XII – DataFrames – Work sheet on Creation of the DataFrame & Different Operations

	RNo	Name	Marks
A	1	Pradeep	90
B	2	Sudeep	80
C	3	Mohan	95
D	4	Pavan	85

Df

CREATION OF DATAFRAME

(1) From a 2D dictionary having values as lists: (One of easy method)

```
import pandas as pd
dict={'RNo':[1,2,3,4],'Name':['Pradeep','Sudeep','Mohan','Pavan'],'Marks':[90,80,95,85]}
df=pd.DataFrame(dict,index=['A','B','C','D'])
print(df)
```

(2) From a 2D dictionary having values as dictionary:

```
import pandas as pd
dict={'RNo':{'A':1,'B':2,'C':3,'D':4},'Name':{'A':'Pradeep','B':'Sudeep','C':'Mohan','D':'Pavan'},
      'Marks':{'A':90,'B':80,'C':95,'D':85}}
df=pd.DataFrame(dict)
print(df)
```

(3) From List of Dictionaries

(Each row details should store as one dictionary, whose keys are column indexes)

```
import pandas as pd
dict1={'RNo':1,'Name':'Pradeep','Marks':90}
dict2={'RNo':2,'Name':'Sudeep','Marks':80}
dict3={'RNo':3,'Name':'Mohan','Marks':95}
dict4={'RNo':4,'Name':'Pavan','Marks':85}
Stu=[dict1,dict2,dict3,dict4]
df=pd.DataFrame(Stu,index=['A','B','C','D'])
print(df)
```

(4) From List of Lists (Very easy method)

(Need to take data row wise. Need to give row indexes and column indexes)

```
import pandas as pd
L=[[1,'Pradeep',90],[2,'Sudeep',80],[3,'Mohan',95],[4,'Pavan',85]]
df=pd.DataFrame(L,index=['A','B','C','D'],columns=['RNo','Name','Marks'])
print(df)
```

(5) From a 2D Dictionary with Values as Series Objects:

```
import pandas as pd
R=pd.Series([1,2,3,4])
N=pd.Series(['Pradeep','Sudeep','Mohan','Pavan'])
M=pd.Series([90,80,95,85])
dict={'RNo':R,'Name':N,'Marks':M}
df=pd.DataFrame(dict)
print(df)
```

Note : In case, index also give while we are creating a datafram, df=pd.DataFrame(dict,index=['A','B','C','D'])
All data will be displayed as NaN

(6) From a 2-D ndarray:

	S1	S2	S3	S4
RNo	1	2	3	4
Marks	90	80	95	85

```
import numpy as np
import pandas as pd
narr=np.array([[1,2,3,4],[90,80,95,85]],np.int32)
df=pd.DataFrame(narr,index=['RNo','Marks'],columns=['S1','S2','S3','S4'])
print(df)
```

DataFrame Operations (TT)

	P1	P2	P3
Mon	IP	Maths	Eng
Tue	Che	Phy	Acc
Wed	IP	Phy	Maths

(1) Create the above DataFrame "TT"

```
import pandas as pd
D={"P1":["IP","Che","IP"],"P2":["Maths","Phy","Phy"],"P3":["Eng","Acc","Maths"]}
TT=pd.DataFrame(D,index=["Mon","Tue","Wed"])
print(TT)
```

(2) Display only the details of First 2 Rows.

	P1	P2	P3
Mon	IP	Maths	Eng
Tue	Che	Phy	Acc

```
TT.head(2)                  # Using head() method
TT.iloc[0:2]                # Using iloc with slicing (index-based selection)
TT.loc[["Mon","Tue"]] # Using loc with index labels
TT[:2]                      # Using slicing directly on the DataFrame (not recommended for all cases, but works here)
TT.loc["Mon":"Tue"] # It uses label-based slicing in loc, and in Pandas, the end label is inclusive when using loc.
```

(3) Display the details of rows Wed and Mon (in the order Wed, Mon)

	P1	P2	P3
Wed	IP	Phy	Maths
Mon	IP	Maths	Eng

```
TT.loc[['Wed','Mon']]       # Using loc with a list (order maintained):
TT.iloc[[2,0]]              # Using iloc by finding their positions manually
TT.reindex(["Wed", "Mon"])  # Using reindex() to reorder explicitly
```

(4) Display the details of Columns "P1" and "P3"

	P1	P3
Mon	IP	Eng
Tue	Che	Acc
Wed	IP	Maths

```
TT[["P1", "P3"]]            # Using double square brackets to specify multiple columns (most common)
TT.loc[:,["P1","P3"]]       # Using loc with all rows and selected columns:
TT.iloc[:,[0,2]]            # Using iloc if you know the positions of columns:
```

- "P1" is column 0
- "P3" is column 2

```
TT.reindex(columns=["P1", "P3"])        # Using reindex on columns:
```

(5) Display the following output.

we need:

- **Rows:** `"Tue"` and `"Mon"` (in this order)
- **Columns:** `"P3"`, `"P1"`, and `"P2"` (in this order)

	P3	P1	P2
Tue	Acc	Che	Phy
Mon	Eng	IP	Maths

```
TT.loc[["Tue","Mon"],["P3","P1","P2"]]      # Using .loc with both row and column reordering:
TT.iloc[[1, 0], [2, 0, 1]]                  #Using .iloc with positions:

TT.reindex(index=["Tue", "Mon"], columns=["P3", "P1", "P2"])
                                            #Using .reindex() for rows and columns:
```

(6) Display the following Output.

- **Rows:** All three in order → `["Mon", "Tue", "Wed"]`
- **Columns:** Only `["P3", "P2"]`

```
          P3      P2
Mon      Eng   Maths
Tue      Acc     Phy
Wed    Maths     Phy
```

```
TT.loc[["Mon", "Tue", "Wed"], ["P3", "P2"]]   #Using .loc with label-based indexing:
TT.loc["Mon":"Wed",["P3","P2"]]
TT.loc[:,["P3","P2"]]
TT.iloc[[0, 1, 2], [2, 1]]                    # Using .iloc with integer positions:
TT.iloc[:,[2,1]]
TT.iloc[0:3,[2,1]]
TT.reindex(index=["Mon", "Tue", "Wed"], columns=["P3", "P2"])
                                              # Using .reindex() with rows and columns:
```

```
          P1      P2      P3
Mon       IP   Maths     Eng
Tue      Che     Phy     Acc
Wed       IP     Phy   Maths
```

(7) Change Tuesday's 3rd Period to "Eng"

```
          P1      P2      P3
Mon       IP   Maths     Eng
Tue      Che     Phy     Eng
Wed       IP     Phy   Maths
```

```
TT.at["Tue","P3"]="Eng"          #Using at (fastest for single value assignment):
TT.iat[1,2]="Eng"
TT.loc["Tue", "P3"] = "Eng"      #Using loc (label-based selection):
TT.at["Tue", "P3"] = "Eng"
TT.iloc[1, 2] = "Eng"            #Using iloc (index-based):
```

- Row `"Tue"` is index `1`
- Column `"P3"` is index `2`

(8) Add a new column "P4" with values Acc,BS,Maths

```
          P1      P2      P3      P4
Mon       IP   Maths     Eng     Acc
Tue      Che     Phy     Eng      BS
Wed       IP     Phy   Maths   Maths
```

```
TT["P4"]="Acc","BS","Maths"
TT["P4"] = ["Acc", "BS", "Maths"]
TT.loc[:, "P4"] = ["Acc", "BS", "Maths"]      #Using .loc with slicing
TT.iloc[:, 3] = ["Acc", "BS", "Maths"]        #Using .iloc with column index (if you know index)

#Using .at[] for each row
TT.at["Mon", "P4"] = "Acc"
TT.at["Tue", "P4"] = "BS"
TT.at["Wed", "P4"] = "Maths"

# Using .loc[] for each row
TT.loc["Mon", "P4"] = "Acc"
TT.loc["Tue", "P4"] = "BS"
TT.loc["Wed", "P4"] = "Maths"
```

(9) Add a row with index Thu, values Bio,Che,Eng,Acc

```
          P1      P2      P3      P4
Mon       IP   Maths     Eng     Acc
Tue      Che     Phy     Eng      BS
Wed       IP     Phy   Maths   Maths
Thu      Bio     Che     Eng     Acc
```

```
TT.loc["Thu"]="Bio","Che","Eng","Acc"          # Using .loc[]
```

```
TT.loc["Thu"] = ["Bio", "Che", "Eng", "Acc"]
TT.loc["Thu"] = {"P1": "Bio", "P2": "Che", "P3": "Eng", "P4": "Acc"}
                                                    # Using .loc[] with dictionary
TT.loc["Thu", :] = ["Bio", "Che", "Eng", "Acc"]     # Using DataFrame.loc[] with slicing
```

(10) Rename Mon to Monday

	P1	P2	P3	P4
Monday	IP	Maths	Eng	Acc
Tue	Che	Phy	Eng	BS
Wed	IP	Phy	Maths	Maths
Thu	Bio	Che	Eng	Acc

```
TT.rename(index={"Mon":"Monday"},inplace=True)
TT = TT.rename(index={"Mon": "Monday"})
```

(11) Rename P2 to Period2

	P1	Period2	P3	P4
Monday	IP	Maths	Eng	Acc
Tue	Che	Phy	Eng	BS
Wed	IP	Phy	Maths	Maths
Thu	Bio	Che	Eng	Acc

```
TT.rename(columns={"P2":"Period2"},inplace=True)
TT = TT.rename(columns={"P2": "Period2"})
```

(12) Delete the Column P3

```
TT.drop("P3",axis=1,inplace=True)

TT = TT.drop("P3", axis=1)
TT = TT.drop(columns="P3")
del TT["P3"]
```

	P1	Period2	P4
Monday	IP	Maths	Acc
Tue	Che	Phy	BS
Wed	IP	Phy	Maths
Thu	Bio	Che	Acc

(13) Delete Row with Index "Tue"

	P1	Period2	P4
Monday	IP	Maths	Acc
Wed	IP	Phy	Maths
Thu	Bio	Che	Acc

```
TT=TT.drop("Tue")
TT.drop("Tue",inplace=True)
TT.drop("Tue",axis=0,inplace=True)
TT = TT.drop("Tue", axis=0)
TT = TT.drop(index="Tue")
TT = TT.drop(["Tue"])
```

(14) Modify the Entire P4 values to “Acc”

	P1	Period2	P4
Monday	IP	Maths	Acc
Wed	IP	Phy	Acc
Thu	Bio	Che	Acc

```
    TT.loc[:,"P4"]="Acc"
    TT.iloc[:,1:3]=”Acc”
TT["P4"] = "Acc"
```

(15) Delete complete DataFrame

```
del TT
```