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

sns.set(style="whitegrid")
```

Interpretation:

This code loads Python executive that needs analysis and visualisation of data. The tool manipulates and works with data with efficiency using Pandas. NumPy contains operations of numbers required in computations. Matplotlib and Seaborn are visualization libraries that are used to create charts and graphs. To increase the visual results, it is necessary to set the Seaborn style to whitegrid to make the results more readable and understandable.

```
df = pd.read_csv("/content/archive (2).zip")
df.head(6)
```

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
0	female	group B	bachelor's degree	standard	none	72	72	74
1	female	group C	some college	standard	completed	69	90	88
2	female	group B	master's degree	standard	none	90	95	93
3	male	group A	associate's degree	free/reduced	none	47	57	44
4	male	group C	some college	standard	none	76	78	75
5	female	group B	associate's degree	standard	none	71	83	78

Next steps: Generate code with df | New interactive sheet

Interpretation:

The information is inserted into a dynamic table named df. It is checked with the help of the head()-function that verifies the correct loading with preliminary records. It shows that the data sets include demographic variables and examination scores. This is necessary step to know what and how the data is organized. It ensures that the dataset is accessible to be analyzed.

```
df.info()
df.describe()
```

```
<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
```

	math score	reading score	writing score
count	1000.00000	1000.000000	1000.000000
mean	66.08900	69.169000	68.054000
std	15.16308	14.600192	15.195657
min	0.00000	17.000000	10.000000
25%	57.00000	59.000000	57.750000
50%	66.00000	70.000000	69.000000
75%	77.00000	79.000000	79.000000
max	100.00000	100.000000	100.000000

Interpretation:

The info command provides the data concerning the dataset including the names of the columns and the type of data available. It also shows the presence of ambiguous values in every column. The summary of the statistics of numerical variables is described. Such statistics includes mean, standard deviation, minimum and the maximum. The prescribed step may help to be acquainted with the overall format and arrangement of test scores.

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

```
np.int64(0)
```

Interpretation:

The step will ensure that the values missing in the columns of the dataset are verified. Data is missing, and this could hamper the validity of the analysis and must be addressed. The duplicate check is performed in order to ensure that the records of all the students are unique. Possible sources of bias may also be duplicate records unless removed. The step concludes upon the cleaning being necessary or otherwise.

```
df = df.drop_duplicates()
df = df.dropna()
```

Interpretation:

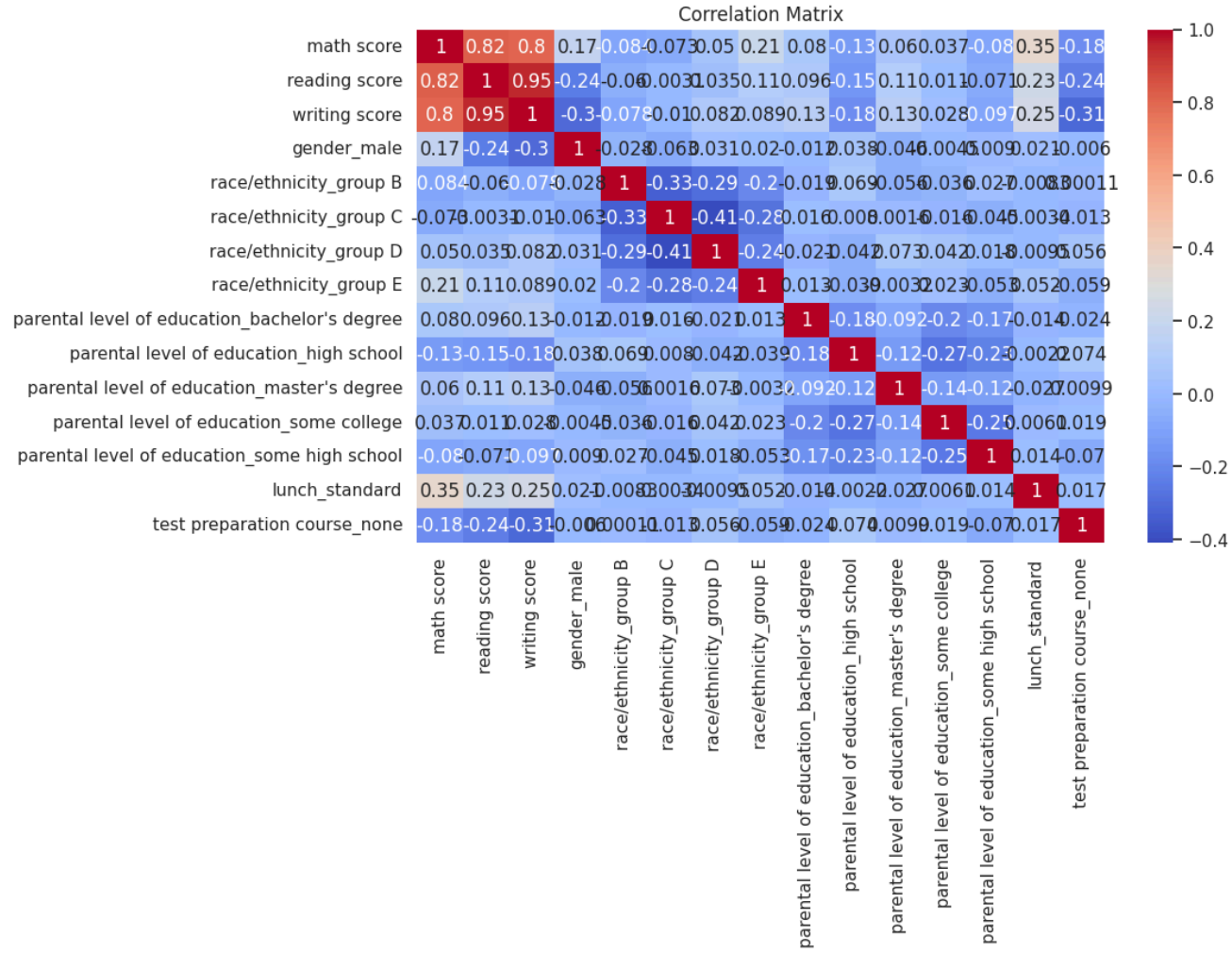
In order to get rid of duplicate records, duplicate records are removed. The other tool is the dropping of missing values in order to have full records. Raw data that is free of contamination improves the accuracy of statistics. This is an action that will be taken to ensure that every entry is valid student observation. It prepares the dataset so that it will be analyzed in a proper way.

```
df_encoded = pd.get_dummies(df, drop_first=True)
```

Interpretation:

Categorical variables are converted to numerical format through one-hot encoding. This allows categorical information to be included in mathematics analysis. The removal of the former type saves on the redundancy of variables. Some kind of encoding is required in correlation and modeling. It will prepare the data to undergo additional statistical analysis.

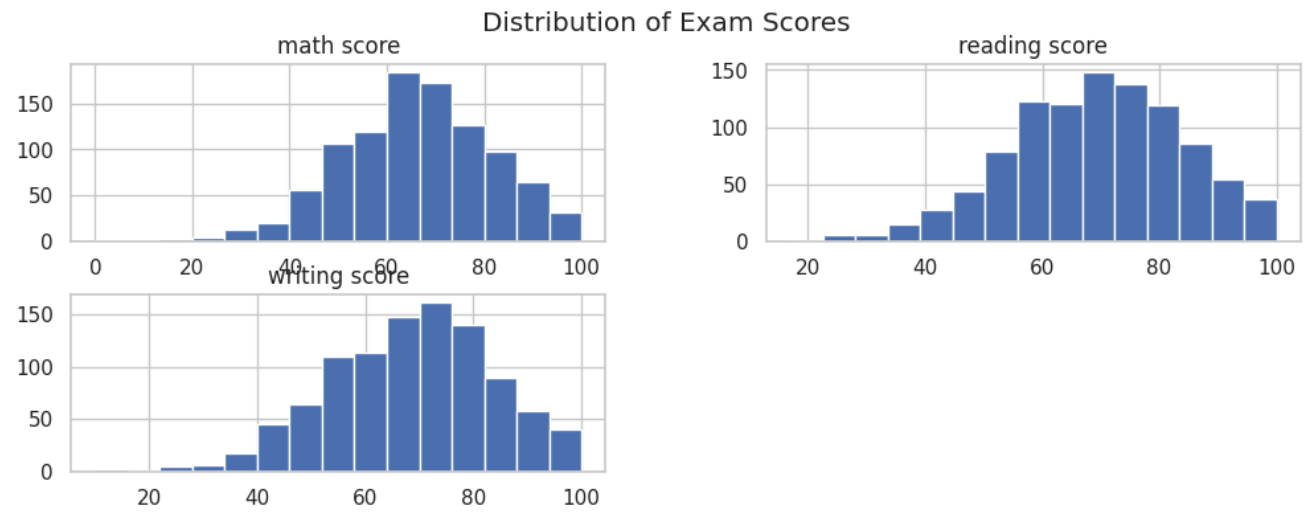
```
plt.figure(figsize=(10,6))
sns.heatmap(df_encoded.corr(), annot=True, cmap="coolwarm")
plt.title("Correlation Matrix")
plt.show()
```



Interpretation:

One such graph is the heatmap that depicts the correlation between all the variables. Math, reading and writing scores appear to correlate with each other strongly on the yellow scale. This is whereby students who are good at one subject are good at others. Demographic variables that are not strongly correlated include the other variables. The heatmap is useful in making the necessary relations in a short time.

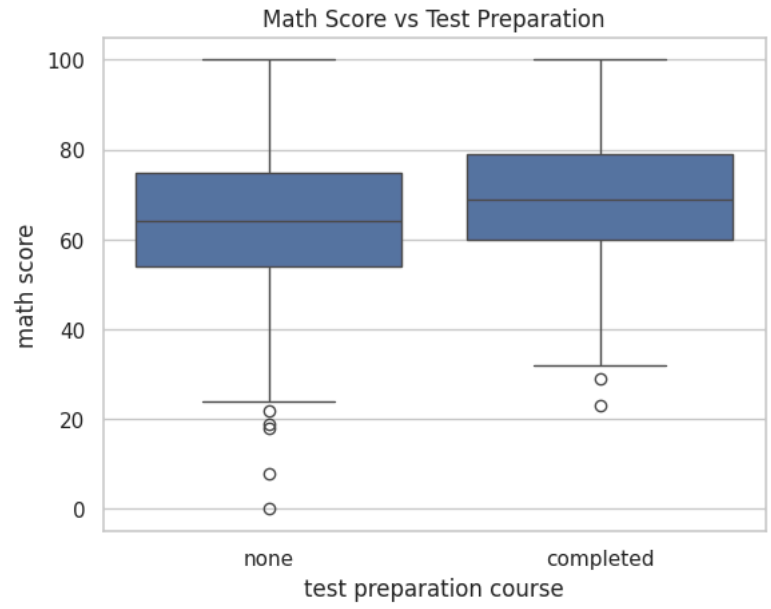
```
df[['math score', 'reading score', 'writing score']].hist(bins=15, figsize=(12,4))
plt.suptitle("Distribution of Exam Scores")
plt.show()
```



Interpretation:

The way scores are spread in the existing examinations is shown with the help of histograms. Most students are getting via middle range rather than extremes. It is of rough normal distribution. This means that there is a moderate performance among the students. There are also no drastic skewness scores.

```
sns.boxplot(x='test preparation course', y='math score', data=df)
plt.title("Math Score vs Test Preparation")
plt.show()
```



Interpretation:

The difference between the math score of learners who attended the preparation and those who did not are going to be described using the boxplot. The prepared students show superior median scores. Their findings also appear more coherent. This is a pointer that preparation courses have a performance enhancing effect. The analysis proves the importance of preparation of studies.

```
plt.figure(figsize=(15,4))

plt.subplot(1,3,1)
plt.scatter(df['math score'], df['reading score'])
plt.xlabel("Math Score")
plt.ylabel("Reading Score")
plt.title("Math vs Reading")
```

```
plt.subplot(1,3,2)
plt.scatter(df['reading score'], df['writing score'])
plt.xlabel("Reading Score")
plt.ylabel("Writing Score")
plt.title("Reading vs Writing")

plt.subplot(1,3,3)
plt.scatter(df['math score'], df['writing score'])
plt.xlabel("Math Score")
plt.ylabel("Writing Score")
plt.title("Math vs Writing")

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