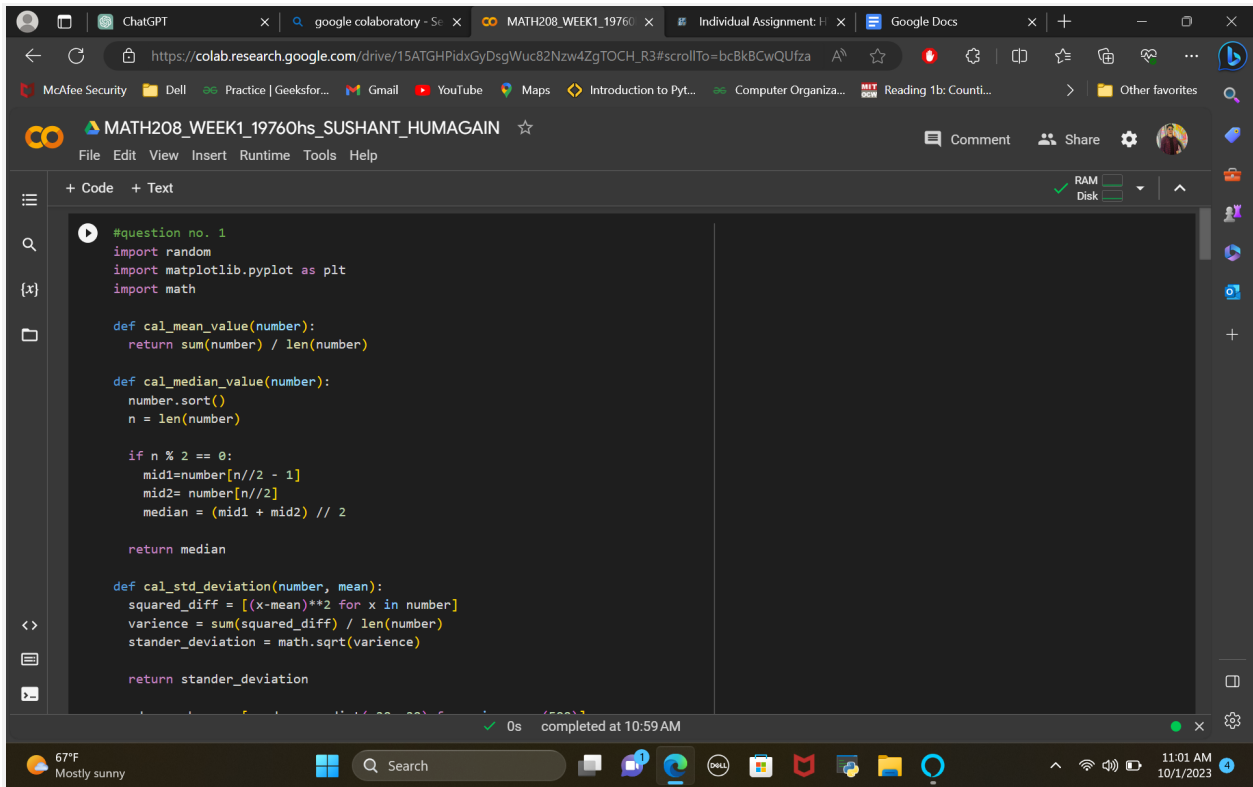


#question no 1.

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 PyWrite 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.



The screenshot shows a Google Colab notebook interface. The browser tabs at the top include 'ChatGPT', 'google colab - S...', 'MATH208_WEEK1_19760', 'Individual Assignment: H...', and 'Google Docs'. The notebook's address bar shows a URL from 'colab.research.google.com'. The notebook title is 'MATH208_WEEK1_19760hs_SUSHANT_HUMAGAIN'. The code editor contains the following Python code:

```
#question no. 1
import random
import matplotlib.pyplot as plt
import math

def cal_mean_value(number):
    return sum(number) / len(number)

def cal_median_value(number):
    number.sort()
    n = len(number)

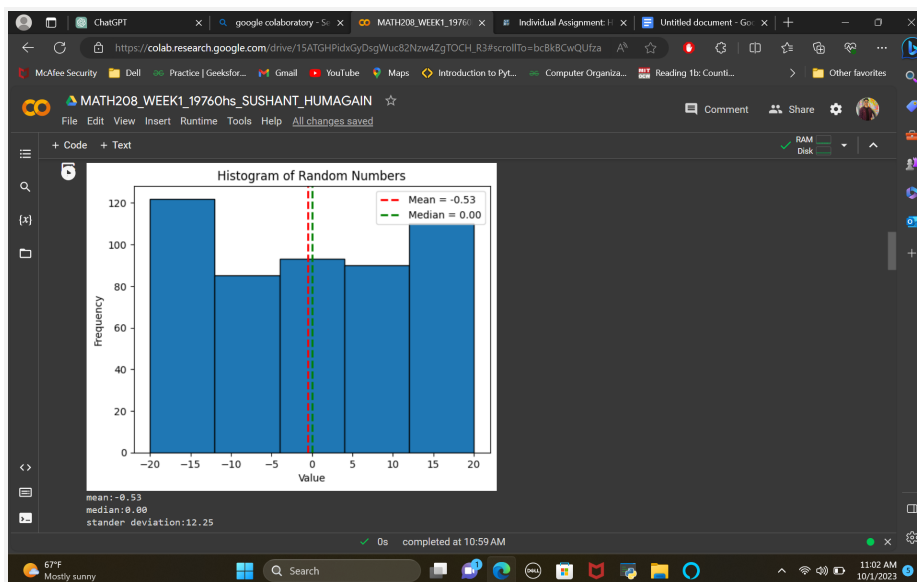
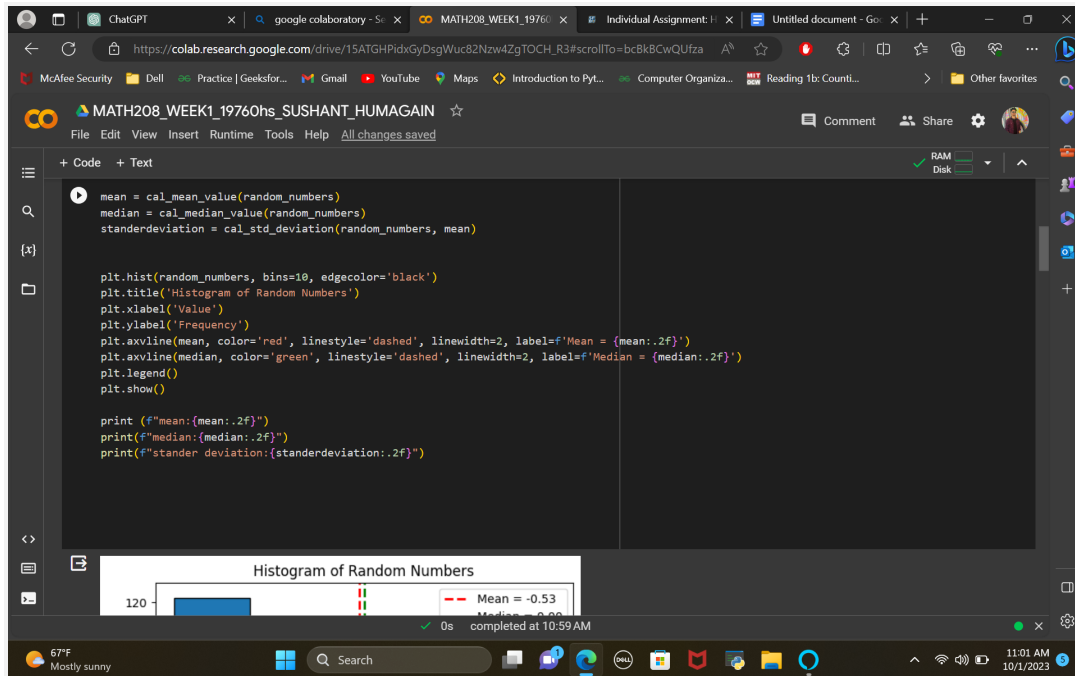
    if n % 2 == 0:
        mid1=number[n//2 - 1]
        mid2= number[n//2]
        median = (mid1 + mid2) // 2

    return median

def cal_std_deviation(number, mean):
    squared_diff = [(x-mean)**2 for x in number]
    variance = sum(squared_diff) / len(number)
    stander_deviation = math.sqrt(variance)

    return stander_deviation
```

The bottom status bar indicates '0s completed at 10:59 AM'. The Windows taskbar at the very bottom shows the date and time as '11:01 AM 10/1/2023'.



#question no .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.

```
#que no.2
import random
import math
import matplotlib.pyplot as plt

def generate_gaussian_numbers(mean, std_deviation, count):
    random_numbers = [random.gauss(mean, std_deviation) for _ in range(count)]
    return random_numbers

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

def cal_median_value(numbers):
    numbers.sort()
    n = len(numbers)
    if n % 2 == 0:
        mid1 = numbers[n // 2 - 1]
        mid2 = numbers[n // 2]
        median = (mid1 + mid2) / 2
    else:
        median = numbers[n // 2]
    return median

def cal_std_deviation(numbers, mean):
    squared_diff = [(x - mean) ** 2 for x in numbers]
    variance = sum(squared_diff) / len(numbers)
```

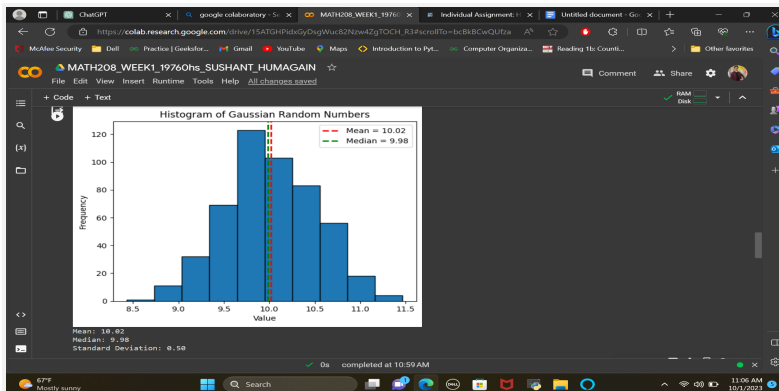
```
def cal_std_deviation(numbers, mean):
    squared_diff = [(x - mean) ** 2 for x in numbers]
    variance = sum(squared_diff) / len(numbers)
    std_deviation = math.sqrt(variance)
    return std_deviation

mean_value = 10
std_deviation_value = 0.5
random_numbers = generate_gaussian_numbers(mean_value, std_deviation_value, 500)

mean = cal_mean_value(random_numbers)
median = cal_median_value(random_numbers)
std_deviation = cal_std_deviation(random_numbers, mean)

plt.hist(random_numbers, bins=10, edgecolor='black')
plt.title('Histogram of Gaussian Random Numbers')
plt.xlabel('Value')
plt.ylabel('Frequency')
plt.axvline(mean, color='red', linestyle='dashed', linewidth=2, label=f'Mean = {mean:.2f}')
plt.axvline(median, color='green', linestyle='dashed', linewidth=2, label=f'Median = {median:.2f}')
plt.legend()
plt.show()

print(f'Mean: {mean:.2f}')
print(f'Median: {median:.2f}')
print(f'Standard Deviation: {std_deviation:.2f}')
```



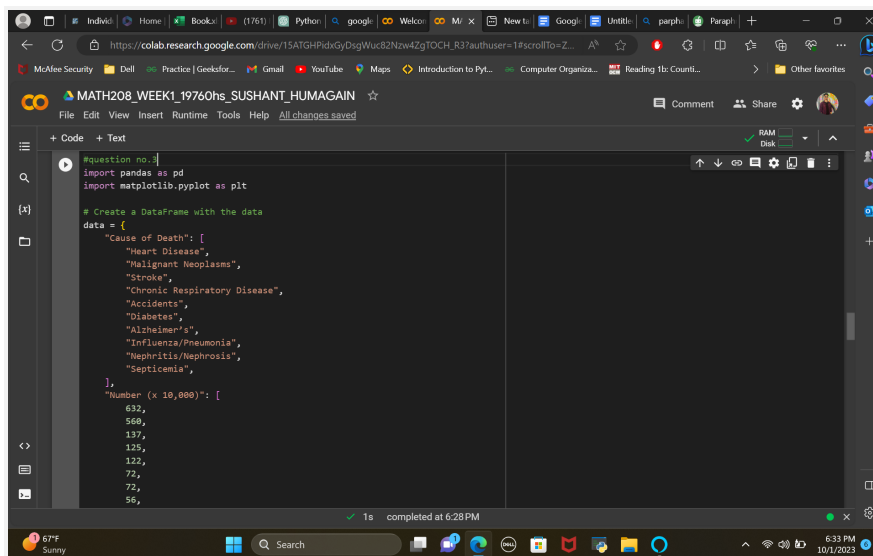
Question no. 3

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

⇒

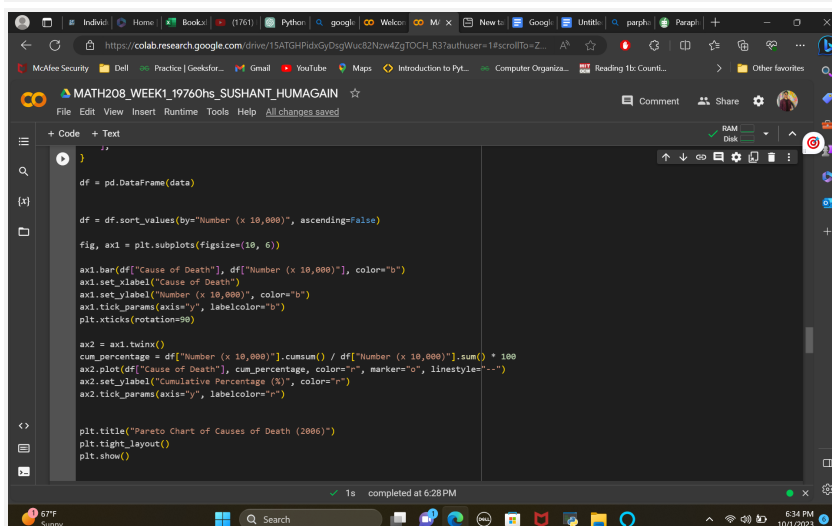
This code first creates a DataFrame using the given data and then sorts it in descending order of frequency. It then plots a bar chart for the cumulative percentage on the right y-axis.

The resulting chart will show the causes of death in descending order of frequency, with the most significant contributors on the left. This helps identify the most significant contributors, following the Pareto principle.



```
#Question no. 3
import pandas as pd
import matplotlib.pyplot as plt

# Create a DataFrame with the data
data = {
    "Cause of Death": [
        "Heart Disease",
        "Malignant Neoplasms",
        "Stroke",
        "Chronic Respiratory Disease",
        "Accidents",
        "Diabetes",
        "Alzheimer's",
        "Influenza/Pneumonia",
        "Nephritis/Nephrosis",
        "Septicemia",
    ],
    "Number (x 10,000)": [
        632,
        560,
        137,
        125,
        122,
        72,
        72,
        56,
    ]
}
```



```
df = pd.DataFrame(data)

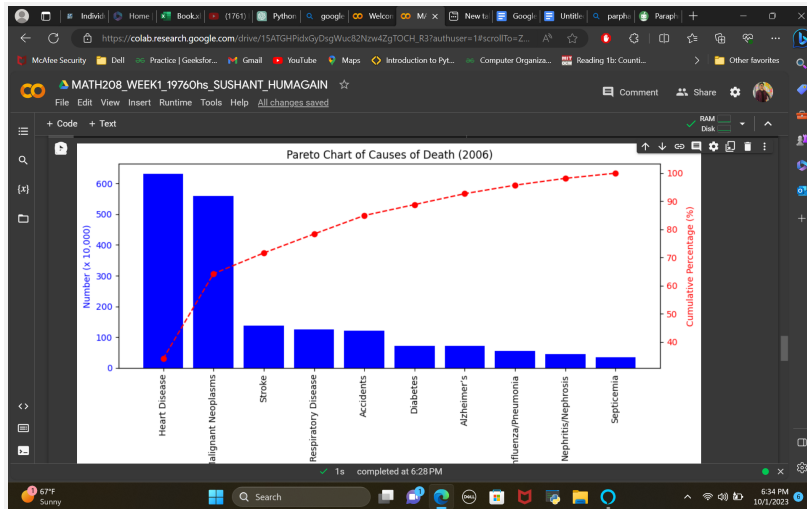
df = df.sort_values(by="Number (x 10,000)", ascending=False)

fig, ax1 = plt.subplots(figsize=(10, 6))

ax1.bar(df["Cause of Death"], df["Number (x 10,000)"], color="b")
ax1.set_xlabel("Cause of Death")
ax1.set_ylabel("Number (x 10,000)", color="b")
ax1.tick_params(axis="x", labelcolor="b")
plt.xticks(rotation=90)

ax2 = ax1.twinx()
cum_percentage = df["Number (x 10,000)"].cumsum() / df["Number (x 10,000)"].sum() * 100
ax2.plot(df["Cause of Death"], cum_percentage, color="r", marker="o", linestyle="--")
ax2.set_ylabel("Cumulative Percentage (%)", color="r")
ax2.tick_params(axis="y", labelcolor="r")

plt.title("Pareto Chart of Causes of Death (2006)")
plt.tight_layout()
plt.show()
```



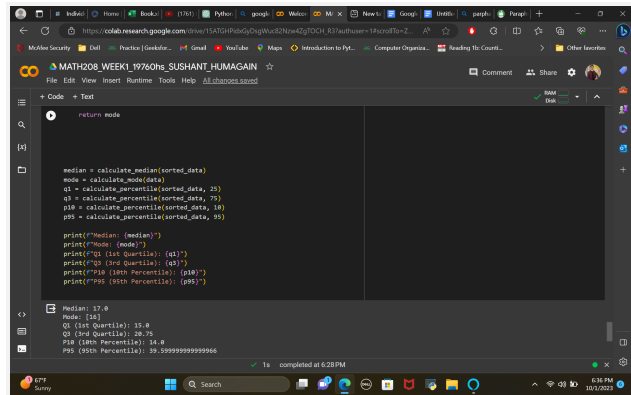
#question no 4

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.

```
#question no.4
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, 16, 16, 17, 18, 19, 21, 24, 35,
    14, 15, 15, 16, 16, 17, 18, 19, 21, 24, 34
]

sorted_data = sorted(data)

def calculate_percentile(data, percentile):
    n = len(data)
    index = (percentile / 100) * (n - 1)
    if index.is_integer():
        return data[int(index)]
    else:
        lower_index = int(index)
        upper_index = lower_index + 1
        lower_value = data[lower_index]
        upper_value = data[upper_index]
        interpolation = (index - lower_index) * (upper_value - lower_value)
        return lower_value + interpolation
```



The screenshot shows a Jupyter Notebook interface with a browser window at the top. The notebook is titled "MATH008 WEEK1 1976Ch2_SUSHANT HUMAGAIN". The code cell contains a Python script that calculates the median and percentiles of a dataset. The output cell shows the results of the calculations.

```
return med

median = calculate_median(sorted_data)
mode = calculate_mode(data)
q1 = calculate_percentile(sorted_data, 25)
q3 = calculate_percentile(sorted_data, 75)
p10 = calculate_percentile(sorted_data, 10)
p90 = calculate_percentile(sorted_data, 90)

print("Median: (median)")
print("Mode: (mode)")
print("Q1 (1st Quartile): (q1)")
print("Q3 (3rd Quartile): (q3)")
print("P10 (10th Percentile): (p10)")
print("P90 (90th Percentile): (p90)")
```

Median: 17.8
Mode: [6]
Q1 (1st Quartile): 15.8
Q3 (3rd Quartile): 20.75
P10 (10th Percentile): 14.8
P90 (90th Percentile): 25.555555555555555

1s completed at 6:28 PM