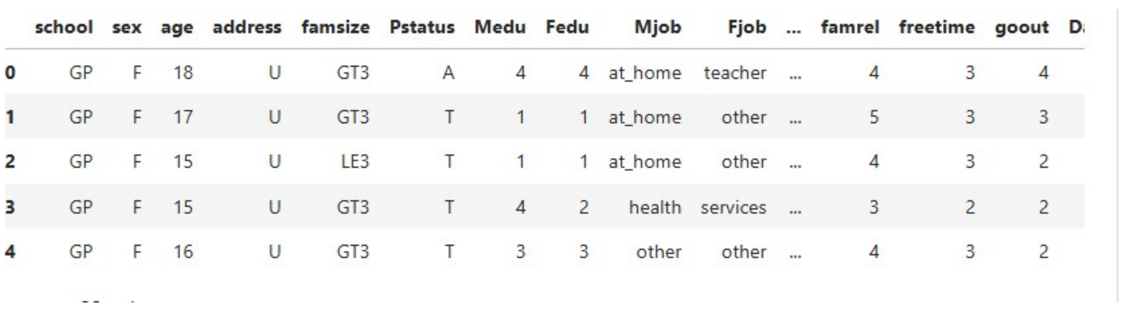
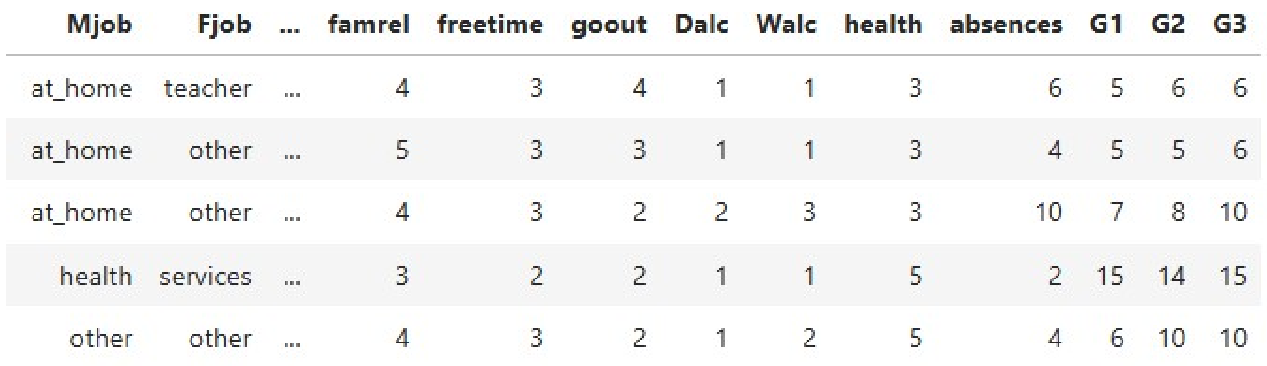
TASK 1: Data Analysis Project Using Python.

➔CODING..........................

```
-> import pandas as pd

   import numpy as np
   import matplotlib.pyplot as plt

   # Load the dataset

   df = pd.read_csv('student-mat.csv')
```



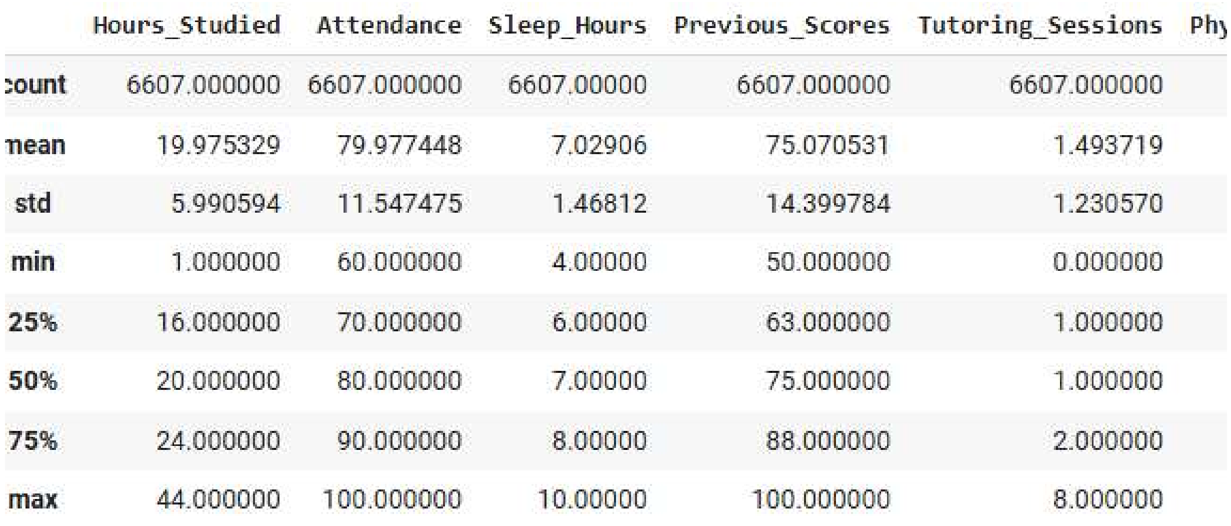
	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	freetime	goout	D
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4	3	4	
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	5	3	3	
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	4	3	2	
3	GP	F	15	U	GT3	T	4	2	health	services	...	3	2	2	
4	GP	F	16	U	GT3	T	3	3	other	other	...	4	3	2	



Mjob	Fjob	...	famrel	freetime	goout	Dalc	Walc	health	absences	G1	G2	G3
at_home	teacher	...	4	3	4	1	1	3	6	5	6	6
at_home	other	...	5	3	3	1	1	3	4	5	5	6
at_home	other	...	4	3	2	2	3	3	10	7	8	10
health	services	...	3	2	2	1	1	5	2	15	14	15
other	other	...	4	3	2	1	2	5	4	6	10	10

#Display the first few rows

```
print("First few rows of the dataset:")
print(df.head())
```



	Hours_Studied	Attendance	Sleep_Hours	Previous_Scores	Tutoring_Sessions	Phy
ount	6607.000000	6607.000000	6607.00000	6607.000000	6607.000000	
nean	19.975329	79.977448	7.02906	75.070531	1.493719	
std	5.990594	11.547475	1.46812	14.399784	1.230570	
min	1.000000	60.000000	4.00000	50.000000	0.000000	
25%	16.000000	70.000000	6.00000	63.000000	1.000000	
50%	20.000000	80.000000	7.00000	75.000000	1.000000	
75%	24.000000	90.000000	8.00000	88.000000	2.000000	
max	44.000000	100.000000	10.00000	100.000000	8.000000	

## Check for missing values

```
print("\nMissing values in the dataset:")
print(df.isnull().sum

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6607 entries, 0 to 6606
Data columns (total 20 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   Hours_Studied               6607 non-null   int64
 1   Attendance                  6607 non-null   int64
 2   Parental_Involvement        6607 non-null   object
 3   Access_to_Resources         6607 non-null   object
 4   Extracurricular_Activities  6607 non-null   object
 5   Sleep_Hours                 6607 non-null   int64
 6   Previous_Scores           6607 non-null   int64
 7   Motivation_Level          6607 non-null   object
 8   Internet_Access           6607 non-null   object
 9   Tutoring_Sessions         6607 non-null   int64
 10  Family_Income             6607 non-null   object
 11  Teacher_Quality           6529 non-null   object
 12  School_Type               6607 non-null   object
 13  Peer_Influence            6607 non-null   object
 14  Physical_Activity         6607 non-null   int64
 15  Learning_Disabilities     6607 non-null   object
 16  Parental_Education_Level  6517 non-null   object
 17  Distance_from_Home        6540 non-null   object
 18  Gender                    6607 non-null   object
 19  Exam_Score                6607 non-null   int64
dtypes: int64(7), object(13)
memory usage: 1.0+ MB
```

```
# Display column data types

print("\nData types of the columns:")
print(df.dtypes)
```

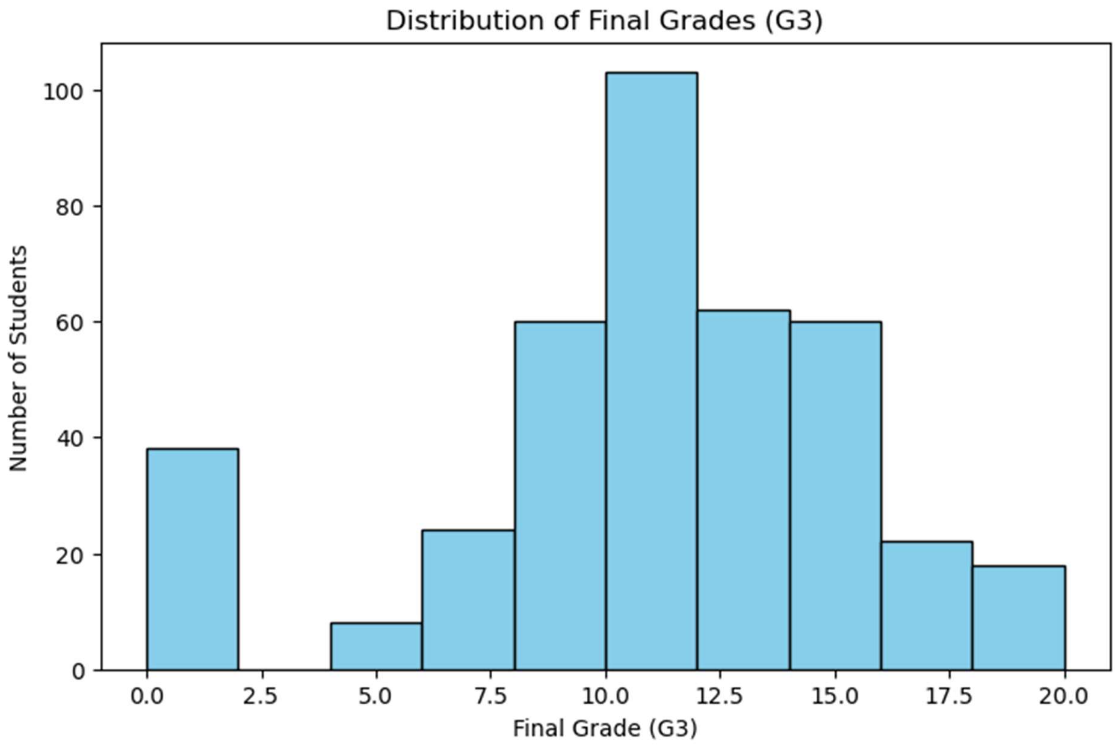
```
Hours_Studied                 int64
Attendance                    int64
Parental_Involvement         object
Access_to_Resources          object
Extracurricular_Activities   object
Sleep_Hours                   int64
Previous_Scores               int64
Motivation_Level             object
```

```
Internet_Access           object
Tutoring_Sessions          int64
Family_Income             object
Teacher_Quality           object
School_Type              object
Peer_Influence            object
Physical_Activity          int64
Learning_Disabilities       object
Parental_Education_Level     object
Distance_from_Home         object
Gender                 object
Exam_Score               int64
dtype: object
```

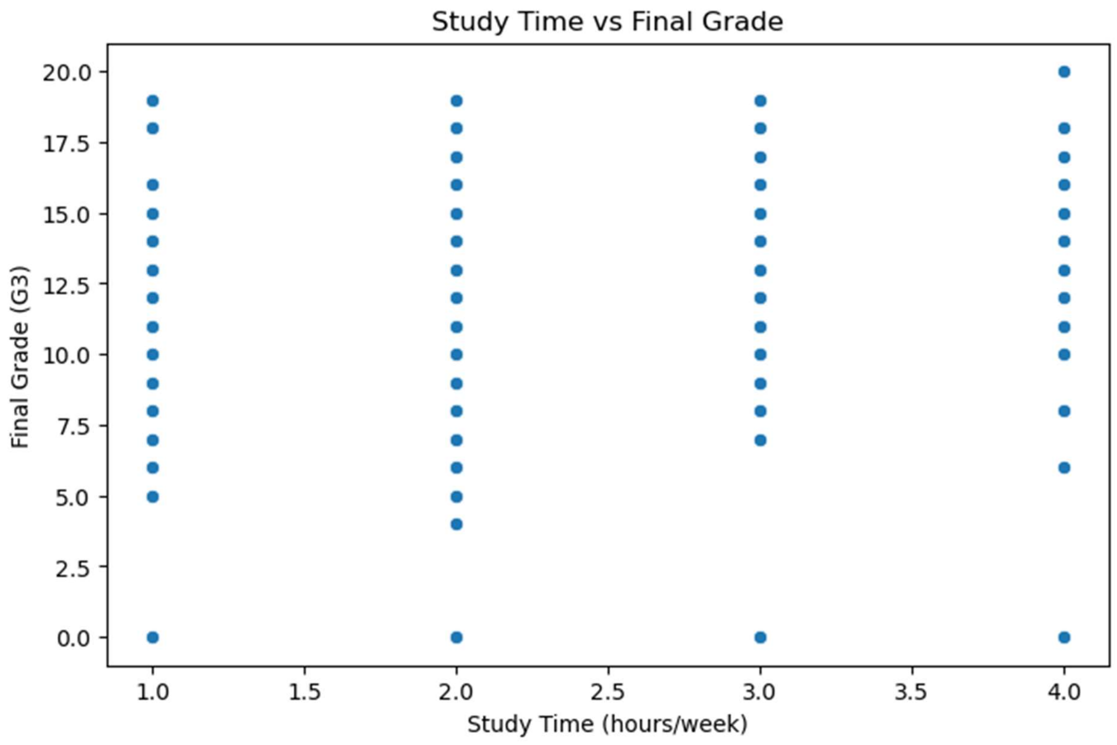
```
# Understand the dataset's size

print(f"\nDataset size: {df.shape}")

# Handle missing values (if any)

for col in df.columns:
   if df[col].dtype == np.float64 or df[col].dtype == np.int64:
      df[col] = df[col].fillna(df[col].median())
```

```
Index(['Hours_Studied', 'Attendance', 'Parental_Involvement',
     'Access_to_Resources', 'Extracurricular_Activities',
'Sleep_Hours',
     'Previous_Scores', 'Motivation_Level', 'Internet_Access',
     'Tutoring_Sessions', 'Family_Income', 'Teacher_Quality',
'School_Type',
      'Peer_Influence', 'Physical_Activity',
'Learning_Disabilities',
      'Parental_Education_Level', 'Distance_from_Home',
'Gender',
      'Exam_Score'],
     dtype='object')

# Remove duplicate entries

df = df.drop_duplicates()

# Calculate average score in math (G3)

average_score = df['G3'].mean()
print(f"\nAverage score in math (G3): {average_score}")

# Count students who scored above 15

students_above_15 = df[df['G3'] > 15].shape[0]
print(f"Number of students scored above 15:
{students_above_15}")

# Calculate correlation between study time and final grade

correlation = df['studytime'].corr(df['G3'])
print(f"Correlation between study time and final grade:
{correlation}")

# Calculate average final grade by gender
```

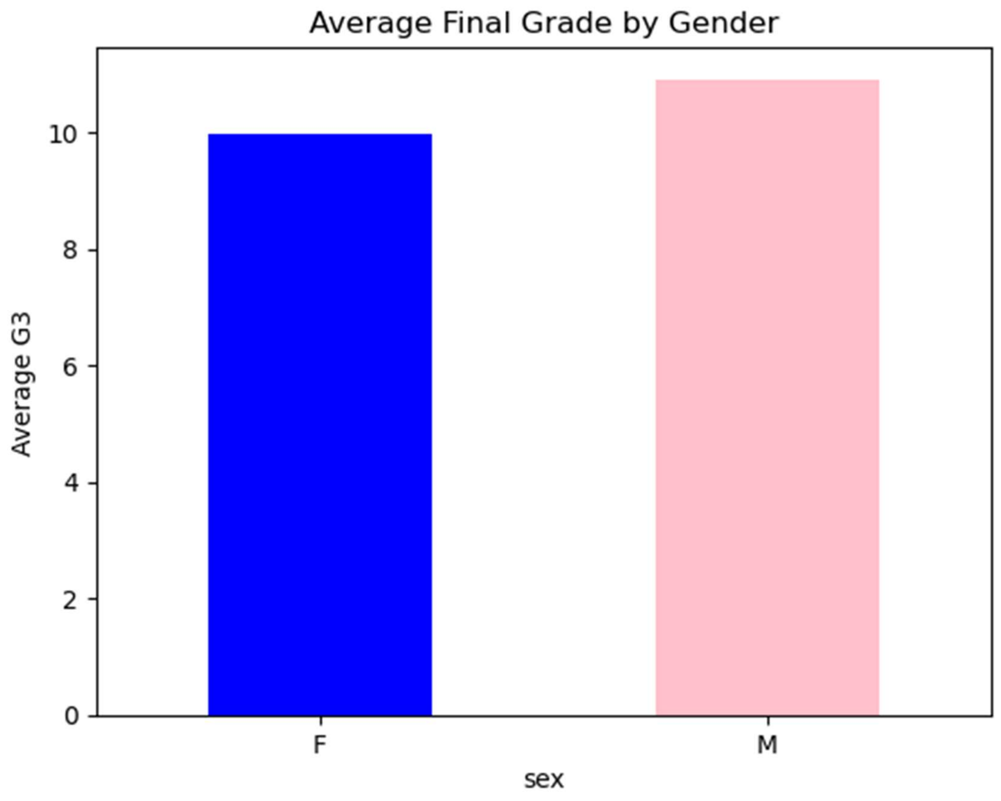
```
average_grade_by_gender = df.groupby('sex')['G3'].mean()
print(f"Average final grade by gender:
\n{average_grade_by_gender}")

# Plot histogram of final grades

plt.hist(df['G3'], bins=10, edgecolor='black')
plt.xlabel('Final Grade')
plt.ylabel('Frequency')
plt.title('Histogram of Final Grades')
plt.show()
```



```
# Create scatter plot between study time and final grade

plt.scatter(df['studytime'], df['G3'])
plt.xlabel('Study Time')
plt.ylabel('Final Grade')
plt.title('Scatter Plot Between Study Time and Final Grade')
plt.show()
```



```
# Create bar chart comparing average scores of male and
female students

average_scores = df.groupby('sex')['G3'].mean()
average_scores.plot(kind='bar')
plt.xlabel('Gender')
plt.ylabel('Average Score')
plt.title('Average Scores by Gender')
plt.show()
```


Average Final Grade by Gender
Average G3
0
2
4
6
8
10
F
M
sex