Series Related Questions

Ravi wants to create a Pandas Series as shown below:

January	31
February	28
March	31

Help him in completing the code below to achieve the desired output.

Note: **ser_data** is a dictionary.

```
import _____ as pd
ser_data = _____
s = pd._____(ser_data)
print(s)
```

Ans:

pandas

{'January': 31, 'February': 28, 'March': 31}

Series

Creation of series using ndarray

Write a Python program to create a Pandas Series as shown below using a ndarray, where the subject names are the indices and the corresponding marks are the values in the series.

Mathematics	85
Science	90
English	78
History	88

Ans:

```
import pandas as pd
import numpy as np
 marks = np.array([85, 90, 78, 88])
series = pd.Series(marks, index=['Mathematics', 'Science', 'English', 'History'])
print(series)
```

Q. What is a Series in Python Pandas? Also, give a suitable example to support your answer.

Ans:

A Series is a one-dimensional array containing a sequence of values of any data type (int, float, list, string, etc) which by default have numeric data labels starting from zero.

We can imagine a Pandas Series as a column in a spreadsheet. An example of a series containing the names of students is given below:

Index	Value
0	Arnab
1	Samridhi
2	Ramit
3	Divyam

Complete the given Python code to get the required output (ignore the dtype attribute) as

Output:

```
Tamil Nadu      Chennai
Uttar Pradesh   Lucknow
Manipur         Imphal
```

Code:

```
import _______ as pd
data = ['Chennai','_______','Imphal']
indx = ['Tamil Nadu','Uttar Pradesh','Manipur']
s = pd.Series(_______, indx)
print(_______)
```

Ans:

```
import pandas as pd
data = ['Chennai', 'Lucknow', 'Imphal']
indx = ['Tamil Nadu','Uttar Pradesh','Manipur']
s = pd.Series(data, indx)
print(s)
```

(1/2 mark for each correct fill in the blank)

Write a Python Program to create a Pandas Series as shown below **using a dictionary**. Note that the left column indicates the indices and the right column displays the data.

Russia	Moscow
Hungary	Budapest
Switzerland	Bern

Ans:

```
import pandas as pd

data = {'Russia':'Moscow','Hungary':'Budapest','Switzerland':'Bern'}  # using dictionary {key:value}

s = pd.Series(data)   # creation of series

print(s)
```

Predict the output of the given Python code:

```
import pandas as pd

list1=[-10,-20,-30]

ser = pd.Series(list1*2)

 print(ser)
```

Ans: #

```
0   -10
1   -20
2   -30
3   -10
4   -20
5   -30
```

Reason

List Repetition (list1 * 2): In Python, when you multiply a list by an integer n, it does not perform mathematical multiplication on the elements. Instead, it performs list replication. It creates a new list by repeating the original list n times.

When pd.Series() is called with this replicated list, Pandas converts it into a Series object. Since no specific index was provided, Pandas assigns a default integer index starting from 0.

Complete the given Python code to get the required output as: **Rajasthan**

```
import _________ as pd
di = {'Corbett': 'Uttarakhand', 'Sariska':
'Rajasthan', 'Kanha': 'Madhya Pradesh',
'Gir':'Gujarat'}
NP = ___________. Series( _____ )
print(NP[ ___________ ])
```

Ans:

```
import pandas as pd
di = {'Corbett': 'Uttarakhand', 'Sariska':'Rajasthan', 'Kanha': 'Madhya
Pradesh','Gir':'Gujarat'}
NP = pd.Series( di)
print(NP[ 'Sariska'])
```

When a dictionary is converted to a Pandas Series, the dictionary keys (Corbett, Sariska, Kanha, Gir) become the index of the Series.

The dictionary values (Uttarakhand, Rajasthan, etc.) become the data of the Series.

Since the required output is "Rajasthan", we must access the Series using the index label 'Sariska'.

Q. Write a program to create a series object **using a dictionary** that stores the number of students in each house of class 12D of your school.

Note: Assume four house names are Beas, Chenab, Ravi and Satluj having 18, 2, 20, 18 students respectively and pandas library has been imported as pd.

Ans:

import pandas as pd

St={‘Beas’ :18, ‘Chenab’ :20 , ‘ Ravi’ :20, ‘ Satluj’ :18}

S1=pd.Series(St)

print(S1)

When a dictionary is converted to a Pandas Series, the dictionary keys become the index of the Series

---

Q.

What will be the output of the following code:

```
>>>import pandas as pd
>>>A=pd.Series(data=[35,45,55,40])
>>>print(A>45)
```

Ans:

```
0    False
1    False
2     True
3    False
```

Reason

When you run A > 45, Pandas does not compare the Series as a single object. Instead, it compares every individual element in the Series against the value 45.

Index 0: `35 > 45` → **False**

Index 1: `45 > 45` → **False** (since 45 is not *greater than* 45)

Index 2: `55 > 45` → **True**

Index 3: `40 > 45` → **False**

Note:
Earlier it is print(A>45)
what if print(A[A>45])

Expiation

Changing the code from print(A > 45) to print(A[A > 45]) completely changes the result from a list of True/False values to a filtered subset of the actual data. This is known as Boolean Indexing.

Output is

```
2    55
dtype: int64
```

Indexing Operation: When you place this mask inside square brackets A[...], Pandas uses it as a filter. It only keeps the rows where the value is True.

Filtering Results:

Index 0 (False) is dropped.

Index 1 (False) is dropped.

Index 2 (True) is kept. The value at this index is 55.

Index 3 (False) is dropped.
Final Output: The result is a new Series containing only the element 55. Because the original data was integers