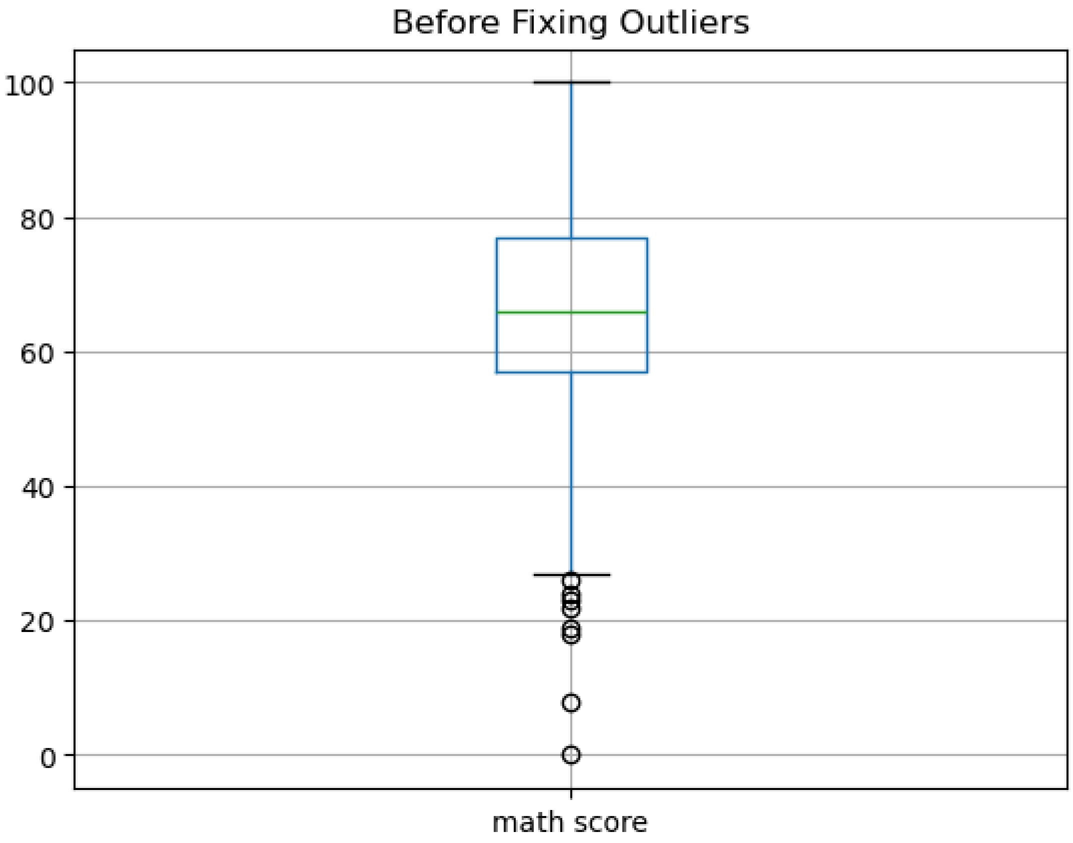
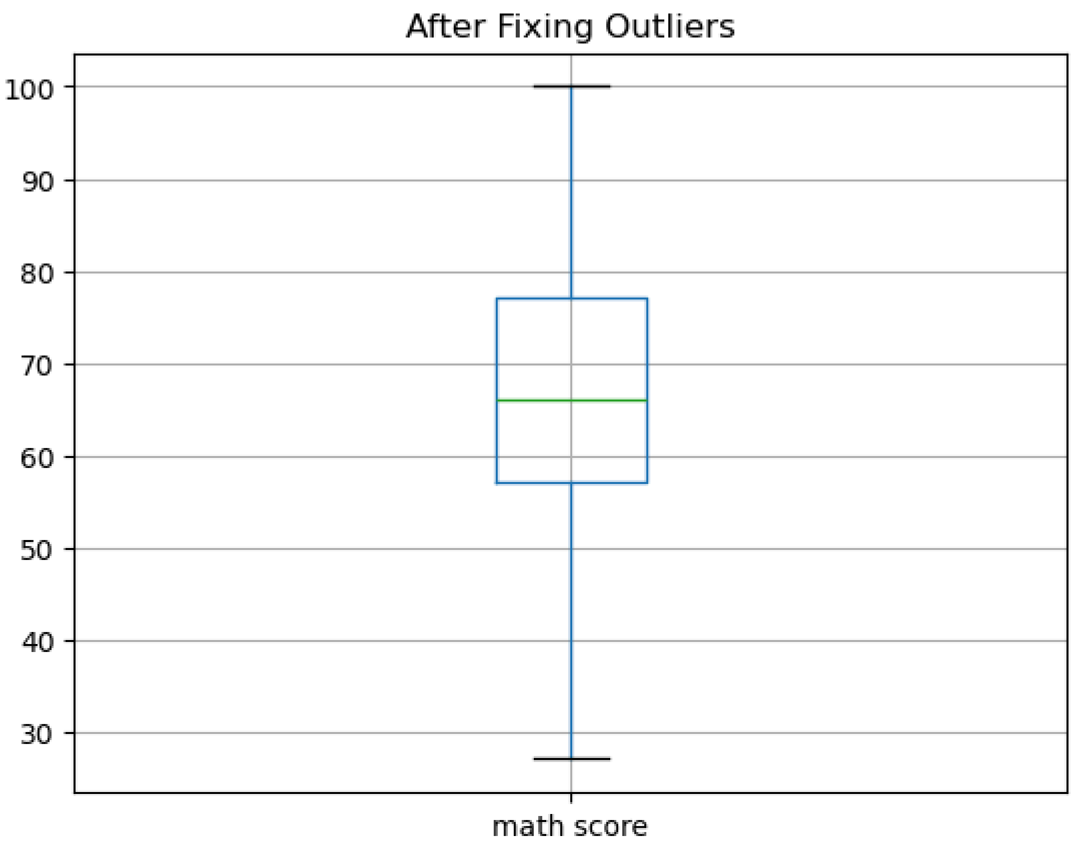
In [1]:

```
#Slip1 Q1
# Q1_DetectOutliers.py
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

stud = pd.read_csv("Student.csv")

# Boxplot before fixing
stud.boxplot(column=['math score'])
plt.title("Before Fixing Outliers")
plt.show()

Q1 = stud['math score'].quantile(0.25)
Q3 = stud['math score'].quantile(0.75)
IQR = Q3 - Q1
lower = Q1 - 1.5 * IQR
upper = Q3 + 1.5 * IQR

stud['math score'] = np.clip(stud['math score'], lower, upper)

# Boxplot after fixing
stud.boxplot(column=['math score'])
plt.title("After Fixing Outliers")
plt.show()
```





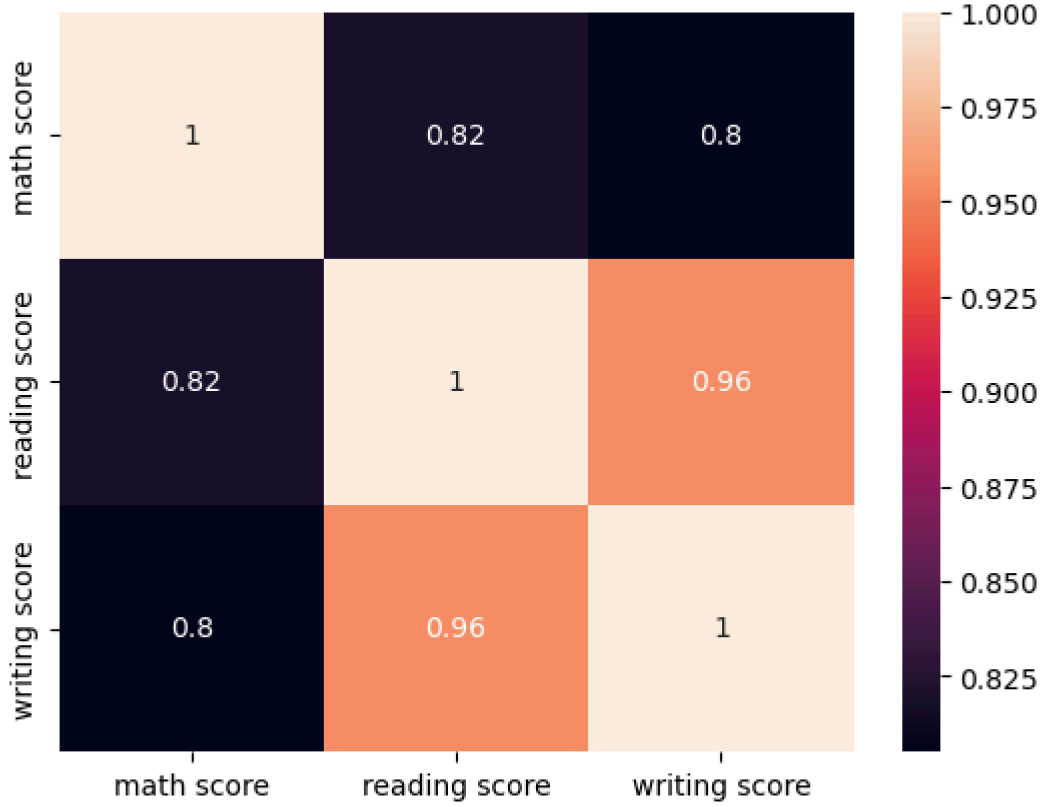
In [ ]:

```
#slip2 Q1
# Q1_MissingValues.py
import pandas as pd

loan = pd.read_csv("loan_data_set.csv")

loan['Gender'].fillna("Male", inplace=True)
loan['Married'].fillna("Yes", inplace=True)
loan['Dependents'].fillna(0, inplace=True)
loan['Self_Employed'].fillna("No", inplace=True)
loan['LoanAmount'].fillna(loan['LoanAmount'].median(), inplace=True)
loan['Loan_Amount_Term'].fillna(360.0, inplace=True)
loan['Credit_History'].fillna(1.0, inplace=True)

print("Missing values handled successfully.")
print(loan.isnull().sum())
```

In [5]:

```
#slip3 Q1
# Q1_Correlation.py
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

stud = pd.read_csv("Student.csv")

corr = stud.corr(numeric_only=True)
print(corr)

sns.heatmap(corr, annot=True)
plt.show()
```

```
               math score  reading score  writing score
math score       1.000000       0.818714       0.804942
reading score    0.818714       1.000000       0.955323
writing score    0.804942       0.955323       1.000000
```



In [7]:

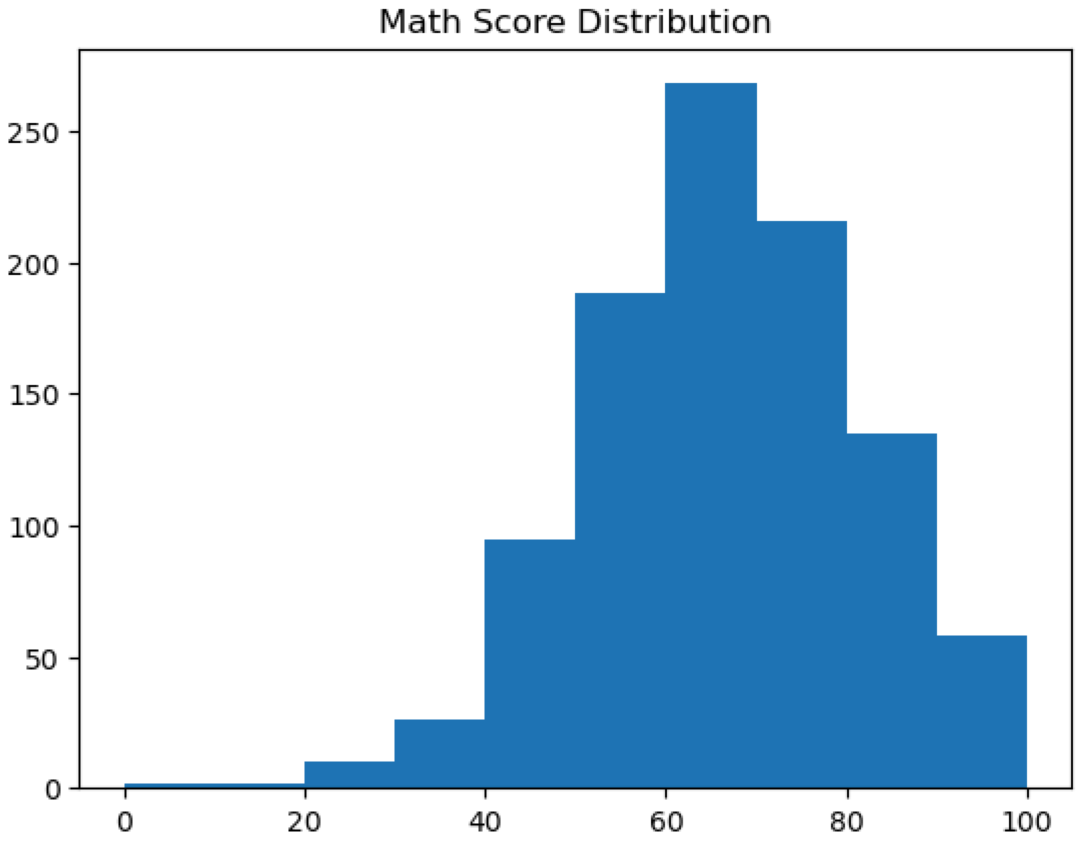
```
#slip4
# Q7_EDA.py
import pandas as pd

add = pd.read_csv("Addidas.csv")

print(add.head())
print(add.tail())
print(add.sample())
print(add.info())
print(add.describe())
print(add.isnull().sum())
```

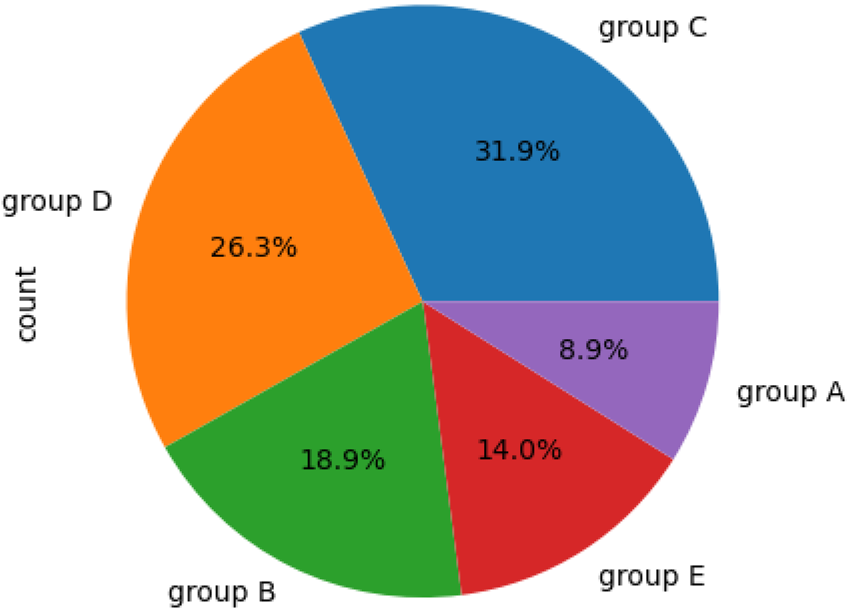
```
      Retailer  Retailer_ID Invoice_Date     Region     State      City  \
0  Foot Locker      1185732   01-01-2020  Northeast  New York  New York   
1  Foot Locker      1185732   02-01-2020  Northeast  New York  New York   
2  Foot Locker      1185732   03-01-2020  Northeast  New York  New York   
3  Foot Locker      1185732   04-01-2020  Northeast  New York  New York   
4  Foot Locker      1185732   05-01-2020  Northeast  New York  New York   

                    Product  Price_per_Unit  Units_Sold  Total_Sales  \
0      Men's Street Footwear            50.0        1200       600000   
1    Men's Athletic Footwear            50.0        1000       500000   
2    Women's Street Footwear            40.0        1000       400000   
3  Women's Athletic Footwear            45.0         850       382500   
4              Men's Apparel            60.0         900       540000   

   Operating_Profit  Operating_Margin Sales_Method  
0            300000              0.50     In-store  
1            150000              0.30     In-store  
2            140000              0.35     In-store  
3            133875              0.35     In-store  
4            162000              0.30     In-store  
         Retailer  Retailer_ID Invoice_Date     Region          State  \
9643  Foot Locker      1185732   24-01-2021  Northeast  New Hampshire   
9644  Foot Locker      1185732   24-01-2021  Northeast  New Hampshire   
9645  Foot Locker      1185732   22-02-2021  Northeast  New Hampshire   
9646  Foot Locker      1185732   22-02-2021  Northeast  New Hampshire   
9647  Foot Locker      1185732   22-02-2021  Northeast  New Hampshire   

            City                  Product  Price_per_Unit  Units_Sold  \
9643  Manchester            Men's Apparel             NaN          64   
9644  Manchester          Women's Apparel            41.0         105   
9645  Manchester    Men's Street Footwear            41.0         184   
9646  Manchester  Men's Athletic Footwear            42.0          70   
9647  Manchester  Women's Street Footwear            29.0          83   

      Total_Sales  Operating_Profit  Operating_Margin Sales_Method  
9643         3200               896              0.28       Outlet  
9644         4305              1378              0.32       Outlet  
9645         7544              2791              0.37       Outlet  
9646         2940              1235              0.42       Outlet  
9647         2407               650              0.27       Outlet  
     Retailer  Retailer_ID Invoice_Date Region     State         City  \
6581  Walmart      1197831   19-07-2021  South  Arkansas  Little Rock   

                     Product  Price_per_Unit  Units_Sold  Total_Sales  \
6581  Women's Street Footwear            43.0         126         5418   

      Operating_Profit  Operating_Margin Sales_Method  
6581              2763              0.51       Online  
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9648 entries, 0 to 9647
Data columns (total 13 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Retailer          9648 non-null   object 
 1   Retailer_ID       9648 non-null   int64  
 2   Invoice_Date      9648 non-null   object 
 3   Region            9648 non-null   object 
 4   State             9648 non-null   object 
 5   City              9648 non-null   object 
 6   Product           9645 non-null   object 
 7   Price_per_Unit    9643 non-null   float64
 8   Units_Sold        9648 non-null   int64
 9   Total_Sales       9648 non-null   int64
 10  Operating_Profit  9648 non-null   int64
 11  Operating_Margin  9648 non-null   float64
 12  Sales_Method      9648 non-null   object
dtypes: float64(2), int64(4), object(7)
memory usage: 980.0+ KB
None
        Retailer_ID  Price_per_Unit   Units_Sold    Total_Sales  \
count  9.648000e+03     9643.000000  9648.000000    9648.000000
mean   1.173850e+06       45.213419   256.930037   93273.437500
std    2.636038e+04       14.707649   214.252030  141916.016727
min    1.128299e+06        7.000000     0.000000       0.000000
25%    1.185732e+06       35.000000   106.000000    4254.500000
50%    1.185732e+06       45.000000   176.000000    9576.000000
75%    1.185732e+06       55.000000   350.000000  150000.000000
max    1.197831e+06      110.000000  1275.000000  825000.000000

       Operating_Profit  Operating_Margin
count       9648.000000       9648.000000
mean       34425.282131          0.422991
std        54193.124141          0.097197
min            0.000000          0.100000
25%         1922.000000          0.350000
50%         4371.500000          0.410000
75%        52063.000000          0.490000
max       390000.000000          0.800000
Retailer            0
Retailer_ID         0
Invoice_Date        0
Region              0
State               0
City                0
Product             3
Price_per_Unit      5
Units_Sold          0
Total_Sales         0
Operating_Profit    0
Operating_Margin    0
Sales_Method        0
dtype: int64
```

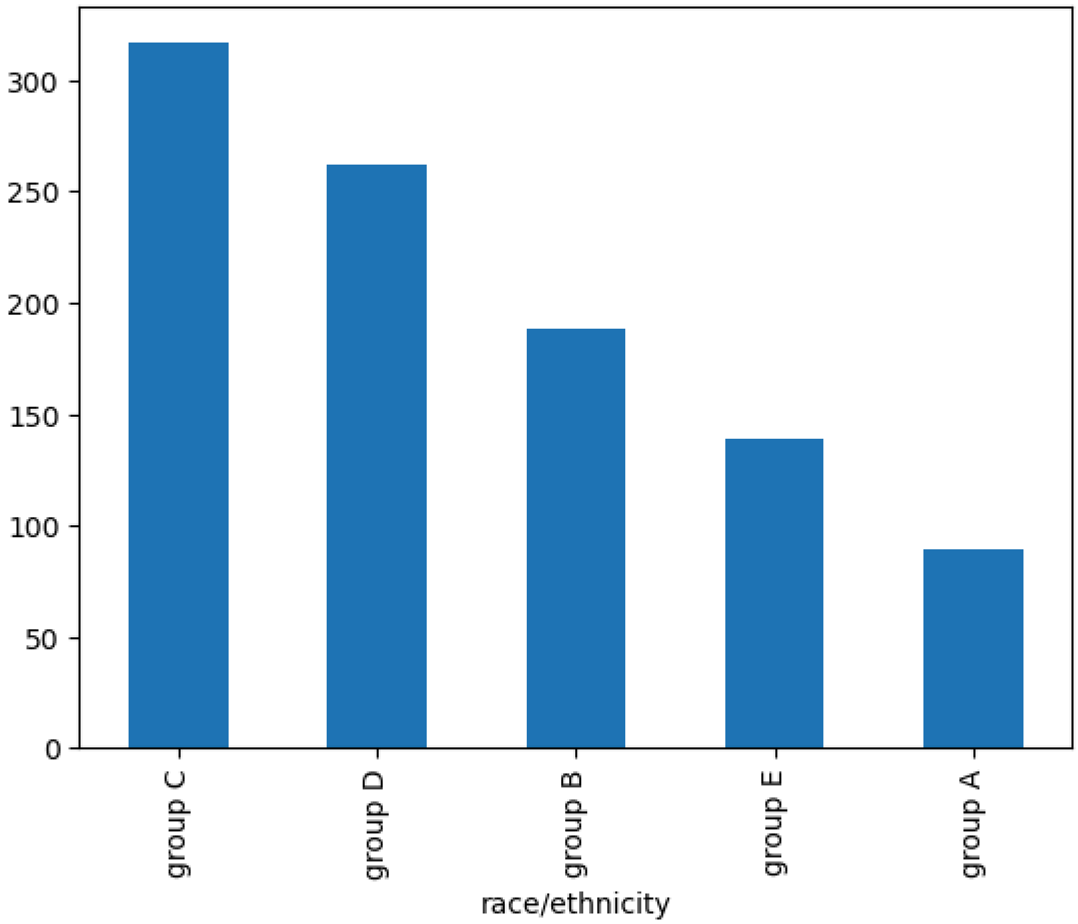
In [8]:

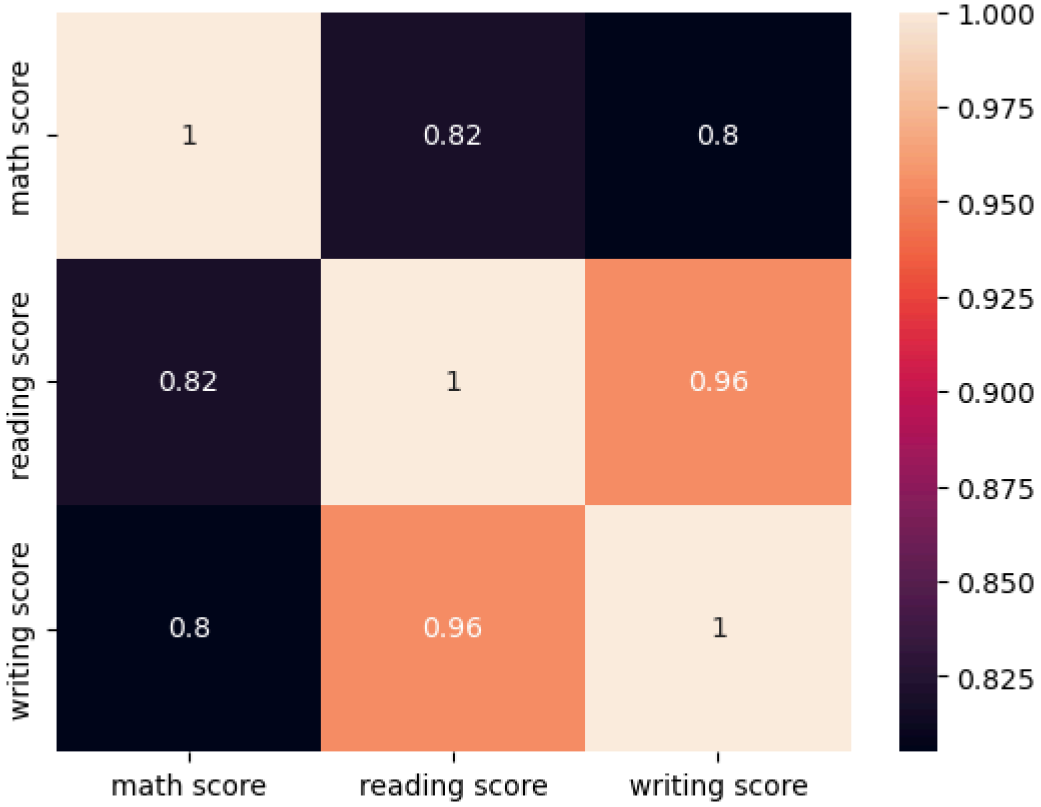
```python
#slip5
# Q8_Visualization.py
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

stud = pd.read_csv("Student.csv")

# Histogram
plt.hist(stud['math score'])
plt.title("Math Score Distribution")
plt.show()

# Pie Chart
stud['race/ethnicity'].value_counts().plot(kind='pie', autopct='%1.1f%%')
plt.show()
```

```
# Bar Graph
stud['race/ethnicity'].value_counts().plot(kind='bar')
plt.show()

# Heatmap
sns.heatmap(stud.corr(numeric_only=True), annot=True)
plt.show()
```


Math Score Distribution





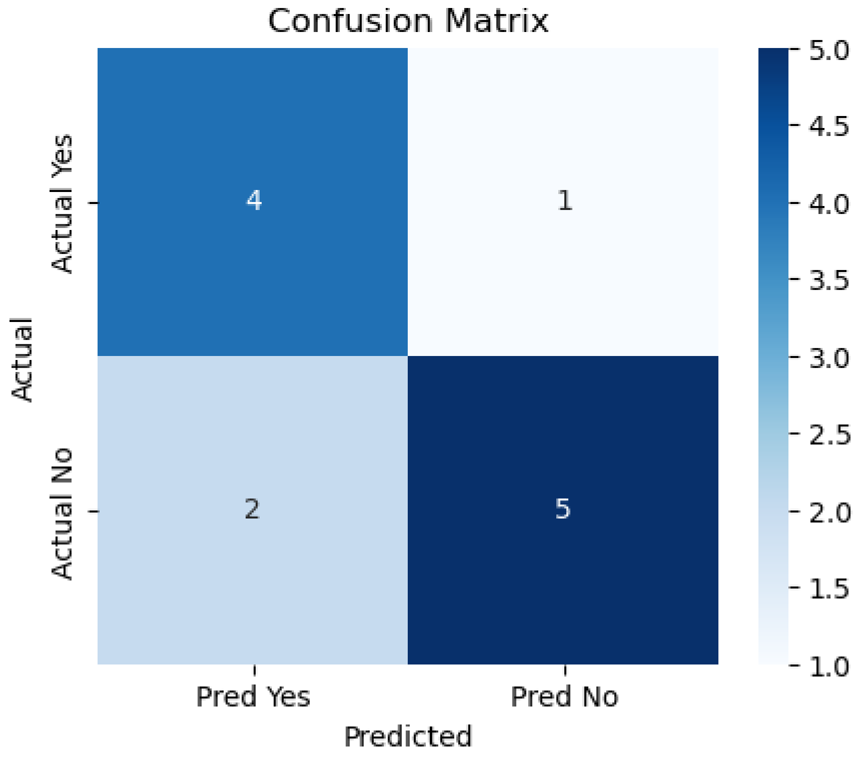


In [13]:

```
#slip6

import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
```

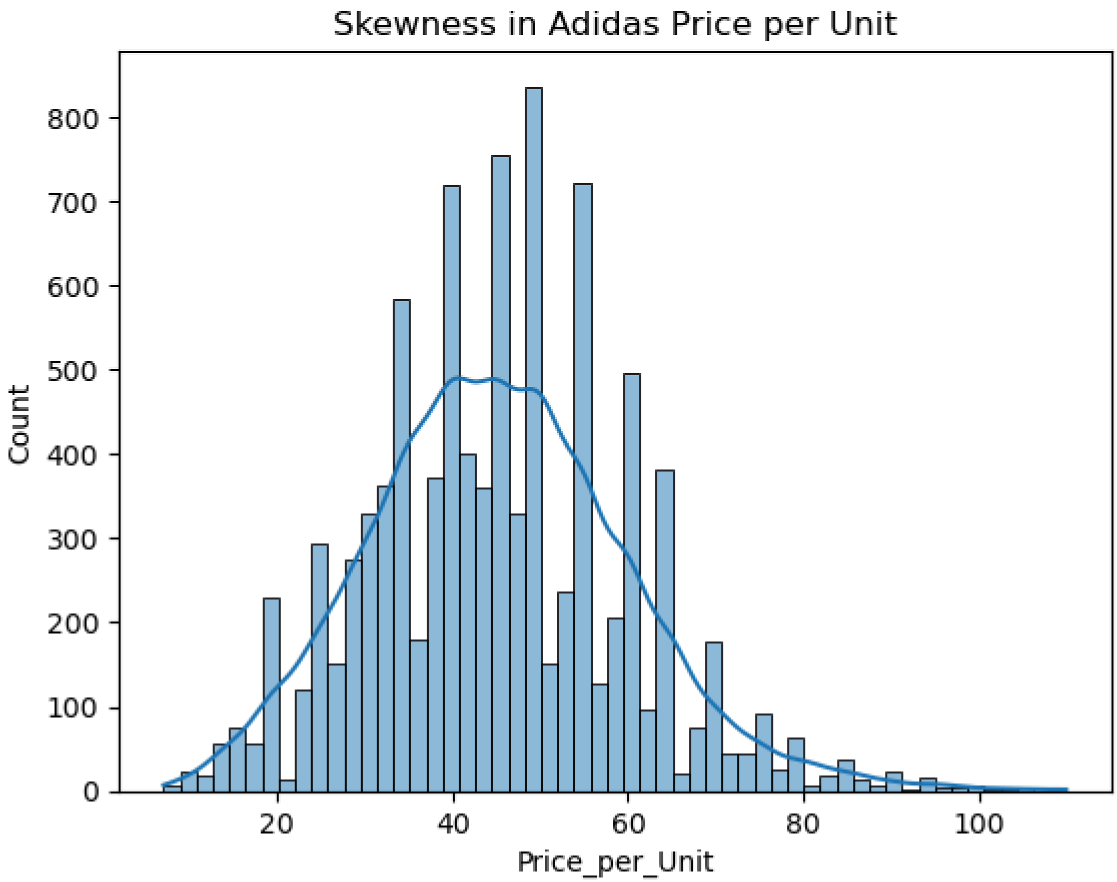
```
data = {
    'y_actual': ["Yes","No","No","Yes","No","Yes","No","No","Yes","No","Yes","No
    'y_predic': ["Yes","Yes","No","Yes","No","Yes","Yes","No","Yes","No","No","N
}

y_actual = data['y_actual']
y_pred = data['y_predic']

# Confusion Matrix
cm = confusion_matrix(y_actual, y_pred, labels=["Yes","No"])
print("Confusion Matrix:\n", cm)

# Heatmap Visualization
plt.figure(figsize=(5,4))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=["Pred Yes","Pred No"],
            yticklabels=["Actual Yes","Actual No"])
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()


#True Positives (TP = 4): Predicted Yes and actually Yes

#False Negatives (FN = 1): Predicted No but actually Yes

#False Positives (FP = 1): Predicted Yes but actually No

#True Negatives (TN = 6): Predicted No and actually No
```

```
Confusion Matrix:
 [[4 1]
 [2 5]]
```



In [16]:

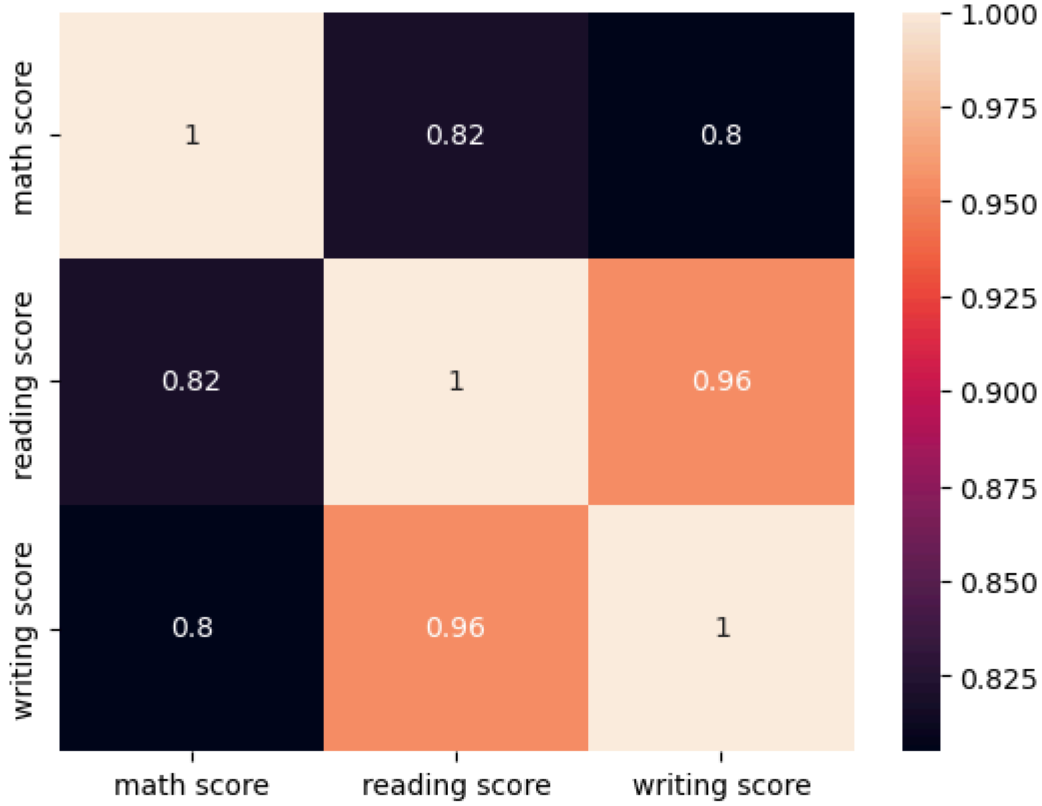
```
#slip7
# Q11_Skewness.py
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

add = pd.read_csv("Addidas.csv")

sns.histplot(add['Price_per_Unit'], kde=True)
plt.title("Skewness in Adidas Price per Unit")
plt.show()
```



In [15]:

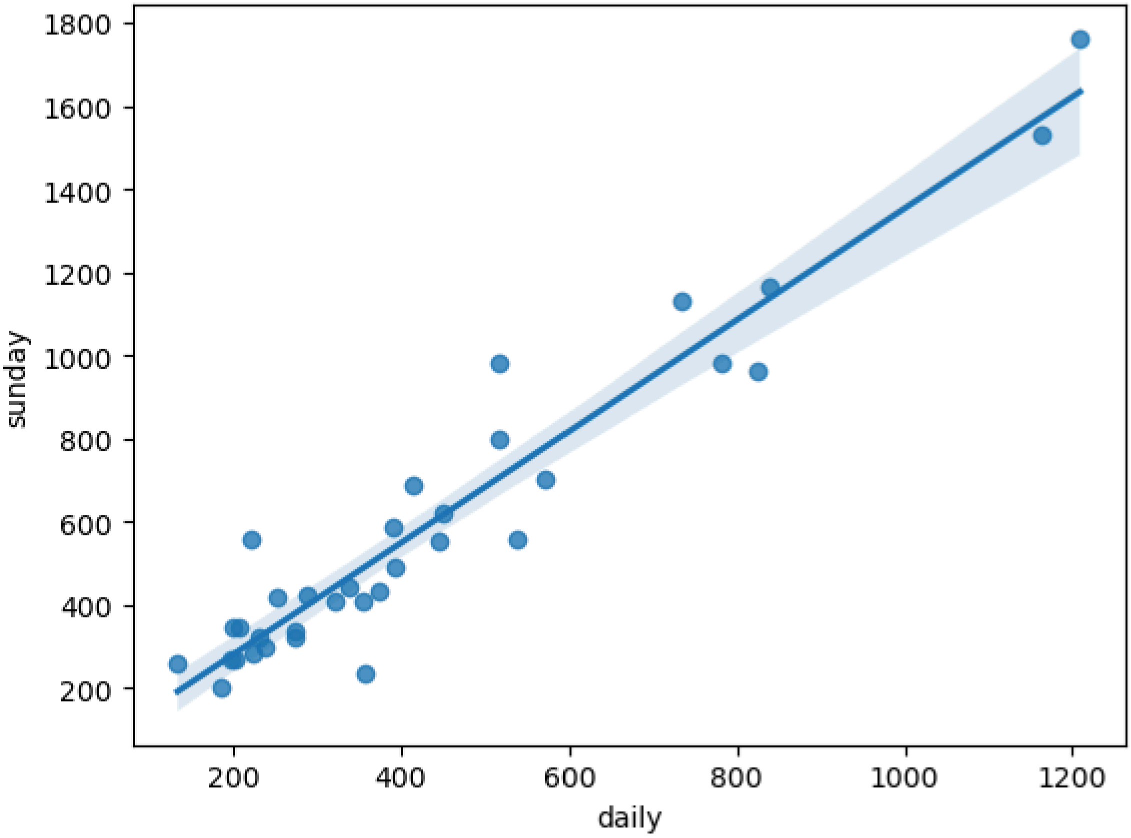
```
#slip8
# Q5_Correlation.py
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

stud = pd.read_csv("Student.csv")

corr = stud.corr(numeric_only=True)
print(corr)

sns.heatmap(corr, annot=True)
plt.show()
```

```
               math score  reading score  writing score
math score       1.000000       0.818714       0.804942
reading score    0.818714       1.000000       0.955323
writing score    0.804942       0.955323       1.000000
```



In [17]:

```
#slip9
# Q13_Regplot.py
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

news = pd.read_csv("NewspaperData.csv")

sns.regplot(x="daily", y="sunday", data=news)
plt.show()
```



In [18]:

```
#slip10
# Q14_TrainTestSplit.py
import pandas as pd
from sklearn.model_selection import train_test_split

news = pd.read_csv("NewspaperData.csv")

x = news[['daily']]
y = news[['sunday']]

xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size=0.2)
print("X-train:\n", xtrain.head())
print("X-test:\n", xtest.head())
```

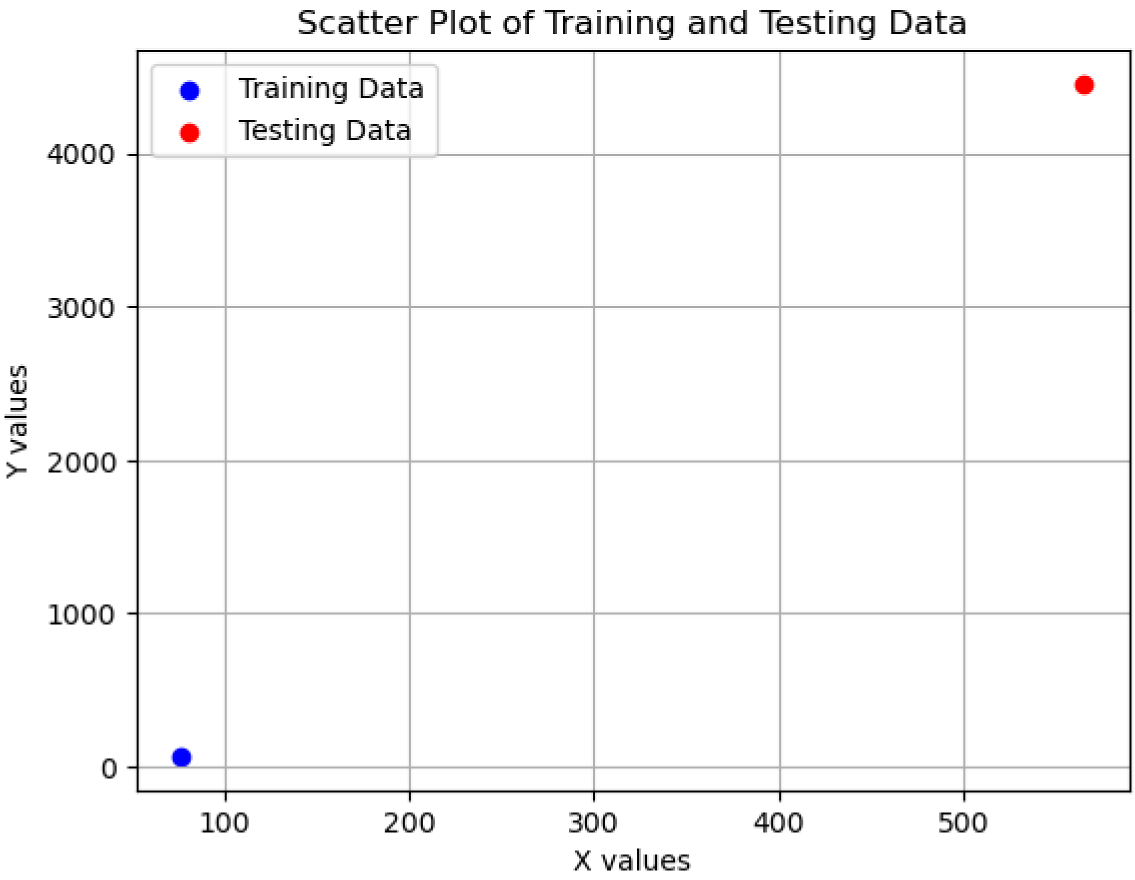
```
X-train:
        daily
20  223.748
10  449.755
15  412.871
24  337.672
14  444.581
X-test:
        daily
30  391.286
22  515.523
8   206.204
4   537.780
3   238.555
```

In [20]:

```
#slip 11
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
n = int(input("Enter number of data points: "))
```

```python
X = []
Y = []

# Input X and Y values
for i in range(n):
    print(f"\nData point {i+1}:")
    x = float(input("Enter X value: "))
    y = float(input("Enter Y value: "))
    X.append(x)
    Y.append(y)
    # Split the data into training and testing sets (80% training, 20% testing)
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_

# Plot the data using scatter plot
plt.scatter(X_train, Y_train, color='blue', label='Training Data')
plt.scatter(X_test, Y_test, color='red', label='Testing Data')
plt.xlabel("X values")
plt.ylabel("Y values")
plt.title("Scatter Plot of Training and Testing Data")
plt.legend()
plt.grid(True)
plt.show()
```

```
Data point 1:
Data point 2:
```



In [29]:

```python
#Slip12
# Q4_ChangeDatatype.py
import pandas as pd

stud = pd.read_csv("Student.csv")

print("Before:\n", stud.dtypes)
```

```
stud['math score'] = stud['math score'].astype(float)
print("After:\n", stud.dtypes)
```

```
Before:
 gender                          object
race/ethnicity                  object
parental level of education     object
lunch                           object
test preparation course         object
math score                       int64
reading score                  float64
writing score                  float64
dtype: object
After:
 gender                          object
race/ethnicity                  object
parental level of education     object
lunch                           object
test preparation course         object
math score                     float64
reading score                  float64
writing score                  float64
dtype: object
```

In [22]:
```
#slip13
same as slip7
```

In [23]:
```
#slip14
same as slip3
```

In [24]:
```
#slip15
same as slip2
```

In [25]:
```
#slip16
same as slip11
```

In [26]:
```
#slip17
same as slip1
```

In [27]:
```
#slip18
same as slip3
```

In [28]:
```
#slip19
same as slip4
```

In [ ]:
```
#slip20
same as slip10
```