```
# This Python 3 environment comes with many helpful analytics
libraries installed
# It is defined by the kaggle/python Docker image:
https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/"
directory
# For example, running this (by clicking run or pressing Shift+Enter)
will list all files under the input directory

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/)
that gets preserved as output when you create a version using "Save &
Run All"
# You can also write temporary files to /kaggle/temp/, but they won't
be saved outside of the current session
```

```
/kaggle/input/students-performance-in-exams/exams.csv
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Load the dataset
file_path = '/kaggle/input/students-performance-in-exams/exams.csv'
df = pd.read_csv(file_path)

# Display the first few rows of the dataset
df.head()
```

```
   gender race/ethnicity parental level of education         lunch  \
0    male        group A                 high school      standard   
1  female        group D            some high school  free/reduced   
2    male        group E                some college  free/reduced   
3    male        group B                 high school      standard   
4    male        group E          associate's degree      standard   

  test preparation course  math score  reading score  writing score  
0               completed          67             67             63  
1                    none          40             59             55  
2                    none          59             60             50  
```

```
3                    none          77             78             68
4               completed          78             73             68
```

```
df.shape
```

```
(1000, 8)
```

```
df.isnull().sum()
```

```
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
```

```
df.describe()
```

```
        math score  reading score  writing score
count  1000.000000    1000.000000    1000.000000
mean     66.396000      69.002000      67.738000
std      15.402871      14.737272      15.600985
min      13.000000      27.000000      23.000000
25%      56.000000      60.000000      58.000000
50%      66.500000      70.000000      68.000000
75%      77.000000      79.000000      79.000000
max     100.000000     100.000000     100.000000
```

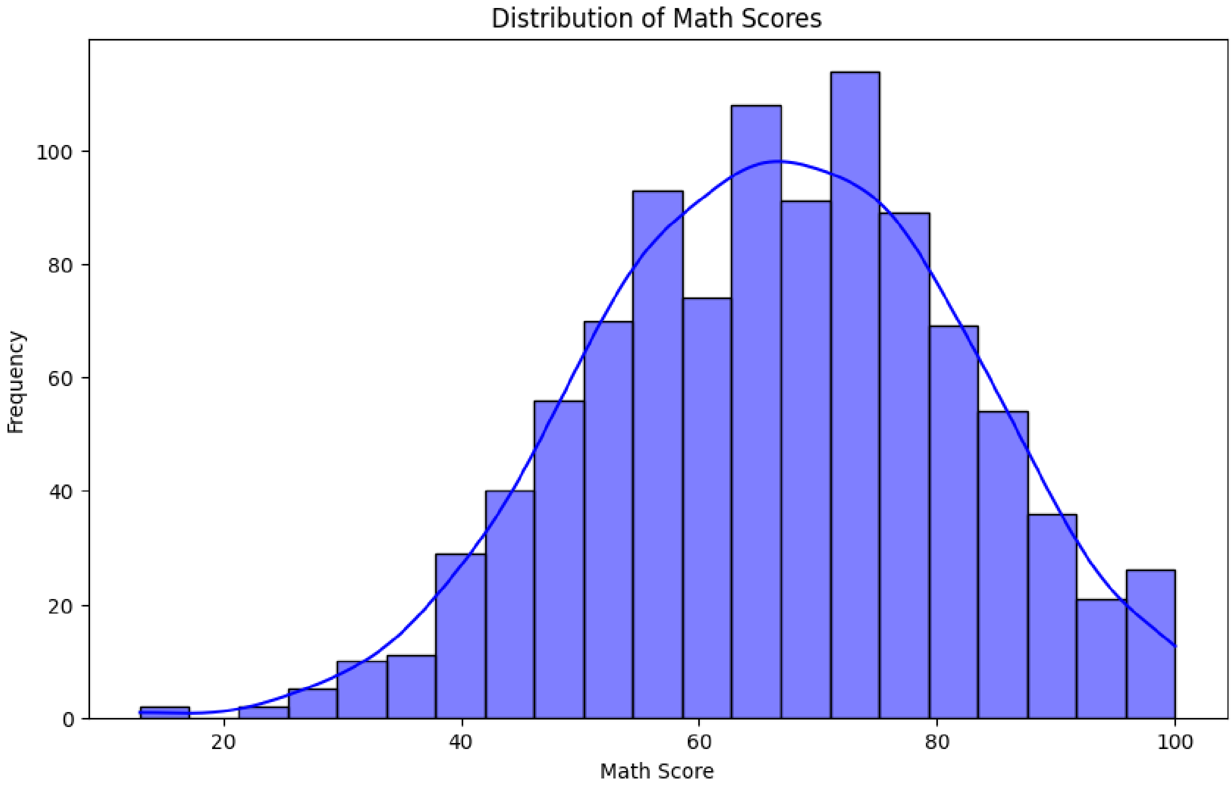
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 8 columns):
 #   Column                       Non-Null Count  Dtype 
---  ------                       --------------  ----- 
 0   gender                       1000 non-null   object
 1   race/ethnicity               1000 non-null   object
 2   parental level of education  1000 non-null   object
 3   lunch                        1000 non-null   object
 4   test preparation course      1000 non-null   object
 5   math score                   1000 non-null   int64 
 6   reading score                1000 non-null   int64 
 7   writing score                1000 non-null   int64 
dtypes: int64(3), object(5)
memory usage: 62.6+ KB
```

```
plt.figure(figsize=(10, 6))
sns.histplot(df['math score'], kde=True, color='blue')
```

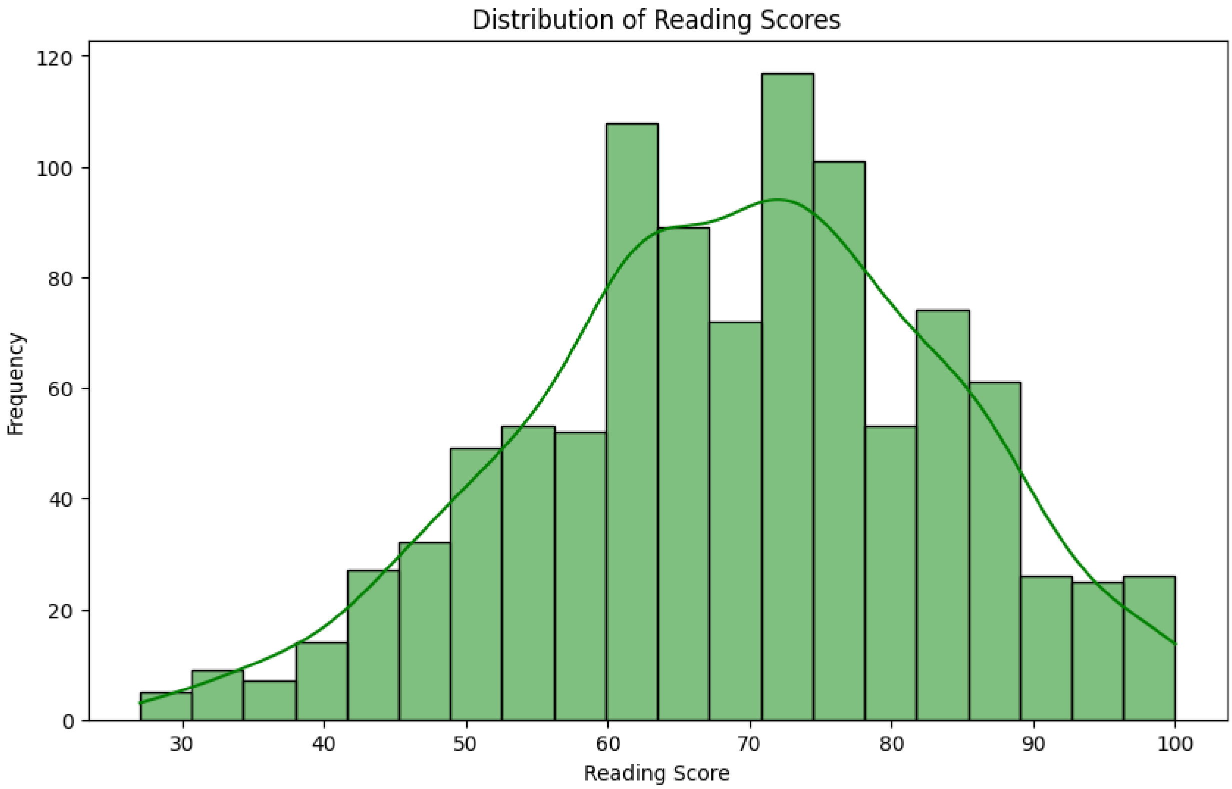
```python
plt.title('Distribution of Math Scores')
plt.xlabel('Math Score')
plt.ylabel('Frequency')
plt.show()
```

```
/opt/conda/lib/python3.10/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



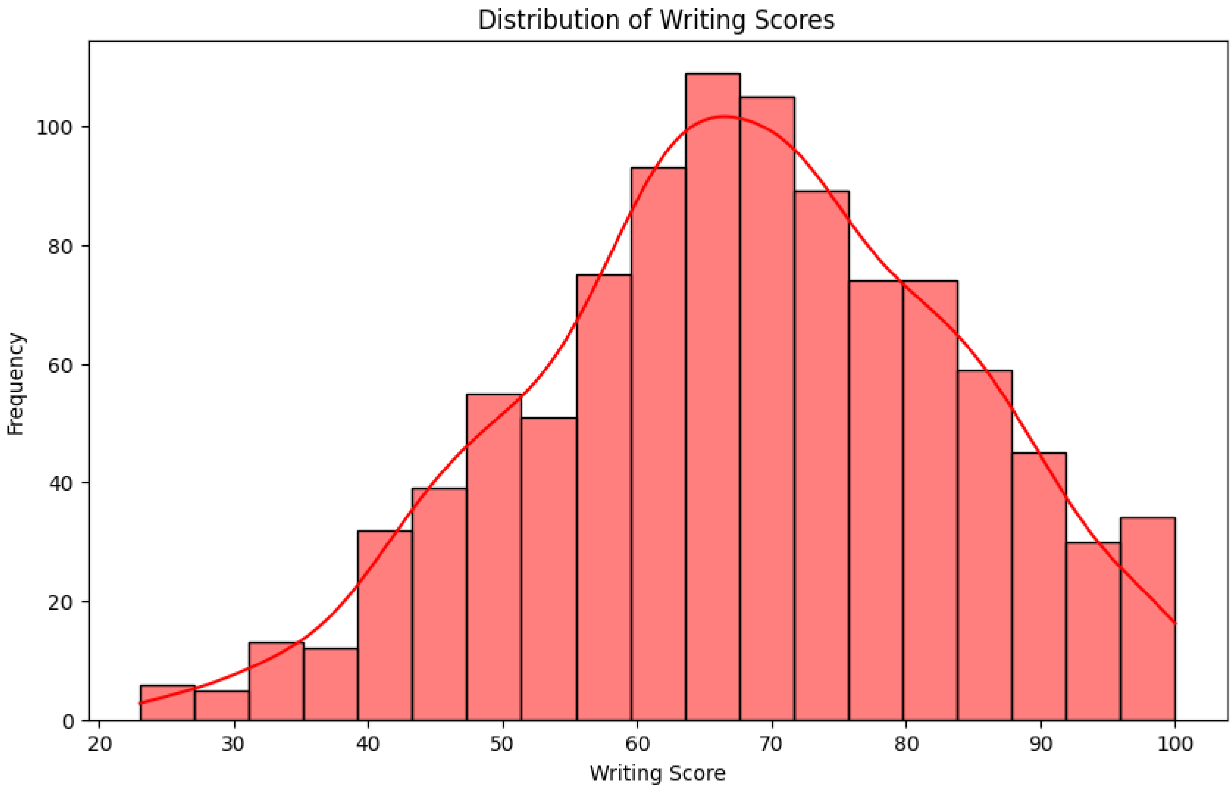
```python
plt.figure(figsize=(10, 6))
sns.histplot(df['reading score'], kde=True, color='green')
plt.title('Distribution of Reading Scores')
plt.xlabel('Reading Score')
plt.ylabel('Frequency')
plt.show()
```

```
/opt/conda/lib/python3.10/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

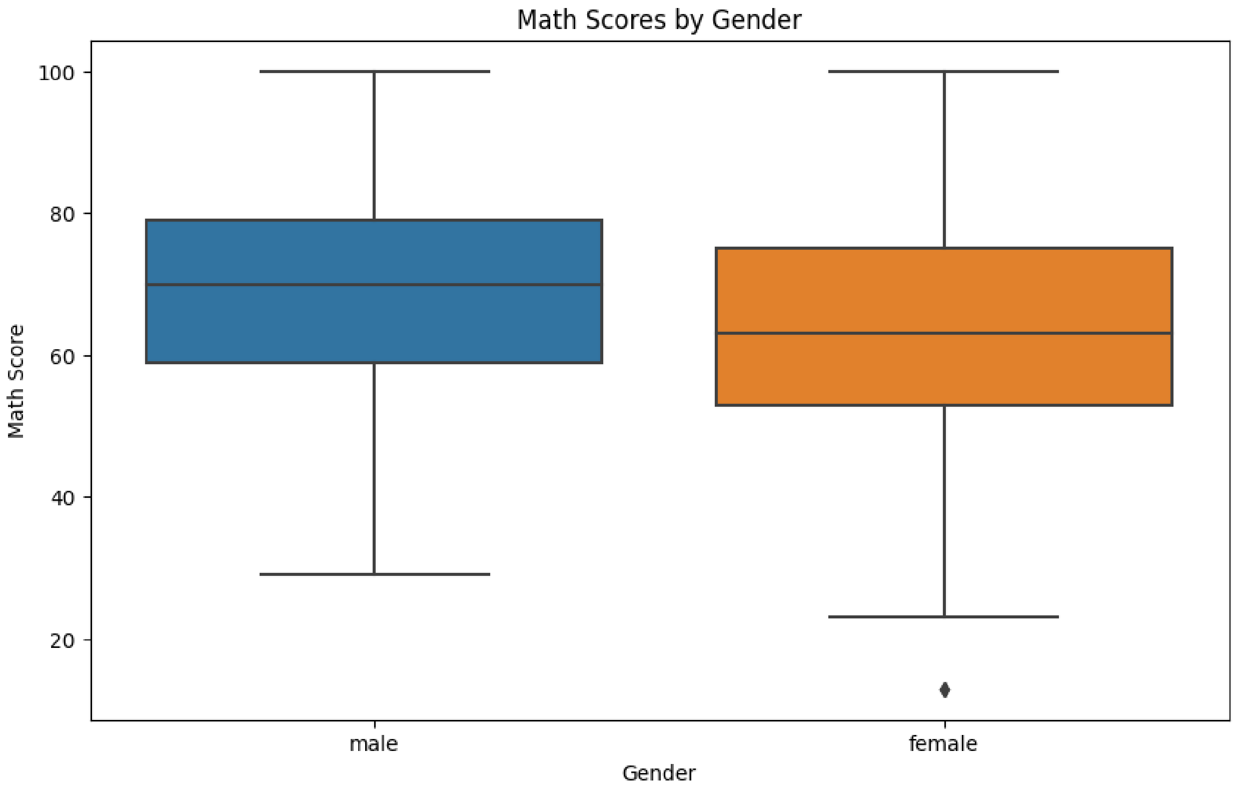


```
plt.figure(figsize=(10, 6))
sns.histplot(df['writing score'], kde=True, color='red')
plt.title('Distribution of Writing Scores')
plt.xlabel('Writing Score')
plt.ylabel('Frequency')
plt.show()
```

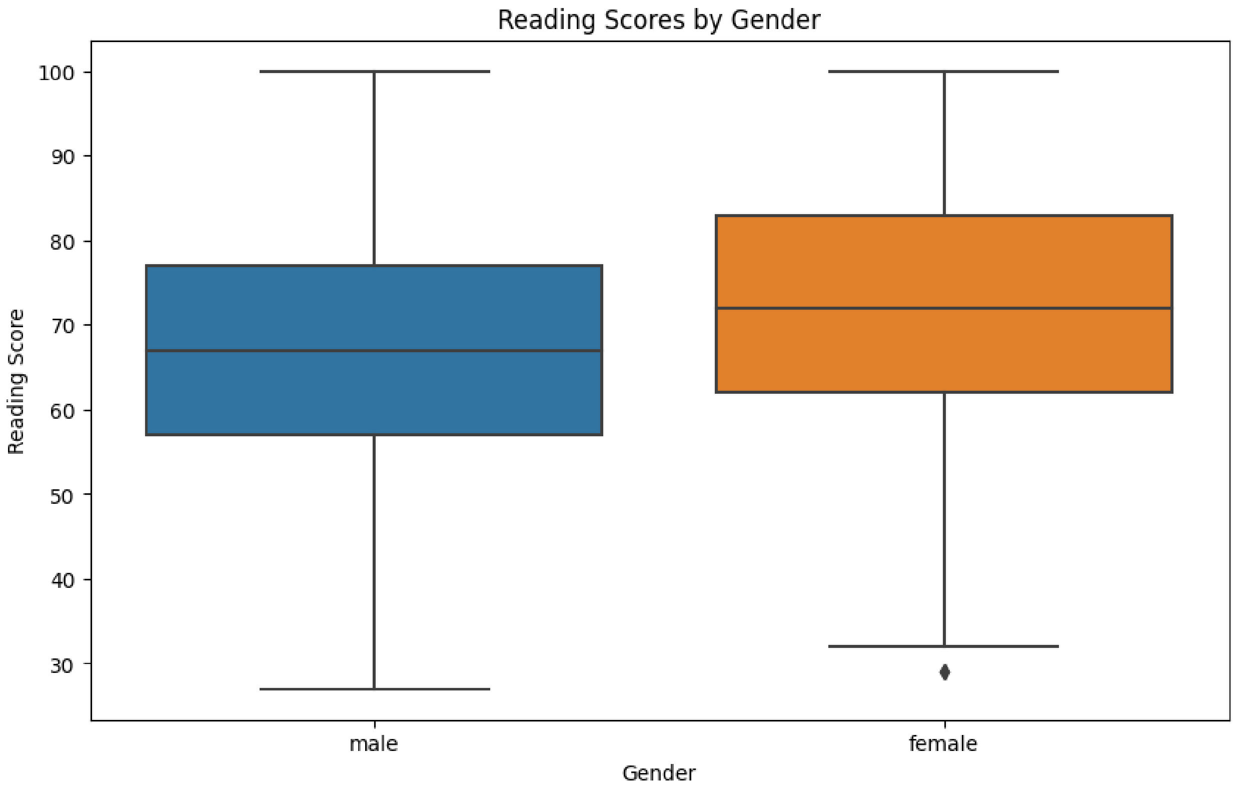
```
/opt/conda/lib/python3.10/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



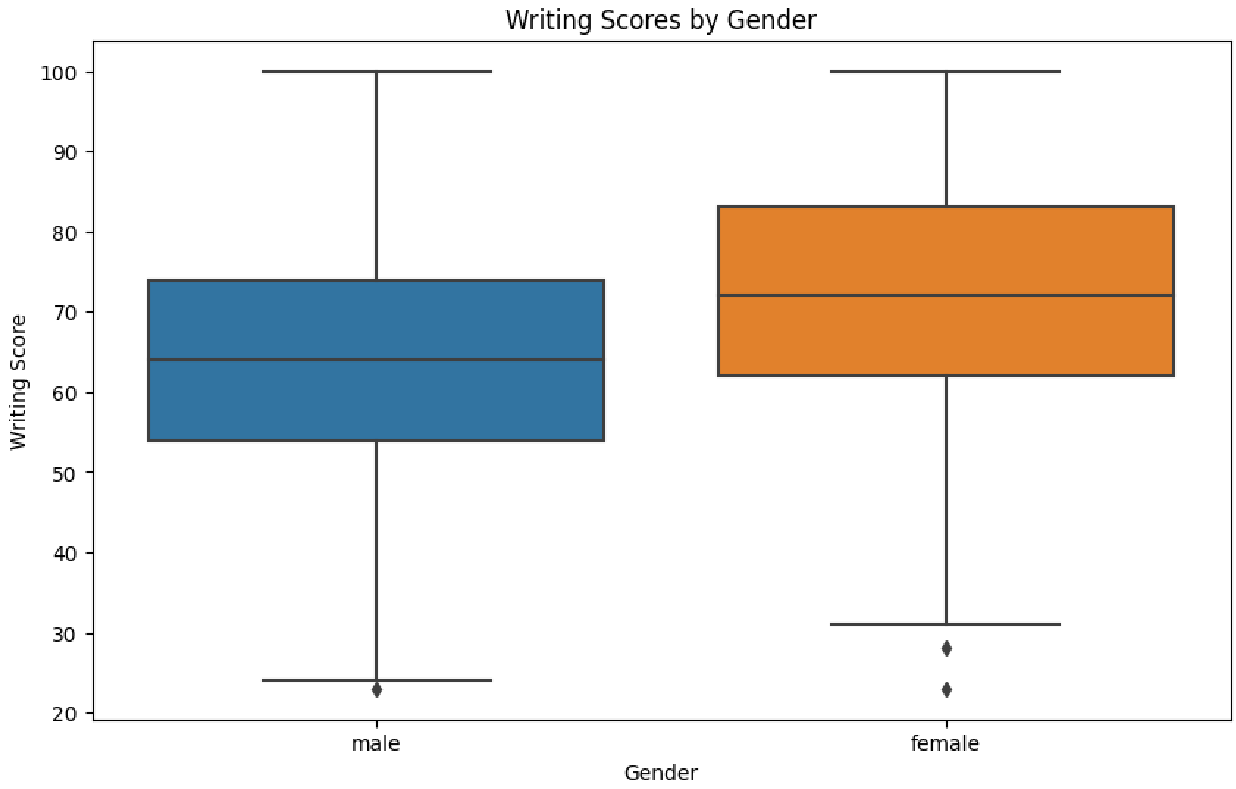
```
plt.figure(figsize=(10, 6))
sns.boxplot(x='gender', y='math score', data=df)
plt.title('Math Scores by Gender')
plt.xlabel('Gender')
plt.ylabel('Math Score')
plt.show()
```



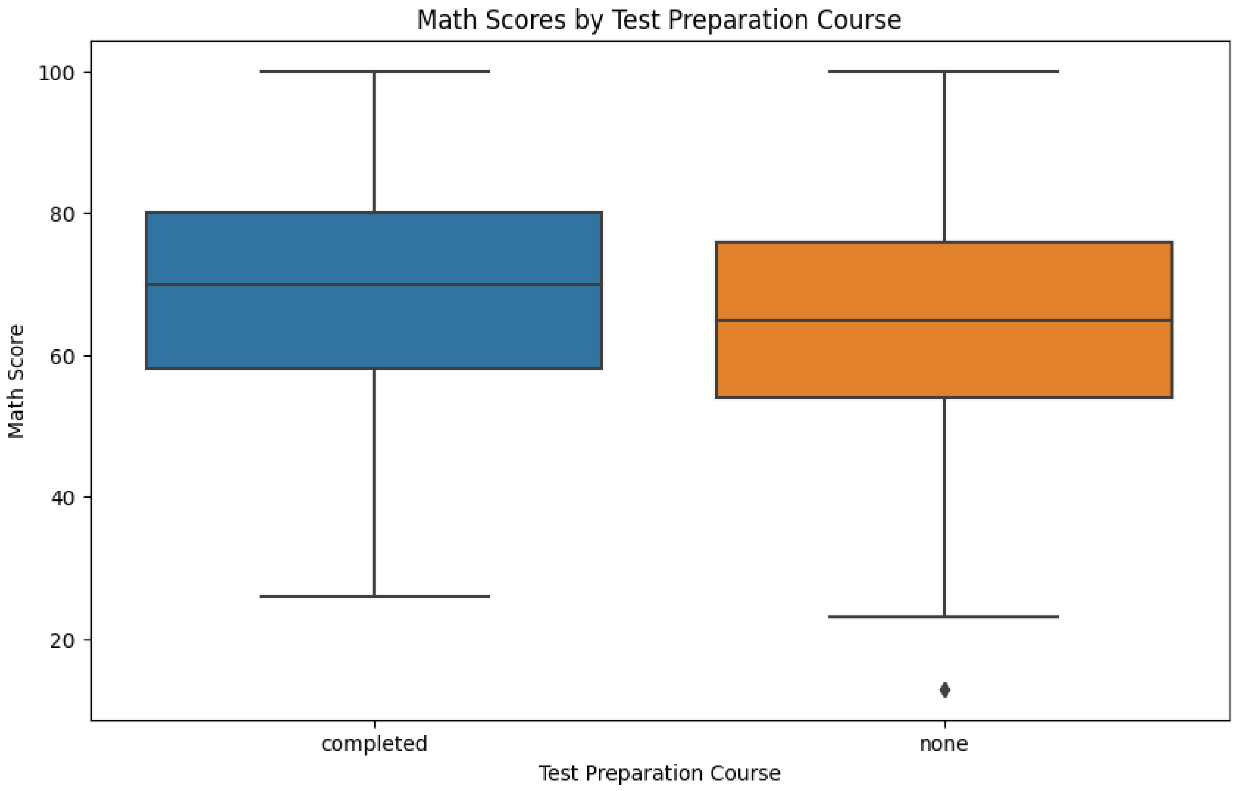
```
plt.figure(figsize=(10, 6))
sns.boxplot(x='gender', y='reading score', data=df)
plt.title('Reading Scores by Gender')
plt.xlabel('Gender')
plt.ylabel('Reading Score')
plt.show()
```



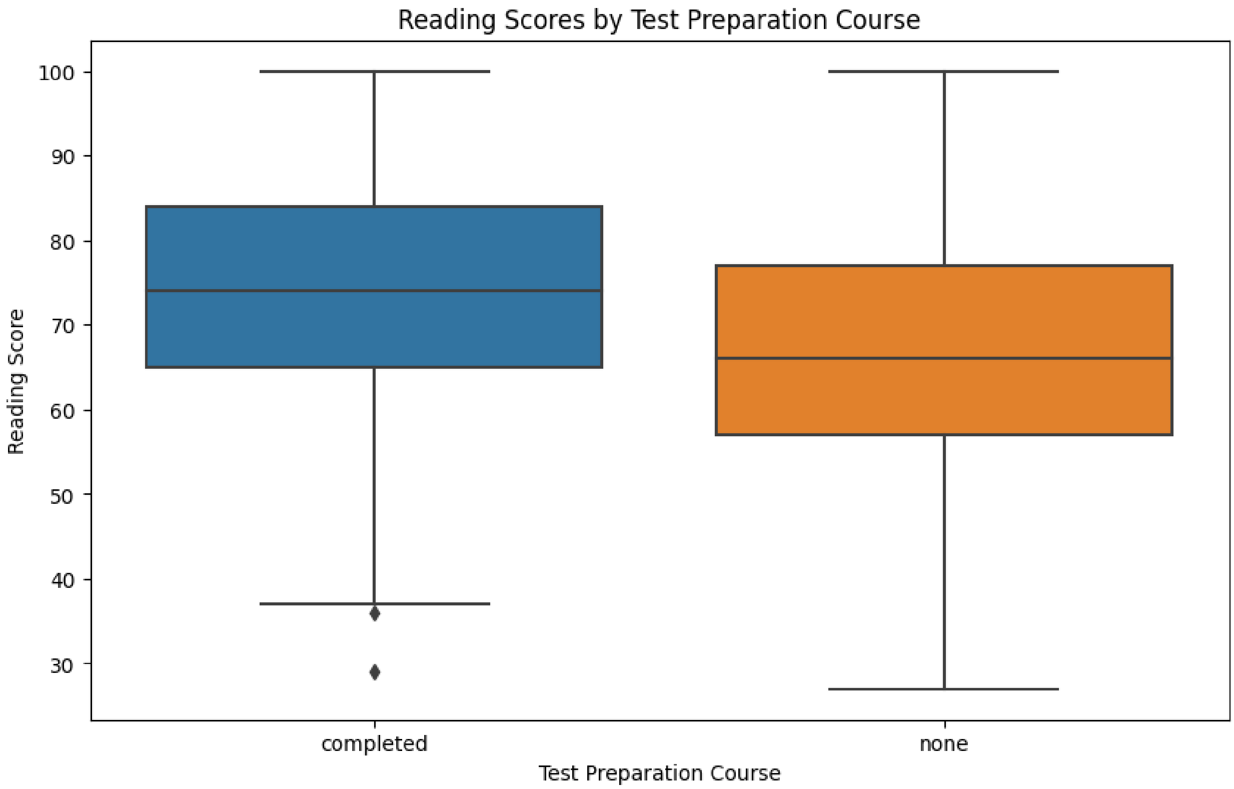
```
plt.figure(figsize=(10, 6))
sns.boxplot(x='gender', y='writing score', data=df)
plt.title('Writing Scores by Gender')
plt.xlabel('Gender')
plt.ylabel('Writing Score')
plt.show()
```



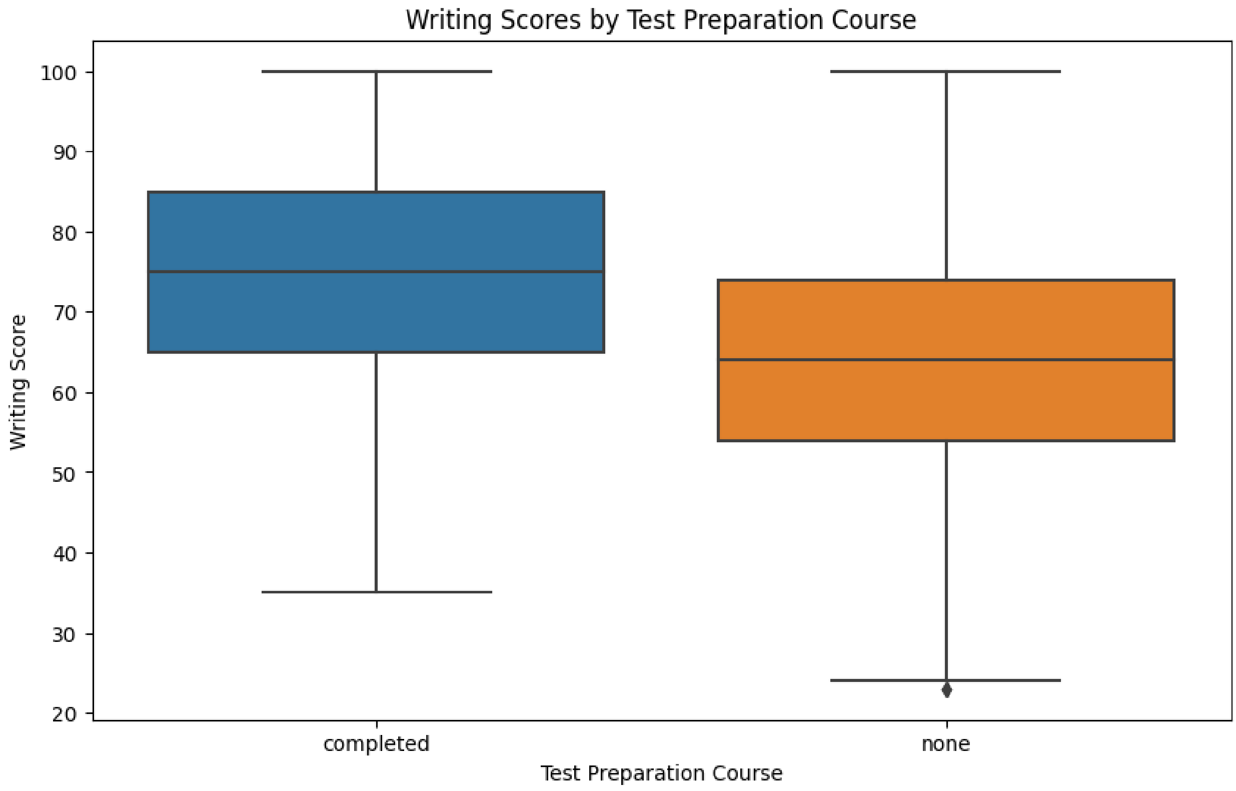
```
plt.figure(figsize=(10, 6))
sns.boxplot(x='test preparation course', y='math score', data=df)
plt.title('Math Scores by Test Preparation Course')
plt.xlabel('Test Preparation Course')
plt.ylabel('Math Score')
plt.show()
```



```
plt.figure(figsize=(10, 6))
sns.boxplot(x='test preparation course', y='reading score', data=df)
plt.title('Reading Scores by Test Preparation Course')
plt.xlabel('Test Preparation Course')
plt.ylabel('Reading Score')
plt.show()
```

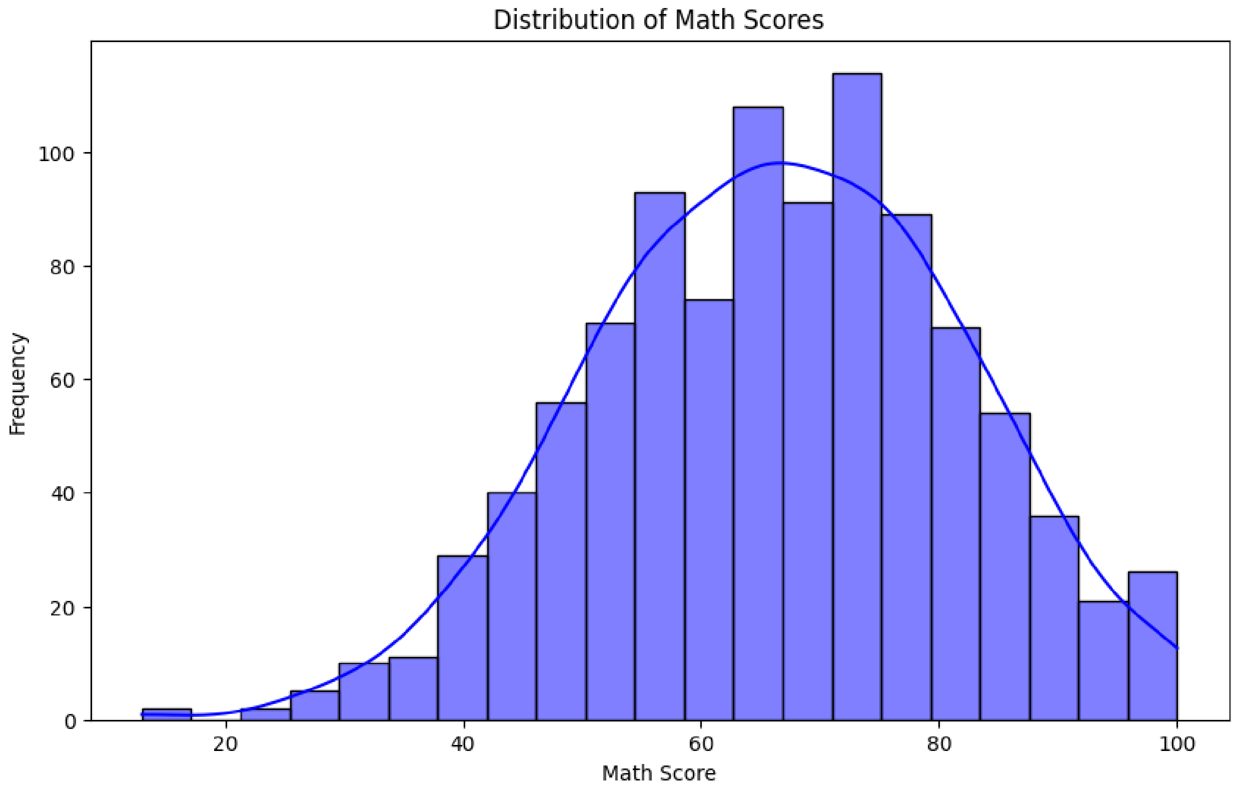


```
plt.figure(figsize=(10, 6))
sns.boxplot(x='test preparation course', y='writing score', data=df)
plt.title('Writing Scores by Test Preparation Course')
plt.xlabel('Test Preparation Course')
plt.ylabel('Writing Score')
plt.show()
```



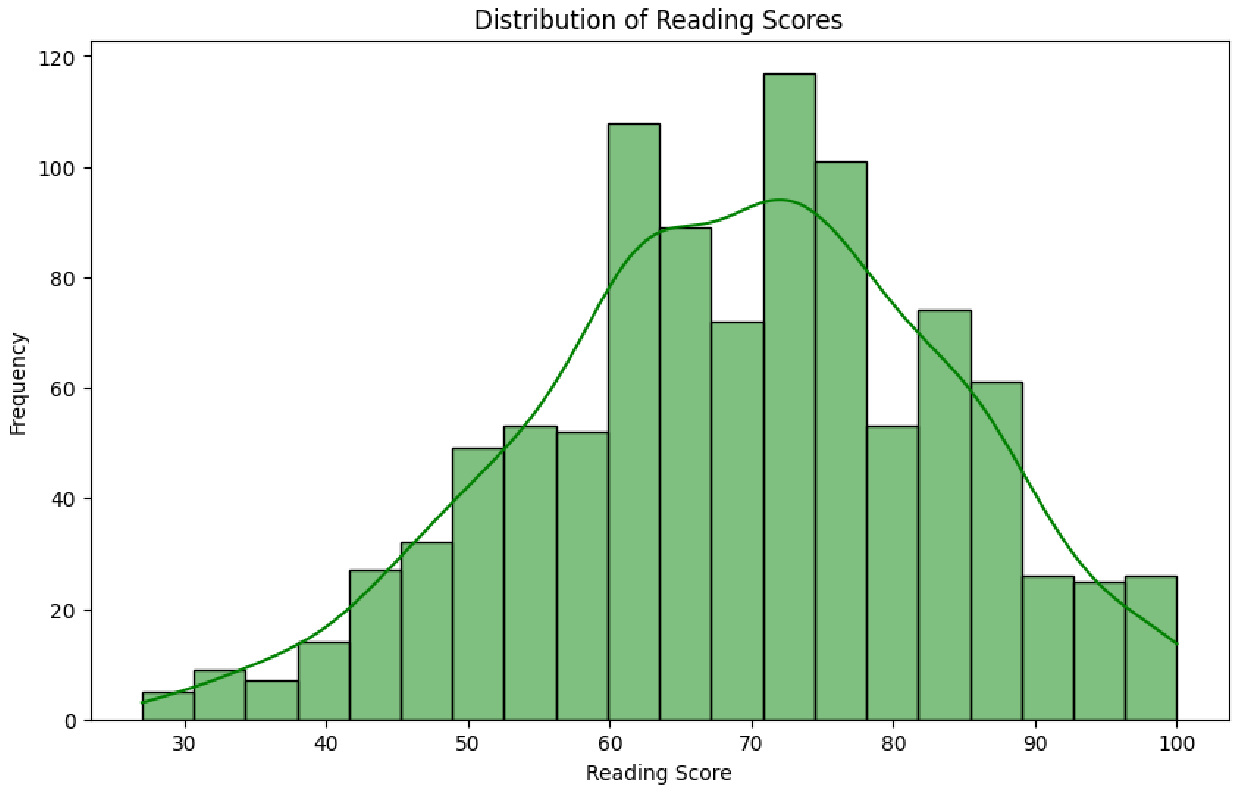
```
plt.figure(figsize=(10, 6))
sns.histplot(df['math score'], kde=True, color='blue')
plt.title('Distribution of Math Scores')
plt.xlabel('Math Score')
plt.ylabel('Frequency')
plt.show()
```

```
/opt/conda/lib/python3.10/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

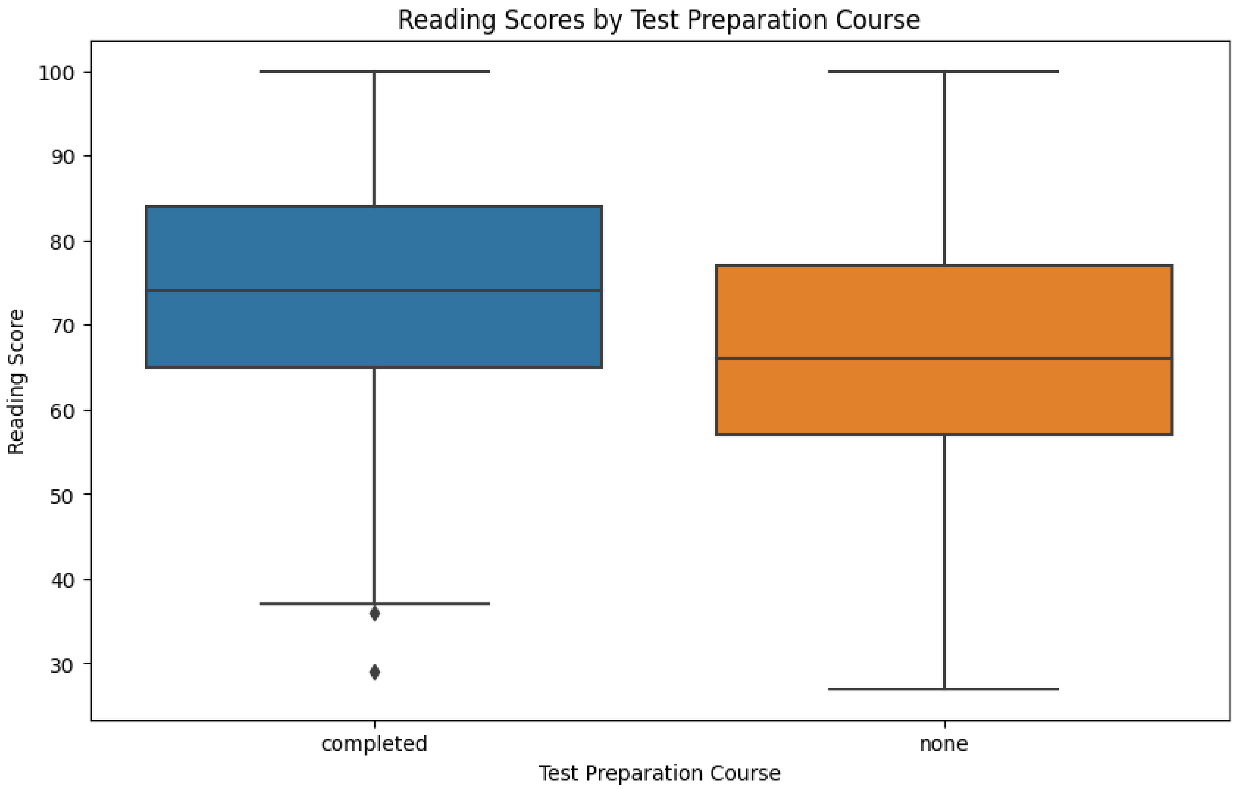


```
plt.figure(figsize=(10, 6))
sns.histplot(df['reading score'], kde=True, color='green')
plt.title('Distribution of Reading Scores')
plt.xlabel('Reading Score')
plt.ylabel('Frequency')
plt.show()
```

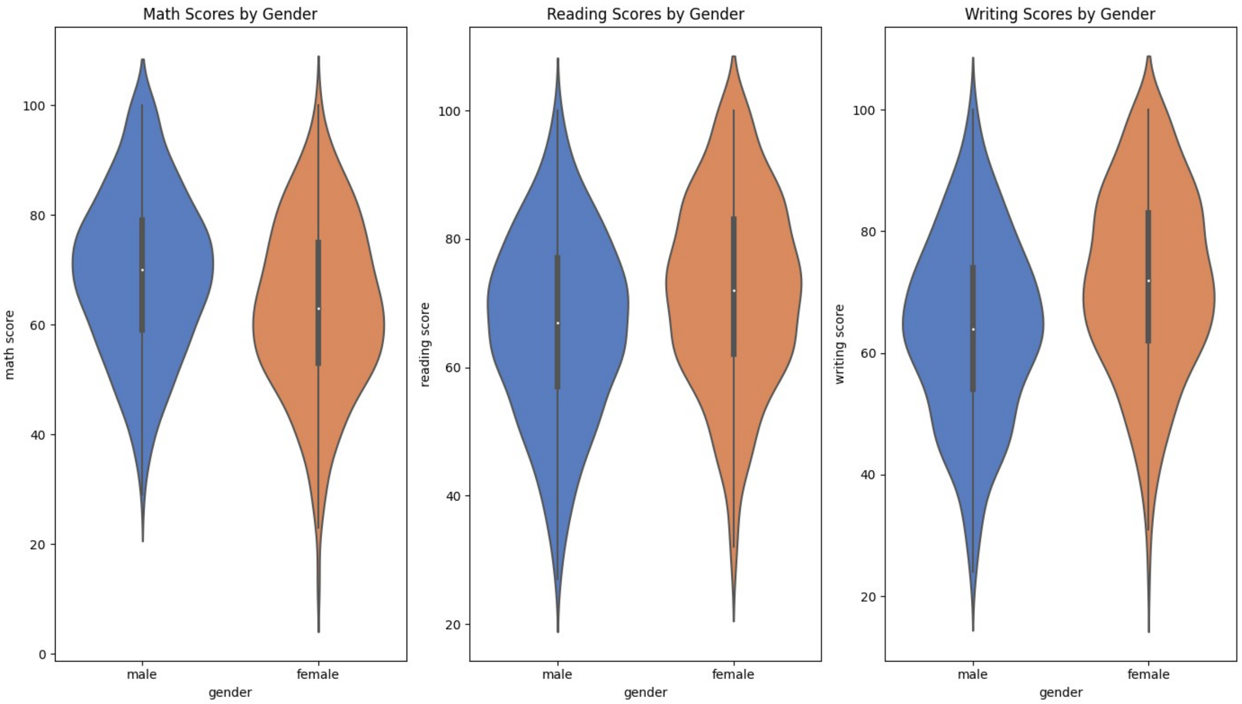
```
/opt/conda/lib/python3.10/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



```
plt.figure(figsize=(10, 6))
sns.boxplot(x='test preparation course', y='reading score', data=df)
plt.title('Reading Scores by Test Preparation Course')
plt.xlabel('Test Preparation Course')
plt.ylabel('Reading Score')
plt.show()
```

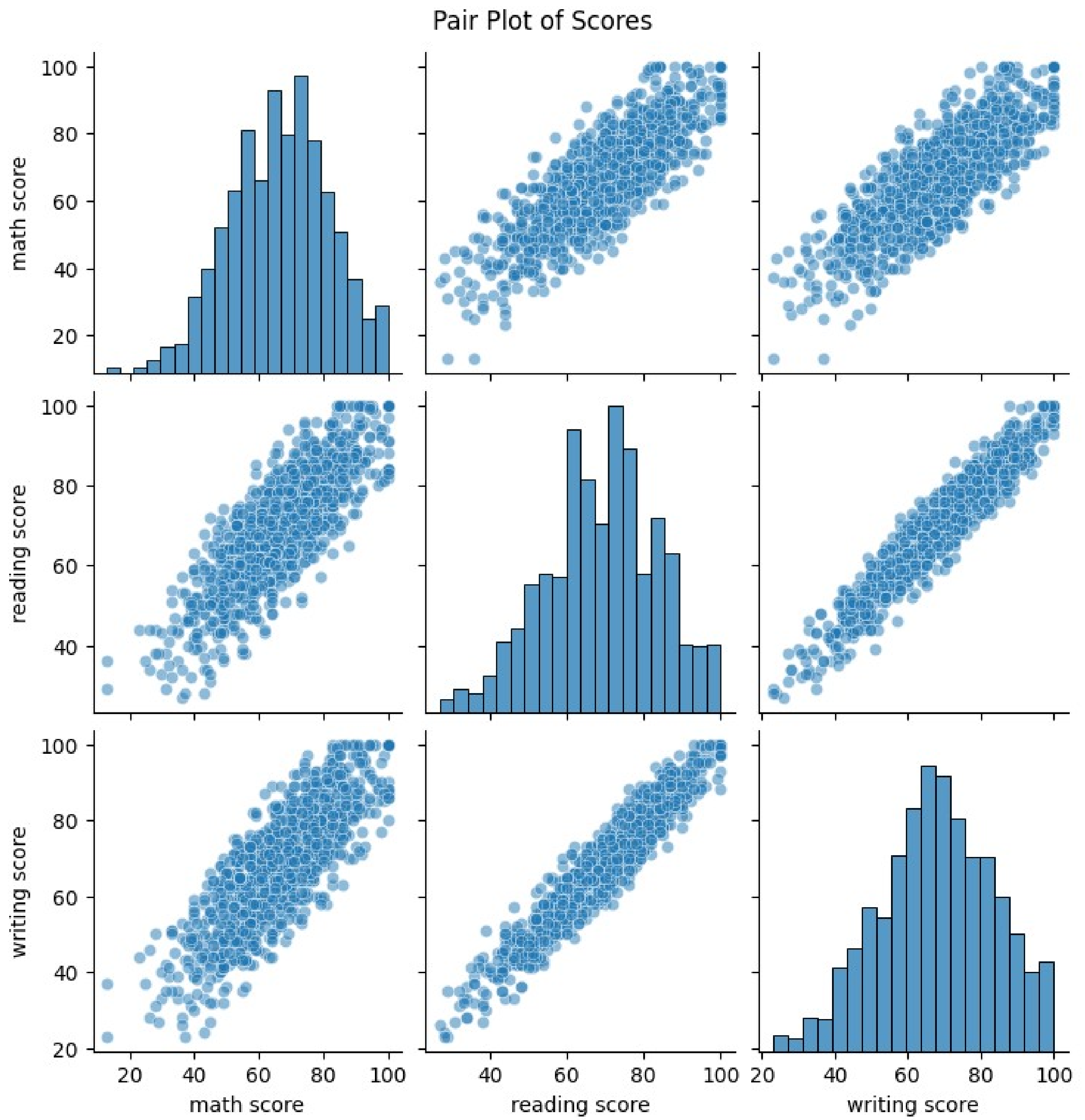


```
plt.figure(figsize=(14, 8))

# Math Score Violin Plot
plt.subplot(1, 3, 1)
sns.violinplot(x='gender', y='math score', data=df, palette='muted')
plt.title('Math Scores by Gender')

# Reading Score Violin Plot
plt.subplot(1, 3, 2)
sns.violinplot(x='gender', y='reading score', data=df,
palette='muted')
plt.title('Reading Scores by Gender')

# Writing Score Violin Plot
plt.subplot(1, 3, 3)
sns.violinplot(x='gender', y='writing score', data=df,
palette='muted')
plt.title('Writing Scores by Gender')

plt.tight_layout()
plt.show()
```

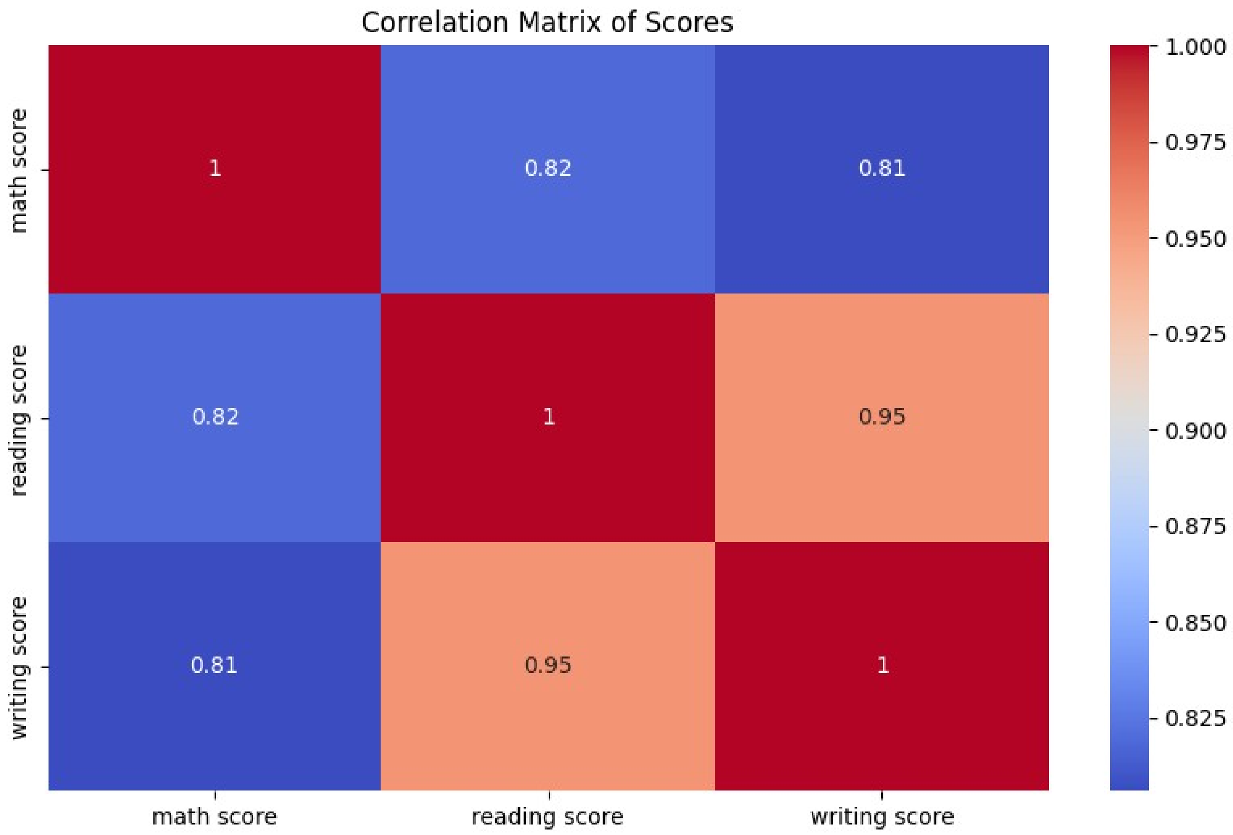


```
sns.pairplot(df[['math score', 'reading score', 'writing score']],
kind='scatter', plot_kws={'alpha':0.5})
plt.suptitle('Pair Plot of Scores', y=1.02)
plt.show()
```

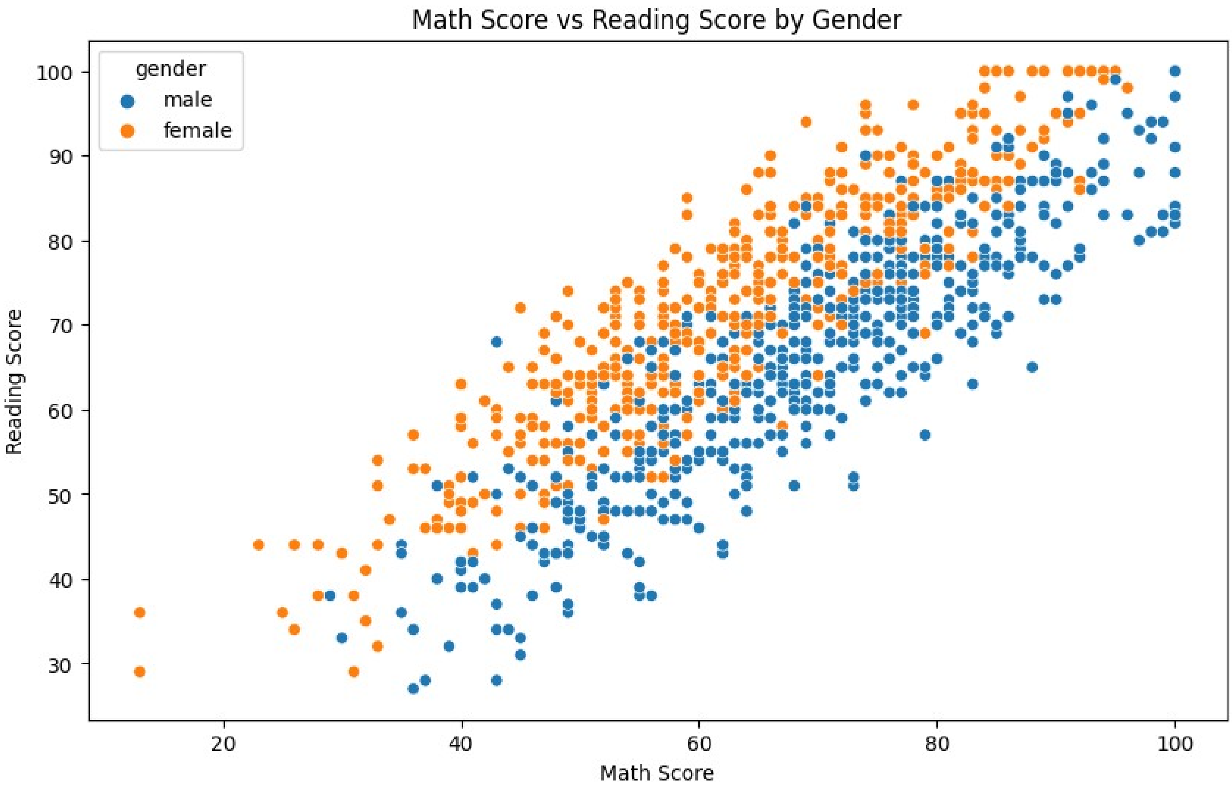
```
/opt/conda/lib/python3.10/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/conda/lib/python3.10/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/conda/lib/python3.10/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



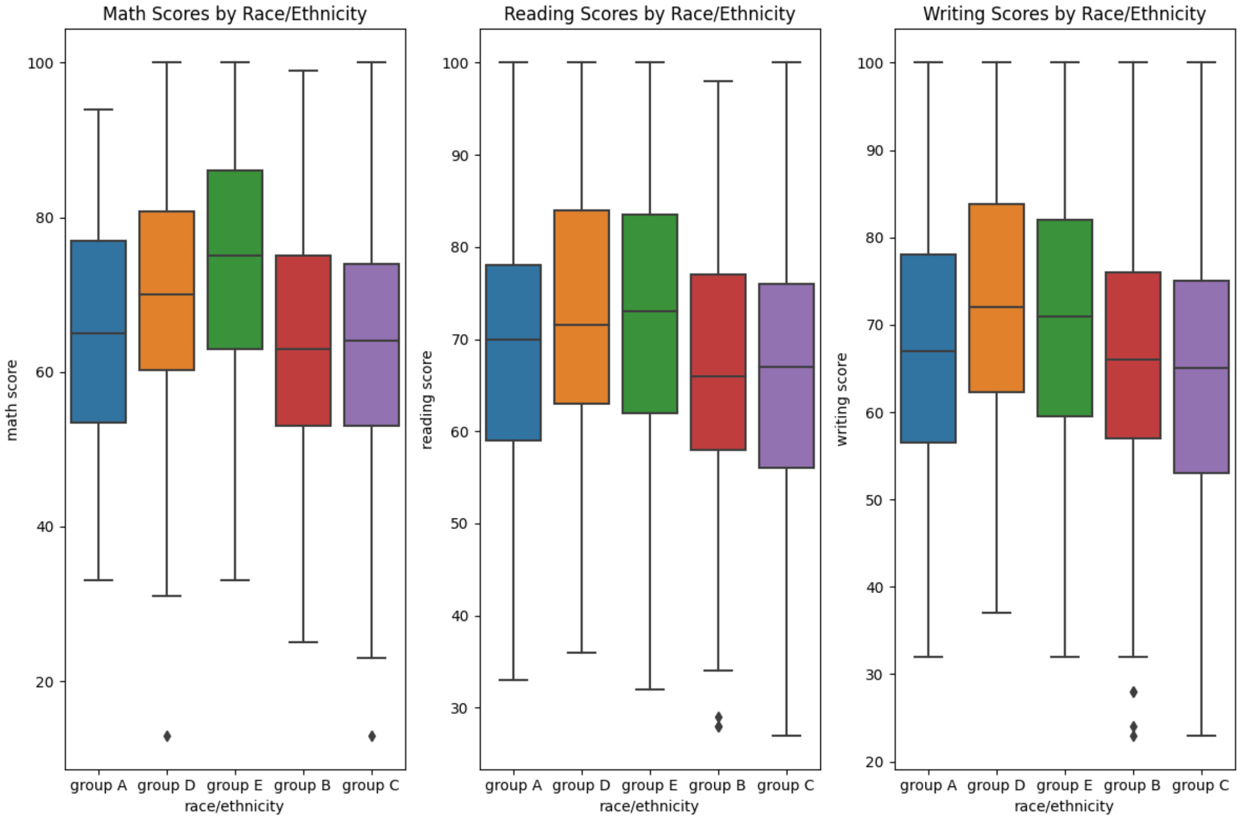
```
plt.figure(figsize=(10, 6))
sns.heatmap(df[['math score', 'reading score', 'writing
score']].corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Matrix of Scores')
plt.show()
```



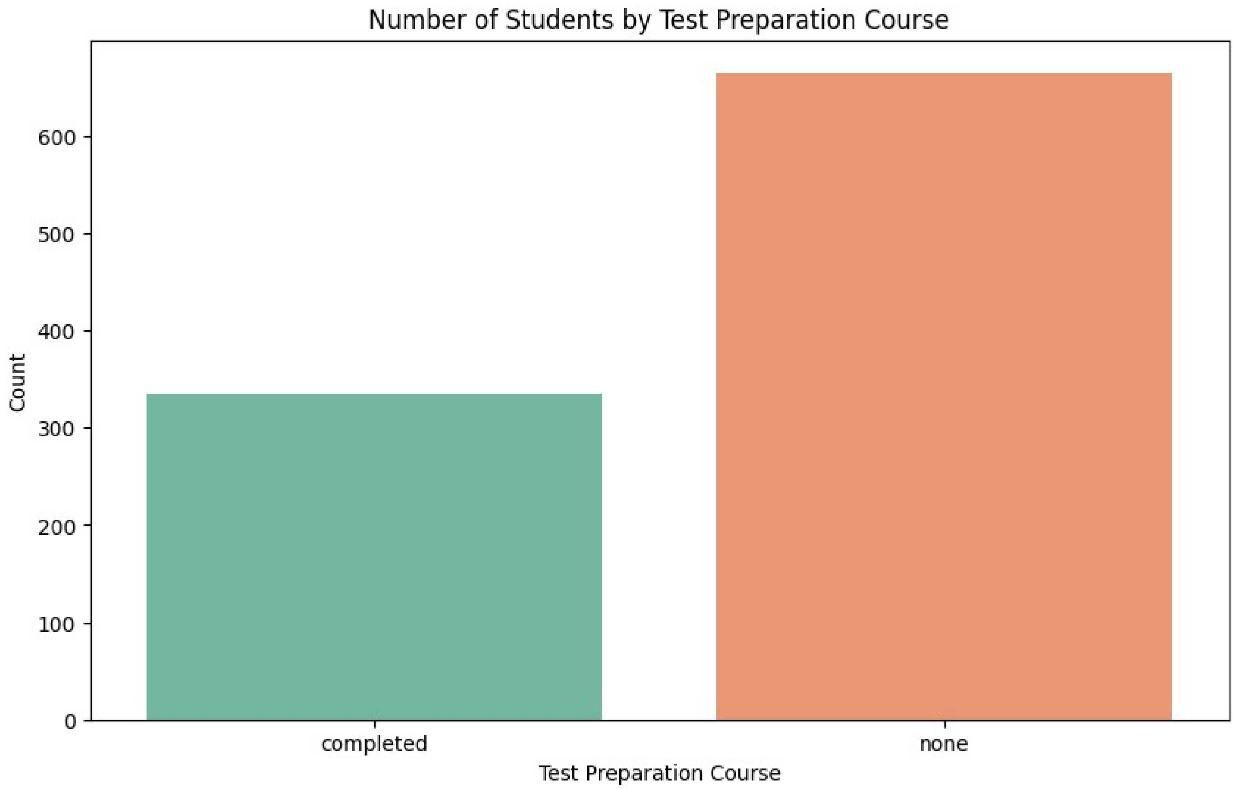
```
plt.figure(figsize=(10, 6))
sns.scatterplot(x='math score', y='reading score', data=df,
hue='gender')
plt.title('Math Score vs Reading Score by Gender')
plt.xlabel('Math Score')
plt.ylabel('Reading Score')
plt.show()
```



```
plt.figure(figsize=(12, 8))

# Math Scores by Race/Ethnicity
plt.subplot(1, 3, 1)
sns.boxplot(x='race/ethnicity', y='math score', data=df)
plt.title('Math Scores by Race/Ethnicity')

# Reading Scores by Race/Ethnicity
plt.subplot(1, 3, 2)
sns.boxplot(x='race/ethnicity', y='reading score', data=df)
plt.title('Reading Scores by Race/Ethnicity')

plt.subplot(1, 3, 3)
sns.boxplot(x='race/ethnicity', y='writing score', data=df)
plt.title('Writing Scores by Race/Ethnicity')

plt.tight_layout()
plt.show()
```



```
plt.figure(figsize=(10, 6))
sns.countplot(x='test preparation course', data=df, palette='Set2')
plt.title('Number of Students by Test Preparation Course')
plt.xlabel('Test Preparation Course')
plt.ylabel('Count')
plt.show()
```



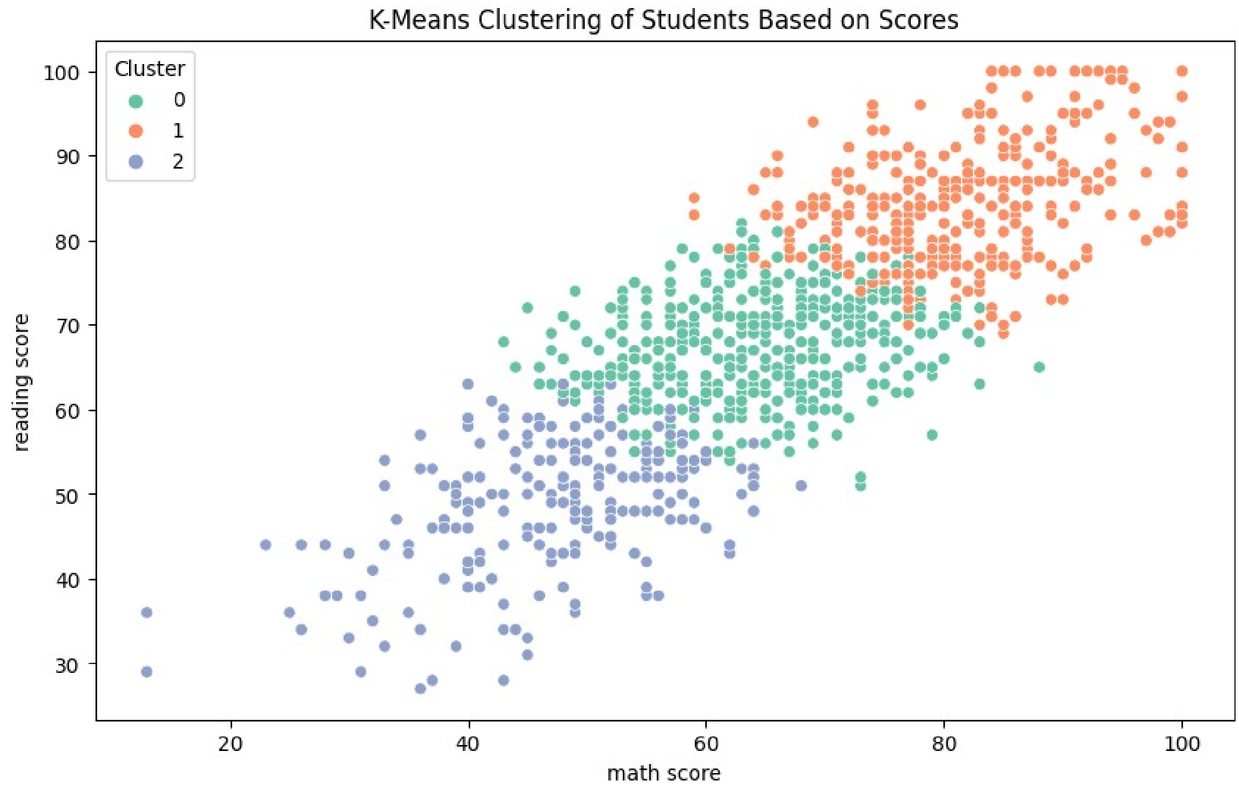
```python
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder

# Encode categorical variables
df_encoded = df.copy()
le = LabelEncoder()

# Encoding categorical columns
df_encoded['gender'] = le.fit_transform(df_encoded['gender'])
df_encoded['race/ethnicity'] =
le.fit_transform(df_encoded['race/ethnicity'])
df_encoded['parental level of education'] =
le.fit_transform(df_encoded['parental level of education'])
df_encoded['lunch'] = le.fit_transform(df_encoded['lunch'])
df_encoded['test preparation course'] =
le.fit_transform(df_encoded['test preparation course'])

# Define features and target variable
X = df_encoded.drop(['math score'], axis=1)  # Features
y = df_encoded['math score']  # Target variable

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)
```

```python
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Initialize the model
model = LinearRegression()

# Train the model
model.fit(X_train, y_train)

# Make predictions
y_pred = model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f'Mean Squared Error: {mse}')
print(f'R-squared: {r2}')
```

```
Mean Squared Error: 31.98807654822675
R-squared: 0.8633325615941331
```

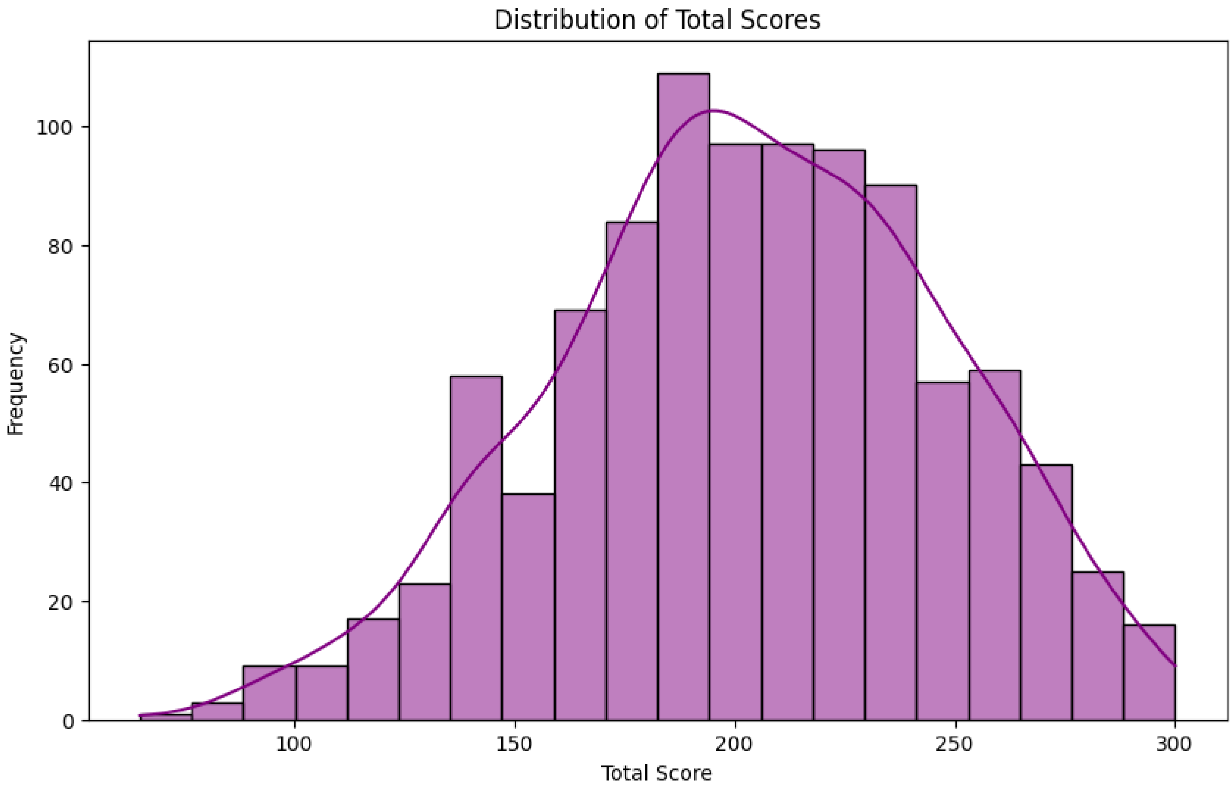
```python
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler

# Standardize the scores
scaler = StandardScaler()
scores_scaled = scaler.fit_transform(df[['math score', 'reading score', 'writing score']])

# Apply K-Means clustering
kmeans = KMeans(n_clusters=3, random_state=42)
df['Cluster'] = kmeans.fit_predict(scores_scaled)

# Visualize the clusters
plt.figure(figsize=(10, 6))
sns.scatterplot(x='math score', y='reading score', hue='Cluster', data=df, palette='Set2')
plt.title('K-Means Clustering of Students Based on Scores')
plt.show()
```

```
/opt/conda/lib/python3.10/site-packages/sklearn/cluster/
_kmeans.py:870: FutureWarning: The default value of `n_init` will
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
  warnings.warn(
```

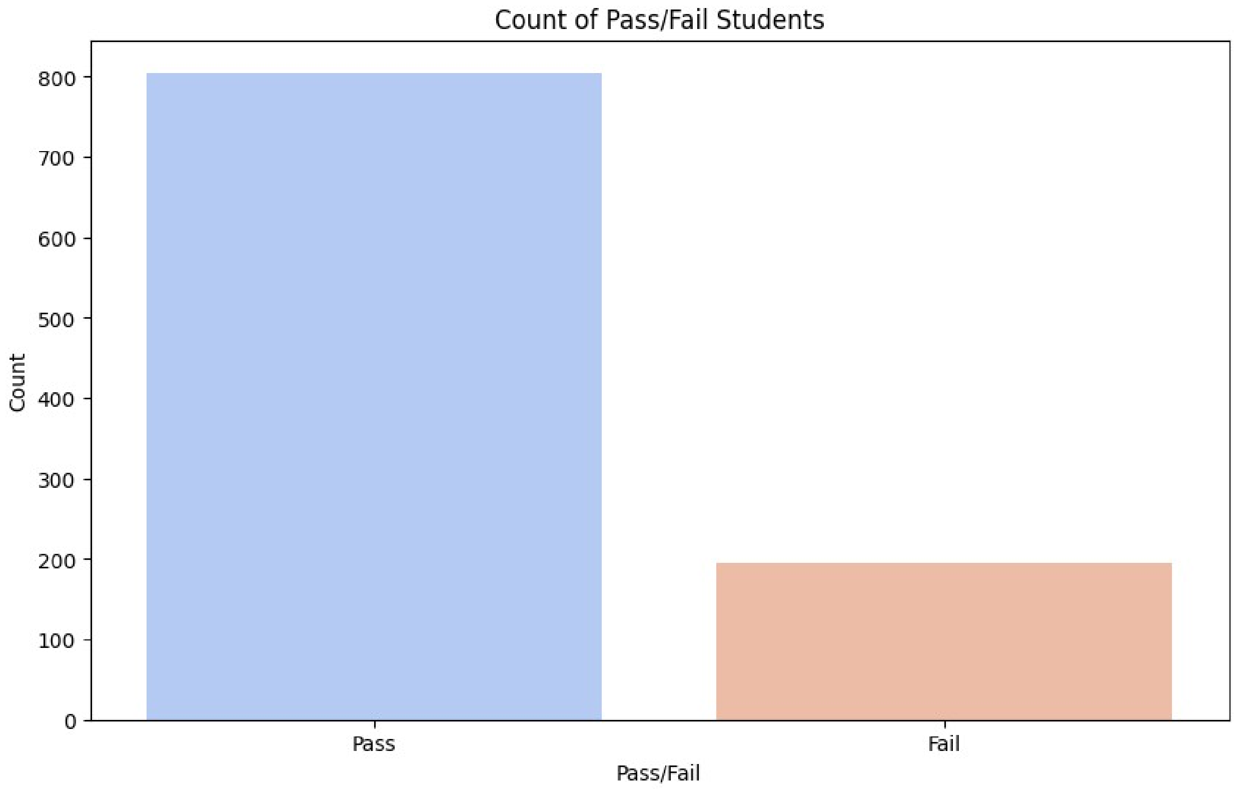


```
df['total score'] = df['math score'] + df['reading score'] +
df['writing score']

# Visualize the distribution of total scores
plt.figure(figsize=(10, 6))
sns.histplot(df['total score'], kde=True, color='purple')
plt.title('Distribution of Total Scores')
plt.xlabel('Total Score')
plt.ylabel('Frequency')
plt.show()
```

```
/opt/conda/lib/python3.10/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



```
df['pass/fail'] = np.where((df['math score'] >= 50) & (df['reading
score'] >= 50) & (df['writing score'] >= 50), 'Pass', 'Fail')

# Visualize the count of students passing and failing
plt.figure(figsize=(10, 6))
sns.countplot(x='pass/fail', data=df, palette='coolwarm')
plt.title('Count of Pass/Fail Students')
plt.xlabel('Pass/Fail')
plt.ylabel('Count')
plt.show()
```


Count of Pass/Fail Students
Count
0
100
200
300
400
500
600
700
800
Pass
Fail
Pass/Fail