Midterm

- Name: Daniel Mehta
- Student Number: n01753264

Example1: (2 Mark)

Given a NumPy array arr, return an array containing the second column from all rows.

In [4]:
```
import numpy as np
```

In [6]:
```
arr = np.array([[11, 12, 13, 14],
                [21, 22, 23, 24],
                [31, 32, 33, 34]])
```

In [9]:
```
print(arr[:, 1])
```

```
[12 22 32]
```

Example2: (2 Mark)

- 1.Generate a random 4x6 matrix and compute the sum of all its elements.
- 2.Create a 3x3 identity matrix and replace all its diagonal elements with the square of their respective row indices.

In [13]:
```
#Task 1
matrix = np.random.rand(4,6)
matrix_sum = np.sum(matrix)
print("Random 4x6 matrix:\n", matrix)
print("Sum of all elements:", matrix_sum)
```

```
Random 4x6 matrix:
 [[0.16593039 0.12679387 0.60459173 0.47702262 0.735445   0.91850412]
 [0.7569896  0.6257445  0.44055711 0.90933904 0.76031862 0.87498443]
 [0.22237977 0.06524177 0.89814257 0.80183416 0.90707513 0.81925016]
 [0.71386474 0.65981673 0.76920525 0.75725548 0.38024607 0.37947976]]
Sum of all elements: 14.770012624542646
```

In [19]:
```
#Task 2
identity_matrix=np.eye(3)

for i in range(3):
    identity_matrix[i,i] = i **2

print("3x3 identity matrix:\n", identity_matrix)
```

```
3x3 identity matrix:
 [[0. 0. 0.]
 [0. 1. 0.]
 [0. 0. 4.]]
```

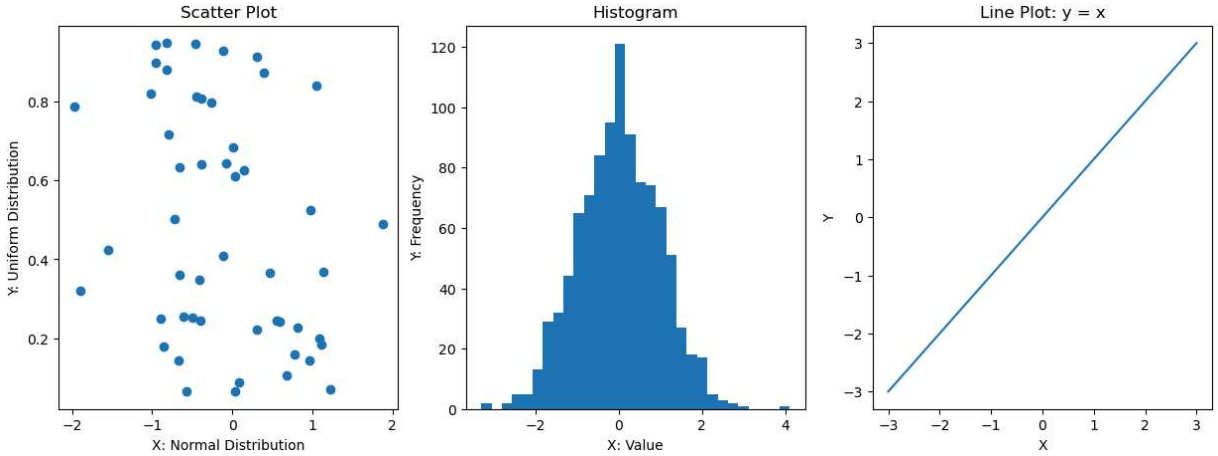
Example3:(3 Mark)

Create a figure with three subplots arranged horizontally. Plot the following in each subplot:

1. A scatter plot of 50 random points where x values are generated from a normal distribution (mean=0, std=1) and y values from a uniform distribution (range=[0, 1]).
2. A histogram of 1000 random numbers generated from a normal distribution.
3. A line plot of the function y=x over the interval [-3, 3].

In [21]:
```
import matplotlib.pyplot as plt
```

In [35]:
```
fig, axes = plt.subplots(1,3, figsize=(15,5))

#Task 1
x_scatter = np.random.normal(0,1,50)
y_scatter = np.random.uniform(0,1,50)
axes[0].scatter(x_scatter, y_scatter)
axes[0].set_title("Scatter Plot")
axes[0].set_xlabel("X: Normal Distribution")
axes[0].set_ylabel("Y: Uniform Distribution")

#Task 2
histogram_data = np.random.normal(0,1,1000)
axes[1].hist(histogram_data, bins=30)
axes[1].set_title("Histogram")
axes[1].set_xlabel("X: Value")
axes[1].set_ylabel("Y: Frequency")

#Task 3
x_line = np.linspace(-3,3,100)
y_line = x_line
axes[2].plot(x_line,y_line)
axes[2].set_title("Line Plot: y = x")
axes[2].set_xlabel("X")
axes[2].set_ylabel("Y")


plt.show()
```



Example4:(8 Mark)

Analyze the dataset and answer questions below.

Dataset Information

- gender: sex of students -> (Male/female)
- race/ethnicity: ethnicity of students -> (Group A, B, C, D, E)
- parental level of education: parents' final education ->(bachelor's degree, some college, master's degree, associate's degree)
- lunch: having lunch before test (standard or free/reduced)
- test preparation course: complete or not complete before test
- math score
- reading score
- writing score

1. Read a CSV dataset and create a DataFrame from it.
2. Check Missing values
3. Check Duplicates
4. Check data type
5. Calculate and display the correlation coefficient between math and reading score.and Plot a scatter plot to visualize the relationship between students' scores in Math and Reading.
6. Check various categories present in the different categorical column
7. Create a bar chart showing the average scores of students in each subject.
8. Is gender has any impact on student's performance?
9. Can we predict a student's performance in Reading based on their scores in Math and Writing?
10. What factors (such as parental level of education or test preparation) seem to influence student scores the most?

In [104…
```
import pandas as pd
import seaborn as sns
```

In [45]:
```
#task 1
df = pd.read_csv("StudentsPerformance.csv")
#df.head()
```

In [53]:
```
#Task 2
missing_vals = df.isnull().sum()
print(f"Missing Values in each column:\n{missing_vals}")
```

```
Missing Values in each column:
gender                         0
race/ethnicity                 0
parental level of education    0
lunch                          0
test preparation course        0
math score                     0
reading score                  0
writing score                  0
dtype: int64
```

In [61]:
```
#Task 3
duplicate = df.duplicated().sum()
print(f"Duplicated Rows Number: {duplicate}")
```

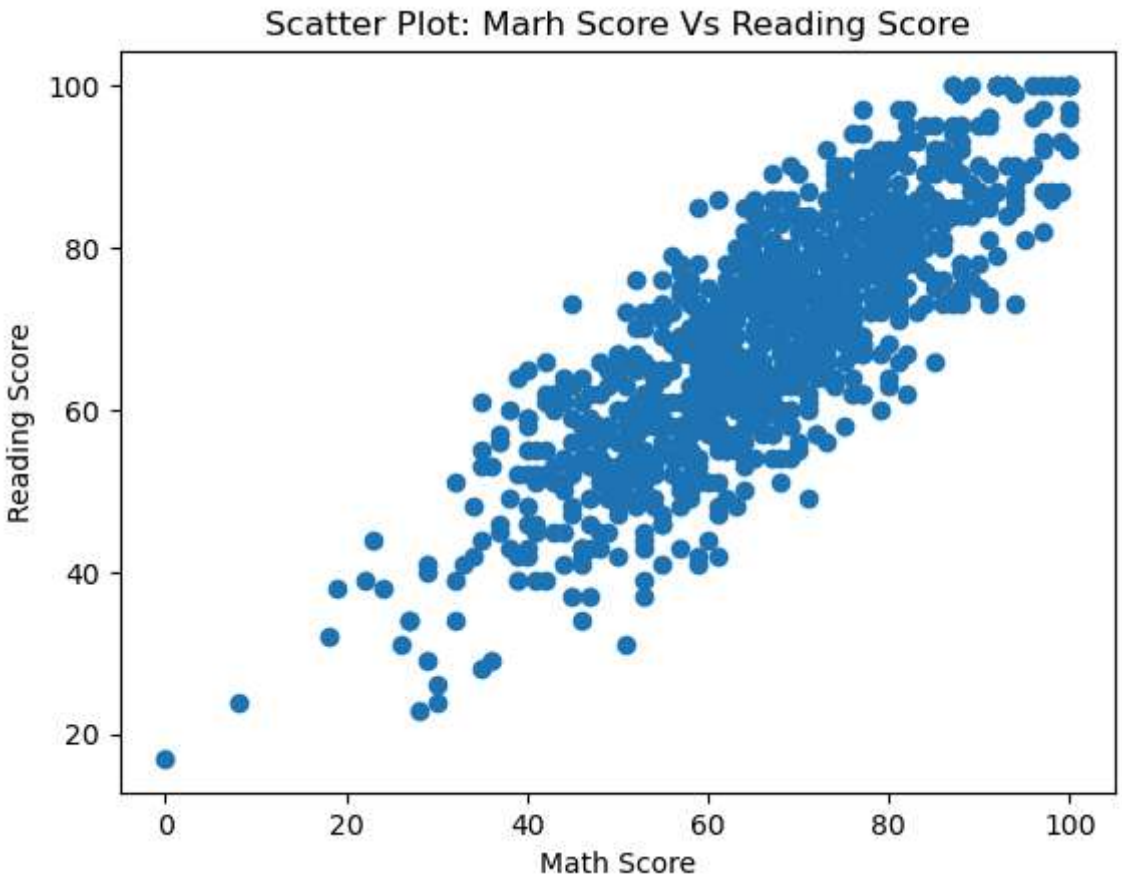
```
Duplicated Rows Number: 0
```

In [67]:
```
#Task 4
print(df.dtypes)
```

```
gender                         object
race/ethnicity                 object
parental level of education    object
lunch                          object
test preparation course        object
math score                      int64
reading score                   int64
writing score                   int64
dtype: object
```

In [73]:
```
#Task 5
correlation = df["math score"].corr(df["reading score"])
print(f"Correlation coefficient between Math and Reading Score: {correlation}")

plt.scatter(df["math score"], df["reading score"])
plt.xlabel("Math Score")
plt.ylabel("Reading Score")
plt.title("Scatter Plot: Marh Score Vs Reading Score")
plt.show()
```

```
Correlation coefficient between Math and Reading Score: 0.8175796636720539
```

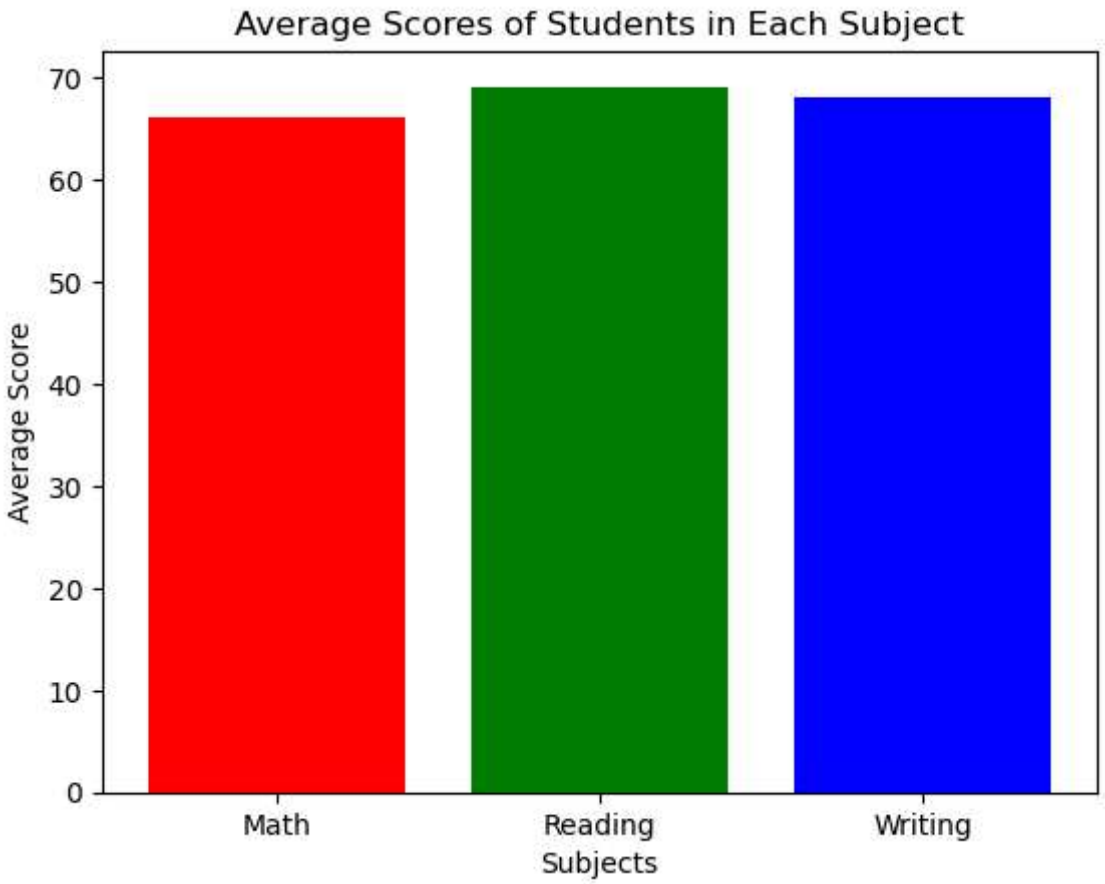


In [82]:
```
#Task 6
categorial_columns = ["gender", "race/ethnicity", "parental level of education", "l

for column in categorial_columns:
    print(f"Unique categories in {column}:")
    print(df[column].unique())
    print("-"*25)
```

```
Unique categories in gender:
['female' 'male']
-------------------------
Unique categories in race/ethnicity:
['group B' 'group C' 'group A' 'group D' 'group E']
-------------------------
Unique categories in parental level of education:
["bachelor's degree" 'some college' "master's degree" "associate's degree"
 'high school' 'some high school']
-------------------------
Unique categories in lunch:
['standard' 'free/reduced']
-------------------------
Unique categories in test preparation course:
['none' 'completed']
-------------------------
```

In [102…
```
#Task 7
math_avg = df["math score"].mean()
reading_avg = df["reading score"].mean()
writing_avg = df["writing score"].mean()
```

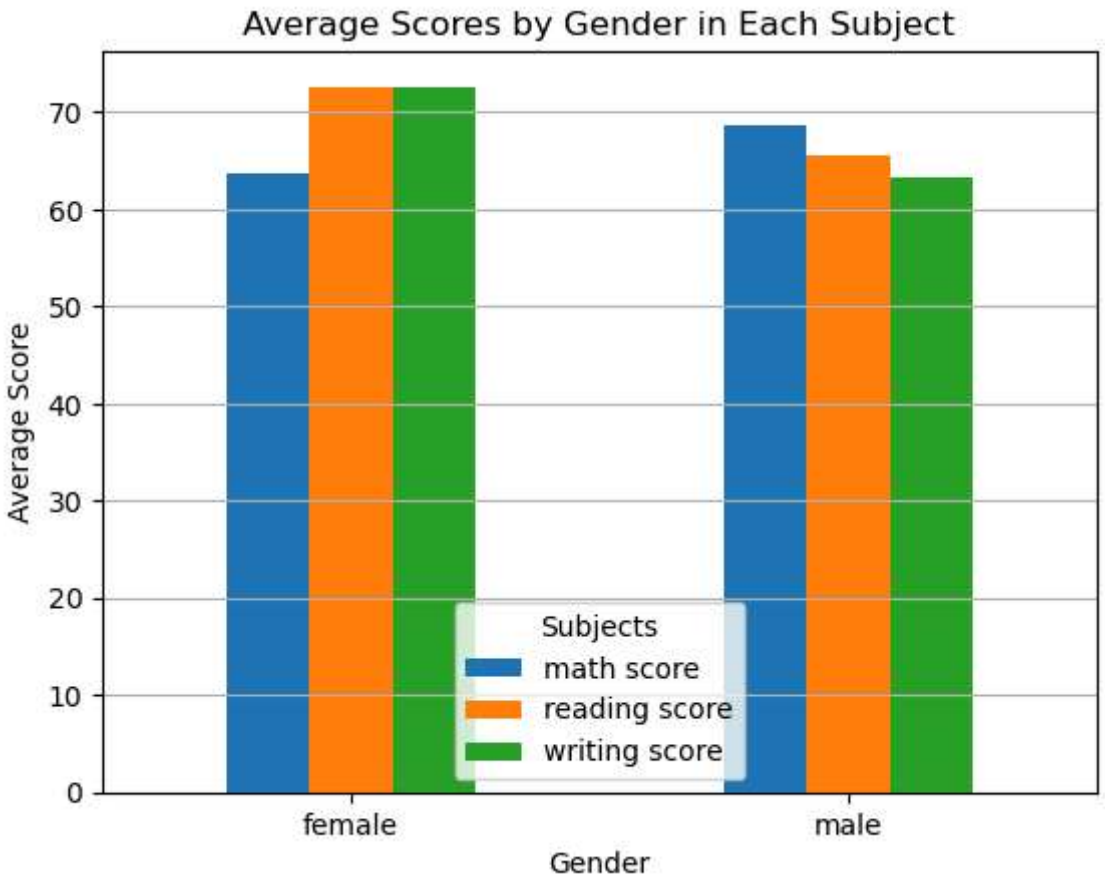
```
avg_scores = {"Math": math_avg, "Reading": reading_avg, "Writing": writing_avg}

plt.bar(avg_scores.keys(), avg_scores.values(), color=['r','g','b'])
plt.xlabel("Subjects")
plt.ylabel("Average Score")
plt.title("Average Scores of Students in Each Subject")
plt.show()
```



In [118…

```
#Task 8
gender_performance = df.groupby("gender")[["math score", "reading score", "writing

print(f"Average scores by gender:\n{gender_performance}")

gender_performance.plot(kind="bar")
plt.xlabel("Gender")
plt.ylabel("Average Score")
plt.title("Average Scores by Gender in Each Subject")

plt.legend(title="Subjects")
plt.xticks(rotation=0)
plt.grid(axis="y", linestyle="-")
plt.show()
```

```
Average scores by gender:
        math score  reading score  writing score
gender
female   63.633205      72.608108      72.467181
male     68.728216      65.473029      63.311203
```



Observations:

- Males score higher in Math on Average
- Females score higher in Reading and Writing on Average

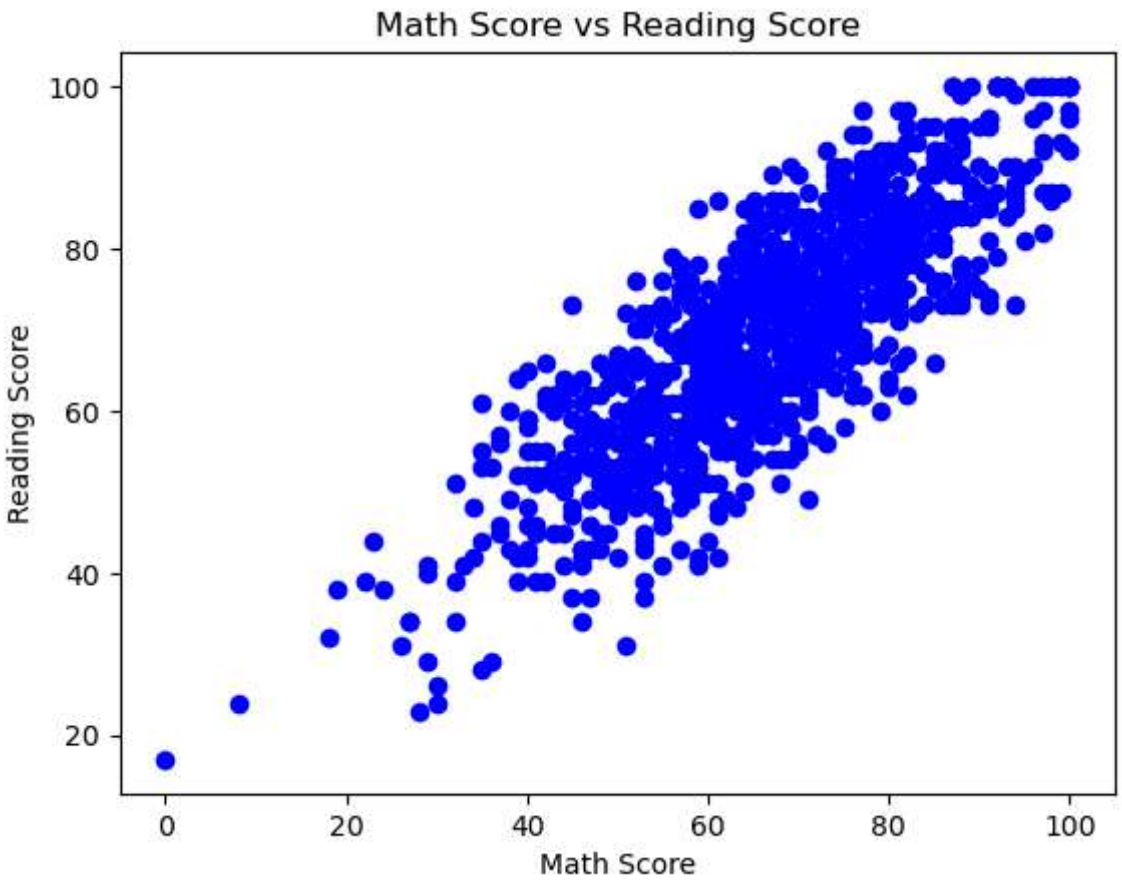
This suggests that gender has a impact on student performance, with males performing better in Math and females in Reading and Writing

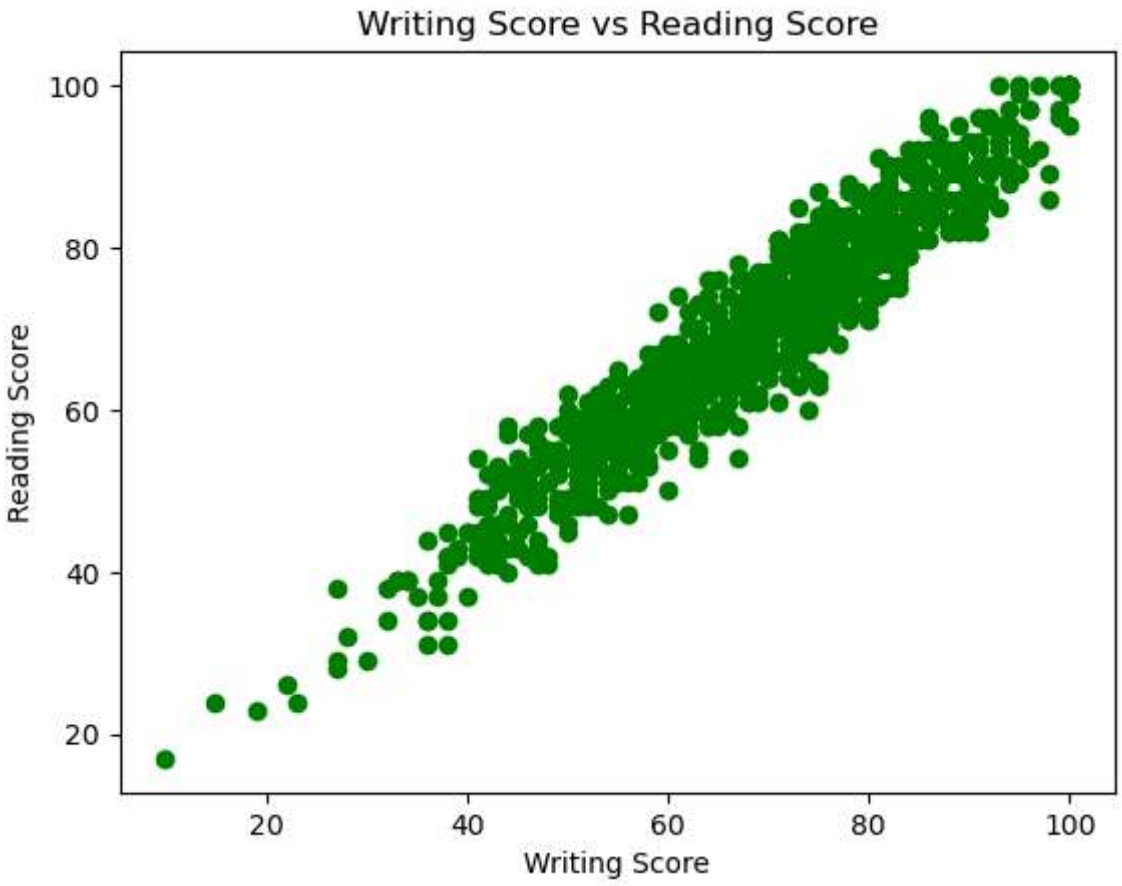
In [123…
```
#Task 9
correlation_math = df["math score"].corr(df["reading score"])
correlation_writing = df["writing score"].corr(df["reading score"])

print(f"Correlation between Math and Reading: {correlation_math}")
print(f"Correlation between Writing and Reading: {correlation_writing}")
```

```
Correlation between Math and Reading: 0.8175796636720539
Correlation between Writing and Reading: 0.954598077146248
```

In [129…
```
plt.scatter(df["math score"], df["reading score"], color='b')
plt.xlabel("Math Score")
plt.ylabel("Reading Score")
plt.title("Math Score vs Reading Score")
plt.show()

plt.scatter(df["writing score"], df["reading score"], color="g")
plt.xlabel("Writing Score")
plt.ylabel("Reading Score")
```

```
plt.title("Writing Score vs Reading Score")
plt.show()
```





Observations

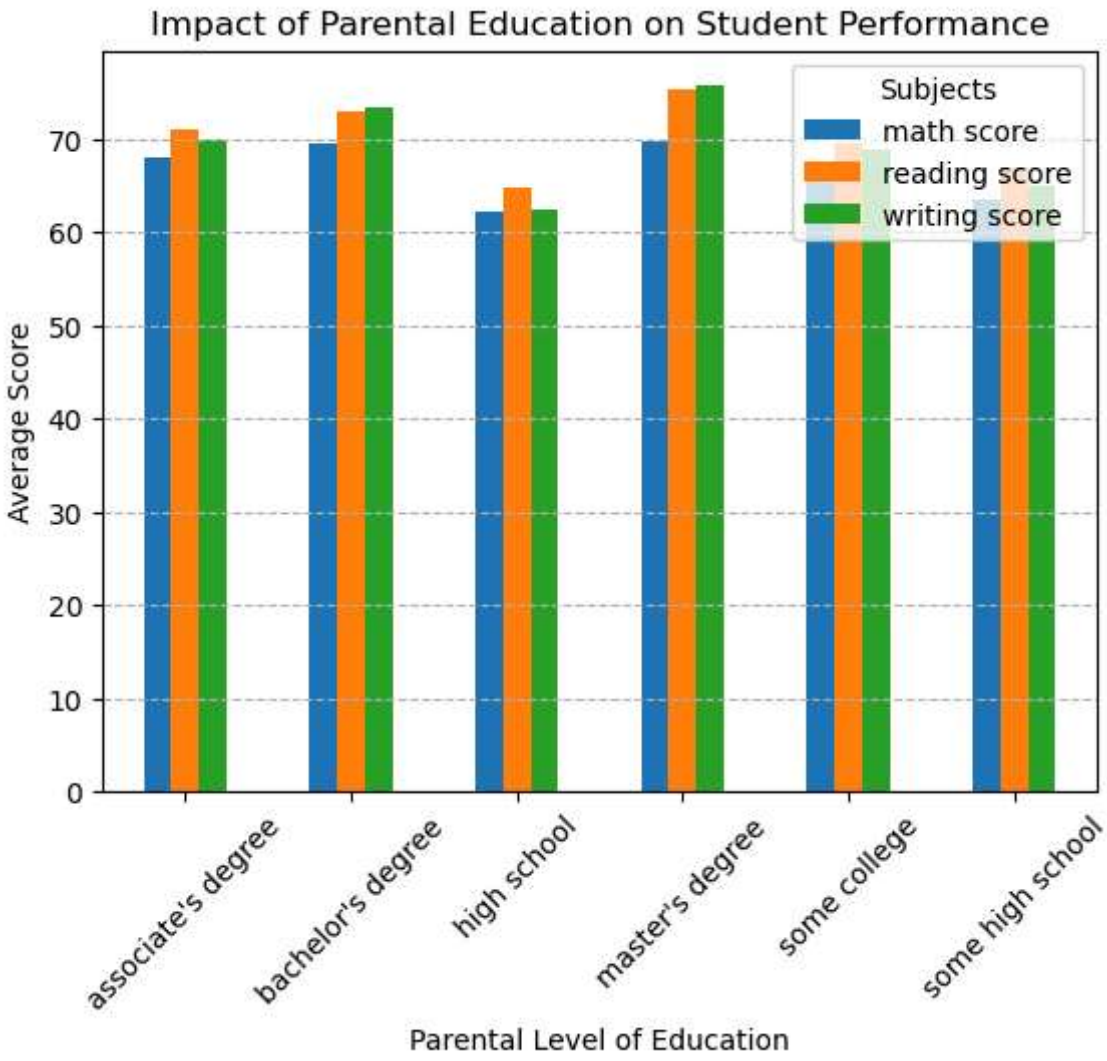
Therefore we can predict Reading scores using Math and Writing scores but writing scores are a much better predictor than math scores

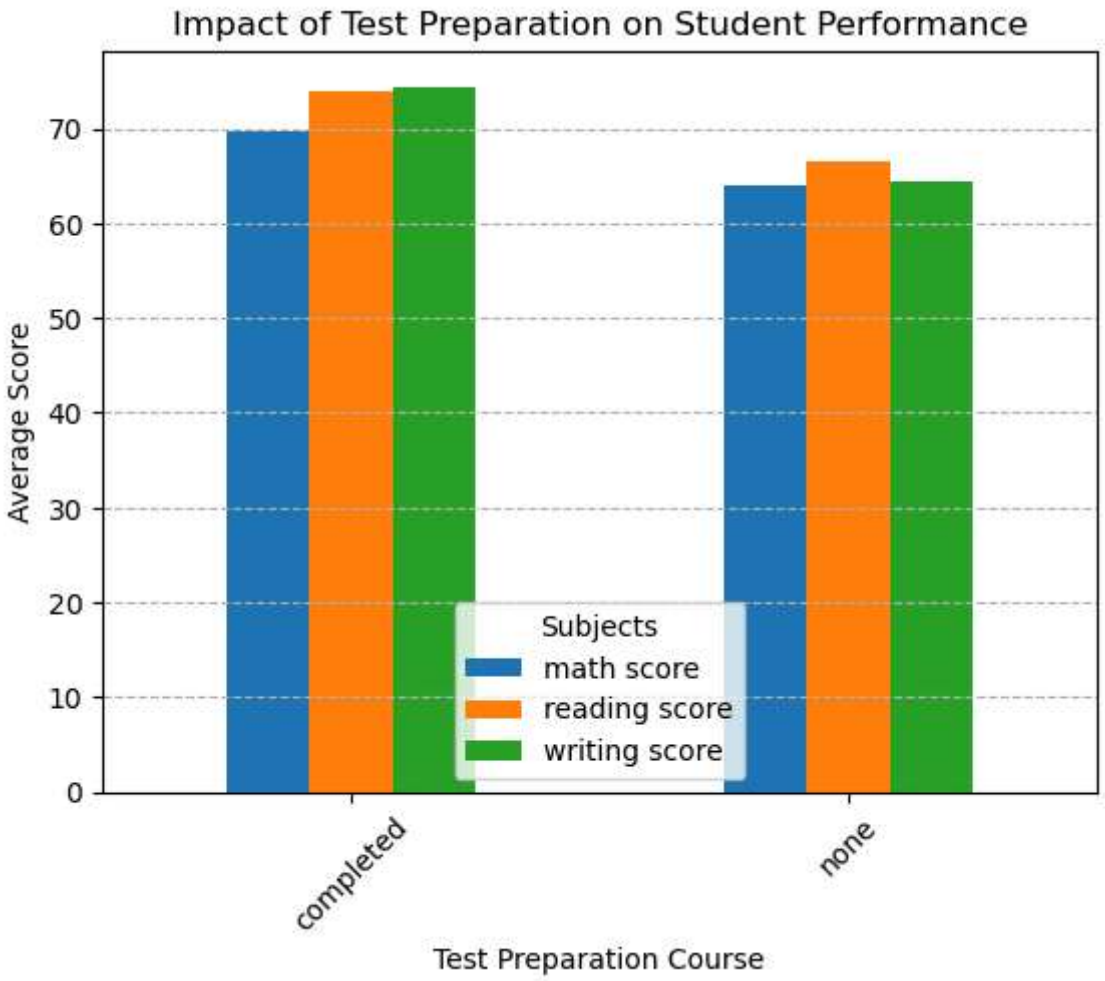
In [148…

```
# Task 10
parental_education_performance = df.groupby("parental level of education")[["math s
test_prep_performance = df.groupby("test preparation course")[["math score", "readi

print(f"Average scores based on parental level of education:\n {parental_education_
print(f"Average scores based on test preparation course:\n {test_prep_performance}"

parental_education_performance.plot(kind="bar")
plt.xlabel("Parental Level of Education")
plt.ylabel("Average Score")
plt.title("Impact of Parental Education on Student Performance")
plt.xticks(rotation=45)
plt.legend(title="Subjects")
plt.grid(axis="y", linestyle="--")
plt.show()

test_prep_performance.plot(kind="bar")
plt.xlabel("Test Preparation Course")
plt.ylabel("Average Score")
plt.title("Impact of Test Preparation on Student Performance")
plt.xticks(rotation=45)
plt.legend(title="Subjects")
```

```
plt.grid(axis="y", linestyle="--")
plt.show()
```

```
Average scores based on parental level of education:
                            math score  reading score  writing score
parental level of education
associate's degree           67.882883      70.927928      69.896396
bachelor's degree            69.389831      73.000000      73.381356
high school                  62.137755      64.704082      62.448980
master's degree              69.745763      75.372881      75.677966
some college                 67.128319      69.460177      68.840708
some high school             63.497207      66.938547      64.888268

Average scores based on test preparation course:
                          math score  reading score  writing score
test preparation course
completed                 69.695531      73.893855      74.418994
none                      64.077882      66.534268      64.504673
```





Observations

- **Parental Education Matters**: higher the education of the parents, the higher the test scores
- **Test Preperation Helps**: Students who complete the course perform better than those who dont

In [ ]: