Chapter 4 - Exercise 1: Thực hiện những yêu cầu liên quan đến series

Part 1: Thực hiện các phép toán trên series

In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
# Câu 1a: Cho arr_1 là mảng số nguyên chẵn [2, 4, 6, 8, 10],
#     arr_2 là mảng số nguyên lẻ [1, 3, 5, 7, 11]
# Tạo biến kiểu Serries ser1 từ arr_1, ser2 từ arr_2
arr_1 = np.array([2, 4, 6, 8, 10])
arr_2 = np.array([1, 3, 5, 7, 11])
ser1 = pd.Series(arr_1)
ser2 = pd.Series(arr_2)
# In danh sách các phần tử của ser1 và ser2
print(ser1)
print(ser2)
```

```
0     2
1     4
2     6
3     8
4    10
dtype: int32
0     1
1     3
2     5
3     7
4    11
dtype: int32
```

In [3]:
```
# Câu 1b: Thực hiện phép toán và thể hiện kết quả của:  ser1 + ser2
ser1 + ser2
```

Out[3]:
```
0     3
1     7
2    11
3    15
4    21
dtype: int32
```

In [4]:
```
# Câu 1c: Thực hiện phép toán và thể hiện kết quả của:  ser1 - ser2
ser1 - ser2
```

Out[4]:
```
0    1
1    1
2    1
3    1
4   -1
dtype: int32
```

In [5]:
```
# Câu 1d: Thực hiện phép toán và thể hiện kết quả của:  ser1 * ser2
ser1 * ser2
```

Out[5]:
```
0      2
1     12
2     30
3     56
4    110
dtype: int32
```

In [6]:
```
# Câu 1e: Thực hiện phép toán và thể hiện kết quả của:  ser1 / ser2
ser1 / ser2
```

Out[6]:
```
0    2.000000
1    1.333333
2    1.200000
3    1.142857
4    0.909091
dtype: float64
```

In [7]:
```
# Câu 2a: Kiểm tra xem các phần tử của ser1 có > các phần tử của ser2 không?
ser1 > ser2
```

Out[7]:
```
0     True
1     True
2     True
3     True
4    False
dtype: bool
```

In [10]:
```
# Câu 2b: Kiểm tra xem các phần tử của ser1 có < các phần tử của ser2 không?
ser1 < ser2
```

Out[10]:
```
0    False
1    False
2    False
3    False
4     True
dtype: bool
```

In [11]:
```
# Câu 2c: Kiểm tra xem các phần tử của ser1 có = các phần tử của ser2 không?
(ser1 == ser2).any()
```

Out[11]:
```
False
```

In [12]:
```
# Câu 3a: Thêm 2 phần tử [6, 12] vào ser2
ser2 = pd.concat([ser2, pd.Series([6,12])], ignore_index=True)
# ser2 = ser2.append(pd.Series([6,12]), ignore_index=True)
# In lại danh sách các phần tử của ser2.
ser2
```

Out[12]:
```
0     1
1     3
2     5
3     7
4    11
5     6
6    12
dtype: int64
```

In [13]:
```
ser1
```

Out[13]:
```
0     2
1     4
2     6
3     8
4    10
dtype: int32
```

In [14]:
```
ser2
```

Out[14]:
```
0     1
1     3
2     5
3     7
4    11
5     6
6    12
dtype: int64
```

In [15]:
```
# Câu 3b: Tạo series ser3 chỉ chứa các phần tử có trong ser1 mà không có trong ser2.
#         In danh sách các phần tử của ser3
ser3 = ser1[~ser1.isin(ser2)]
print(ser3)
```

```
0     2
1     4
3     8
4    10
dtype: int32
```

In [16]:
```
np.setdiff1d(ser1,ser2) # cách khác (trả về array)
```

Out[16]:
```
array([ 2,  4,  8, 10])
```

In [17]:
```
# Câu 3c: Tạo series ser4 chỉ chứa các phần tử có trong ser2 mà không có trong ser1.
#         In danh sách các phần tử của ser4
ser4 = ser2[~ser2.isin(ser1)]
print(ser4)
```

```
0     1
1     3
2     5
3     7
4    11
6    12
dtype: int64
```

In [18]:
```
# Câu 4: Tạo series ser5 chứa các phần tử chỉ có trong ser1 và chỉ có trong ser2
#        In danh sách các phần tử của ser5
ser5 = pd.concat([ser3, ser4])
# ser5 = ser3.append(ser4)
print(ser5)
```

```
0     2
1     4
3     8
4    10
0     1
1     3
2     5
3     7
4    11
6    12
dtype: int64
```

Part 2: Truy xuất các phần tử, và thống kê thông tin trên series

In [19]:
```
# Câu 1a: Tạo series ser6 có 35 phần tử số nguyên ngẫu nhiên
# có giá trị trong khoảng từ 1 đến 9.
np.random.seed(42)
ser6 = pd.Series(np.random.randint(1,10,35))
# Cho biết kích thước (shape) của ser6
print(ser6.shape)

# Xem 5 dòng dữ liệu đầu tiên (head) và
# 5 dòng dữ liệu cuối cùng (tail) có trong ser6
print(ser6.head())
print(ser6.tail())
```

```
(35,)
0    7
1    4
2    8
3    5
4    7
dtype: int32
30    3
31    7
32    5
33    9
34    7
dtype: int32
```

In [20]:
```
# Câu 1b: In danh sách các phần tử của ser6 theo dạng array
print(ser6.values)
```

```
[7 4 8 5 7 3 7 8 5 4 8 8 3 6 5 2 8 6 2 5 1 6 9 1 3 7 4 9 3 5 3 7 5 9 7]
```

In [21]:
```
print(ser6.tolist())
```

```
[7, 4, 8, 5, 7, 3, 7, 8, 5, 4, 8, 8, 3, 6, 5, 2, 8, 6, 2, 5, 1, 6, 9, 1, 3, 7, 4, 9,
3, 5, 3, 7, 5, 9, 7]
```

In [22]:
```
# Câu 1c: Cho biết thông tin thống kê chung (describe()) của ser6
ser6.describe()
```

Out[22]:
```
count    35.000000
mean      5.428571
std       2.342519
min       1.000000
25%       3.500000
50%       5.000000
75%       7.000000
max       9.000000
dtype: float64
```

In [23]:
```
# Câu 1d: Cho biết tổng của các phần tử có trong ser6
ser6.sum()
```

Out[23]:
```
190
```

In [24]:
```
# Câu 1e: Cho biết phần tử có tần suất xuất hiện nhiều nhất trong ser6
ser6.mode()
```

Out[24]:
```
0    5
1    7
dtype: int32
```

In [25]:
```
from scipy.stats import mode
mode(ser6)
```

Out[25]:
```
ModeResult(mode=array([5]), count=array([6]))
```

In [26]:
```
from collections import Counter
Counter(ser6)
```

Out[26]:
```
Counter({7: 6, 4: 3, 8: 5, 5: 6, 3: 5, 6: 3, 2: 2, 1: 2, 9: 3})
```

In [27]:
```
# Câu 2:  Liệt kê các dòng trong ser6 mà giá trị chia hết cho 2 và cho 3
ser6[(ser6 % 2 == 0) & (ser6 % 3 == 0)]
```

Out[27]:
```
13    6
17    6
21    6
dtype: int32
```

In [28]:
```
# Câu 3: In các phần tử ở vị trí 0, 5, 10, 15 trong ser6
pos = [0, 5, 10, 15]
ser6[pos]
```

Out[28]:
```
0     7
5     3
10    8
15    2
dtype: int32
```

In [29]:
```
# Câu 4: In ra các giá trị unique (array) trong ser6
ser6.unique()
```

Out[29]:
```
array([7, 4, 8, 5, 3, 6, 2, 1, 9])
```

In [30]:
```
# Câu 5: Tạo series ser7 với mỗi phần tử có
# giá trị = lập phương của phần tử trong ser6.
import math
ser7 = ser6.map(lambda x: math.pow(x,3))

# Xem 5 dòng dữ liệu đầu tiên (head) của ser7
print(ser7.head())
```

```
0    343.0
1     64.0
2    512.0
3    125.0
4    343.0
dtype: float64
```

Part 3: Tạo series từ list, chuỗi và biểu thức điều kiện

In [31]:
```
# Câu 1: Cho list sau:
lst = ["abc", "defg", "htlmj", "dfg", "ljsac"]
```

In [32]:
```
# Câu 1a: Tạo series ser_chuoi từ lst
ser_chuoi = pd.Series(lst)
print(ser_chuoi)
```

```
0      abc
1     defg
2    htlmj
3      dfg
4    ljsac
dtype: object
```

In [33]:
```
# Câu 1b: Tạo series ser_dodai với mỗi phần tử có giá trị
# là chiều dài của mỗi phần tử trong ser_chuoi
ser_dodai = ser_chuoi.map(lambda x: len(x))
ser_dodai
```

Out[33]:
```
0    3
1    4
2    5
3    3
4    5
dtype: int64
```

In [34]:
```
# Câu 2: Cho ser = pd.Series(np.array([1, 2, 4, 5, 8, 7, 6, 9])).
# Sử dụng biểu thức điều kiện thích hợp để in ra các dòng trong ser
# có giá trị là số nguyên tố

ser = pd.Series(np.array([1, 2, 4, 5, 8, 7, 6, 9]))
# Kiểm tra số nguyên tố
def test_prime(number):
    count = 0
    for i in range(1, number + 1):
        if number % i == 0:
            count += 1
    return count == 2

ser_nguyento = ser.map(lambda x: test_prime(x))

print(ser[ser_nguyento])
```

```
1    2
3    5
5    7
dtype: int32
```

In [35]:
```
# Câu 3a: Cho mẫu email như sau:
pattern ='[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\\.[A-Za-z]{2,4}'

# Tạo một series ser_ch, với mỗi phần tử trong ser_ch là một chuỗi
# Gợi ý: 'reading newspaper from tuoitre.vn', 'tubirona@gmail.com',
# 'nguyen.nn@yahoo.com', 'tran_2014@hotmail.com.vn'
ser_ch = pd.Series(['reading newspaper from tuoitre.vn', 'tubirona@gmail.com',
                    'nguyen.nn@yahoo.com', 'tran_2014@hotmail.com.vn'])

# In ra những dòng trong ser_ch thỏa điều kiện chuỗi là email
import re
is_email = ser_ch.map(lambda x: bool(re.match(pattern, x)))
ser_ch[is_email]
```

Out[35]:
```
1          tubirona@gmail.com
2         nguyen.nn@yahoo.com
3    tran_2014@hotmail.com.vn
dtype: object
```

In [36]:
```
# Câu 3b: In ra những dòng trong ser_ch thỏa điều kiện
# chuỗi là email và kết thúc là vn
is_email = ser_ch.map(lambda x: bool(re.match(pattern, x))
                      and bool(re.findall('vn$', x)))
ser_ch[is_email]
```

Out[36]:
```
3    tran_2014@hotmail.com.vn
dtype: object
```

In [37]:
```
# Câu 4a: Cho series:
ser_names = pd.Series(['Manufacturer', 'Model', 'CarType', 'Min_Price',
       'Price', 'Max_Price',
       'MPG_city', 'MPG_highway', 'AirBags', 'DriveTrain', 'Cylinders',
       'EngineSize', 'Horsepower', 'RPM', 'Rev_per_mile', 'Man_trans_avail',
       'Fuel_tank_capacity', 'Passengers', 'Length', 'Wheelbase', 'Width',
       'Turn_circle', 'Rear_seat_room', 'Luggage_room', 'Weight', 'Origin',
       'Make'])
# Sử dụng biểu thức điều kiện thích hợp để in ra các dòng của ser_names
# thỏa điều kiện trong chuỗi có chữ 'Price'

Price = ser_names.map(lambda s: 'Price' in s)

names_Price = ser_names[Price]

print(names_Price)
```

```
3    Min_Price
4        Price
5    Max_Price
dtype: object
```

In [38]:
```
# Câu 4b: Cho series:
ser_names = pd.Series(['Manufacturer', 'Model', 'CarType', 'Min_Price',
       'price', 'Max_price',
       'MPG_city', 'MPG_highway', 'AirBags', 'DriveTrain', 'Cylinders',
       'EngineSize', 'Horsepower', 'RPM', 'Rev_per_mile', 'Man_trans_avail',
       'Fuel_tank_capacity', 'Passengers', 'Length', 'Wheelbase', 'Width',
       'Turn_circle', 'Rear_seat_room', 'Luggage_room', 'Weight', 'Origin',
       'Make'])
# In ra các dòng của ser_names trong chuỗi
# có chữ 'Price' (không phân biệt thường/hoa)
Price = ser_names.map(lambda s: 'price' in s.lower())

names_Price = ser_names[Price]

print(names_Price)
```

```
3    Min_Price
4        price
5    Max_price
dtype: object
```

In [ ]: