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.

CE305-week1-19757-AadityaPathak.ipynb

```python
        median = numbers[n//2]
    return median

# Function to calculate standard deviation
def calculate_std_dev(numbers, mean):
    return (sum((x - mean) ** 2 for x in numbers) / len(numbers)) ** 0.5

# Calculate mean, median and standard deviation
mean = calculate_mean(numbers)
median = calculate_median(numbers)
std_dev = calculate_std_dev(numbers, mean)

print(f"Mean: {mean}")
print(f"Median: {median}")
print(f"Standard Deviation: {std_dev}")

# Plot histogram
plt.hist(numbers, bins=10, edgecolor='black')
plt.title('Histogram of Random Numbers')
plt.xlabel('Number')
plt.ylabel('Frequency')
plt.show()
```

0s completed at 9:37 AM

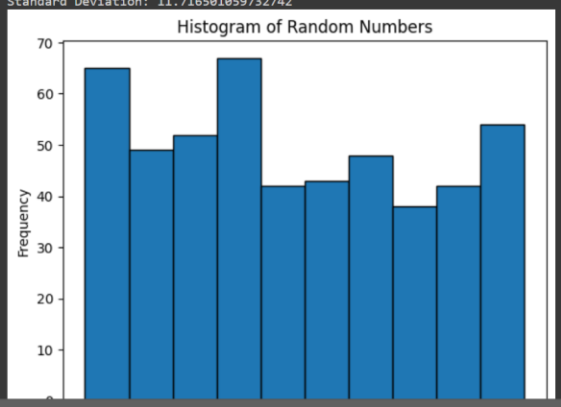
CE305-week1-19757-AadityaPathak.ipynb

```python
#Answer of 1

import random
import matplotlib.pyplot as plt

# Generate 500 random numbers from -20 to +20
numbers = [random.uniform(-20, 20) for _ in range(500)]

# Function to calculate mean
def calculate_mean(numbers):
    return sum(numbers) / len(numbers)

# Function to calculate median
def calculate_median(numbers):
    numbers.sort()
    n = len(numbers)
    if n % 2 == 0:
        median1 = numbers[n//2]
        median2 = numbers[n//2 - 1]
        median = (median1 + median2)/2
    else:
        median = numbers[n//2]
    return median
```

0s completed at 9:37 AM

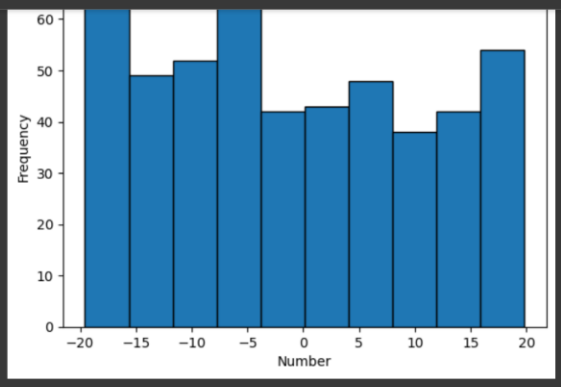
CE305-week1-19757-AadityaPathak.ipynb

```
Mean: -0.8896169979888864
Median: -2.3836974764012595
Standard Deviation: 11.716501059732742
```



0s completed at 9:37 AM

CE305-week1-19757-AadityaPathak.ipynb




```
[7] #Answer of 2
```

0s completed at 9:37 AM

2. 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.

CE305-week1-19757-AadityaPathak.ipynb

```python
#Answer of 2

import random
import matplotlib.pyplot as plt

# Generate 500 random numbers with mean 10 and standard deviation 0.5
numbers = [random.gauss(10, 0.5) for _ in range(500)]

# Function to calculate mean
def calculate_mean(numbers):
    return sum(numbers) / len(numbers)

# Function to calculate median
def calculate_median(numbers):
    numbers.sort()
    n = len(numbers)
    if n % 2 == 0:
        median1 = numbers[n//2]
        median2 = numbers[n//2 - 1]
        median = (median1 + median2)/2
    else:
        median = numbers[n//2]
    return median
```

0s completed at 9:37 AM

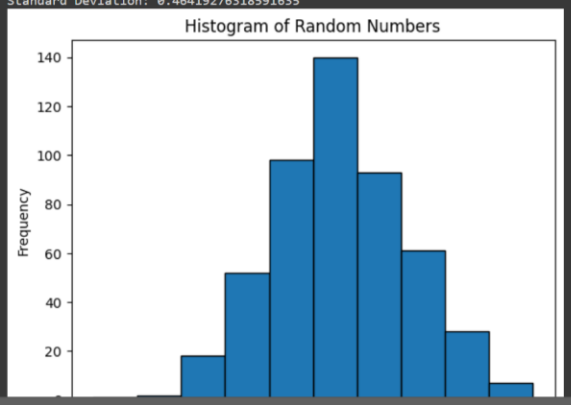
CE305-week1-19757-AadityaPathak.ipynb

```python
        median = numbers[n//2]
    return median

# Function to calculate standard deviation
def calculate_std_dev(numbers, mean):
    return (sum((x - mean) ** 2 for x in numbers) / len(numbers)) ** 0.5

# Calculate mean, median and standard deviation
mean = calculate_mean(numbers)
median = calculate_median(numbers)
std_dev = calculate_std_dev(numbers, mean)

print(f"Mean: {mean}")
print(f"Median: {median}")
print(f"Standard Deviation: {std_dev}")

# Plot histogram
plt.hist(numbers, bins=10, edgecolor='black')
plt.title('Histogram of Random Numbers')
plt.xlabel('Number')
plt.ylabel('Frequency')
plt.show()
```

0s completed at 9:37 AM

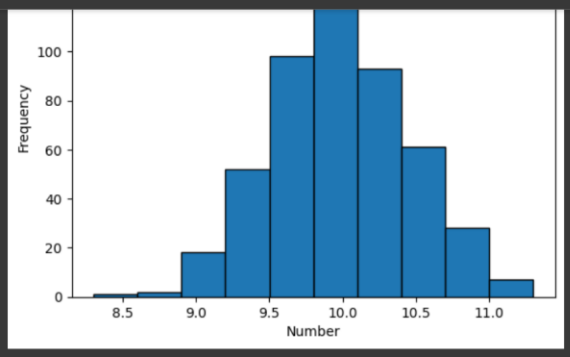
CE305-week1-19757-AadityaPathak.ipynb

```
Mean: 9.991275717652663
Median: 9.985116557208414
Standard Deviation: 0.46419276318591635
```



0s completed at 9:37 AM

CE305-week1-19757-AadityaPathak.ipynb




```
[10] #Answer of 3

     import pandas as pd
     import matplotlib.pyplot as plt
```

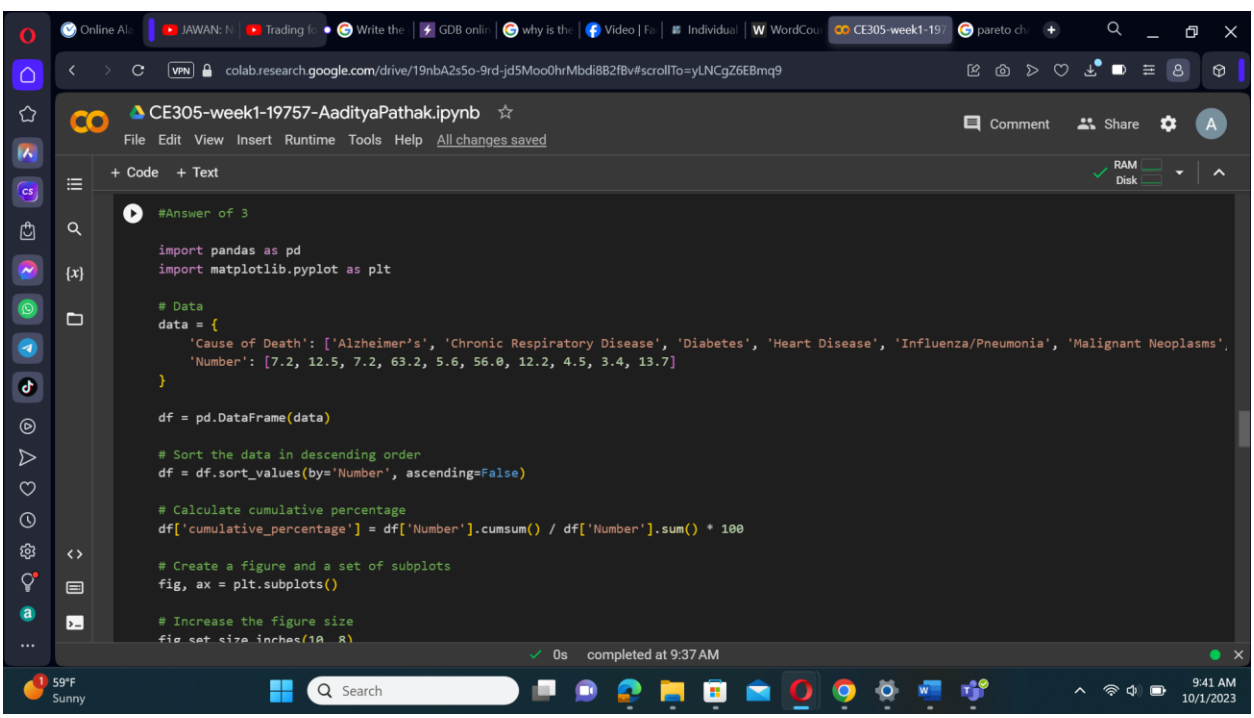
0s completed at 9:37 AM

3. 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

This code first creates a data frame with the given data. It then sorts the data in descending order and calculates the cumulative percentage. The bar plot represents the number of deaths for each cause, and the line plot represents the cumulative percentage. The x-axis labels are the causes of death.

The Pareto chart helps identify the most significant causes of death in terms of their contribution to the total number of deaths. In this case, we can see that Heart Disease and Malignant Neoplasms are the leading causes of death, contributing to a significant portion of the total deaths. The other causes contribute to a lesser extent. This kind of analysis can be useful in prioritizing efforts to prevent these causes of death.

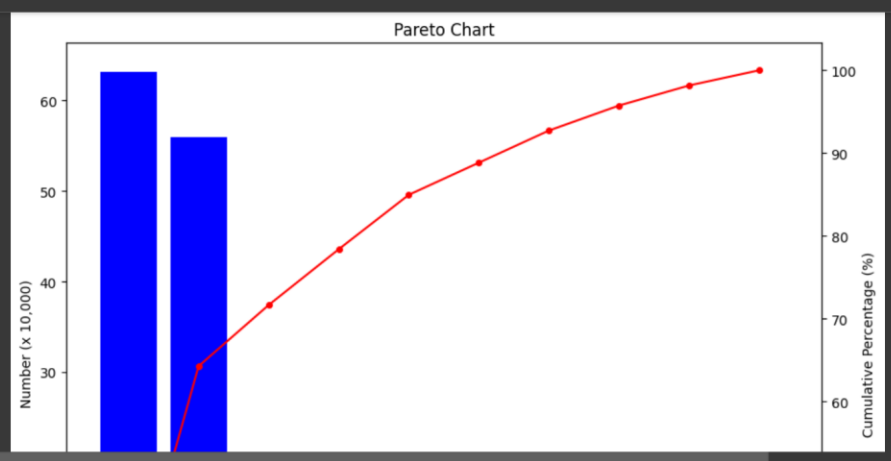


CE305-week1-19757-AadityaPathak.ipynb

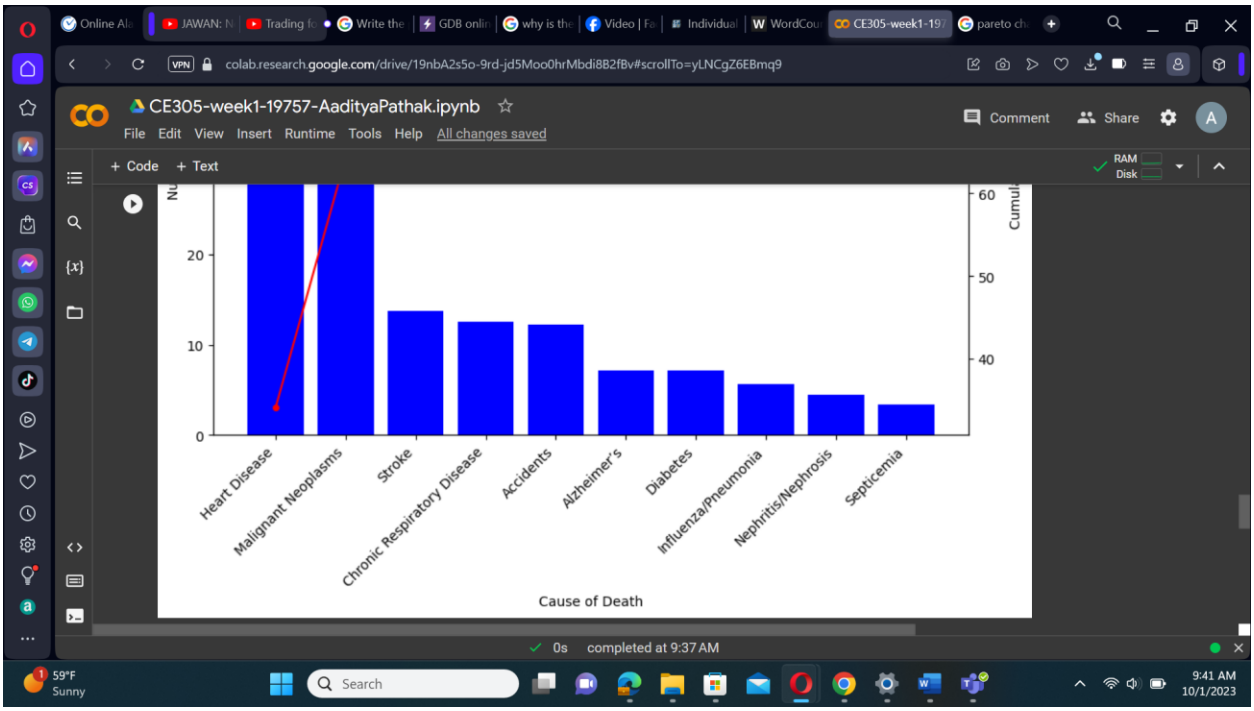
```python
# Increase the figure size
fig.set_size_inches(10, 8)

# Plot bars (Cause of Death)
ax.bar(df['Cause of Death'], df['Number'], color='blue', align='center')

# Plot line (cumulative percentage)
ax2 = ax.twinx()
ax2.plot(df['Cause of Death'], df['cumulative_percentage'], color='red', marker='o', ms=4)

# Add labels and title
plt.title('Pareto Chart')
ax.set_xlabel('Cause of Death')
ax.set_ylabel('Number (x 10,000)')
ax2.set_ylabel('Cumulative Percentage (%)')

# Rotate x-axis labels for better visibility
plt.setp(ax.get_xticklabels(), rotation=45, horizontalalignment='right')

# Show the plot
plt.show()
```

Pareto Chart

0s completed at 9:37 AM

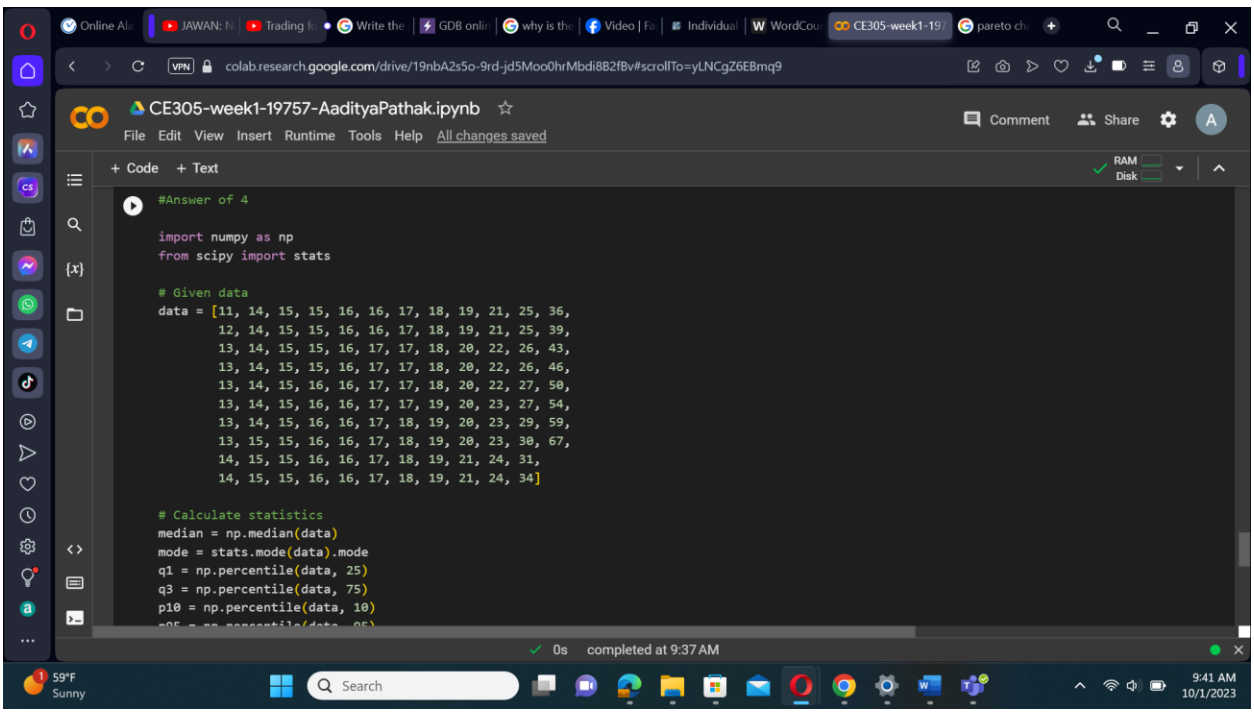


0s completed at 9:37 AM


CE305-week1-19757-AadityaPathak.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
Comment
Share
0
10
20
40
50
60
Heart Disease
Malignant Neoplasms
Stroke
Chronic Respiratory Disease
Accidents
Alzheimer's
Diabetes
Influenza/Pneumonia
Nephritis/Nephrosis
Septicemia
Cause of Death
0s completed at 9:37 AM

4. 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	



CE305-week1-19757-AadityaPathak.ipynb

```python
# Calculate statistics
median = np.median(data)
mode = stats.mode(data).mode
q1 = np.percentile(data, 25)
q3 = np.percentile(data, 75)
p10 = np.percentile(data, 10)
p95 = np.percentile(data, 95)

# Print results
print(f"Median: {median}")
print(f"Mode: {mode}")
print(f"Q1: {q1}")
print(f"Q3: {q3}")
print(f"P10: {p10}")
print(f"P95: {p95}")
```

```
Median: 17.0
Mode: 16
Q1: 15.0
Q3: 20.75
P10: 14.0
P95: 39.599999999999966
```

0s completed at 9:37 AM