Welcome to Colab!

```
import pandas as pd
print("Pandas library imported successfully.")
```

```
Pandas library imported successfully.
```

```
df = pd.read_csv('/content/StudentsPerformance[1] (1).csv')
print("Dataset loaded successfully.")
df.head()
```

```
Dataset loaded successfully.
```

```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 1000,\n  \"fields\":
[\n    {\n      \"column\": \"gender\",\n      \"properties\": {\n
\"dtype\": \"category\",\n        \"num_unique_values\": 2,\n
\"samples\": [\n          \"male\",\n          \"female\"\n        ],\
n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"race/ethnicity\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 5,\n        \"samples\": [\n          \"group
C\",\n          \"group E\"\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"parental level of education\",\n      \"properties\": {\
n        \"dtype\": \"category\",\n        \"num_unique_values\": 6,\n
\"samples\": [\n          \"bachelor's degree\",\n          \"some
college\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"lunch\",\n      \"properties\": {\n        \"dtype\": \"category\",\
n        \"num_unique_values\": 2,\n        \"samples\": [\n
\"free/reduced\",\n          \"standard\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"test preparation course\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 2,\n        \"samples\": [\n
\"completed\",\n          \"none\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"math score\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
15,\n        \"min\": 0,\n        \"max\": 100,\n
\"num_unique_values\": 81,\n        \"samples\": [\n          55,\n
72\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"reading score\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 14,\n        \"min\": 17,\n
\"max\": 100,\n        \"num_unique_values\": 72,\n
\"samples\": [\n          78,\n          23\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"writing score\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
```

```
15,\n        \"min\": 10,\n        \"max\": 100,\n
\"num_unique_values\": 77,\n        \"samples\": [\n          75,\n
76\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"df"}
```

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

print("\nColumn information:")
df.info()

print("\nDescriptive statistics:")
df.describe()
```

```
First few rows of the DataFrame:

Column information:
<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

Descriptive statistics:
```

```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 8,\n  \"fields\": [\n
{\n      \"column\": \"math score\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 335.8676421540409,\n
\"min\": 0.0,\n        \"max\": 1000.0,\n
\"num_unique_values\": 8,\n        \"samples\": [\n          66.089,\n
66.0,\n          1000.0\n        ],\n        \"semantic_type\": \"\",\
n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"reading score\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 334.2004716262942,\n
\"min\": 14.60019193725222,\n        \"max\": 1000.0,\n
\"num_unique_values\": 8,\n        \"samples\": [\n          69.169,\n
70.0,\n          1000.0\n        ],\n        \"semantic_type\": \"\",\
n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"writing score\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 334.8025670597152,\n
\"min\": 10.0,\n        \"max\": 1000.0,\n
\"num_unique_values\": 8,\n        \"samples\": [\n          68.054,\n
69.0,\n          1000.0\n        ],\n        \"semantic_type\": \"\",\
n        \"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe"}
```

```
from statsmodels.formula.api import ols

# Rename columns to remove spaces for easier use in formulas
df_renamed = df.rename(columns={'math score': 'math_score', 'reading
score': 'reading_score'})

# Define the linear regression model with the renamed columns
model = ols('math_score ~ reading_score', data=df_renamed)

# Fit the model to the data
results = model.fit()

# Print the summary of the regression results
print(results.summary())
```

```
                            OLS Regression Results

======================================================================
========
Dep. Variable:             math_score   R-squared:
0.668
Model:                            OLS   Adj. R-squared:
0.668
Method:                 Least Squares   F-statistic:
2012.
Date:                Thu, 13 Nov 2025   Prob (F-statistic):
1.79e-241
Time:                        07:51:17   Log-Likelihood:
-3585.3
No. Observations:                1000   AIC:
7175.
Df Residuals:                     998   BIC:
7184.
Df Model:                           1

Covariance Type:            nonrobust

======================================================================
===========
                    coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------
-----------
```

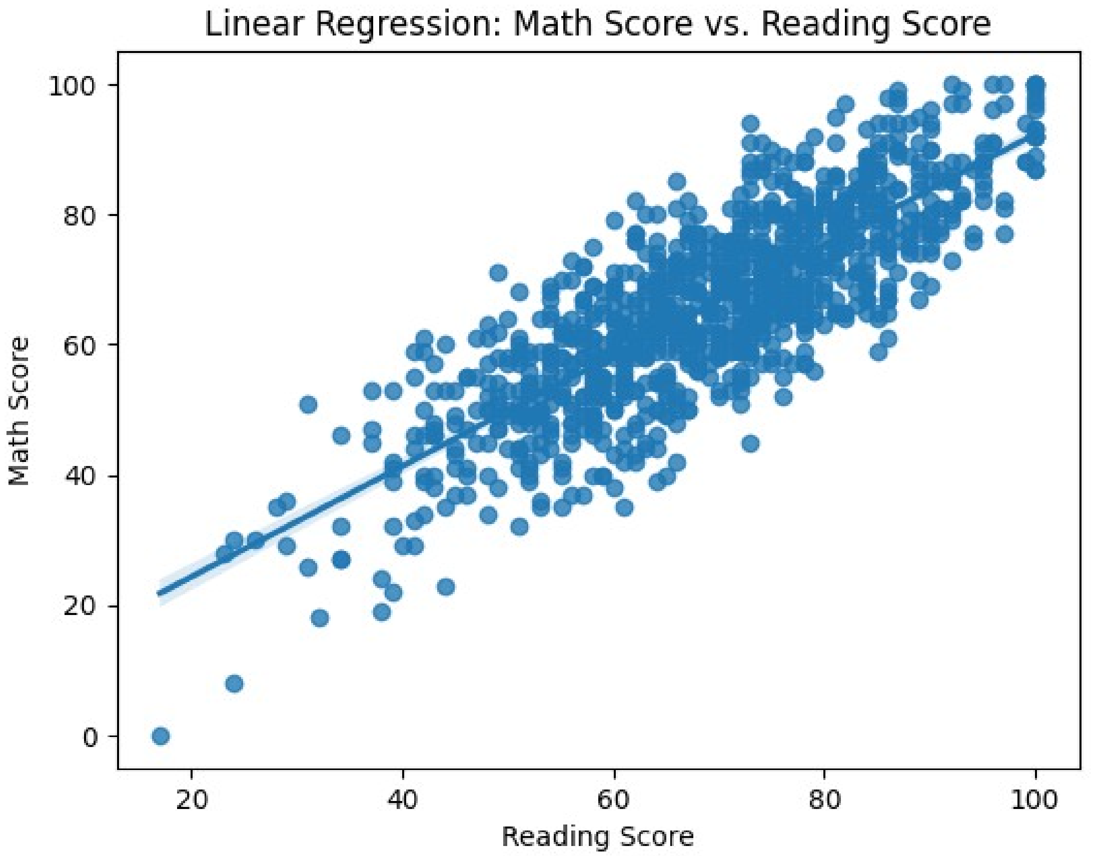
```
Intercept          7.3576      1.338      5.498      0.000       4.732
9.984
reading_score      0.8491      0.019     44.855      0.000       0.812
0.886
======================================================================
========
Omnibus:                        4.508   Durbin-Watson:
2.084
Prob(Omnibus):                  0.105   Jarque-Bera (JB):
3.578
Skew:                          -0.010   Prob(JB):
0.167
Kurtosis:                       2.708   Cond. No.
343.
======================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
```

```python
from statsmodels.formula.api import ols

# Rename columns to remove spaces for easier use in formulas
df_renamed = df.rename(columns={'math score': 'math_score', 'reading
score': 'reading_score'})

# Define the linear regression model with the renamed columns
model = ols('math_score ~ reading_score', data=df_renamed)

# Fit the model to the data
results = model.fit()

# Print the summary of the regression results
print(results.summary())
```

```
                            OLS Regression Results

======================================================================
========
Dep. Variable:             math_score   R-squared:
0.668
Model:                            OLS   Adj. R-squared:
0.668
Method:                 Least Squares   F-statistic:
2012.
Date:                Thu, 13 Nov 2025   Prob (F-statistic):
1.79e-241
Time:                        07:52:27   Log-Likelihood:
-3585.3
```

```
No. Observations:                1000   AIC:
7175.
Df Residuals:                     998   BIC:
7184.
Df Model:                           1

Covariance Type:            nonrobust

======================================================================
===========
                    coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------
-----------
Intercept         7.3576      1.338      5.498      0.000       4.732
9.984
reading_score     0.8491      0.019     44.855      0.000       0.812
0.886
======================================================================
========
Omnibus:                        4.508   Durbin-Watson:
2.084
Prob(Omnibus):                  0.105   Jarque-Bera (JB):
3.578
Skew:                          -0.010   Prob(JB):
0.167
Kurtosis:                       2.708   Cond. No.
343.
======================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
```

```python
import matplotlib.pyplot as plt
import seaborn as sns

# Create a scatter plot with the linear regression line
sns.regplot(x='reading_score', y='math_score', data=df_renamed)

# Add title and labels
plt.title('Linear Regression: Math Score vs. Reading Score')
plt.xlabel('Reading Score')
plt.ylabel('Math Score')

# Display the plot
plt.show()
```

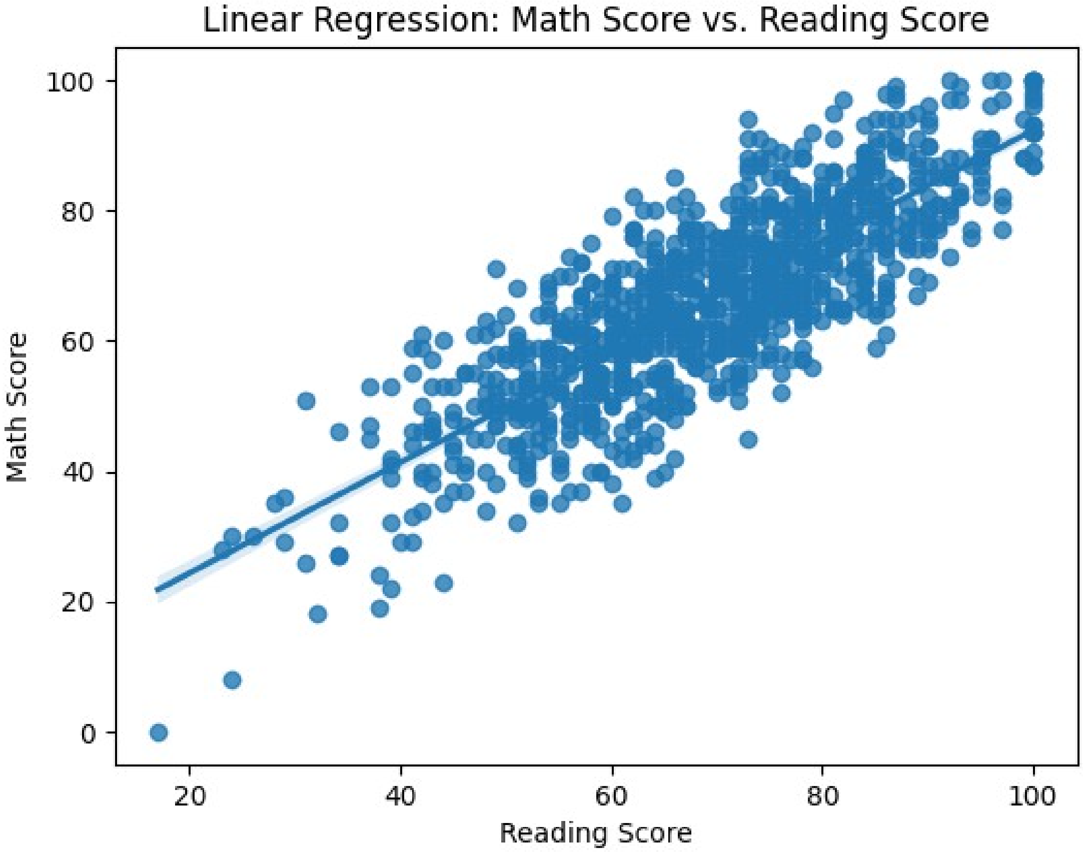


```
from statsmodels.formula.api import ols

# Rename columns to remove spaces for easier use in formulas
df_renamed = df.rename(columns={'math score': 'math_score', 'reading
score': 'reading_score'})

# Define the linear regression model with the renamed columns
model = ols('math_score ~ reading_score', data=df_renamed)

# Fit the model to the data
results = model.fit()

# Print the summary of the regression results
print(results.summary())
```

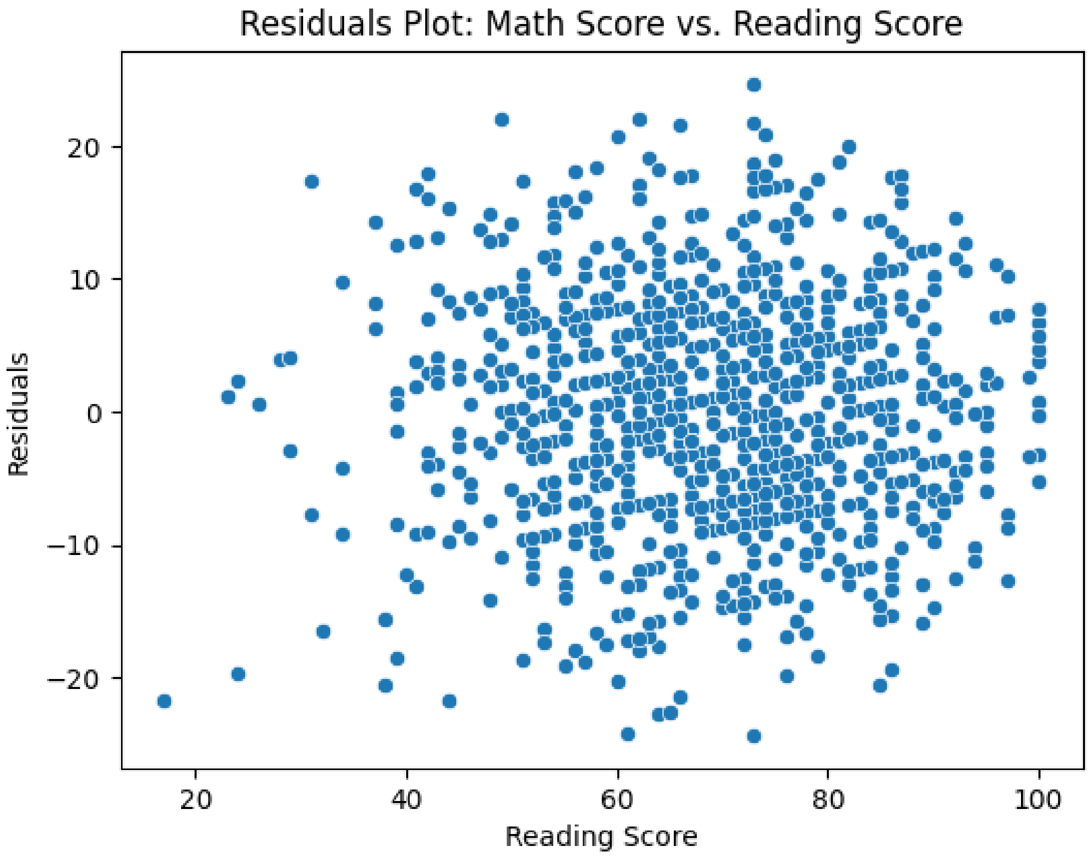
```
                            OLS Regression Results                            
                                                                              
======================================================================
========
Dep. Variable:             math_score   R-squared:                       
0.668
Model:                            OLS   Adj. R-squared:                  
0.668
```

```
Method:                 Least Squares   F-statistic:
2012.
Date:                Thu, 13 Nov 2025   Prob (F-statistic):
1.79e-241
Time:                        07:52:35   Log-Likelihood:
-3585.3
No. Observations:                1000   AIC:
7175.
Df Residuals:                     998   BIC:
7184.
Df Model:                           1

Covariance Type:            nonrobust

======================================================================
===========
                    coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------
-----------
Intercept         7.3576      1.338      5.498      0.000       4.732
9.984
reading_score     0.8491      0.019     44.855      0.000       0.812
0.886
======================================================================
========
Omnibus:                        4.508   Durbin-Watson:
2.084
Prob(Omnibus):                  0.105   Jarque-Bera (JB):
3.578
Skew:                          -0.010   Prob(JB):
0.167
Kurtosis:                       2.708   Cond. No.
343.
======================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
```

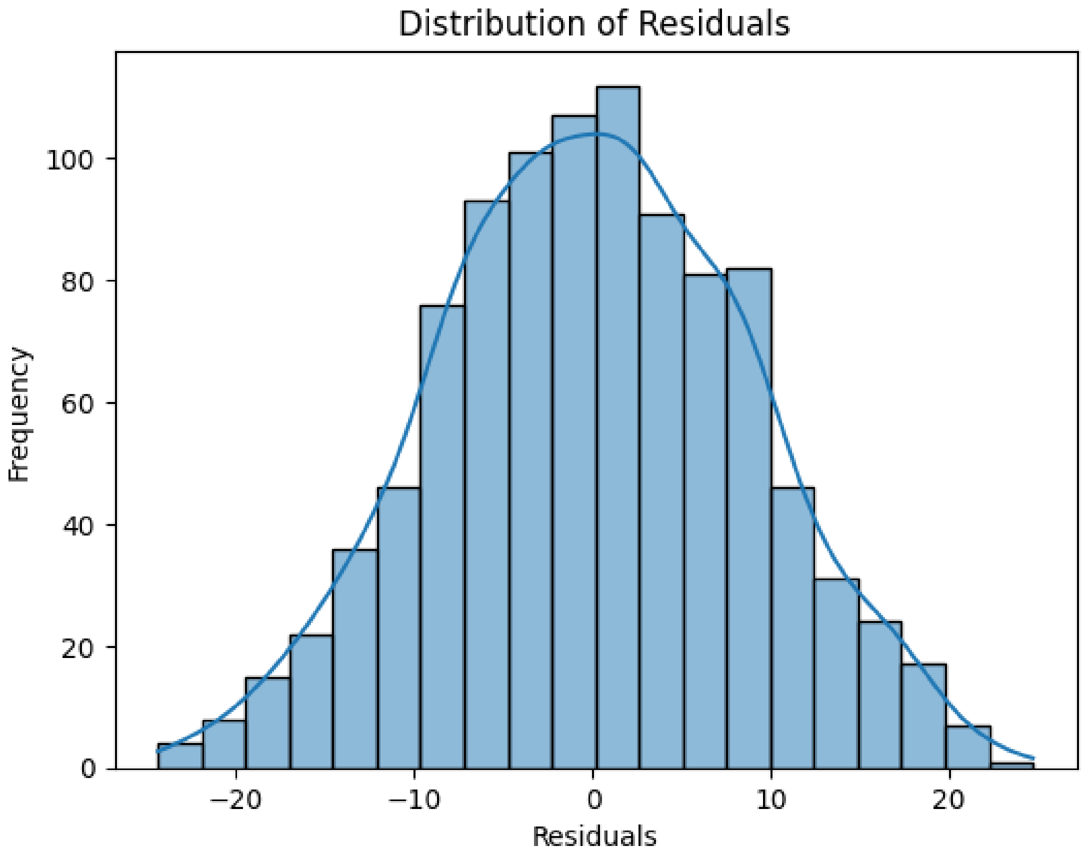
```python
import matplotlib.pyplot as plt
import seaborn as sns

# Create a scatter plot with the linear regression line
sns.regplot(x='reading_score', y='math_score', data=df_renamed)

# Add title and labels
plt.title('Linear Regression: Math Score vs. Reading Score')
plt.xlabel('Reading Score')
plt.ylabel('Math Score')

# Display the plot
plt.show()
```



```
import matplotlib.pyplot as plt
import seaborn as sns

# Get the residuals from the fitted model
residuals = results.resid

# Create a scatter plot of residuals against the independent variable
sns.scatterplot(x=df_renamed['reading_score'], y=residuals)

# Add title and labels
plt.title('Residuals Plot: Math Score vs. Reading Score')
plt.xlabel('Reading Score')
plt.ylabel('Residuals')

# Display the plot
plt.show()
```



```
import matplotlib.pyplot as plt
import seaborn as sns

# Create a histogram of the residuals
sns.histplot(residuals, kde=True)

# Add title and labels
plt.title('Distribution of Residuals')
plt.xlabel('Residuals')
plt.ylabel('Frequency')

# Display the plot
plt.show()
```


Distribution of Residuals
Frequency
0
20
40
60
80
100
−20
−10
0
10
20
Residuals