'''

Experiment 12 : Program to demonstrate series and dataframe in Pandas

Name : Khan Arshad Abdulla

Roll No : 20CO24

Academic Year : 2021-22

THEORY :

Pandas is an open-source library that is made mainly for working with relational or labeled data both easily and intuitively. It provides various data structures and operations for manipulating numerical data and time series. This library is built on top of the NumPy library. Pandas is fast and it has high performance & productivity for users.

Pandas were initially developed by Wes McKinney in 2008 while he was working at AQR Capital Management. He convinced the AQR to allow him to open source the Pandas. Another AQR employee, Chang She, joined as the second major contributor to the library in 2012. Over time many versions of pandas have been released. The latest version of the pandas is 1.4.1

Pandas Series is a one-dimensional labelled array capable of holding data of any type (integer, string, float, python objects, etc.). The axis labels are collectively called indexes. Pandas Series is nothing but a column in an excel sheet. Labels need not be unique but must be a hashable type. The object supports both integer and label-based indexing and provides a host of methods for performing operations involving the index.

'''

```
import pandas as pd


#Series
```

'''

Pandas Series is a one-dimensional labeled array capable of holding data of any type (integer, string, float, python objects, etc.). The axis labels are collectively called index. Pandas Series is nothing but a column in an excel sheet.

Labels need not be unique but must be a hashable type. The object supports both integer and label-based indexing and provides a host of methods for performing operations involving the index.

'''

```
s = pd.Series([10,20,30,40,50])

print(s)

print(s[0])
```

```
#some slicing, indexing, other features of Series
print("Indexing of Series:")
print(f"{s[4]}\n")

print("Slicing of Series:")
print(f"{s[1:4]}\n")



#Dataframe
'''
Pandas DataFrame is two-dimensional size-mutable, potentially heterogeneous tabular data
structure with labeled axes (rows and columns). A Data frame is a two-dimensional data structure,
i.e., data is aligned in a tabular fashion in rows and columns. Pandas DataFrame consists of three
principal components, the data, rows, and columns.
'''
df = pd.DataFrame([10,20,30,40,50])
print(df)
#some slicing, indexing, other features of Dataframe

#Dataframe operations using csv dataset
df = pd.read_csv("C:/Users/arsha/OneDrive/Desktop/20CO24 Python/true_car_listings.csv")
print('First 10 rows:\n',df.head(10))
print('Last 5 rows:\n',df.tail(5))
print('Total Columns/Attributes/Features:',df.columns)

print('Displaying Make Column:\n',df['Make'])
print('Displaying first 5 rows with 4 columns:\n',df[['Make','Model','Year','Price']].head())
print('Displaying rows 5 and 6:\n',df[5:7])
print('Displaying rows 1, 10 and 20:\n',df.loc[[1,10,20]])
print('Displaying rows 1 to 9 and columns 6 and 7:\n',df.iloc[1:10,6:8])
print('Displaying rows with Price less than 5000:\n',df.loc[df['Price']<5000])
print('Null Values count for all columns:\n',df.isna().sum())

#filling na values of Price col with mean value of Price
df['Price'].fillna(df['Price'].mean)

#Grouping records based on Year value.
groupby = df.groupby('Year')
for year, group in groupby:
   print(year)
   print(group)

'''
OUTPUT :

PS C:\Users\arsha> python -u "c:\Users\arsha\OneDrive\Desktop\20CO24 Python\Exp12.py"
0    10
1    20
2    30
3    40
4    50
dtype: int64
10
Indexing of Series:
50

Slicing of Series:
1    20
2    30
3    40
dtype: int64
```

```
   0
0  10
1  20
2  30
3  40
4  50
First 10 rows:
   Price  Year  Mileage            City State               Vin   Make        Model
0   8995  2014    35725          El Paso    TX  19VDE2E53EE000083  Acura    ILX6-Speed
1  10888  2013    19606  Long Island City    NY  19VDE1F52DE012636  Acura    ILX5-Speed
2   8995  2013    48851          El Paso    TX  19VDE2E52DE000025  Acura    ILX6-Speed
3  10999  2014    39922          Windsor    CO  19VDE1F71EE003817  Acura    ILX5-Speed
4  14799  2016    22142           Lindon    UT  19UDE2F32GA001284  Acura  ILXAutomatic
5   7989  2012   105246            Miami    FL  JH4CU2F83CC019895  Acura  TSXAutomatic
6  14490  2014    34032        Greatneck    NY  JH4CU2F84EC002686  Acura    TSXSpecial
7  13995  2013    32384      West Jordan    UT  JH4CU2F64DC006203  Acura    TSX5-Speed
8  10495  2013    57596        Waterbury    CT  19VDE2E50DE000234  Acura    ILX6-Speed
9   9995  2013    63887          El Paso    TX  19VDE1F50DE010450  Acura    ILX5-Speed
Last 5 rows:
        Price  Year  Mileage         City State               Vin   Make    Model
852117  63215  2017        9  Culver City    CA  YV1A22MK9H1013237  Volvo    S90T6
852118  72260  2017     3201    Englewood    NJ  YV4A22PL3H1186162  Volvo   XC90T6
852119  55999  2016    28941  Fort Collins    CO  YV4A22PL4G1000868  Volvo  XC90AWD
852120  60240  2017     3005  San Leandro    CA  YV4A22NLXH1006162  Volvo      V90
852121  76995  2017     2502     New York    NY  YV4BC0ZX1H1109845  Volvo   XC90T8
Total Columns/Attributes/Features: Index(['Price', 'Year', 'Mileage', 'City', 'State', 'Vin', 'Make',
'Model'], dtype='object')
4  Acura  ILXAutomatic  2016  14799
Displaying rows 5 and 6:
   Price  Year  Mileage      City State               Vin   Make        Model
5   7989  2012   105246     Miami    FL  JH4CU2F83CC019895  Acura  TSXAutomatic
```

```
6  14490  2014   34032  Greatneck   NY  JH4CU2F84EC002686  Acura   TSXSpecial
Displaying rows 1, 10 and 20:
    Price  Year  Mileage            City State             Vin   Make       Model
1   10888  2013   19606  Long Island City   NY  19VDE1F52DE012636  Acura   ILX5-Speed
10  12921  2012   58550           Boise   ID  JH4CU2F44CC003220  Acura  TSXAutomatic
20  16994  2015   23946    St. Augustine   FL  19VDE1F32FE000651  Acura   ILX5-Speed
Displaying rows 1 to 9 and columns 6 and 7:
    Make       Model
1  Acura   ILX5-Speed
2  Acura   ILX6-Speed
3  Acura   ILX5-Speed
4  Acura  ILXAutomatic
5  Acura  TSXAutomatic
6  Acura   TSXSpecial
7  Acura   TSX5-Speed
8  Acura   ILX6-Speed
9  Acura   ILX5-Speed
Displaying rows with Price less than 5000:
       Price  Year  Mileage         City State             Vin   Make       Model
648     4950  2006   142587     Littleton   CO  JH4CL96826C031231  Acura  TSXAutomatic
1179     4990  2008   159601     Boardman   OH  JH4CL96878C000866  Acura      TSX4dr
1222     4899  2006   144259     Haverhill   MA  JH4CL96806C012614  Acura  TSXAutomatic
1298     4599  2005    90008  PINELLAS PARK   FL  19UUA66245A038764  Acura   TLAutomatic
1448     4990  2006   170470  Brooklyn Park   MN  JH4CL96846C007545  Acura  TSXAutomatic
...      ...   ...     ...          ...   ...             ...    ...         ...
851712   3999  2004   154898     Longwood   FL  YV1SW64AX42429597  Volvo      V702.4L
851771   4990  1998    96543      Fairfax   VA  YV1LS5577W1535110  Volvo      S704dr
851774   3999  1998   109198     Longmont   CO  YV1LS5549W2445313  Volvo      S704dr
851785   3998  2006   204001     Marietta   GA  YV4CY592861284131  Volvo     XC902.5L
851925   3111  2001   202691       Odessa   TX  YV1SW61R512089006  Volvo      V702.4
```

```
[18111 rows x 8 columns]
```

CONCLUSION: In this experiment we have successfully implemented series and dataframe in pandas.

'''