In [1]:

```
import pandas as pd
import numpy as np
import seaborn as sns
```

In [2]:

```
data = pd.read_csv("StudentsPerformance_3b35c0ed594b2791571329dfc0caf59f.csv")
```

In [3]:

```
data
```

Out[3]:

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

1000 rows × 8 columns

In [4]:

```
data.shape
```

Out[4]:

```
(1000, 8)
```

In [5]:

```
data.describe()
# only for integer values
```

Out[5]:

	math score	reading score	writing score
count	1000.00000	1000.000000	1000.000000
mean	66.08900	69.169000	68.054000
std	15.16308	14.600192	15.195657
min	0.00000	17.000000	10.000000
25%	57.00000	59.000000	57.750000
50%	66.00000	70.000000	69.000000
75%	77.00000	79.000000	79.000000
max	100.00000	100.000000	100.000000

In [6]:

```
data.columns
```

Out[6]:

```
Index(['gender', 'race/ethnicity', 'parental level of education', 'lunch',
       'test preparation course', 'math score', 'reading score',
       'writing score'],
      dtype='object')
```

In [7]:
```
# check for unique values
data.nunique()
```

Out[7]:
```
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
```

In [8]:
```
# check for specific column
data['gender'].unique()
```

Out[8]:
```
array(['female', 'male'], dtype=object)
```

In [9]:
```
data['lunch'].unique()
```

Out[9]:
```
array(['standard', 'free/reduced'], dtype=object)
```

In [10]:
```
data['parental level of education'].unique()
```

Out[10]:
```
array(["bachelor's degree", 'some college', "master's degree",
       "associate's degree", 'high school', 'some high school'],
      dtype=object)
```

2. Cleaning the data

In [11]:
```
# checking null values in data
data.isnull()
```

Out[11]:

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
0	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...
995	False	False	False	False	False	False	False	False
996	False	False	False	False	False	False	False	False
997	False	False	False	False	False	False	False	False
998	False	False	False	False	False	False	False	False
999	False	False	False	False	False	False	False	False

1000 rows × 8 columns

In [12]:
```
data.isnull().sum()
```

Out[12]:
```
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
```

In [13]:
```
#dropping the redundant data
student = data.drop(['race/ethnicity', 'parental level of education'], axis = 1)
```

In [14]:
```
student.head()
```

Out[14]:

	gender	lunch	test preparation course	math score	reading score	writing score
0	female	standard	none	72	72	74
1	female	standard	completed	69	90	88
2	female	standard	none	90	95	93
3	male	free/reduced	none	47	57	44
4	male	standard	none	76	78	75

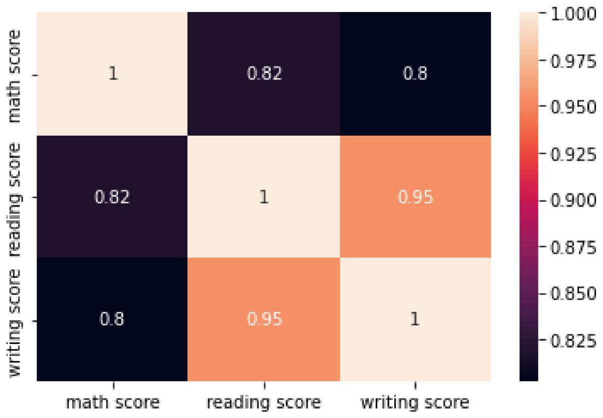
In [15]:
```
# checking for outliers
# here data is almost clean so we do not have any value which shows a distinct variation
```

3. Relationship analysis

In [16]:
```
# using correlation between variables
corelation = student.corr()
```

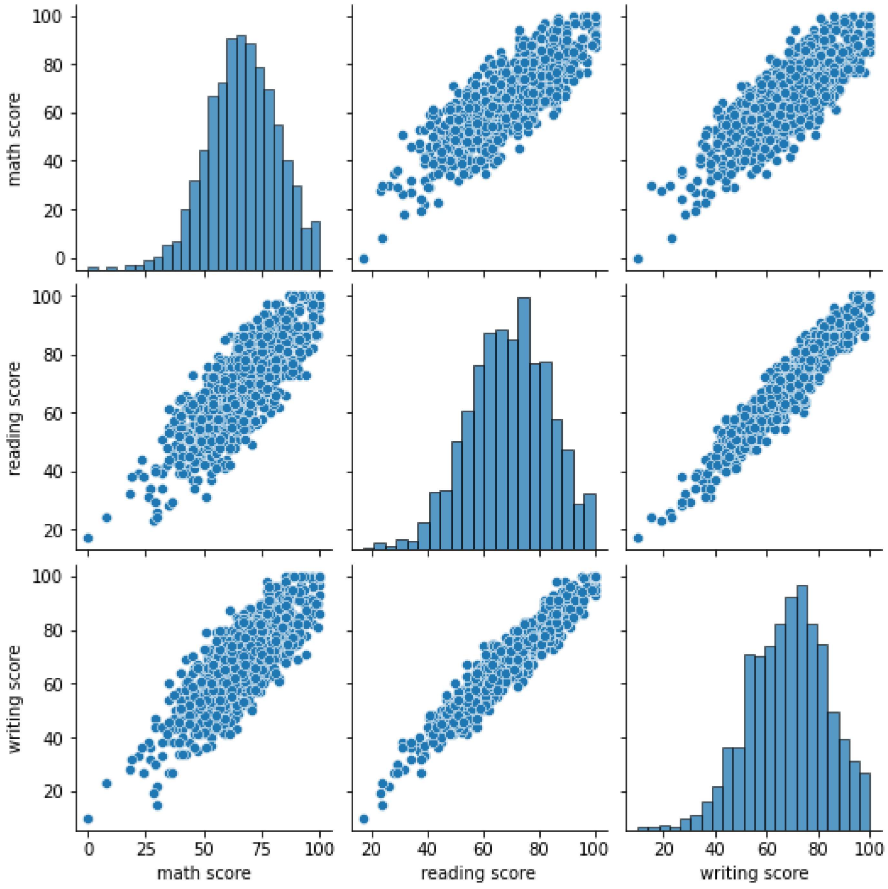
In [17]:
```
sns.heatmap(corelation, xticklabels=corelation.columns, yticklabels=corelation.columns, annot=True
```

Out[17]:
```
<AxesSubplot:>
```



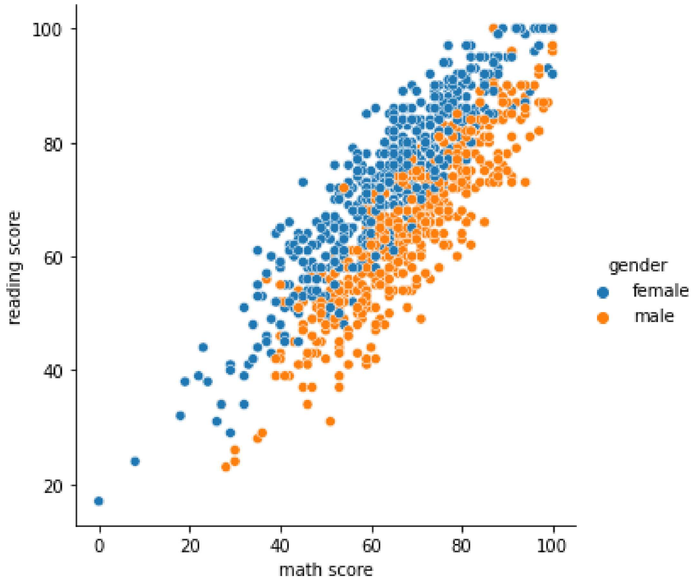
In [18]:
```
sns.pairplot(student)
# sued to view realtionship between any two variables : continous, categorical, boolean
```

Out[18]:
```
<seaborn.axisgrid.PairGrid at 0x1a73c321850>
```



In [19]:
```
# use scatter plot to see relationship between two numerical variables
# use relation plot
sns.relplot(x = 'math score', y = 'reading score', hue = 'gender', data = student)
```

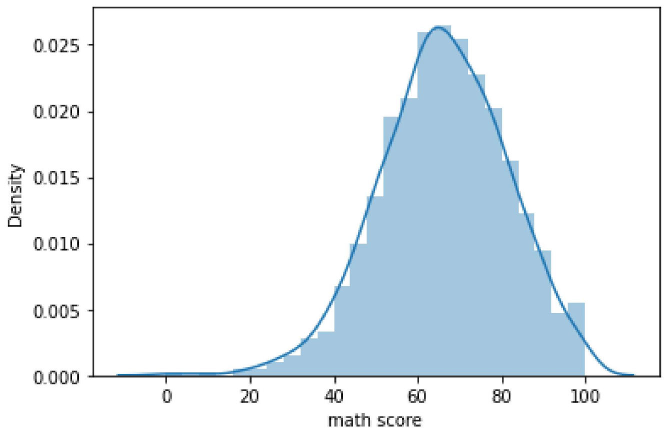
Out[19]:
```
<seaborn.axisgrid.FacetGrid at 0x1a73cd46b20>
```



In [20]:
```
# using histograms
sns.distplot(student['math score'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplo
t` is a deprecated function and will be removed in a future version. Please adapt your code to us
e either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-leve
l function for histograms).
  warnings.warn(msg, FutureWarning)
```

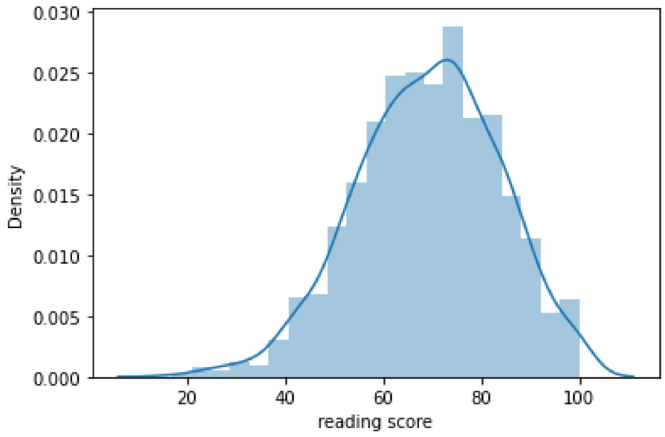
Out[20]:
```
<AxesSubplot:xlabel='math score', ylabel='Density'>
```



In [21]:
```
sns.distplot(student['reading score'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplo
t` is a deprecated function and will be removed in a future version. Please adapt your code to us
e either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-leve
l function for histograms).
  warnings.warn(msg, FutureWarning)
```

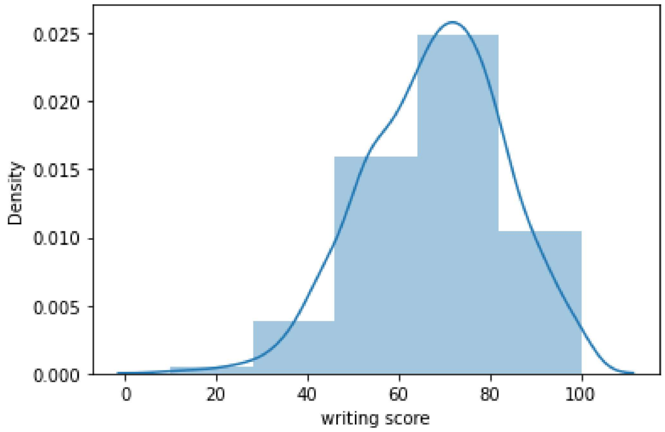
Out[21]:
```
<AxesSubplot:xlabel='reading score', ylabel='Density'>
```



```
In [22]: sns.distplot(student['writing score'], bins=5)
```

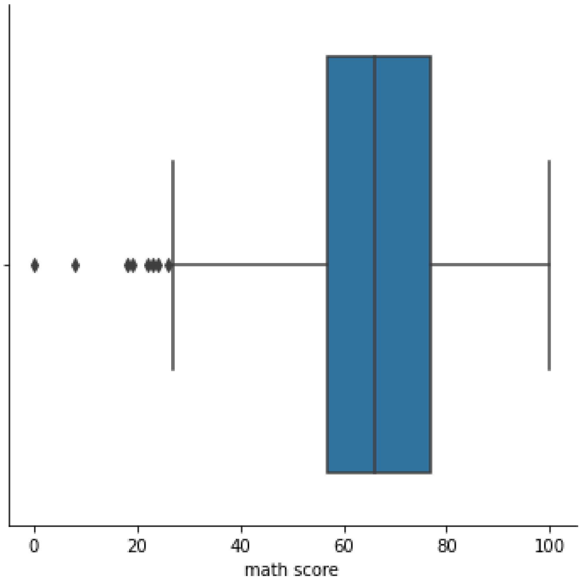
```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplo
t` is a deprecated function and will be removed in a future version. Please adapt your code to us
e either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-leve
l function for histograms).
  warnings.warn(msg, FutureWarning)
```

```
Out[22]: <AxesSubplot:xlabel='writing score', ylabel='Density'>
```



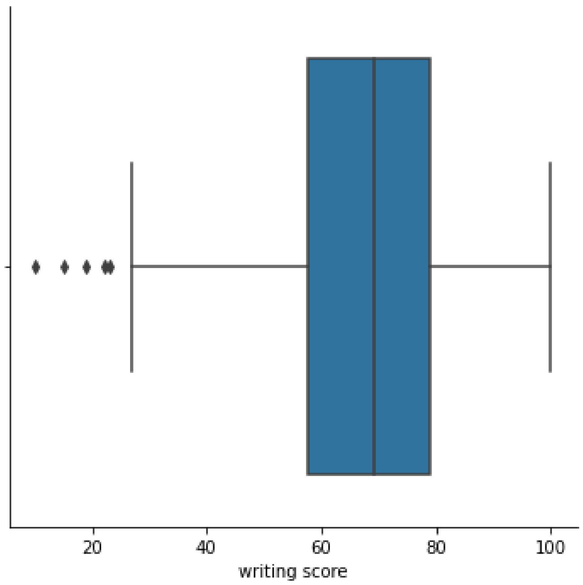
```
In [23]: # categorical plot
         sns.catplot(x = 'math score', kind= 'box', data = student)
```

```
Out[23]: <seaborn.axisgrid.FacetGrid at 0x1a73d111be0>
```



```
In [24]: sns.catplot(x = 'writing score', kind= 'box', data = student)
```

```
Out[24]: <seaborn.axisgrid.FacetGrid at 0x1a73d867430>
```



```
In [ ]:
```