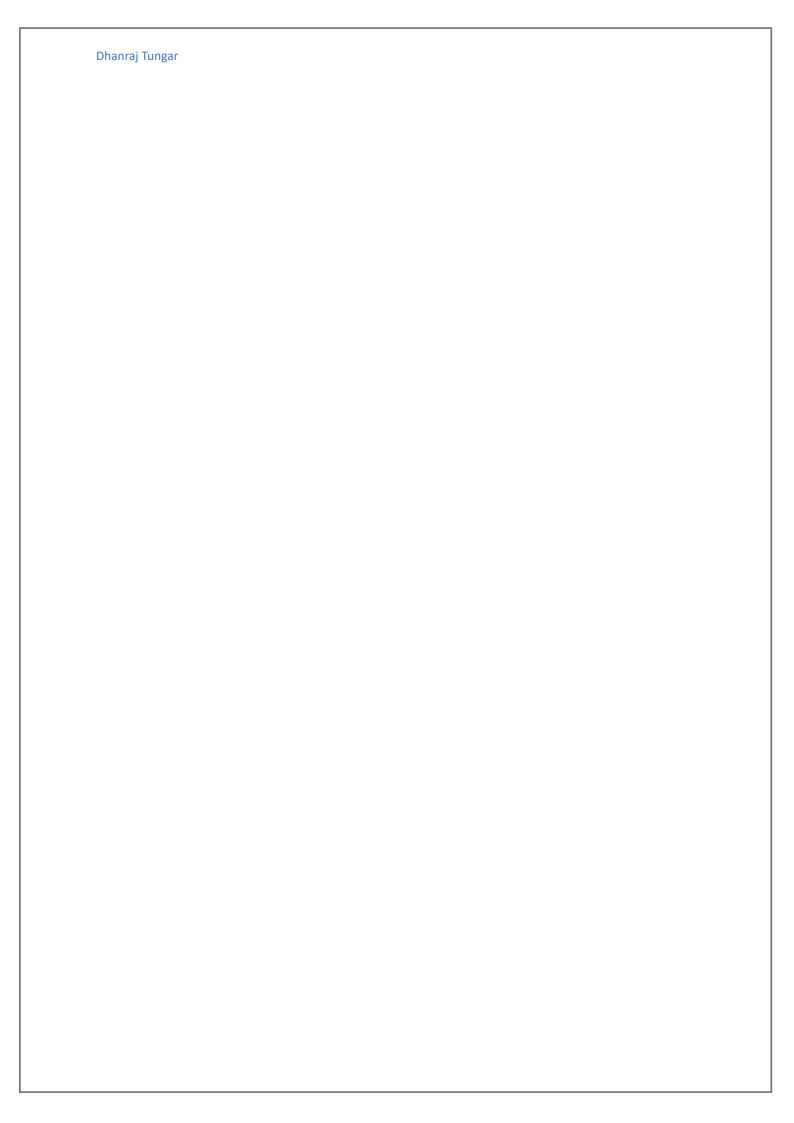
Dhanraj Tungar
Data analytics completed in Lab.
(Assignment 2)
Dhanraj Tungar TY-BCA (A) 2022018100094671
2022016100094671



import pandas as pd

df = pd.read\_csv('Student\_Performance.csv')

#1. Question: Display the first 5 rows of the DataFrame.

# df.head()

[10	]:					
	Hours Studied	Previous Scores	Extracurricular Activities	Sleep Hours	Sample Question Papers Practiced	Performance Index
0	7	99	Yes	9	1	91.0
1	4	82	No	4	2	65.0
2	8	51	Yes	7	2	45.0
3	5	52	Yes	5	2	36.0
4	7	75	No	8	5	66.0

#2. Question: Display the last 5 rows of the DataFrame.

# df.tail()

	df.ta	il()					
5]:		Hours Studied	Previous Scores	Extracurricular Activities	Sleep Hours	Sample Question Papers Practiced	Performance Index
	9995	1	49	Yes	4	2	23.0
	9996	7	64	Yes	8	5	58.0
	9997	6	83	Yes	8	5	74.0
	9998	9	97	Yes	7	0	95.0
	9999	7	74	No	8	1	64.0

#3. Question: What is the shape of the DataFrame (number of rows and columns)? df.shape

```
[9]: #3. Question: What is the shape of the DataFrame (number of rows and columns)?
df.shape[9]: (10000, 6)
```

#4. Question: Display summary information about the DataFrame, including data types and non-null counts.

# df.info()

```
[9]: #3. Question: What is the shape of the DataFrame (number of rows and columns)?
    df.shape
[9]: (10000, 6)
```

#5. Question: Get descriptive statistics for numerical columns in the DataFrame. df.describe()

E 3	df.des	scribe()				
[6]:		Hours Studied	Previous Scores	Sleep Hours	Sample Question Papers Practiced	Performance Index
	count	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000
	mean	4.992900	69.445700	6.530600	4.583300	55.224800
	std	2.589309	17.343152	1.695863	2.867348	19.212558
	min	1.000000	40.000000	4.000000	0.000000	10.000000
	25%	3.000000	54.000000	5.000000	2.000000	40.000000
	50%	5.000000	69.000000	7.000000	5.000000	55.000000
	75%	7.000000	85.000000	8.000000	7.000000	71.000000
	max	9.000000	99.000000	9.000000	9.000000	100.000000

#6. Question: Select the "Hours Studied" column from the DataFrame. df['Hours Studied']

```
df['Hours Studied']
[11]: 0
              4
      2
              8
      3
              5
      4
              7
      9995
              1
      9996
              7
      9997
      9998
              9
      9999
      Name: Hours Studied, Length: 10000, dtype: int64
```

#7. Question: Select the "Hours Studied" and "Performance Index" columns from the DataFrame.

df[['Hours Studied','Performance Index']]

[14]:	#7. Question: Select the "Hours Studied" and "Performance Index" columns from the DataFrame.
	<pre>df[['Hours Studied','Performance Index']]</pre>

[14]:		Hours Studied	Performance Index
	0	7	91.0
	1	4	65.0
	2	8	45.0
	3	5	36.0
	4	7	66.0
	9995	1	23.0
	9996	7	58.0
	9997	6	74.0
	9998	9	95.0
	9999	7	64.0

10000 rows × 2 columns

#8. Question: Select the first row from the DataFrame by index.

# df.iloc[0]

```
•[31]: #8. Question: Select the first row from the DataFrame by index.

df.iloc[0]

[31]: 1
```

#9. Question: Select the row with label/index 0 from the DataFrame.

# df.loc[0,'Hours Studied']

```
[30]: #9. Question: Select the row with label/index 0 from the DataFrame.

df.loc[0,'Hours Studied']

[30]: 7
```

#10. Question: Select rows from 1 to 5 and columns from 2 to 3.

# df.iloc[1:5, 2:4]

	#10. Question: Select rows from 1 to 5 and columns from 2 to 3. df.iloc[1:5 , 2:4]							
[26]:		Extracurricular Activities	Sleep Hours					
	1	No	4					
	2	Yes	7					
	3	Yes	5					
	4	No	8					

#11. Question: Filter the DataFrame for rows where "Performance Index" is greater than 80.  $df[df['Performance\ Index'] > 80]$ 

	Hours Studied	Previous Scores	Extracurricular Activities	Sleep Hours	Sample Question Papers Practiced	Performance Index
0	7	99	Yes	9	1	91.0
10	8	91	No	4	5	84.0
21	6	96	No	9	0	85.0
27	9	84	Yes	6	6	83.0
55	6	99	No	4	7	91.0
9976	8	93	Yes	9	8	91.0
9977	9	84	No	6	6	82.0
9985	8	99	No	5	5	92.0
9991	5	97	Yes	7	4	83.0
9998	9	97	Yes	7	0	95.0

#12. Question: Use the query method to filter rows where "Sleep Hours" is greater than 6. df.query('`Sleep Hours` > 6')

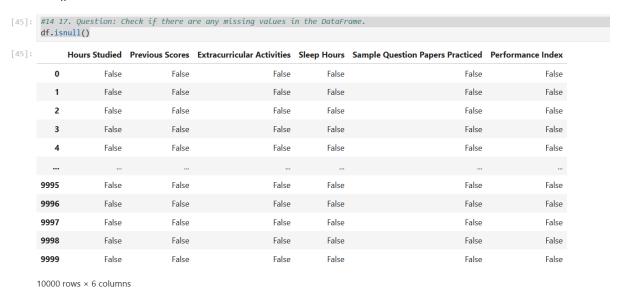
	Hours Studied	Previous Scores	Extracurricular Activities	Sleep Hours	Sample Question Papers Practiced	Performance Index
0	7	99	Yes	9	1	91.0
2	8	51	Yes	7	2	45.0
4	7	75	No	8	5	66.0
5	3	78	No	9	6	61.0
8	5	77	No	8	2	61.0
	•••					
9994	6	46	Yes	8	0	39.0
9996	7	64	Yes	8	5	58.0
9997	6	83	Yes	8	5	74.0
9998	9	97	Yes	7	0	95.0
9999	7	74	No	8	1	64.0

#13. Question: Sort the DataFrame by the "Sample Question Papers Practiced" column in ascending order.

df.sort\_values('Sample Question Papers Practiced')

	Hours Studied	Previous Scores	Extracurricular Activities	Sleep Hours	Sample Question Papers Practiced	Performance Index
5932	4	81	No	7	0	61.0
1007	3	65	No	7	0	44.0
5331	8	45	No	9	0	38.0
5334	2	43	No	9	0	22.0
5345	1	55	Yes	4	0	25.0
2738	1	41	No	5	9	17.0
8672	4	87	Yes	5	9	75.0
4574	1	70	Yes	8	9	45.0
2728	1	72	Yes	5	9	45.0
2696	2	64	No	6	9	42.0

# #14. Question: Check if there are any missing values in the DataFrame. df.isnull()



# #15. Question: Count the number of missing values in each column.

# df.isnull().sum()

# #16 Question: Fill missing values with 0.

# df.fillna(0)

	lna(0)					
	Hours Studied	Previous Scores	Extracurricular Activities	Sleep Hours	Sample Question Papers Practiced	Performance Index
0	7	99	Yes	9	1	91.0
1	4	82	No	4	2	65.0
2	8	51	Yes	7	2	45.0
3	5	52	Yes	5	2	36.0
4	7	75	No	8	5	66.0
9995	1	49	Yes	4	2	23.0
9996	7	64	Yes	8	5	58.0
9997	6	83	Yes	8	5	74.0
9998	9	97	Yes	7	0	95.0
9999	7	74	No	8	1	64.0

10000 rows × 6 columns

# #17 Sorting data: Sort the dataframe based on a specific column. df.sort\_values(by='Previous Scores', ascending=False)

	Hours Studied	<b>Previous Scores</b>	Extracurricular Activities	Sleep Hours	Sample Question Papers Practiced	Performance Index
0	7	99	Yes	9	1	91.0
1025	5	99	No	6	3	87.0
8495	4	99	No	5	3	83.0
9686	6	99	Yes	5	2	89.0
8534	1	99	No	8	1	75.0
5105	4	40	No	6	2	18.0
4467	7	40	No	6	2	33.0
7860	2	40	No	7	8	18.0
6041	4	40	No	9	4	21.0
1210	8	40	Yes	6	7	34.0

# #18 Filter rows where the Student had Extracurricular Activities df[df['Extracurricular Activities']== 'Yes']

	Hours Studied	Previous Scores	Extracurricular Activities	Sleep Hours	Sample Question Papers Practiced	Performance Index		
0	7	99	Yes	9	1	91.0		
2	8	51	Yes	7	2	45.0		
3	5	52	Yes	5	2	36.0		
6	7	73	Yes	5	6	63.0		
7	8	45	Yes	4	6	42.0		
9994	6	46	Yes	8	0	39.0		
9995	1	49	Yes	4	2	23.0		
9996	7	64	Yes	8	5	58.0		
9997	6	83	Yes	8	5	74.0		
9998	9	97	Yes	7	0	95.0		

4948 rows × 6 columns

df

#19 # Convert 'Yes'/'No' responses to 1/0

df = pd.read\_csv('Student\_Performance.csv')

df['Extracurricular Activities'] = df['Extracurricular Activities'].map({ 'Yes':1 , 'No':0 }).fillna(0)

df	'Extracurricular Activities'] = df['Extracurricular Activities'].map({ 'Yes':1 , 'No':0 }).fillna(0)									
	Hours Studied	Previous Scores	Extracurricular Activities	Sleep Hours	Sample Question Papers Practiced	Performance Index				
0	7	99	1	9	1	91.0				
1	4	82	0	4	2	65.0				
2	8	51	1	7	2	45.0				
3	5	52	1	5	2	36.0				
4	7	75	0	8	5	66.0				
9995	1	49	1	4	2	23.0				
9996	7	64	1	8	5	58.0				
9997	6	83	1	8	5	74.0				
9998	9	97	1	7	0	95.0				
9999	7	74	0	8	1	64.0				

#20 Find all rows where Hours Studied is > "7" and Performance Index >= 80 df[(df['Hours Studied'] > 7) & (df['Performance Index'] >= 80)]

	<pre>df[(df['Hours Studied'] &gt; 7) &amp; (df['Performance Index'] &gt;= 80)]</pre>								
	Hours Studied	Previous Scores	Extracurricular Activities	Sleep Hours	Sample Question Papers Practiced	Performance Index			
10	8	91	0	4	5	84.0			
27	9	84	1	6	6	83.0			
61	9	97	1	8	5	98.0			
62	8	92	1	4	7	87.0			
68	9	94	0	9	1	91.0			
9955	9	88	0	8	2	84.0			
9976	8	93	1	9	8	91.0			
9977	9	84	0	6	6	82.0			
9985	8	99	0	5	5	92.0			
9998	9	97	1	7	0	95.0			

#21 Count unique values in the 'Sample Question Papers Practiced' column df['Sample Question Papers Practiced'].nunique()

```
• [74]: #21 Count unique values in the 'Sample Question Papers Practiced' column df['Sample Question Papers Practiced'].nunique()

[74]: #21 Count unique values in the 'Sample Question Papers Practiced' column df['Sample Question Papers Practiced'].nunique()
```

df['Sample Question Papers Practiced'].unique()

```
[75]: df['Sample Question Papers Practiced'].unique()
[75]: array([1, 2, 5, 6, 0, 8, 3, 4, 9, 7], dtype=int64)
```

#22 Calculate the average Performance Index df['Performance Index'].mean()



#23 Select the first 3 rows of the 'Previous Scores' and 'Performance Index' columns df.loc[:2,['Performance Index','Previous Scores']]



```
#24 Change Sleep Hours for the student with Hours Studied = 8 to "9 hours"

df.loc[df['Hours Studied']== 8,'Sleep Hours'] = 9

#25 Find the unique value counts

df['Hours Studied'].value_counts()
```

```
#25 Find the unique value counts
• [99]:
        df['Hours Studied'].value counts()
[99]: Hours Studied
             1152
        1
        6
             1133
        7
             1129
        3
             1119
        9
             1115
        5
             1094
        8
             1088
        4
             1085
             1085
        2
        Name: count, dtype: int64
```

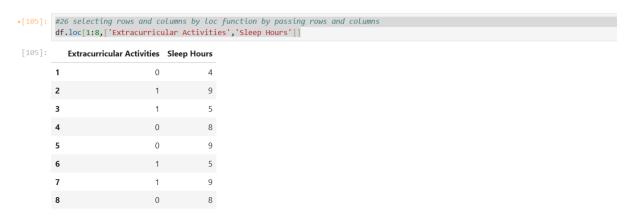
#26 selecting rows and columns by loc function by passing rows and columns df.iloc[[0,1]]



#26 selecting rows and columns by loc function by passing rows and columns df.loc[[1,4,7],['Extracurricular Activities','Sleep Hours']]



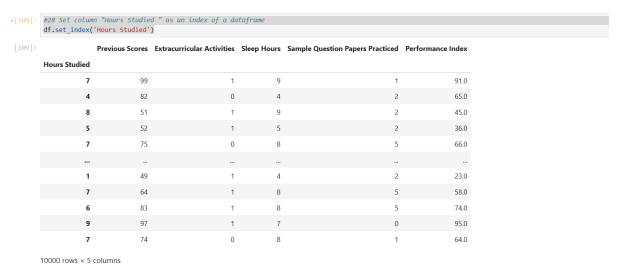
#26 selecting rows and columns by loc function by passing rows and columns df.loc[1:8,['Extracurricular Activities','Sleep Hours']]



#27 selecting rows and columns by loc function by passing rows and columns as range df.loc[1:8:2,'Hours Studied':'Sleep Hours']



#28 Set column "Hours Studied " as an index of a dataframe df.set\_index('Hours Studied')



# df.index

```
[110]: df.index
[110]: RangeIndex(start=0, stop=10000, step=1)
```

### #29 reset index column

df.reset\_index(inplace = True)

df['Performance Index'] <= 99

# df.query('`Performance Index` <= 99')

5]:		II 0			D	s Extracurricular Activities Sleep Hours Sample Question Papers Practiced Performance In					
		level_0	index	Hours Studied	Previous Scores	Extracurricular Activities	Sleep Hours	Sample Question Papers Practiced	Performance Index		
	0	0	0	7	99	1	9	1	91.0		
	1	1	1	4	82	0	4	2	65.0		
	2	2	2	8	51	1	9	2	45.0		
	3	3	3	5	52	1	5	2	36.0		
	4	4	4	7	75	0	8	5	66.0		
9	9995	9995	9995	1	49	1	4	2	23.0		
•	9996	9996	9996	7	64	1	8	5	58.0		
9	9997	9997	9997	6	83	1	8	5	74.0		
9	9998	9998	9998	9	97	1	7	0	95.0		
9	9999	9999	9999	7	74	0	8	1	64.0		

p = df.query('`Performance Index` <= 99')[['Extracurricular Activities','Sample Question Papers Practiced']]

print(p.head())

df[(df['Previous Scores'] == 99) & (df['Performance Index'] == 99)][['Hours Studied','Sample Question Papers Practiced']]



# Evaluating isin() function

filter = df['Performance Index'].isin([80,90,100])

df.loc[filter,'Performance Index']

```
•[135]: # Evaluating isin() function
        filter = df['Performance Index'].isin([80,90,100])
        df.loc[filter,'Performance Index']
[135]: 145
                100.0
        382
                 80.0
        493
                 80.0
        608
                 90.0
        9571
                 80.0
        9642
                 80.0
        9829
                 80.0
        9887
        Name: Performance Index, Length: 173, dtype: float64
```