EXERCISE 4

PROGRAM 1

AIM :

Write a python program to implement list-to series conversion

CODE:

```
import pandas as pd
lst =[1,2,3,4,5]
print(pd.Series(lst))
```

OUTPUT:

```
0    1
1    2
2    3
3    4
4    5
dtype: int64
```

PROGRAM 2

AIM:
Write a python program to generate the series of dates from first May 2021 to 12th May 2022(both inclusive)

CODE:

```
import pandas as pd
dat = pd.date_range(start='05-01-2021' , end='05-12-2021')
print(dat)
```

OUTPUT:

```
DatetimeIndex(['2021-05-01', '2021-05-02', '2021-05-03', '2021-05-04',
               '2021-05-05', '2021-05-06', '2021-05-07', '2021-05-08',
               '2021-05-09', '2021-05-10', '2021-05-11', '2021-05-12'],
              dtype='datetime64[ns]', freq='D')
```

PROGRAM 3

AIM:

Given a dictionary convert it into corresponding dataframe and display it.

CODE:

```
import pandas as pd
data= {'name': ['john','sarah'],
       'Age': [21,20],
        'Courses': ['CS','Physics']}
df = pd.DataFrame.from_dict(data)
print(df)
```

OUTPUT:

```
     name  Age  Courses
0   john   21       CS
1  sarah   20  Physics
```

PROGRAM 4

AIM:

Given a 2D list convert it into corresponding dataframe and display it.

CODE:

```
import pandas as pd
lst =[ ['Apple',150],
     ['orange',65],
     ['avacado',165]]
df = pd.DataFrame(lst,columns =['Fruits','Price'])
print(df)
```

OUTPUT:

```
      Fruits  Price
0    Apple    150
1   orange     65
2  avacado    165
```

PROGRAM 5

AIM:

Given a CSV file read it into a dataframe and display it.

CODE:

```
import panda as pd
df = pd.read_csv("details.csv")
print(df)
```

OUTPUT:

```
      Name  Age  Gender
0  Anamika   22  Female
1      Ann   22  Female
2   Joseph   20    Male
```

PROGRAM 6

AIM:

Given a dataframe sort it by multiple columns

CODE:

```
import panda as pd
datafra ={'Name':['Anna','Elsa','Christoph'],
          'Score':[12.5,9,16.5],
           'Attempts':[1,0,2]}
df = pd.DataFrame(datafra)
print("original Dataframe:")
print(df)
print("\nSorted Dataframe:")
print(df.sort_values(['Attempts','Name']))
```

OUTPUT:

```
original Dataframe:
           Name  Score  Attempts
0      Anna   12.5        1
1      Elsa    9.0        0
2  Christoph   16.5        2

Sorted Dataframe:
           Name  Score  Attempts
1      Elsa    9.0        0
0      Anna   12.5        1
2  Christoph   16.5        2
```

PROGRAM 7

AIM:

Given a dataframe with custom indexing, convert and it to default indexing and display it

CODE:

```
import panda as pd
stud ={'name':['Anna','elss'],
       'Marks':[85,77]}
stud_df=pd.DataFrame(stud,index=['a','b'])
print(stud_df)
print("Dataframe with default index")
stud1=stud_df.reset_index()
print(stud1)
```

OUTPUT:

```
  name  Marks
a  Anna    85
b  elss    77

Dataframe with default index

   index  name  Marks
0     a  Anna    85
1     b  elss    77
```

PROGRAM 8

AIM:

Given a dataframe select first two rows and output them

CODE:

```
import panda as pd
datafra ={'Name':['Anna','Elsa','Christoph'],
          'Score':[12.5,9,16.5],
           'Attempts':[1,0,2]}
df = pd.DataFrame(datafra)
print(df.head(2))
```

OUTPUT:

```
    Name  Score  Attempts
0  Anna   12.5         1
1  Elsa    9.0         0
```

PROGRAM 9

AIM:

Given is a dataframe showing name, occupation, salary of people. Find the average salary per occupation.

CODE:

```
import panda as pd
data = {
   'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Eve'],
   'Occupation': ['Engineer', 'Teacher', 'Doctor', 'Engineer', 'Doctor'],
   'Salary': [60000, 45000, 75000, 65000, 80000]
}

df = pd.DataFrame(data)

average_salary_per_occupation = df.groupby('Occupation')['Salary'].mean()

print(average_salary_per_occupation)
```

OUTPUT:

```
Occupation
Doctor      77500.0
Engineer    62500.0
Teacher     45000.0
Name: Salary, dtype: float64
```

PROGRAM 10

AIM:

Given a dataframe with NaN values fill the NaN values with zero.

CODE:

```
import panda as pd
import numpy as np
data = {
    'A': [1, 2, np.nan, 4, 5],
    'B': [np.nan, 2, 3, np.nan, 5],
    'C': [1, 2, 3, 4, np.nan]
}

df = pd.DataFrame(data)

df_filled = df.fillna(0)


print(df_filled)
```

OUTPUT:

```
     A    B    C
0  1.0  0.0  1.0
1  2.0  2.0  2.0
2  0.0  3.0  3.0
3  4.0  0.0  4.0
4  5.0  5.0  0.0
```

PROGRAM 11

AIM:

Given a dataframe showing company names(ename) and corresponding profits(profit). Convert the values of profit columns such that the values in it greater than zero are set to True and the rest are set to False;

CODE:

```
import pandas as pd

data = {
   'cname': ['Company A', 'Company B', 'Company C', 'Company D'],
   'profit': [10000, -5000, 0, 7500]
}

df = pd.DataFrame(data)

df['profit'] = df['profit'].apply(lambda x: x > 0)

print(df)
```

OUTPUT:

```
          cname  profit
0  Company A    True
1  Company B   False
2  Company C   False
3  Company D    True
```

PROGRAM 12

AIM:

Given are two dataframes, with one dataframe containing employee ID(eid),employee Name(ename) and stipend(stipend) and the other dataframe containing employee ID(eid) and designation of the employee(designation). Output the dataframe containing employee ID(eid), employee Name(ename), stipend(stipend) and position(position)

CODE:

```
import panda as pd
data1 = {
   'eid': [101, 102, 103, 104],
   'ename': ['Alice', 'Bob', 'Charlie', 'David'],
   'stipend': [5000, 6000, 5500, 7000]
}

data2 = {
   'eid': [101, 103, 104, 105],
   'designation': ['Engineer', 'Manager', 'Analyst', 'Technician']
}

df1 = pd.DataFrame(data1)
df2 = pd.DataFrame(data2)

result_df = df1.merge(df2, on='eid', how='left')

result_df = result_df.rename(columns={'designation': 'position'})

print(result_df)
```

OUTPUT:

```
    eid    ename  stipend  position
0  101    Alice     5000  Engineer
1  102      Bob     6000       NaN
2  103  Charlie     5500   Manager
3  104    David     7000   Analyst
```