```
import numpy as np
import pandas as pd
```

1. Create 3 list: productId, Product Name, Product Price with 5 elements each. Add this element to create a DataFrame.

```
pdt_Id = [101,102,103,104,105]
pdt_Name = ['Phone','Laptop','Tablet','PC','Modem']
pdt_Price = [45000,75000,50000,170000,25000]
pd.DataFrame(list(zip(pdt_Id,pdt_Name,pdt_Price)),columns=['Product ID','Product
Name','Product Price'])
```

	Product ID	Product Name	Product Price
0	101	Phone	45000
1	102	Laptop	75000
2	103	Tablet	50000
3	104	PC	170000
4	105	Modem	25000

```
dict = {'Product ID':pdt_Id,'Product Price':pdt_Price}
pd.DataFrame(dict,index=pdt_Name)
```

	Product ID	Product Price
Phone	101	45000
Laptop	102	75000
Tablet	103	50000
PC	104	170000
Modem	105	25000

2. Extract one by one column and display seperately from above dataframe

```
df = pd.DataFrame(list(zip(pdt_Id,pdt_Name,pdt_Price)),columns=['Product ID','Product
Name','Product Price'])
print(df['Product ID'],'\n')
print(df['Product Name'],'\n')
print(df['Product Price'])
```

```
0    101
1    102
2    103
3    104
4    105
Name: Product ID, dtype: int64
```

```
0     Phone
1    Laptop
2    Tablet
3        PC
4     Modem
Name: Product Name, dtype: object

0     45000
1     75000
2     50000
3    170000
4     25000
Name: Product Price, dtype: int64
```

3. Display seperately each and every row from an above dataframe (use loc function3)

```
print(df)
print(df.loc[0],'\n')
print(df.loc[1],'\n')
print(df.loc[2],'\n')
print(df.loc[3],'\n')
print(df.loc[4])
```

```
   Product ID Product Name  Product Price
0         101        Phone          45000
1         102       Laptop          75000
2         103       Tablet          50000
3         104           PC         170000
4         105        Modem          25000
Product ID         101
Product Name     Phone
Product Price    45000
Name: 0, dtype: object

Product ID          102
Product Name     Laptop
Product Price     75000
Name: 1, dtype: object

Product ID          103
Product Name     Tablet
Product Price     50000
Name: 2, dtype: object

Product ID          104
Product Name         PC
Product Price    170000
Name: 3, dtype: object

Product ID         105
Product Name     Modem
Product Price    25000
Name: 4, dtype: object
```

```
for i in range(len(df)):
    print(df.loc[i],'\n')
```

```
Product ID         101
Product Name     Phone
Product Price    45000
Name: 0, dtype: object

Product ID          102
Product Name     Laptop
Product Price     75000
Name: 1, dtype: object

Product ID          103
Product Name     Tablet
Product Price     50000
Name: 2, dtype: object

Product ID          104
Product Name         PC
Product Price    170000
Name: 3, dtype: object

Product ID         105
Product Name     Modem
Product Price    25000
Name: 4, dtype: object
```

4. Display data name from 3 row using loc function

```
df.loc[3:6,['Product Name']]
```

	Product Name
3	PC
4	Modem

5. Display price located in 2 row using iloc

```
df.iloc[2,2]
```

```
50000
```

6. Display 2nd to 4th row data excluding price.

```
df
```

	Product ID	Product Name	Product Price
0	101	Phone	45000
1	102	Laptop	75000
2	103	Tablet	50000
3	104	PC	170000
4	105	Modem	25000

```
df.iloc[1:4,0:2]
```

	Product ID	Product Name
1	102	Laptop
2	103	Tablet
3	104	PC

7. Display 1 to 3rd row price data only.

```
df
```

	Product ID	Product Name	Product Price
0	101	Phone	45000
1	102	Laptop	75000
2	103	Tablet	50000
3	104	PC	170000
4	105	Modem	25000

```
df.iloc[1:4,-1]
```

```
1     75000
2     50000
3    170000
Name: Product Price, dtype: int64
```

8. Change marks in Maths column to 41 for students who have scored less than 40.

```
np.random.seed(1)
marks = pd.DataFrame(np.random.randint(0,100,(25,4)),index=range(210,235),columns=
['MATHS','PHYSICS','CHEMISTRY','BIOLOGY'])
print(marks)
```

```
     MATHS  PHYSICS  CHEMISTRY  BIOLOGY
210     37       12         72        9
211     75        5         79       64
212     16        1         76       71
213      6       25         50       20
214     18       84         11       28
215     29       14         50       68
216     87       87         94       96
217     86       13          9        7
218     63       61         22       57
219      1        0         60       81
220      8       88         13       47
221     72       30         71        3
222     70       21         49       57
223      3       68         24       43
224     76       26         52       80
225     41       82         15       64
226     68       25         98       87
227      7       26         25       22
228      9       67         23       27
229     37       57         83       38
230      8       32         34       10
231     23       15         87       25
232     71       92         74       62
233     46       32         88       23
234     55       65         77        3
```

```
marks.loc[marks['MATHS']<40,'MATHS']=41

marks
```

	MATHS	PHYSICS	CHEMISTRY	BIOLOGY
210	41	12	72	9
211	75	5	79	64
212	41	1	76	71
213	41	25	50	20
214	41	84	11	28
215	41	14	50	68
216	87	87	94	96
217	86	13	9	7
218	63	61	22	57
219	41	0	60	81
220	41	88	13	47
221	72	30	71	3

222	70	21	49	57
223	41	68	24	43
224	76	26	52	80
225	41	82	15	64
226	68	25	98	87
227	41	26	25	22
228	41	67	23	27
229	41	57	83	38
230	41	32	34	10
231	41	15	87	25
232	71	92	74	62
233	46	32	88	23
234	55	65	77	3

9. Add 8 marks to all the subjects marks. If any of the marks exceed 100 change it to 100.

```
n_marks=marks[:]+8


n_marks
```

	MATHS	PHYSICS	CHEMISTRY	BIOLOGY
210	49	20	80	17
211	83	13	87	72
212	49	9	84	79
213	49	33	58	28
214	49	92	19	36
215	49	22	58	76
216	95	95	102	104
217	94	21	17	15
218	71	69	30	65
219	49	8	68	89
220	49	96	21	55
221	80	38	79	11

222	78	29	57	65
223	49	76	32	51
224	84	34	60	88
225	49	90	23	72
226	76	33	106	95
227	49	34	33	30
228	49	75	31	35
229	49	65	91	46
230	49	40	42	18
231	49	23	95	33
232	79	100	82	70
233	54	40	96	31
234	63	73	85	11

```
n_marks[n_marks[:]>100]=100
n_marks
```

	MATHS	PHYSICS	CHEMISTRY	BIOLOGY
210	49	20	80	17
211	83	13	87	72
212	49	9	84	79
213	49	33	58	28
214	49	92	19	36
215	49	22	58	76
216	95	95	100	100
217	94	21	17	15
218	71	69	30	65
219	49	8	68	89
220	49	96	21	55
221	80	38	79	11
222	78	29	57	65
223	49	76	32	51
224	84	34	60	88

225	49	90	23	72
226	76	33	100	95
227	49	34	33	30
228	49	75	31	35
229	49	65	91	46
230	49	40	42	18
231	49	23	95	33
232	79	100	82	70
233	54	40	96	31
234	63	73	85	11

10. Display students who have scored more than 50 in maths and 60 in biology.

```
marks
```

	MATHS	PHYSICS	CHEMISTRY	BIOLOGY
210	41	12	72	9
211	75	5	79	64
212	41	1	76	71
213	41	25	50	20
214	41	84	11	28
215	41	14	50	68
216	87	87	94	96
217	86	13	9	7
218	63	61	22	57
219	41	0	60	81
220	41	88	13	47
221	72	30	71	3
222	70	21	49	57
223	41	68	24	43
224	76	26	52	80
225	41	82	15	64
226	68	25	98	87

227	41	26	25	22
228	41	67	23	27
229	41	57	83	38
230	41	32	34	10
231	41	15	87	25
232	71	92	74	62
233	46	32	88	23
234	55	65	77	3

```
# Display students who have scored more than 50 in maths and 60 in biology

n_mat = marks.loc[marks['MATHS']>50,'MATHS']
n_bio = marks.loc[marks['BIOLOGY']>60,'BIOLOGY']
```

```
print(n_mat,n_bio)
```

```
211    75
216    87
217    86
218    63
221    72
222    70
224    76
226    68
232    71
234    55
Name: MATHS, dtype: int32 211    64
212    71
215    68
216    96
219    81
224    80
225    64
226    87
232    62
Name: BIOLOGY, dtype: int32
```

```
marks[(marks['MATHS']>50) & (marks['BIOLOGY'])]
```

	MATHS	PHYSICS	CHEMISTRY	BIOLOGY
217	86	13	9	7
218	63	61	22	57
221	72	30	71	3
222	70	21	49	57

226	68	25	98	87
234	55	65	77	3

11. Display students records whose name starts with letter A

```
std = pd.DataFrame(np.random.randint(0,101,(5,4)),index=
['Abhiraj','Abhishek','Ajin','Rohan','Parita'],columns=
['MATHS','PHYSICS','CHEMISTRY','BIOLOGY'])
```

```
std
```

	MATHS	PHYSICS	CHEMISTRY	BIOLOGY
Abhiraj	0	77	6	52
Abhishek	85	70	2	76
Ajin	91	21	75	7
Rohan	77	72	75	76
Parita	43	20	30	36

```
std[std.index.str.startswith('A')]
```

	MATHS	PHYSICS	CHEMISTRY	BIOLOGY
Abhiraj	0	77	6	52
Abhishek	85	70	2	76
Ajin	91	21	75	7

12. Display students records whose name ends with letter N.

```
std[std.index.str.endswith('n')]
```

	MATHS	PHYSICS	CHEMISTRY	BIOLOGY
Ajin	91	21	75	7
Rohan	77	72	75	76

13. Calculate total and percentage for all the students.

```
std
```

	MATHS	PHYSICS	CHEMISTRY	BIOLOGY
Abhiraj	0	77	6	52
Abhishek	85	70	2	76

Ajin	91	21	75	7
Rohan	77	72	75	76
Parita	43	20	30	36

```
std['Total'] = std.sum(axis=1)
print(std)
```

```
          MATHS  PHYSICS  CHEMISTRY  BIOLOGY  Total
Abhiraj      24       82         97        2    205
Abhishek     92       98         10       54    254
Ajin         96       82         86       70    334
Rohan        66       71         48       54    239
Parita       15        5         17       42     79
```

```
std['TOTAL'] = (std['MATHS']+std['PHYSICS']+std['CHEMISTRY']+std['BIOLOGY'])
std['PERCENTAGE'] = ((std['TOTAL']/400)*100)
```

```
std
```

	MATHS	PHYSICS	CHEMISTRY	BIOLOGY	TOTAL	PERCENTAGE
Abhiraj	0	77	6	52	135	33.75
Abhishek	85	70	2	76	233	58.25
Ajin	91	21	75	7	194	48.50
Rohan	77	72	75	76	300	75.00
Parita	43	20	30	36	129	32.25

14. Calculate Status (PASS/FAIL) for the students.

```
# std['STATUS'] = ['PASS' if marks >=160 else 'FAIL' for marks in std['TOTAL']]
# print(std)
```

```
          MATHS  PHYSICS  CHEMISTRY  BIOLOGY  Total  TOTAL  PERCENTAGE STATUS
Abhiraj      24       82         97        2    205    205       51.25   PASS
Abhishek     92       98         10       54    254    254       63.50   PASS
Ajin         96       82         86       70    334    334       83.50   PASS
Rohan        66       71         48       54    239    239       59.75   PASS
Parita       15        5         17       42     79     79       19.75   FAIL
```

```
std['STATUS'] = np.where((std['TOTAL'] < 160) | (std[['MATHS', 'PHYSICS', 'CHEMISTRY',
'BIOLOGY']].min(axis=1) < 40), 'FAIL', 'PASS')
```

```
std
```

	MATHS	PHYSICS	CHEMISTRY	BIOLOGY	TOTAL	PERCENTAGE	STATUS

Abhiraj	0	77	6	52	135	33.75	FAIL
Abhishek	85	70	2	76	233	58.25	FAIL
Ajin	91	21	75	7	194	48.50	FAIL
Rohan	77	72	75	76	300	75.00	PASS
Parita	43	20	30	36	129	32.25	FAIL

15. Calculate percentage based grade value

per>=90 then A-grade,

per>=80 then B-grade,

per>=70 then C-grade,

per>=60 then D-grade,

per>=50 then E-grade,

per<50 then F-grade.

```
std.loc[std['PERCENTAGE']<50,'GRADE'] = 'F'
std.loc[std['PERCENTAGE']>=50,'GRADE'] = 'E'
std.loc[std['PERCENTAGE']>=60,'GRADE'] = 'D'
std.loc[std['PERCENTAGE']>=70,'GRADE'] = 'C'
std.loc[std['PERCENTAGE']>=80,'GRADE'] = 'B'
std.loc[std['PERCENTAGE']>=90,'GRADE'] = 'A'

print(std)
```

```
          MATHS  PHYSICS  CHEMISTRY  BIOLOGY  TOTAL  PERCENTAGE STATUS GRADE
Abhiraj       0       77          6       52    135       33.75   FAIL     F
Abhishek     85       70          2       76    233       58.25   FAIL     E
Ajin         91       21         75        7    194       48.50   FAIL     F
Rohan        77       72         75       76    300       75.00   PASS     C
Parita       43       20         30       36    129       32.25   FAIL     F
```

```
std['GRADE'] = std['PERCENTAGE'].apply(lambda x:'A' if x >=90 else ('B' if x >=80 else
('C' if x >=70 else ('D' if x >=60 else ('E' if x >=50 else 'F')))))

print(std)
```

```
          MATHS  PHYSICS  CHEMISTRY  BIOLOGY  TOTAL  PERCENTAGE STATUS GRADE
Abhiraj       0       77          6       52    135       33.75   FAIL     F
Abhishek     85       70          2       76    233       58.25   FAIL     E
Ajin         91       21         75        7    194       48.50   FAIL     F
Rohan        77       72         75       76    300       75.00   PASS     C
Parita       43       20         30       36    129       32.25   FAIL     F
```

16. Also calculate overall columns total, maximum, minimum and standard deviation for each numeric columns.

```
marks
```

	MATHS	PHYSICS	CHEMISTRY	BIOLOGY
210	41	12	72	9
211	75	5	79	64
212	41	1	76	71
213	41	25	50	20
214	41	84	11	28
215	41	14	50	68
216	87	87	94	96
217	86	13	9	7
218	63	61	22	57
219	41	0	60	81
220	41	88	13	47
221	72	30	71	3
222	70	21	49	57
223	41	68	24	43
224	76	26	52	80
225	41	82	15	64
226	68	25	98	87
227	41	26	25	22
228	41	67	23	27
229	41	57	83	38
230	41	32	34	10
231	41	15	87	25
232	71	92	74	62
233	46	32	88	23
234	55	65	77	3

```
overall_total = marks.sum()  # Overall total for each column
overall_max = marks.max()  # Overall maximum for each column
overall_min = marks.min()  # Overall minimum for each column
overall_std = marks.std()  # Overall standard deviation for each column
overall_var = marks.var()  # Overall variance for each column
```

```
# Print the results
print("Overall Column Total:")
print(overall_total)
print("\nOverall Column Maximum:")
print(overall_max)
print("\nOverall Column Minimum:")
print(overall_min)
print("\nOverall Column Standard Deviation:")
print(overall_std)
print("\nOverall Column Variance:")
print(overall_var)
```

```
Overall Column Total:
MATHS        1343
PHYSICS      1028
CHEMISTRY    1336
BIOLOGY      1092
dtype: int64

Overall Column Maximum:
MATHS        87
PHYSICS      92
CHEMISTRY    98
BIOLOGY      96
dtype: int32

Overall Column Minimum:
MATHS        41
PHYSICS       0
CHEMISTRY     9
BIOLOGY       3
dtype: int32

Overall Column Standard Deviation:
MATHS        16.599498
PHYSICS      30.512730
CHEMISTRY    29.271829
BIOLOGY      28.539038
dtype: float64

Overall Column Variance:
MATHS        275.543333
PHYSICS      931.026667
CHEMISTRY    856.840000
BIOLOGY      814.476667
dtype: float64
```

17. Display meta-data for each column for the above dataframe.

```
marks.describe()
```

	MATHS	PHYSICS	CHEMISTRY	BIOLOGY
count	25.000000	25.00000	25.000000	25.000000
mean	53.720000	41.12000	53.440000	43.680000
std	16.599498	30.51273	29.271829	28.539038
min	41.000000	0.00000	9.000000	3.000000
25%	41.000000	15.00000	24.000000	22.000000
50%	41.000000	30.00000	52.000000	43.000000
75%	70.000000	67.00000	77.000000	64.000000
max	87.000000	92.00000	98.000000	96.000000

18. Convert the above DataFrame to list within a dictionary.

```
std_dict = marks.to_dict(orient='list')

# Print the dictionary
print("Dictionary with each column as a list:")
print(std_dict)
```

```
Dictionary with each column as a list:
{'MATHS': [41, 75, 41, 41, 41, 41, 87, 86, 63, 41, 41, 72, 70, 41, 76, 41, 68, 41, 41,
41, 41, 41, 71, 46, 55], 'PHYSICS': [12, 5, 1, 25, 84, 14, 87, 13, 61, 0, 88, 30, 21,
68, 26, 82, 25, 26, 67, 57, 32, 15, 92, 32, 65], 'CHEMISTRY': [72, 79, 76, 50, 11, 50,
94, 9, 22, 60, 13, 71, 49, 24, 52, 15, 98, 25, 23, 83, 34, 87, 74, 88, 77], 'BIOLOGY':
[9, 64, 71, 20, 28, 68, 96, 7, 57, 81, 47, 3, 57, 43, 80, 64, 87, 22, 27, 38, 10, 25,
62, 23, 3]}
```

```
dict = df.to_dict(orient='list')
print(dict)
```

```
{'Product ID': [101, 102, 103, 104, 105], 'Product Name': ['Phone', 'Laptop',
'Tablet', 'PC', 'Modem'], 'Product Price': [45000, 75000, 50000, 170000, 25000]}
```

19. Convert this dictionary again to a separate dataframe by adding 500 to each element in product price column

```
df['Product Price']+=500
```

```
df
```

	Product ID	Product Name	Product Price
0	101	Phone	45500
1	102	Laptop	75500
2	103	Tablet	50500

3	104	PC	170500
4	105	Modem	25500