

assignment=19

July 27, 2023

- 1 Q1. Create a Pandas Series that contains the following data: 4, 8, 15, 16, 23, and 42. Then, print the series.

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

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

- 2 Q2. Create a variable of list type containing 10 elements in it, and apply pandas.Series function on the variable print it.

```
[2]: import pandas as pd  
  
my_list = [2, 4, 6, 8, 10, 12, 14, 16, 18, 20]  
my_series = pd.Series(my_list)  
print(my_series)
```

```
0    2  
1    4  
2    6  
3    8  
4   10  
5   12  
6   14  
7   16  
8   18  
9   20
```

```
dtype: int64
```

3 Q3. Create a Pandas DataFrame that contains the following data:

```
[3]: import pandas as pd
df = pd.DataFrame({'Name':['Alice', 'Bob', 'Claire'], 'Age':[25, 30, 27], 'Gender':['Female', 'Male', 'Female']})
df
```

```
[3]:      Name  Age  Gender
0    Alice   25  Female
1      Bob   30     Male
2  Claire   27  Female
```

4 Q4. What is ‘DataFrame’ in pandas and how is it different from pandas.series? Explain with an example.

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:

```
[6]: import pandas as pd

# Creating a Pandas Series
my_series = pd.Series([10, 20, 30, 40, 50])

# Creating a Pandas DataFrame
my_dataframe = pd.DataFrame({
    'Name': ['sonal', 'Jadavbhai', 'Bob', 'Alice', 'Mike'],
    'Age': [21, 47, 35, 40, 45],
    'Salary': [1000000, 1000000, 70000, 80000, 90000]
})

# Printing the Series and DataFrame
print("Pandas Series:\n", my_series)
print("\nPandas DataFrame:\n", my_dataframe)
```

Pandas Series:

```
0    10
1    20
```

```

2    30
3    40
4    50
dtype: int64

Pandas DataFrame:
      Name  Age   Salary
0    sonal   21  1000000
1  Jadavbhai   47  1000000
2       Bob   35    70000
3     Alice   40    80000
4      Mike   45    90000

```

5 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:

```
[8]: import pandas as pd

# Creating a sample DataFrame
employee_data = {
    'Name': ['sonal', 'Jadavbhai', 'Bob', 'Alice', 'Mike'],
    'Age': [21, 47, 35, 40, 45],
    'Salary': [1000000, 1000000, 70000, 80000, 90000]
}
df = pd.DataFrame(employee_data)

# Using the describe() function to view the statistical summary of the Salary
# column
print(df['Salary'].describe())
```

```

count      5.000000
mean      448000.000000
std       503954.363013

```

```
min        70000.000000
25%       80000.000000
50%       90000.000000
75%      1000000.000000
max      1000000.000000
Name: Salary, dtype: float64
```

6 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.

7 Q7. Create a DataFrame using multiple Series. Explain with an example.

```
[11]: import pandas as pd

# Creating Series for Name, Age and Gender
name = pd.Series(['sonal', 'Jadavbhai', 'Bob', 'Alice', 'Mike'])
age = pd.Series([21, 47, 35, 40, 45])
gender = pd.Series(['feMale', 'male', 'Male', 'Female', 'Male'])

# Creating a DataFrame by passing the Series as arguments
df = pd.DataFrame({'Name': name, 'Age': age, 'Gender': gender})

# Printing the DataFrame
print(df)
```

	Name	Age	Gender
0	sonal	21	feMale
1	Jadavbhai	47	male
2	Bob	35	Male
3	Alice	40	Female
4	Mike	45	Male

```
[ ]:
```