assigment

July 19, 2023

[5]:
```
Q1. Create a Pandas Series that contains the following data: 4, 8, 15, 16, 23,␣
 ↪and 42. Then, print the series.
```

[17]:
```
import pandas as pd
data = [4,8,15,16,23,42]
series = pd.Series(data)
series
```

[17]:
```
0     4
1     8
2    15
3    16
4    23
5    42
dtype: int64
```

[8]:
```
Q2. Create a variable of list type containing 10 elements in it, and apply␣
 ↪pandas.Series function on the
 variable print it.
```

[11]:
```
list = [1,2,3,4,5,6,7,8,9,10]
series = pd.Series(list)
print(series)
```

```
0     1
1     2
2     3
3     4
4     5
5     6
6     7
7     8
8     9
9    10
dtype: int64
```

[ ]:
```
Q3. Create a Pandas DataFrame that contains the following data:
```

[14]:
```
df = pd.DataFrame({ 'Name':['Alice','Bob','claire'] , 'Age':[25,30,27]␣
↪,'Gander':['Female','Male','Female']})
print(df)
```

```
     Name  Age  Gander
0   Alice   25  Female
1     Bob   30    Male
2  claire   27  Female
```

[15]:
```
Q4. What is 'DataFrame' in pandas and how is it different from pandas.series?␣
↪Explain with an example.
```

[16]:
```
A DataFrame in Pandas is a two-dimensional labeled data structure with columns␣
↪of potentially different types. It is similar to a spreadsheet or a SQL␣
↪table, where each column can have a different type (e.g., numeric, string,␣
↪boolean), and rows are labeled with an index. A DataFrame can be thought of␣
↪as a collection of Series objects, where each Series represents a column in␣
↪the DataFrame.

 The main difference between a Series and a DataFrame is that a Series␣
↪represents a single column of data, while a DataFrame represents multiple␣
↪columns of data, arranged in a tabular format.

Here's an example to illustrate the difference between a Series and a DataFrame:
```

[27]:
```
series = pd.Series([1,2,3,4,5])

df = pd.DataFrame({'Name':['Alice','Bob','claire'] ,'Age':[25,30,27] ,'Gander':
↪['Female','Male','Female']})



print( series)

print(df)
```

```
0    1
1    2
2    3
3    4
4    5
dtype: int64
     Name  Age  Gander
0   Alice   25  Female
1     Bob   30    Male
2  claire   27  Female
```

[ ]:
```
Q5. What are some common functions you can use to manipulate data in a Pandas␣
↪DataFrame?
Can you give an example of when you might use one of these functions?
```

[ ]:
```
Pandas provides a wide range of functions that can be used to manipulate data␣
↪in a DataFrame. Some common functions include:

head() and tail(): to view the first or last n rows of a DataFrame describe():
 to view the statistical summary of the DataFrame shape:
 to view the number of rows and columns in the DataFrame drop():
 to remove rows or columns from the DataFrame groupby():
 to group rows based on a column and apply a function to each group␣
↪sort_values():
to sort the DataFrame by one or more columns fillna():
 to fill missing values in the DataFrame with a specified value or method␣
↪apply():
to apply a function to each element of a DataFrame or a Series merge():
 to join two or more DataFrames based on a common column or index Here's an␣
↪example of when you might use one of these functions.
Suppose you have a DataFrame containing information about employees in a␣
↪company, and you want to view the statistical summary of their salaries:
```

[33]:
```
employee_data = { 'Name':['ram','shyam','mohan'],
                  'Age' :[25,29,28],
                  'Sallery':[50000,100000,60000]
}

df = pd.DataFrame(employee_data)

print(df['Sallery'].describe())
```

```
count         3.000000
mean      70000.000000
std       26457.513111
min       50000.000000
25%       55000.000000
50%       60000.000000
75%       80000.000000
max      100000.000000
Name: Sallery, dtype: float64
```

[ ]:
```
Q6. Which of the following is mutable in nature Series, DataFrame, Panel?
```

[ ]:
```
Among the three data structures provided by Pandas, only the DataFrame and␣
↪Panel are mutable in nature.
```

```
A DataFrame is mutable because you can add, remove or modify columns and rows.␣
↪Similarly, a Panel is mutable because
you can add or remove items along the axis.

On the other hand, a Series is immutable because it represents a single column␣
↪of data with an index.
Once created, you cannot add or remove elements from a Series. However, you␣
↪can modify the values of existing elements in a Series.
```

[ ]:
```
Q7. Create a DataFrame using multiple Series. Explain with an example.
```

[34]:
```
name = pd.Series(['ram','mohan','bob','steav','zeus','mark'])
age = pd.Series([21,23,25,28,26,])
sallery = pd.Series([500000,230000,440000,540000,90000,858000])

df = pd.DataFrame({'Name':name , 'Age':age, 'Sallery': sallery})

print(df)
```

```
    Name   Age  Sallery
0    ram  21.0   500000
1  mohan  23.0   230000
2    bob  25.0   440000
3  steav  28.0   540000
4   zeus  26.0    90000
5   mark   NaN   858000
```

[ ]:
```

```