Data Science And Big Data Analytics Practical 2

==

Name:- Rohini Janardan Devkar

Roll no:- 23272

PRN no:- 72030818G

Class :- TE2

==

Problem Statement:-

Perform the following operations using Python on any open source dataset (eg. data.csv)

1. 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.

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

3. 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. Reason and document your approach properly

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:
```
url_link = 'prac2.csv'
df=pd.read_csv(url_link)
```

In [3]:
```
df.head(20)
```

Out[3]:

	Rollno	Name	Gender	Branch	Attendance	Phy_marks	Che_marks	EM1_marks	PPS_marks	SME_marks
0	1	Mohammed	M	Comp	72.0	62.0	98.0	63.0	89.0	62.0
1	2	Reyansh	M	IT	58.0	62.0	83.0	83.0	88.0	58.0
2	3	Aarav	M	IT	57.0	-20.0	100.0	NaN	56.0	100.0
3	4	Atharv	M	IT	60.0	89.0	83.0	70.0	33.0	85.0
4	5	Vivaan	M	Comp	85.0	90.0	NaN	78.0	23.0	60.0
5	6	Advik	M	ENTC	94.0	99.0	84.0	100.0	56.0	NaN
6	7	Ansh	M	ENTC	98.0	88.0	95.0	81.0	78.0	-13.0
7	8	Ishaan	M	ENTC	75.0	66.0	66.0	83.0	-99.0	56.0
8	9	Dhruv	M	ENTC	63.0	NaN	NaN	97.0	56.0	33.0
9	10	Siddarth	M	ENTC	96.0	67.0	78.0	95.0	NaN	23.0
10	11	Vihaan	M	ENTC	82.0	54.0	70.0	88.0	55.0	77.0
11	12	Arjun	M	IT	75.0	64.0	67.0	71.0	66.0	65.0
12	13	Aarush	M	IT	67.0	56.0	81.0	NaN	90.0	86.0
13	14	Leo	M	IT	98.0	-34.0	70.0	94.0	77.0	NaN
14	15	Maryam	F	IT	64.0	87.0	60.0	90.0	65.0	64.0
15	16	Sakshi	F	Comp	66.0	90.0	95.0	67.0	99.0	93.0
16	17	Zaranew	F	Comp	93.0	54.0	NaN	75.0	90.0	53.0
17	18	Inaya	F	Comp	74.0	67.0	93.0	93.0	87.0	-20.0
18	19	Aarya	F	Comp	72.0	88.0	84.0	81.0	86.0	78.0
19	20	Pari	F	Comp	53.0	76.0	81.0	93.0	65.0	-23.0

In [4]:
```
df.shape
```

Out[4]:
```
(20, 10)
```

In [5]:
```
df.dtypes.value_counts()
```

Out[5]:
```
float64    6
object     3
int64      1
dtype: int64
```

In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20 entries, 0 to 19
Data columns (total 10 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Rollno      20 non-null     int64
 1   Name        20 non-null     object
 2   Gender      20 non-null     object
 3   Branch      20 non-null     object
 4   Attendance  20 non-null     float64
 5   Phy_marks   19 non-null     float64
 6   Che_marks   17 non-null     float64
 7   EM1_marks   18 non-null     float64
 8   PPS_marks   19 non-null     float64
 9   SME_marks   18 non-null     float64
dtypes: float64(6), int64(1), object(3)
memory usage: 1.7+ KB
```

In [7]:
```
df.describe()
```

Out[7]:

	Rollno	Attendance	Phy_marks	Che_marks	EM1_marks	PPS_marks	SME_marks
count	20.00000	20.000000	19.000000	17.000000	18.000000	19.000000	18.000000
mean	10.50000	75.100000	63.421053	81.647059	83.444444	61.052632	52.055556
std	5.91608	14.660724	34.940133	12.036098	11.078449	43.767661	37.805185
min	1.00000	53.000000	-34.000000	60.000000	63.000000	-99.000000	-23.000000
25%	5.75000	63.750000	59.000000	70.000000	75.750000	56.000000	38.000000
50%	10.50000	73.000000	67.000000	83.000000	83.000000	66.000000	61.000000
75%	15.25000	87.000000	88.000000	93.000000	93.000000	87.500000	77.750000
max	20.00000	98.000000	99.000000	100.000000	100.000000	99.000000	100.000000

In [8]:
```
df.isnull().sum()
```

Out[8]:
```
Rollno        0
Name          0
Gender        0
Branch        0
Attendance    0
Phy_marks     1
Che_marks     3
EM1_marks     2
PPS_marks     1
SME_marks     2
dtype: int64
```

In [9]:
```
data=df
coln=[]
miss=[]
coln.extend(data.columns)
#print(miss)
for i in coln:
    t=data[i].isnull().sum()
    if t!=0:
        miss.append(i)
    else:
        continue
print(miss)
```

```
['Phy_marks', 'Che_marks', 'EM1_marks', 'PPS_marks', 'SME_marks']
```

In [10]:
```
pd.options.mode.chained_assignment = None
for j in miss:
    q=data[j].dtypes
    if (q=='int64' or q=='float64') :
        f=data[j]
        for k in range(data.shape[0]):
            if (f[k]<0 or f[k]>100) :
                f[k]=(np.nan)
    else:
        continue
```

In [11]:
```
for j in miss:
    q=data[j].dtypes
    if (q=='int64' or q=='float64') :
        data[j].fillna((data[j].mean()),inplace=True)
    else:
        data.fillna(method='bfill')
data.head(10)
```

Out[11]:

	Rollno	Name	Gender	Branch	Attendance	Phy_marks	Che_marks	EM1_marks	PPS_marks	SME_marks
0	1	Mohammed	M	Comp	72.0	62.000000	98.000000	63.000000	89.000000	62.0
1	2	Reyansh	M	IT	58.0	62.000000	83.000000	83.000000	88.000000	58.0
2	3	Aarav	M	IT	57.0	74.058824	100.000000	83.444444	56.000000	100.0
3	4	Atharv	M	IT	60.0	89.000000	83.000000	70.000000	33.000000	85.0
4	5	Vivaan	M	Comp	85.0	90.000000	81.647059	78.000000	23.000000	60.0
5	6	Advik	M	ENTC	94.0	99.000000	84.000000	100.000000	56.000000	66.2
6	7	Ansh	M	ENTC	98.0	88.000000	95.000000	81.000000	78.000000	66.2
7	8	Ishaan	M	ENTC	75.0	66.000000	66.000000	83.000000	69.944444	56.0
8	9	Dhruv	M	ENTC	63.0	74.058824	81.647059	97.000000	56.000000	33.0
9	10	Siddarth	M	ENTC	96.0	67.000000	78.000000	95.000000	69.944444	23.0

In [12]:
```
data['Total Marks']=data['Phy_marks']+data['Che_marks']+data['EM1_marks']+data['PPS_marks']+data['SME_marks']
data['Percentage']=data['Total Marks']/5
```

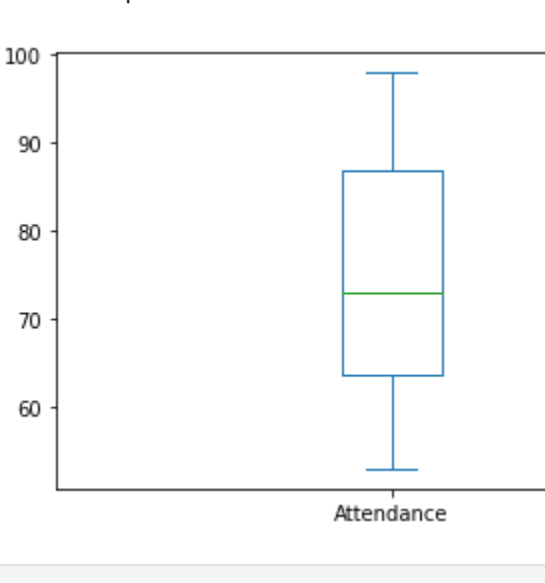
In [13]:
```
data
```

Out[13]:

	Rollno	Name	Gender	Branch	Attendance	Phy_marks	Che_marks	EM1_marks	PPS_marks	SME_marks	Total Marks	Percentage
0	1	Mohammed	M	Comp	72.0	62.000000	98.000000	63.000000	89.000000	62.0	374.000000	74.800000
1	2	Reyansh	M	IT	58.0	62.000000	83.000000	83.000000	88.000000	58.0	374.000000	74.800000
2	3	Aarav	M	IT	57.0	74.058824	100.000000	83.444444	56.000000	100.0	413.503268	82.700654
3	4	Atharv	M	IT	60.0	89.000000	83.000000	70.000000	33.000000	85.0	360.000000	72.000000
4	5	Vivaan	M	Comp	85.0	90.000000	81.647059	78.000000	23.000000	60.0	332.647059	66.529412
5	6	Advik	M	ENTC	94.0	99.000000	84.000000	100.000000	56.000000	66.2	405.200000	81.040000
6	7	Ansh	M	ENTC	98.0	88.000000	95.000000	81.000000	78.000000	66.2	408.200000	81.640000
7	8	Ishaan	M	ENTC	75.0	66.000000	66.000000	83.000000	69.944444	56.0	340.944444	68.188889
8	9	Dhruv	M	ENTC	63.0	74.058824	81.647059	97.000000	56.000000	33.0	341.705882	68.341176
9	10	Siddarth	M	ENTC	96.0	67.000000	78.000000	95.000000	69.944444	23.0	332.944444	66.588889
10	11	Vihaan	M	ENTC	82.0	54.000000	70.000000	88.000000	55.000000	77.0	344.000000	68.800000
11	12	Arjun	M	IT	75.0	64.000000	67.000000	71.000000	66.000000	65.0	333.000000	66.600000
12	13	Aarush	M	IT	67.0	56.000000	81.000000	83.444444	90.000000	86.0	396.444444	79.288889
13	14	Leo	M	IT	98.0	74.058824	70.000000	94.000000	77.000000	66.2	381.258824	76.251765
14	15	Maryam	F	IT	64.0	87.000000	60.000000	90.000000	65.000000	64.0	366.000000	73.200000
15	16	Sakshi	F	Comp	66.0	90.000000	95.000000	67.000000	99.000000	93.0	444.000000	88.800000
16	17	Zaranew	F	Comp	93.0	54.000000	81.647059	75.000000	90.000000	53.0	353.647059	70.729412
17	18	Inaya	F	Comp	74.0	67.000000	93.000000	93.000000	87.000000	66.2	406.200000	81.240000
18	19	Aarya	F	Comp	72.0	88.000000	84.000000	81.000000	86.000000	78.0	417.000000	83.400000
19	20	Pari	F	Comp	53.0	76.000000	81.000000	93.000000	65.000000	66.2	381.200000	76.240000

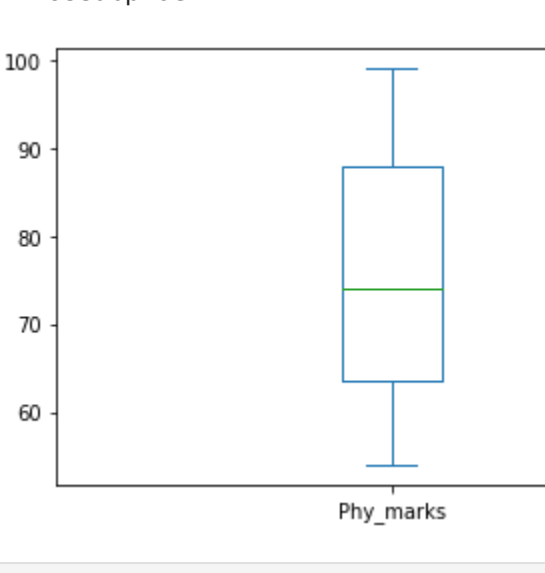
In [14]:
```
data['Attendance'].plot(kind='box')
```

Out[14]:
```
<AxesSubplot:>
```



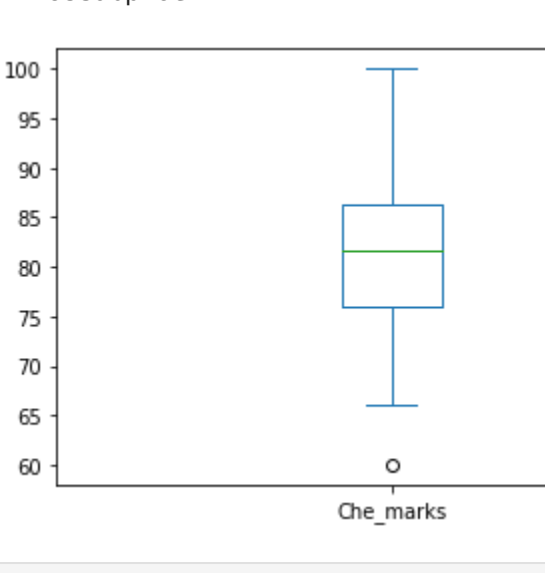
In [15]:
```
data['Phy_marks'].plot(kind='box')
```

Out[15]:
```
<AxesSubplot:>
```



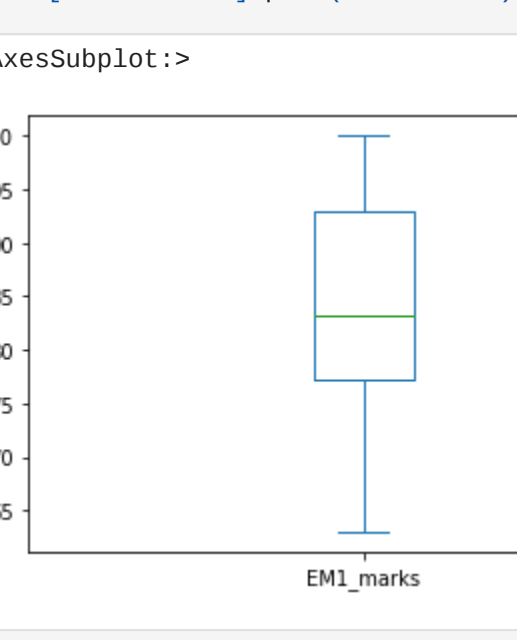
In [16]:
```
data['Che_marks'].plot(kind='box')
```

Out[16]:
```
<AxesSubplot:>
```



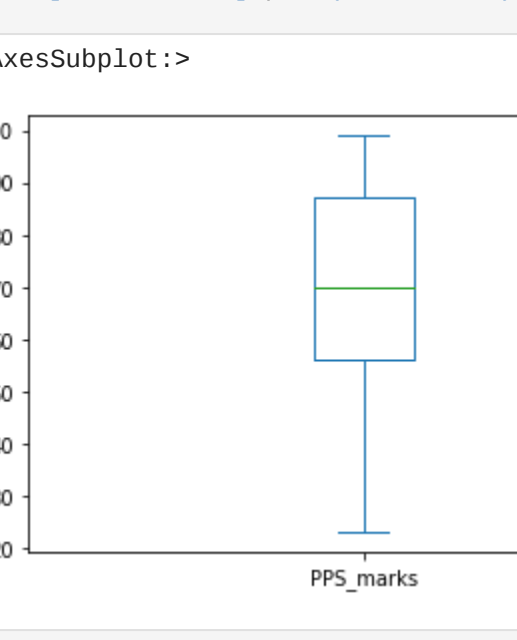
In [17]:
```
data['EM1_marks'].plot(kind='box')
```

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



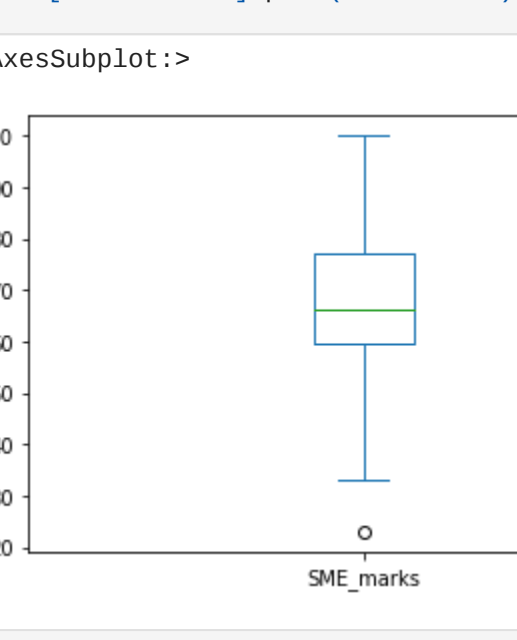
In [18]:
```
data['PPS_marks'].plot(kind='box')
```

Out[18]:
```
<AxesSubplot:>
```



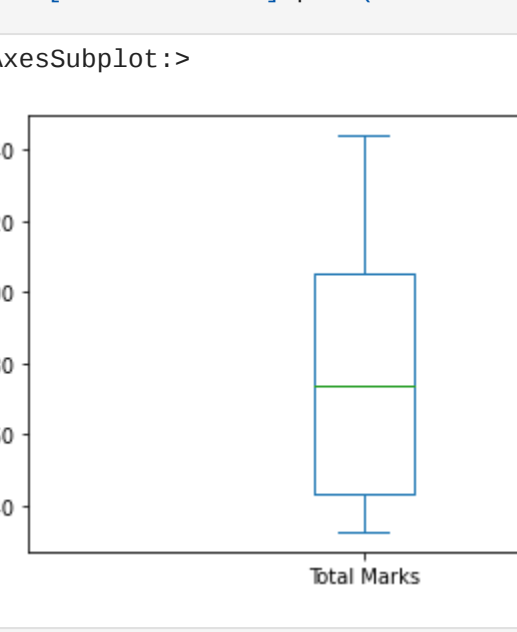
In [19]:
```
data['SME_marks'].plot(kind='box')
```

Out[19]:
```
<AxesSubplot:>
```



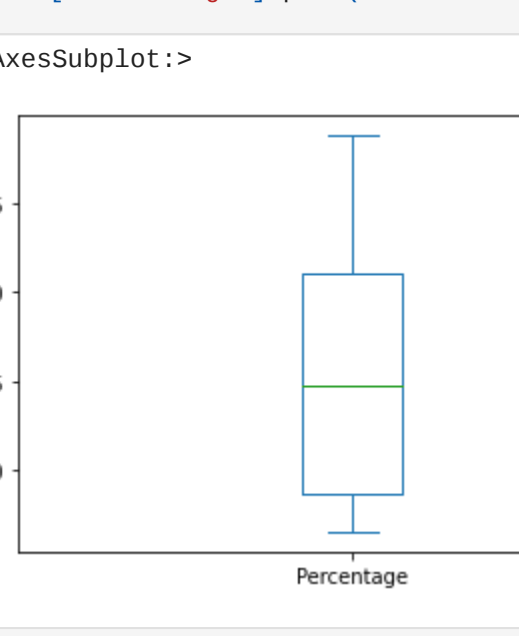
In [20]:
```
data['Total Marks'].plot(kind='box')
```

Out[20]:
```
<AxesSubplot:>
```



In [21]:
```
data['Percentage'].plot(kind='box')
```

Out[21]:
```
<AxesSubplot:>
```



In [22]:
```
Q1 = data['Attendance'].quantile(0.25)
Q3 = data['Attendance'].quantile(0.75)
IQR = Q3 - Q1
Lower_limit = Q1 - 1.5*IQR
Upper_limit = Q3 + 1.5*IQR
print("Q1 :",Q1,"\nQ3 :",Q3,"\nIQR :",IQR,"\nLower_limit :",Lower_limit,"\nUpper_limit:",Upper_limit)
```

```
Q1 : 63.75
Q3 : 87.0
IQR : 23.25
Lower_limit : 28.875
Upper_limit: 121.875
```

In [23]:
```
data[(data['Attendance']<Lower_limit)|(data['Attendance']>Upper_limit)]
```

Out[23]:

Rollno	Name	Gender	Branch	Attendance	Phy_marks	Che_marks	EM1_marks	PPS_marks	SME_marks	Total Marks	Percentage

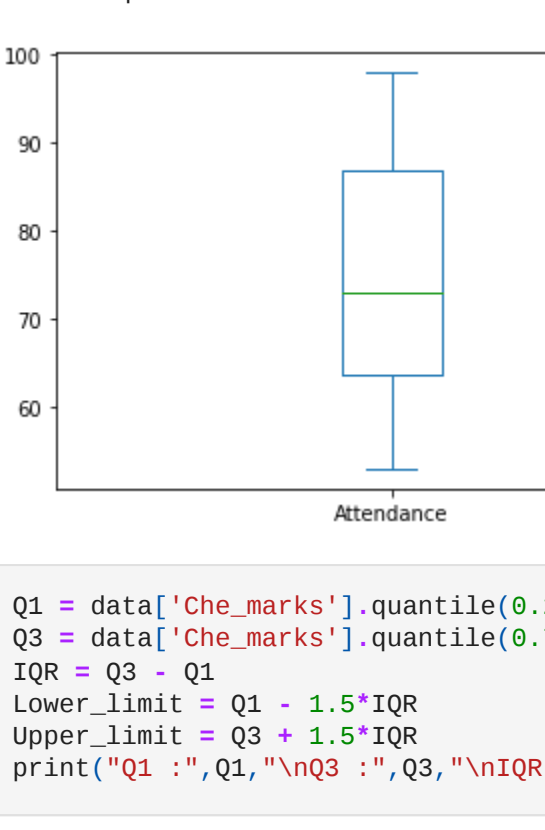
In [24]:
```
data=data[(data['Attendance']>Lower_limit)&(data['Attendance']<Upper_limit)]
data[60:70]
```

Out[24]:

Rollno	Name	Gender	Branch	Attendance	Phy_marks	Che_marks	EM1_marks	PPS_marks	SME_marks	Total Marks	Percentage

In [25]:
```
data['Attendance'].plot(kind='box')
```

Out[25]:
```
<AxesSubplot:>
```



In [26]:
```
Q1 = data['Che_marks'].quantile(0.25)
Q3 = data['Che_marks'].quantile(0.75)
IQR = Q3 - Q1
Lower_limit = Q1 - 1.5*IQR
Upper_limit = Q3 + 1.5*IQR
print("Q1 :",Q1,"\nQ3 :",Q3,"\nIQR :",IQR,"\nLower_limit :",Lower_limit,"\nUpper_limit:",Upper_limit)
```

```
Q1 : 76.0
Q3 : 86.25
IQR : 10.25
Lower_limit : 60.625
Upper_limit: 101.625
```

In [27]:
```
data[(data['Che_marks']<Lower_limit)|(data['Che_marks']>Upper_limit)]
```

Out[27]:

	Rollno	Name	Gender	Branch	Attendance	Phy_marks	Che_marks	EM1_marks	PPS_marks	SME_marks	Total Marks	Percentage
14	15	Maryam	F	IT	64.0	87.0	60.0	90.0	65.0	64.0	366.0	73.2

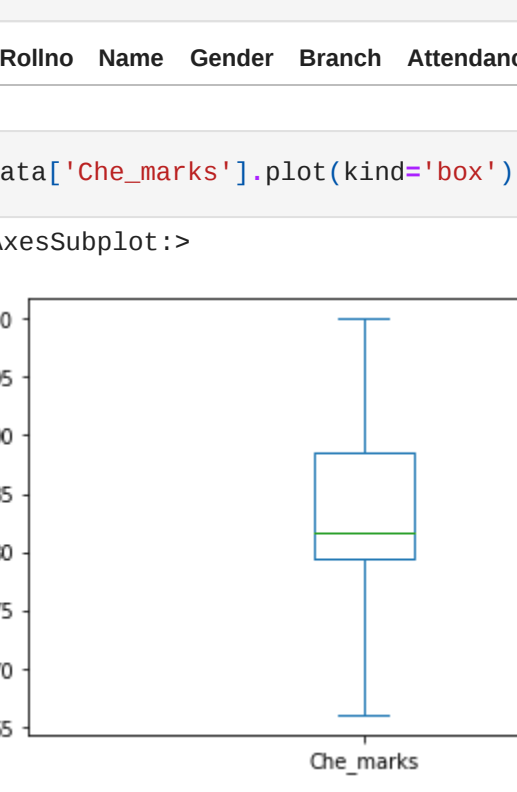
In [28]:
```
data=data[(data['Che_marks']>Lower_limit)&(data['Che_marks']<Upper_limit)]
data[140:170]
```

Out[28]:

Rollno	Name	Gender	Branch	Attendance	Phy_marks	Che_marks	EM1_marks	PPS_marks	SME_marks	Total Marks	Percentage

In [29]:
```
data['Che_marks'].plot(kind='box')
```

Out[29]:
```
<AxesSubplot:>
```



In [30]:
```
Q1 = data['EM1_marks'].quantile(0.25)
Q3 = data['EM1_marks'].quantile(0.75)
IQR = Q3 - Q1
Lower_limit = Q1 - 1.5*IQR
Upper_limit = Q3 + 1.5*IQR
print("Q1 :",Q1,"\nQ3 :",Q3,"\nIQR :",IQR,"\nLower_limit :",Lower_limit,"\nUpper_limit",Upper_limit)
```

```
Q1 : 76.5
Q3 : 93.0
IQR : 16.5
Lower_limit : 51.75
Upper_limit 117.75
```

In [31]:
```
data[(data['EM1_marks']<Lower_limit)|(data['EM1_marks']>Upper_limit)]
```

Out[31]:

Rollno	Name	Gender	Branch	Attendance	Phy_marks	Che_marks	EM1_marks	PPS_marks	SME_marks	Total Marks	Percentage

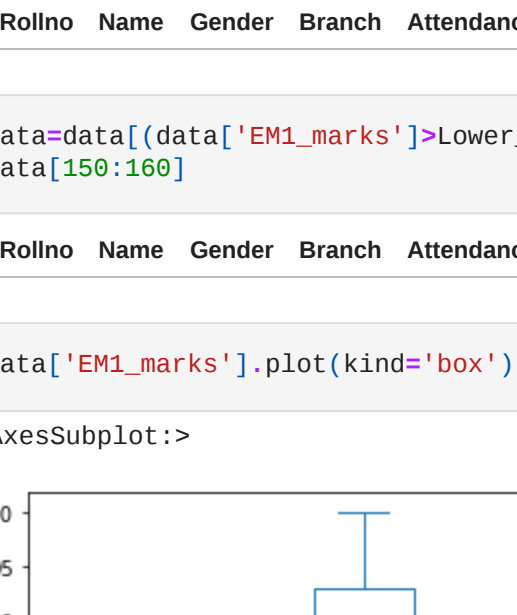
In [32]:
```
data=data[(data['EM1_marks']>Lower_limit)&(data['EM1_marks']<Upper_limit)]
data[150:160]
```

Out[32]:

Rollno	Name	Gender	Branch	Attendance	Phy_marks	Che_marks	EM1_marks	PPS_marks	SME_marks	Total Marks	Percentage

In [33]:
```
data['EM1_marks'].plot(kind='box')
```

Out[33]:
```
<AxesSubplot:>
```



In [34]:
```
def BinningFunction(column,cut_points,labels=None):
    minper=column.min()
    maxper=column.max()
    break_points=[minper]+cut_points+[maxper]
    print("Gradding According to percentage \n>60 = F \n60-70 = B \n70-80 = A\n80-100 = O")
    t=pd.cut(column,bins=break_points,labels=labels,include_lowest=True)
    return t
```

In [35]:
```
cut_points=[60,70,80]
labels=['F','B','A','O']
data["Grade"]=BinningFunction(data['Percentage'],cut_points,labels)
data
```

```
Gradding According to percentage
>60 = F
60-70 = B
70-80 = A
80-100 = O
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_1284/2930164550.py in <module>
      1 cut_points=[60,70,80]
      2 labels=['F','B','A','O']
----> 3 data["Grade"]=BinningFunction(data['Percentage'],cut_points,labels)
      4 data

~\AppData\Local\Temp/ipykernel_1284/502084138.py in BinningFunction(column, cut_points, labels)
      4     break_points=[minper]+cut_points+[maxper]
      5     print("Gradding According to percentage \n>60 = F \n60-70 = B \n70-80 = A\n80-100 = O")
----> 6     t=pd.cut(column,bins=break_points,labels=labels,include_lowest=True)
      7     return t

~\anaconda3\lib\site-packages\pandas\core\reshape\tile.py in cut(x, bins, right, labels, retbins, precision, include_lowest, duplicates, ordered)
    283         # GH 26045: cast to float64 to avoid an overflow
    284         if (np.diff(bins.astype("float64")) < 0).any():
--> 285             raise ValueError("bins must increase monotonically.")
    286
    287     fac, bins = _bins_to_cuts(

ValueError: bins must increase monotonically.
```

In [36]:
```
new_data1=data
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
column=['Attendance','Phy_marks','Che_marks','EM1_marks','PPS_marks','SME_marks','Total Marks']
scaler.fit(new_data1[column])
new_data1[column] = scaler.transform(new_data1[column])
new_data1
```

Out[36]:

	Rollno	Name	Gender	Branch	Attendance	Phy_marks	Che_marks	EM1_marks	PPS_marks	SME_marks	Total Marks	Percentage
0	1	Mohammed	M	Comp	0.422222	0.177778	0.941176	0.000000	0.868421	0.506494	0.371368	74.800000
1	2	Reyansh	M	IT	0.111111	0.177778	0.500000	0.540541	0.855263	0.454545	0.371368	74.800000
2	3	Aarav	M	IT	0.088889	0.445752	1.000000	0.552553	0.434211	1.000000	0.726125	82.700654
3	4	Atharv	M	IT	0.155556	0.777778	0.500000	0.189189	0.131579	0.805195	0.245642	72.000000
4	5	Vivaan	M	Comp	0.711111	0.800000	0.460208	0.405405	0.000000	0.480519	0.000000	66.529412
5	6	Advik	M	ENTC	0.911111	1.000000	0.529412	1.000000	0.434211	0.561039	0.651558	81.040000
6	7	Ansh	M	ENTC	1.000000	0.755556	0.852941	0.486486	0.723684	0.561039	0.678500	81.640000
7	8	Ishaan	M	ENTC	0.488889	0.266667	0.000000	0.540541	0.617690	0.428571	0.074514	68.188889
8	9	Dhruv	M	ENTC	0.222222	0.445752	0.460208	0.918919	0.434211	0.129870	0.081352	68.341176
9	10	Siddarth	M	ENTC	0.955556	0.288889	0.352941	0.864865	0.617690	0.000000	0.002671	66.588889
10	11	Vihaan	M	ENTC	0.644444	0.000000	0.117647	0.675676	0.421053	0.701299	0.101955	68.800000
11	12	Arjun	M	IT	0.488889	0.222222	0.029412	0.216216	0.565789	0.545455	0.003170	66.600000
12	13	Aarush	M	IT	0.311111	0.044444	0.441176	0.552553	0.881579	0.818182	0.572930	79.288889
13	14	Leo	M	IT	1.000000	0.445752	0.117647	0.837838	0.710526	0.561039	0.436556	76.251765
15	16	Sakshi	F	Comp	0.288889	0.800000	0.852941	0.108108	1.000000	0.909091	1.000000	88.800000
16	17	Zaranew	F	Comp	0.888889	0.000000	0.460208	0.324324	0.881579	0.389610	0.188590	70.729412
17	18	Inaya	F	Comp	0.466667	0.288889	0.794118	0.810811	0.842105	0.561039	0.660539	81.240000
18	19	Aarya	F	Comp	0.422222	0.755556	0.529412	0.486486	0.828947	0.714286	0.757528	83.400000
19	20	Pari	F	Comp	0.000000	0.488889	0.441176	0.810811	0.552632	0.561039	0.436027	76.240000

In [37]:
```
new_data2=data
from scipy import stats
column=['Attendance','Phy_marks','Che_marks','EM1_marks','PPS_marks','SME_marks','Total Marks']
new_data2[column] = stats.zscore(new_data2[column])
new_data2
```

Out[37]:

	Rollno	Name	Gender	Branch	Attendance	Phy_marks	Che_marks	EM1_marks	PPS_marks	SME_marks	Total Marks	Percentage
0	1	Mohammed	M	Comp	-0.255386	-0.847468	1.552468	-1.939174	0.951404	-0.238275	-0.053970	74.800000
1	2	Reyansh	M	IT	-1.225853	-0.847468	0.021799	-0.009591	0.900785	-0.459115	-0.053970	74.800000
2	3	Aarav	M	IT	-1.295172	0.050733	1.756557	0.033288	-0.719029	1.859705	1.141116	82.700654
3	4	Atharv	M	IT	-1.087215	1.163626	0.021799	-1.263820	-1.883270	1.031555	-0.477510	72.000000
4	5	Vivaan	M	Comp	0.645762	1.238111	-0.116261	-0.491987	-2.389462	-0.348695	-1.305015	66.529412
5	6	Advik	M	ENTC	1.269633	1.908475	0.123844	1.630553	-0.719029	-0.006393	0.889919	81.040000
6	7	Ansh	M	ENTC	1.546910	1.089141	1.246334	-0.202550	0.394593	-0.006393	0.980677	81.640000
7	8	Ishaan	M	ENTC	-0.047429	-0.549528	-1.712959	-0.009591	-0.013173	-0.569535	-1.053995	68.188889
8	9	Dhruv	M	ENTC	-0.879258	0.050733	-0.116261	1.341116	-0.719029	-1.839364	-1.030959	68.341176
9	10	Siddarth	M	ENTC	1.408272	-0.475043	-0.488424	1.148158	-0.013173	-2.391464	-1.296018	66.588889
10	11	Vihaan	M	ENTC	0.437805	-1.443348	-1.304781	0.472804	-0.769648	0.589875	-0.961556	68.800000
11	12	Arjun	M	IT	-0.047429	-0.698498	-1.610915	-1.167341	-0.212837	-0.072645	-1.294337	66.600000
12	13	Aarush	M	IT	-0.601981	-1.294378	-0.182290	0.033288	1.002023	1.086765	0.625038	79.288889
13	14	Leo	M	IT	1.546910	0.050733	-1.304781	1.051679	0.343974	-0.006393	0.165630	76.251765
15	16	Sakshi	F	Comp	-0.671300	1.238111	1.246334	-1.553257	1.457595	1.473235	2.063730	88.800000
16	17	Zaranew	F	Comp	1.200314	-1.443348	-0.116261	-0.781424	1.002023	-0.735164	-0.669705	70.729412
17	18	Inaya	F	Comp	-0.116748	-0.475043	1.042245	0.955200	0.850165	-0.006393	0.920172	81.240000
18	19	Aarya	F	Comp	-0.255386	1.089141	0.123844	-0.202550	0.799546	0.645085	1.246903	83.400000
19	20	Pari	F	Comp	-1.572448	0.195321	-0.182290	0.955200	-0.263457	-0.006393	0.163850	76.240000