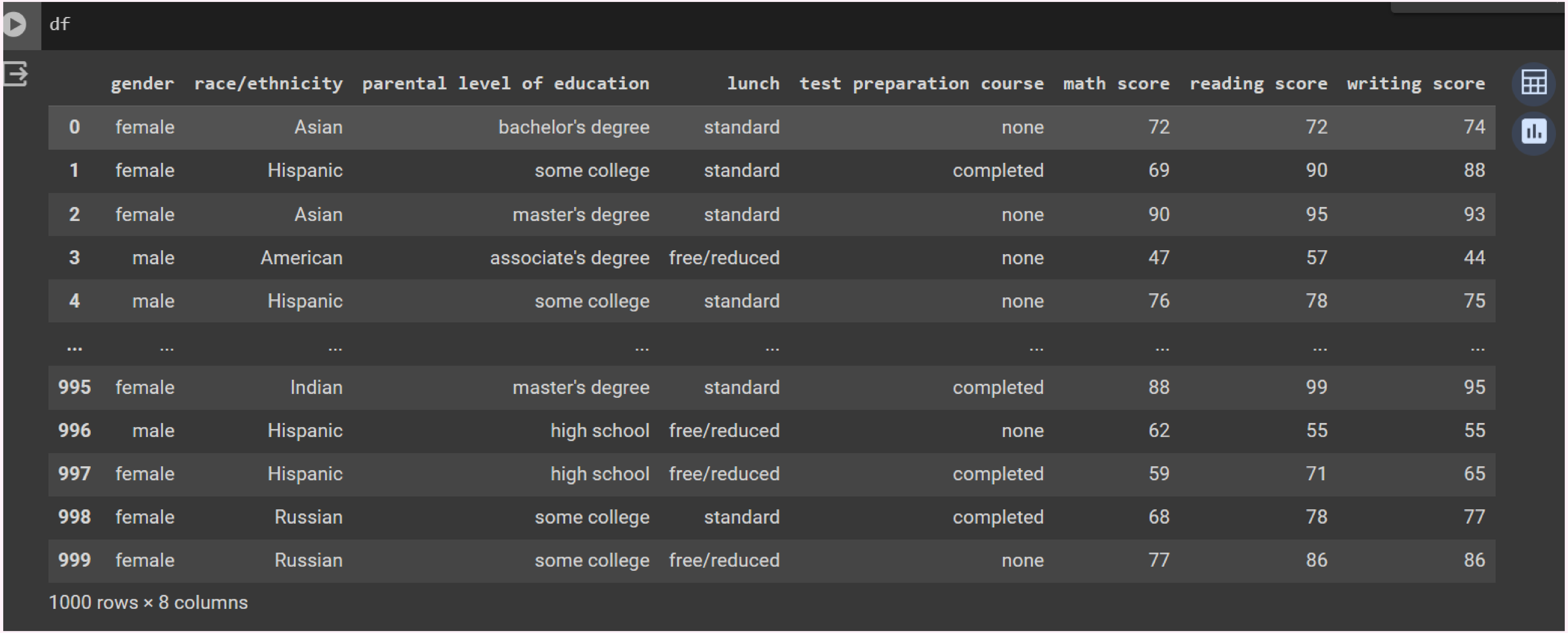
Designer Portfolio

www.reallygreatsite.com

STUDENT PERFORMANCE IN EXAMS

DYLAN JAMES N. DEJORAS

INITIAL DATAFRAME



```
df
```

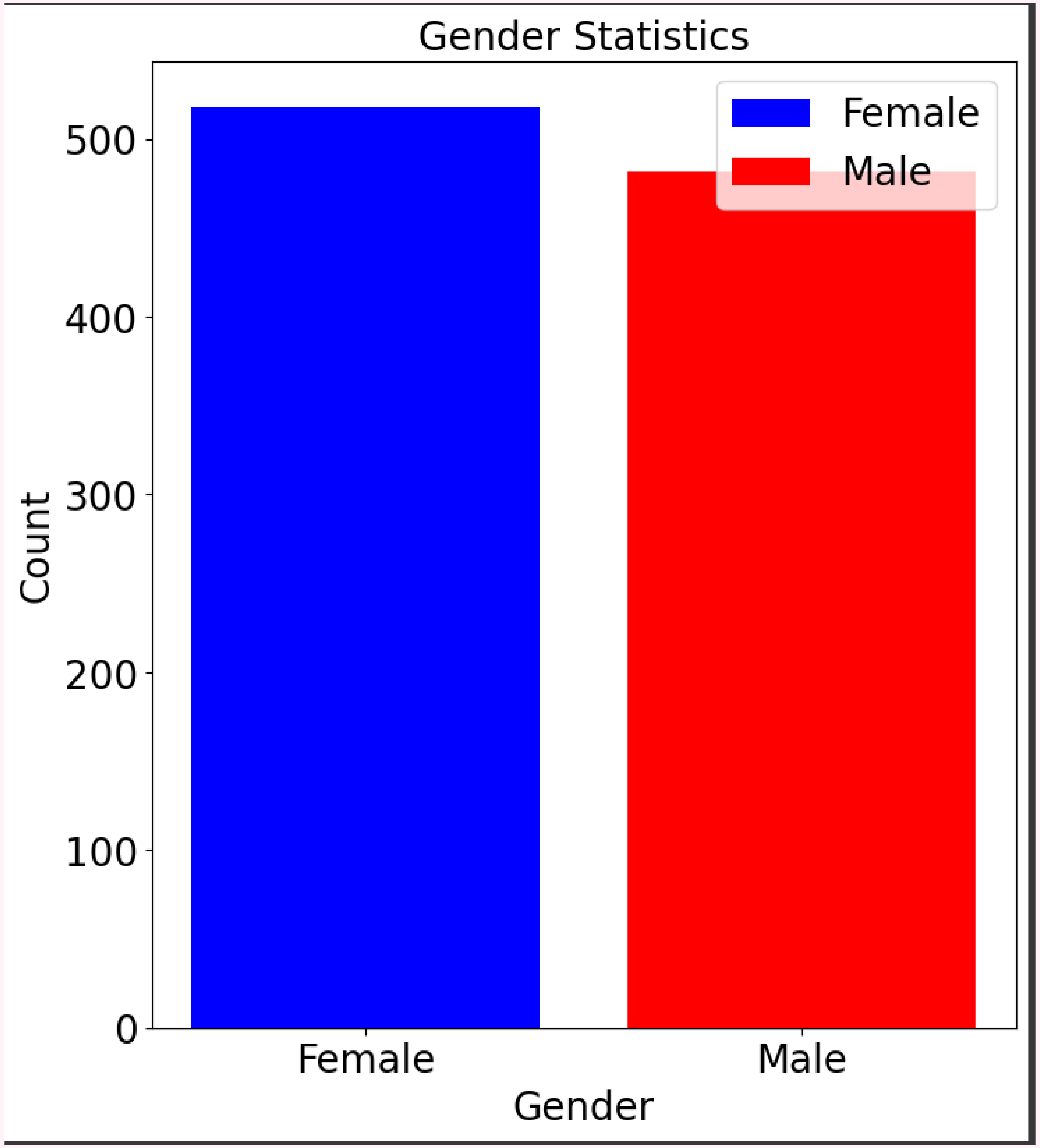
	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
0	female	Asian	bachelor's degree	standard	none	72	72	74
1	female	Hispanic	some college	standard	completed	69	90	88
2	female	Asian	master's degree	standard	none	90	95	93
3	male	American	associate's degree	free/reduced	none	47	57	44
4	male	Hispanic	some college	standard	none	76	78	75
...	...	...	...	...	...	...	...	...
995	female	Indian	master's degree	standard	completed	88	99	95
996	male	Hispanic	high school	free/reduced	none	62	55	55
997	female	Hispanic	high school	free/reduced	completed	59	71	65
998	female	Russian	some college	standard	completed	68	78	77
999	female	Russian	some college	free/reduced	none	77	86	86

1000 rows × 8 columns

STUDENT GENDER

OUT OF 1000 STUDENTS:

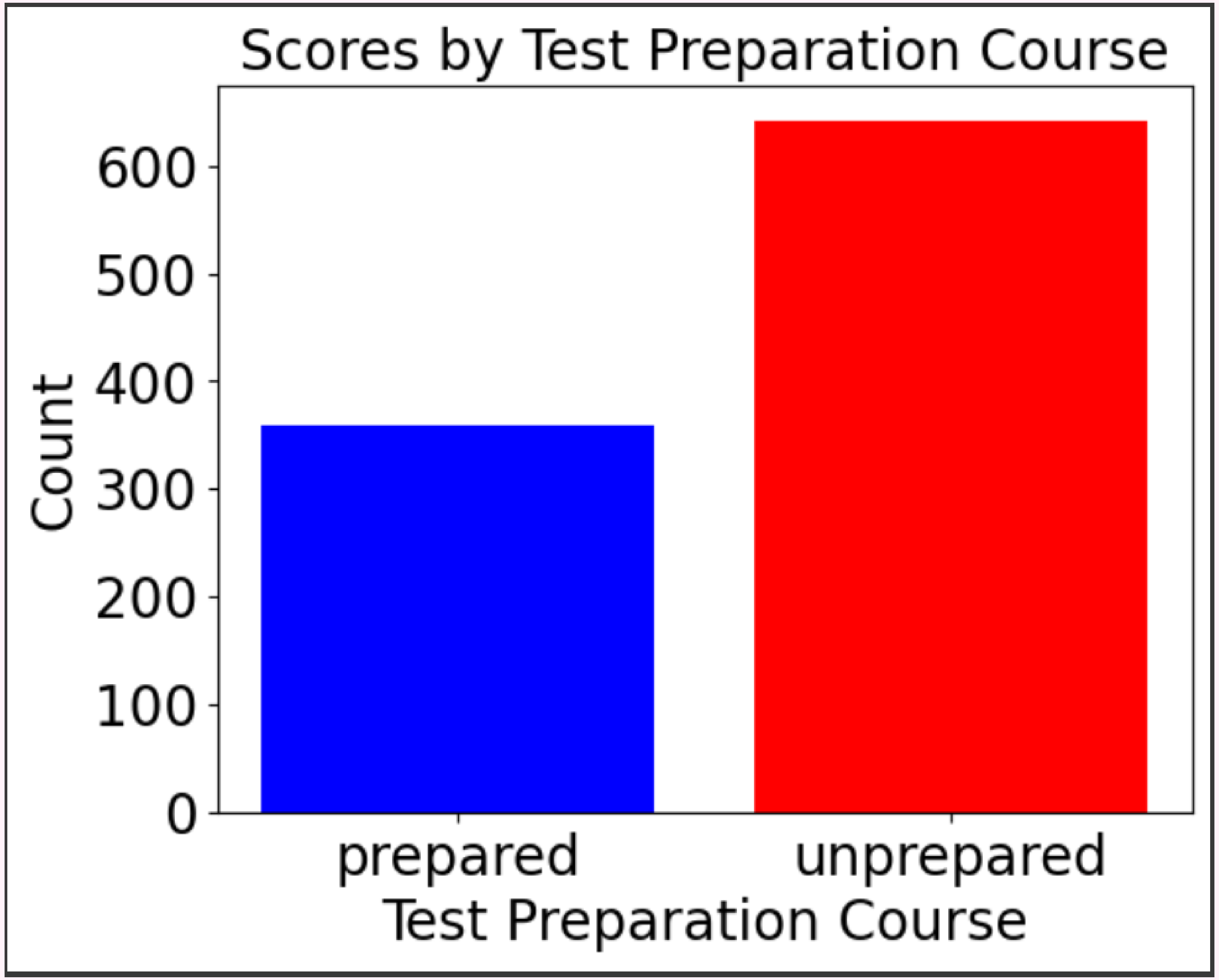
- **FEMALE - 518**
- **MALE - 482**



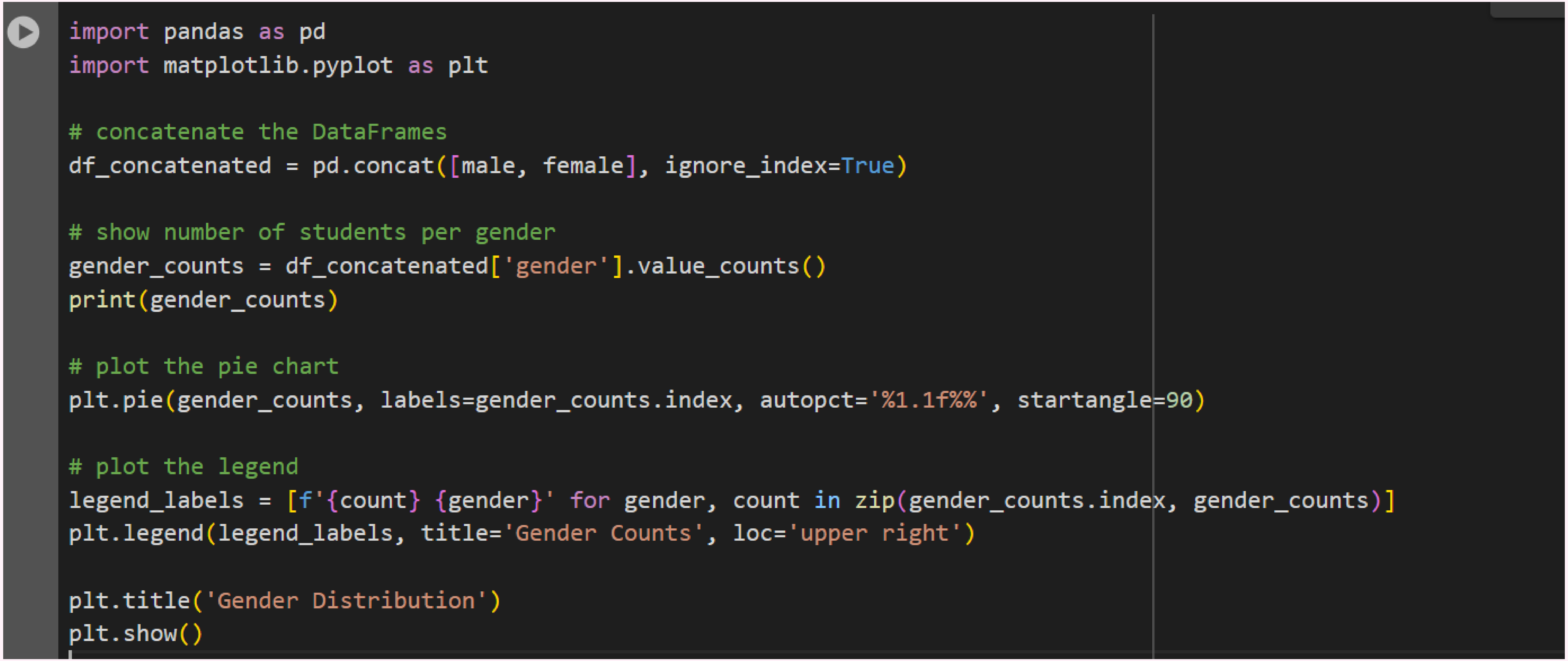
STUDENT PREPARATION

OUT OF 1000 STUDENTS:

- **TOOK THE TEST PREPARATION COURSE- 358**
- **DID NOT TOOK THE TEST PREPARTION COURSE- 642**



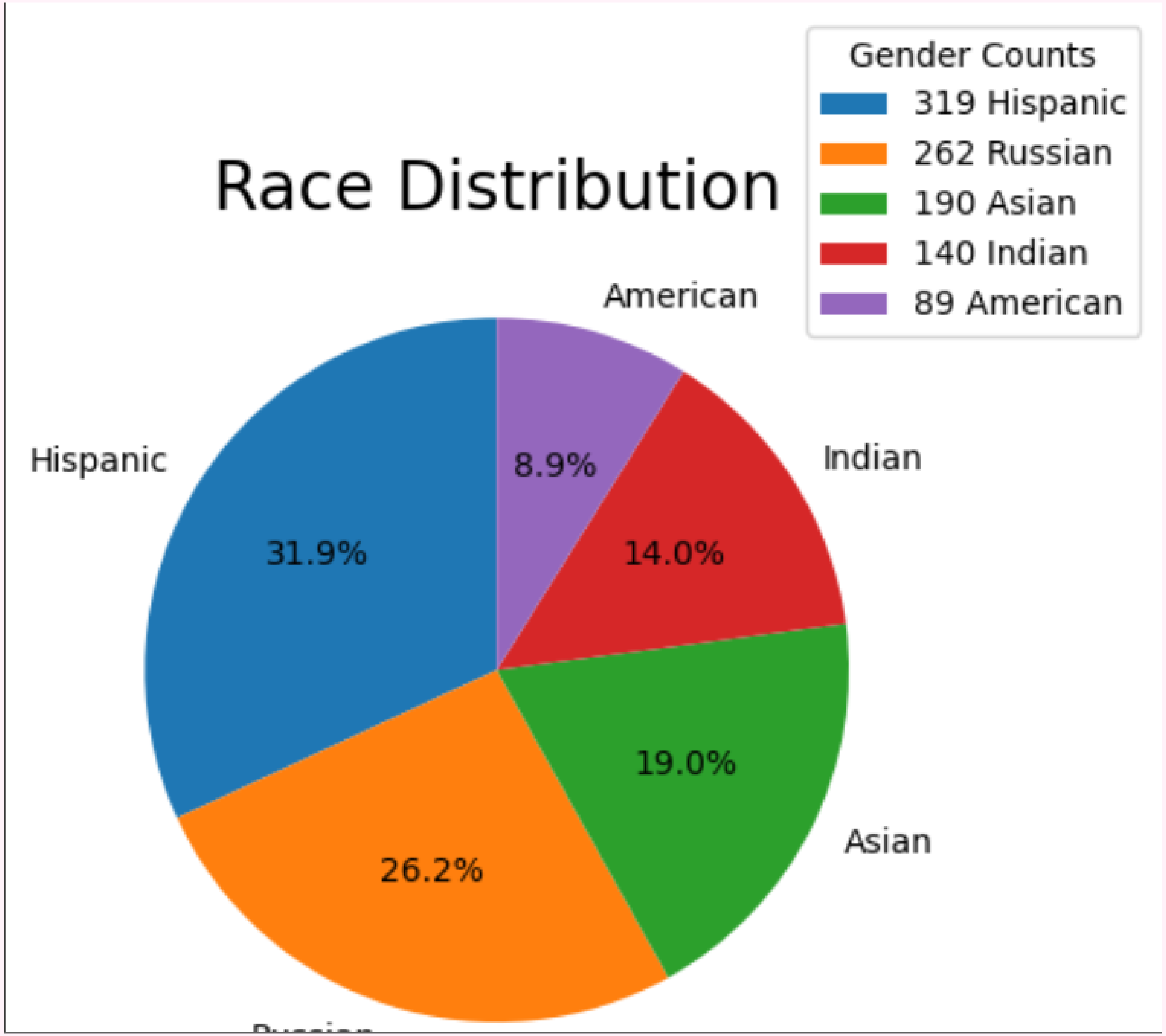
BAR GRAPH CODE



```python
import pandas as pd
import matplotlib.pyplot as plt

# concatenate the DataFrames
df_concatenated = pd.concat([male, female], ignore_index=True)

# show number of students per gender
gender_counts = df_concatenated['gender'].value_counts()
print(gender_counts)

# plot the pie chart
plt.pie(gender_counts, labels=gender_counts.index, autopct='%1.1f%%', startangle=90)

# plot the legend
legend_labels = [f'{count} {gender}' for gender, count in zip(gender_counts.index, gender_counts)]
plt.legend(legend_labels, title='Gender Counts', loc='upper right')

plt.title('Gender Distribution')
plt.show()
```

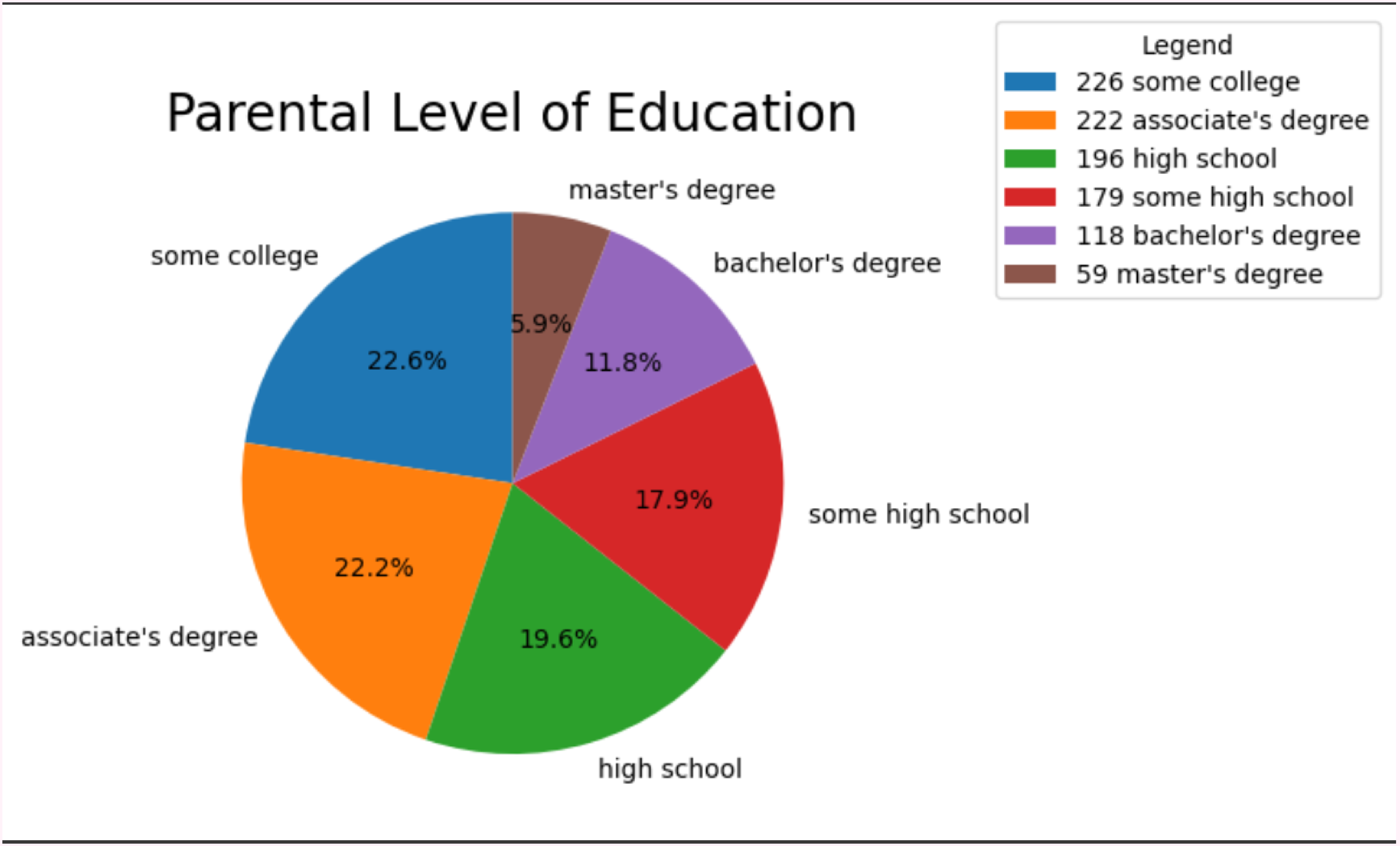
STUDENT RACE

- **AMERICAN - 89**
- **ASIAN - 190**
- **HISPANIC - 319**
- **RUSSIAN - 262**
- **INDIAN - 140**

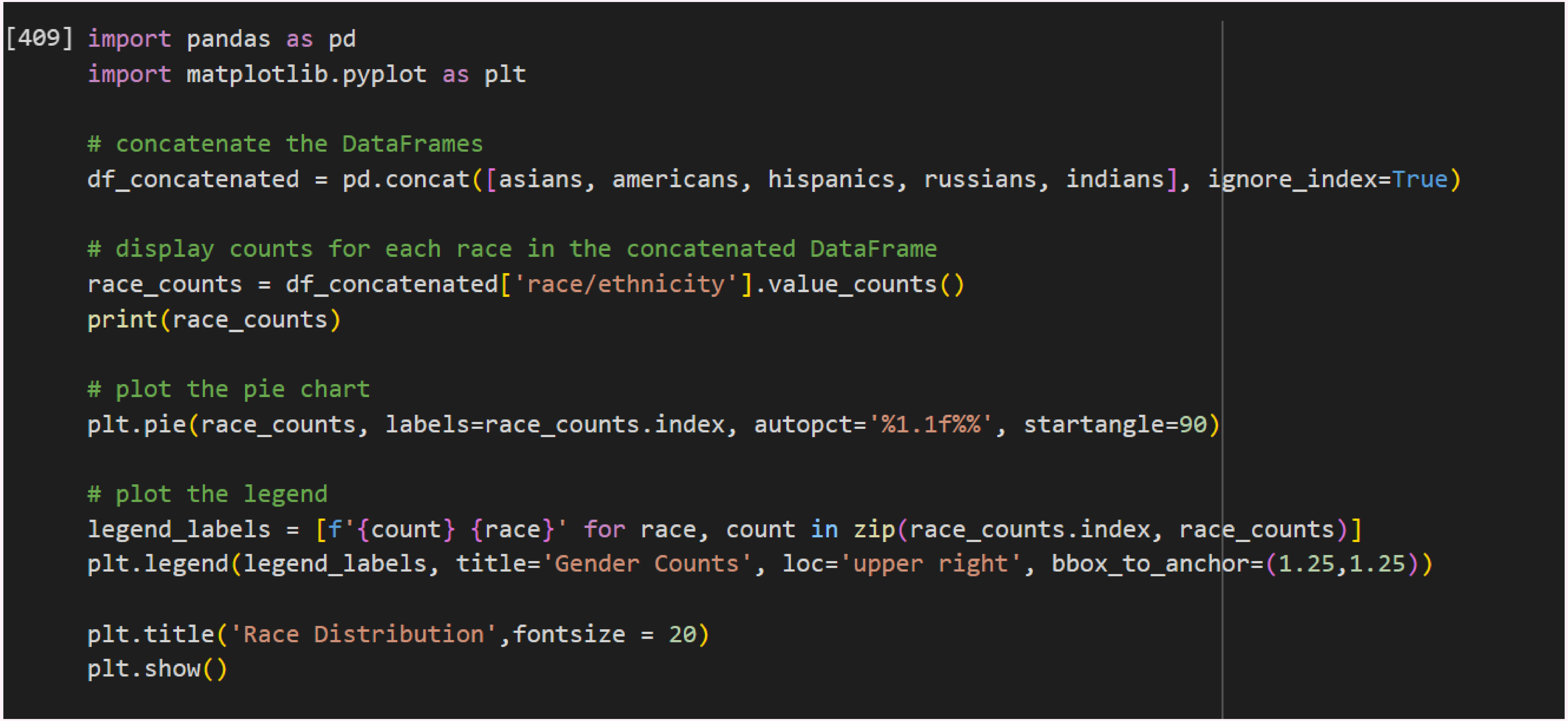


STUDENT EDUCATION

- BACHELOR’S DEGREE - 118
- MASTER’S DEGREE- 59
- ASSOCIATE’S DEGREE- 222
- HIGH SCHOOL- 196
- SOME COLLEGE - 226
- SOME HIGH SCHOOL - 179



PIE GRAPH CODE



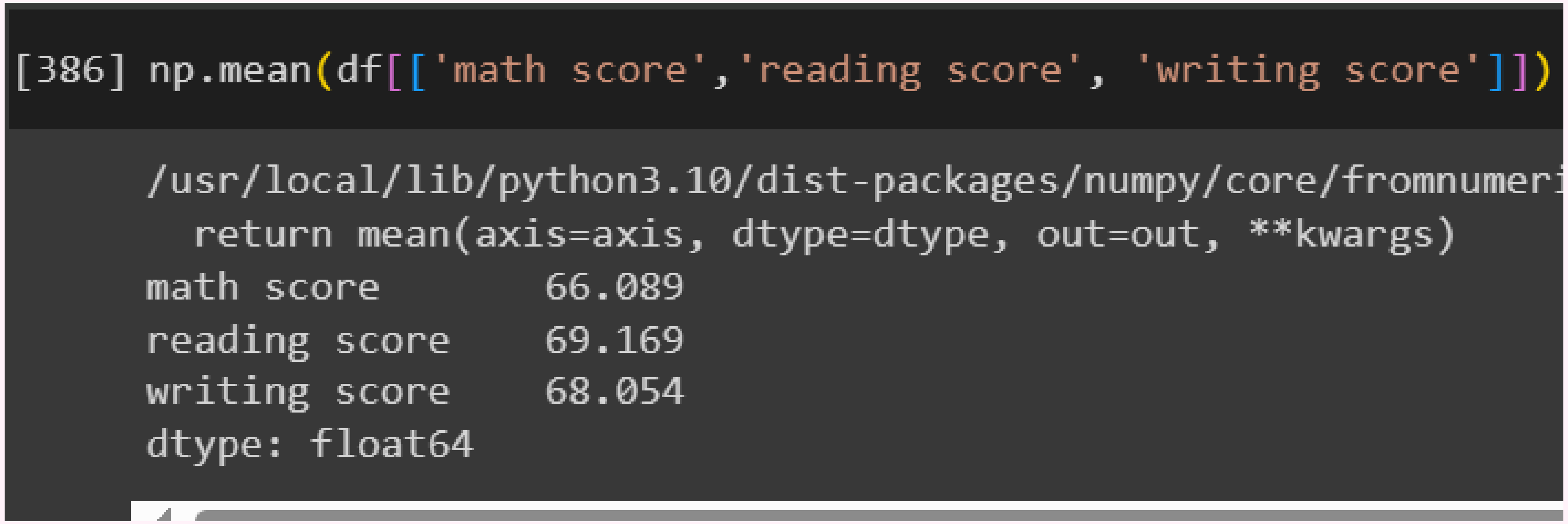
```
[409] import pandas as pd
      import matplotlib.pyplot as plt

      # concatenate the DataFrames
      df_concatenated = pd.concat([asians, americans, hispanics, russians, indians], ignore_index=True)

      # display counts for each race in the concatenated DataFrame
      race_counts = df_concatenated['race/ethnicity'].value_counts()
      print(race_counts)

      # plot the pie chart
      plt.pie(race_counts, labels=race_counts.index, autopct='%1.1f%%', startangle=90)

      # plot the legend
      legend_labels = [f'{count} {race}' for race, count in zip(race_counts.index, race_counts)]
      plt.legend(legend_labels, title='Gender Counts', loc='upper right', bbox_to_anchor=(1.25,1.25))

      plt.title('Race Distribution',fontsize = 20)
      plt.show()
```

WHY IS FREQUENCY DISTRIBUTION IMPORTANT?

MEAN

- MATH SCORE - 66.089
- READING SCORE - 69.169
- WRITING SCORE - 68.054

CODE

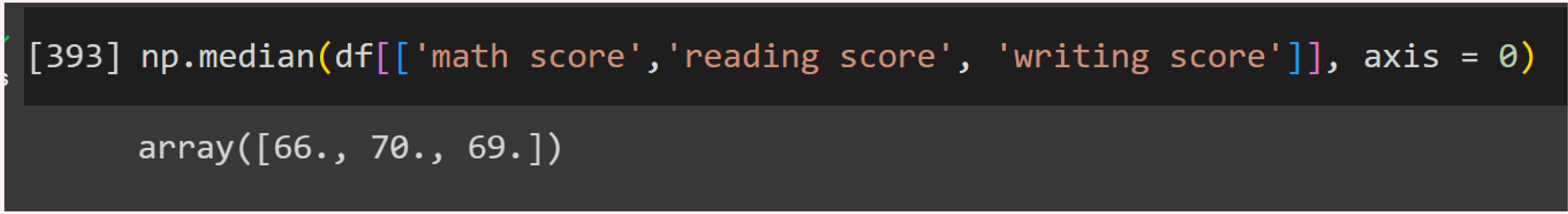


```
[386] np.mean(df[['math score','reading score', 'writing score']])

      /usr/local/lib/python3.10/dist-packages/numpy/core/fromnumeri
        return mean(axis=axis, dtype=dtype, out=out, **kwargs)
      math score       66.089
      reading score    69.169
      writing score    68.054
      dtype: float64
```

MEDIAN

- **MATH SCORE - 66**
- **READING SCORE - 70**
- **WRITING SCORE - 69**

CODE

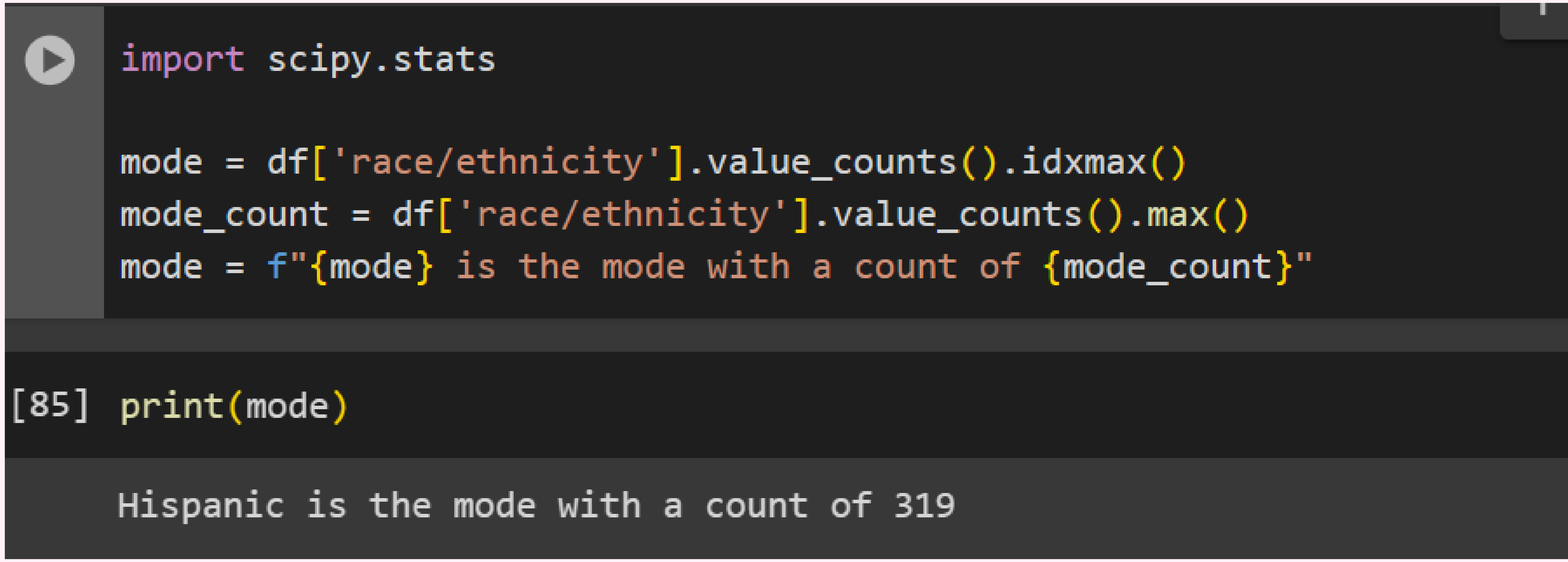


```
[393] np.median(df[['math score','reading score', 'writing score']], axis = 0)

      array([66., 70., 69.])
```

MODE

- AMERICAN - 89
- ASIAN - 190
- HISPANIC - 319
- RUSSIAN - 262
- INDIAN - 140

CODE



```
import scipy.stats

mode = df['race/ethnicity'].value_counts().idxmax()
mode_count = df['race/ethnicity'].value_counts().max()
mode = f"{mode} is the mode with a count of {mode_count}"
```

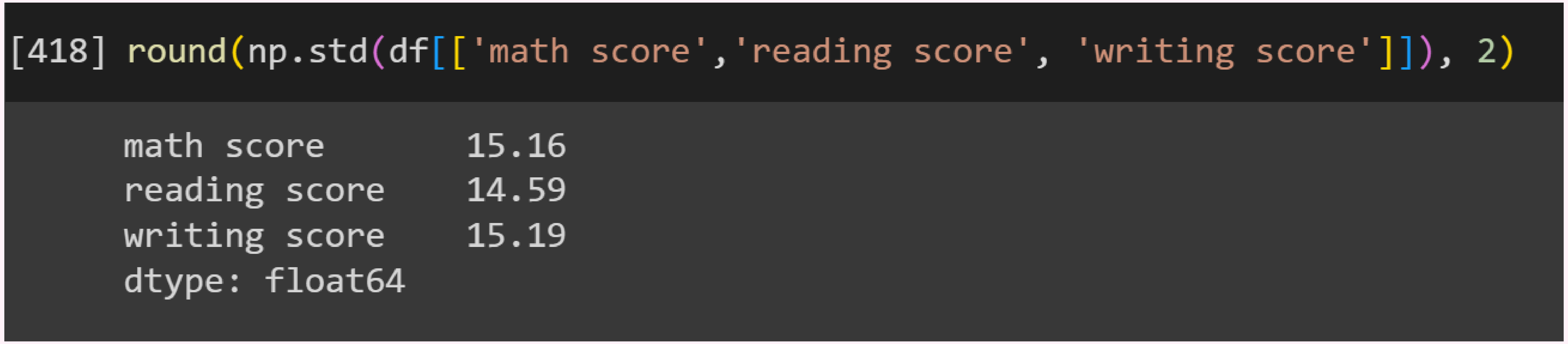
```
[85] print(mode)
```

```
Hispanic is the mode with a count of 319
```

STANDARD DEVIATION

- **MATH SCORE - 15.16**
- **READING SCORE - 14.59**
- **WRITING SCORE - 15.19**

CODE

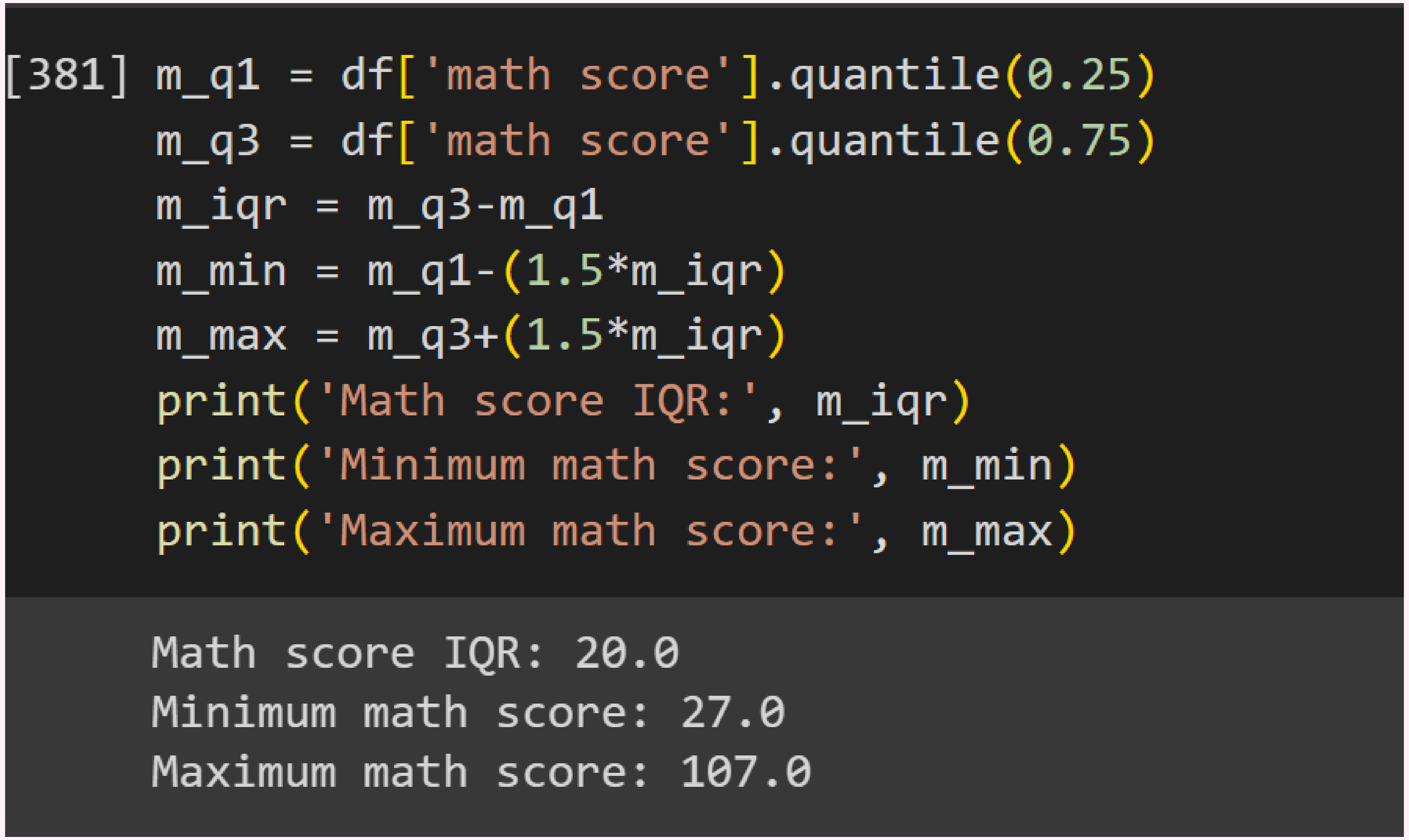


```
[418] round(np.std(df[['math score','reading score', 'writing score']]), 2)

      math score       15.16
      reading score    14.59
      writing score    15.19
      dtype: float64
```

IQR, MINIMUM, AND MAXIMUM

- **MATH SCORE IQR - 20.0**
- **MINIMUM MATH SCORE - 27.0**
- **MAXIMUM MATH SCORE - 107.0**

CODE



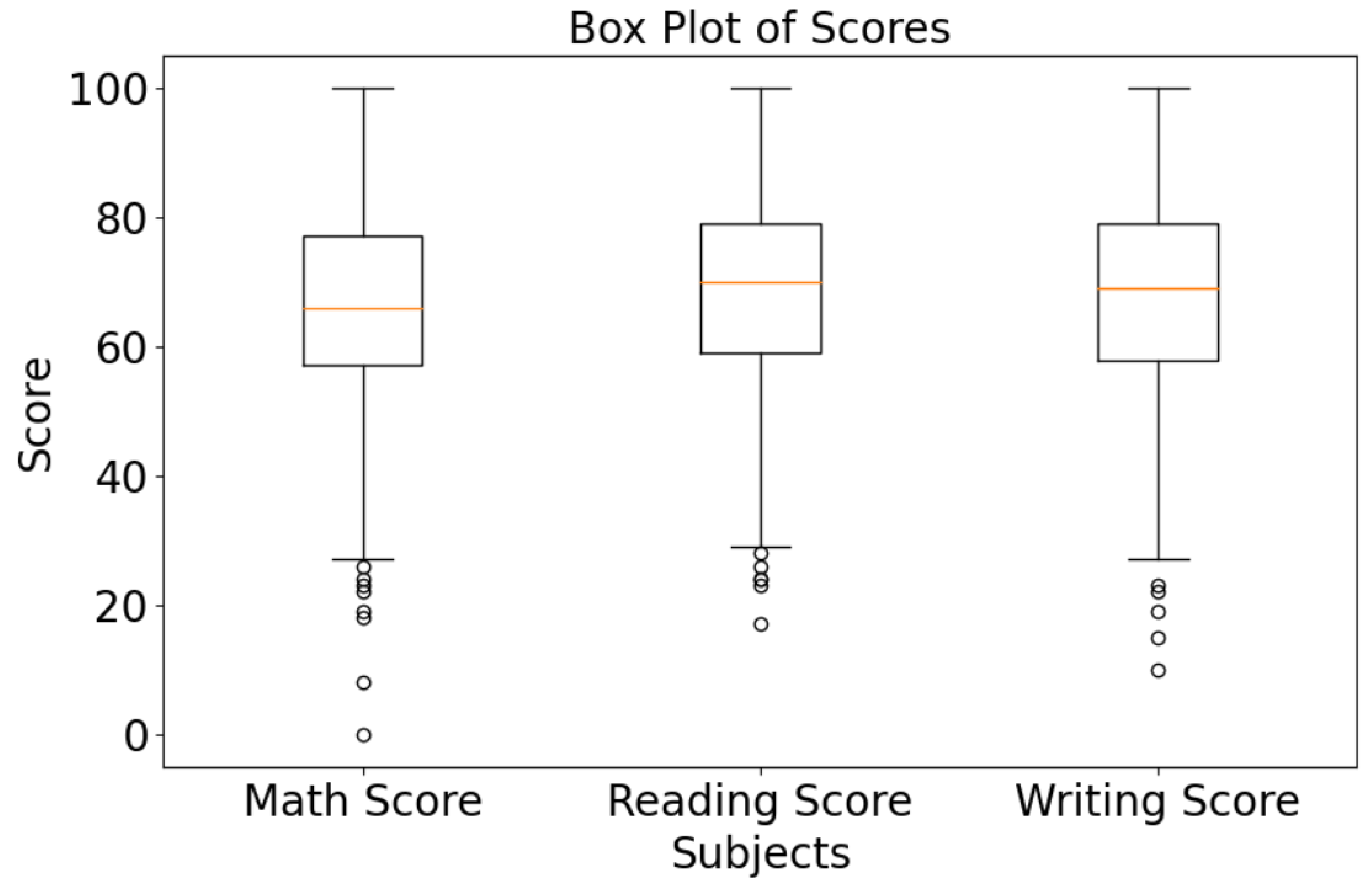
```
[381] m_q1 = df['math score'].quantile(0.25)
      m_q3 = df['math score'].quantile(0.75)
      m_iqr = m_q3-m_q1
      m_min = m_q1-(1.5*m_iqr)
      m_max = m_q3+(1.5*m_iqr)
      print('Math score IQR:', m_iqr)
      print('Minimum math score:', m_min)
      print('Maximum math score:', m_max)
```

```
Math score IQR: 20.0
Minimum math score: 27.0
Maximum math score: 107.0
```

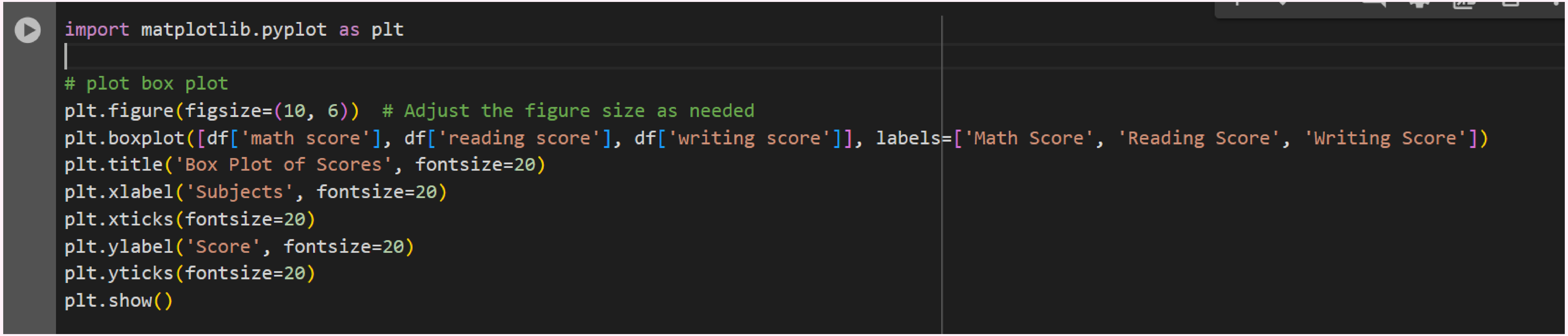
BOX PLOT

MATH SCORE

- **MEDIAN - 66**
- **IQR - 20**
- **1ST QUARTILE - 57**
- **3RD QUARTILE - 77**
- **MINIMUM - 27**
- **MAXIMUM - 107**



BOX PLOT CODE



```
import matplotlib.pyplot as plt

# plot box plot
plt.figure(figsize=(10, 6))  # Adjust the figure size as needed
plt.boxplot([df['math score'], df['reading score'], df['writing score']], labels=['Math Score', 'Reading Score', 'Writing Score'])
plt.title('Box Plot of Scores', fontsize=20)
plt.xlabel('Subjects', fontsize=20)
plt.xticks(fontsize=20)
plt.ylabel('Score', fontsize=20)
plt.yticks(fontsize=20)
plt.show()
```

THE END