EXPERIMENT NO.11

Nidhi Kochar,Batch-S13,Roll no.44

AIM-Write python programs to implement Pandas and its various functions

THEORY:

In computer programming, pandas is a software library written for the Python programming language for data manipulation and analysis.

In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. The name is derived from the term "panel data", an econometrics term for data sets that include observations over multiple time periods for the same individuals.

Its name is a play on the phrase "Python data analysis" itself. Wes McKinney started building what would become pandas at AQR

LIBRARY FEATURES:

1.DataFrame object for data manipulation with integrated indexing.

2.Tools for reading and writing data between in-memory data structures and different file formats.

3.Data alignment and integrated handling of missing data.

4.Reshaping and pivoting of data sets.

5.Label-based slicing, fancy indexing, and subsetting of large data sets.

6.Data structure column insertion and deletion.

7.Group by engine allowing split-apply-combine operations on data sets.

8.Data set merging and joining.

9.Hierarchical axis indexing to work with high-dimensional data in a lower-dimensional data structure.

10.Time series-functionality: Date range generation and frequency conversion, moving window statistics, moving window

11.linear regressions, date shifting and lagging.

12.Provides data filtration.

13.The library is highly optimized for performance, with critical code paths written in Cython or C.

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

In [3]:
```
pd.Series([13,17,19],index=['P1','P2','P3'])
```

Out[3]:
```
P1    13
P2    17
P3    19
dtype: int64
```

In [29]:
```
h=pd.Series([13,17,19],index=['P1','P2','P3'])
h[1:]#Slicing value from index location 1 till the end
```

Out[29]:
```
P2    17
P3    19
dtype: int64
```

In [31]:
```
h.loc['P2']#Using explicit indexing
```

Out[31]:
```
17
```

In [32]:
```
h.iloc[1]#Using implicit indexing
```

Out[32]:
```
17
```

In [7]:
```
d = {'Prime nos': [13,17,19], 'Composite Nos.': [4,6,20]}
df = pd.DataFrame(data=d)
df#Creates table
```

Out[7]:

	Prime nos	Composite Nos.
0	13	4
1	17	6
2	19	20

In [12]:
```
df.values#Prints all the values
```

Out[12]:
```
array([[13,  4],
       [17,  6],
       [19, 20]], dtype=int64)
```

In [11]:
```
df.index#Prints row labels of Dataframe
```

Out[11]:
```
RangeIndex(start=0, stop=3, step=1)
```

In [15]:
```
df.columns#Prints the names of all coloumns
```

Out[15]:
```
Index(['Prime nos', 'Composite Nos.'], dtype='object')
```

In [19]:
```
import numpy as np
g=np.arange(14).reshape(7,2)
g#Creating a numpy array
```

Out[19]:
```
array([[ 0,  1],
       [ 2,  3],
       [ 4,  5],
       [ 6,  7],
       [ 8,  9],
       [10, 11],
       [12, 13]])
```

In [20]:
```
pd.DataFrame(g,columns=['A','B'])#Converting the numpy array to DataFrame
```

Out[20]:

	A	B
0	0	1
1	2	3
2	4	5
3	6	7
4	8	9
5	10	11
6	12	13

In [34]:
```
df = pd.read_csv("student_records.csv")
df#Reading the csv file from the records
```

Out[34]:

	Name	OverallGrade	Obedient	ResearchScore	ProjectScore	Recommend
0	Henry	A	Y	90	85	Yes
1	John	C	N	85	51	Yes
2	David	F	N	10	17	No
3	Holmes	B	Y	75	71	No
4	Marvin	E	N	20	30	No
5	Simon	A	Y	92	79	Yes
6	Robert	B	Y	60	59	No
7	Trent	C	Y	75	33	No

In [36]:
```
df.head()# Prints first 5 values of the table
```

Out[36]:

	Name	OverallGrade	Obedient	ResearchScore	ProjectScore	Recommend
0	Henry	A	Y	90	85	Yes
1	John	C	N	85	51	Yes
2	David	F	N	10	17	No
3	Holmes	B	Y	75	71	No
4	Marvin	E	N	20	30	No

In [37]:
```
df.tail()# prints last 5 values of the Table
```

Out[37]:

	Name	OverallGrade	Obedient	ResearchScore	ProjectScore	Recommend
3	Holmes	B	Y	75	71	No
4	Marvin	E	N	20	30	No
5	Simon	A	Y	92	79	Yes
6	Robert	B	Y	60	59	No
7	Trent	C	Y	75	33	No

In [38]:
```
df.describe#Describes the whole table
```

Out[38]:
```
<bound method NDFrame.describe of      Name OverallGrade Obedient  ResearchScor
e  ProjectScore Recommend
0   Henry            A        Y            90            85       Yes
1    John            C        N            85            51       Yes
2   David            F        N            10            17        No
3  Holmes            B        Y            75            71        No
4  Marvin            E        N            20            30        No
5   Simon            A        Y            92            79       Yes
6  Robert            B        Y            60            59        No
7   Trent            C        Y            75            33        No>
```

In [39]:
```
df.shape#Prints the rows and columns of table
```

Out[39]:
```
(8, 6)
```

In [40]:
```
df.info#Similar to describe functionality
```

Out[40]:
```
<bound method DataFrame.info of      Name OverallGrade Obedient  ResearchScore
ProjectScore Recommend
0   Henry            A        Y            90            85       Yes
1    John            C        N            85            51       Yes
2   David            F        N            10            17        No
3  Holmes            B        Y            75            71        No
4  Marvin            E        N            20            30        No
5   Simon            A        Y            92            79       Yes
6  Robert            B        Y            60            59        No
7   Trent            C        Y            75            33        No>
```

In [42]:
```
i=np.arange(12).reshape(3,4)
i#Creating numpy array of 3 rows and 4 coloumns
```

Out[42]:
```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

In [49]:
```
o=pd.DataFrame(i,columns=['A','B','C','D'])
o#Converting the array to DataFrame
```

Out[49]:

	A	B	C	D
0	0	1	2	3
1	4	5	6	7
2	8	9	10	11

In [50]:
```
o_array = o.values
print(o_array)
print(type(o_array))#Prints the type of array
```

```
[[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]]
<class 'numpy.ndarray'>
```

In [52]:
```
from numpy import nan
o.iloc[2,2]=nan
o#Converting one of the values to Nan
```

Out[52]:

	A	B	C	D
0	0	1	2.0	3
1	4	5	6.0	7
2	8	9	NaN	11

In [57]:
```
o.fillna(10)
#Filling the Nan value to specified number
```

Out[57]:

	A	B	C	D
0	0	1	2.0	3
1	4	5	6.0	7
2	8	9	10.0	11

In [78]:
```
x=np.full((3,3),6)
x
```

Out[78]:
```
array([[6, 6, 6],
       [6, 6, 6],
       [6, 6, 6]])
```

In [79]:
```
y=pd.DataFrame(x,columns=['C1','C2','C3'])
y
```

Out[79]:

	C1	C2	C3
0	6	6	6
1	6	6	6
2	6	6	6

In [83]:
```
y.groupby('C2').mean()#Prints the mean of column 2
```

Out[83]:

	C1	C3
C2		
6	6	6

In [112]:
```
import pandas as pd
df = pd.read_csv("record.csv")
df
```

Out[112]:

	Roll No	Name	Math	Python	COA	AT	CNND	Total Score	Percentage
0	1	S1	45	32	67	70	60	274	55.0
1	2	S2	56	34	98	37	80	305	61.0
2	3	S3	67	77	87	53	98	382	76.0
3	4	S4	71	77	43	2	49	243	49.0
4	5	S5	97	77	87	42	96	398	80.0
5	6	S6	50	4	20	35	54	163	33.0
6	7	S7	52	100	40	35	71	298	60.0
7	8	S8	69	12	88	65	23	256	51.0
8	9	S9	98	12	83	50	40	284	57.0
9	10	S10	44	90	32	25	42	233	47.0
10	11	S11	73	69	57	23	10	233	47.0
11	12	S12	22	55	20	4	36	137	27.0
12	13	S13	70	71	15	99	45	300	60.0
13	14	S14	55	34	59	48	21	217	43.0
14	15	S15	36	9	16	59	46	166	33.0
15	16	S16	43	69	21	76	34	244	49.0
16	17	S17	59	69	59	41	42	269	54.0
17	18	S18	94	24	62	37	18	235	47.0
18	19	S19	42	87	7	31	46	214	43.0
19	20	S20	32	23	76	81	57	269	54.0

In [113]:
```
#a. Extract the top 5 rankers.
df.nlargest(5,'Total Score')
```

Out[113]:

	Roll No	Name	Math	Python	COA	AT	CNND	Total Score	Percentage
4	5	S5	97	77	87	42	96	398	80.0
2	3	S3	67	77	87	53	98	382	76.0
1	2	S2	56	34	98	37	80	305	61.0
12	13	S13	70	71	15	99	45	300	60.0
6	7	S7	52	100	40	35	71	298	60.0

In [114]:
```
#b. Extract the last 5 losers.
df.nsmallest(5,'Total Score')
```

Out[114]:

	Roll No	Name	Math	Python	COA	AT	CNND	Total Score	Percentage
11	12	S12	22	55	20	4	36	137	27.0
5	6	S6	50	4	20	35	54	163	33.0
14	15	S15	36	9	16	59	46	166	33.0
18	19	S19	42	87	7	31	46	214	43.0
13	14	S14	55	34	59	48	21	217	43.0

In [125]:
```
#c. Display the names of the students whose Total Score is below median.
df.median()#finding the median
```

Out[125]:
```
Roll No         10.5
Math            55.5
Python          62.0
COA             58.0
AT              41.5
CNND            45.5
Total Score    250.0
Percentage      50.0
dtype: float64
```

In [123]:
```
df[df.Percentage < 50]#Printing values less than median
```

Out[123]:

	Roll No	Name	Math	Python	COA	AT	CNND	Total Score	Percentage
3	4	S4	71	77	43	2	49	243	49.0
5	6	S6	50	4	20	35	54	163	33.0
9	10	S10	44	90	32	25	42	233	47.0
10	11	S11	73	69	57	23	10	233	47.0
11	12	S12	22	55	20	4	36	137	27.0
13	14	S14	55	34	59	48	21	217	43.0
14	15	S15	36	9	16	59	46	166	33.0
15	16	S16	43	69	21	76	34	244	49.0
17	18	S18	94	24	62	37	18	235	47.0
18	19	S19	42	87	7	31	46	214	43.0

In [18]:
```
#Bifurcate the students and assign ranks depending on their scores.
#You can choose to have your range of marks to be assigned for grades A, B, C, D,
```

In [8]:
```
import pandas as pd
df = pd.read_csv("record.csv")
df
```

Out[8]:

	Roll No	Name	Math	Python	COA	AT	CNND	Total Score	Percentage
0	1	S1	45	32	67	70	60	274	55.0
1	2	S2	56	34	98	37	80	305	61.0
2	3	S3	67	77	87	53	98	382	76.0
3	4	S4	71	77	43	2	49	243	49.0
4	5	S5	97	77	87	42	96	398	80.0
5	6	S6	50	4	20	35	54	163	33.0
6	7	S7	52	100	40	35	71	298	60.0
7	8	S8	69	12	88	65	23	256	51.0
8	9	S9	98	12	83	50	40	284	57.0
9	10	S10	44	90	32	25	42	233	47.0
10	11	S11	73	69	57	23	10	233	47.0
11	12	S12	22	55	20	4	36	137	27.0
12	13	S13	70	71	15	99	45	300	60.0
13	14	S14	55	34	59	48	21	217	43.0
14	15	S15	36	9	16	59	46	166	33.0
15	16	S16	43	69	21	76	34	244	49.0
16	17	S17	59	69	59	41	42	269	54.0
17	18	S18	94	24	62	37	18	235	47.0
18	19	S19	42	87	7	31	46	214	43.0
19	20	S20	32	23	76	81	57	269	54.0

In [14]:
```
def fun(x):
    if 90<x<100:
        return 'A'
    elif 70<x<90:
        return 'B'
    elif 60<x<70:
        return 'C'
    elif 40<x<60:
        return 'D'
    else :
        return 'E'
#Defining function to map grades
```

Out[14]: 'B'

In [17]:
```
df['Grades'] = df['Grades'].apply(fun)
df# Applying function to grades coloumn
```

Out[17]:

	Roll No	Name	Math	Python	COA	AT	CNND	Total Score	Percentage	Grades
0	1	S1	45	32	67	70	60	274	55.0	D
1	2	S2	56	34	98	37	80	305	61.0	C
2	3	S3	67	77	87	53	98	382	76.0	B
3	4	S4	71	77	43	2	49	243	49.0	D
4	5	S5	97	77	87	42	96	398	80.0	B
5	6	S6	50	4	20	35	54	163	33.0	E
6	7	S7	52	100	40	35	71	298	60.0	E
7	8	S8	69	12	88	65	23	256	51.0	D
8	9	S9	98	12	83	50	40	284	57.0	D
9	10	S10	44	90	32	25	42	233	47.0	D
10	11	S11	73	69	57	23	10	233	47.0	D
11	12	S12	22	55	20	4	36	137	27.0	E
12	13	S13	70	71	15	99	45	300	60.0	E
13	14	S14	55	34	59	48	21	217	43.0	D
14	15	S15	36	9	16	59	46	166	33.0	E
15	16	S16	43	69	21	76	34	244	49.0	D
16	17	S17	59	69	59	41	42	269	54.0	D
17	18	S18	94	24	62	37	18	235	47.0	D
18	19	S19	42	87	7	31	46	214	43.0	D
19	20	S20	32	23	76	81	57	269	54.0	D

In [22]:
```
#e. Find out how many students have received grade A.
#f. Find out how many students have received grade B.
#g. Find out how many students have received grade C.
#h. Find out how many students have received grade D.
#i. Find out how many students have received grade E.
df['Grades'].value_counts()
#value counts function prints the unique values and their counts in specified col
```

Out[22]:
```
D    12
E     5
B     2
C     1
Name: Grades, dtype: int64
```

In [23]:
```
#g. The grace marks allowed for 100 marks subject is 5 Marks.
#Consider this as a new functionality and add another column Total Score which mu
#Now find out, which subject has the highest failure rate.
```

In [7]:
```
import pandas as pd
df = pd.read_csv("record.csv")
df
```

Out[7]:

	Roll No	Name	Math	Python	COA	AT	CNND	Total Score	Percentage
0	1	S1	45	32	67	70	60	274	55
1	2	S2	56	34	98	37	80	305	61
2	3	S3	67	77	87	53	98	382	76
3	4	S4	71	77	43	2	49	243	49
4	5	S5	97	77	87	42	96	398	80
5	6	S6	50	4	20	35	54	163	33
6	7	S7	52	100	40	35	71	298	60
7	8	S8	69	12	88	65	23	256	51
8	9	S9	98	12	83	50	40	284	57
9	10	S10	44	90	32	25	42	233	47
10	11	S11	73	69	57	23	10	233	47
11	12	S12	22	55	20	4	36	137	27
12	13	S13	70	71	15	99	45	300	60
13	14	S14	55	34	59	48	21	217	43
14	15	S15	36	9	16	59	46	166	33
15	16	S16	43	69	21	76	34	244	49
16	17	S17	59	69	59	41	42	269	54
17	18	S18	94	24	62	37	18	235	47
18	19	S19	42	87	7	31	46	214	43
19	20	S20	32	23	76	81	57	269	54

In [4]:
```
def func(x):
    if x<=35:
        return x+5
    else:
        return x
#Defining a function to increase marks by 5 if student has scored below 35 else r
```

In [5]:
```
df["Math"] = df["Math"].apply(func)
df["AT"] = df["AT"].apply(func)
df["CNND"] = df["CNND"].apply(func)
df["Python"] = df["Python"].apply(func)
df["COA"] = df["COA"].apply(func)
df
#Applying the function to all the subject coloumns
```

Out[5]:

	Roll No	Name	Math	Python	COA	AT	CNND	Total Score	Percentage
0	1	S1	45	37	67	70	60	274	55
1	2	S2	56	39	98	37	80	305	61
2	3	S3	67	77	87	53	98	382	76
3	4	S4	71	77	43	7	49	243	49
4	5	S5	97	77	87	42	96	398	80
5	6	S6	50	9	25	40	54	163	33
6	7	S7	52	100	40	40	71	298	60
7	8	S8	69	17	88	65	28	256	51
8	9	S9	98	17	83	50	40	284	57
9	10	S10	44	90	37	30	42	233	47
10	11	S11	73	69	57	28	15	233	47
11	12	S12	27	55	25	9	36	137	27
12	13	S13	70	71	20	99	45	300	60
13	14	S14	55	39	59	48	26	217	43
14	15	S15	36	14	21	59	46	166	33
15	16	S16	43	69	26	76	39	244	49
16	17	S17	59	69	59	41	42	269	54
17	18	S18	94	29	62	37	23	235	47
18	19	S19	42	87	12	36	46	214	43
19	20	S20	37	28	76	81	57	269	54

In [15]:
```
def fun2(x):
    if x<35:
        return "F"
    else:
        return "P"
#Defining a function to print F-Failed if student has scored below 35 else print
```

In [ ]:
```
df["Math"] = df["Math"].apply(fun2)
df["CNND"] = df["CNND"].apply(fun2)
df["Python"] = df["Python"].apply(fun2)
df["COA"] = df["COA"].apply(fun2)
#Applying the functions to all subject coloumn
```

In [23]:
```
df
```

Out[23]:

	Roll No	Name	Math	Python	COA	AT	CNND	Total Score	Percentage
0	1	S1	P	P	P	P	P	274	55
1	2	S2	P	P	P	P	P	305	61
2	3	S3	P	P	P	P	P	382	76
3	4	S4	P	P	P	F	P	243	49
4	5	S5	P	P	P	P	P	398	80
5	6	S6	P	F	F	P	P	163	33
6	7	S7	P	P	P	P	P	298	60
7	8	S8	P	F	P	P	F	256	51
8	9	S9	P	F	P	P	P	284	57
9	10	S10	P	P	P	F	P	233	47
10	11	S11	P	P	P	F	F	233	47
11	12	S12	F	P	F	F	P	137	27
12	13	S13	P	P	F	P	P	300	60
13	14	S14	P	P	P	P	F	217	43
14	15	S15	P	F	F	P	P	166	33
15	16	S16	P	P	F	P	P	244	49
16	17	S17	P	P	P	P	P	269	54
17	18	S18	P	F	P	P	F	235	47
18	19	S19	P	P	F	P	P	214	43
19	20	S20	P	F	P	P	P	269	54

In [27]:
```
df['Math'].value_counts()#Prints the count of students who have passed and failed
```

Out[27]:
```
P    19
F     1
Name: Math, dtype: int64
```

In [26]:
```
df['AT'].value_counts()
```

Out[26]:
```
P    16
F     4
Name: AT, dtype: int64
```

In [28]:
```
df['Python'].value_counts()
```

Out[28]:
```
P    14
F     6
Name: Python, dtype: int64
```

In [29]:
```
df['COA'].value_counts()
```

Out[29]:
```
P    14
F     6
Name: COA, dtype: int64
```

In [30]:
```
df['CNND'].value_counts()
```

Out[30]:
```
P    16
F     4
Name: CNND, dtype: int64
```

As we can see from the above results

g)subjects with highest failure rate is **Python and COA**

i)subjects with least failure rate is **Math**

CONCLUSION:

```
Thus we have successfully implemented Pandas in Python.
```