plt.axvline(mean_value, color='red', linestyle='dashed', linewidth=1, label=f'Mean: {mean_value:.2f}')
plt.axvline(median_value, color='green', linestyle='dashed', linewidth=1, label=f'Median: {median_value
plt.legend()
plt.grid( visible: True, linestyle='--', alpha=0.5)
plt.show()
current_datetime = datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S')
print(f"Current date and time: {current_datetime}")
```

```
Connected to pydev debugger (build 232.9559.58)
random numbers are [9.28257106262889, 9.945067381004835, 9.91521967740176, 10.138888561755516, 10.709938081875634, 10.77728125505565, 10.011492070326844, 8.886932523219
Mean: 9.992657072593625
Median: 9.997286986185266
Standard Deviation: 0.47222994573542815
Current date and time: 2023-10-01 01:29:36

Process finished with exit code 0
```

GITHUB LINK :

Q3. The 10-leading causes of death in the United States during 2006 were listed on the Centers for Disease Control and Prevention website. There are a total of 1,855,610 deaths recorded. Plot the Pareto chart in Python or Excel and explain your results.



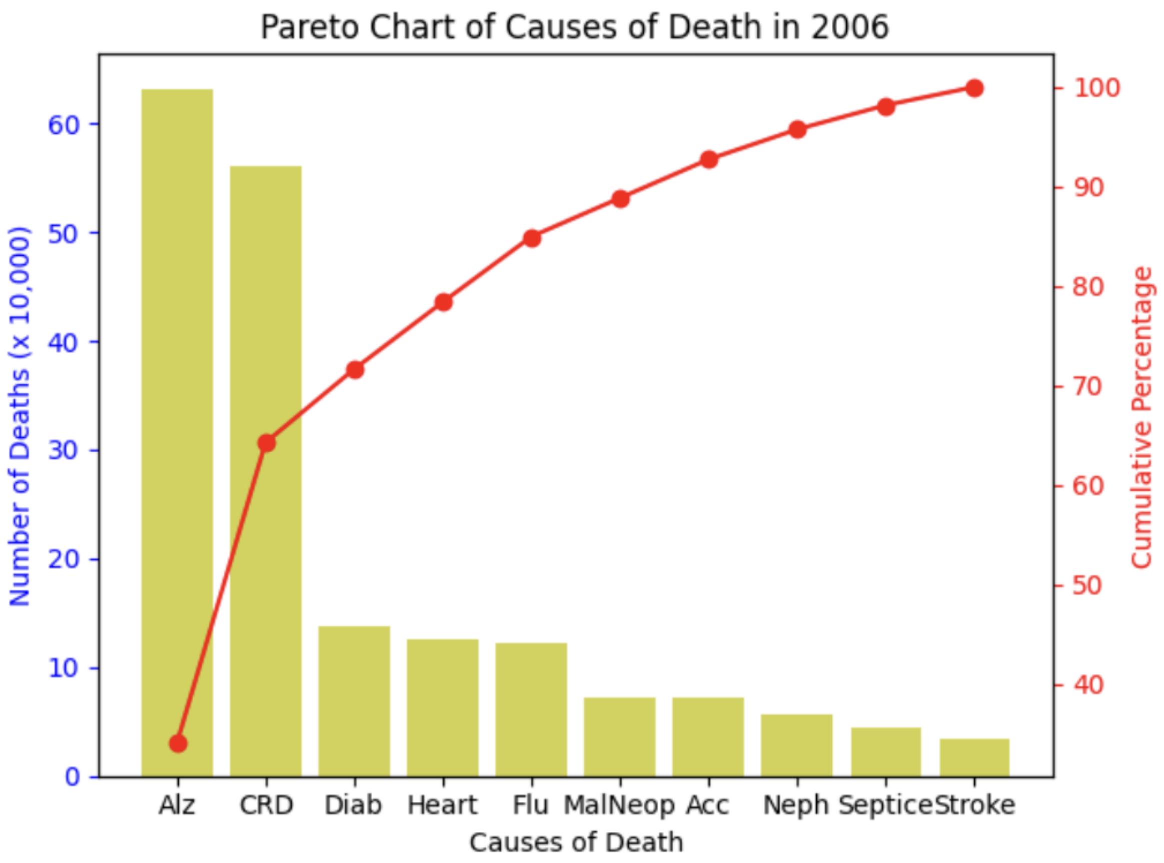
Cause of Death	Number (x 10,000)
Alzheimer’s	7.2
Chronic Respiratory Disease	12.5
Diabetes	7.2
Heart Disease	63.2
Influenza/Pneumonia	5.6
Malignant Neoplasms	56.0
Accidents	12.2
Nephritis/Nephrosis	4.5
Septicemia	3.4
Stroke	13.7

CODE:

```
import datetime
import matplotlib.pyplot as plt

# Data for causes of death and their respective numbers
causes_of_death = [
   'Alz', 'CRD', 'Diab', 'Heart', 'Flu', 'MalNeop', 'Acc',
         'Neph', 'Septice', 'Stroke']

deaths = [
   63.2,56.0, 13.7, 12.5, 12.2, 7.2, 7.2, 5.6, 4.5, 3.4 ]


# Sort the causes and deaths in descending order
sorted_indices = sorted(range(len(deaths)), key=lambda k: deaths[k], reverse=True)
sorted_causes = [causes_of_death[i] for i in sorted_indices]
sorted_deaths = [deaths[i] for i in sorted_indices]

# Calculate cumulative percentages
total_deaths = sum(sorted_deaths)
cumulative_percentages = [sum(sorted_deaths[:i+1]) / total_deaths * 100 for i in range(le

# Create the Pareto chart
fig, ax1 = plt.subplots()

ax1.bar(sorted_causes, sorted_deaths, color='y', alpha=0.7)
ax1.set_xlabel('Causes of Death')
```

```
ax1.set_ylabel('Number of Deaths (x 10,000)', color='b')
ax1.tick_params('y', colors='b')

ax2 = ax1.twinx()
ax2.plot(sorted_causes, cumulative_percentages, color='r', marker='o')
ax2.set_ylabel('Cumulative Percentage', color='r')
ax2.tick_params('y', colors='r')

plt.title('Pareto Chart of Causes of Death in 2006')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
current_datetime = datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S')
print(f"Current date and time: {current_datetime}")
```

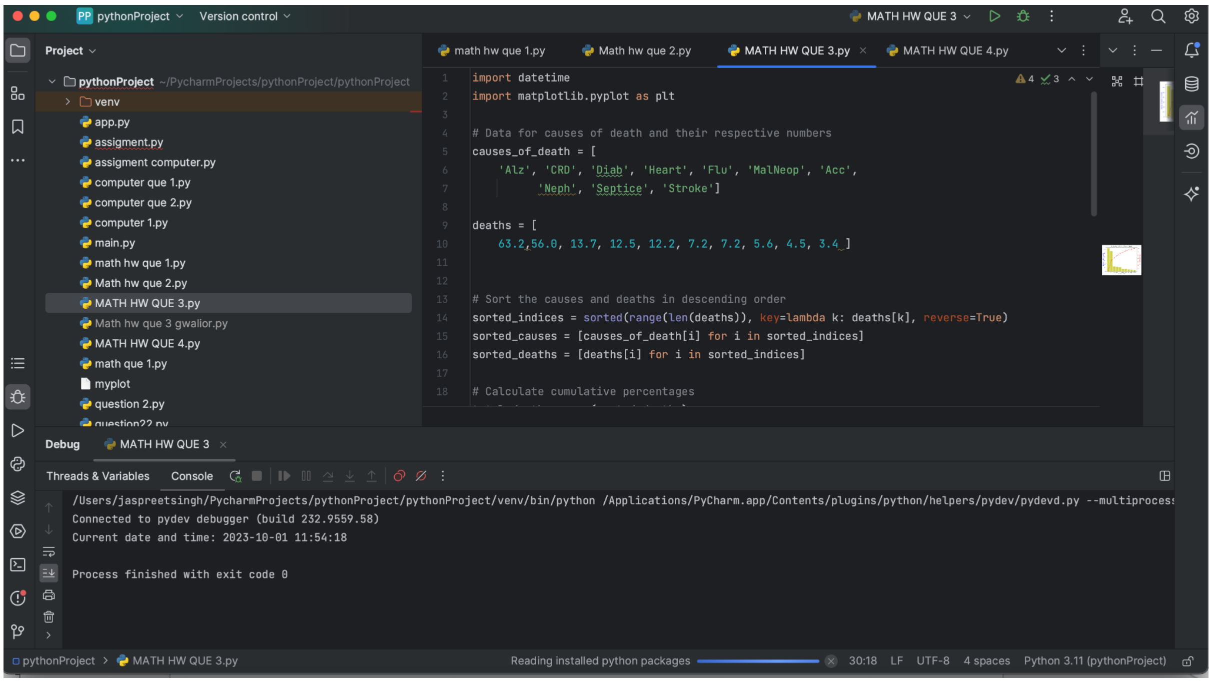
OUTPUT:



We calculate the cumulative percentage of deaths for each cause.

We use matplotlib to plot a bar chart for the number of deaths and a line chart for the cumulative percentage.

The 1-axis represents the causes of death, the left y-axis represents the number of deaths, and the right y-axis represents the cumulative percentage of death.



```
import datetime
import matplotlib.pyplot as plt

# Data for causes of death and their respective numbers
causes_of_death = [
    'Alz', 'CRD', 'Diab', 'Heart', 'Flu', 'MalNeop', 'Acc',
          'Neph', 'Septice', 'Stroke']

deaths = [
    63.2,56.0, 13.7, 12.5, 12.2, 7.2, 7.2, 5.6, 4.5, 3.4 ]


# Sort the causes and deaths in descending order
sorted_indices = sorted(range(len(deaths)), key=lambda k: deaths[k], reverse=True)
sorted_causes = [causes_of_death[i] for i in sorted_indices]
sorted_deaths = [deaths[i] for i in sorted_indices]

# Calculate cumulative percentages
```

```
/Users/jaspreetsingh/PycharmProjects/pythonProject/pythonProject/venv/bin/python /Applications/PyCharm.app/Contents/plugins/python/helpers/pydev/pydevd.py --multiprocess
Connected to pydev debugger (build 232.9559.58)
Current date and time: 2023-10-01 11:54:18

Process finished with exit code 0
```

GITHUB LINK :

Q4.The following data are the ages of 118 known offenders who committed an auto Theft last year in Garden City, Michigan. Write the program to find the median, the mode, Q1 and Q3, P10 and P95

11	14	15	15	16	16	17	18	19	21	25	36
12	14	15	15	16	16	17	18	19	21	25	39
13	14	15	15	16	17	17	18	20	22	26	43
13	14	15	15	16	17	17	18	20	22	26	46
13	14	15	16	16	17	17	18	20	22	27	50
13	14	15	16	16	17	17	19	20	23	27	54
13	14	15	16	16	17	18	19	20	23	29	59
13	15	15	16	16	17	18	19	20	23	30	67
14	15	15	16	16	17	18	19	21	24	31	
14	15	15	16	16	17	18	19	21	24	34	

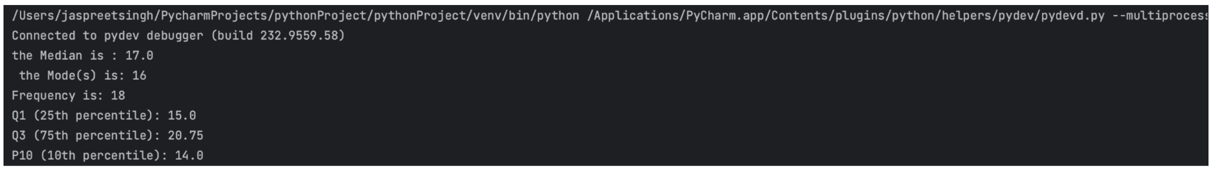
CODE:

```
import datetime

import numpy as np
```

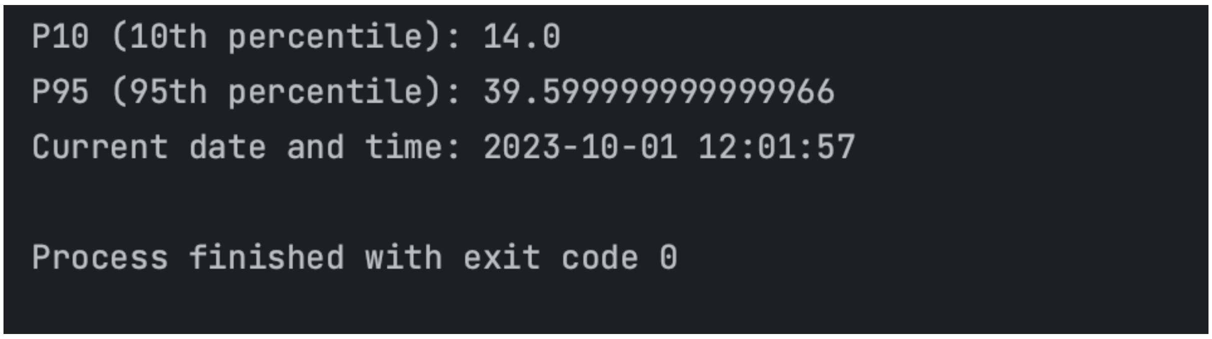
```python
from scipy import stats

# Data
data = [11, 14, 15, 15, 16, 16, 17, 18, 19, 21, 25, 36,
       12, 14, 15, 15, 16, 16, 17, 18, 19, 21, 25, 39,
       13, 14, 15, 15, 16, 17, 17, 18, 20, 22, 26, 43,
       13, 14, 15, 15, 16, 17, 17, 18, 20, 22, 26, 46,
       13, 14, 15, 16, 16, 17, 17, 18, 20, 22, 27, 50,
       13, 14, 15, 16, 16, 17, 17, 19, 20, 23, 27, 54,
       13, 14, 15, 16, 16, 17, 18, 19, 20, 23, 29, 59,
       13, 15, 15, 16, 16, 17, 18, 19, 20, 23, 30, 67,
       14, 15, 15, 16, 16, 17, 18, 19, 21, 24, 31,
       14, 15, 15, 16, 16, 17, 18, 19, 21, 24, 34]

# Calculate Median
median = np.median(data)

# Calculate Mode
mode = stats.mode(data)
mode_values = mode.mode.tolist()
mode_counts = mode.count.tolist()

# Calculate Quartiles (Q1 and Q3)
q1 = np.percentile(data, 25)
q3 = np.percentile(data, 75)

# Calculate Percentiles (P10 and P95)
p10 = np.percentile(data, 10)
p95 = np.percentile(data, 95)

# Print the results
```

```
print("the Median is :", median)

# Print the mode and its counts (handling multiple modes)

print(" the Mode(s) is:", mode_values)

print("Frequency is:", mode_counts)

print("Q1 (25th percentile):", q1)

print("Q3 (75th percentile):", q3)

print("P10 (10th percentile):", p10)

print("P95 (95th percentile):", p95)

current_datetime = datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S')

print(f"Current date and time: {current_datetime}")
```

OUTPUT:



```
/Users/jaspreetsingh/PycharmProjects/pythonProject/pythonProject/venv/bin/python /Applications/PyCharm.app/Contents/plugins/python/helpers/pydev/pydevd.py --multiprocess
Connected to pydev debugger (build 232.9559.58)
the Median is : 17.0
 the Mode(s) is: 16
Frequency is: 18
Q1 (25th percentile): 15.0
Q3 (75th percentile): 20.75
P10 (10th percentile): 14.0
```



```
P10 (10th percentile): 14.0
P95 (95th percentile): 39.599999999999966
Current date and time: 2023-10-01 12:01:57

Process finished with exit code 0
```

GITHUB: