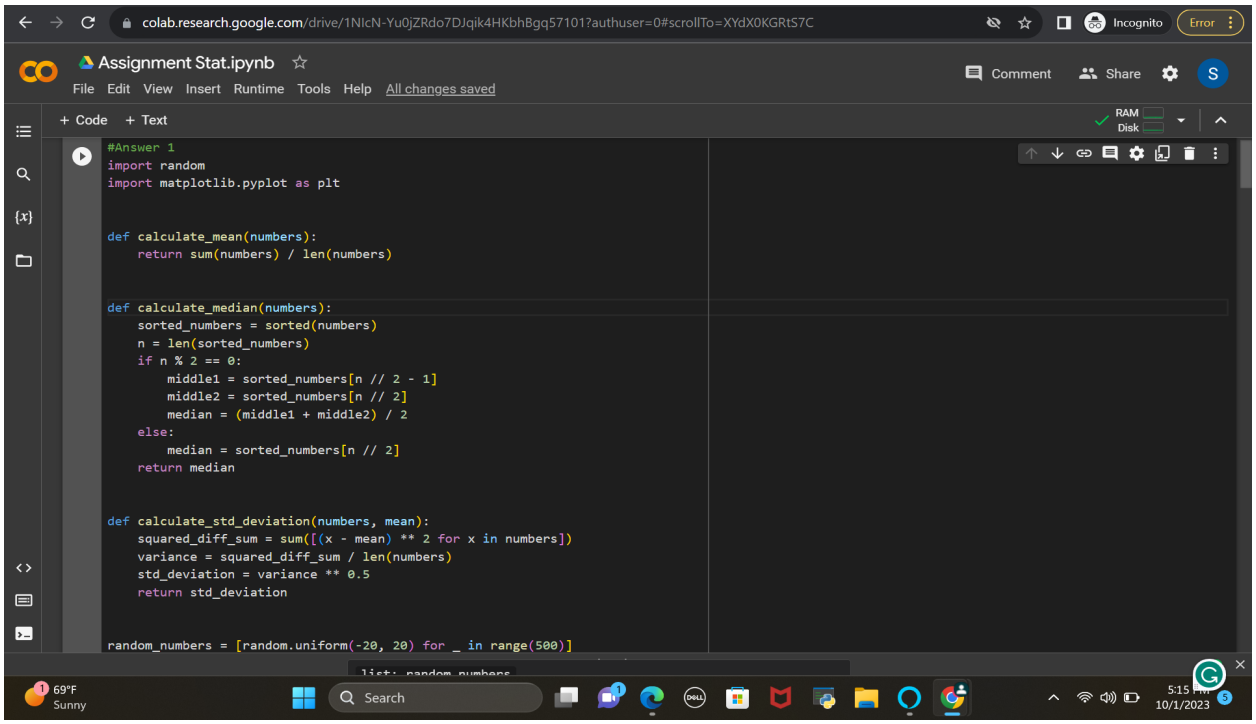
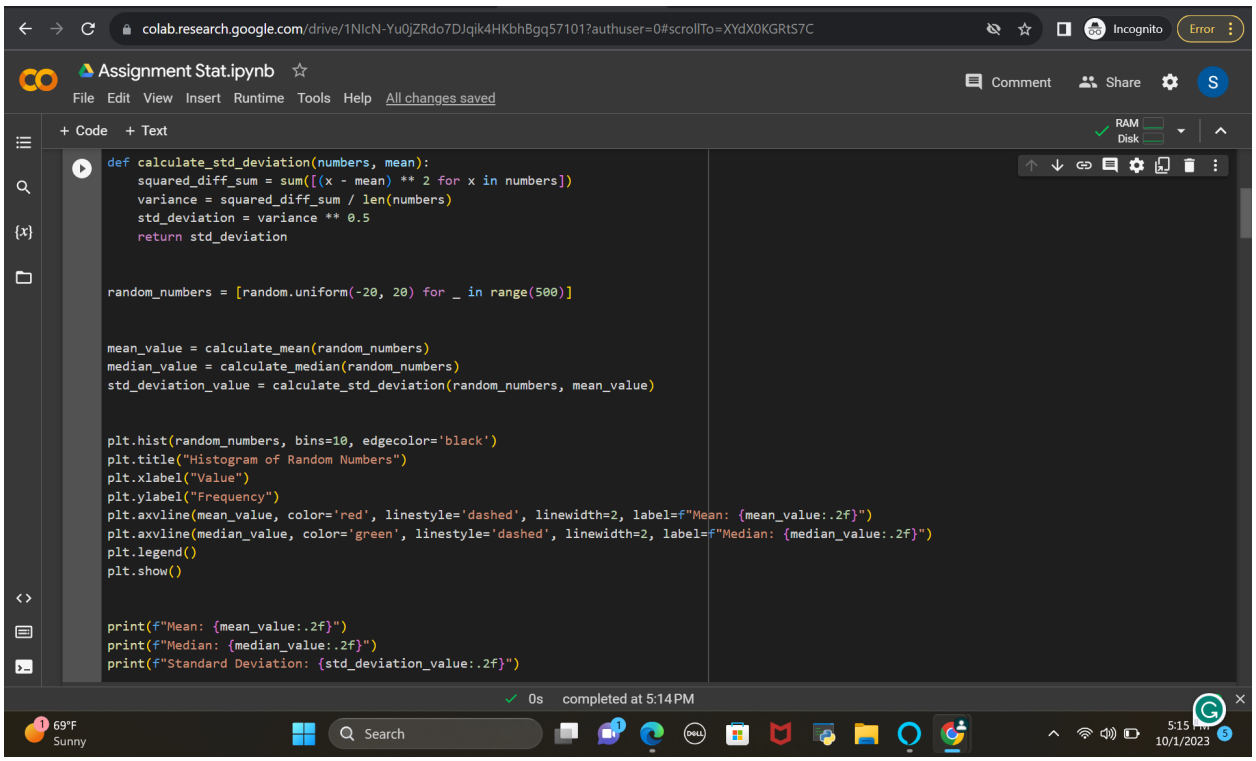
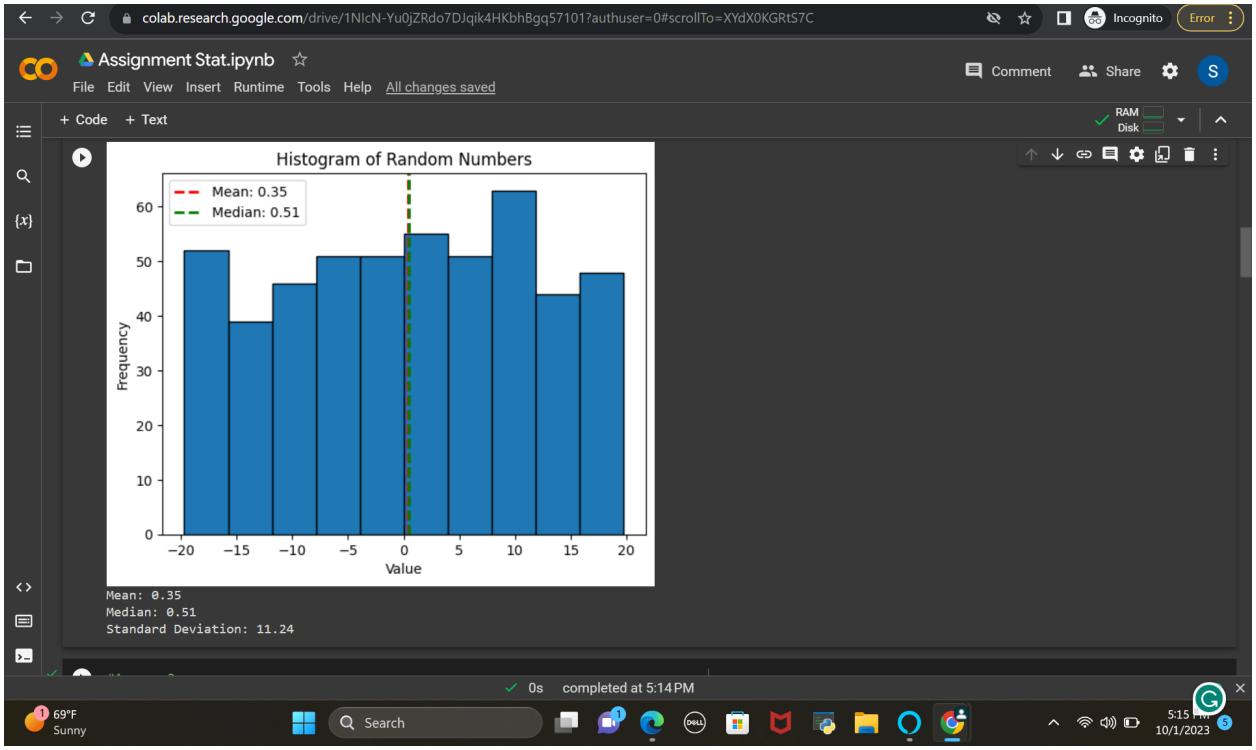
Samir Khadka 19701ks Computer Science (Assignment 1)

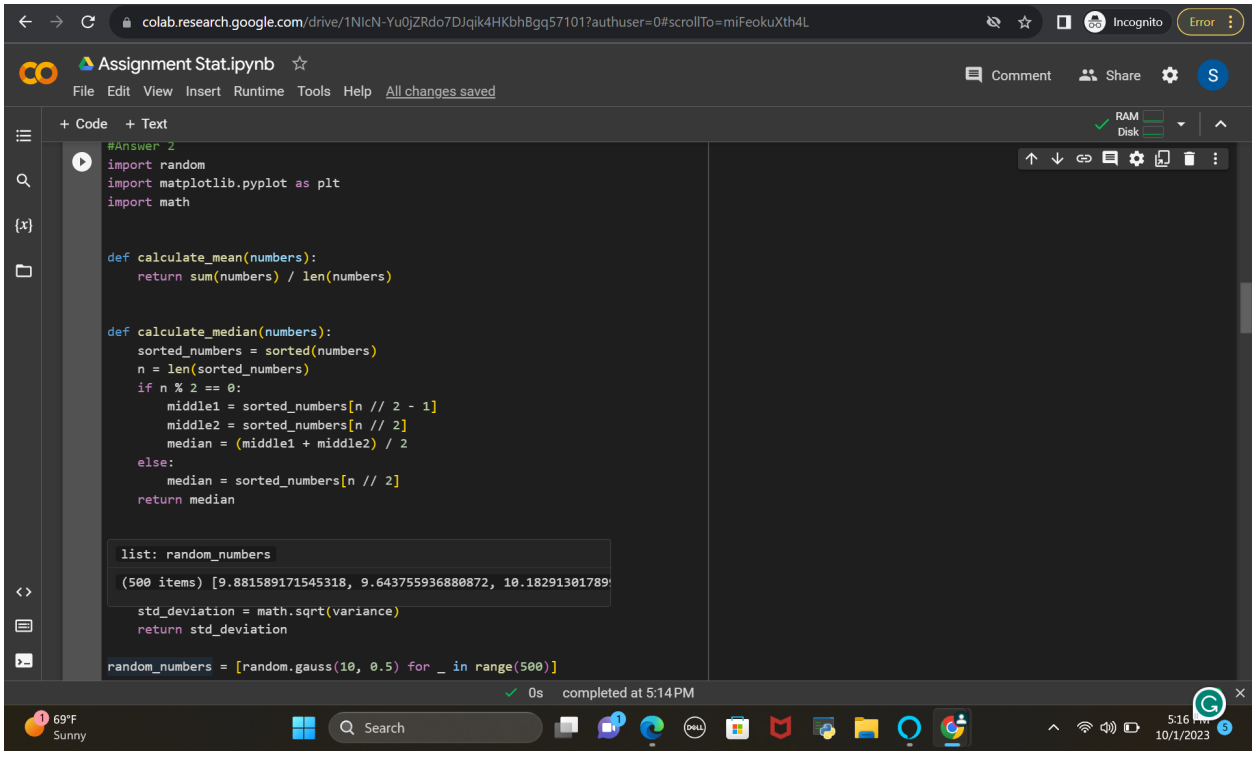
Question 1

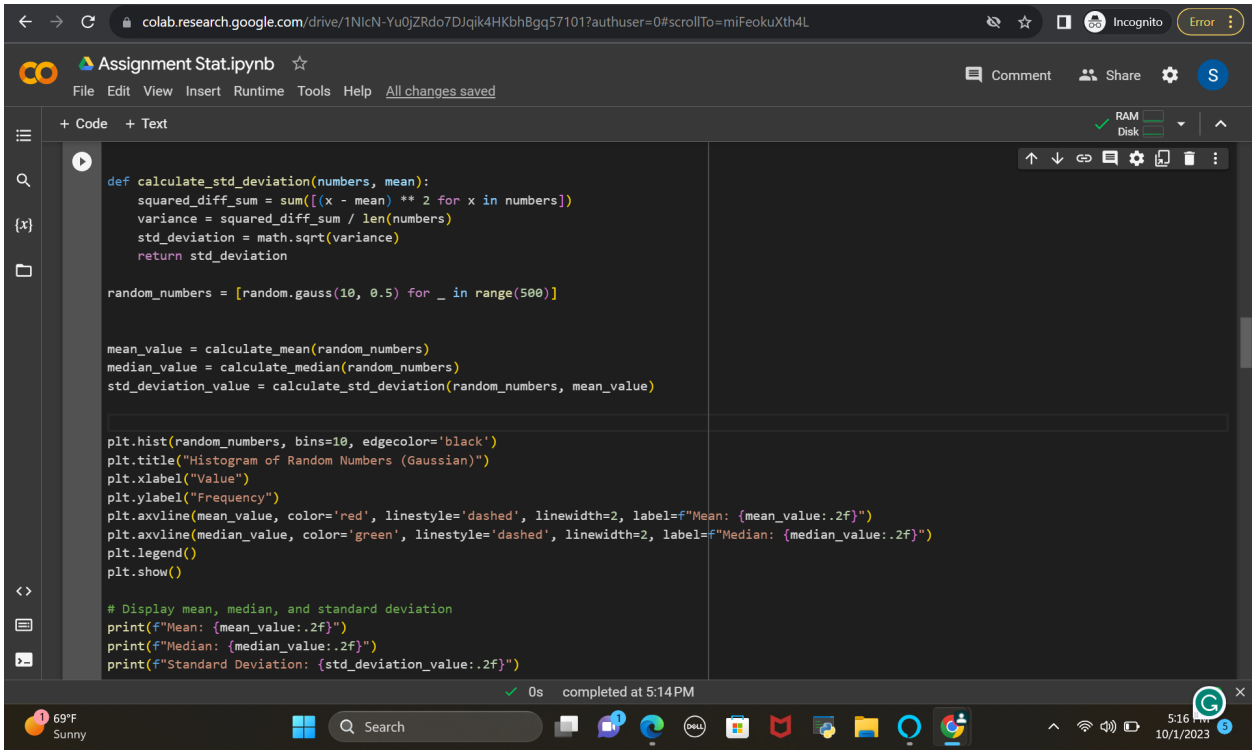


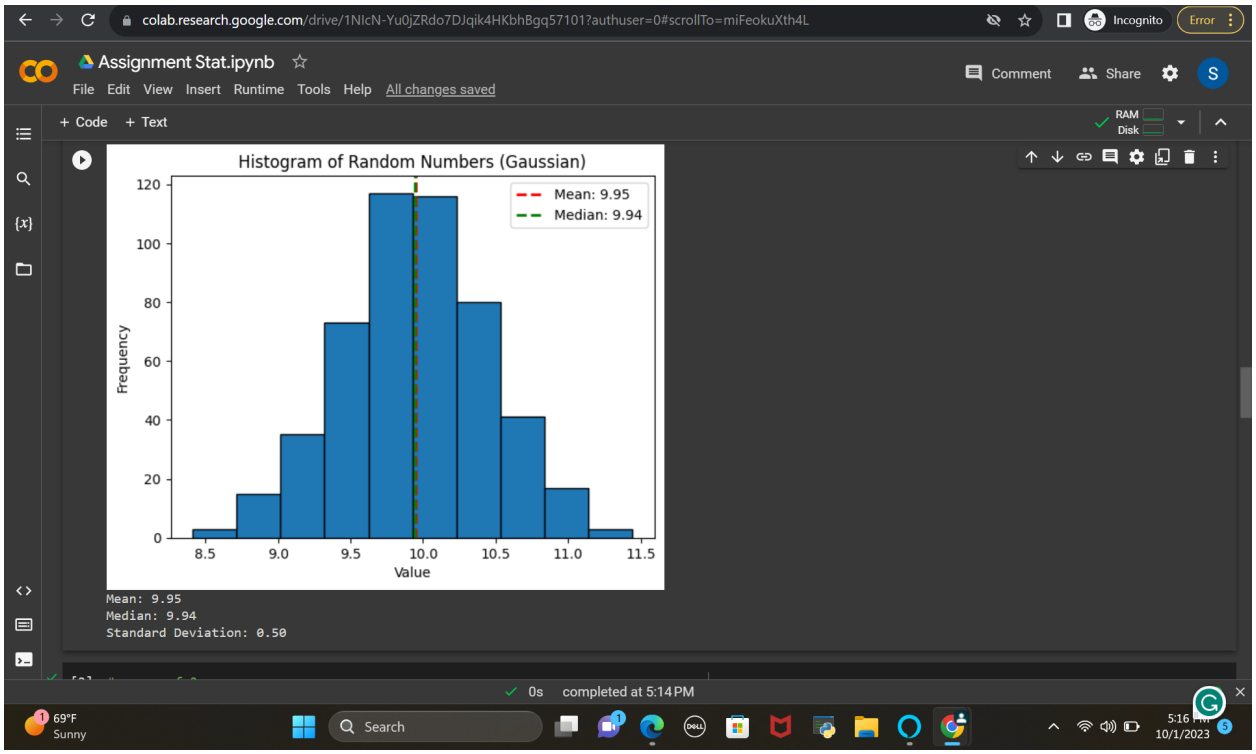



colab.research.google.com/drive/1NIcN-Yu0jZRdo7DJqik4HKbhBgq57101?authuser=0#scrollTo=XYdX0KGRtS7C
Incognito
Error
Assignment Stat.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
S
+ Code + Text
RAM
Disk
Histogram of Random Numbers
Mean: 0.35
Median: 0.51
Frequency
Value
0
10
20
30
40
50
60
−20
−15
−10
−5
0
5
10
15
20
Mean: 0.35
Median: 0.51
Standard Deviation: 11.24
0s completed at 5:14PM
69°F
Sunny
Search
5:15
10/1/2023

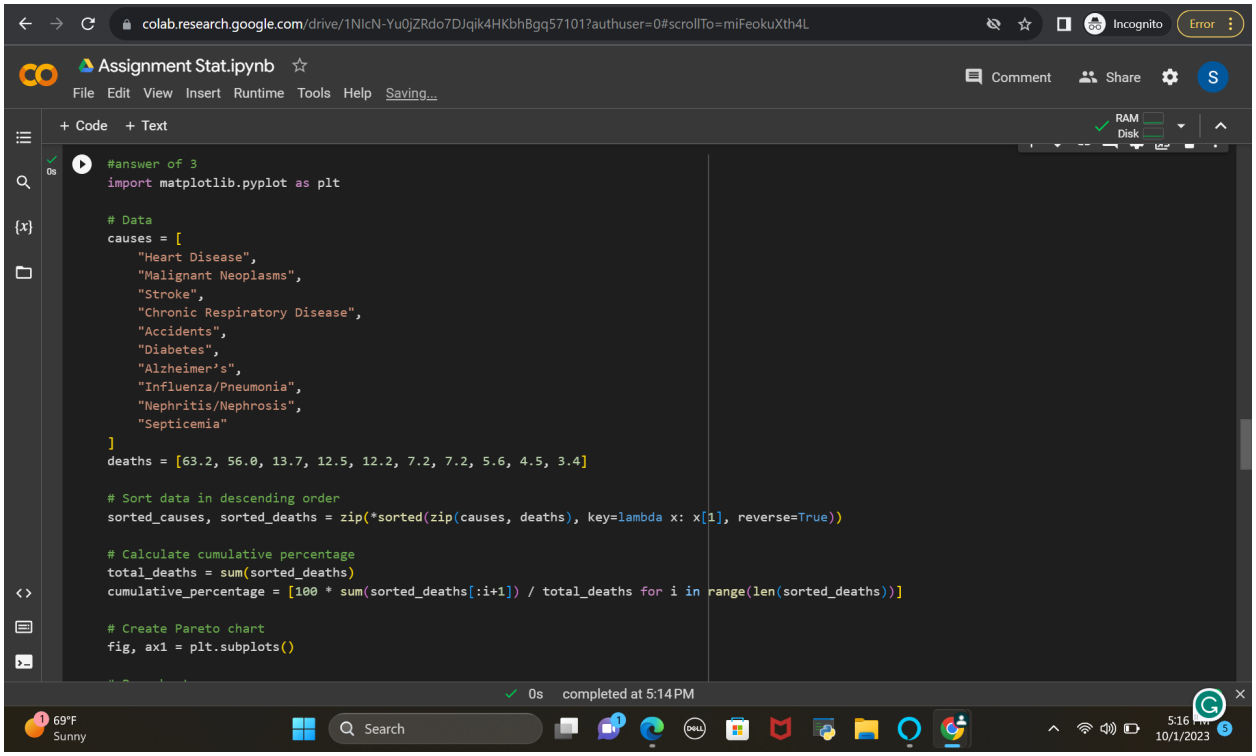
Question 2


colab.research.google.com/drive/1NIcN-Yu0jZRdo7DJqik4HKbhBgq57101?authuser=0#scrollTo=miFeokuXth4L
Incognito
Error
Assignment Stat.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
S
+ Code + Text
RAM
Disk
#Answer 2
import random
import matplotlib.pyplot as plt
import math
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:
middle1 = sorted_numbers[n // 2 - 1]
middle2 = sorted_numbers[n // 2]
median = (middle1 + middle2) / 2
else:
median = sorted_numbers[n // 2]
return median
list: random_numbers
(500 items) [9.881589171545318, 9.643755936880872, 10.18291301789
std_deviation = math.sqrt(variance)
return std_deviation
random_numbers = [random.gauss(10, 0.5) for _ in range(500)]
0s completed at 5:14PM
69°F
Sunny
Search
5:16
10/1/2023

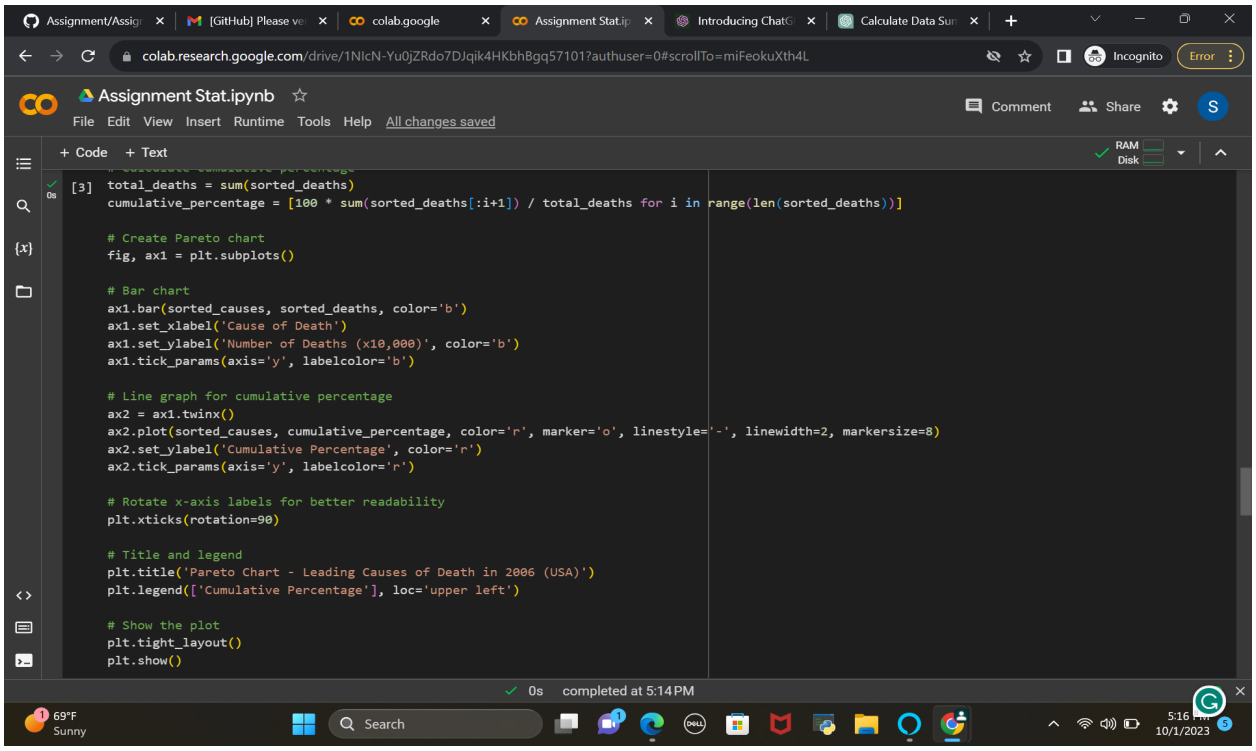

colab.research.google.com/drive/1NIcN-Yu0jZRdo7DJqik4HKbhBgq57101?authuser=0#scrollTo=miFeokuXth4L
Incognito
Error
Assignment Stat.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
def calculate_std_deviation(numbers, mean):
squared_diff_sum = sum([(x - mean) ** 2 for x in numbers])
variance = squared_diff_sum / len(numbers)
std_deviation = math.sqrt(variance)
return std_deviation
random_numbers = [random.gauss(10, 0.5) for _ in range(500)]
mean_value = calculate_mean(random_numbers)
median_value = calculate_median(random_numbers)
std_deviation_value = calculate_std_deviation(random_numbers, mean_value)
plt.hist(random_numbers, bins=10, edgecolor='black')
plt.title("Histogram of Random Numbers (Gaussian)")
plt.xlabel("Value")
plt.ylabel("Frequency")
plt.axvline(mean_value, color='red', linestyle='dashed', linewidth=2, label=f"Mean: {mean_value:.2f}")
plt.axvline(median_value, color='green', linestyle='dashed', linewidth=2, label=f"Median: {median_value:.2f}")
plt.legend()
plt.show()
# Display mean, median, and standard deviation
print(f"Mean: {mean_value:.2f}")
print(f"Median: {median_value:.2f}")
print(f"Standard Deviation: {std_deviation_value:.2f}")
0s completed at 5:14PM
69°F
Sunny
Search
5:16
10/1/2023


colab.research.google.com/drive/1NIcN-Yu0jZRdo7DJqik4HKbhBgq57101?authuser=0#scrollTo=miFeokuXth4L
Incognito
Error
Assignment Stat.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Histogram of Random Numbers (Gaussian)
Mean: 9.95
Median: 9.94
Frequency
Value
0
20
40
60
80
100
120
8.5
9.0
9.5
10.0
10.5
11.0
11.5
Mean: 9.95
Median: 9.94
Standard Deviation: 0.50
0s completed at 5:14PM
69°F
Sunny
Search
5:16
10/1/2023

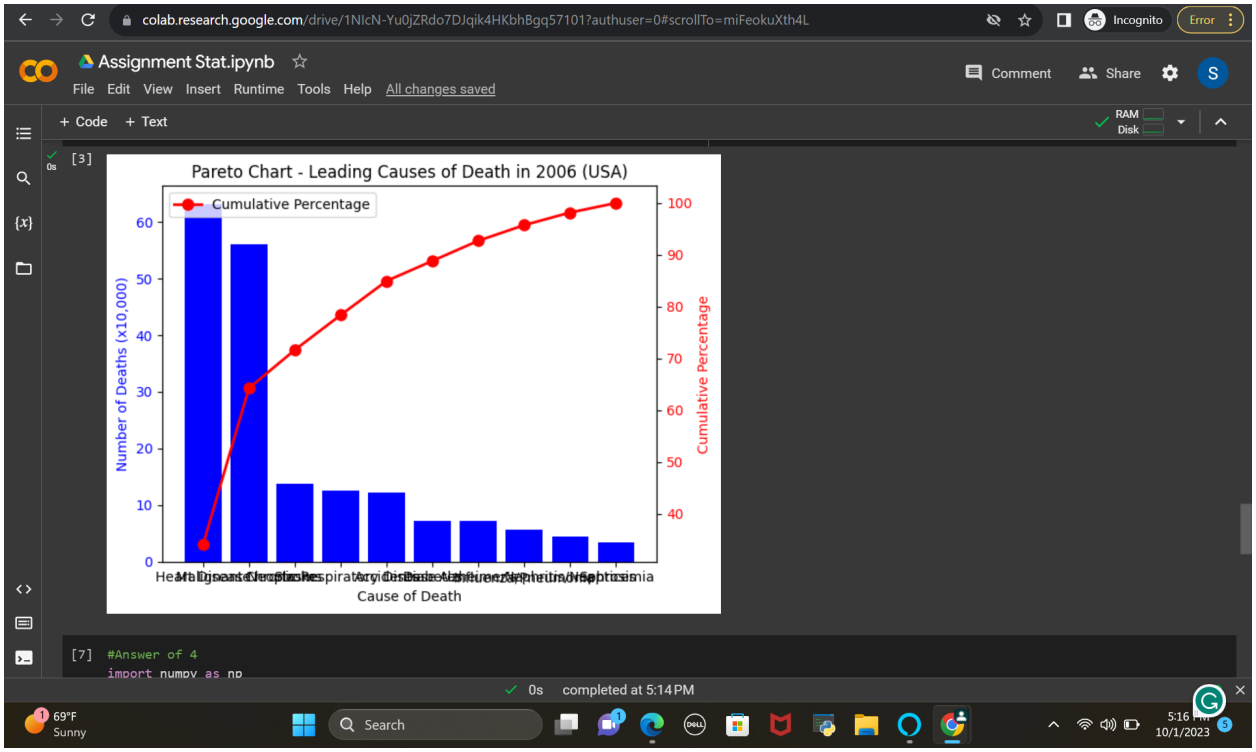
Question 3



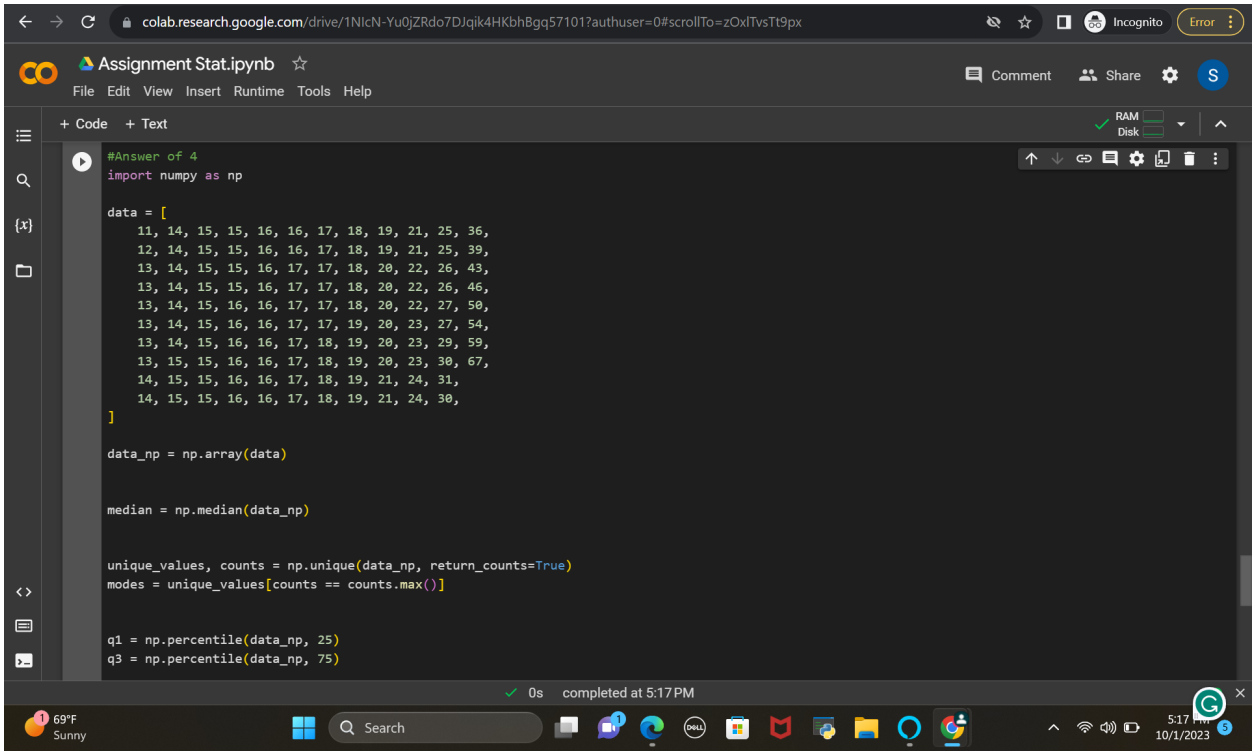
```
#answer of 3
import matplotlib.pyplot as plt

# Data
causes = [
    "Heart Disease",
    "Malignant Neoplasms",
    "Stroke",
    "Chronic Respiratory Disease",
    "Accidents",
    "Diabetes",
    "Alzheimer's",
    "Influenza/Pneumonia",
    "Nephritis/Nephrosis",
    "Septicemia"
]
deaths = [63.2, 56.0, 13.7, 12.5, 12.2, 7.2, 7.2, 5.6, 4.5, 3.4]

# Sort data in descending order
sorted_causes, sorted_deaths = zip(*sorted(zip(causes, deaths), key=lambda x: x[1], reverse=True))

# Calculate cumulative percentage
total_deaths = sum(sorted_deaths)
cumulative_percentage = [100 * sum(sorted_deaths[:i+1]) / total_deaths for i in range(len(sorted_deaths))]

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

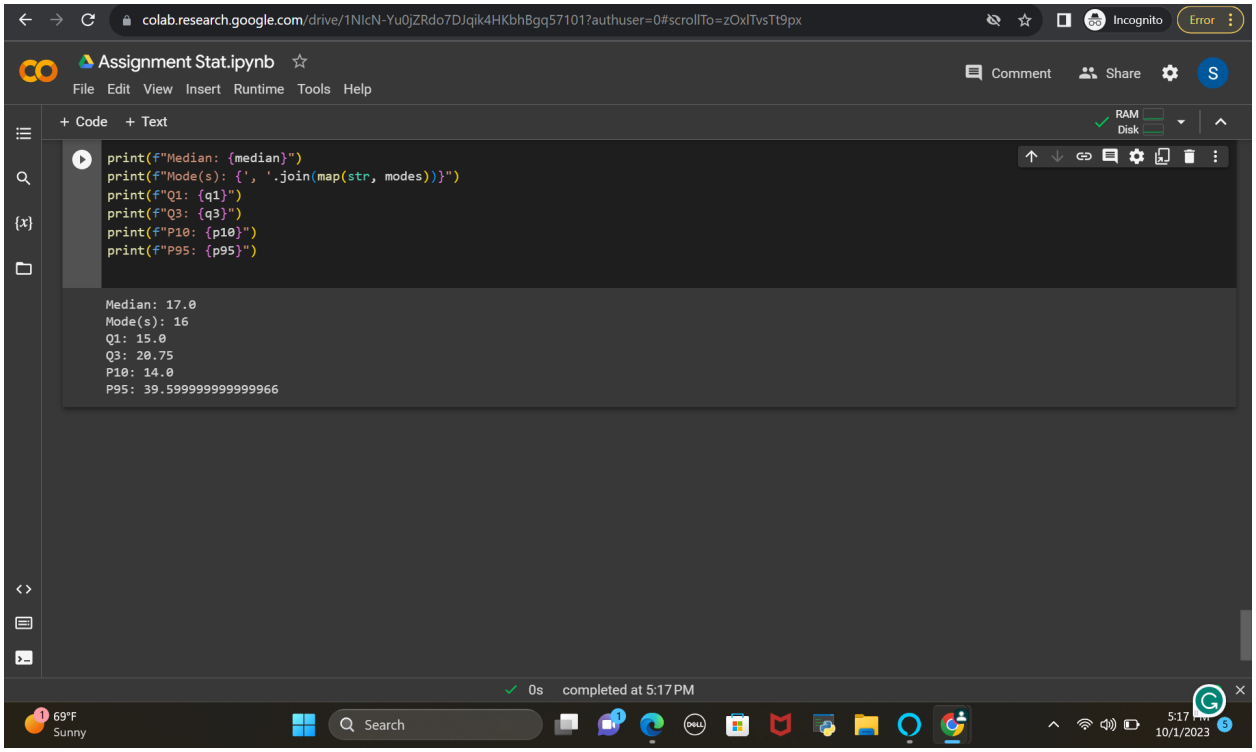


```
[3] total_deaths = sum(sorted_deaths)
    cumulative_percentage = [100 * sum(sorted_deaths[:i+1]) / total_deaths for i in range(len(sorted_deaths))]

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

    # Bar chart
    ax1.bar(sorted_causes, sorted_deaths, color='b')
    ax1.set_xlabel('Cause of Death')
    ax1.set_ylabel('Number of Deaths (x10,000)', color='b')
    ax1.tick_params(axis='y', labelcolor='b')

    # Line graph for cumulative percentage
    ax2 = ax1.twinx()
    ax2.plot(sorted_causes, cumulative_percentage, color='r', marker='o', linestyle='-', linewidth=2, markersize=8)
    ax2.set_ylabel('Cumulative Percentage', color='r')
    ax2.tick_params(axis='y', labelcolor='r')

    # Rotate x-axis labels for better readability
    plt.xticks(rotation=90)

    # Title and legend
    plt.title('Pareto Chart - Leading Causes of Death in 2006 (USA)')
    plt.legend(['Cumulative Percentage'], loc='upper left')

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


colab.research.google.com/drive/1NIcN-Yu0jZRdo7DJqik4HKbhBgq57101?authuser=0#scrollTo=miFeokuXth4L
Incognito
Error
Assignment Stat.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
S
+ Code + Text
RAM
Disk
[3]
Pareto Chart - Leading Causes of Death in 2006 (USA)
Cumulative Percentage
Number of Deaths (x10,000)
Cumulative Percentage
Cause of Death
[7] #Answer of 4
import numpy as np
0s completed at 5:14 PM
69°F
Sunny
Search
5:16
10/1/2023

Question 4


colab.research.google.com/drive/1NIcN-Yu0jZRdo7DJqik4HKbhBgq57101?authuser=0#scrollTo=zOxlTvsTt9px
Incognito
Error
Assignment Stat.ipynb
File Edit View Insert Runtime Tools Help
Comment
Share
S
+ Code + Text
RAM
Disk
#Answer of 4
import numpy as np
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, 30,
]
data_np = np.array(data)
median = np.median(data_np)
unique_values, counts = np.unique(data_np, return_counts=True)
modes = unique_values[counts == counts.max()]
q1 = np.percentile(data_np, 25)
q3 = np.percentile(data_np, 75)
0s completed at 5:17 PM
69°F
Sunny
Search
5:17
10/1/2023


colab.research.google.com/drive/1NIcN-Yu0jZRdo7DJqik4HKbhBgq57101?authuser=0#scrollTo=zOxlTvsTt9px
Incognito
Error
Assignment Stat.ipynb
File Edit View Insert Runtime Tools Help
Comment
Share
S
+ Code + Text
RAM
Disk
print(f"Median: {median}")
print(f"Mode(s): {', '.join(map(str, modes))}")
print(f"Q1: {q1}")
print(f"Q3: {q3}")
print(f"P10: {p10}")
print(f"P95: {p95}")
Median: 17.0
Mode(s): 16
Q1: 15.0
Q3: 20.75
P10: 14.0
P95: 39.599999999999966
0s completed at 5:17PM
69°F
Sunny
Search
5:17
10/1/2023