Group A

Assignment 2

Data Wrangling II

Import all the required Python Libraries.

```
# code here
import pandas as pd
import numpy as np
import random
import seaborn as sns
```

Create a DataFrame from the dictionary

```
# code here
data={"Stud_id":[i  for i in range(1,101)],
     "Gender":[random.choice(['Female','Male']) for _  in 
range(100) ],
     "Class":[random.choice(['SE','TE','BE']) for _  in range(100) ],
     "Attendance":[random.uniform(0,100) for _ in range(100)],
      "DSBDA":[random.randint(0,100) for _ in range(100)],
     "WT":[random.randint(0,100) for _ in range(100)],
     "AI":[random.randint(0,100) for _ in range(100)]}

data

df=pd.DataFrame(data)

df

    Stud_id  Gender Class  Attendance  DSBDA  WT  AI
0         1    Male    BE   30.239580     56  82   7
1         2  Female    SE   44.727962     63  49  34
2         3  Female    SE   53.430196     60  65  45
3         4    Male    BE   32.110630     91  69  98
4         5  Female    BE   84.151293     94  46  66
..      ...     ...   ...         ...    ...  ..  ..
95       96  Female    SE   82.940270     35  28  93
96       97    Male    TE   27.447840     88  82  49
97       98    Male    TE   93.815668     15  74  53
98       99  Female    TE    8.431059     94  81  71
99      100  Female    SE   62.632263     77  71  20

[100 rows x 7 columns]

df.to_csv('AcademicPerformance.csv')
```

Load the Dataset into pandas dataframe.

```
# code here
Std_df=pd.read_csv('StudentPerformance.csv')

Std_df.head()

  gender race/ethnicity parental level of education        lunch  \
0  female        group B           bachelor's degree      standard
1  female        group C                some college      standard
2  female        group B             master's degree      standard
3    male        group A          associate's degree  free/reduced
4    male        group C                some college      standard

 test preparation course  math score  reading score  writing score
0                    none        72.0           72.0           74.0
1               completed        69.0           90.0           88.0
2                    none        90.0           95.0           93.0
3                    none        47.0           57.0           44.0
4                    none        76.0           78.0           75.0

Std_df.shape

(1000, 8)

Std_df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 8 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   gender                       985 non-null    object
 1   race/ethnicity               989 non-null    object
 2   parental level of education  993 non-null    object
 3   lunch                        999 non-null    object
 4   test preparation course      990 non-null    object
 5   math score                   990 non-null    float64
 6   reading score                990 non-null    float64
 7   writing score                990 non-null    float64
dtypes: float64(3), object(5)
memory usage: 62.6+ KB

Std_df.describe()

       math score  reading score  writing score
count  990.000000     990.000000     990.000000
mean    66.055556      69.116162      68.082828
std     15.137922      14.594195      15.158456
min      0.000000      17.000000      10.000000
25%     57.000000      59.000000      58.000000
50%     66.000000      70.000000      69.000000
```

```
75%     77.000000      79.000000      79.000000
max    100.000000     100.000000     100.000000
```

Data Preprocessing

Scan all variables for missing values and inconsistencies. If there are missing values and/or inconsistencies, use any of the suitable techniques to deal with them.

```
# code here
Std_df.isnull().sum()
```

```
gender                          15
race/ethnicity                  11
parental level of education      7
lunch                            1
test preparation course         10
math score                      10
reading score                   10
writing score                   10
dtype: int64
```

```
Std_df.nunique()
```

```
gender                           2
race/ethnicity                   5
parental level of education      6
lunch                            2
test preparation course          2
math score                      81
reading score                   72
writing score                   77
dtype: int64
```

```
Std_df['gender'].value_counts()
```

```
gender
female    510
male      475
Name: count, dtype: int64
```

```
Std_df.gender.fillna('female',inplace=True)
```

```
Std_df.isnull().sum()
```

```
gender                           0
race/ethnicity                  11
parental level of education      7
lunch                            1
test preparation course         10
math score                      10
reading score                   10
```

```
writing score                    10
dtype: int64
```

```
Std_df['race/ethnicity'].value_counts()
```

```
race/ethnicity
group C    315
group D    259
group B    189
group E    139
group A     87
Name: count, dtype: int64
```

```
Std_df['race/ethnicity'].fillna('group C',inplace=True)
```

```
Std_df.isnull().sum()
```

```
gender                          0
race/ethnicity                  0
parental level of education     7
lunch                           1
test preparation course        10
math score                     10
reading score                  10
writing score                  10
dtype: int64
```

```
Std_df['parental level of education'].value_counts()
```

```
parental level of education
some college          224
associate's degree    221
high school           196
some high school      178
bachelor's degree     116
master's degree        58
Name: count, dtype: int64
```

```
Std_df['parental level of education'].fillna('some college
',inplace=True)
```

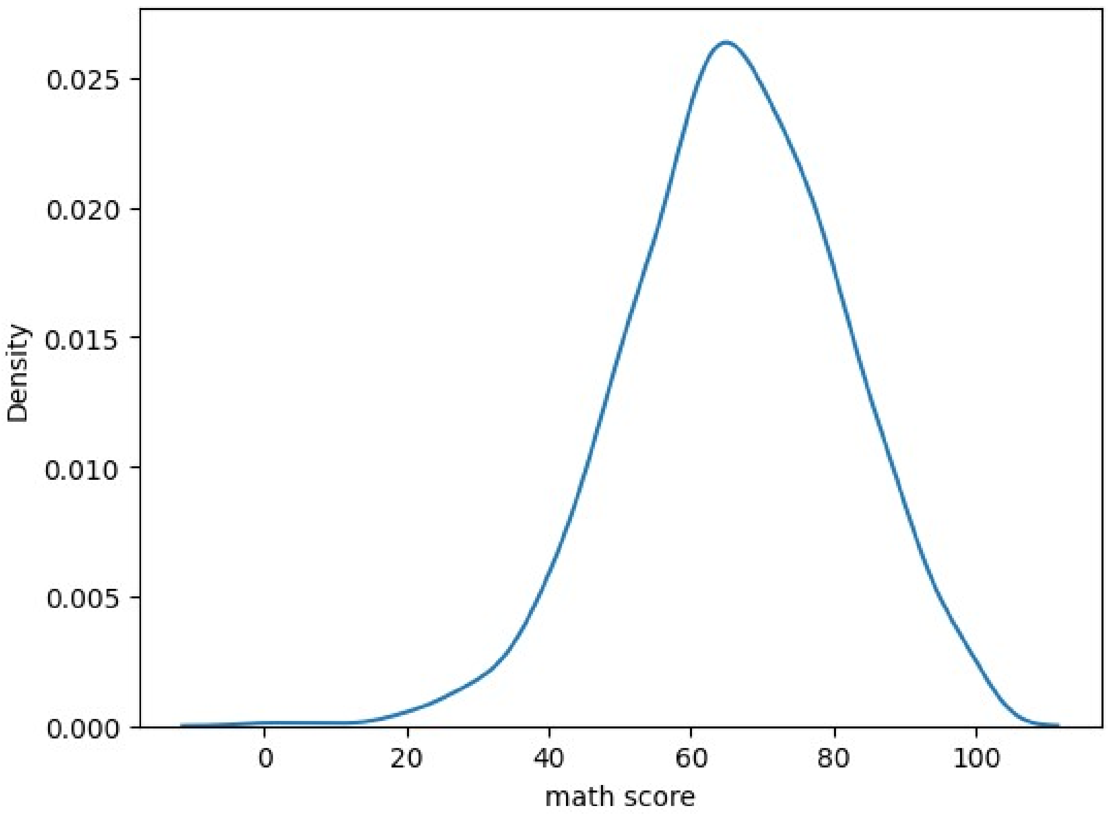
```
Std_df.isnull().sum()
```

```
gender                          0
race/ethnicity                  0
parental level of education     0
lunch                           1
test preparation course        10
math score                     10
reading score                  10
```

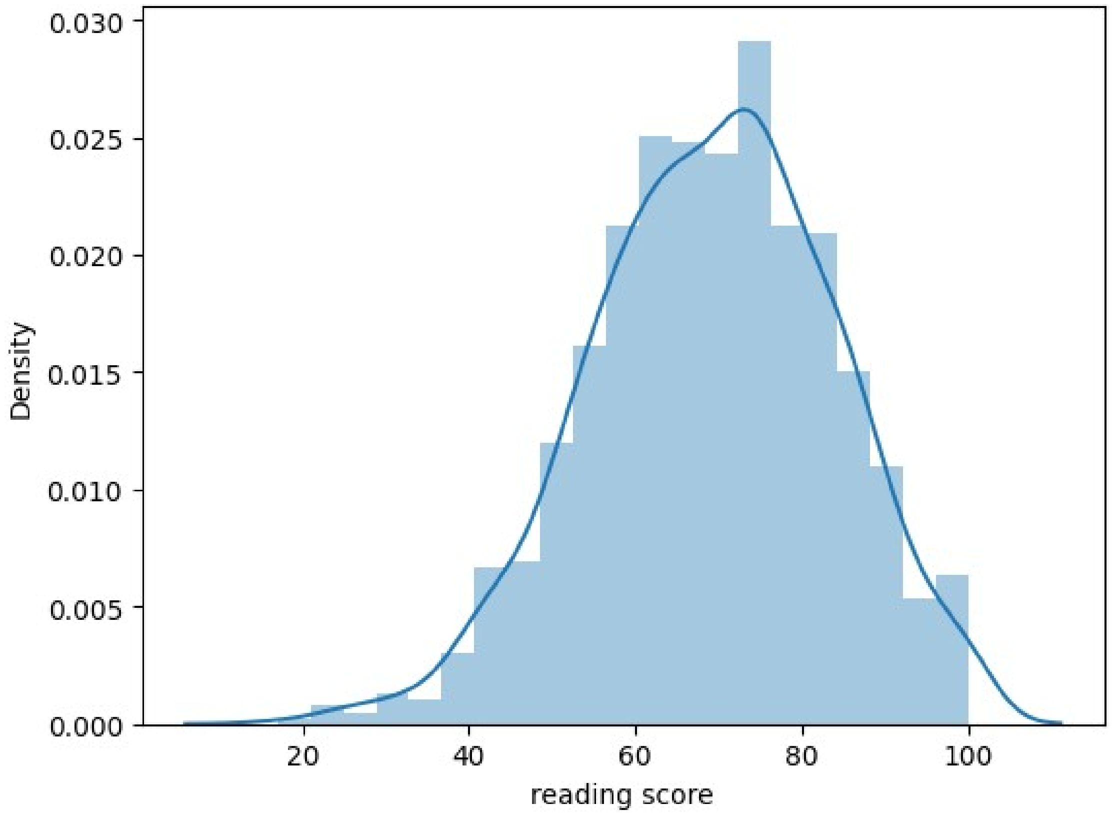
```
writing score                  10
dtype: int64

Std_df['lunch'].value_counts()

lunch
standard        644
free/reduced    355
Name: count, dtype: int64

Std_df.lunch.fillna('standard',inplace=True)

Std_df.isnull().sum()

gender                          0
race/ethnicity                  0
parental level of education     0
lunch                           0
test preparation course        10
math score                     10
reading score                  10
writing score                  10
dtype: int64

Std_df['test preparation course'].value_counts()

test preparation course
none         632
completed    358
Name: count, dtype: int64

Std_df['test preparation course'].fillna('none',inplace=True)

Std_df.isnull().sum()

gender                          0
race/ethnicity                  0
parental level of education     0
lunch                           0
test preparation course         0
math score                     10
reading score                  10
writing score                  10
dtype: int64
```

Using Seaborn lib to plot graph

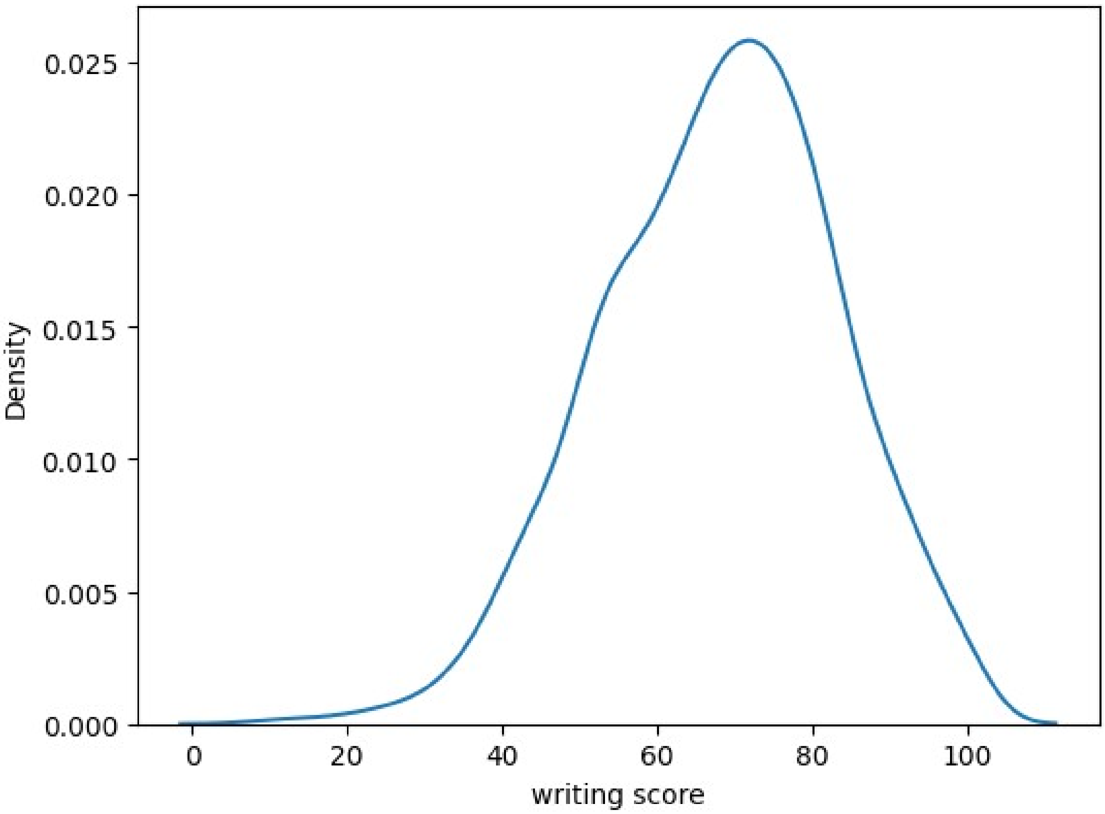
```
sns.kdeplot(Std_df['math score'])

<Axes: xlabel='math score', ylabel='Density'>
```



```
sns.distplot(Std_df['reading score'])

C:\Users\Ashitosh Warghade\AppData\Local\Temp\
ipykernel_14772\60741464.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn
v0.14.0.

Please adapt your code to use either `displot` (a figure-level
function with
similar flexibility) or `histplot` (an axes-level function for
histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(Std_df['reading score'])

<Axes: xlabel='reading score', ylabel='Density'>
```

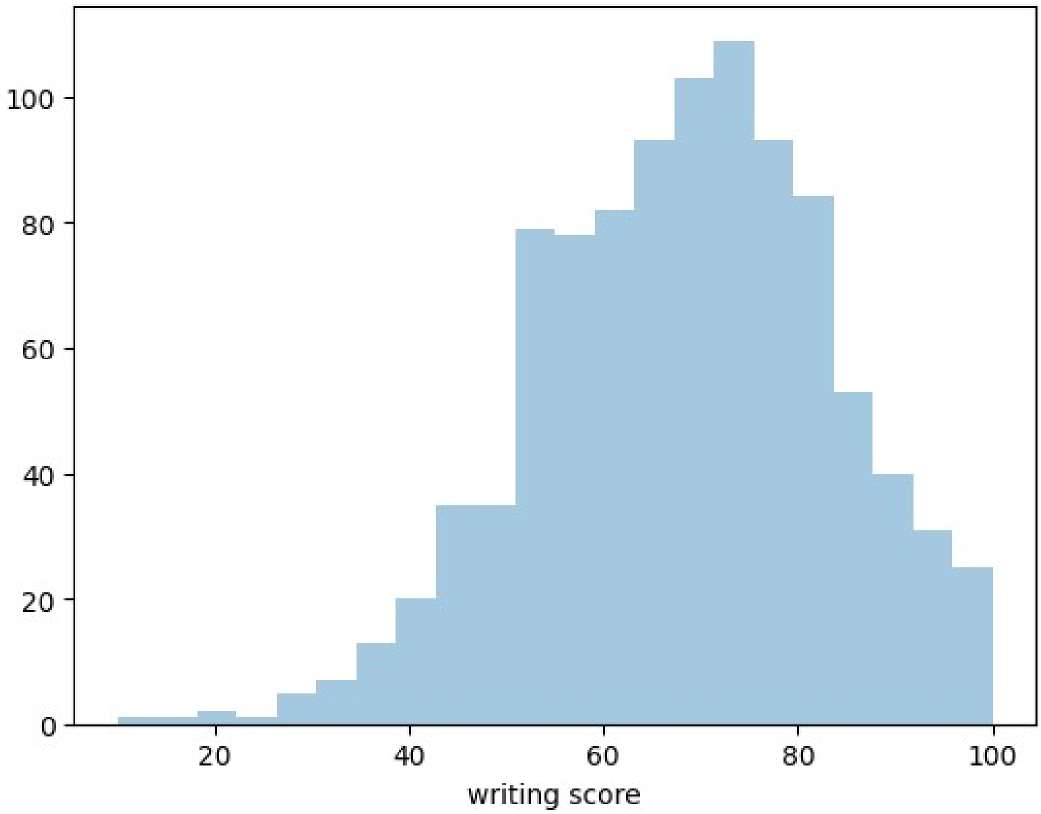


```
sns.distplot(Std_df['writing score'],hist=False)

C:\Users\Ashitosh Warghade\AppData\Local\Temp\
ipykernel_14772\3580858108.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn
v0.14.0.

Please adapt your code to use either `displot` (a figure-level
function with
similar flexibility) or `kdeplot` (an axes-level function for kernel
density plots).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(Std_df['writing score'],hist=False)

<Axes: xlabel='writing score', ylabel='Density'>
```



```
sns.distplot(Std_df['writing score'],kde=False)
```

```
C:\Users\Ashitosh Warghade\AppData\Local\Temp\
ipykernel_14772\2837882268.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn
v0.14.0.

Please adapt your code to use either `displot` (a figure-level
function with
similar flexibility) or `histplot` (an axes-level function for
histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(Std_df['writing score'],kde=False)

<Axes: xlabel='writing score'>
```



```
Std_df["math score"].skew()
```

```
-0.27756569959735494
```

```
Std_df["math score"].fillna(Std_df["math score"].mean(), inplace=True)
```

```
Std_df.isnull().sum()
```

```
gender                          0
race/ethnicity                  0
parental level of education     0
lunch                           0
test preparation course         0
math score                      0
reading score                  10
writing score                  10
dtype: int64
```

```
Std_df["writing score"].mode()
```

```
0    74.0
Name: writing score, dtype: float64
```

```
Std_df["writing score"].mode()[0]
```

```
74.0
```

```
Std_df["writing score"].fillna(Std_df["writing score"].mode()[0],
inplace=True)

Std_df.isnull().sum()
```

```
gender                          0
race/ethnicity                  0
parental level of education     0
lunch                           0
test preparation course         0
math score                      0
reading score                  10
writing score                   0
dtype: int64
```

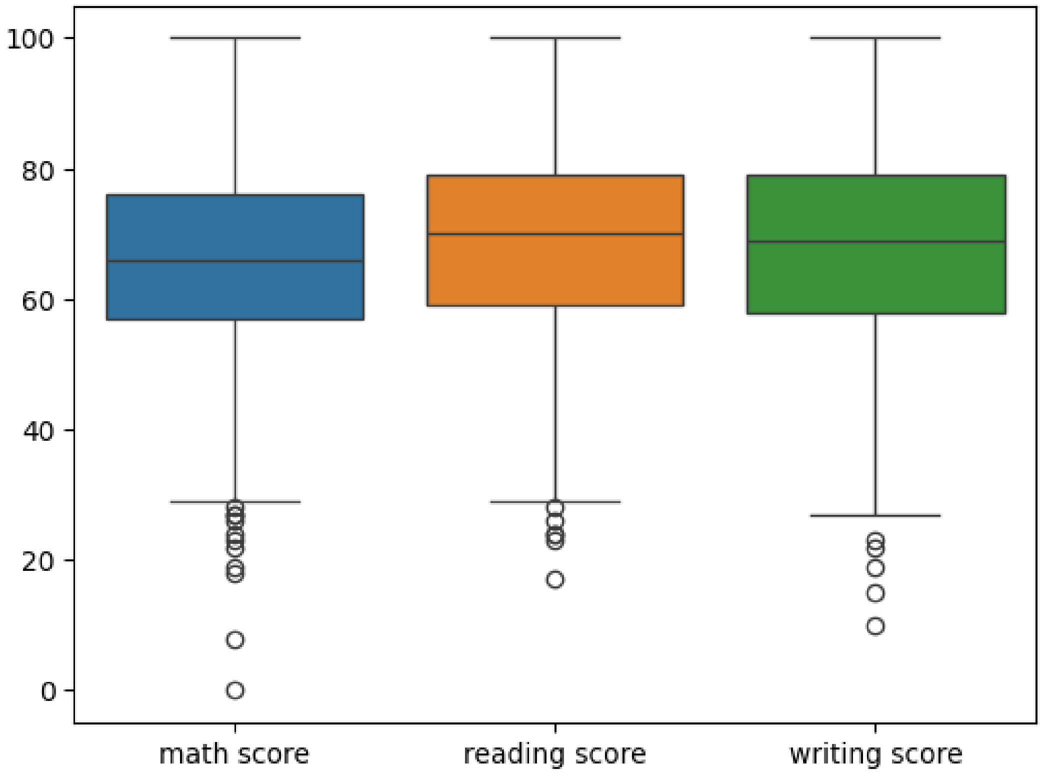
```
Std_df["reading score"].fillna(Std_df["reading score"].median(),
inplace=True)

Std_df.isnull().sum()
```

```
gender                         0
race/ethnicity                 0
parental level of education    0
lunch                          0
test preparation course        0
math score                     0
reading score                  0
writing score                  0
dtype: int64
```

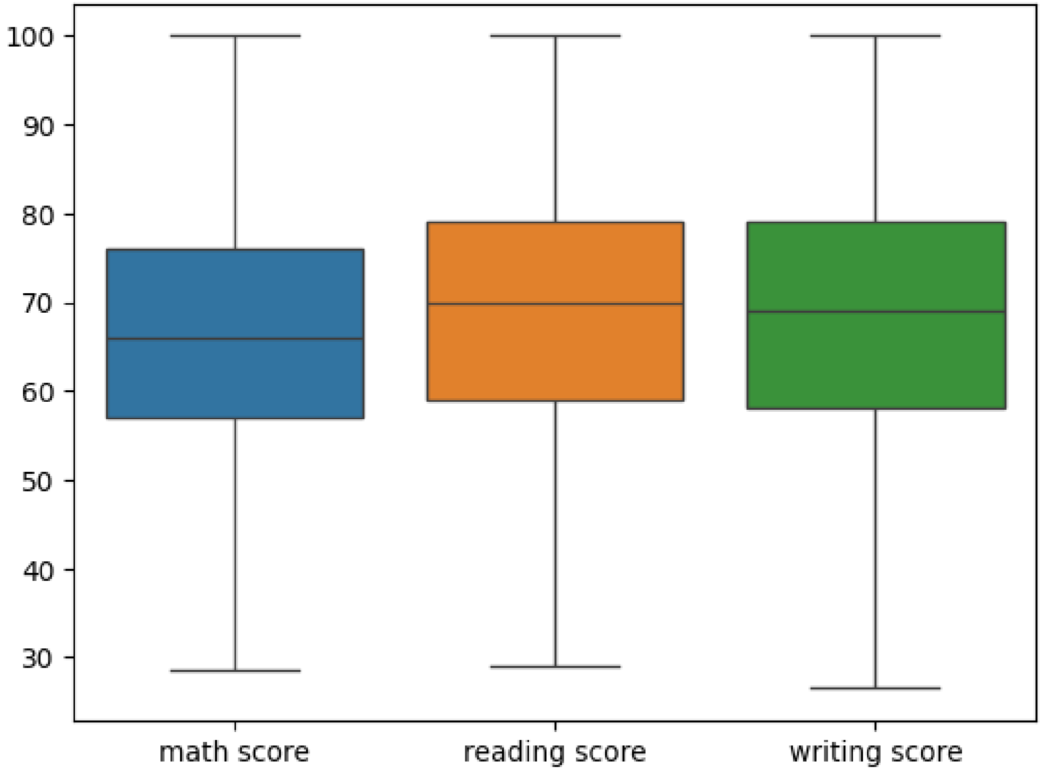
Scan all numeric variables for outliers. If there are outliers, use any of the suitable techniques to deal with them.

```
# code here
sns.boxplot(Std_df)
```

```
<Axes: >
```



```
def treat_outlier(df,col_list):
    for col_name in col_list:
        q1=df[col_name].quantile(0.25)
        q3=df[col_name].quantile(0.75)
        iqr=q3-q1
        lower=q1-(1.5*iqr)
        upper=q3+(1.5*iqr)
        np.clip(df[col_name],lower,upper,inplace=True)

treat_outlier(Std_df,['math score','writing score','reading score'])

sns.boxplot(Std_df)

<Axes: >
```



Apply data transformations on at least one of the variables. The purpose of this transformation should be one of the following reasons: to change the scale for better understanding of the variable, to convert a non-linear relation into a linear one, or to decrease the skewness and convert the distribution into a normal distribution.

```
# code here
Std_df["math score"].skew()
```

```
-0.12912399951580147
```

```
Std_df["writing score"].skew()
```

```
-0.24169581899585696
```

```
Std_df["reading score"].skew()
```

```
-0.20943942810853586
```

```
Std_df.info()
```

```
<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   float64
 6   reading score               1000 non-null   float64
 7   writing score               1000 non-null   float64
dtypes: float64(3), object(5)
memory usage: 62.6+ KB
```

```
# when we don't know the data
# data normally disturbute then use standard scale(mostly -3 to 3)
# data is not normally then min max scale(0-1)

from sklearn.preprocessing import MinMaxScaler , StandardScaler

# object scaler, class StandardScaler

scaler=StandardScaler()

# to scaler to learn the data frame math score col pattern
scaler.fit(Std_df[['math score']])
```

```
StandardScaler()
```

```
scaled_mathscore= scaler.transform(Std_df[['math score']])

scaled_mathscore

# to insert we need 3 parameter 1st col index,col name,inserting val

Std_df.insert(6,'scaled math score',scaled_mathscore)

Std_df.head()
```

```
   gender race/ethnicity parental level of education        lunch  \
0  female        group B           bachelor's degree      standard   
1  female        group C                some college      standard   
2  female        group B             master's degree      standard   
3    male        group A          associate's degree  free/reduced   
4    male        group C                some college      standard   

  test preparation course  math score  scaled math score  reading 
score  \
0                    none        72.0           0.396213          
72.0   
1               completed        69.0           0.193129          
90.0   
2                    none        90.0           1.614718          
95.0   
3                    none        47.0          -1.296154          
57.0   
```

```
4                    none        76.0           0.666992
78.0

   writing score
0           74.0
1           88.0
2           93.0
3           44.0
4           75.0
```

```python
scaler.fit(Std_df[['reading score']])
```

```
StandardScaler()
```

```python
scaled_readscore= scaler.transform(Std_df[['reading score']])
```

```python
Std_df.insert(8,'scaled reading score',scaled_readscore)
```

```python
Std_df.head()
```

```
   gender race/ethnicity parental level of education         lunch  \
0  female        group B           bachelor's degree      standard
1  female        group C                some college      standard
2  female        group B             master's degree      standard
3    male        group A          associate's degree  free/reduced
4    male        group C                some college      standard

  test preparation course  math score  scaled math score  reading
score  \
0                    none        72.0           0.396213
72.0
1               completed        69.0           0.193129
90.0
2                    none        90.0           1.614718
95.0
3                    none        47.0          -1.296154
57.0
4                    none        76.0           0.666992
78.0

   scaled reading score  writing score
0              0.197199           74.0
1              1.445736           88.0
2              1.792552           93.0
3             -0.843248           44.0
4              0.613378           75.0
```

```python
scaler=MinMaxScaler()
```

```python
scaler.fit(Std_df[['writing score']])
```

```
MinMaxScaler()

scaled_writescore= scaler.transform(Std_df[['writing score']])

Std_df.insert(10,'scaled writing score',scaled_writescore)

Std_df.tail()

     gender race/ethnicity parental level of education
lunch  \
995  female        group E             master's degree      standard

996    male        group C                 high school  free/reduced

997  female        group C                 high school  free/reduced

998  female        group D                some college      standard

999  female        group D                some college  free/reduced


    test preparation course  math score  scaled math score  reading
score  \
995               completed        88.0           1.479328
99.0
996                    none        62.0          -0.280734
55.0
997               completed        59.0          -0.483818
71.0
998               completed        68.0           0.125434
78.0
999                    none        77.0           0.734687
86.0

     scaled reading score  writing score  scaled writing score
995              2.070004           95.0              0.931973
996             -0.981974           55.0              0.387755
997              0.127836           65.0              0.523810
998              0.613378           77.0              0.687075
999              1.168284           86.0              0.809524
```