In [ ]:
```
#Data Wrangling II
Create an “Academic performance” dataset of students and perform the following
Python.
1. Scan all variables for missing values and inconsistencies. If there are mis
inconsistencies, use any of the suitable techniques to deal with them.
2. Scan all numeric variables for outliers. If there are outliers, use any of
techniques to deal with them.
3. Apply data transformations on at least one of the variables. The purpose of
transformation should be one of the following reasons: to change the scale for
understanding of the variable, to convert a non-linear relation into a linear
decrease the skewness and convert the distribution into a normal distribution.
Reason and document your approach properly.

#StudentPerformance.csv
```

In [1]:
```
import pandas as pd
import numpy as np
```

In [2]:
```
df=pd.read_csv("StudentPerformance.csv")
```

In [3]:
```
df
```

Out[3]:

	gender	math_score	reading_score	writing_score	placement_score	placement_offer_count
0	female	77	75	60	98	2
1	female	61	85	69	89	3
2	male	69	89	73	83	4
3	male	62	88	78	85	1
4	female	63	93	74	80	1
5	female	77	87	77	86	2
6	male	63	82	75	83	1
7	female	74	87	77	88	0
8	male	67	94	76	75	2
9	female	76	90	66	76	2
10	male	66	90	73	93	0
11	male	70	94	64	93	1
12	male	63	87	62	85	0
13	female	80	78	66	76	0
14	female	66	92	68	75	0
15	female	75	89	68	75	3
16	male	74	87	76	99	4
17	female	64	81	76	90	3
18	male	68	87	70	83	4
19	female	62	88	79	99	2
20	male	74	76	74	77	2
21	female	72	76	78	81	4
22	male	60	88	70	86	4
23	male	64	91	71	93	1
24	female	70	79	79	100	0
25	female	64	86	72	97	2
26	male	69	75	65	99	3
27	female	67	75	65	84	3
28	female	60	88	70	93	1
29	male	77	89	75	96	1

checking missing values using notnull()

In [4]:
```
df.notnull()
```

Out[4]:

	gender	math_score	reading_score	writing_score	placement_score	placement_offer_count
0	True	True	True	True	True	True
1	True	True	True	True	True	True
2	True	True	True	True	True	True
3	True	True	True	True	True	True
4	True	True	True	True	True	True
5	True	True	True	True	True	True
6	True	True	True	True	True	True
7	True	True	True	True	True	True
8	True	True	True	True	True	True
9	True	True	True	True	True	True
10	True	True	True	True	True	True
11	True	True	True	True	True	True
12	True	True	True	True	True	True
13	True	True	True	True	True	True
14	True	True	True	True	True	True
15	True	True	True	True	True	True
16	True	True	True	True	True	True
17	True	True	True	True	True	True
18	True	True	True	True	True	True
19	True	True	True	True	True	True
20	True	True	True	True	True	True
21	True	True	True	True	True	True
22	True	True	True	True	True	True
23	True	True	True	True	True	True
24	True	True	True	True	True	True
25	True	True	True	True	True	True
26	True	True	True	True	True	True
27	True	True	True	True	True	True
28	True	True	True	True	True	True
29	True	True	True	True	True	True

In [5]:
```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
df['gender']=le.fit_transform(df['gender'])
newdf=df
```

In [6]:
```
df
```

Out[6]:

	gender	math_score	reading_score	writing_score	placement_score	placement_offer_count
0	0	77	75	60	98	2
1	0	61	85	69	89	3
2	1	69	89	73	83	4
3	1	62	88	78	85	1
4	0	63	93	74	80	1
5	0	77	87	77	86	2
6	1	63	82	75	83	1
7	0	74	87	77	88	0
8	1	67	94	76	75	2
9	0	76	90	66	76	2
10	1	66	90	73	93	0
11	1	70	94	64	93	1
12	1	63	87	62	85	0
13	0	80	78	66	76	0
14	0	66	92	68	75	0
15	0	75	89	68	75	3
16	1	74	87	76	99	4
17	0	64	81	76	90	3
18	1	68	87	70	83	4
19	0	62	88	79	99	2
20	1	74	76	74	77	2
21	0	72	76	78	81	4
22	1	60	88	70	86	4
23	1	64	91	71	93	1
24	0	70	79	79	100	0
25	0	64	86	72	97	2
26	1	69	75	65	99	3
27	0	67	75	65	84	3
28	0	60	88	70	93	1
29	1	77	89	75	96	1

Filling missing values using dropna(), fillna(), replace()

In [7]:
```
missing_values = ["na"]
```

In [8]:
```
df=pd.read_csv("StudentPerformance.csv",
               na_values=missing_values)
```

In [9]:
```
df
```

Out[9]:

	gender	math_score	reading_score	writing_score	placement_score	placement_offer_count
0	female	77	75	60	98	2
1	female	61	85	69	89	3
2	male	69	89	73	83	4
3	male	62	88	78	85	1
4	female	63	93	74	80	1
5	female	77	87	77	86	2
6	male	63	82	75	83	1
7	female	74	87	77	88	0
8	male	67	94	76	75	2
9	female	76	90	66	76	2
10	male	66	90	73	93	0
11	male	70	94	64	93	1
12	male	63	87	62	85	0
13	female	80	78	66	76	0
14	female	66	92	68	75	0
15	female	75	89	68	75	3
16	male	74	87	76	99	4
17	female	64	81	76	90	3
18	male	68	87	70	83	4
19	female	62	88	79	99	2
20	male	74	76	74	77	2
21	female	72	76	78	81	4
22	male	60	88	70	86	4
23	male	64	91	71	93	1
24	female	70	79	79	100	0
25	female	64	86	72	97	2
26	male	69	75	65	99	3
27	female	67	75	65	84	3
28	female	60	88	70	93	1
29	male	77	89	75	96	1

filling null values with a single value

In [10]:
```
ndf=df
ndf.fillna(0)
```

Out[10]:

	gender	math_score	reading_score	writing_score	placement_score	placement_offer_count
0	female	77	75	60	98	2
1	female	61	85	69	89	3
2	male	69	89	73	83	4
3	male	62	88	78	85	1
4	female	63	93	74	80	1
5	female	77	87	77	86	2
6	male	63	82	75	83	1
7	female	74	87	77	88	0
8	male	67	94	76	75	2
9	female	76	90	66	76	2
10	male	66	90	73	93	0
11	male	70	94	64	93	1
12	male	63	87	62	85	0
13	female	80	78	66	76	0
14	female	66	92	68	75	0
15	female	75	89	68	75	3
16	male	74	87	76	99	4
17	female	64	81	76	90	3
18	male	68	87	70	83	4
19	female	62	88	79	99	2
20	male	74	76	74	77	2
21	female	72	76	78	81	4
22	male	60	88	70	86	4
23	male	64	91	71	93	1
24	female	70	79	79	100	0
25	female	64	86	72	97	2
26	male	69	75	65	99	3
27	female	67	75	65	84	3
28	female	60	88	70	93	1
29	male	77	89	75	96	1

filling null values using replace()

In [11]:
```
ndf.replace(to_replace=np.nan, value=-1)
```

Out[11]:

	gender	math_score	reading_score	writing_score	placement_score	placement_offer_count
0	female	77	75	60	98	2
1	female	61	85	69	89	3
2	male	69	89	73	83	4
3	male	62	88	78	85	1
4	female	63	93	74	80	1
5	female	77	87	77	86	2
6	male	63	82	75	83	1
7	female	74	87	77	88	0
8	male	67	94	76	75	2
9	female	76	90	66	76	2
10	male	66	90	73	93	0
11	male	70	94	64	93	1
12	male	63	87	62	85	0
13	female	80	78	66	76	0
14	female	66	92	68	75	0
15	female	75	89	68	75	3
16	male	74	87	76	99	4
17	female	64	81	76	90	3
18	male	68	87	70	83	4
19	female	62	88	79	99	2
20	male	74	76	74	77	2
21	female	72	76	78	81	4
22	male	60	88	70	86	4
23	male	64	91	71	93	1
24	female	70	79	79	100	0
25	female	64	86	72	97	2
26	male	69	75	65	99	3
27	female	67	75	65	84	3
28	female	60	88	70	93	1
29	male	77	89	75	96	1

deleting null values using dropna() method

In [12]:
```
ndf.dropna()
```

Out[12]:

	gender	math_score	reading_score	writing_score	placement_score	placement_offer_count
0	female	77	75	60	98	2
2	male	69	89	73	83	4
4	female	63	93	74	80	1
5	female	77	87	77	86	2
8	male	67	94	76	75	2
10	male	66	90	73	93	0
11	male	70	94	64	93	1
13	female	80	78	66	76	0
14	female	66	92	68	75	0
16	male	74	87	76	99	4
17	female	64	81	76	90	3
18	male	68	87	70	83	4
19	female	62	88	79	99	2
21	female	72	76	78	81	4
22	male	60	88	70	86	4
23	male	64	91	71	93	1
25	female	64	86	72	97	2
27	female	67	75	65	84	3
28	female	60	88	70	93	1

Identification and handling outliers

In [13]:
```
#Go to line
```

In [14]:
```
df
```

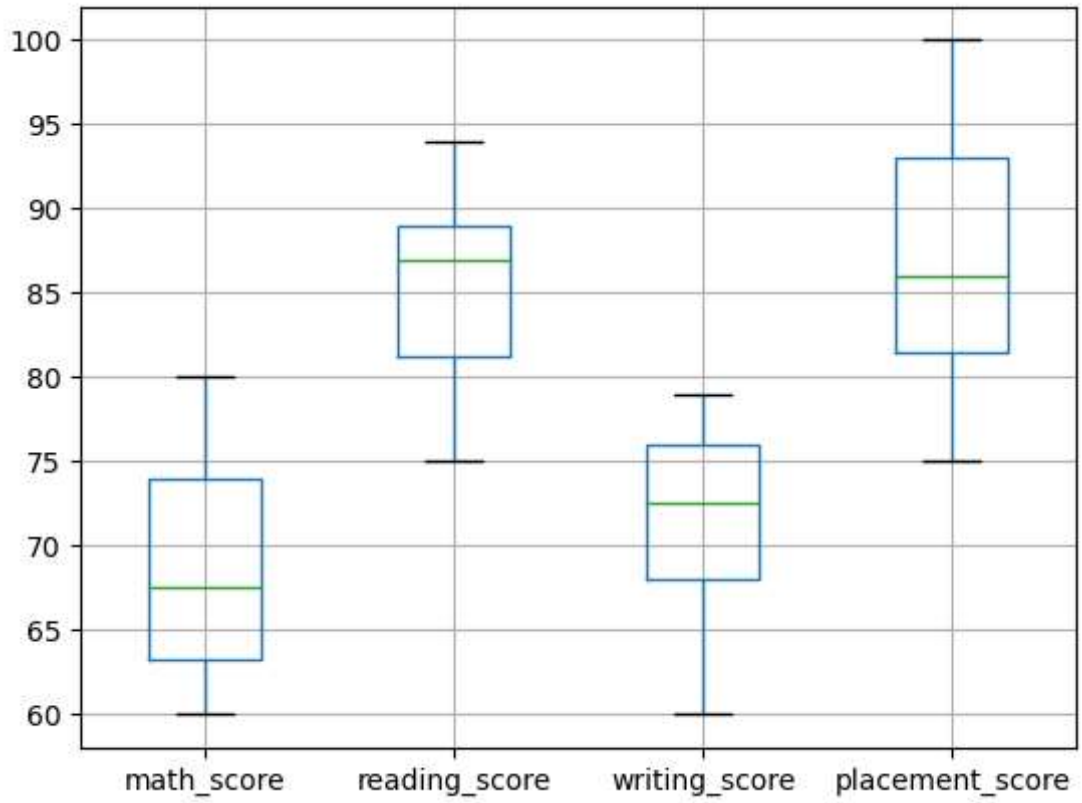
Out[14]:

	gender	math_score	reading_score	writing_score	placement_score	placement_offer_count
0	female	77	75	60	98	2
1	female	61	85	69	89	3
2	male	69	89	73	83	4
3	male	62	88	78	85	1
4	female	63	93	74	80	1
5	female	77	87	77	86	2
6	male	63	82	75	83	1
7	female	74	87	77	88	0
8	male	67	94	76	75	2
9	female	76	90	66	76	2
10	male	66	90	73	93	0
11	male	70	94	64	93	1
12	male	63	87	62	85	0
13	female	80	78	66	76	0
14	female	66	92	68	75	0
15	female	75	89	68	75	3
16	male	74	87	76	99	4
17	female	64	81	76	90	3
18	male	68	87	70	83	4
19	female	62	88	79	99	2
20	male	74	76	74	77	2
21	female	72	76	78	81	4
22	male	60	88	70	86	4
23	male	64	91	71	93	1
24	female	70	79	79	100	0
25	female	64	86	72	97	2
26	male	69	75	65	99	3
27	female	67	75	65	84	3
28	female	60	88	70	93	1
29	male	77	89	75	96	1

detecting outlier using boxplot

In [15]:
```
col=['math_score','reading_score',
     'writing_score','placement_score']
df.boxplot(col)
```

Out[15]: <AxesSubplot:>



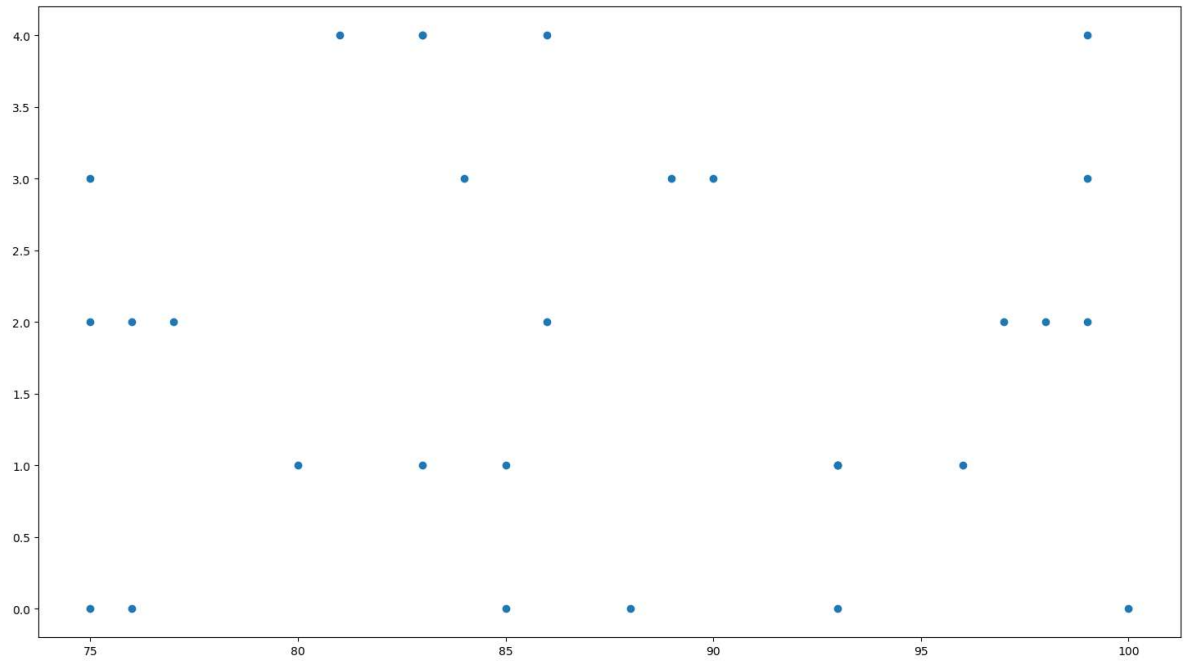
In [16]:
```
print(np.where(df['math_score']>90))
print(np.where(df['reading_score']<25))
print(np.where(df['writing_score']<30))
```

```
(array([], dtype=int64),)
(array([], dtype=int64),)
(array([], dtype=int64),)
```

detecting outlier using scatterplot

In [17]:
```
import matplotlib.pyplot as plt
```

In [18]:
```
fig, ax=plt.subplots(figsize=(18,10))
ax.scatter(df['placement_score'],
           df['placement_offer_count'])
plt.show()
```



Detecting outliers using Z-Score:

In [19]:
```
import numpy as np
from scipy import stats
```

In [20]:
```
z = np.abs(stats.zscore(df['math_score']))
```

In [21]:
```
print(z)
```

```
0     1.471070
1     1.287186
2     0.091942
3     1.114795
4     0.942404
5     1.471070
6     0.942404
7     0.953897
8     0.252840
9     1.298679
10    0.425231
11    0.264333
12    0.942404
13    1.988243
14    0.425231
15    1.126288
16    0.953897
17    0.770013
18    0.080449
19    1.114795
20    0.953897
21    0.609115
22    1.459577
23    0.770013
24    0.264333
25    0.770013
26    0.091942
27    0.252840
28    1.459577
29    1.471070
Name: math_score, dtype: float64
```

In [22]:
```
threshold=0.18
sample_outliers = np.where(z<threshold)
```

In [23]:
```
sample_outliers
```

Out[23]:
```
(array([ 2, 18, 26], dtype=int64),)
```

Detecting outliers using Inter Quantile Range(IQR):

In [24]:
```
sorted_rscore= sorted(df['reading_score'])
sorted_rscore
```

Out[24]:
```
[75,
 75,
 75,
 76,
 76,
 78,
 79,
 81,
 82,
 85,
 86,
 87,
 87,
 87,
 87,
 87,
 88,
 88,
 88,
 88,
 89,
 89,
 89,
 90,
 90,
 91,
 92,
 93,
 94,
 94]
```

In [25]:
```
q1 = np.percentile(sorted_rscore, 25)
q3 = np.percentile(sorted_rscore, 75)
print(q1,q3)
```

```
81.25 89.0
```

In [26]:
```
IQR = q3-q1
```

In [27]:
```
lwr_bound = q1-(1.5*IQR)
upr_bound = q3+(1.5*IQR)
print(lwr_bound, upr_bound)
```

```
69.625 100.625
```

handling outliers

trimming/removing the outliers

In [28]:
```
new_df = df
for i in sample_outliers:
    new_df.drop(i,inplace=True)
new_df
```

Out[28]:

	gender	math_score	reading_score	writing_score	placement_score	placement_offer_count
0	female	77	75	60	98	2
1	female	61	85	69	89	3
3	male	62	88	78	85	1
4	female	63	93	74	80	1
5	female	77	87	77	86	2
6	male	63	82	75	83	1
7	female	74	87	77	88	0
8	male	67	94	76	75	2
9	female	76	90	66	76	2
10	male	66	90	73	93	0
11	male	70	94	64	93	1
12	male	63	87	62	85	0
13	female	80	78	66	76	0
14	female	66	92	68	75	0
15	female	75	89	68	75	3
16	male	74	87	76	99	4
17	female	64	81	76	90	3
19	female	62	88	79	99	2
20	male	74	76	74	77	2
21	female	72	76	78	81	4
22	male	60	88	70	86	4
23	male	64	91	71	93	1
24	female	70	79	79	100	0
25	female	64	86	72	97	2
27	female	67	75	65	84	3
28	female	60	88	70	93	1
29	male	77	89	75	96	1

quantile based flooring and capping

In [29]:
```
df_stud=df
ninetieth_percentile=np.percentile(df_stud['math_score'],90)
b=np.where(df_stud['math_score']>ninetieth_percentile,
ninetieth_percentile,df_stud['math_score'])
print("new Array: ",b)
```

```
new Array:  [77. 61. 62. 63. 77. 63. 74. 67. 76. 66. 70. 63. 77. 66. 75. 74.
64. 62.
 74. 72. 60. 64. 70. 64. 67. 60. 77.]
```

In [30]:
```
df_stud.insert(1,"m_score",b,True)
df_stud
```

Out[30]:

	gender	m_score	math_score	reading_score	writing_score	placement_score	placement_of
0	female	77.0	77	75	60	98	
1	female	61.0	61	85	69	89	
3	male	62.0	62	88	78	85	
4	female	63.0	63	93	74	80	
5	female	77.0	77	87	77	86	
6	male	63.0	63	82	75	83	
7	female	74.0	74	87	77	88	
8	male	67.0	67	94	76	75	
9	female	76.0	76	90	66	76	
10	male	66.0	66	90	73	93	
11	male	70.0	70	94	64	93	
12	male	63.0	63	87	62	85	
13	female	77.0	80	78	66	76	
14	female	66.0	66	92	68	75	
15	female	75.0	75	89	68	75	
16	male	74.0	74	87	76	99	
17	female	64.0	64	81	76	90	
19	female	62.0	62	88	79	99	
20	male	74.0	74	76	74	77	
21	female	72.0	72	76	78	81	
22	male	60.0	60	88	70	86	
23	male	64.0	64	91	71	93	
24	female	70.0	70	79	79	100	
25	female	64.0	64	86	72	97	
27	female	67.0	67	75	65	84	
28	female	60.0	60	88	70	93	
29	male	77.0	77	89	75	96	

mean/median imputation

In [31]:
```
col=['reading_score']
```

In [32]:
```
df.boxplot(col)
```

Out[32]:
```
<AxesSubplot:>
```

In [33]:
```
###optional uper ka 2
```

In [34]:
```
median = np.median(sorted_rscore)
median
```

Out[34]:
```
87.0
```

In [35]:
```
refined_df=df
```

In [36]:
```
refined_df['reading_score'] = np.where(refined_df['reading_score']
>upr_bound, median,refined_df['reading_score'])
```

In [37]:
```
refined_df
```

Out[37]:

	gender	m_score	math_score	reading_score	writing_score	placement_score	placement_of
0	female	77.0	77	75.0	60	98	
1	female	61.0	61	85.0	69	89	
3	male	62.0	62	88.0	78	85	
4	female	63.0	63	93.0	74	80	
5	female	77.0	77	87.0	77	86	
6	male	63.0	63	82.0	75	83	
7	female	74.0	74	87.0	77	88	
8	male	67.0	67	94.0	76	75	
9	female	76.0	76	90.0	66	76	
10	male	66.0	66	90.0	73	93	
11	male	70.0	70	94.0	64	93	
12	male	63.0	63	87.0	62	85	
13	female	77.0	80	78.0	66	76	
14	female	66.0	66	92.0	68	75	
15	female	75.0	75	89.0	68	75	
16	male	74.0	74	87.0	76	99	
17	female	64.0	64	81.0	76	90	
19	female	62.0	62	88.0	79	99	
20	male	74.0	74	76.0	74	77	
21	female	72.0	72	76.0	78	81	
22	male	60.0	60	88.0	70	86	
23	male	64.0	64	91.0	71	93	
24	female	70.0	70	79.0	79	100	
25	female	64.0	64	86.0	72	97	
27	female	67.0	67	75.0	65	84	
28	female	60.0	60	88.0	70	93	
29	male	77.0	77	89.0	75	96	

In [38]:
```
refined_df['reading_score'] = np.where(refined_df['reading_score']
<lwr_bound, median,refined_df['reading_score'])
refined_df
```

Out[38]:

	gender	m_score	math_score	reading_score	writing_score	placement_score	placement_of
0	female	77.0	77	75.0	60	98	
1	female	61.0	61	85.0	69	89	
3	male	62.0	62	88.0	78	85	
4	female	63.0	63	93.0	74	80	
5	female	77.0	77	87.0	77	86	
6	male	63.0	63	82.0	75	83	
7	female	74.0	74	87.0	77	88	
8	male	67.0	67	94.0	76	75	
9	female	76.0	76	90.0	66	76	
10	male	66.0	66	90.0	73	93	
11	male	70.0	70	94.0	64	93	
12	male	63.0	63	87.0	62	85	
13	female	77.0	80	78.0	66	76	
14	female	66.0	66	92.0	68	75	
15	female	75.0	75	89.0	68	75	
16	male	74.0	74	87.0	76	99	
17	female	64.0	64	81.0	76	90	
19	female	62.0	62	88.0	79	99	
20	male	74.0	74	76.0	74	77	
21	female	72.0	72	76.0	78	81	
22	male	60.0	60	88.0	70	86	
23	male	64.0	64	91.0	71	93	
24	female	70.0	70	79.0	79	100	
25	female	64.0	64	86.0	72	97	
27	female	67.0	67	75.0	65	84	
28	female	60.0	60	88.0	70	93	
29	male	77.0	77	89.0	75	96	

In [39]:
```
col = ['reading_score']
refined_df.boxplot(col)
```

Out[39]: <AxesSubplot:>

Data Transformation

In [40]:
```
df
```

Out[40]:

	gender	m_score	math_score	reading_score	writing_score	placement_score	placement_of
0	female	77.0	77	75.0	60	98	
1	female	61.0	61	85.0	69	89	
3	male	62.0	62	88.0	78	85	
4	female	63.0	63	93.0	74	80	
5	female	77.0	77	87.0	77	86	
6	male	63.0	63	82.0	75	83	
7	female	74.0	74	87.0	77	88	
8	male	67.0	67	94.0	76	75	
9	female	76.0	76	90.0	66	76	
10	male	66.0	66	90.0	73	93	
11	male	70.0	70	94.0	64	93	
12	male	63.0	63	87.0	62	85	
13	female	77.0	80	78.0	66	76	
14	female	66.0	66	92.0	68	75	
15	female	75.0	75	89.0	68	75	
16	male	74.0	74	87.0	76	99	
17	female	64.0	64	81.0	76	90	
19	female	62.0	62	88.0	79	99	
20	male	74.0	74	76.0	74	77	
21	female	72.0	72	76.0	78	81	
22	male	60.0	60	88.0	70	86	
23	male	64.0	64	91.0	71	93	
24	female	70.0	70	79.0	79	100	
25	female	64.0	64	86.0	72	97	
27	female	67.0	67	75.0	65	84	
28	female	60.0	60	88.0	70	93	
29	male	77.0	77	89.0	75	96	

In [41]:
```
import matplotlib.pyplot as plt
new_df['math_score'].plot(kind = 'hist')
```

Out[41]:
```
<AxesSubplot:ylabel='Frequency'>
```

In [42]:
```
df['log_math'] = np.log10(df['math_score'])
```

In [43]:
```
df['log_math'].plot(kind = 'hist')
```

Out[43]:
```
<AxesSubplot:ylabel='Frequency'>
```

In [44]:
```
#-------------------------------------------------------------------------
```

In [45]:
```
#Outliers Detection
```

In [46]:
```
import matplotlib.pyplot as plt
plt.rcParams["figure.figsize"] = (9, 6)
df_list = ['math_score', 'reading_score', 'writing_score', 'placement_score',
fig, axes = plt.subplots(2, 3)
fig.set_dpi(120)

count=0
for r in range(2):
    for c in range(3):
        df[df_list[count]].plot(kind = 'box', ax=axes[r,c])
        count+=1
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_25188\792931691.py in <module>
      8 for r in range(2):
      9     for c in range(3):
---> 10         df[df_list[count]].plot(kind = 'box', ax=axes[r,c])
     11         count+=1

IndexError: list index out of range
```

In [ ]:
```

```