Group A

Assignment 2

Data Wrangling II

Unsupported cell type. Double-click to inspect/edit the content.

Import all the required Python Libraries.

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

```
df=pd.read_csv('StudentPerformance.csv')
```

```
df.describe
```

```
<bound method NDFrame.describe of      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
..      ...            ...                         ...           ...
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  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
..                      ...         ...            ...            ...
995               completed        88.0           99.0           95.0
996                    none        62.0           55.0           55.0
997               completed        59.0           71.0           65.0
```

```
998                completed        68.0            78.0            77.0
999                     none        77.0            86.0            86.0

[1000 rows x 8 columns]>
```

```
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
```

Create a DataFrame from the dictionary

```
# code here
data={'studentid':[i for  i in  range(1,101)],
      'Age':[rd.randint(15,18) for i in range(100)],
      'class':[rd.choice(['9 th','10 th' ,'11 th' ,'12 th']) for _ in range(100)]
      ,'attendence':[rd.uniform(10,100) for _ in range(100)],
      'score':[rd.randint(30,100) for _ in range(100) ]
      }
```

```
df.to_csv('StudentPerformance' , index =False)
```

```
df=pd.DataFrame(data )
```

```
df
```

	studentid	Age	class	attendence	score
0	1	18	12 th	43.031925	30
1	2	16	11 th	42.414792	84
2	3	16	11 th	54.104171	72
3	4	15	9 th	13.776707	83
4	5	15	12 th	58.093743	30
...	...	...	...	...	...
95	96	17	9 th	83.385795	38
96	97	15	12 th	44.920736	67
97	98	15	10 th	56.505661	35
98	99	15	11 th	95.003787	96
99	100	16	10 th	72.273267	43

100 rows × 5 columns

Load the Dataset into pandas dataframe.

```
# code here
import pandas as pd
df=pd.read_csv("StudentPerformance.csv");
```

```
df.head()
```

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	wr
0	female	group B	bachelor's degree	standard	none	72.0	72.0	
1	female	group C	some college	standard	completed	69.0	90.0	
2	female	group B	master's degree	standard	none	90.0	95.0	
3	male	group A	associate's degree	free/reduced	none	47.0	57.0	
4	male	group C	some college	standard	none	76.0	78.0	

```
df.shape
```

```
(1000, 8)
```

```
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
```

```
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

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.

Data Preprocessing

```
# code here
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
```

```
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
```

```
df["gender"].value_counts() #categorical column
```

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

```
df['gender'].fillna('female',inplace=True)
```

```
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
```

```
df['gender'].mode(0)
```

```
0    female
Name: gender, dtype: object
```

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

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

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

```
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
```

```
df['lunch'].value_counts()
```

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

```
df['lunch'].mode()
```

```
0    standard
Name: lunch, dtype: object
```

```
df['lunch'].mode()[0]
```

```
'standard'
```

```
df['lunch'].fillna(df['lunch'].mode()[0],inplace=True)
```

```
df.isnull().sum()
```

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

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

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

```
df['parental level of education'].fillna(df['parental level of education'].mode()[0],inpla
```

```
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
```

```
df['test preparation course'].value_counts()
```

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

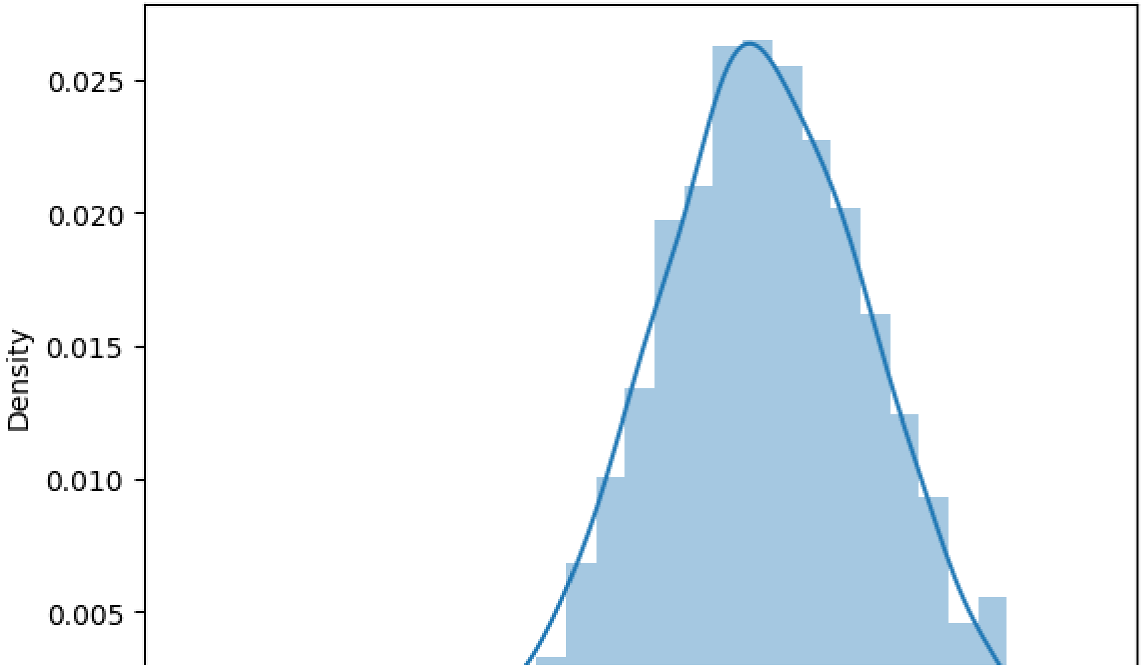
```
df['test preparation course'].fillna(df['test preparation course'].mode()[0],inplace=True)
```

```
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
```

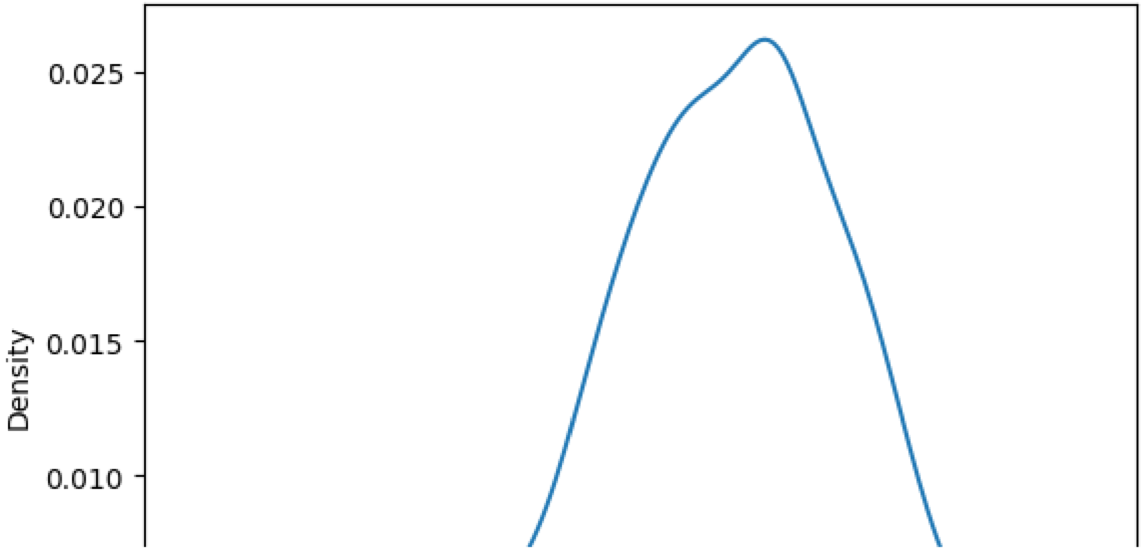
```
sns.distplot(df['math score'])
```

```
<ipython-input-113-913cdd0f89a7>: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(df['math score'])
<Axes: xlabel='math score', ylabel='Density'>
```



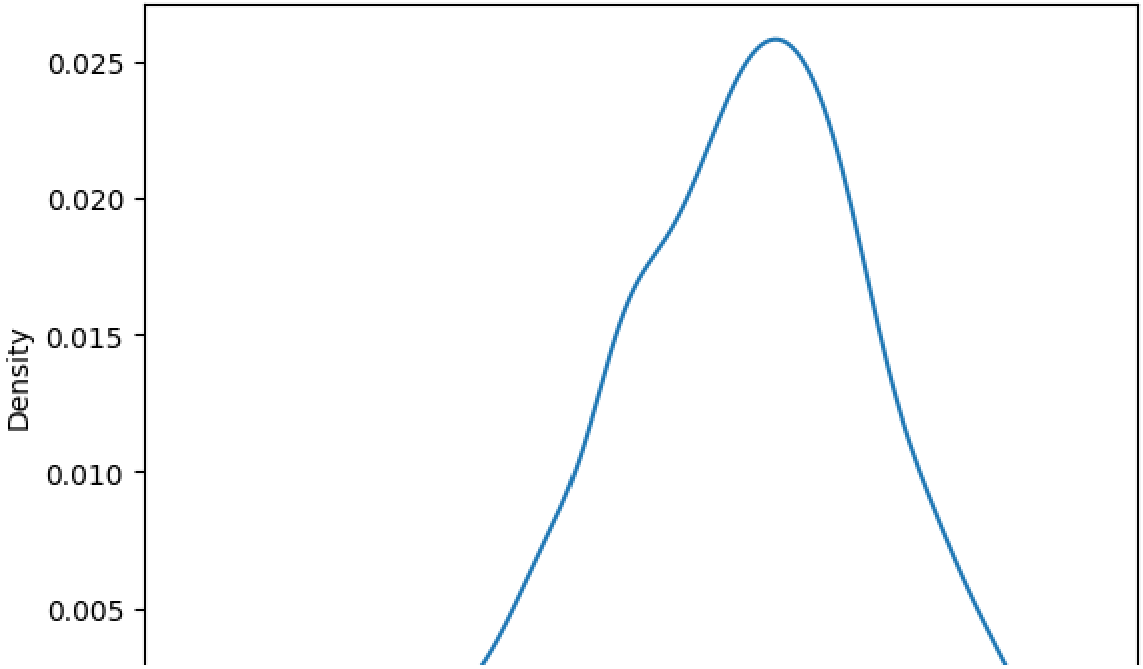
```
sns.kdeplot(df['reading score'])#normally distributed
```

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



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

```
<ipython-input-115-0c4d8e6059b9>: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 pl

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

  sns.distplot(df['writing score'],hist=False,)
<Axes: xlabel='writing score', ylabel='Density'>
```



```
df['math score'].fillna(df['math score'].mean(),inplace=True)
```

```
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
```

```
df['reading score'].fillna(df['reading score'].median(),inplace=True)
```

```
df['writing score'].fillna(df['writing score'].mode()[0],inplace=True)
```

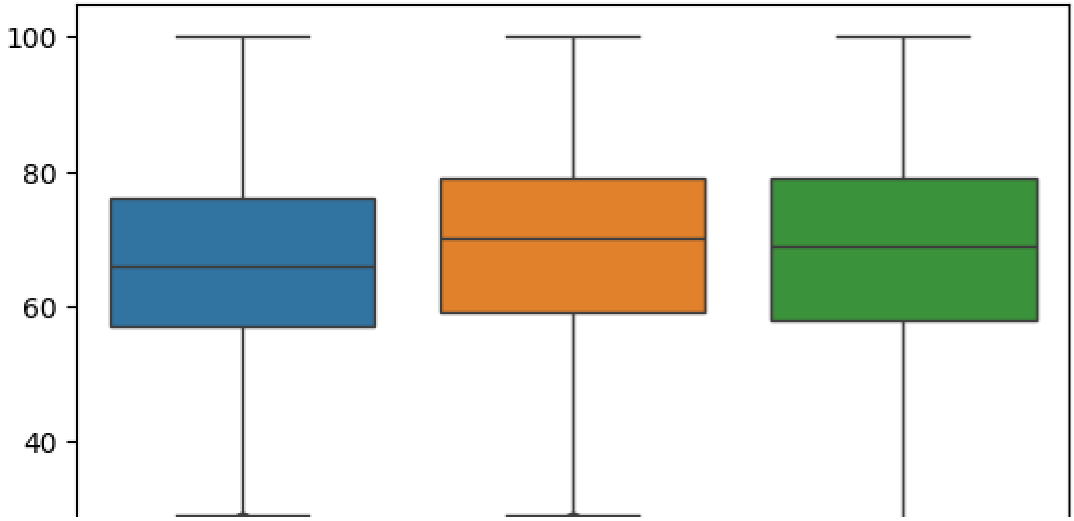
```
df.isnull().sum()
```

```
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.

```
sns.boxplot(df)
```

```
<Axes: >
```

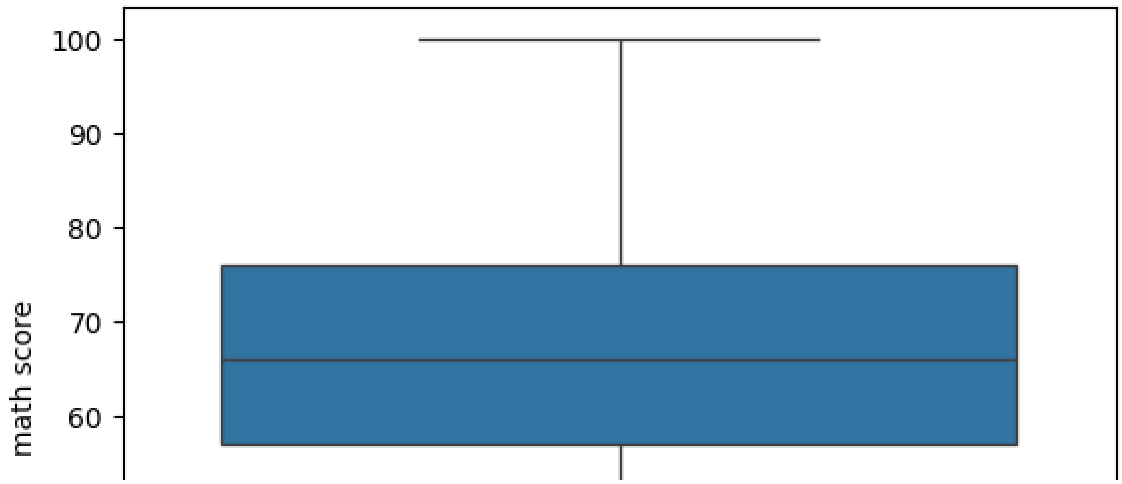


```
Q1=df['math score'].quantile(0.25)
Q3=df['math score'].quantile(0.75)
IQR=Q3-Q1
lower= Q1-(1.5*IQR)
upper=Q3+(1.5*IQR)
```

```
np.clip(df['math score'],lower,upper,inplace=True)
```

```
sns.boxplot(df['math score'])
```

```
<Axes: ylabel='math score'>
```



```
# code here
```

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
df['math score'].skew()
```

```
-0.12912399951580147
```

```
df['reading score'].skew()
```

```
-0.2595478399998487
```

```
df['writing score'].skew()
```

```
-0.3125529577143879
```

```
from sklearn.preprocessing import StandardScaler,MinMaxScaler
```

```
scaler=StandardScaler()
```

```
scaler.fit(df[['math score']])
```

```
▾ StandardScaler
StandardScaler()
```

```
scaled_mscore=scaler.transform(df[['math score']])
```

```
df.insert(6,'scaled math score',scaled_mscore)
```

```
df.head()
```

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	scaled math score	re
0	female	group B	bachelor's degree	standard	none	72.0	0.396213	
1	female	group C	some college	standard	completed	69.0	0.193129	
2	female	group B	master's degree	standard	none	90.0	1.614718	
3	male	group A	associate's degree	free/reduced	none	47.0	-1.296154	
4	male	group C	some college	standard	none	76.0	0.666992	

```
scaler.fit(df[['reading score']])
```

```
▾ StandardScaler
StandardScaler()
```

```
scaled_rscore=scaler.transform(df[['reading score']])
```