

PANDAS DATA SERIES EXERCISES

- Cho nhập số nguyên n. Tạo 1 list gồm n phần tử với giá trị ngẫu nhiên từ -100 đến +100. Tạo ra 1 Series (S1) từ list L1
Tạo ra 1 list L2 từ Series S1. Cho biết kiểu dữ liệu của L2
- Tạo ra list L3 bằng cách tương tự như khi tạo list L1. Tạo series S2 từ L3. Thực hiện các phép toán: cộng, trừ, nhân, chia giữa S1 và S2 để có kết quả như minh họa sau:

S1 + S2 =	S1 - S2 =	S1 * S2 =	S1 / S2 =
0 -89	0 49	0 1380	0 0.289855
1 -111	1 -11	1 3050	1 1.220000
2 66	2 10	2 1064	2 1.357143
3 173	3 -1	3 7482	3 0.988506
4 -76	4 -60	4 544	4 8.500000
5 33	5 11	5 242	5 2.000000
6 -48	6 -142	6 -4465	6 -2.021277
7 131	7 67	7 3168	7 3.093750
8 99	8 87	8 558	8 15.500000
9 46	9 112	9 -2607	9 -2.393939
dtype: int64	dtype: int64	dtype: int64	dtype: float64

- Thực hiện so sánh từng giá trị trong 2 series S1 và S2 với các phép so sánh =, >, < để có kết quả như minh họa sau:

So sánh giá trị giữa từng phần tử trong 2 Series S1 và S2: So sánh S1=S2:	So sánh S1>S2:	So sánh S1<S2:
0 False	0 True	0 False
1 False	1 False	1 True
2 False	2 True	2 False
3 False	3 False	3 True
4 False	4 False	4 True
5 False	5 True	5 False
6 False	6 False	6 True
7 False	7 True	7 False
8 False	8 True	8 False
9 False	9 True	9 False
dtype: bool	dtype: bool	dtype: bool

- Tạo dictionary gồm n phần tử (n<26), với key là các ký tự từ a-z và value là các giá trị ngẫu nhiên từ 1-50. Thực hiện chuyển dictionary này thành Pandas series S3:

Ví dụ:

Original dictionary: {'a': 10, 'b': 20, 'c': 37, 'd': 42, 'e': 50}

Converted series:

a 10

b 20

c 37

d 42

e 50

dtype: int64

- Tạo ra Numpy array A1 gồm n phần tử với giá trị ngẫu nhiên từ -10 đến +20. Tạo ra Pandas Series từ A1.

Ví dụ: NumPy array: [10 20 30 40 50]

Converted Pandas series:

0 10

1 20

2 30

```

3      40
4      50
dtype: int64

```

6. Gán giá trị 'Python' cho phần tử thứ 5 trong S1 và giá trị True cho phần tử thứ 8 trong S1. Thực hiện chuyển kiểu dữ liệu của S1 từ kiểu int sang kiểu float (sử dụng phương thức `to_numeric` với 2 đối số: tên Series cần chuyển và đối số `errors='coerce'`). Hiển thị kết quả vừa thực hiện.

7.

- Tạo dictionary D1 với key lần lượt là col1, col2, col3; với value của mỗi key là 1 list gồm 6 phần tử.
- Tạo ra DataFrame DF1 từ D1.
- Sử dụng `iloc` để tạo ra các Pandas Series S4, S5, S6 ứng với 3 cột của D1.
- Xem kiểu của các Pandas Series S4, S5, S6

```

Original DataFrame
   col1  col2  col3
0      1     4     7
1      2     5     5
2      3     6     8
3      4     9    12
4      7     5     1
5     11     0    11

```

```

1st column as a Series:
0      1
1      2
2      3
3      4
4      7
5     11
Name: col1, dtype: int64
<class 'pandas.core.series.Series'>

```

8. Viết chương trình chuyển Series S1 sang Numpy array. Minh họa kết quả:

```

Original Data Series:
0      -20
1     -61
2      38
3      86
4     -68
5    Python
6     -95
7      99
8     True
9      79
dtype: object
Series to an array
[-20 -61 38 86 -68 'Python' -95 99 True 79]
<class 'numpy.ndarray'>

```

9. Tạo 1 series S7 gồm nhiều phần tử, mỗi phần tử là 1 list các chuỗi với số lượng tùy ý. Viết chương trình chuyển S7 thành series S8 bằng phương thức `stack().reset_index(drop=True)`

```

Sample Output:
Original Series of list

```

```

0      [Red, Green, White]
1      [Red, Black]
2      [Yellow]
dtype: object
One Series
0      Red
1      Green
2      White
3      Red
4      Black
5      Yellow
dtype: object

```

10. Sử dụng phương thức `sort_values()` để sắp xếp dữ liệu trong S8.
11. Sử dụng `pd.concat` để thêm 3 giá trị tùy ý vào S8
12. Tạo các subset của Pandas Series S1 thỏa điều kiện sau:
 - Cho nhập số nguyên m, tạo S1A chứa m phần tử đầu tiên có trong S1
 - Cho nhập tiếp số nguyên k ($k < m$), tạo S1B là chứa các phần tử từ vị trí k đến vị trí m của S1.
 - Tạo series S1C chứa các giá trị ≤ 0 có trong S1
13. Tạo series S9 gồm 5 phần tử với index lần lượt là các ký tự từ A-E, giá trị tùy ý. Thực hiện sắp xếp lại series dựa trên phương thức `reindex` lại cho S9 để index có thứ tự là: d,a,d,b,c.
 Minh họa kết quả:
 Original Data Series:


```

A      1
B      2
C      3
D      4
E      5
dtype: int64

```

 Data Series after changing the order of index:


```

B      2
A      1
C      3
D      4
E      5
dtype: int64

```
14. Xác định các giá trị, mean, standard deviation của S1.
15. Sử dụng toán tử `~` và phương thức `isin` để xác định các giá trị có trong S1 nhưng không có trong S2 và các giá trị có trong S1 lẫn S2.
16. Sử dụng các phương thức `union1d` và `intersect1d` trong numpy để tìm những phần hội và giao của 2 series S1 và S2
17. Sử dụng phương thức `value_counts()` để tính tần suất xuất hiện của các giá trị có trong S1.
18. Xác định giá trị mode có trong Series S1 và thay thế mọi giá trị khác thành 'Other'.
19. Cho 2 Series S1 và S2
 - Sử dụng phương thức `percentile` của Numpy để xác định Minimum, 25th percentile, median, 75th, and maximum của S1 và S2.
 - Thay thế các giá trị trong S1 và S2 thành others nếu giá trị tại vị trí đó khác với giá trị mode vừa tìm được ở trên.
20. Thực hiện trên S1 các yêu cầu sau:

- (a).- Sử dụng hàm `where` trong Pandas để tìm vị trí của các số là bội số của 5 có trong S1. Nếu không tìm thấy giá trị nào là bội số của 5, cần thông báo cho người dùng biết.
- (b).- Sử dụng phương thức `diff` tạo list L1 chứa hiệu chênh lệch giữa các giá trị liên kề trong S1. Tạo tiếp L2 chứa hiệu chênh lệch giữa các giá trị liên kề trong L1 (*difference of differences*). Minh họa:

```
S= 1 3 5 8 10 11 15
Chênh lệch giữa 2 giá trị liên kề:
[nan, 2.0, 2.0, 3.0, 2.0, 1.0, 4.0]
[nan, nan, 0.0, 1.0, -1.0, -1.0, 3.0]
```

21. Tạo 1 list chứa các số nguyên từ 0-10. Trích xuất các phần tử tại các vị trí là bội số của
22. Tạo 2 Series S1 và S2 với số lượng phần tử là khác nhau ($S1n > S2n$). Giá trị các phần tử trong 2 Series được phát sinh ngẫu nhiên, không trùng nhau và được sắp xếp tăng dần. Sử dụng phương thức `Index(series1).get_loc(i)` để tìm vị trí của những phần tử trong S2 xuất hiện trong S1.
23. Tạo 1 Series chứa các chuỗi ký tự. Thực hiện:
- Chuyển ký tự đầu của các chuỗi này thành ký tự hoa và các ký tự còn lại là chữ thường.
 - Đếm số lượng ký tự có trong mỗi chuỗi con của Series
24. Tạo 1 Series chứa chuỗi về date (gồm nhiều dạng biểu diễn ngày khác nhau như: 01 Jan 2015, 10-02-2016, 20180307, 2014/05/06, 2016-04-12, 2019-04-06T11:20).

- (a).- Viết chương trình sử dụng hàm `to_datetime` để chuyển dữ liệu trong Series sang timeseries. Minh họa kết quả thực hiện:

```
0    2015-01-01 00:00:00
1    2016-10-02 00:00:00
2    2018-03-07 00:00:00
3    2014-05-06 00:00:00
4    2016-04-12 00:00:00
5    2019-04-06 11:20:00
dtype: datetime64[ns]
```

- (b).- Dựa vào kết quả trên, lấy ngày trong tháng, ngày trong năm, số tuần và thứ của các ngày đó.

```
Day of month: [1, 2, 7, 6, 12, 6]
Day of year: [1, 276, 66, 126, 103, 96]
Week number: [1, 39, 10, 19, 15, 14]
Day of week: ['Thursday', 'Sunday', 'Wednesday', 'Tuesday',
'Tuesday', 'Saturday']
```


25. Cho 1 Series chỉ chứa tháng và ngày. Tạo ra chuỗi hoàn chỉnh dạng year-month-day với day đều là 15. Minh họa:

```
S= ( 'Jan 2015', 'Feb 2016', 'Mar 2017', 'Apr 2018', 'May 2019')
=> Skq= (2015-01-15, 2016-02-15, 2017-03-15, 2018-04-15, 2019-05-15)
```

26. Cho 1 Series chứa các từ tiếng Anh. Lọc ra các từ có ít nhất 2 nguyên âm

```
S=(Red, Green, Orange, Pink, Yellow, White)
⇒ Skq= (Green, Orange, Yellow, White)
```

27. Tạo 2 Series có số lượng phần tử là bằng nhau, chứa các số nguyên có giá trị ngẫu nhiên từ 1-10. Tính khoảng cách Euclid giữa 2 Series

 Khoảng cách Euclid là chiều dài của đoạn thẳng nối 2 điểm trong không gian.