QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

How to ...

```
import matplotlib.pyplot as plt
import pandas as pd
imprt numpy as np
```

✓ Python

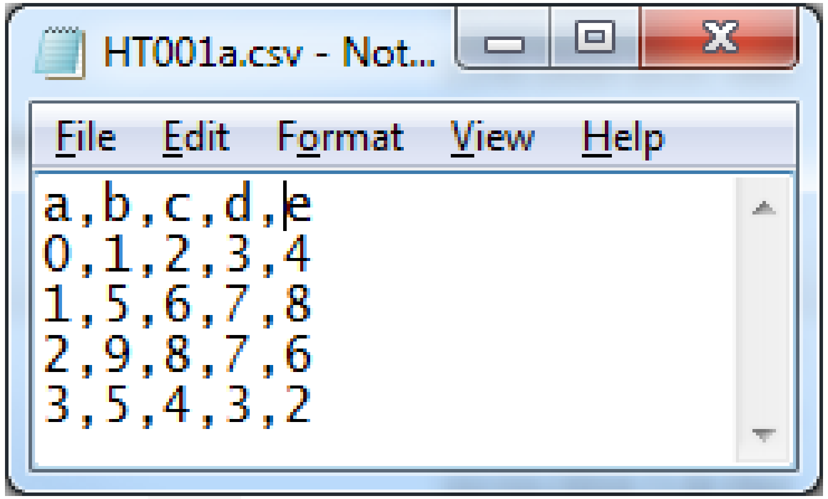
✓ MATLAB

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

HT001: **Use one command** to import data from a CSV file, **HT001a.csv**, using the **first row as column labels**, and the **first column as row labels**. Name the data imported as **data**.



```
a,b,c,d,e
0,1,2,3,4
1,5,6,7,8
2,9,8,7,6
3,5,4,3,2
```

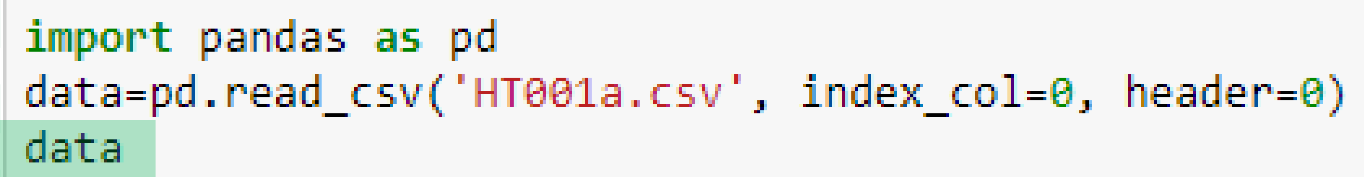
✓ Python: **pandas.read_csv**

```
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
```

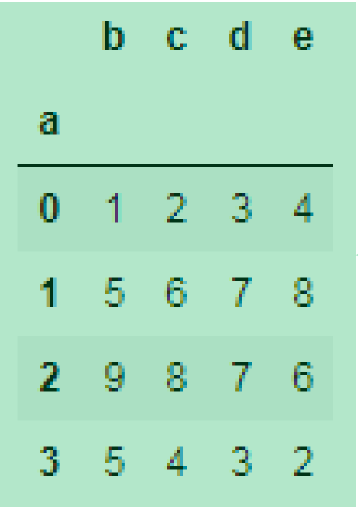
099

? MATLAB: **readtable**

```
data=readtable('HT001a.csv','ReadRowNames',true,'ReadVariableNames',true)
```

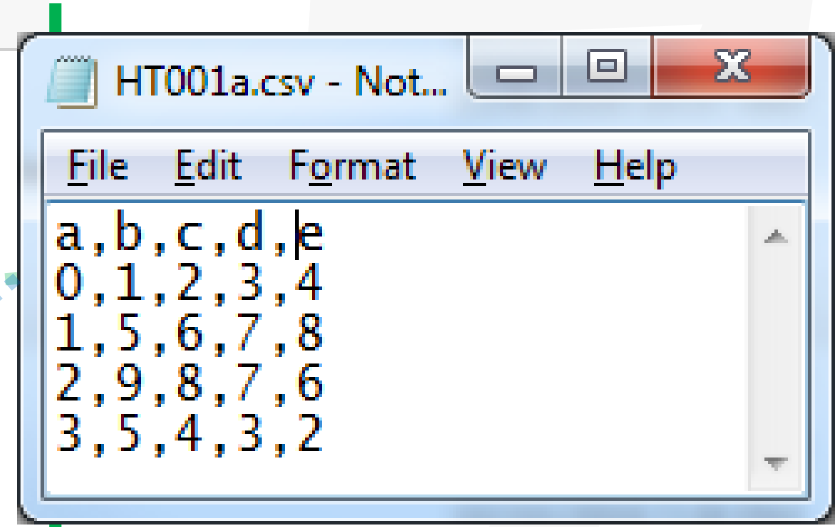


```
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
data
```

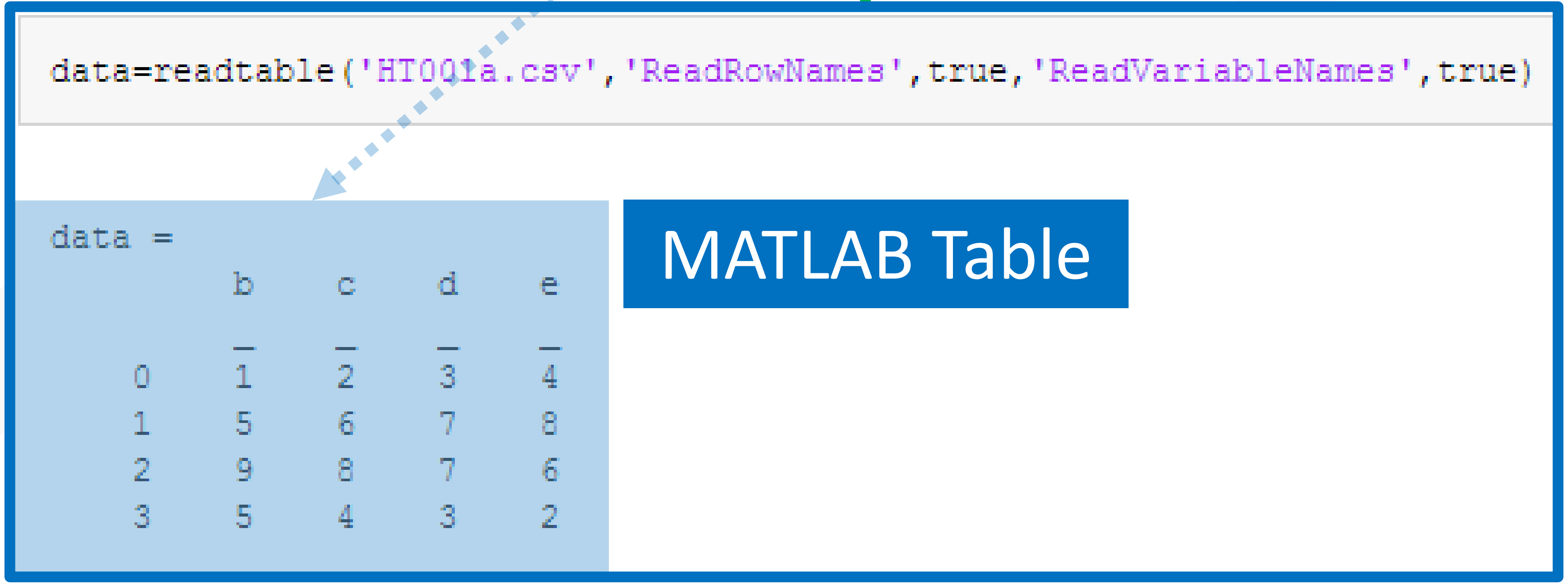


```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
```

Pandas DataFrame



```
a,b,c,d,e
0,1,2,3,4
1,5,6,7,8
2,9,8,7,6
3,5,4,3,2
```



```
data=readtable('HT001a.csv','ReadRowNames',true,'ReadVariableNames',true)
```

```
data =
              b      c      d      e
              _      _      _      _
      0       1      2      3      4
      1       5      6      7      8
      2       9      8      7      6
      3       5      4      3      2
```

MATLAB Table

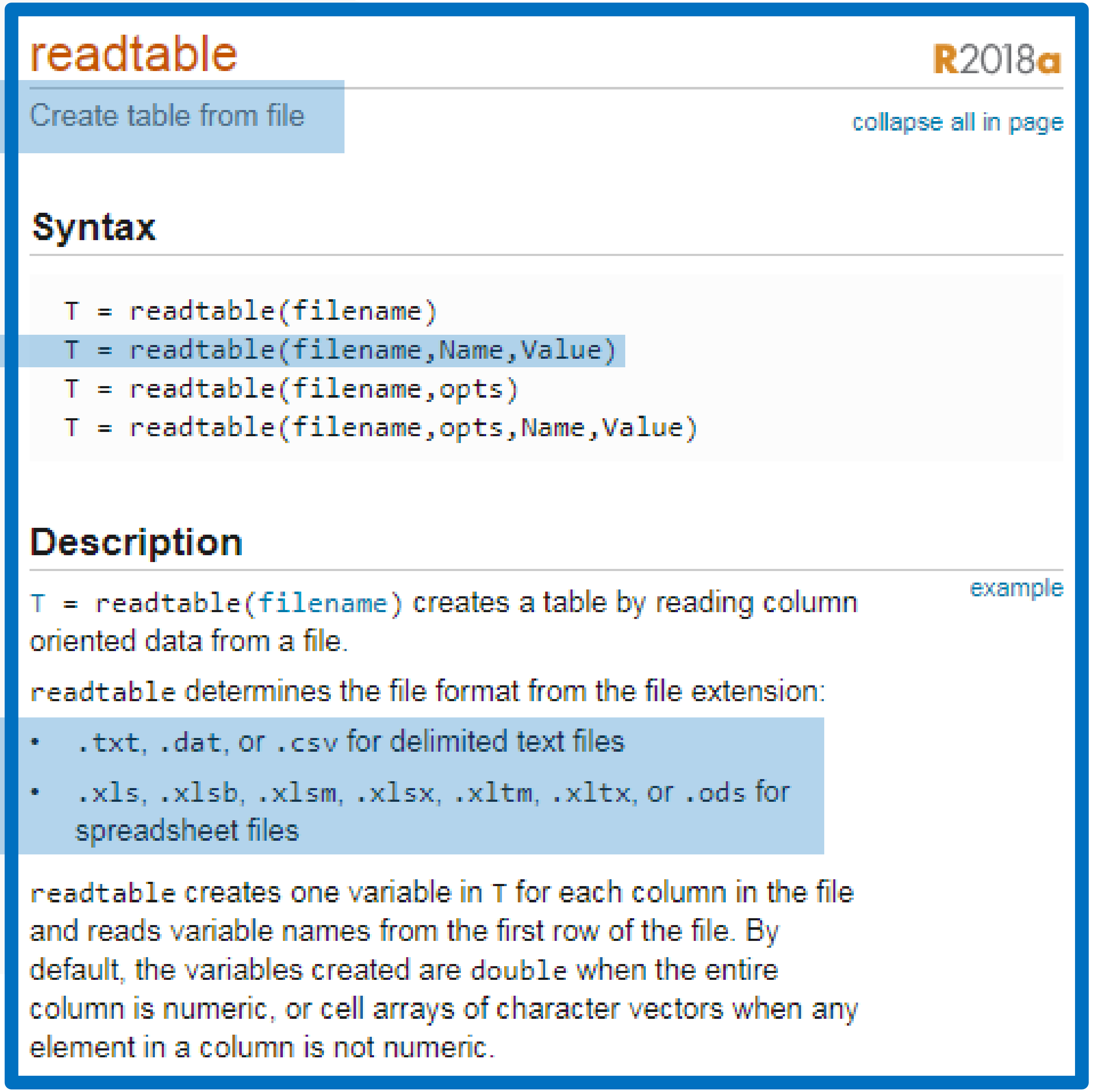
099

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance

https://www.mathworks.com/help/matlab/ref/readtable.html



readtable

R2018a

Create table from file

collapse all in page

Syntax

```
T = readtable(filename)
T = readtable(filename,Name,Value)
T = readtable(filename,opts)
T = readtable(filename,opts,Name,Value)
```

Description

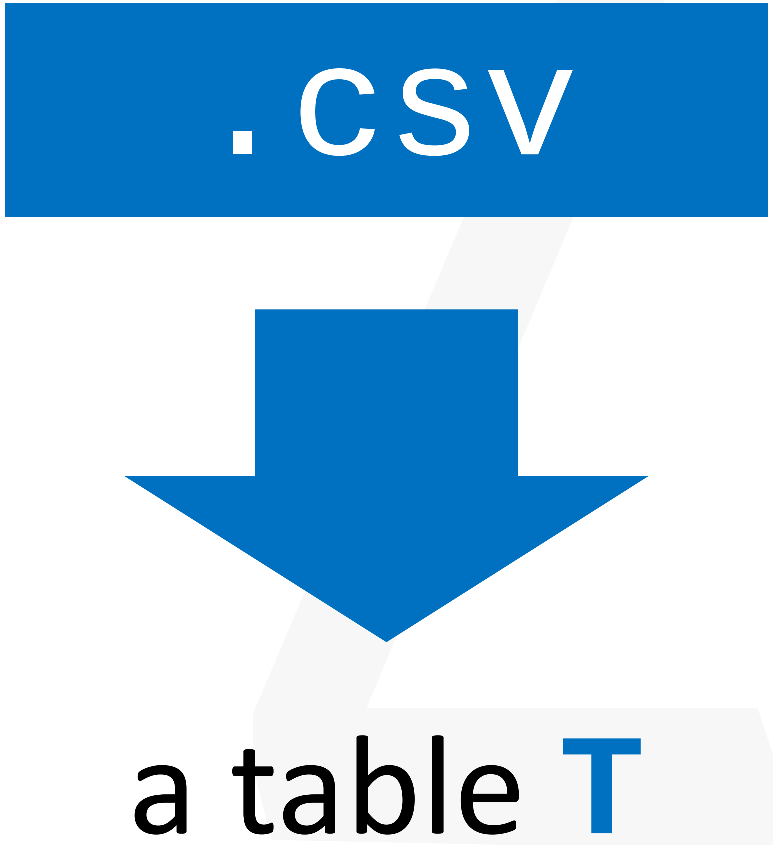
example

T = readtable(filename) creates a table by reading column oriented data from a file.

readtable determines the file format from the file extension:

- .txt, .dat, or .csv for delimited text files
- .xls, .xlsb, .xlsm, .xlsx, .xltm, .xltx, or .ods for spreadsheet files

readtable creates one variable in T for each column in the file and reads variable names from the first row of the file. By default, the variables created are double when the entire column is numeric, or cell arrays of character vectors when any element in a column is not numeric.



.csv

a table T

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

What if the file is newfolder\newfile.csv?

```
✓ data=pd.read_csv('newfolder\\newfile.csv')
✓ data=pd.read_csv(r'newfolder\newfile.csv')
```

(Mac)
```
data=pd.read_csv('newfolder/newfile.csv')
```

100

```
✓ data=readtable('newfolder\newfile.csv')
```

(Mac)
```
data=readtable('newfolder/newfile.csv')
```

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Name-Value Pair Arguments

- □ 'ReadVariableNames' – **Read first row as variable names**
- □ 'ReadRowNames' – **Read first column as row names**
- □ 'DatetimeType' – Type for imported date and time date
- □ 'Delimiter' – Field delimiter character
- □ 'HeaderLines' – Lines to skip
- □ 'Format' – Column format
- □ 'EmptyValue' – Returned value for empty numeric fields
- □ 'DurationType' – Output data type of duration data

Spreadsheet Files Only

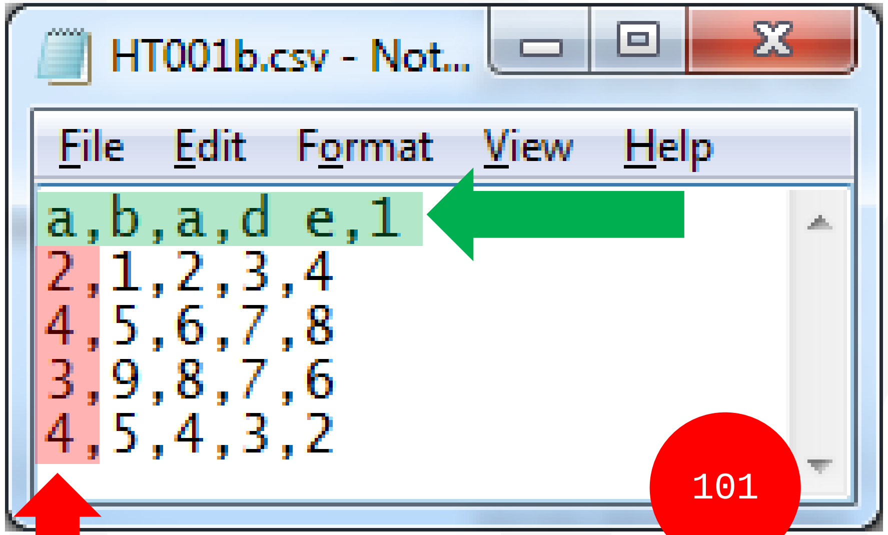
- □ 'Sheet' – Worksheet to read
- □ 'Range' – Portion of worksheet to read

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

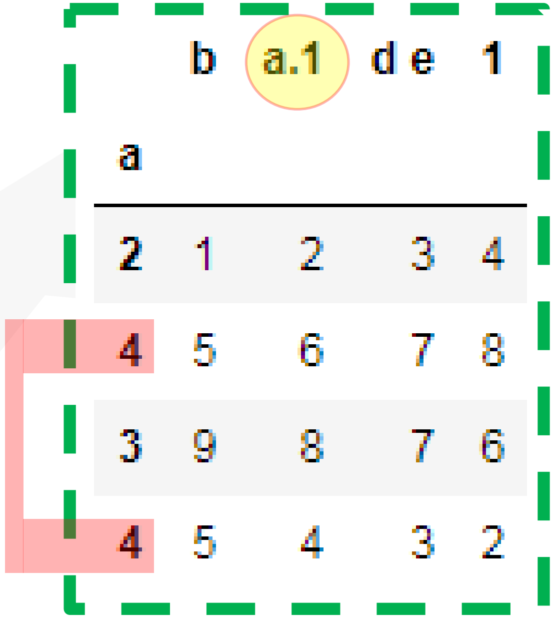
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

What if the labels are as follows?



```
a,b,a,d e,1
2,1,2,3,4
4,5,6,7,8
3,9,8,7,6
4,5,4,3,2
```

101



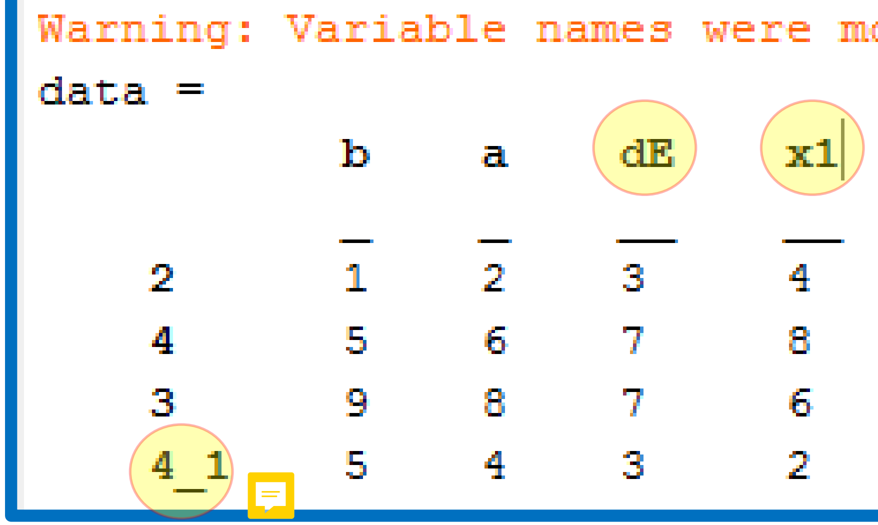
```
     b  a.1  d e  1
a
2    1    2    3  4
4    5    6    7  8
3    9    8    7  6
4    5    4    3  2
```

`mangle_dupe_cols=True`

mangle_dupe_cols : *boolean, default True*

Duplicate columns will be specified as 'X', 'X.1', ...'X.N', rather than 'X'...'X'. Passing in False will cause data to be overwritten if there are duplicate names in the columns.

Remember DataFrame is a dict-like container of Series??



```
Warning: Variable names were modified to make them valid MATLAB identifiers.
data =

               b    a    dE    x1
               _    _    __    __

      2        1    2    3     4
      4        5    6    7     8
      3        9    8    7     6
      4_1      5    4    3     2
```

```
d e -> dE
1 -> x1   (column label)
4 -> 4_1 (row label)
```

102

no duplicate names

QF666 Programming and Computational Finance

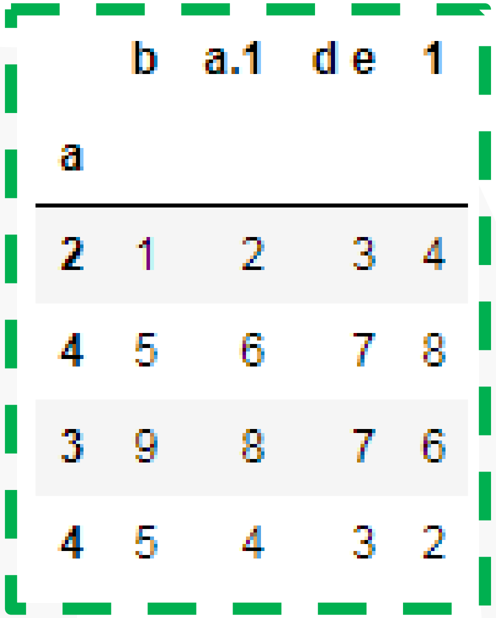
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

SMU SINGAPORE MANAGEMENT UNIVERSITY

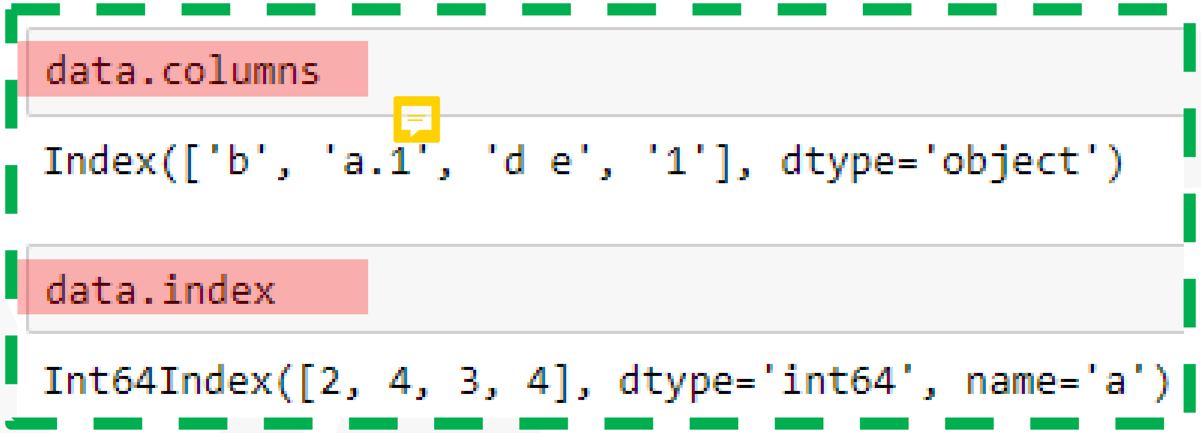
QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance

(continued)

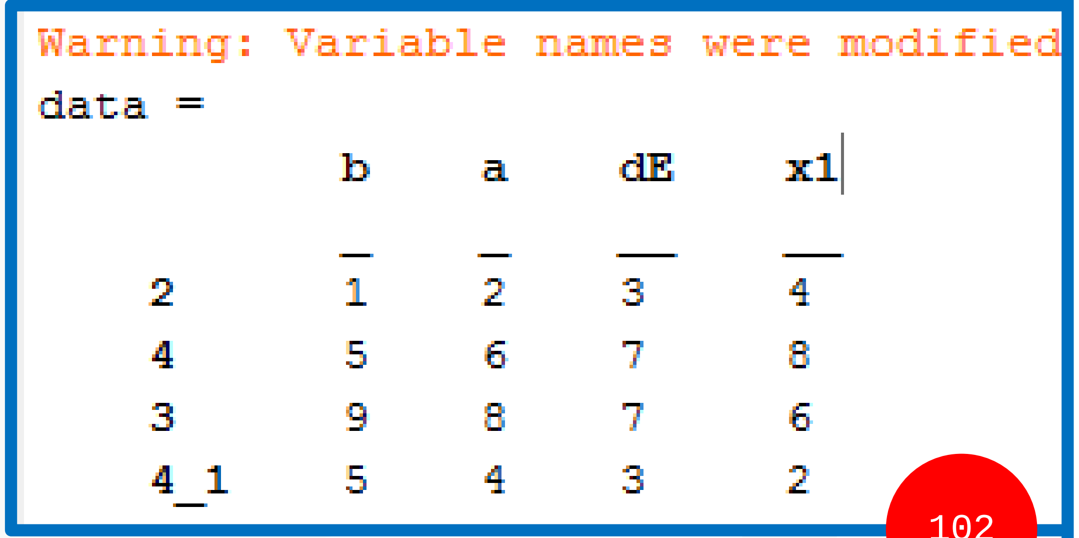


```
   b  a.1  d e  1
a
2  1  2    3    4
4  5  6    7    8
3  9  8    7    6
4  5  4    3    2
```



```
data.columns

Index(['b', 'a.1', 'd e', '1'], dtype='object')

data.index

Int64Index([2, 4, 3, 4], dtype='int64', name='a')
```

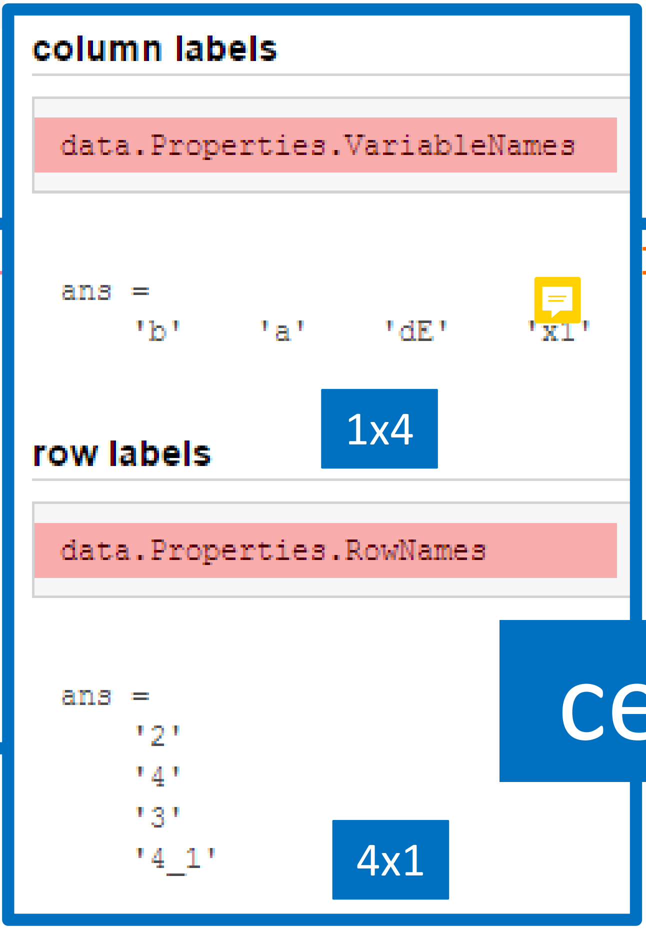
101



```
Warning: Variable names were modified
data =
              b      a      dE      x1
              _      _      __      __
   2          1      2      3       4
   4          5      6      7       8
   3          9      8      7       6
   4_1        5      4      3       2
```

```
LAB identifiers.
```

102



column labels

```
data.Properties.VariableNames

ans =
    'b'    'a'    'dE'    'x1'
```

1x4

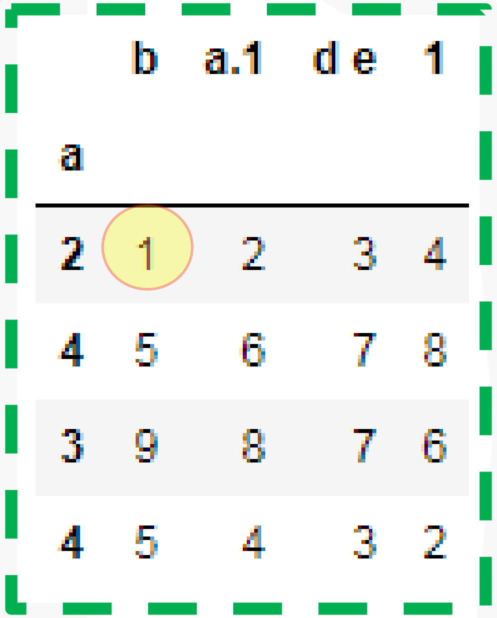
row labels

```
data.Properties.RowNames

ans =
    '2'
    '4'
    '3'
    '4_1'
```

4x1

cell array

HT002: Use an appropriate **position-based indexing/slicing** method to select data in **`data`**.



```
    b  a.1  d e  1
a
2   1    2    3  4
4   5    6    7  8
3   9    8    7  6
4   5    4    3  2
```

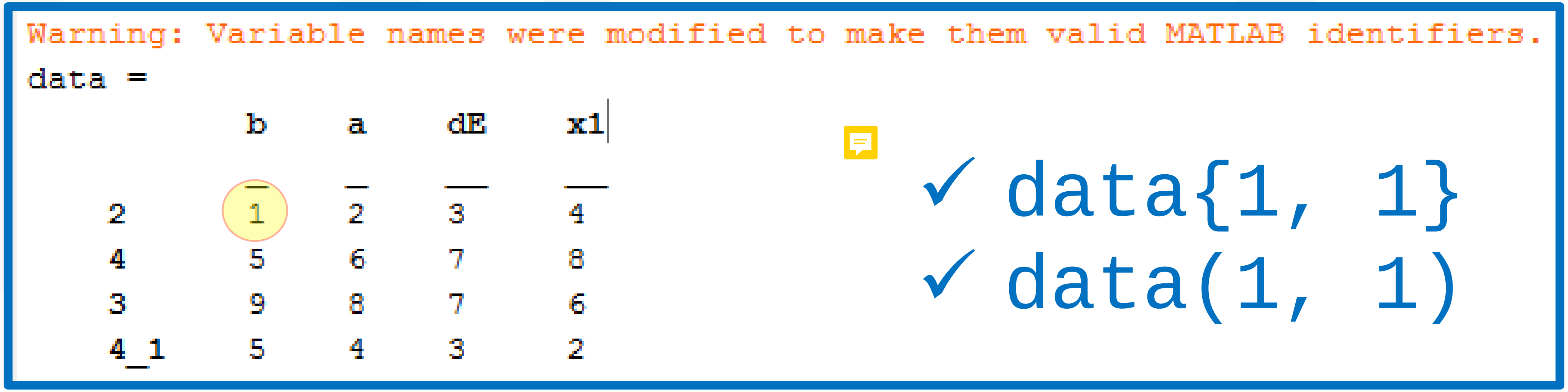
✓ `data.iloc[0,0]`

× `data[0][0]`

× `data[0,0]`

103

DataFrame ⇨ Number



```
Warning: Variable names were modified to make them valid MATLAB identifiers.
data =
                b     a     dE     x1
                _     _     __     __
    2           1     2     3      4
    4           5     6     7      8
    3           9     8     7      6
    4_1         5     4     3      2
```

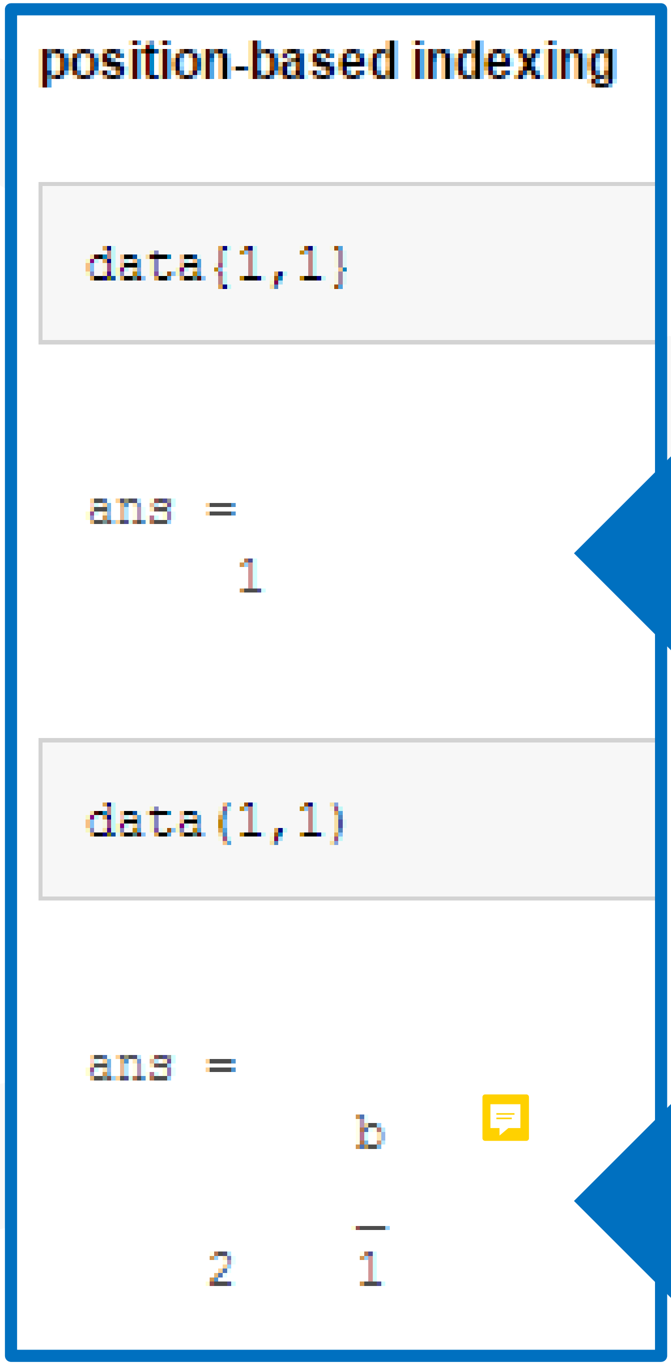
✓ `data{1, 1}`

✓ `data(1, 1)`

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



position-based indexing

```
data{1,1}
```

```
ans =
     1
```

```
data(1,1)
```

```
ans =
          b
          _
  2       1
```

1x1 **double**

103

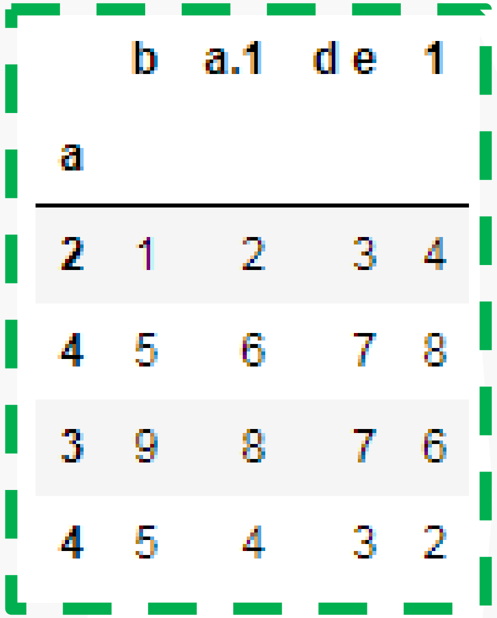
1x1 **table**

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

(continued)

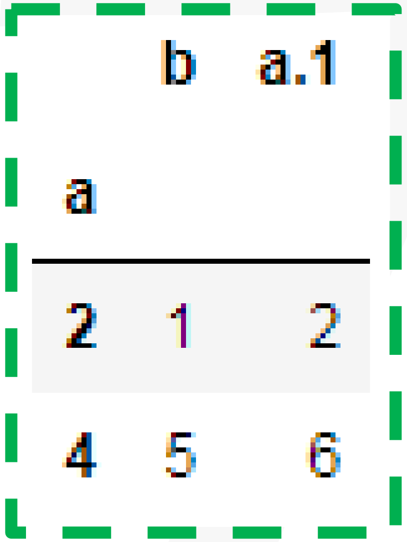
position-based slicing (1)



	b	a.1	d e	1
a				
2	1	2	3	4
4	5	6	7	8
3	9	8	7	6
4	5	4	3	2

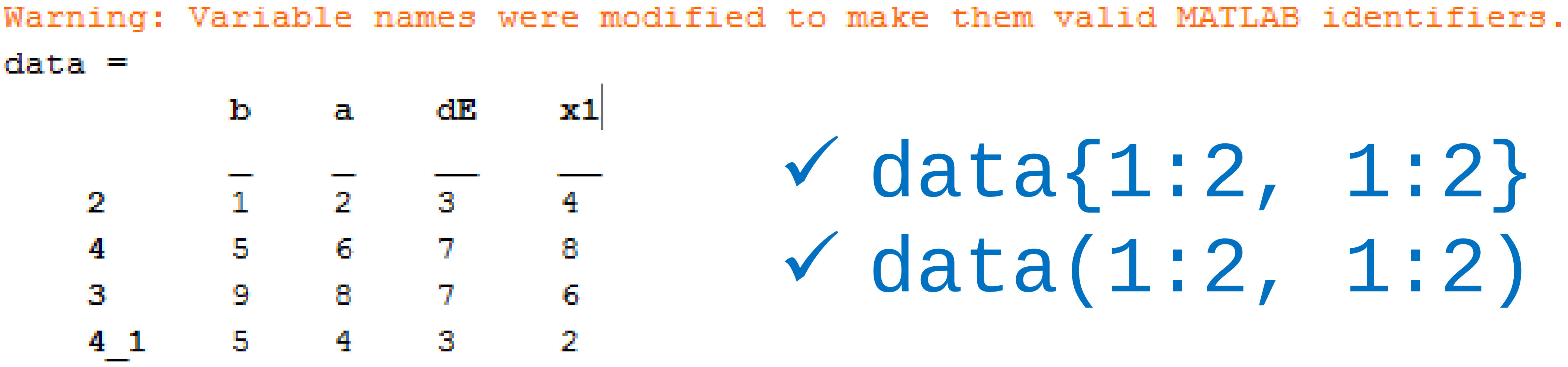
data.iloc[0:2,0:2]

DataFrame⇨DataFrame



	b	a.1
a		
2	1	2
4	5	6

103



```
Warning: Variable names were modified to make them valid MATLAB identifiers.
data =
                b     a     dE     x1
                _     _     __     __
      2         1     2     3      4
      4         5     6     7      8
      3         9     8     7      6
      4_1       5     4     3      2
```

✓ data{1:2, 1:2}

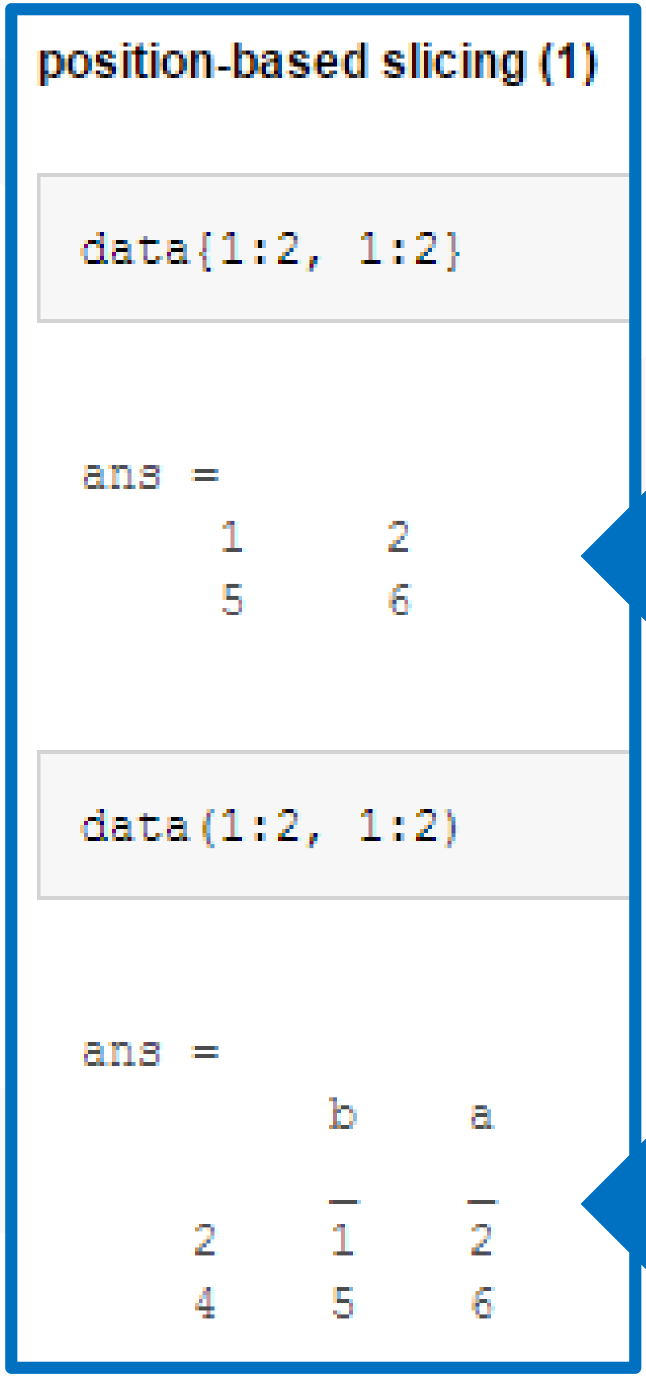
✓ data(1:2, 1:2)

https://www.mathworks.com/help/matlab/matlab_prog/access-data-in-a-table.html

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



position-based slicing (1)

```
data{1:2, 1:2}
```

```
ans =

     1     2
     5     6
```

```
data(1:2, 1:2)
```

```
ans =

           b    a
           _    _

     2     1    2
     4     5    6
```

2x2 **double**

103

2x2 **table**

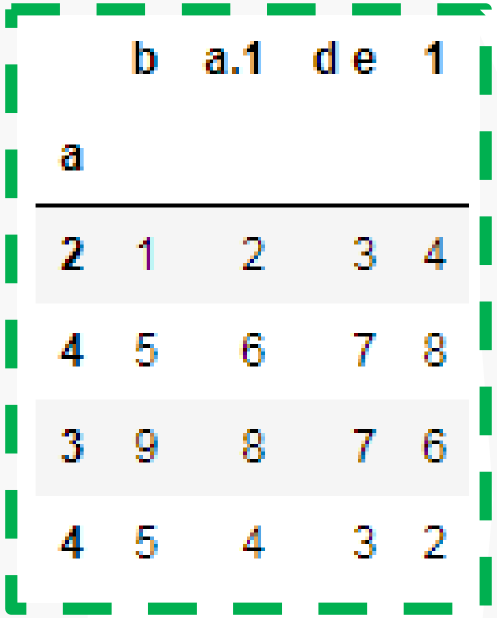
SMU
SINGAPORE MANAGEMENT
UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

(continued)

position-based slicing (2)

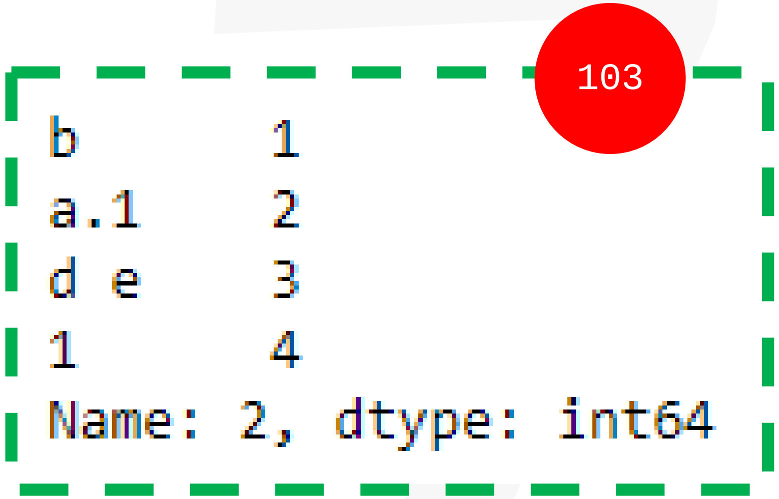


```
     b  a.1  d e  1
a
2    1  2    3    4
4    5  6    7    8
3    9  8    7    6
4    5  4    3    2
```

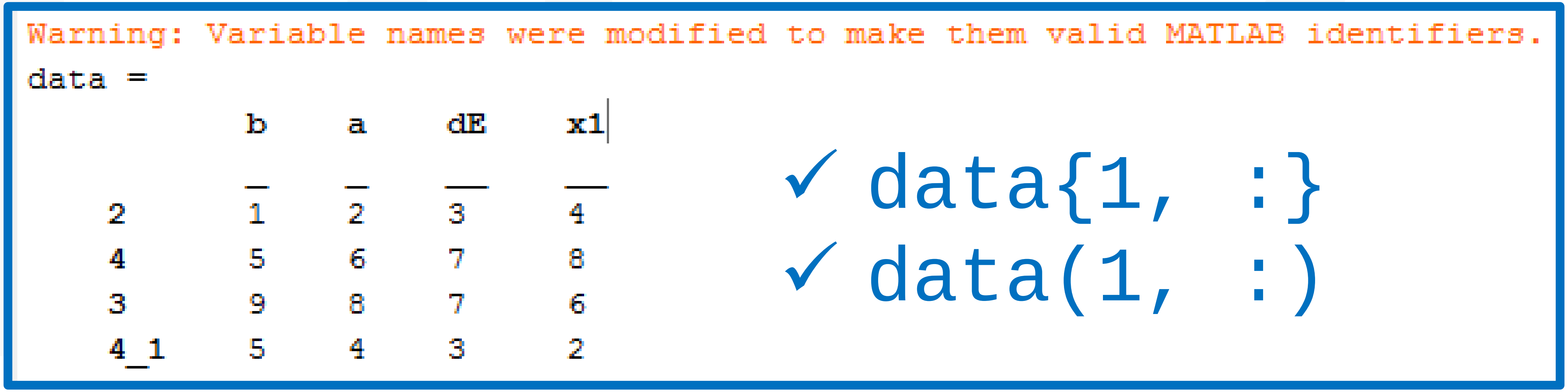
(the 1st row)

data.iloc[0,:]

DataFrame⇨Series



```
b       1
a.1     2
d e     3
1       4
Name: 2, dtype: int64
```



```
Warning: Variable names were modified to make them valid MATLAB identifiers.
data =
              b    a    dE    x1
              _    _    __    __
      2       1    2    3     4
      4       5    6    7     8
      3       9    8    7     6
      4_1     5    4    3     2
```

✓ data{1, :}

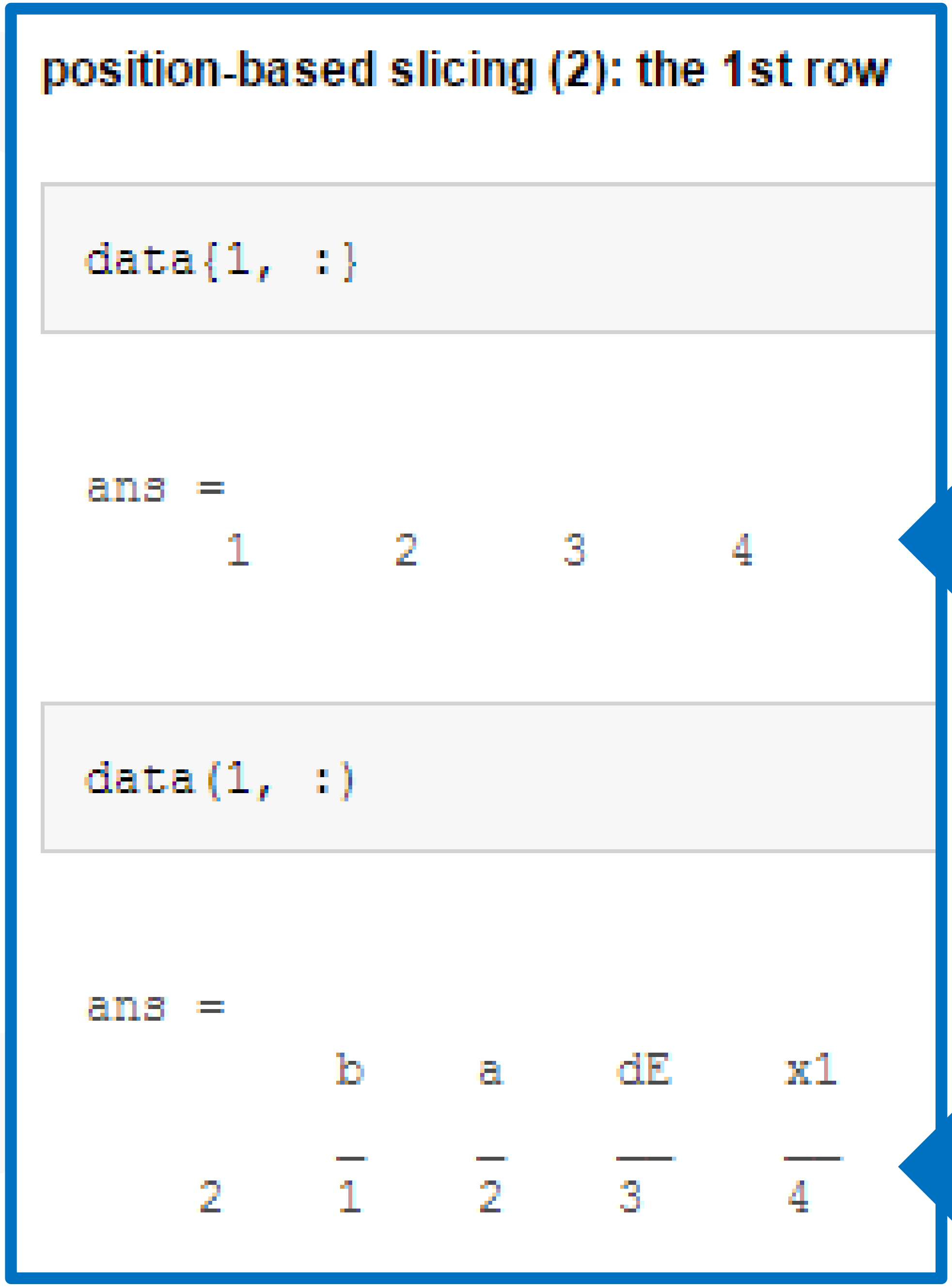
✓ data(1, :)

https://www.mathworks.com/help/matlab/matlab_prog/access-data-in-a-table.html

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



position-based slicing (2): the 1st row

```
data{1, :}
```

```
ans =

     1     2     3     4
```

```
data(1, :)
```

```
ans =

          b    a    dE    x1
          _    _    __    __
     2    1    2    3     4
```

1x4 **double**

103

1x4 **table**

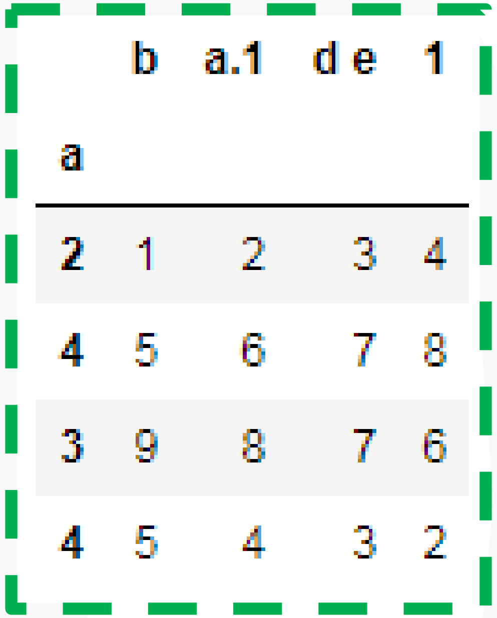
SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

(continued)

position-based slicing (3)

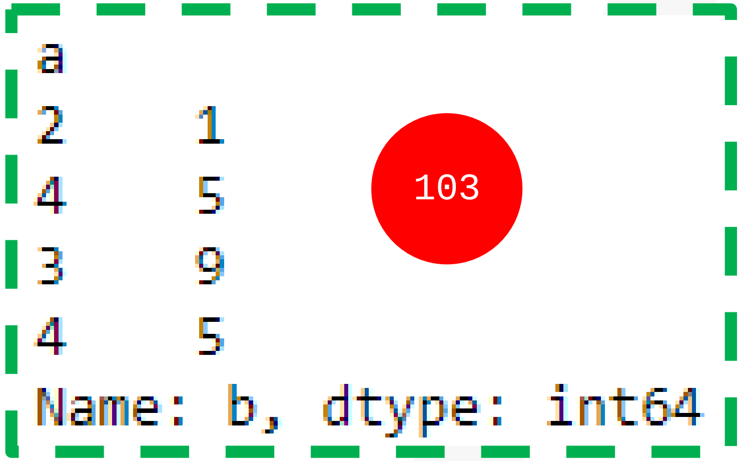


```
     b  a.1  d e  1
a
2    1  2    3    4
4    5  6    7    8
3    9  8    7    6
4    5  4    3    2
```

(the 1st column)

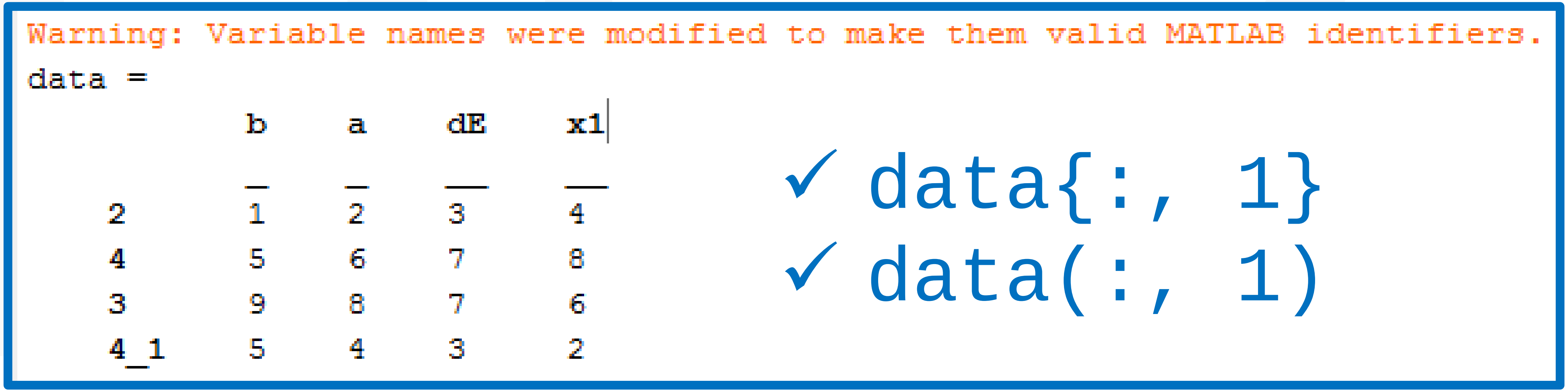
data.iloc[:,0]

DataFrame⇨Series



```
a
2      1
4      5
3      9
4      5
Name: b, dtype: int64
```

103



```
Warning: Variable names were modified to make them valid MATLAB identifiers.
data =
               b     a     dE     x1
               _     _     __     __
    2          1     2     3      4
    4          5     6     7      8
    3          9     8     7      6
    4_1        5     4     3      2
```

✓ data{:, 1}

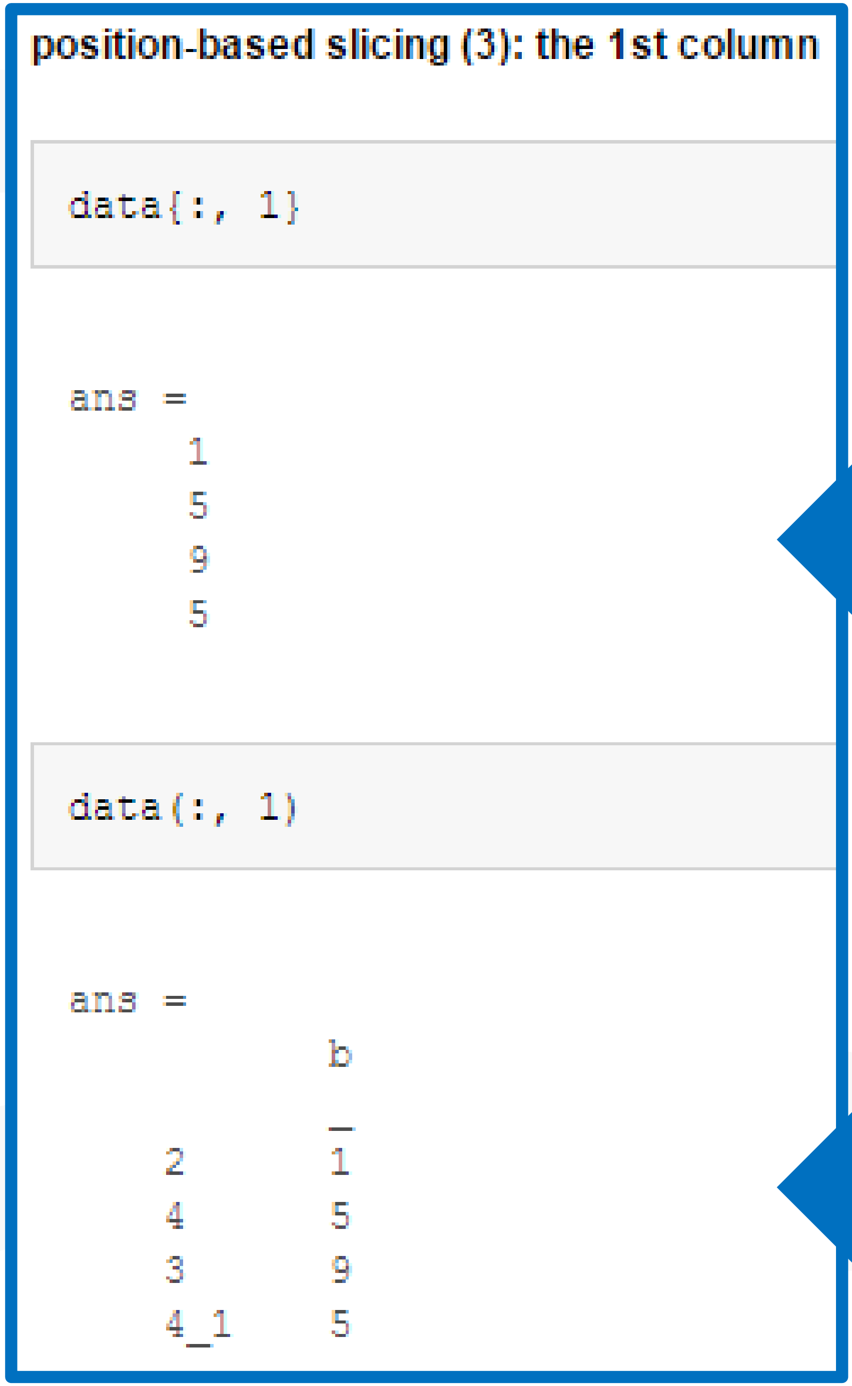
✓ data(:, 1)

https://www.mathworks.com/help/matlab/matlab_prog/access-data-in-a-table.html

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



position-based slicing (3): the 1st column

```
data{:, 1}
```

```
ans =
      1
      5
      9
      5
```

```
data(:, 1)
```

```
ans =
              b
              _
    2         1
    4         5
    3         9
    4_1       5
```

4x1 **double**

103

4x1 **table**

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Assume the table consists of numbers only.

104

data.values ⇔ **data{:, :}**

⇨ Numpy 2D ndarray

⇨ 2D array

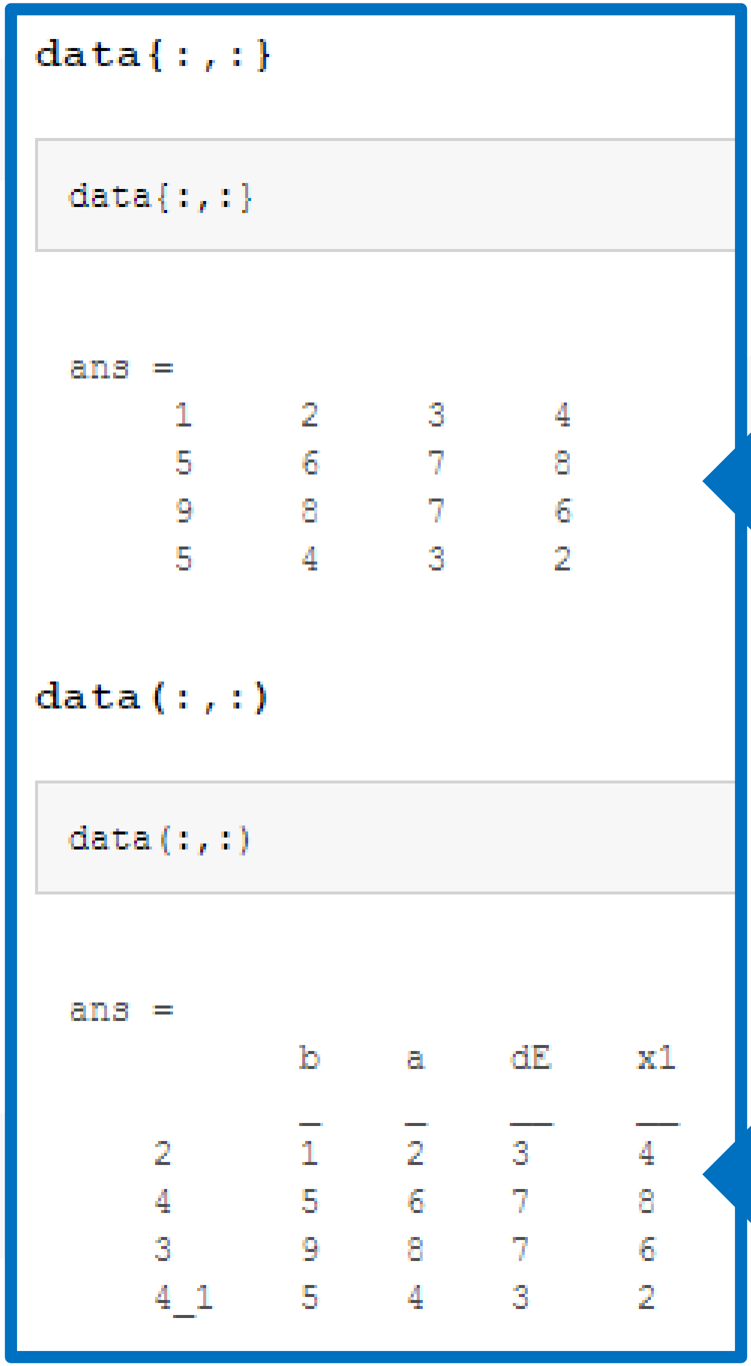
data.Variables

R2018a

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



data{:,:}

```
data{:,:}
```

```
ans =

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

data(:,:)

```
data(:,:)
```

```
ans =

             b    a    dE    x1
             _    _    __    __

    2        1    2    3     4
    4        5    6    7     8
    3        9    8    7     6
    4_1      5    4    3     2
```

4x4 **double**

104

4x4 **table**

QF666 Programming and Computational Finance

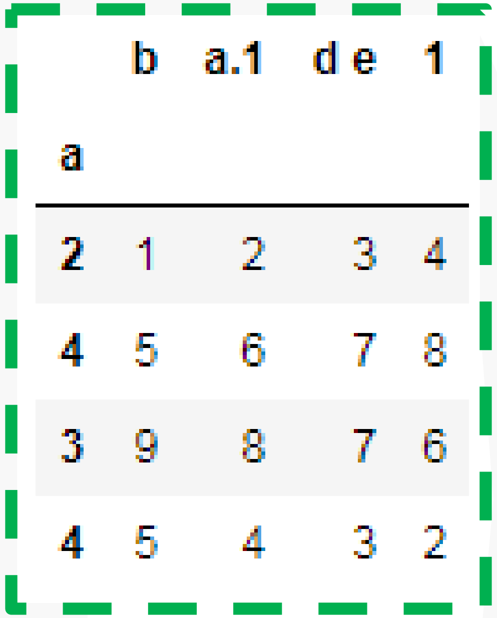
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Slide
185/307

(continued)

position-based fancy indexing

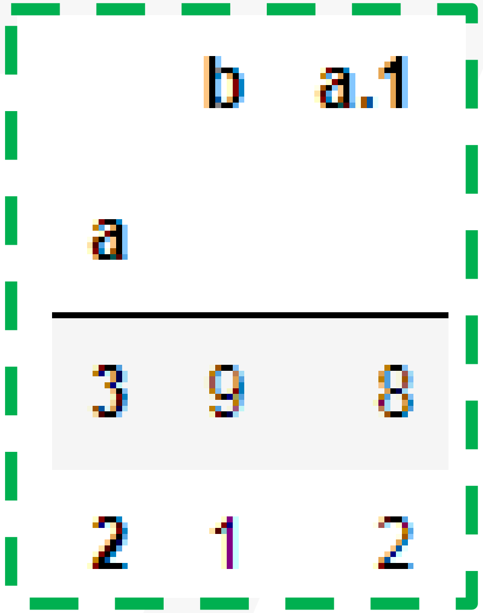
103



a	b	a.1	d e	1
2	1	2	3	4
4	5	6	7	8
3	9	8	7	6
4	5	4	3	2

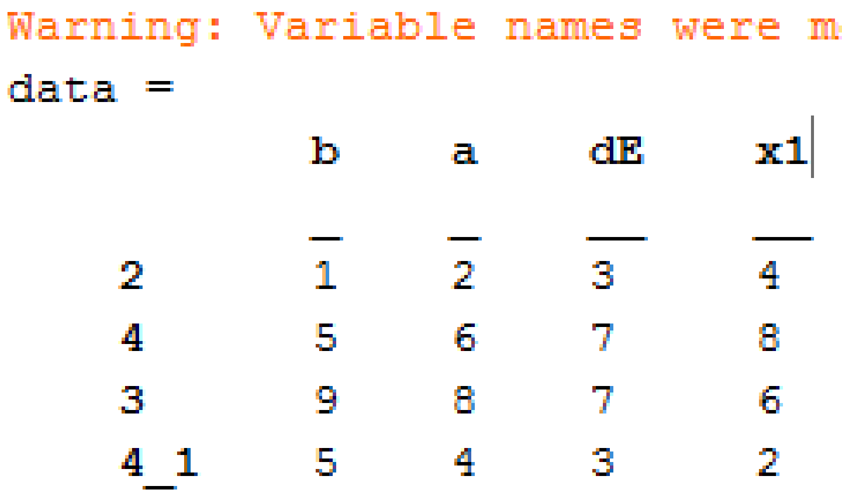
data.iloc[[2,0],[0,1]]

DataFrame⇨DataFrame



a	b	a.1
3	9	8
2	1	2

(Dr. Z: Fancy indexing is similar to slicing.)



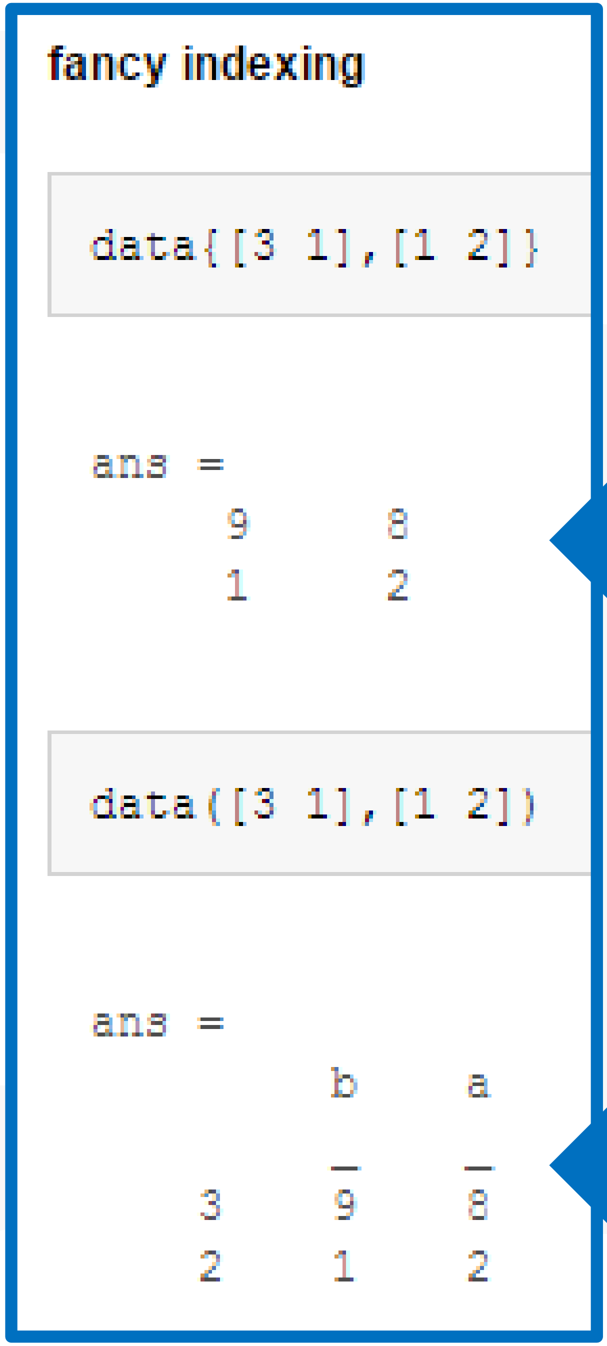
```
Warning: Variable names were modified to make them valid MATLAB identifiers.
data =

                b     a     dE     x1
               _     _     __     __
       2       1     2     3      4
       4       5     6     7      8
       3       9     8     7      6
       4_1     5     4     3      2
```

✓ data{[3 1], [1 2]}

✓ data([3 1], [1 2])

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



fancy indexing

```
data{[3 1],[1 2]}
```

```
ans =
         9         8
         1         2
```

```
data([3 1],[1 2])
```

```
ans =
              b         a
              _         _
       3      9         8
       2      1         2
```

2x2 **double**

2x2 **table**

103

QF666
Programming and Computational Finance

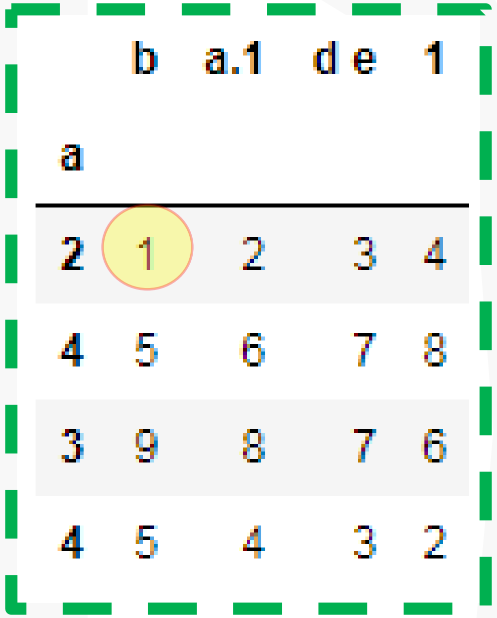
Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance

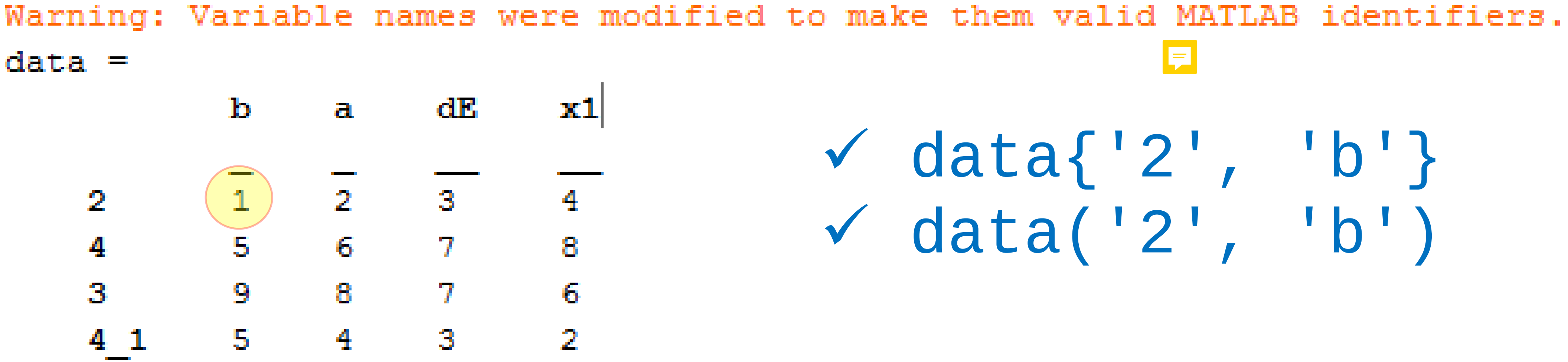
HT003: Use an appropriate **label-based indexing/slicing** method to select data in **`data`**.



```
   b  a.1  d e  1
a
2  1   2    3   4
4  5   6    7   8
3  9   8    7   6
4  5   4    3   2
```

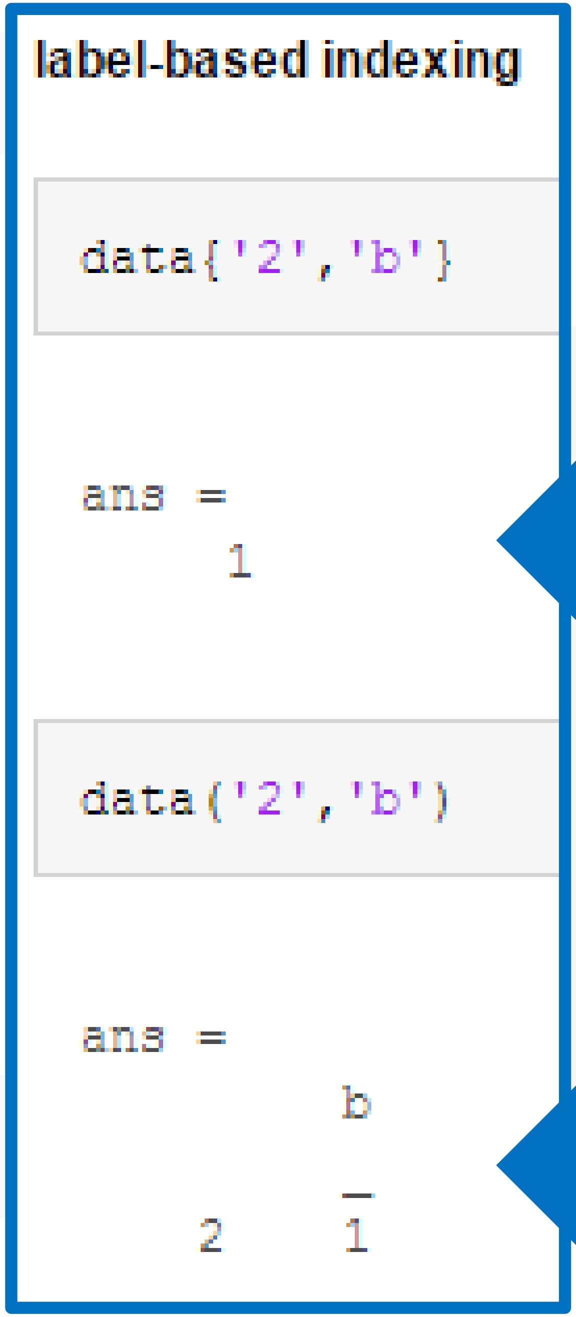
✓ `data.loc[2,'b']`
✓ `data['b'][2]`
× `data[2,'b']`

105



```
Warning: Variable names were modified to make them valid MATLAB identifiers.
data =
            b    a    dE    x1
            _    _    __    __
    2       1    2    3     4
    4       5    6    7     8
    3       9    8    7     6
    4_1     5    4    3     2
```

✓ `data{'2', 'b'}`
✓ `data('2', 'b')`



label-based indexing

```
data{'2','b'}

ans =

     1

data('2','b')

ans =

            b
            _

     2      1
```

1x1 **double**

105

1x1 **table**

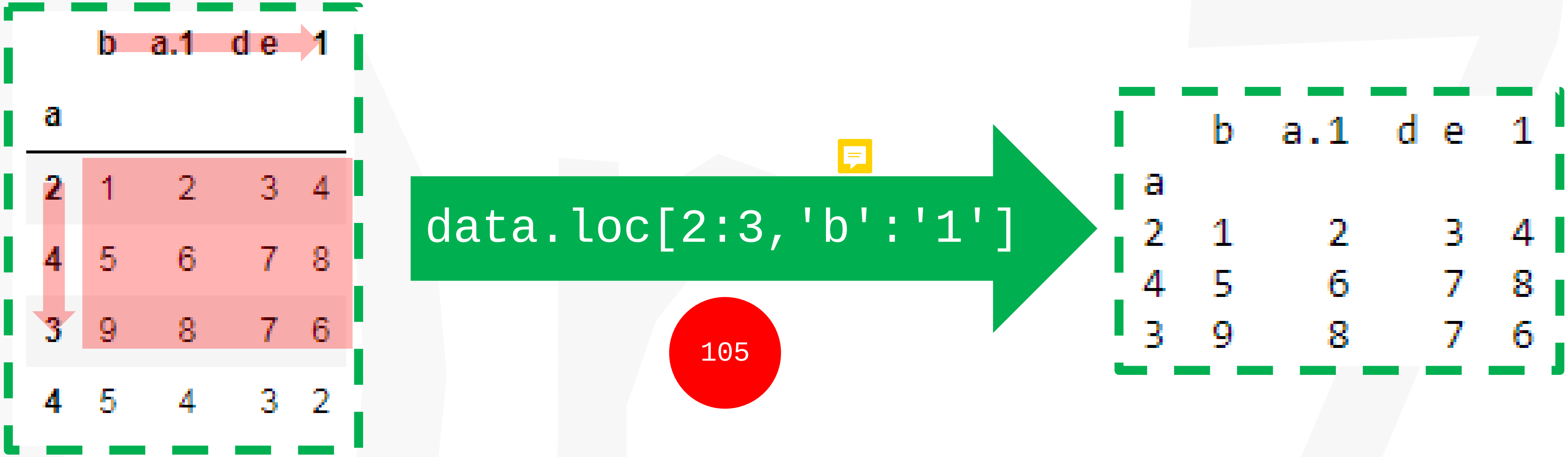
SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

(continued)

label-based slicing (1)

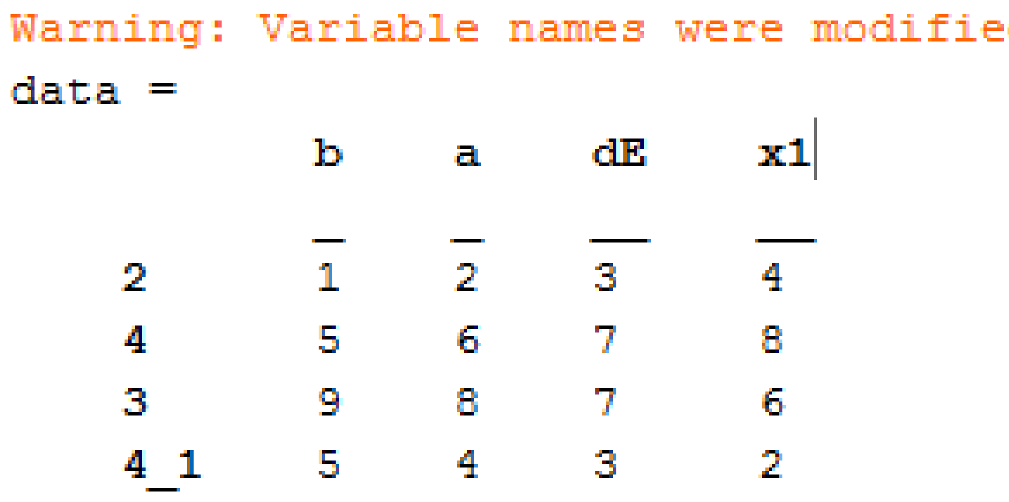


```
   b  a.1  d e  1
a
2  1   2   3  4
4  5   6   7  8
3  9   8   7  6
4  5   4   3  2
```

data.loc[2:3,'b':'1']

```
    b   a.1   d e   1
a
2   1     2     3   4
4   5     6     7   8
3   9     8     7   6
```

105



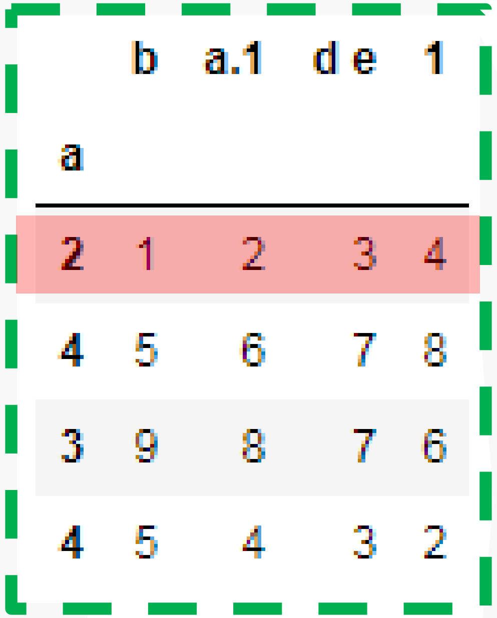
```
Warning: Variable names were modified to make them valid MATLAB identifiers.
data =

          b     a     dE     x1
          _     _     __     __

  2       1     2     3      4
  4       5     6     7      8
  3       9     8     7      6
  4_1     5     4     3      2
```

https://www.mathworks.com/help/matlab/matlab_prog/access-data-in-a-table.html

(continued)

label-based slicing (2)

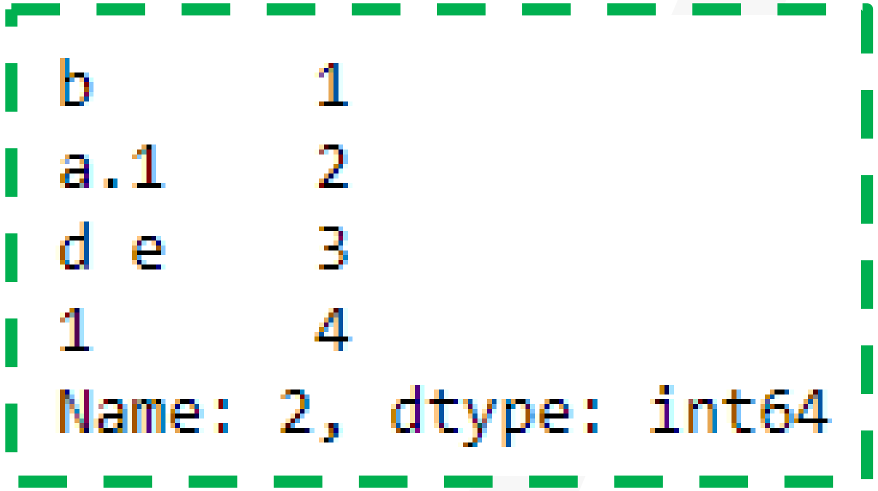
(row with label 2)



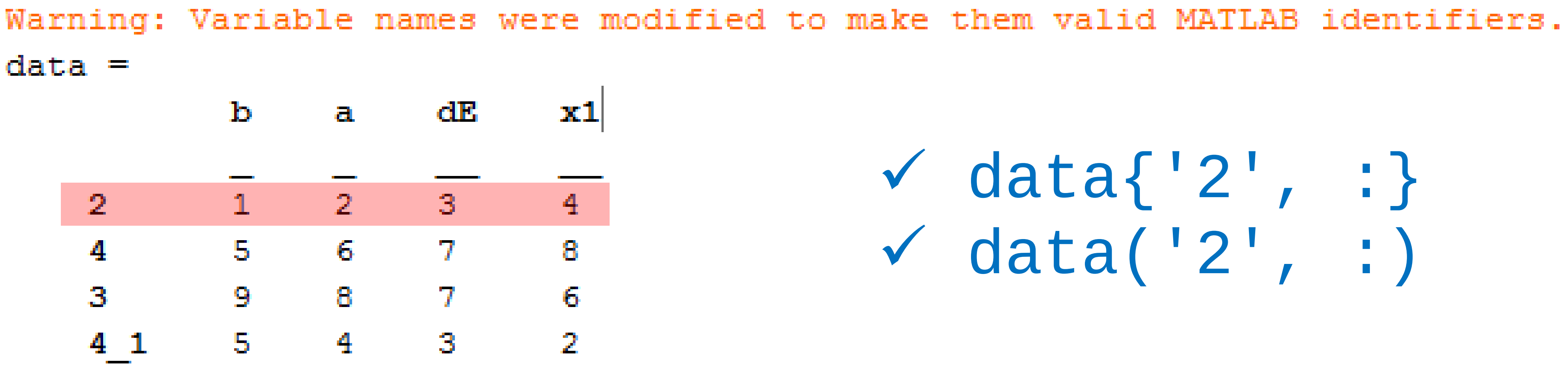
```
   b  a.1  d e  1
a
2  1   2    3   4
4  5   6    7   8
3  9   8    7   6
4  5   4    3   2
```

data.loc[2,:]

105



```
b        1
a.1      2
d e      3
1        4
Name: 2, dtype: int64
```



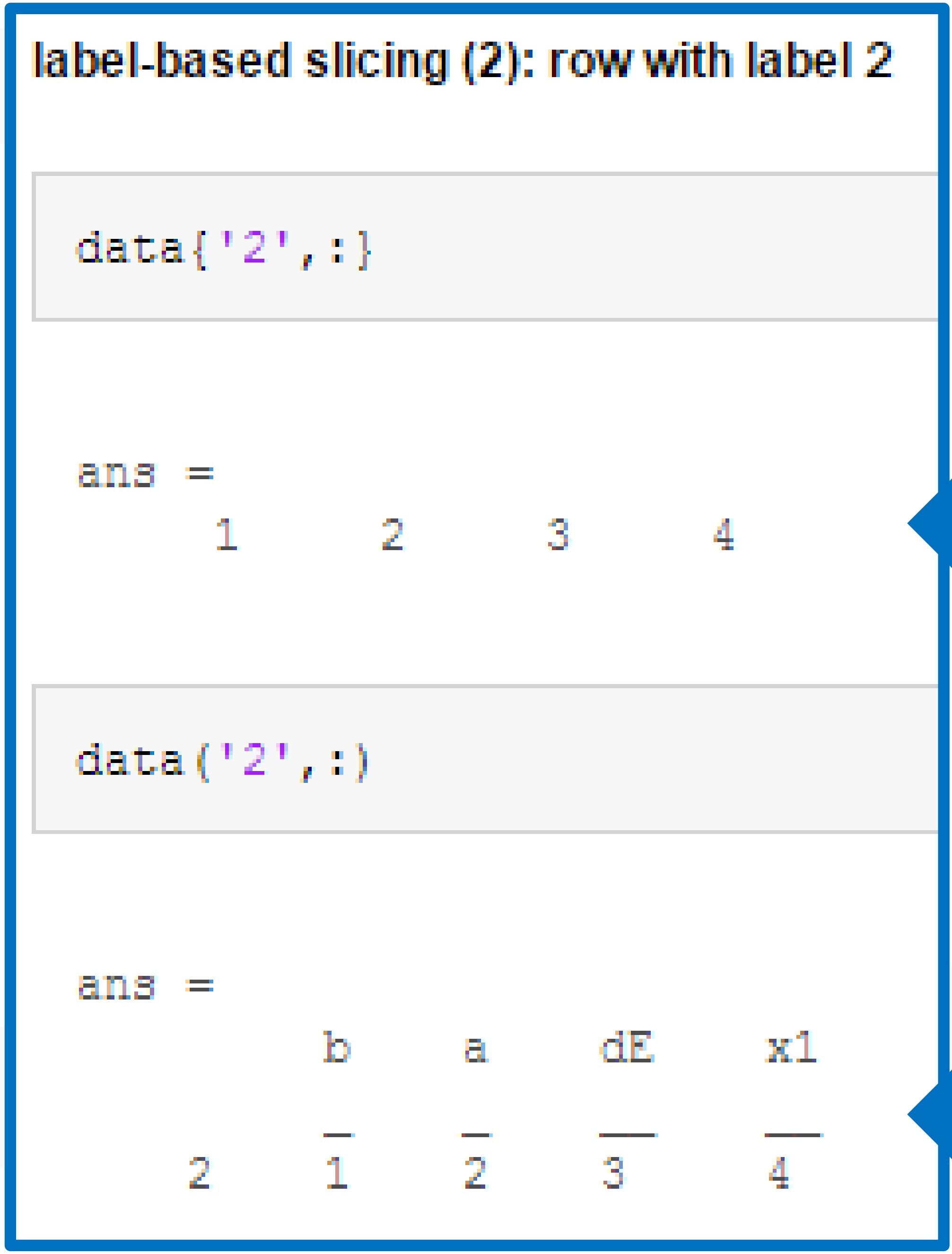
```
Warning: Variable names were modified to make them valid MATLAB identifiers.
data =
              b     a     dE     x1
              _     _     __     __
    2         1     2     3      4
    4         5     6     7      8
    3         9     8     7      6
    4_1       5     4     3      2
```

✓ data{'2', :}
✓ data('2', :)

https://www.mathworks.com/help/matlab/matlab_prog/access-data-in-a-table.html

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



label-based slicing (2): row with label 2

```
data{'2',:}
```

```
ans =
        1         2         3         4
```

```
data('2',:)
```

```
ans =
            b       a       dE       x1
            _       _       __       __
     2      1       2       3        4
```

1x4 **double**

105

1x4 **table**

SMU
SINGAPORE MANAGEMENT UNIVERSITY

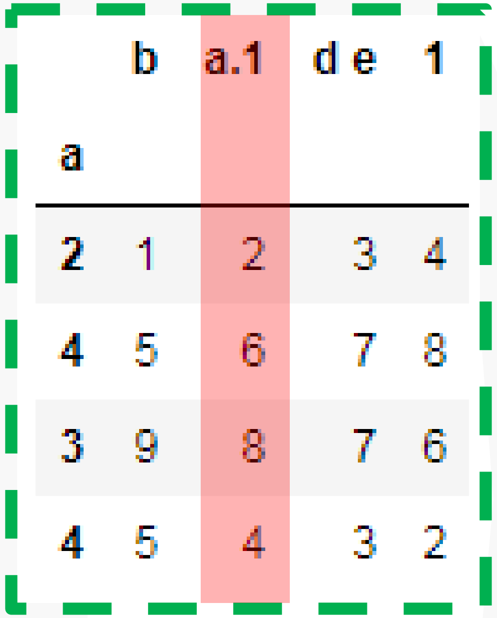
QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

(continued)

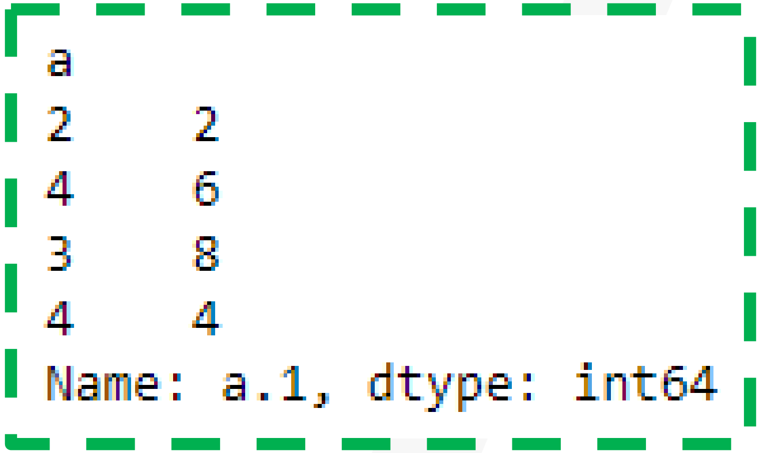
label-based slicing (3)

(column with label a.1)



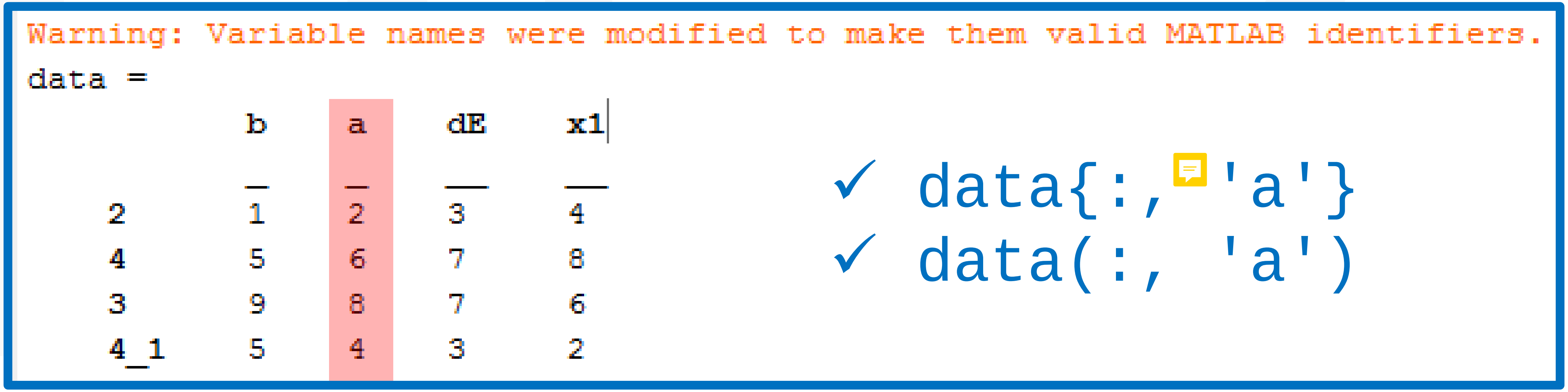
	b	a.1	d e	1
a				
2	1	2	3	4
4	5	6	7	8
3	9	8	7	6
4	5	4	3	2

`data.loc[:, 'a.1']`



```
a
2     2
4     6
3     8
4     4
Name: a.1, dtype: int64
```

105



```
Warning: Variable names were modified to make them valid MATLAB identifiers.
data =

                b     a     dE     x1
                _     _     __     __
    2           1     2     3      4
    4           5     6     7      8
    3           9     8     7      6
    4_1         5     4     3      2
```

✓ `data{:,'a'}`

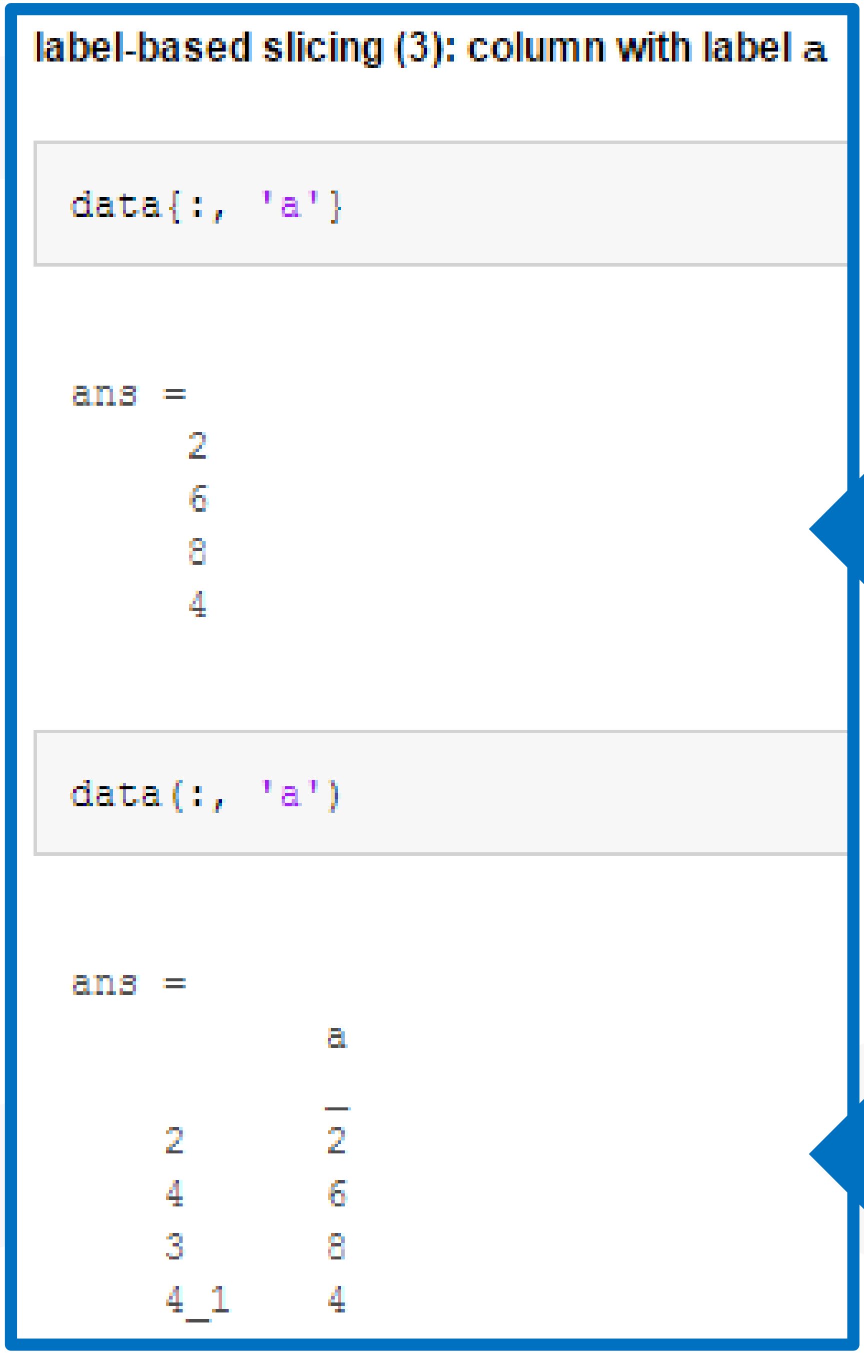
✓ `data(:, 'a')`

https://www.mathworks.com/help/matlab/matlab_prog/access-data-in-a-table.html

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



label-based slicing (3): column with label a

```
data{:, 'a'}
```

```
ans =
      2
      6
      8
      4
```

```
data(:, 'a')
```

```
ans =
                 a
                 _
    2            2
    4            6
    3            8
    4_1          4
```

4x1 **double**

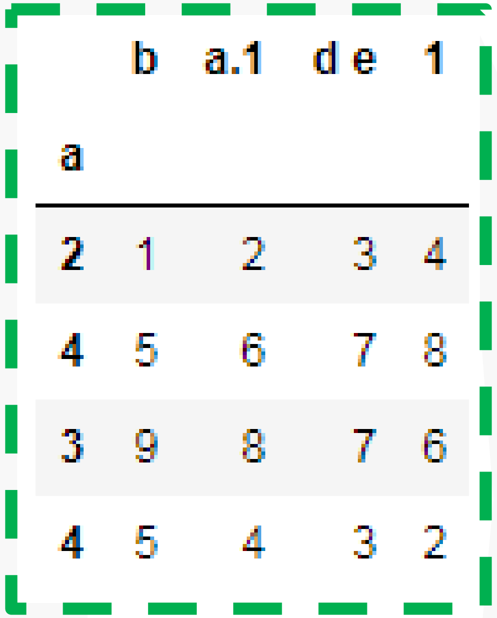
4x1 **table**

105

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

(continued)

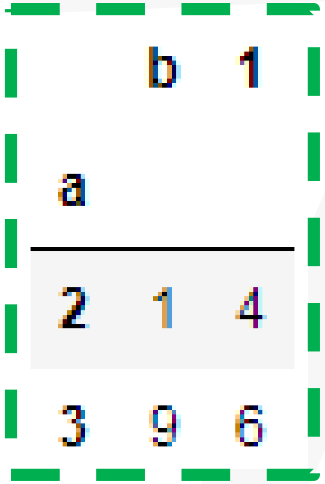
label-based fancy indexing



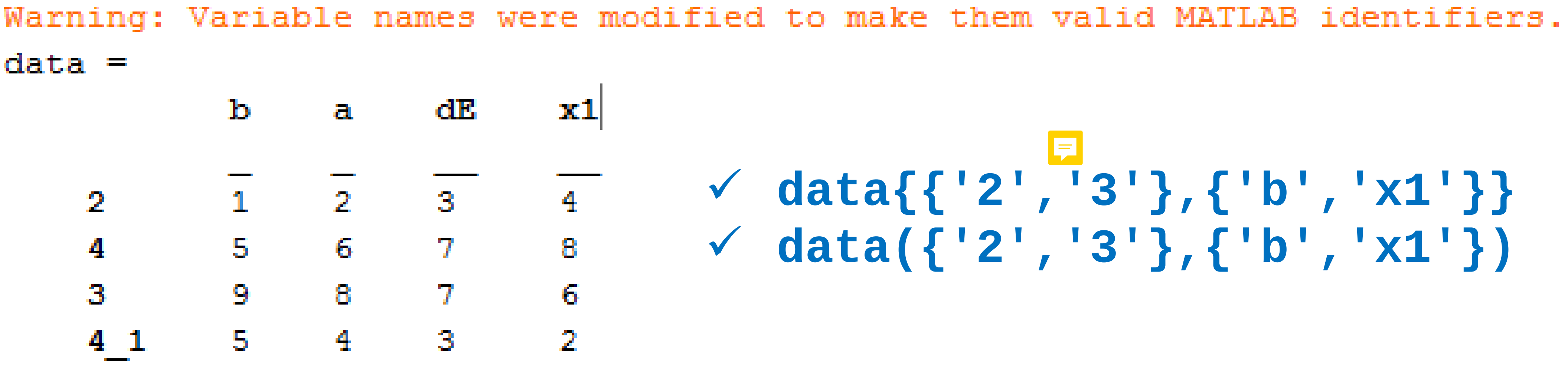
a	b	a.1	d e	1
2	1	2	3	4
4	5	6	7	8
3	9	8	7	6
4	5	4	3	2

`data.loc[[2,3],['b','1']]`

105



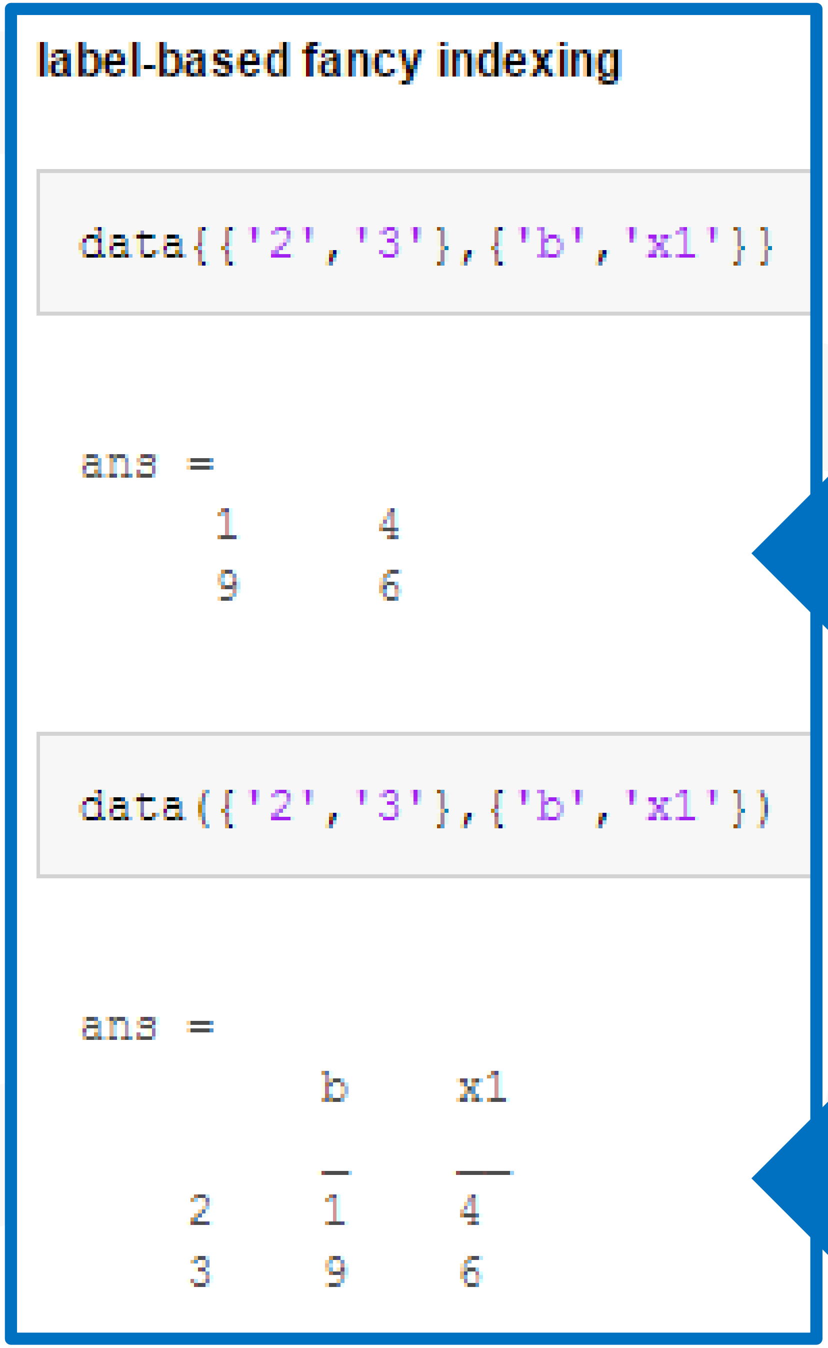
a	b	1
2	1	4
3	9	6



```
Warning: Variable names were modified to make them valid MATLAB identifiers.
data =
                 b        a        dE        x1
                 _        _        __        __
     2           1        2        3         4
     4           5        6        7         8
     3           9        8        7         6
     4_1         5        4        3         2
```

✓ `data{{'2','3'},{'b','x1'}}`
✓ `data({'2','3'},{'b','x1'})`

https://www.mathworks.com/help/matlab/matlab_prog/access-data-in-a-table.html



label-based fancy indexing

```
data{{'2','3'},{'b','x1'}}
```

```
ans =

     1     4
     9     6
```

```
data({'2','3'},{'b','x1'})
```

```
ans =

          b    x1
          _    __

    2     1    4
    3     9    6
```

2x2 **double**

105

2x2 **table**

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

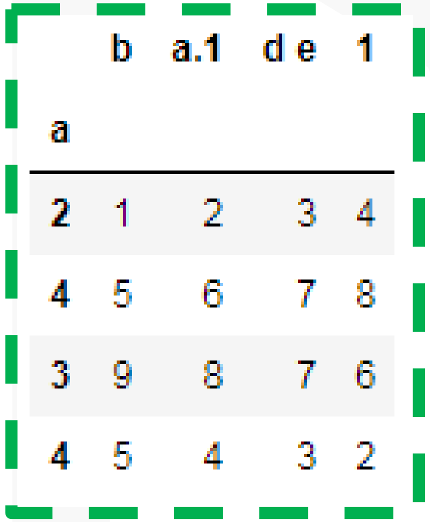
SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance

HT004: Use one command to add a column to **data** with a specific name.

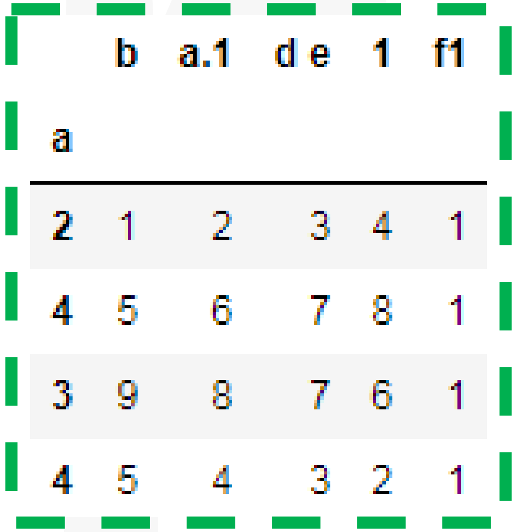
1. a number
2. an iterable
3. A Numpy array
4. a Series
5. a DataFrame



	b	a.1	d e	1
a				
2	1	2	3	4
4	5	6	7	8
3	9	8	7	6
4	5	4	3	2

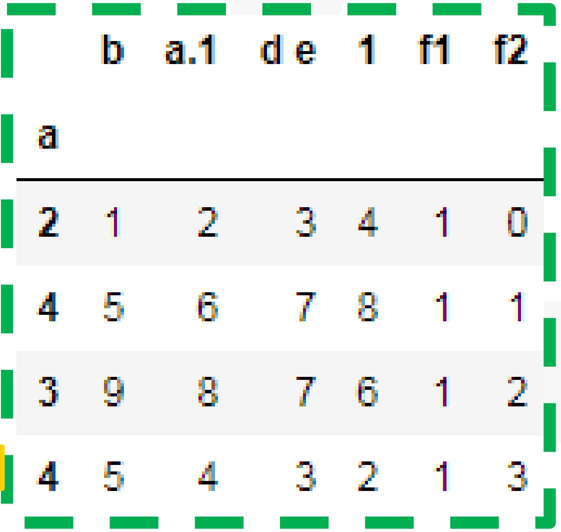
```
✓ data['f1']=1
✓ data.loc[:,'f1']=1
```

106



	b	a.1	d e	1	f1
a					
2	1	2	3	4	1
4	5	6	7	8	1
3	9	8	7	6	1
4	5	4	3	2	1

```
✓   data['f2']=range(4)
✓ data.loc[:,'f2']=range(4)
✓                 …=[0,1,2,3]
✓         …=[[0],[1],[2],[3]]
×              …=[[0,1,2,3]]
```



	b	a.1	d e	1	f1	f2
a						
2	1	2	3	4	1	0
4	5	6	7	8	1	1
3	9	8	7	6	1	2
4	5	4	3	2	1	3

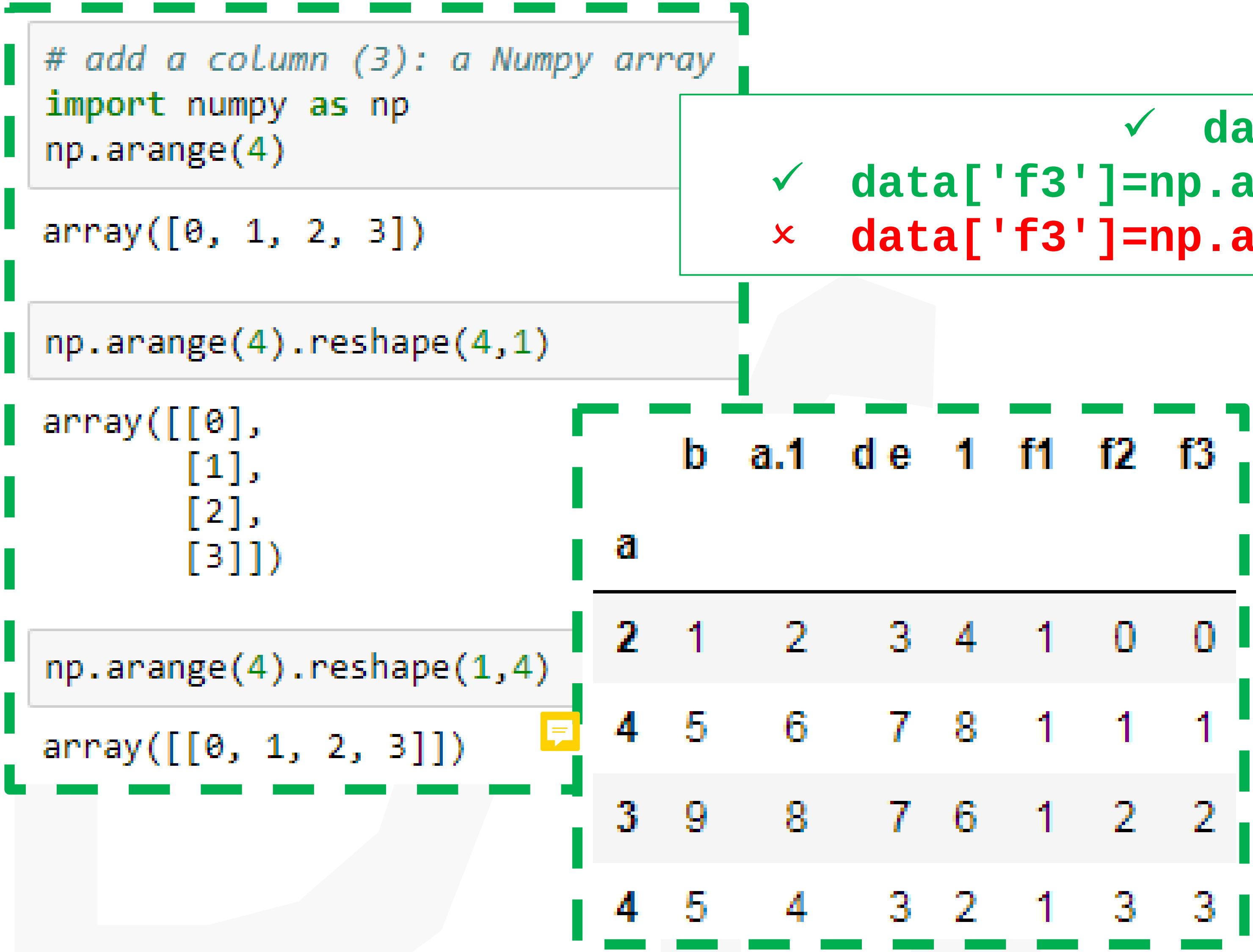
107

Slide
197/307

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance



```
# add a column (3): a Numpy array
import numpy as np
np.arange(4)
```

```
array([0, 1, 2, 3])
```

```
np.arange(4).reshape(4,1)
```

```
array([[0],
       [1],
       [2],
       [3]])
```

```
np.arange(4).reshape(1,4)
```

```
array([[0, 1, 2, 3]])
```

	b	a.1	d e	1	f1	f2	f3
a							
2	1	2	3	4	1	0	0
4	5	6	7	8	1	1	1
3	9	8	7	6	1	2	2
4	5	4	3	2	1	3	3

✓ `data['f3']=np.arange(4)`

✓ `data['f3']=np.arange(4).reshape(4,1)`

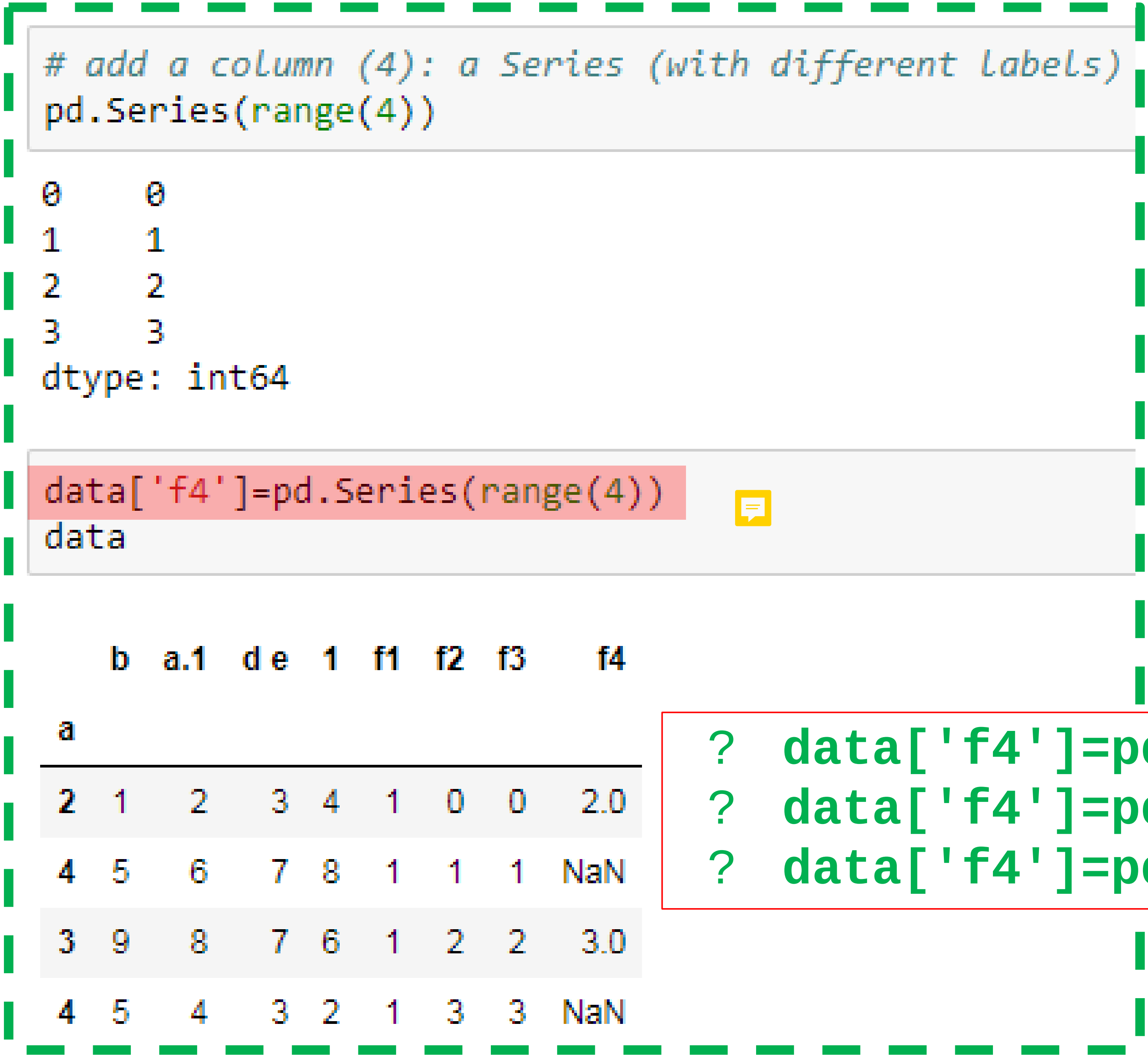
× `data['f3']=np.arange(4).reshape(1,4)`

1D array or 2D column array

1. a number
2. an iterable
3. A Numpy array
4. a Series
5. a DataFrame

108

Slide
198/307



```
# add a column (4): a Series (with different labels)
pd.Series(range(4))
```

```
0    0
1    1
2    2
3    3
dtype: int64
```

```
data['f4']=pd.Series(range(4))
data
```

	b	a.1	d	e	1	f1	f2	f3	f4
a									
2	1	2		3	4	1	0	0	2.0
4	5	6		7	8	1	1	1	NaN
3	9	8		7	6	1	2	2	3.0
4	5	4		3	2	1	3	3	NaN

1. a number
2. an iterable
3. A Numpy array
4. a Series
5. a DataFrame

109

```
? data['f4']=pd.Series(range(3))
? data['f4']=pd.Series(range(4))
? data['f4']=pd.Series(range(5))
```

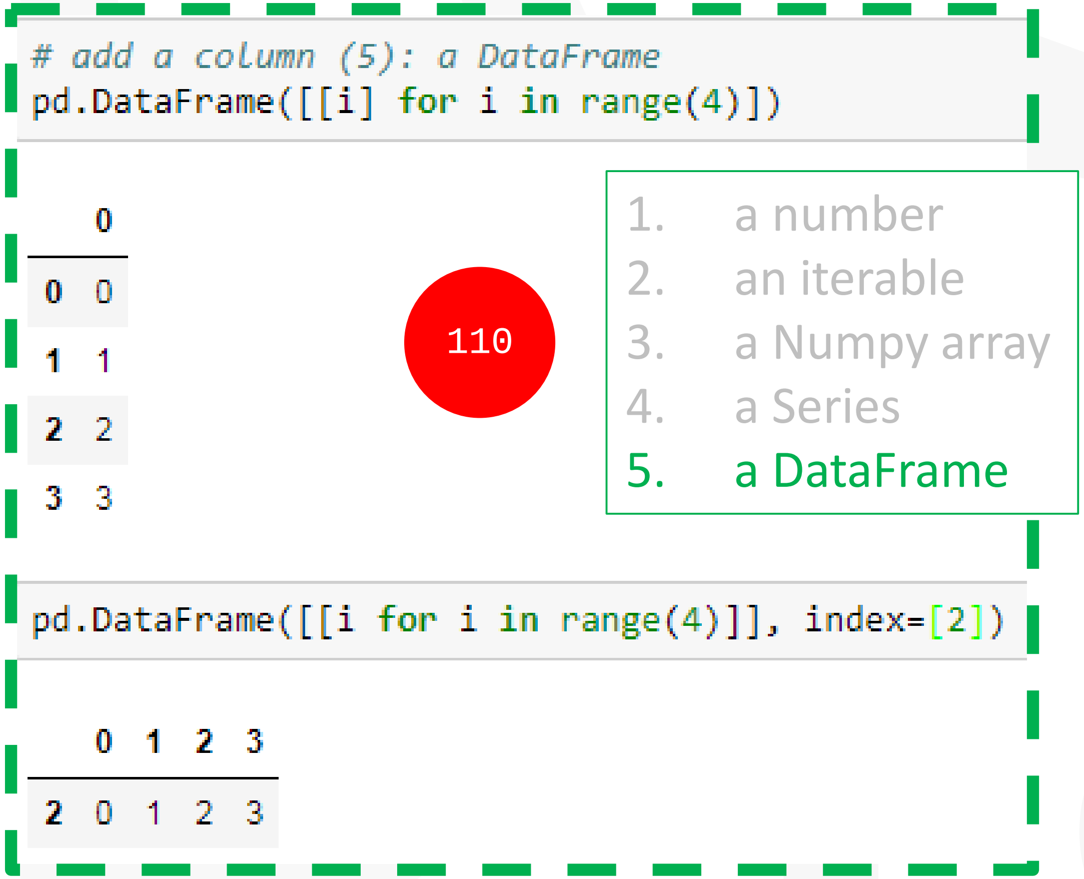
SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

Slide
199/307

```
? data['f5']=pd.DataFrame([[i] for i in range(3)])
? data['f5']=pd.DataFrame([[i] for i in range(4)])
? data['f5']=pd.DataFrame([[i] for i in range(5)])
```



```
# add a column (5): a DataFrame
pd.DataFrame([[i] for i in range(4)])
```

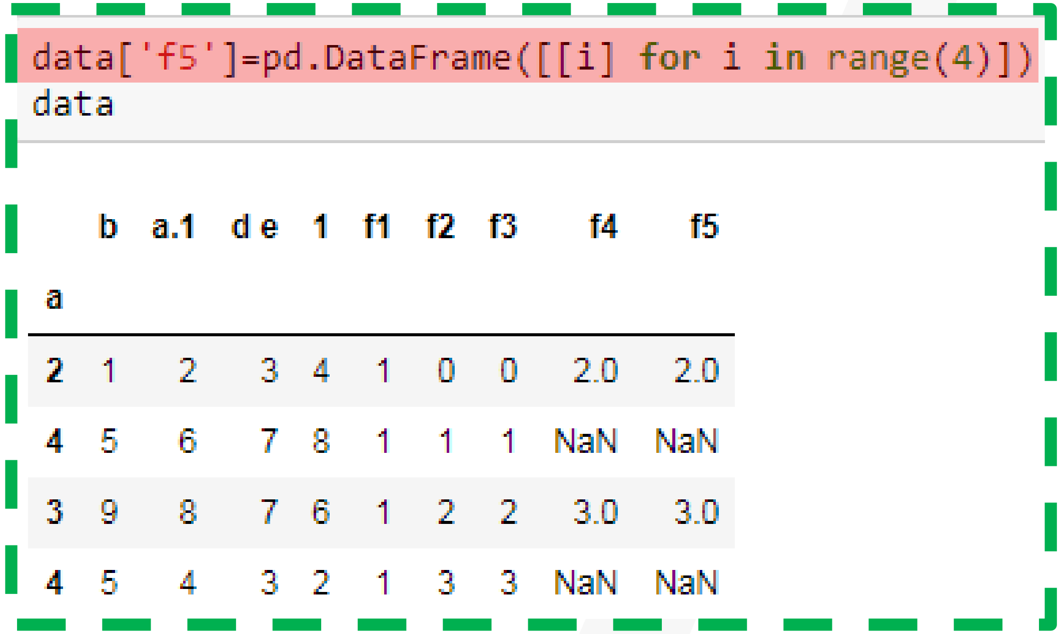
```
   0
0  0
1  1
2  2
3  3
```

```
pd.DataFrame([[i for i in range(4)]], index=[2])
```

```
   0  1  2  3
2  0  1  2  3
```

110

1. a number
2. an iterable
3. a Numpy array
4. a Series
5. a DataFrame



```
data['f5']=pd.DataFrame([[i] for i in range(4)])
data
```

```
   b  a.1  d  e  1  f1  f2  f3   f4   f5
a
2  1   2   3  4  1   0   0  2.0  2.0
4  5   6   7  8  1   1   1  NaN  NaN
3  9   8   7  6  1   2   2  3.0  3.0
4  5   4   3  2  1   3   3  NaN  NaN
```

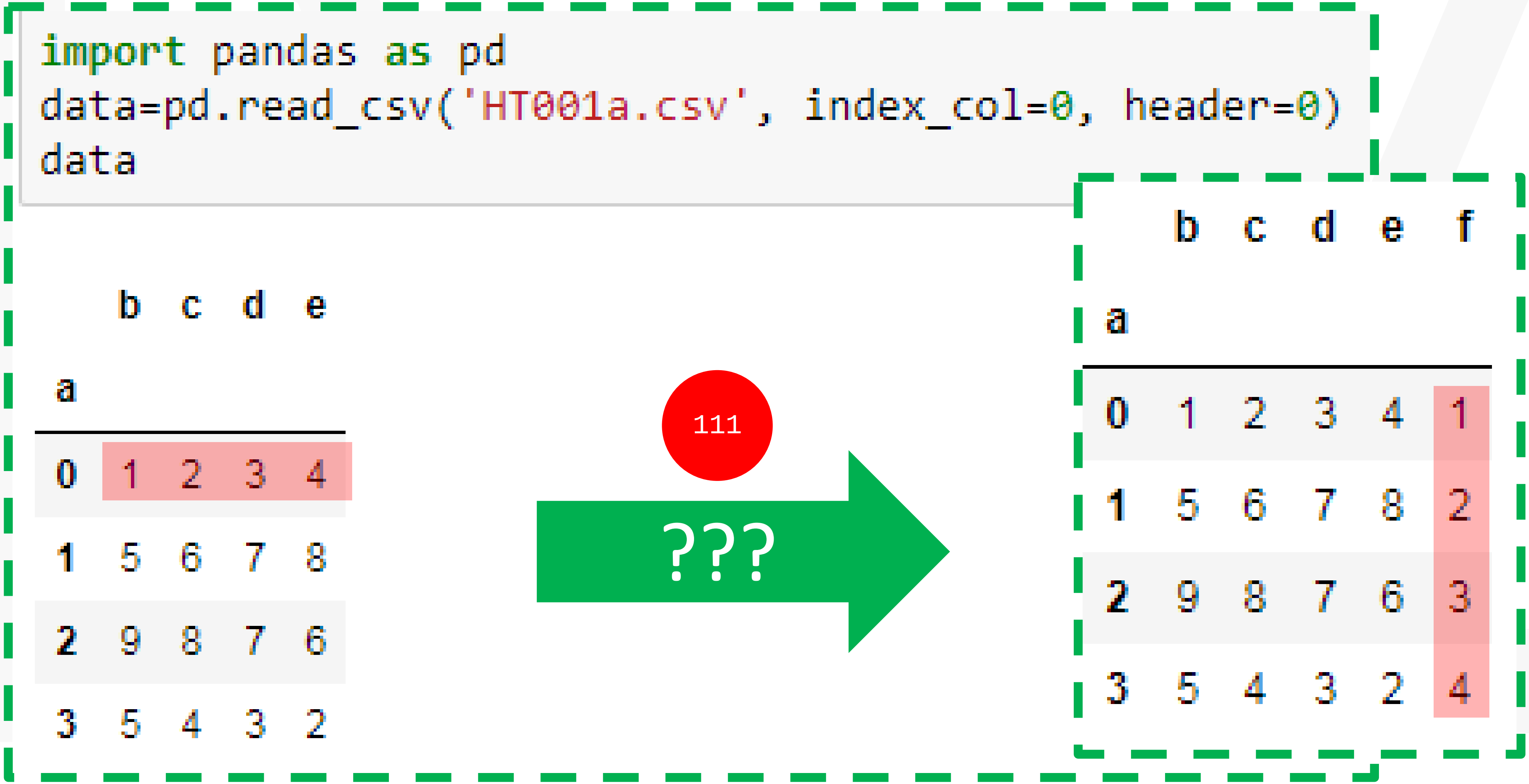
So, how to assign a row’s value to a column in a DataFrame??

```
× data['f5']=pd.DataFrame([[i for i in range(4)]],index=[2])
```

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Homework: Use one command to add a column to **data**, using the first row of **data**, and name this column **f**.



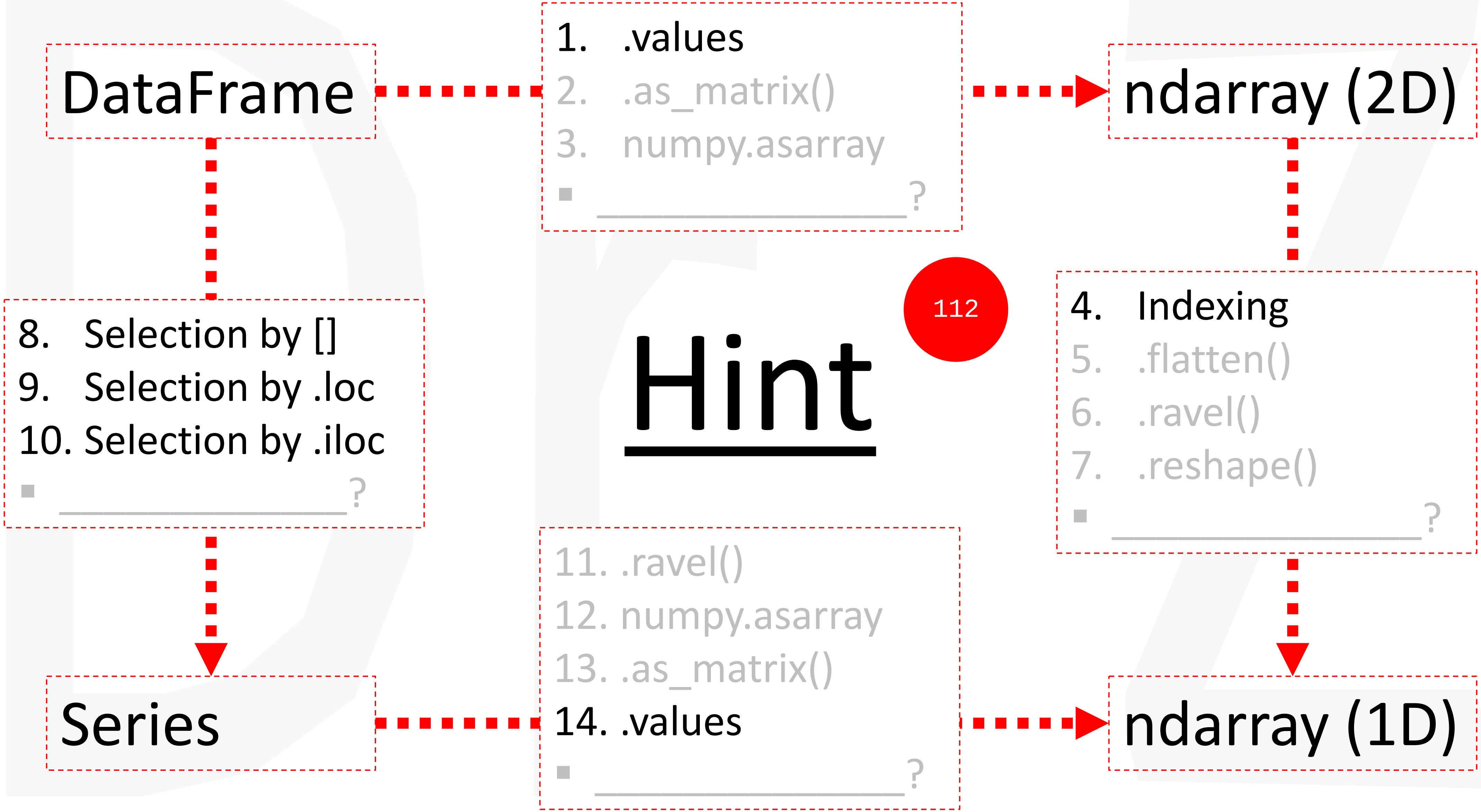
```
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
data
```

	b	c	d	e
a				
0	1	2	3	4
1	5	6	7	8
2	9	8	7	6
3	5	4	3	2

	b	c	d	e	f
a					
0	1	2	3	4	1
1	5	6	7	8	2
2	9	8	7	6	3
3	5	4	3	2	4

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



DataFrame

1. .values
2. .as_matrix()
3. numpy.asarray
▪ ________________?

ndarray (2D)

4. Indexing
5. .flatten()
6. .ravel()
7. .reshape()
▪ ________________?

8. Selection by []
9. Selection by .loc
10. Selection by .iloc
▪ ________________?

Hint

112

Series

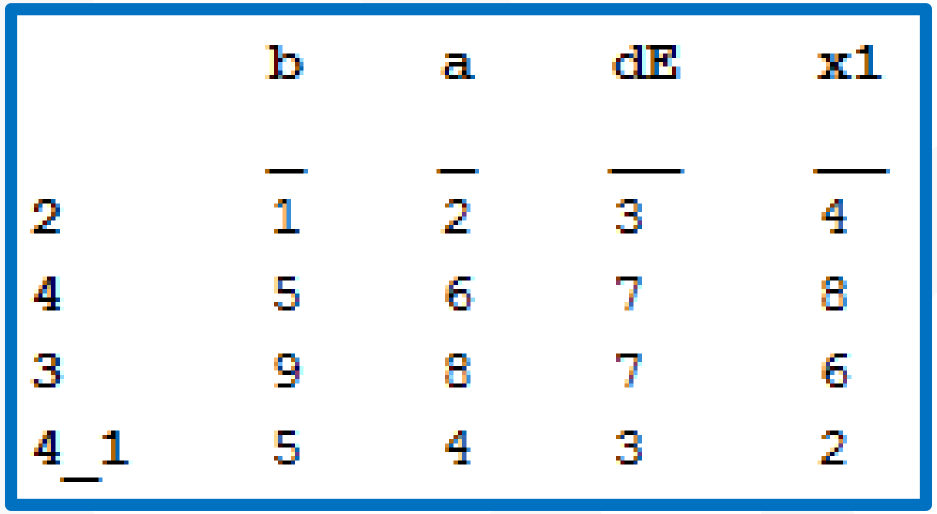
11. .ravel()
12. numpy.asarray
13. .as_matrix()
14. .values
▪ ________________?

ndarray (1D)

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

HT004: Use one command to add a column to **data** with a specific name.



```
          b     a     dE     x1
          _     _     __     __
2         1     2     3      4
4         5     6     7      8
3         9     8     7      6
4_1       5     4     3      2
```

113

× `data.f1=1`

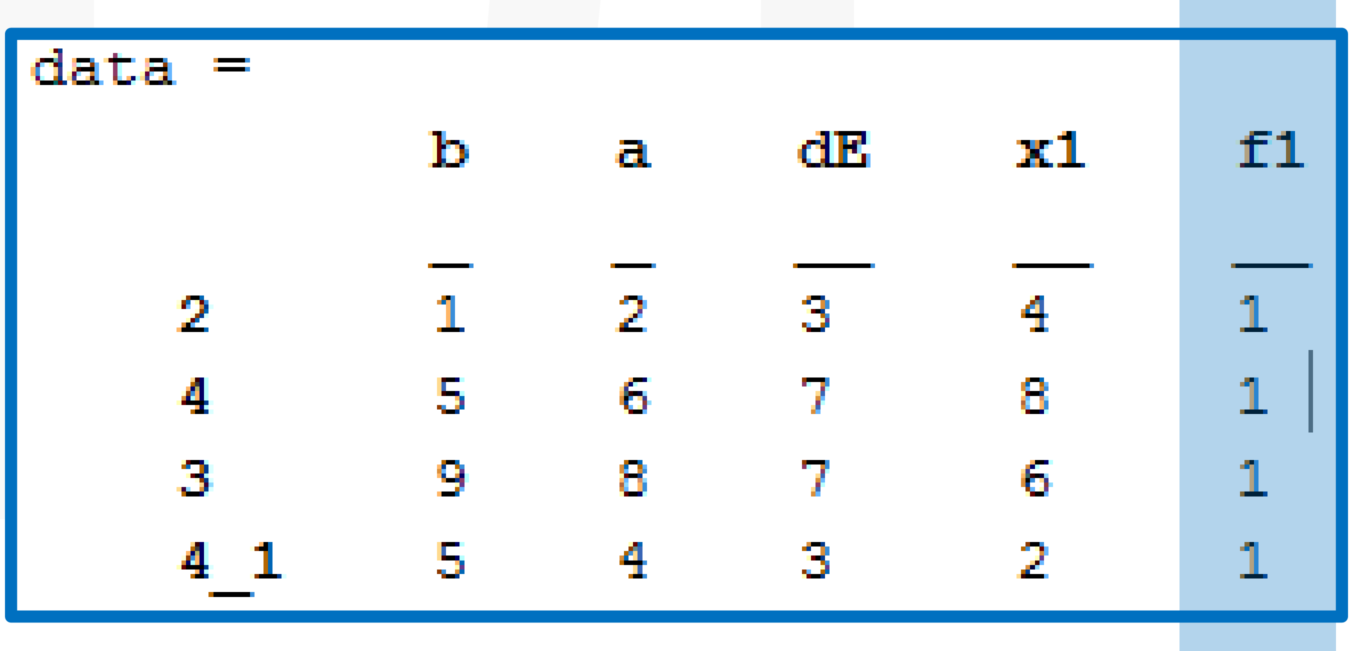
⇨ `data.f1(:)=1`

✓ `data.f1(:,1)=1`

✓ `data{:,'f1'}=1`

× `data(:,'f1')=1`

✓ `data(:,'f1')={1}`



```
data =

               b     a     dE     x1     f1
               _     _     __     __     __
        2      1     2     3      4      1
        4      5     6     7      8      1
        3      9     8     7      6      1
        4_1    5     4     3      2      1
```

1. a number
2. an array

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

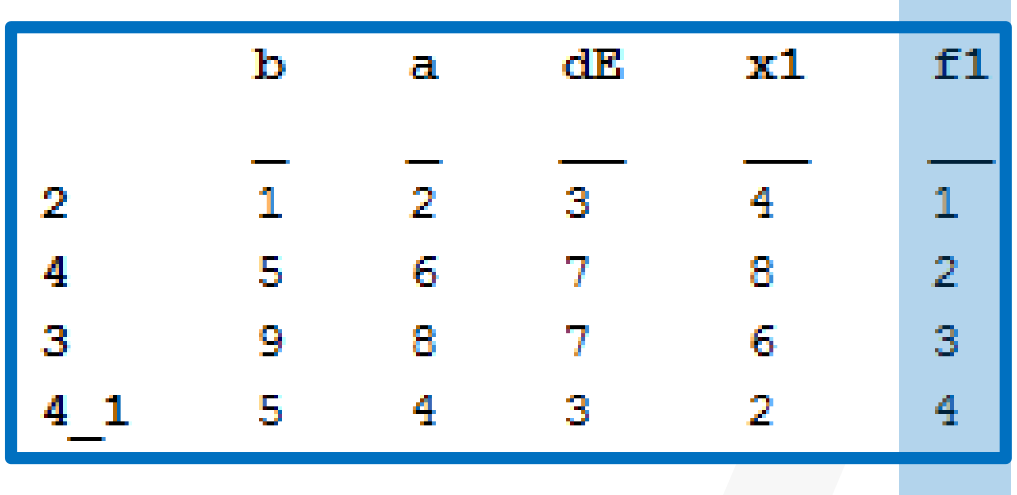
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

1. a number
2. an array

```
× data.f1=[1,2,3,4]
✓ data.f1=[1;2;3;4]
? data.f1={1;2;3;4}
× data.f1(:)=[1;2;3;4]
× data.f1(:)={1;2;3;4}
× data.f1(:)=1:4
✓ data.f1(:,1)=[1;2;3;4]
✓ data.f1(:,1)=1:4
✓ data{:,'f1'}=[1;2;3;4]
? data{:,'f1'}=1:4
```



	b	a	dE	x1	f1
2	1	2	3	4	1
4	5	6	7	8	2
3	9	8	7	6	3
4_1	5	4	3	2	4

To **assign to** or **create** a **variable** in a table, **the number of rows must match the height of the table**.

In an assignment A(:) = B, the number of elements in A and B must be the same.

114

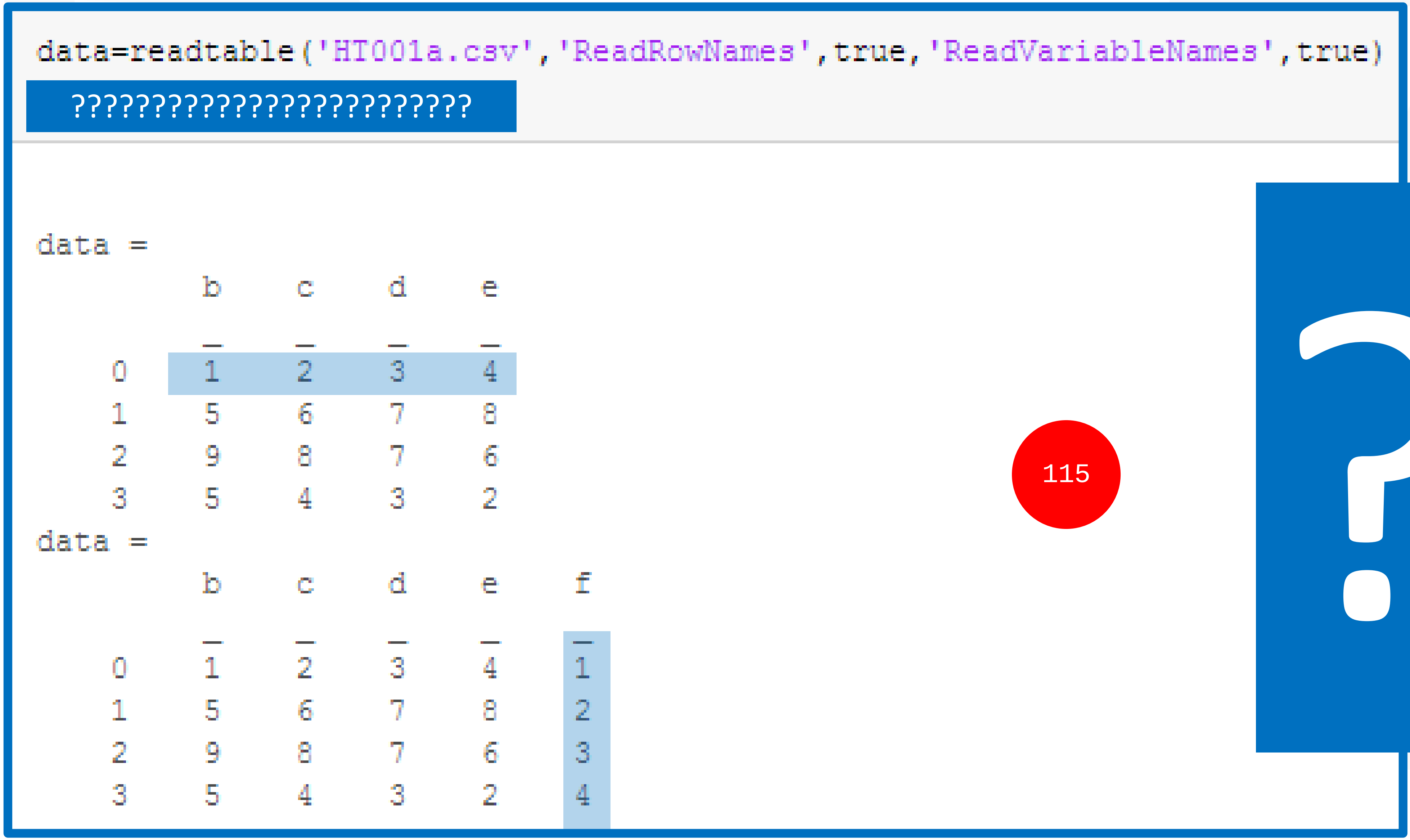
[1;2;3;4]⇔(1:4)’

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

Homework: Use one command to add a column to **data**, using the first row of **data**, and name this column **f**.



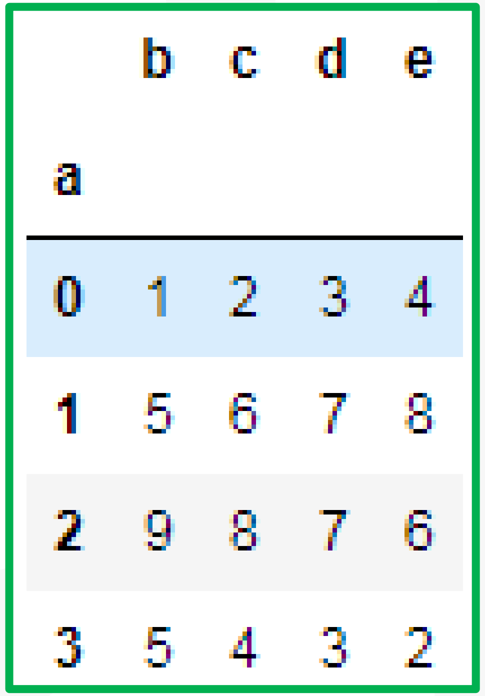
```
data=readtable('HT001a.csv','ReadRowNames',true,'ReadVariableNames',true)
????????????????????????
```

```
data =

          b    c    d    e
          _    _    _    _

     0    1    2    3    4
     1    5    6    7    8
     2    9    8    7    6
     3    5    4    3    2
data =

          b    c    d    e    f
          _    _    _    _    _

     0    1    2    3    4    1
     1    5    6    7    8    2
     2    9    8    7    6    3
     3    5    4    3    2    4
```

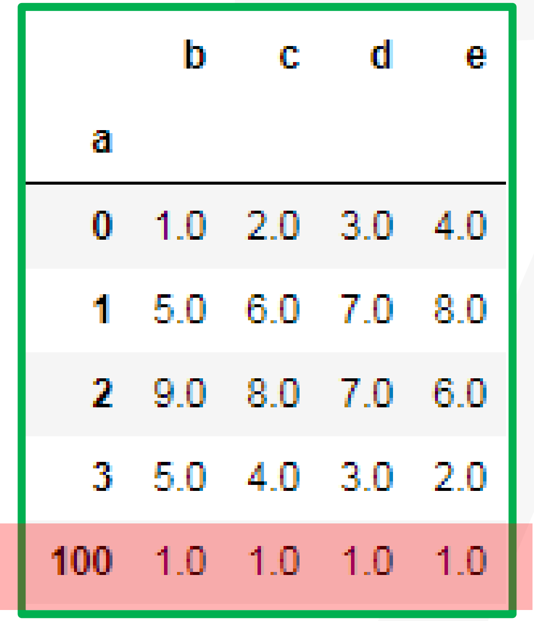
115

HT005: Use one command to add a row to **data** with a specific name.

- ✓ `data.loc[rowname,:]=…`
- ✓ `data=data.append(…)`

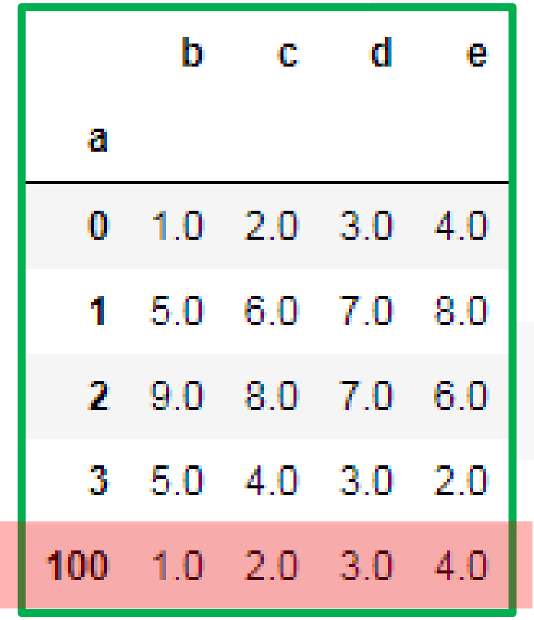


	b	c	d	e
a				
0	1	2	3	4
1	5	6	7	8
2	9	8	7	6
3	5	4	3	2



	b	c	d	e
a				
0	1.0	2.0	3.0	4.0
1	5.0	6.0	7.0	8.0
2	9.0	8.0	7.0	6.0
3	5.0	4.0	3.0	2.0
100	1.0	1.0	1.0	1.0

116



	b	c	d	e
a				
0	1.0	2.0	3.0	4.0
1	5.0	6.0	7.0	8.0
2	9.0	8.0	7.0	6.0
3	5.0	4.0	3.0	2.0
100	1.0	2.0	3.0	4.0

117

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

✓ `data.loc[rowname,:]=…`
✓ `data.append(…)`

HT005: Use one command to add a row to **`data`** with a specific name.

✓ `data.loc[100,:]=1`

✓ `data.loc[100,:]=[1]`

× `data.loc[100,:]=[[1]]`

✓ `data.loc[100,:]=np.array(1)`

✓ `data.loc[100,:]=np.array([1])`

✓ `data.loc[100,:]=np.array([[1]])`

116

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

- ✓ `data.loc[rowname,:]=…`
- ✓ `data.append(…)`

HT005: Use one command to add a row to **data** with a specific name.

- ✓ `data.loc[100,:]=range(1,5)`
- ✓ `data.loc[100,:]=[1, 2, 3, 4]`
- × `data.loc[100,:]=[[1, 2, 3, 4]]`
- × `data.loc[100,:]=[[1],[2],[3],[4]]`
- ✓ `data.loc[100,:]=np.array([1,2,3,4])`
- ✓ `data.loc[100,:]=np.array([[1,2,3,4]])`
- × `data.loc[100,:]=np.array([[1],[2],[3],[4]])`

117

Dr. Z: 1D for 1D (100%).

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

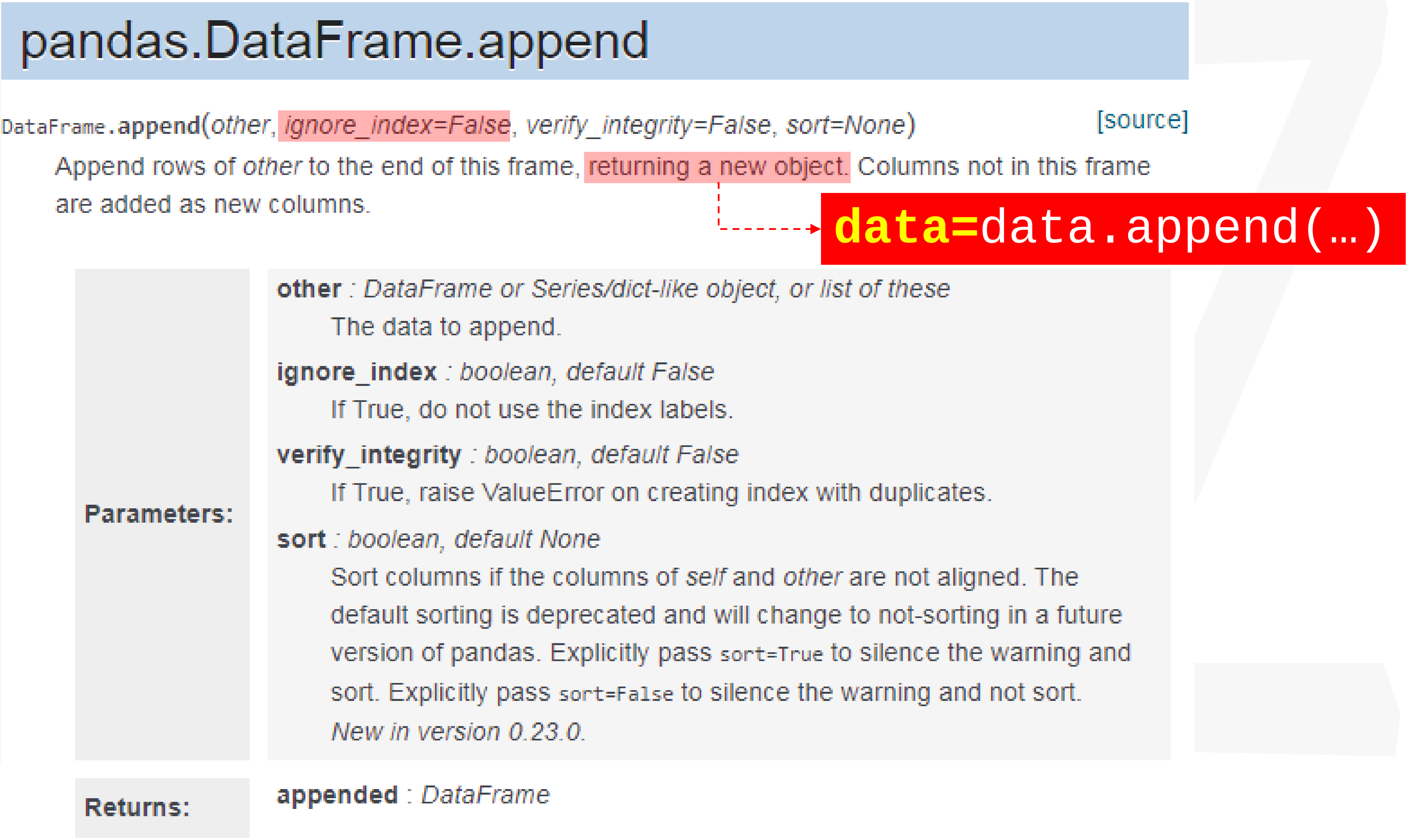
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.append.html



pandas.DataFrame.append

`DataFrame.append`(*other*, *ignore_index=False*, *verify_integrity=False*, *sort=None*) [source]

Append rows of *other* to the end of this frame, returning a new object. Columns not in this frame are added as new columns.

`data=data.append(…)`

Parameters:

other : *DataFrame or Series/dict-like object, or list of these*

The data to append.

ignore_index : *boolean, default False*

If True, do not use the index labels.

verify_integrity : *boolean, default False*

If True, raise ValueError on creating index with duplicates.

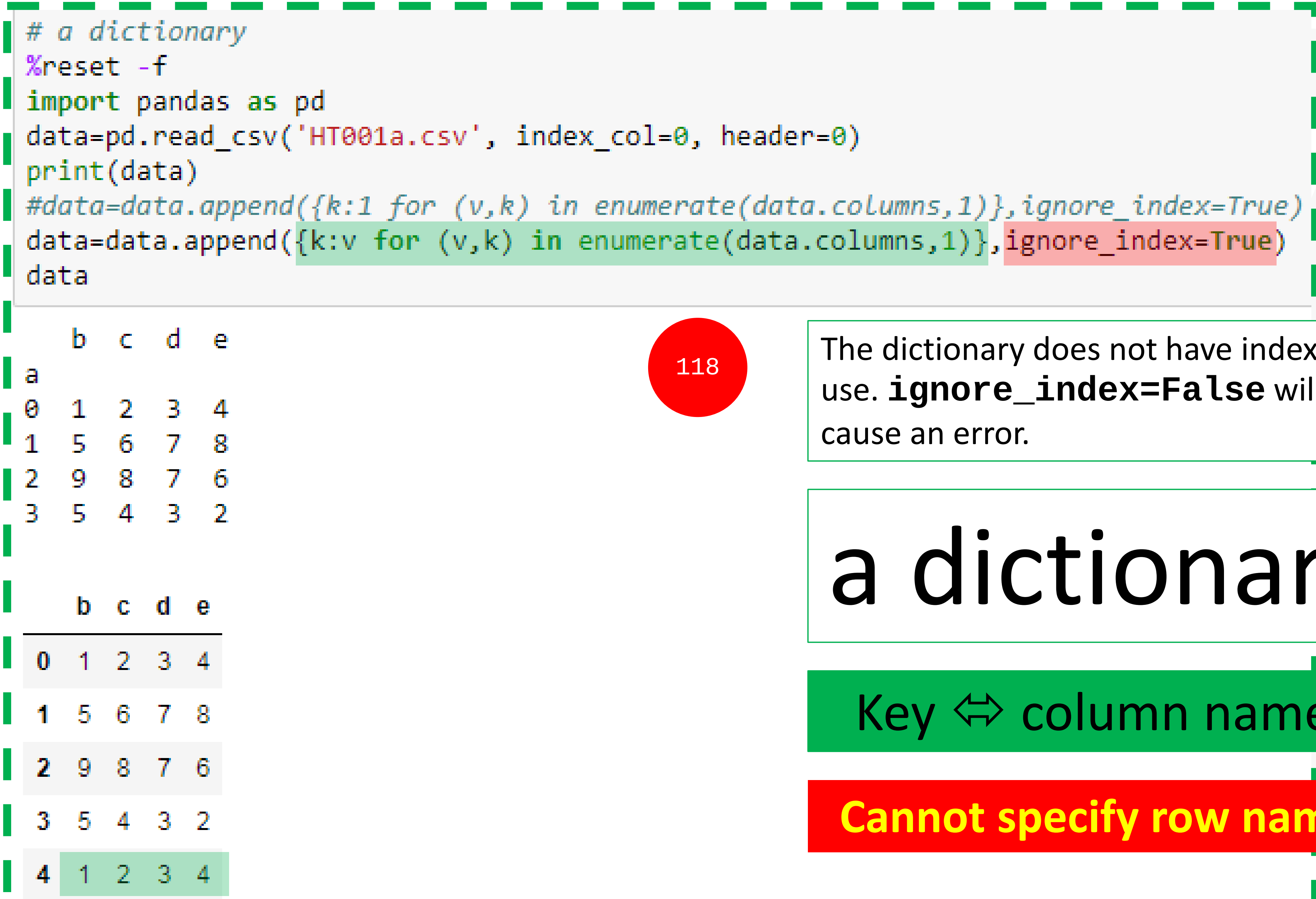
sort : *boolean, default None*

Sort columns if the columns of *self* and *other* are not aligned. The default sorting is deprecated and will change to not-sorting in a future version of pandas. Explicitly pass `sort=True` to silence the warning and sort. Explicitly pass `sort=False` to silence the warning and not sort.

New in version 0.23.0.

Returns:

appended : *DataFrame*



```
# a dictionary
%reset -f
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
#data=data.append({k:1 for (v,k) in enumerate(data.columns,1)},ignore_index=True)
data=data.append({k:v for (v,k) in enumerate(data.columns,1)},ignore_index=True)
data
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
```

	b	c	d	e
0	1	2	3	4
1	5	6	7	8
2	9	8	7	6
3	5	4	3	2
4	1	2	3	4

118

The dictionary does not have index to use. **`ignore_index=False`** will cause an error.

a dictionary

Key ⇔ column names

Cannot specify row name.

SMU SINGAPORE MANAGEMENT UNIVERSITY

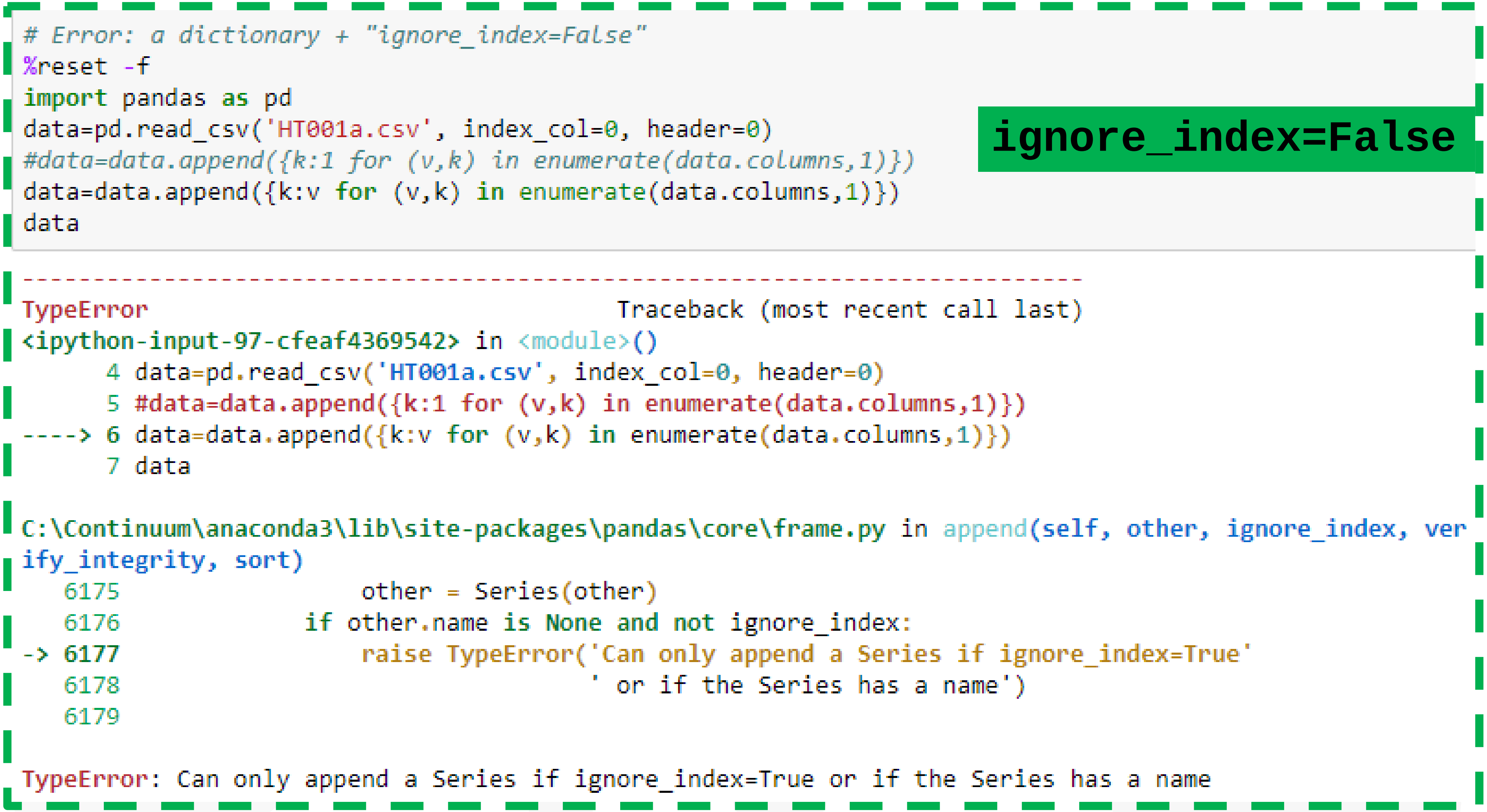
QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



ignore_index=False

```
# Error: a dictionary + "ignore_index=False"
%reset -f
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
#data=data.append({k:1 for (v,k) in enumerate(data.columns,1)})
data=data.append({k:v for (v,k) in enumerate(data.columns,1)})
data
```

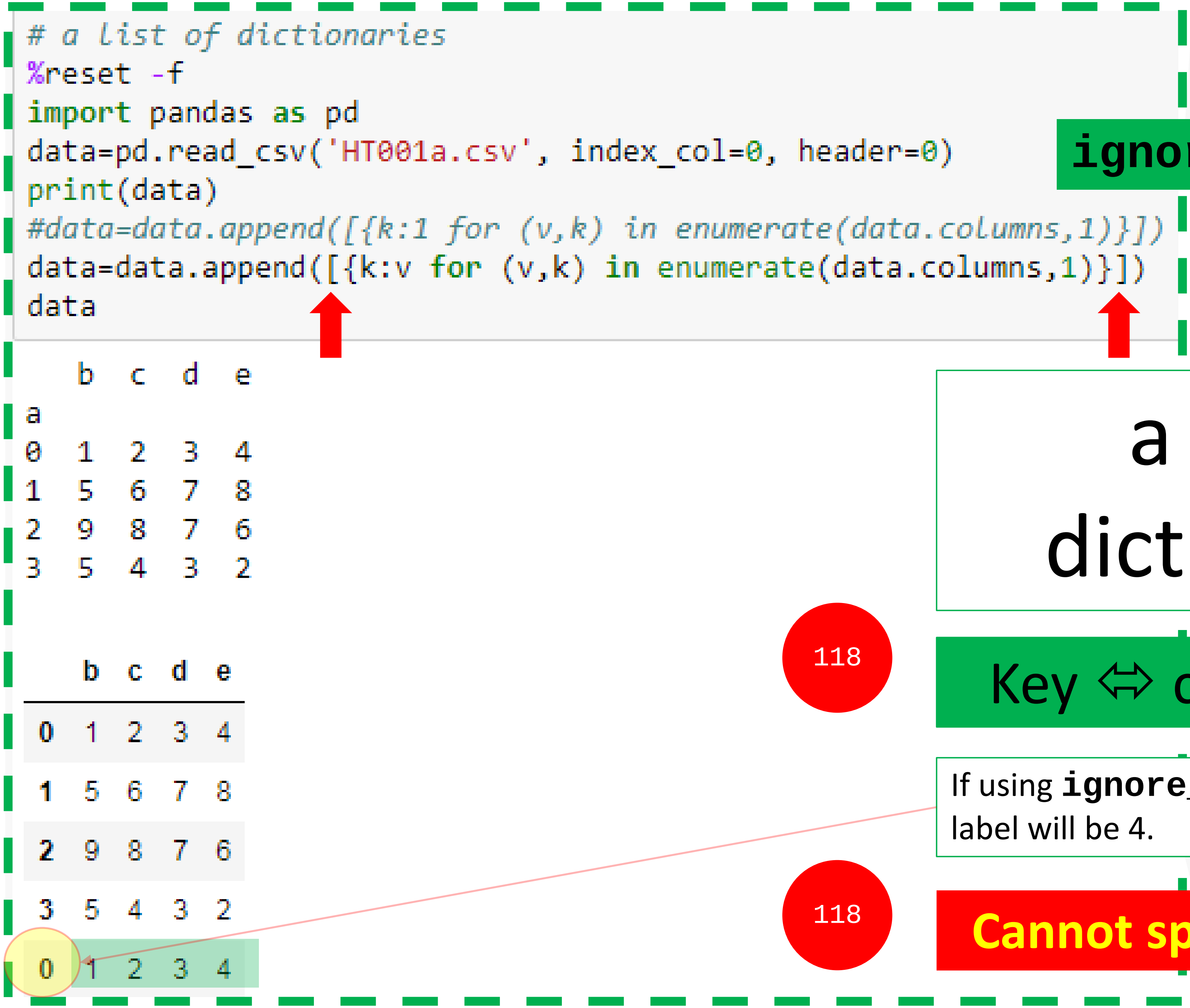
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-97-cfeaf4369542> in <module>()
      4 data=pd.read_csv('HT001a.csv', index_col=0, header=0)
      5 #data=data.append({k:1 for (v,k) in enumerate(data.columns,1)})
----> 6 data=data.append({k:v for (v,k) in enumerate(data.columns,1)})
      7 data

C:\Continuum\anaconda3\lib\site-packages\pandas\core\frame.py in append(self, other, ignore_index, ver
ify_integrity, sort)
   6175                 other = Series(other)
   6176             if other.name is None and not ignore_index:
-> 6177                 raise TypeError('Can only append a Series if ignore_index=True'
   6178                                 ' or if the Series has a name')
   6179

TypeError: Can only append a Series if ignore_index=True or if the Series has a name
```

(Dr. Z: The dictionary does not suggest any name.)

Slide
211/307

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance



```
# a list of dictionaries
%reset -f
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
#data=data.append([{k:1 for (v,k) in enumerate(data.columns,1)}])
data=data.append([{k:v for (v,k) in enumerate(data.columns,1)}])
data
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
```

```
   b  c  d  e
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
0  1  2  3  4
```

ignore_index=False

a list of dictionaries

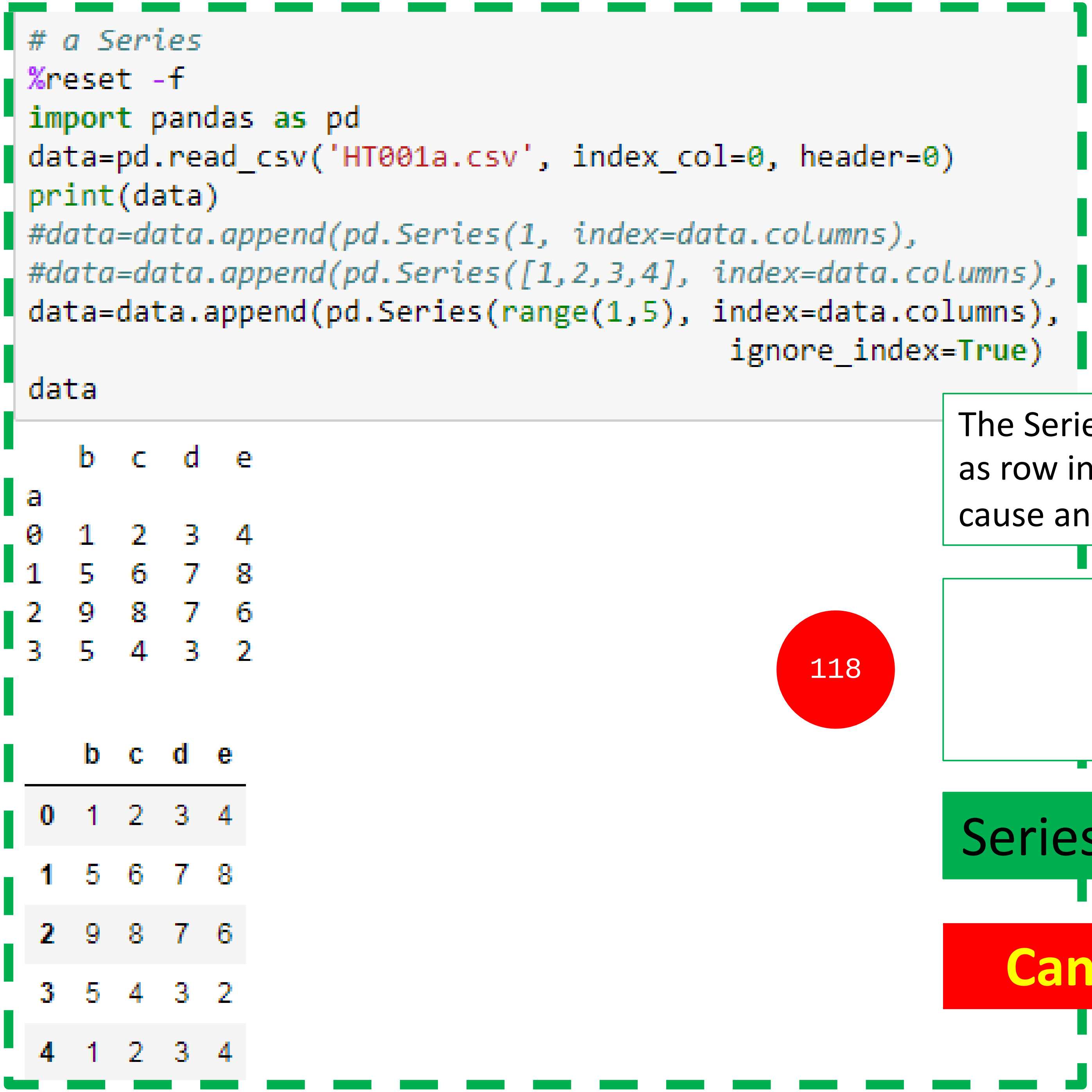
118

Key ⇔ column names

If using **ignore_index=True**, the row label will be 4.

118

Cannot specify row name.



```python
# a Series
%reset -f
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
#data=data.append(pd.Series(1, index=data.columns),
#data=data.append(pd.Series([1,2,3,4], index=data.columns),
data=data.append(pd.Series(range(1,5), index=data.columns),
                                        ignore_index=True)
data
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
```

	b	c	d	e
0	1	2	3	4
1	5	6	7	8
2	9	8	7	6
3	5	4	3	2
4	1	2	3	4

118

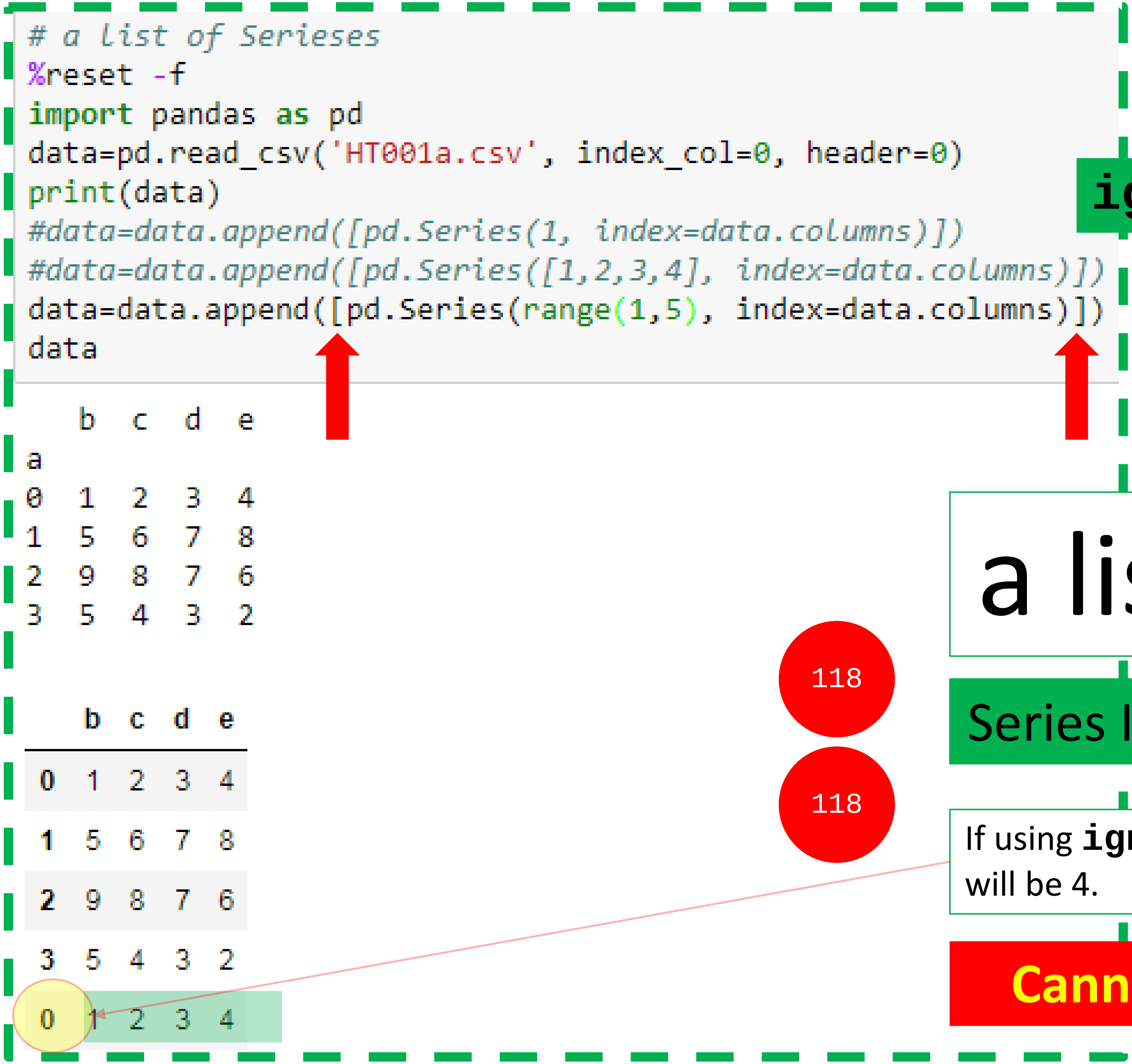
The Series does not have an index to be used as row index. **`ignore_index=False`** will cause an error.

a Series

Series Index ⇔ column names

Cannot specify row name.

Slide
213/307



```
# a list of Serieses
%reset -f
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
#data=data.append([pd.Series(1, index=data.columns)])
#data=data.append([pd.Series([1,2,3,4], index=data.columns)])
data=data.append([pd.Series(range(1,5), index=data.columns)])
data
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
```

```
   b  c  d  e
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
0  1  2  3  4
```

118

118

ignore_index=False

a list of Series

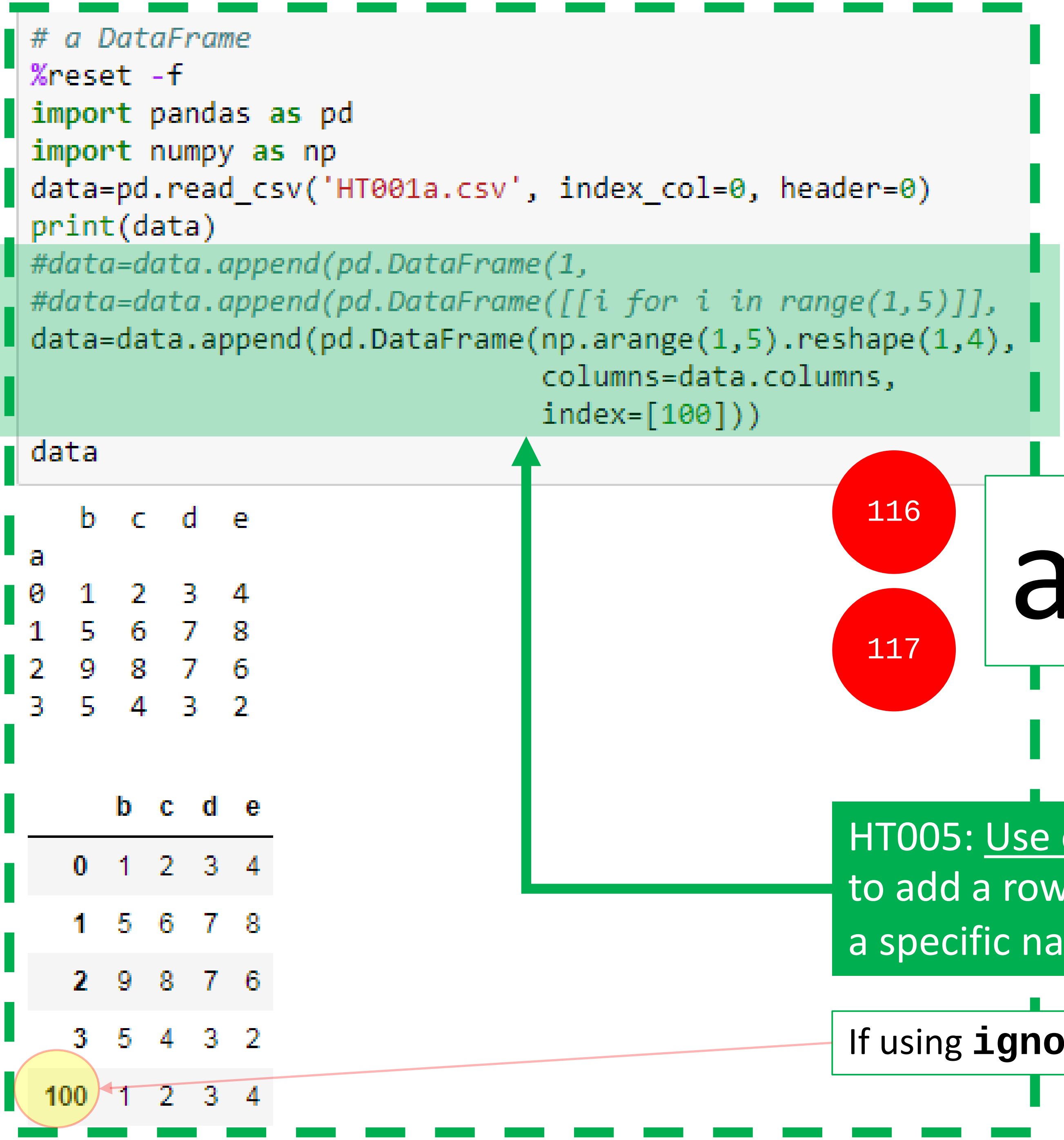
Series Index ⇔ column names

If using **ignore_index=True**, the row label will be 4.

Cannot specify row name.

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
# a DataFrame
%reset -f
import pandas as pd
import numpy as np
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
#data=data.append(pd.DataFrame(1,
#data=data.append(pd.DataFrame([[i for i in range(1,5)]],
data=data.append(pd.DataFrame(np.arange(1,5).reshape(1,4),
                              columns=data.columns,
                              index=[100]))
data
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
```

```
     b  c  d  e
  0  1  2  3  4
  1  5  6  7  8
  2  9  8  7  6
  3  5  4  3  2
100  1  2  3  4
```

116

117

ignore_index=False

index=[100]

Index or array-like

a DataFrame

HT005: Use one command to add a row to **data** with a specific name.

If using **ignore_index=True**, the row label will be 4.

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

HT005: Use one command to add a row to **data** with a specific name.

✓ Add a column

× data.f1=1

⇨ data.f1(:)=1

✓ data.f1(:,1)=1

✓ data{:,'f1'}=1

× data(:,'f1')=1

✓ data(:,'f1')={1}

× data.f1=[1,2,3,4]

✓ data.f1=[1;2;3;4]

? data.f1={1;2;3;4}

× data.f1(:)=[1;2;3;4]

× data.f1(:)={1;2;3;4}

× data.f1(:)=1:4

✓ data.f1(:,1)=[1;2;3;4]

✓ data.f1(:,1)=1:4

✓ data{:,'f1'}=[1;2;3;4]

? data{:,'f1'}=1:4

✓ Add a row

~~× data.f1=1~~

~~× data.f1(:)=1~~

~~✓ data.f1(1,:)=1~~

119

✓ data{'4',:}=1

~~× data('4',:)=1~~

✓ data('4',:)={1}

~~× data.f1=[1,2,3,4]~~

~~× data.f1=[1;2;3;4]~~

~~× data.f1={1;2;3;4}~~

~~× data.f1(:)=[1;2;3;4]~~

~~× data.f1(:)={1;2;3;4}~~

~~× data.f1(:)=1:4~~

~~× data.f1(1,:)=[1;2;3;4]~~

~~× data.f1(1,:)=1:4~~

120

✓ data{'4',:}=[1,2,3,4]

✓ data{'4',:}=1:4

HT006: Delete a row/column from **data**

1. Delete a column as deleting an item in a dictionary

```
del data['b']
```

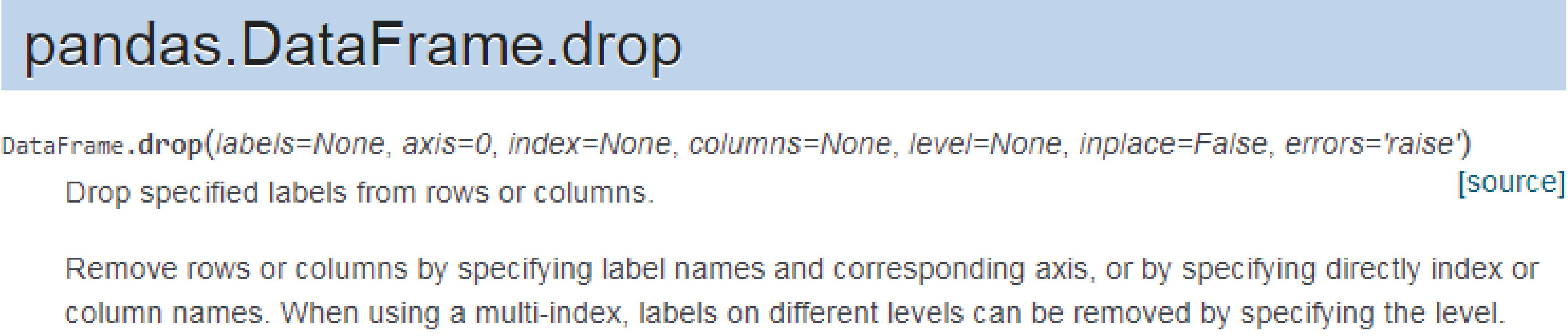
(in-place delete)

2. Use **`pandas.DataFrame.drop`**

```
axis=0, inplace=False
```

(default)

```
0 or 'index', 1 or 'columns'
```



pandas.DataFrame.drop

`DataFrame.drop`(*labels=None*, *axis=0*, *index=None*, *columns=None*, *level=None*, *inplace=False*, *errors='raise'*)

[source]

Drop specified labels from rows or columns.

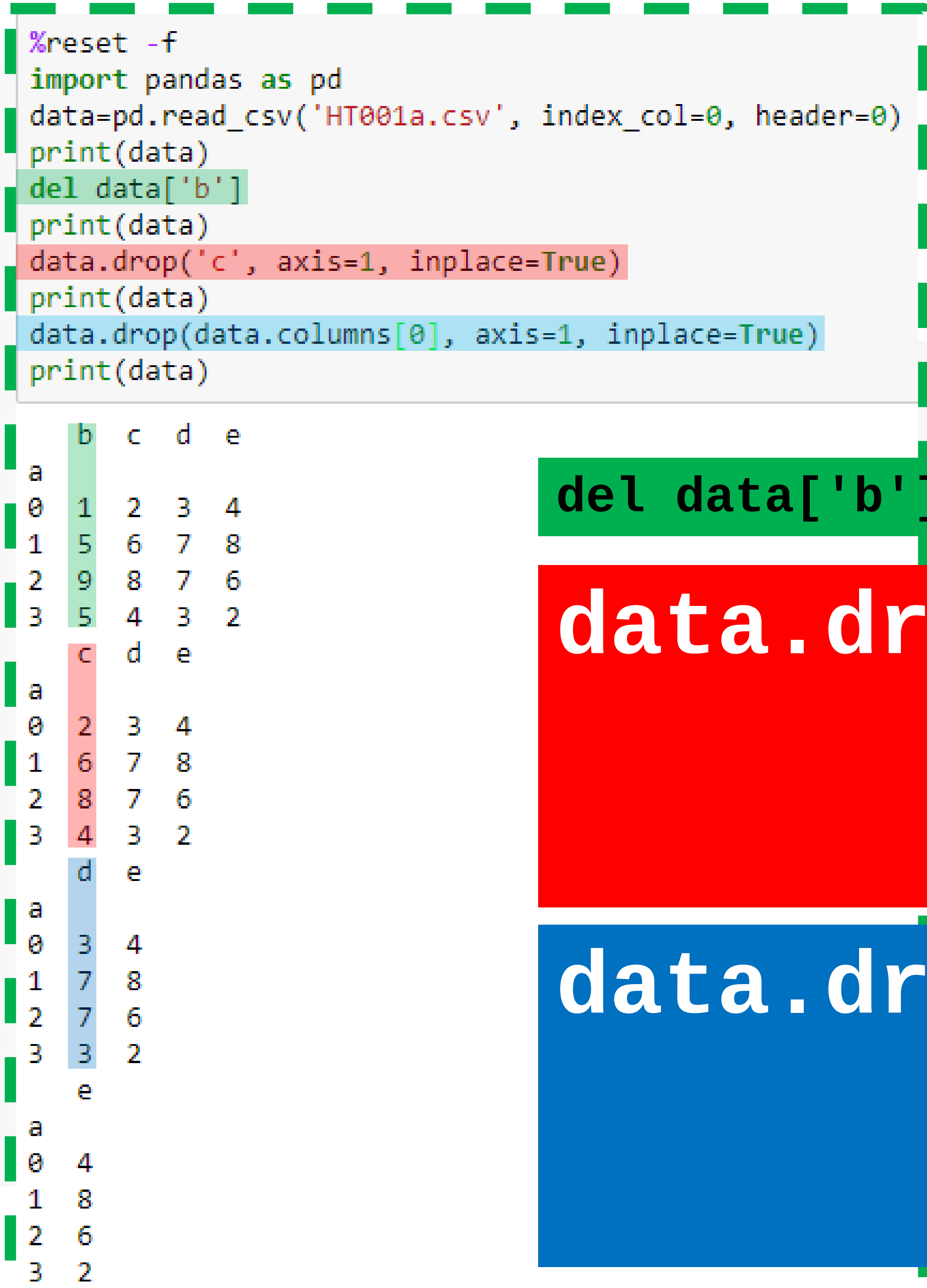
Remove rows or columns by specifying label names and corresponding axis, or by specifying directly index or column names. When using a multi-index, labels on different levels can be removed by specifying the level.

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Slide
217/307



```
%reset -f
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
del data['b']
print(data)
data.drop('c', axis=1, inplace=True)
print(data)
data.drop(data.columns[0], axis=1, inplace=True)
print(data)
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
   c  d  e
a
0  2  3  4
1  6  7  8
2  8  7  6
3  4  3  2
   d  e
a
0  3  4
1  7  8
2  7  6
3  3  2
   e
a
0  4
1  8
2  6
3  2
```

121 122

`del data['b']` or `del data[data.columns[0]]`

```
data.drop('c',
          axis=1,
          inplace=True)
```

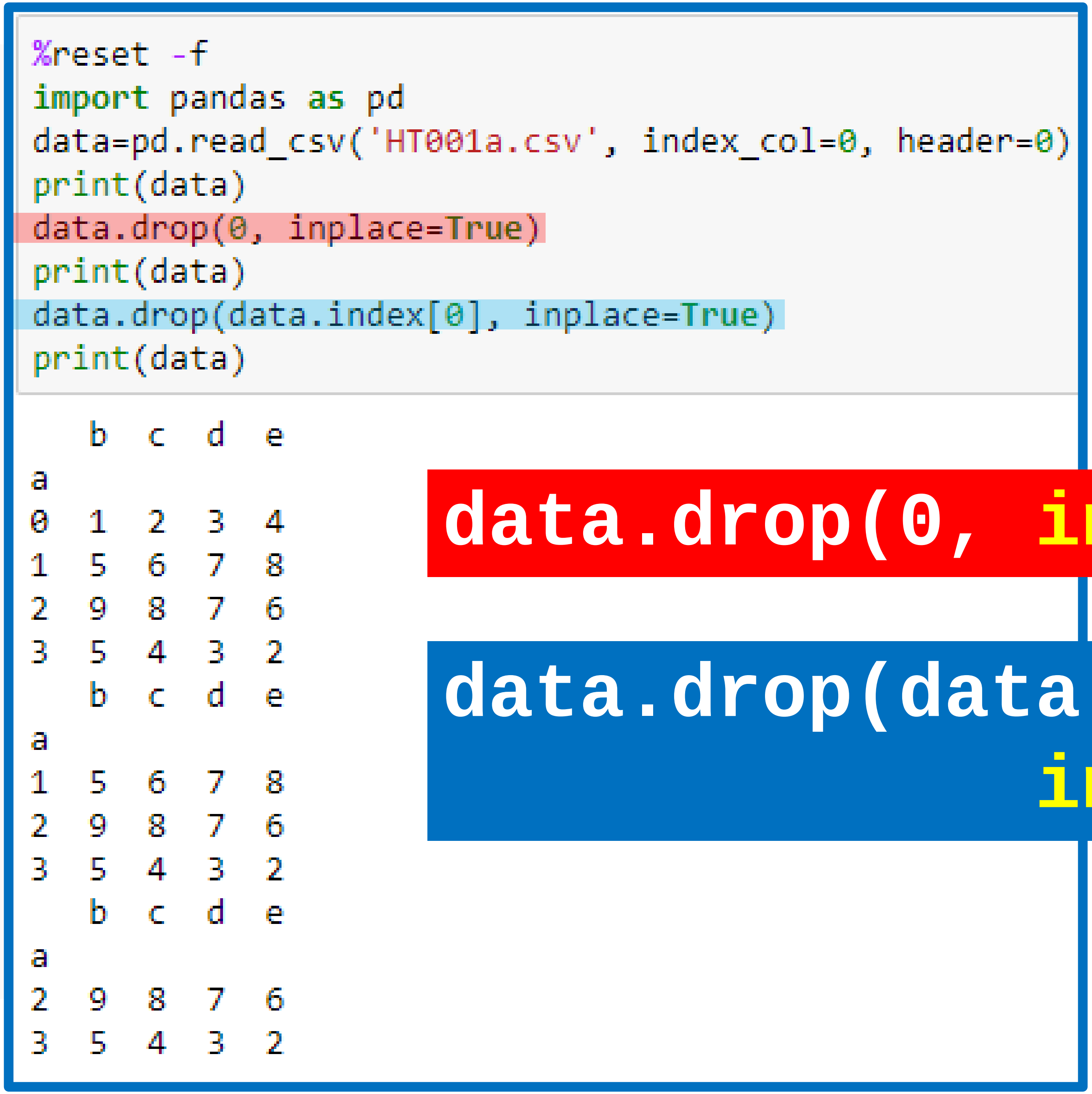
```
data.drop(data.columns[0],
          axis=1,
          inplace=True)
```

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

HT006: Delete a row/column from `data`

```
pandas.DataFrame.drop

data.index

axis=0

inplace=True
```



```
%reset -f
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
data.drop(0, inplace=True)
print(data)
data.drop(data.index[0], inplace=True)
print(data)
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
   b  c  d  e
a
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
   b  c  d  e
a
2  9  8  7  6
3  5  4  3  2
```

123 124

```
data.drop(0, inplace=True)
```

```
data.drop(data.index[0],
                 inplace=True)
```

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

HT006: Delete a row/column from **data**

✓ Delete a column using the “dot syntax”

```
data.b=[]
```

✓ Delete a column using the indexing (position-based and label-based)

```
data(:,1)=[]
```

```
data(:,'b')=[]
```

✓ Delete a row by the row number

```
data(1,:)=[]
```

✓ Delete a row by the row name

```
data('0',:)=[]
```

125 126

127 128

✓ ( )

× { }

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

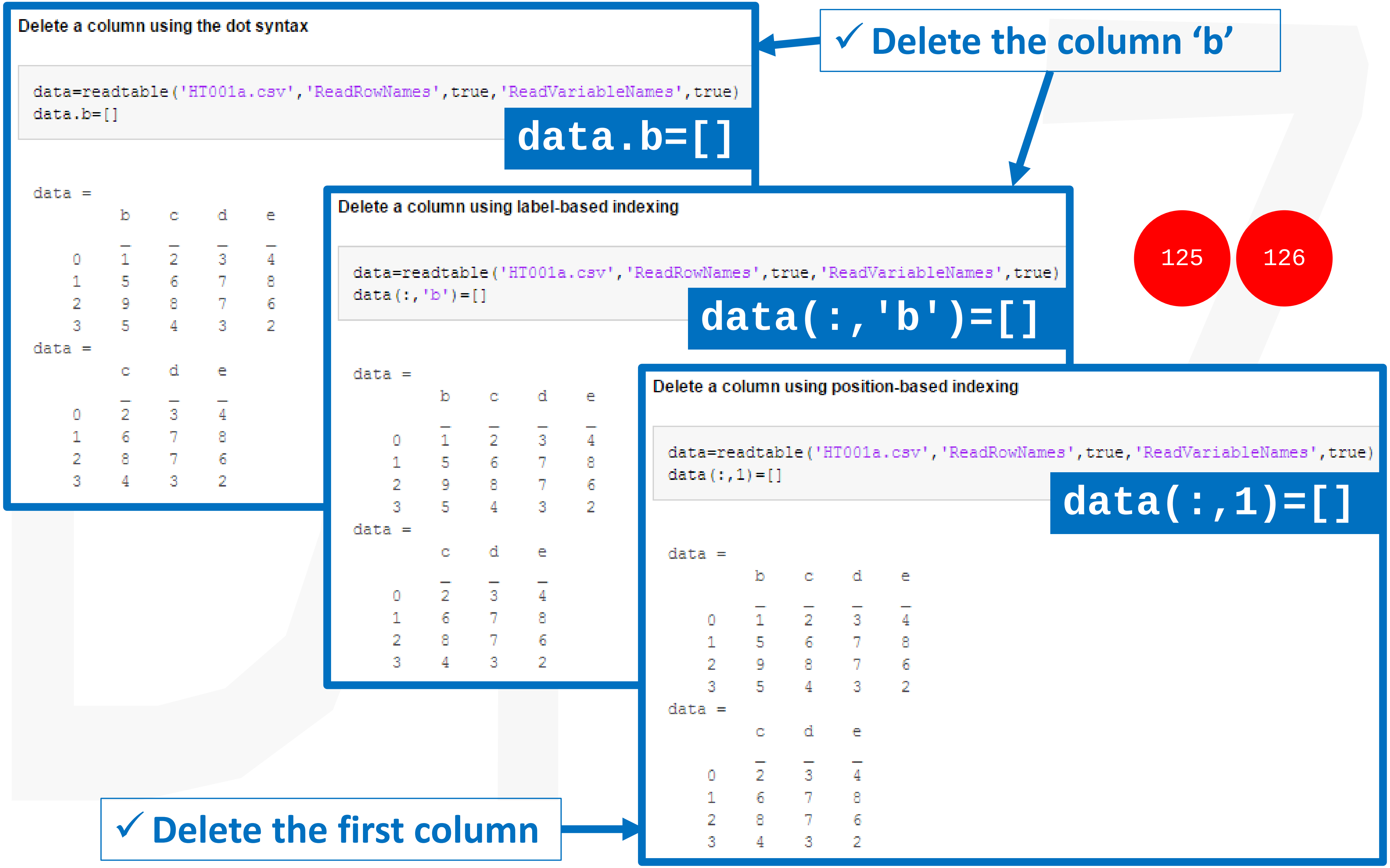
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Slide
221/307

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance



✓ Delete the column ‘b’

Delete a column using the dot syntax

```
data=readtable('HT001a.csv','ReadRowNames',true,'ReadVariableNames',true)
data.b=[]
```

data.b=[]

```
data =
          b    c    d    e
          _    _    _    _
     0    1    2    3    4
     1    5    6    7    8
     2    9    8    7    6
     3    5    4    3    2
data =
          c    d    e
          _    _    _
     0    2    3    4
     1    6    7    8
     2    8    7    6
     3    4    3    2
```

Delete a column using label-based indexing

```
data=readtable('HT001a.csv','ReadRowNames',true,'ReadVariableNames',true)
data(:,'b')=[]
```

data(:,'b')=[]

```
data =
          b    c    d    e
          _    _    _    _
     0    1    2    3    4
     1    5    6    7    8
     2    9    8    7    6
     3    5    4    3    2
data =
          c    d    e
          _    _    _
     0    2    3    4
     1    6    7    8
     2    8    7    6
     3    4    3    2
```

Delete a column using position-based indexing

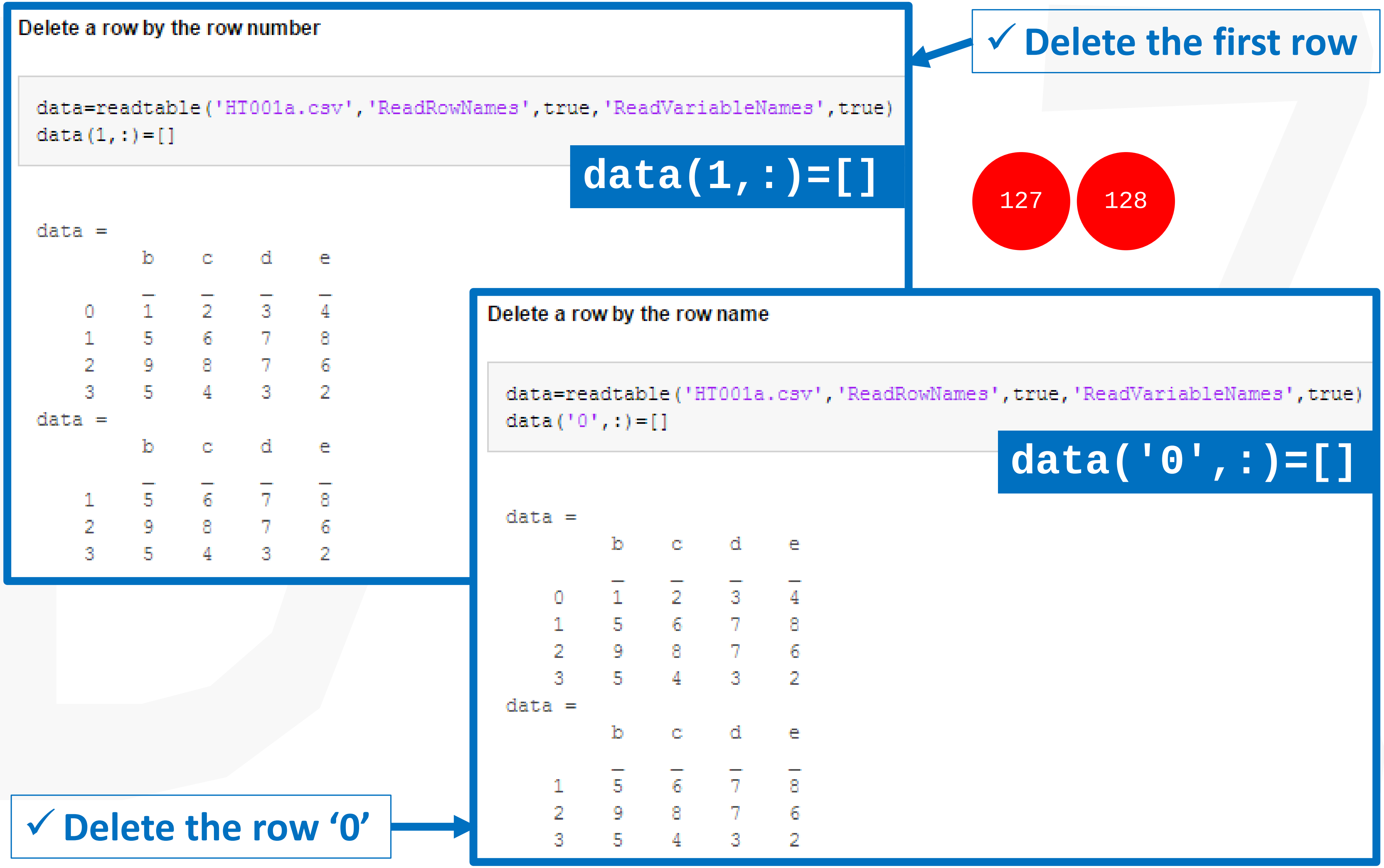
```
data=readtable('HT001a.csv','ReadRowNames',true,'ReadVariableNames',true)
data(:,1)=[]
```

data(:,1)=[]

```
data =
          b    c    d    e
          _    _    _    _
     0    1    2    3    4
     1    5    6    7    8
     2    9    8    7    6
     3    5    4    3    2
data =
          c    d    e
          _    _    _
     0    2    3    4
     1    6    7    8
     2    8    7    6
     3    4    3    2
```

✓ Delete the first column

125 126



Delete a row by the row number

```
data=readtable('HT001a.csv','ReadRowNames',true,'ReadVariableNames',true)
data(1,:)=[]
```

```
data =
            b    c    d    e

            _    _    _    _
      0     1    2    3    4
      1     5    6    7    8
      2     9    8    7    6
      3     5    4    3    2
data =
            b    c    d    e

            _    _    _    _
      1     5    6    7    8
      2     9    8    7    6
      3     5    4    3    2
```

`data(1,:)=[]`

✓ Delete the first row

127 128

Delete a row by the row name

```
data=readtable('HT001a.csv','ReadRowNames',true,'ReadVariableNames',true)
data('0',:)=[]
```

```
data =
            b    c    d    e

            _    _    _    _
      0     1    2    3    4
      1     5    6    7    8
      2     9    8    7    6
      3     5    4    3    2
data =
            b    c    d    e

            _    _    _    _
      1     5    6    7    8
      2     9    8    7    6
      3     5    4    3    2
```

`data('0',:)=[]`

✓ Delete the row ‘0’

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Q: How to swap two values in variables **a** and **b**?

??????????????????????????

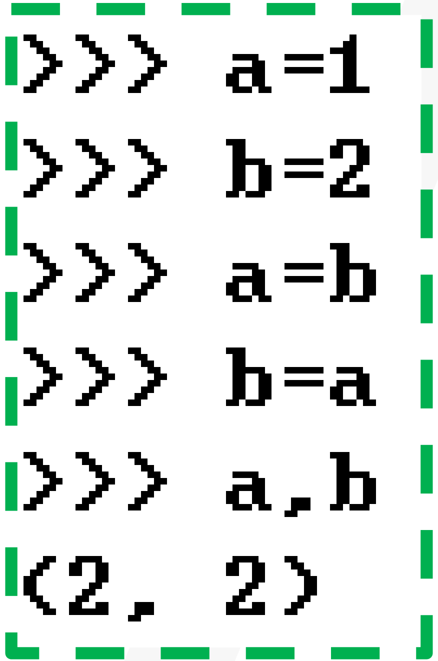
?

? a=b

? b=a

?

??????????????????????????



```
>>> a=1
>>> b=2
>>> a=b
>>> b=a
>>> a,b
(2, 2)
```

SMU SINGAPORE MANAGEMENT UNIVERSITY

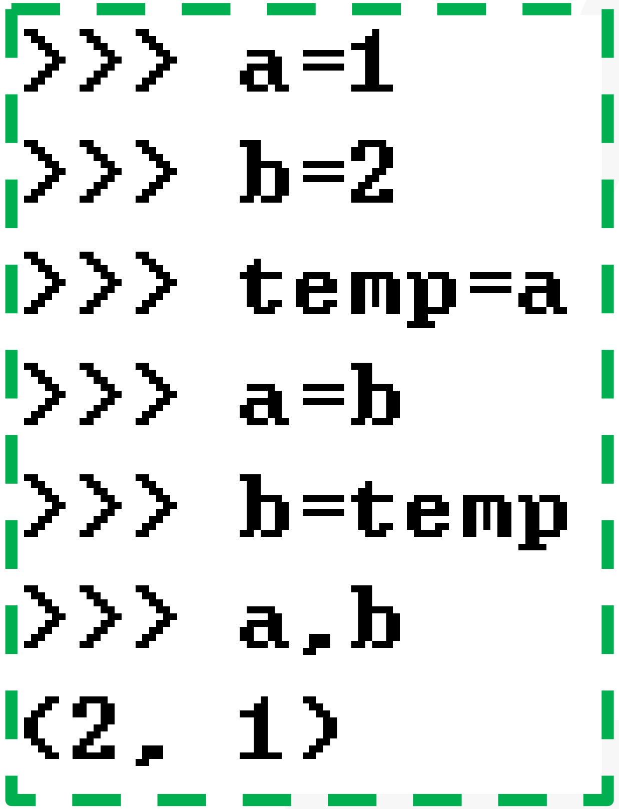
QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

(continued)

```
temp=a
a=b
b=temp
```

129



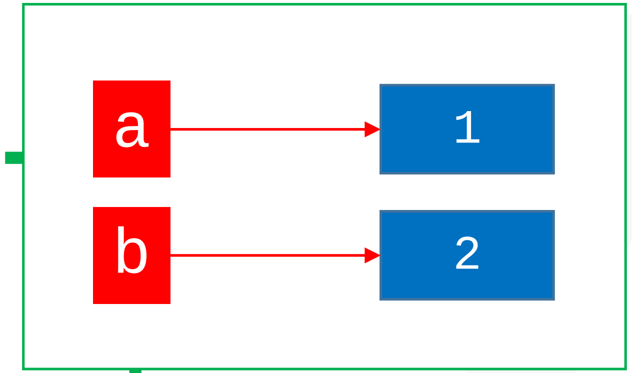
```
>>> a=1
>>> b=2
>>> temp=a
>>> a=b
>>> b=temp
>>> a,b
(2, 1)
```

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

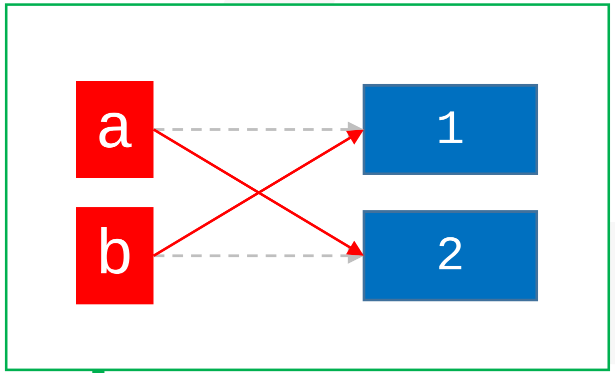
(continued)

```
>>> a=1
>>> b=2
>>> a, b=b, a
>>> a,b
(2, 1)
```

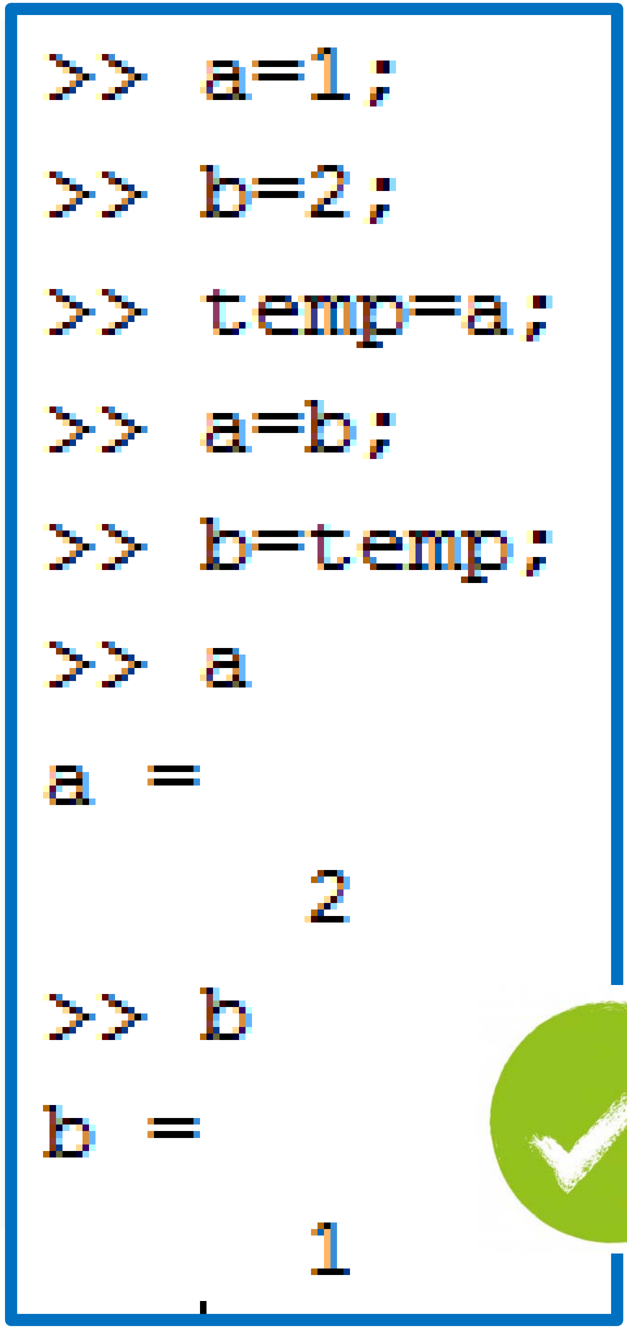


expr3, expr4=expr1, expr2

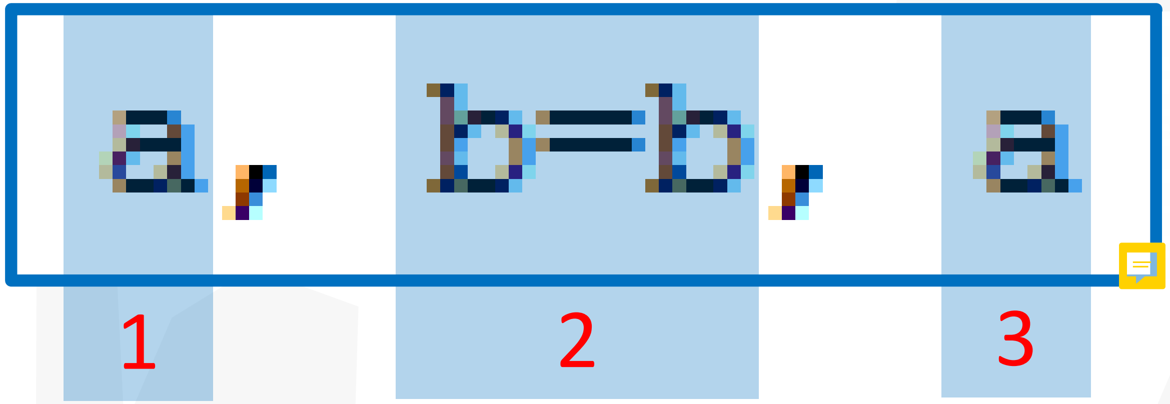
129



Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

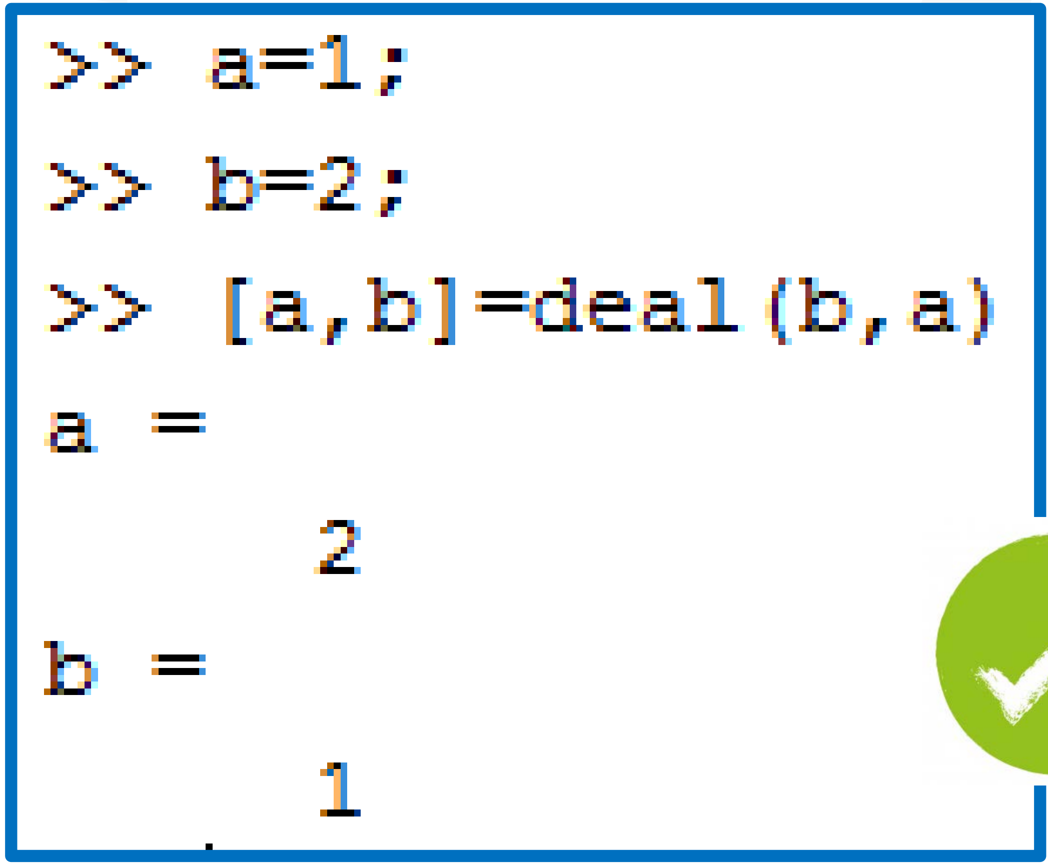


```
>> a=1;
>> b=2;
>> temp=a;
>> a=b;
>> b=temp;
>> a
a =
     2
>> b
b =
     1
```



```
a, b=b, a
```

130



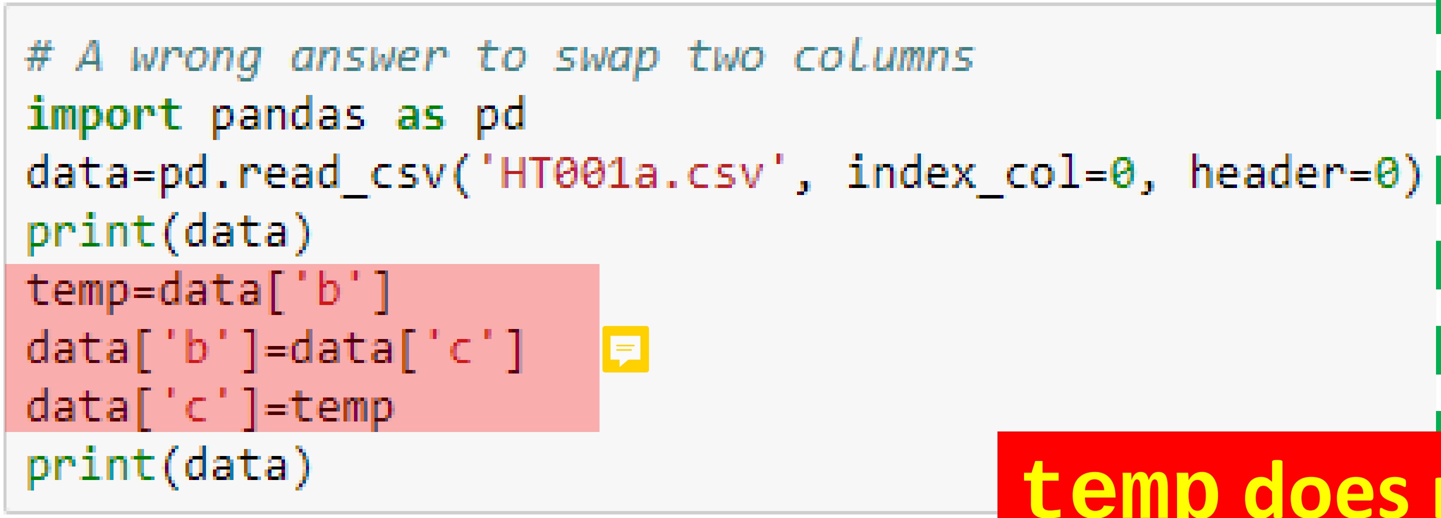
```
>> a=1;
>> b=2;
>> [a,b]=deal(b,a)
a =
     2
b =
     1
```

SMU
SINGAPORE MANAGEMENT UNIVERSITY

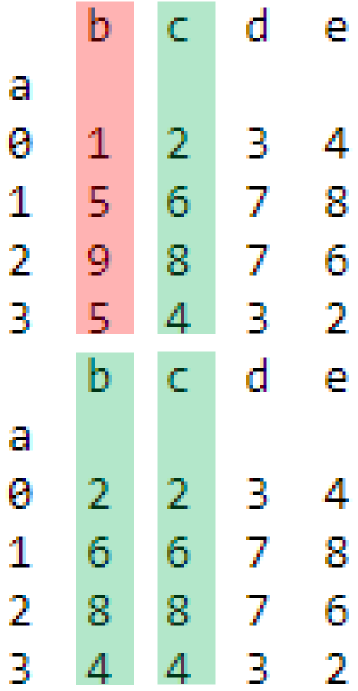
QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance

HT007: Swap two rows/columns in **data**



```
# A wrong answer to swap two columns
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
temp=data['b']
data['b']=data['c']
data['c']=temp
print(data)
```



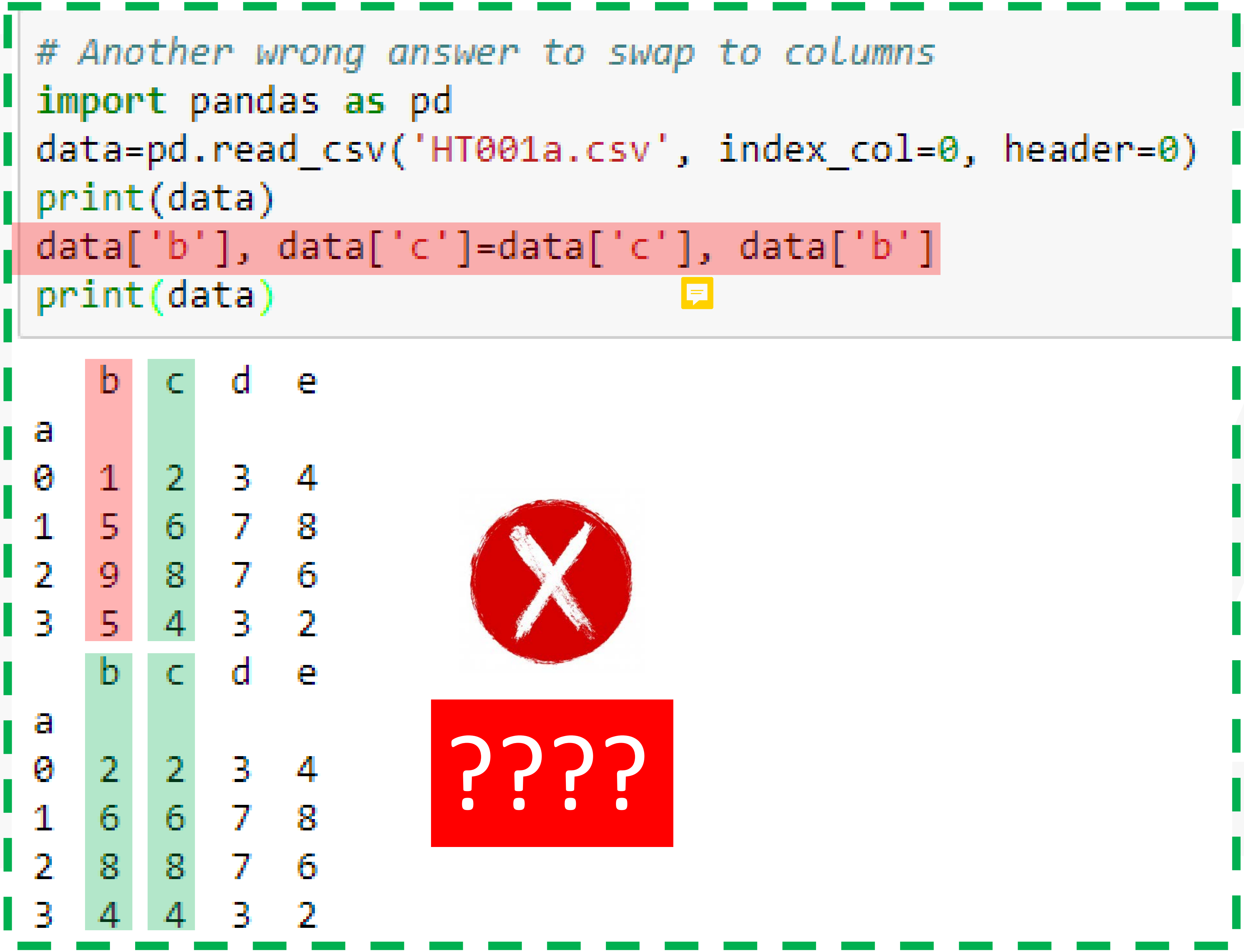
```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
   b  c  d  e
a
0  2  2  3  4
1  6  6  7  8
2  8  8  7  6
3  4  4  3  2
```

????

1. Select data via `[]`
2. Select data via `.loc` or `.iloc`

temp does not work!

`data['b']` and **`data['c']`** are views. A view is something like a “pointer”. Assignment to a view is function to use the pointer to assign values.



```python
# Another wrong answer to swap to columns
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
data['b'], data['c']=data['c'], data['b']
print(data)
```

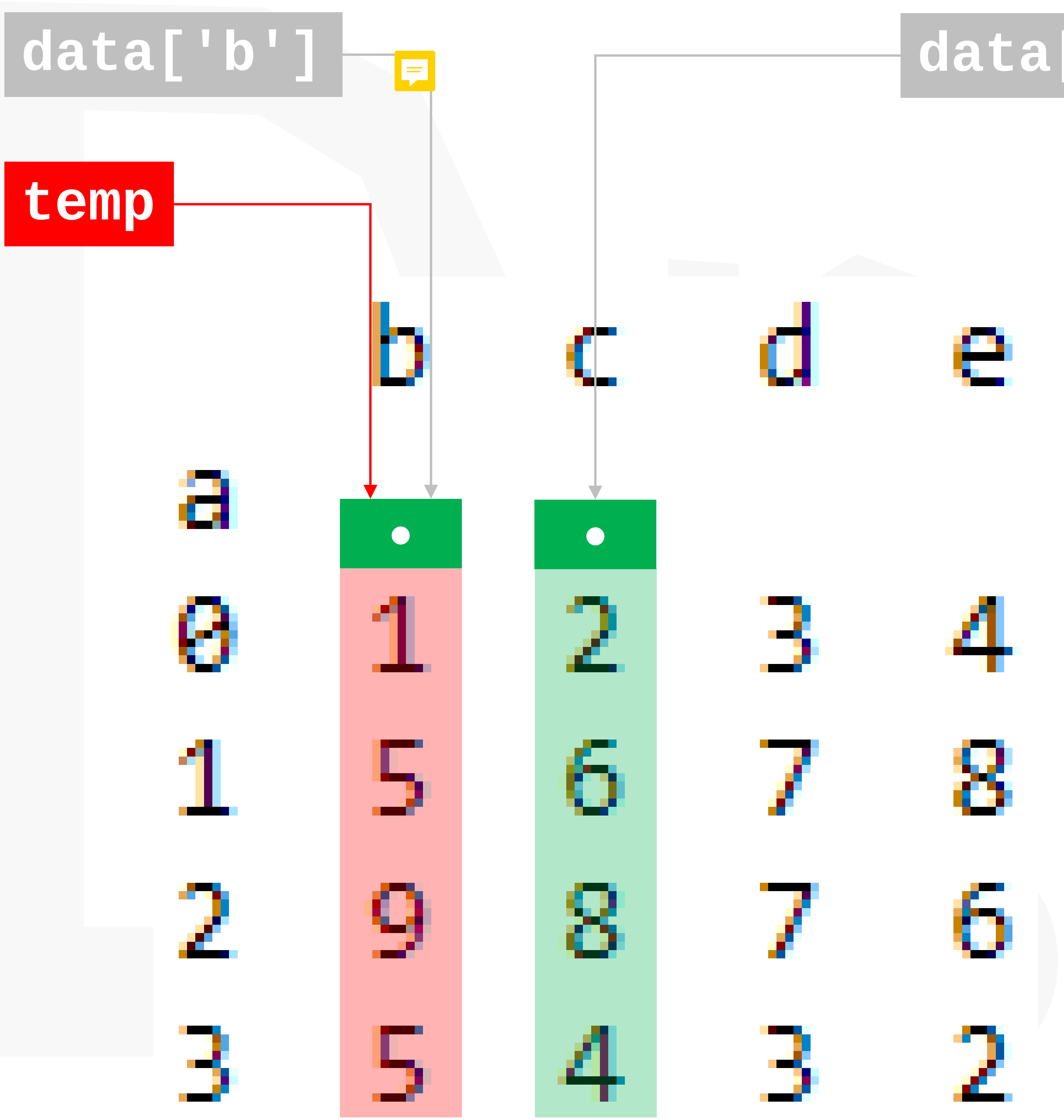
```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
   b  c  d  e
a
0  2  2  3  4
1  6  6  7  8
2  8  8  7  6
3  4  4  3  2
```

????

SMU SINGAPORE MANAGEMENT UNIVERSITY

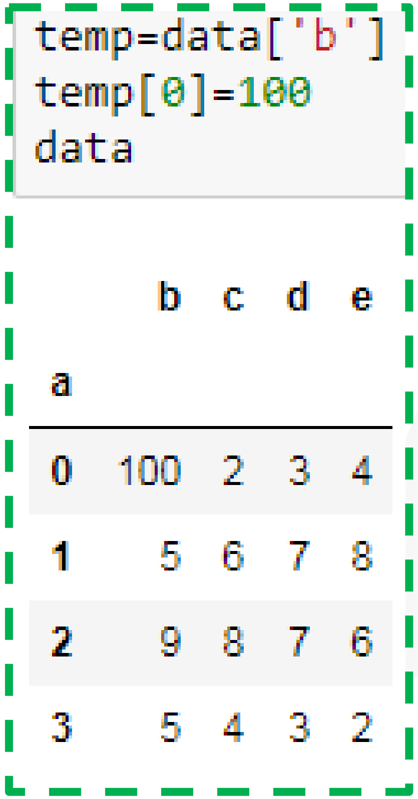
QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



	b	c	d	e
a	●	●		
0	1	2	3	4
1	5	6	7	8
2	9	8	7	6
3	5	4	3	2

```
temp=data['b']
data['b']=data['c']
data['c']=temp
```



```
temp=data['b']
temp[0]=100
data
```

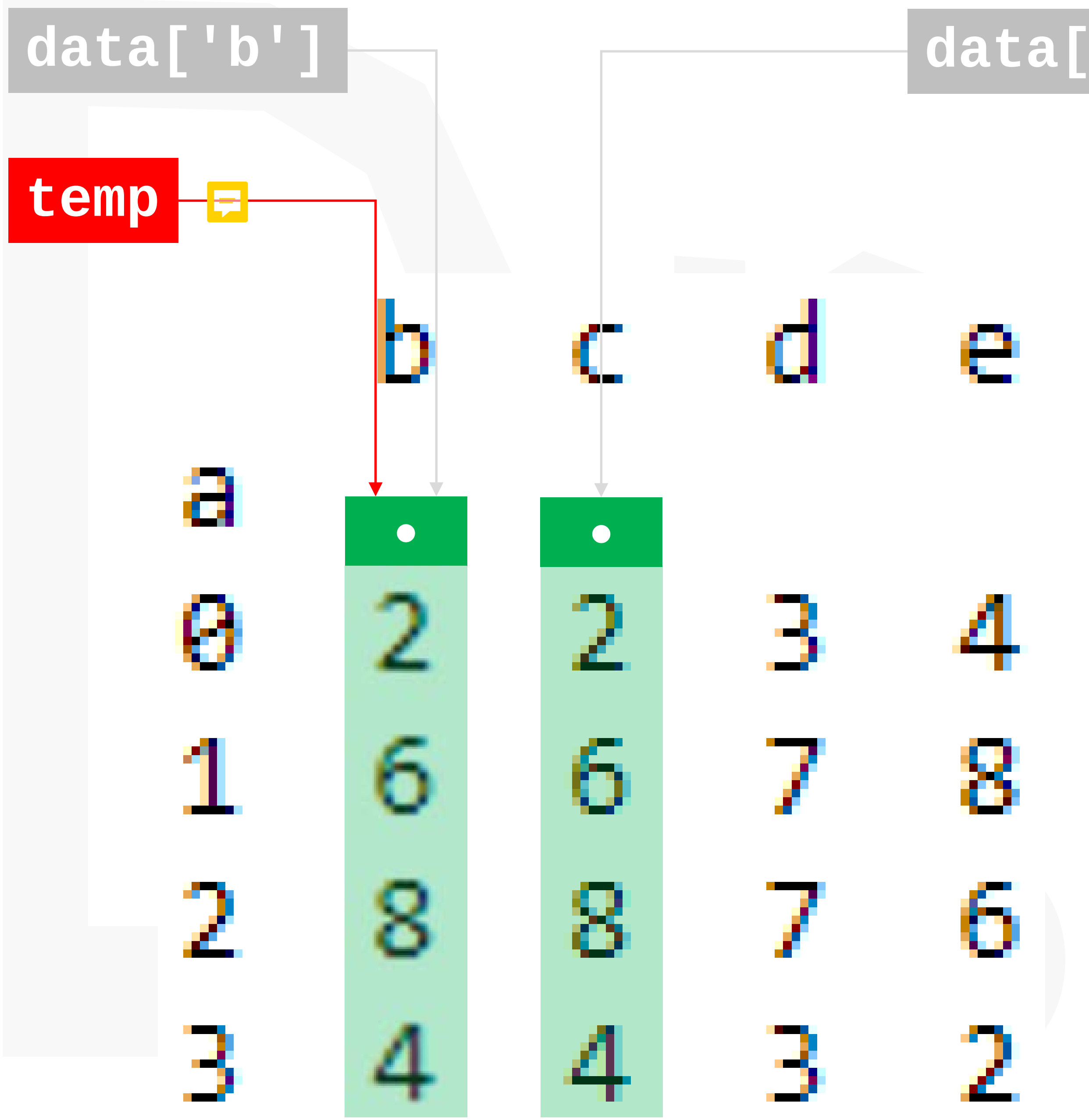
```
     b  c  d  e
a
0  100  2  3  4
1    5  6  7  8
2    9  8  7  6
3    5  4  3  2
```

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

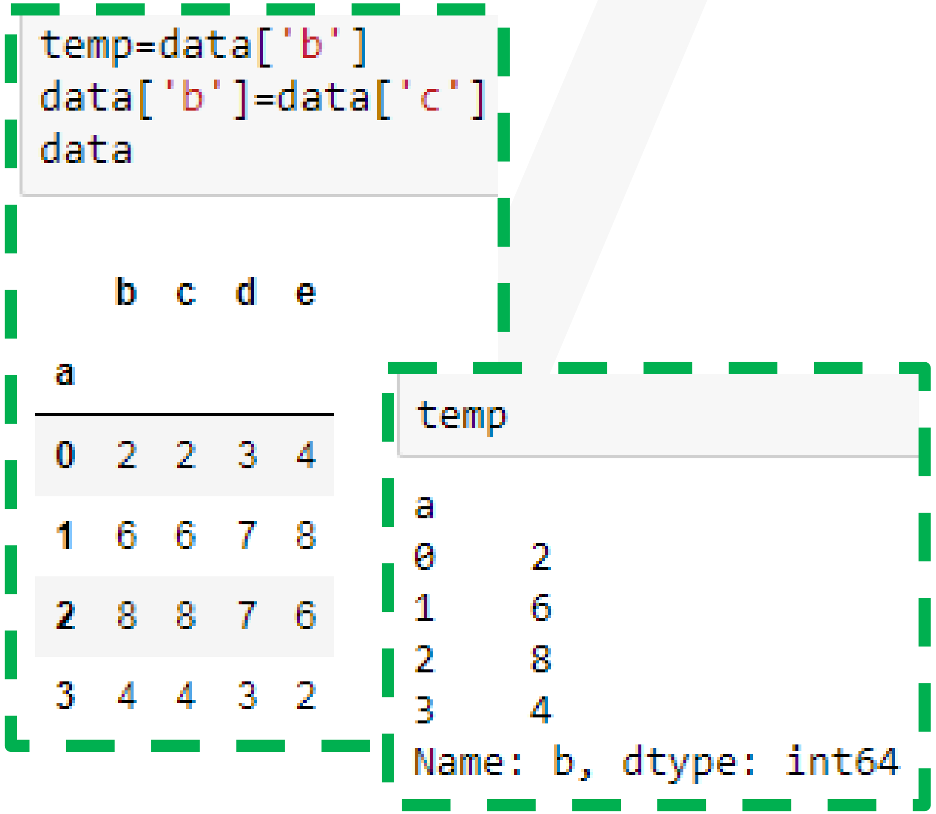
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Slide
230/307



	b	c	d	e
a	●	●		
0	2	2	3	4
1	6	6	7	8
2	8	8	7	6
3	4	4	3	2

```
temp=data['b']
data['b']=data['c']
data['c']=temp
```



```
temp=data['b']
data['b']=data['c']
data
```

```
   b  c  d  e
a
0  2  2  3  4
1  6  6  7  8
2  8  8  7  6
3  4  4  3  2
```

```
temp
```

```
a
0    2
1    6
2    8
3    4
Name: b, dtype: int64
```

SMU
SINGAPORE MANAGEMENT UNIVERSITY

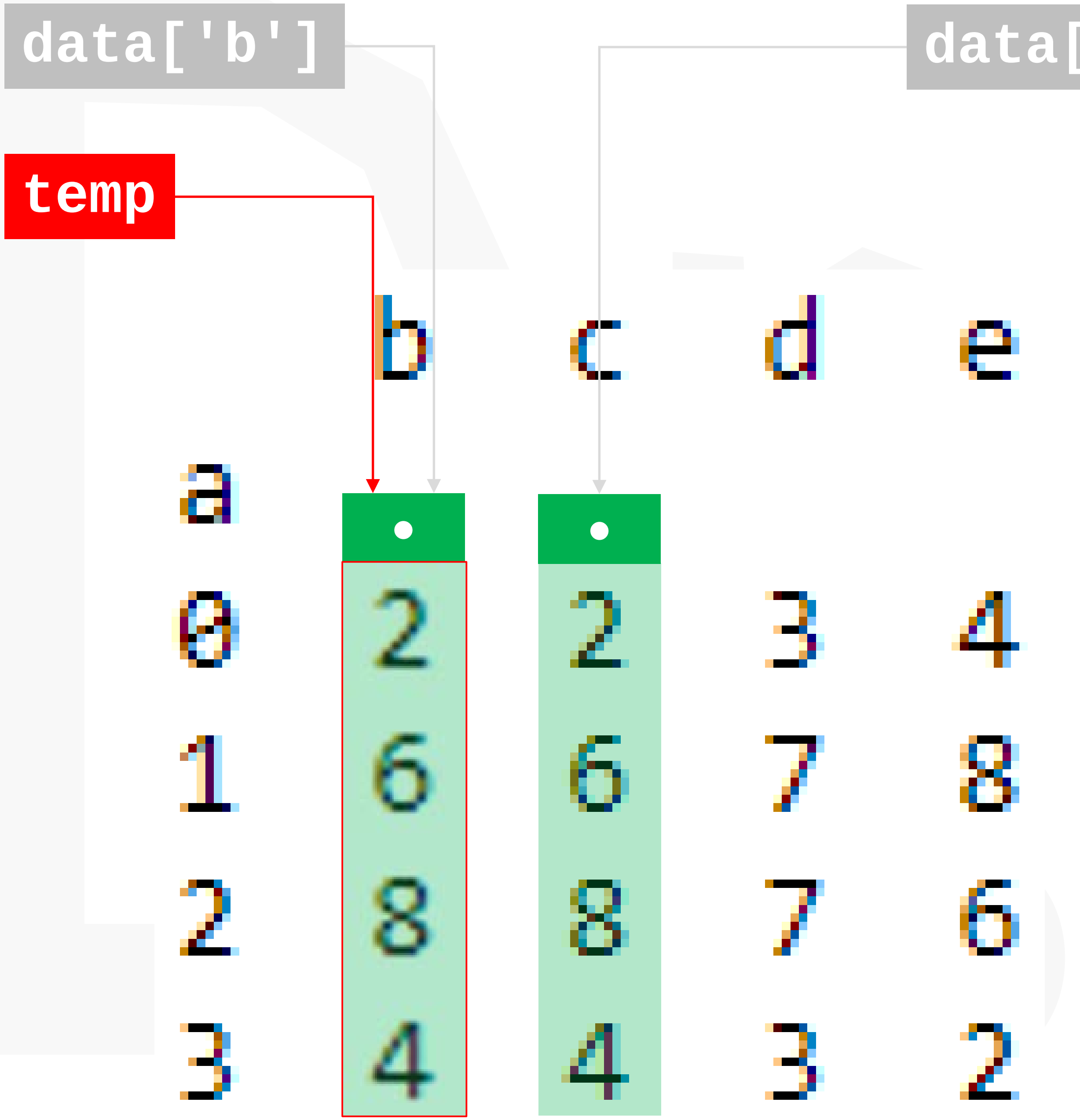
QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance



	b	c	d	e
a	•	•		
0	2	2	3	4
1	6	6	7	8
2	8	8	7	6
3	4	4	3	2

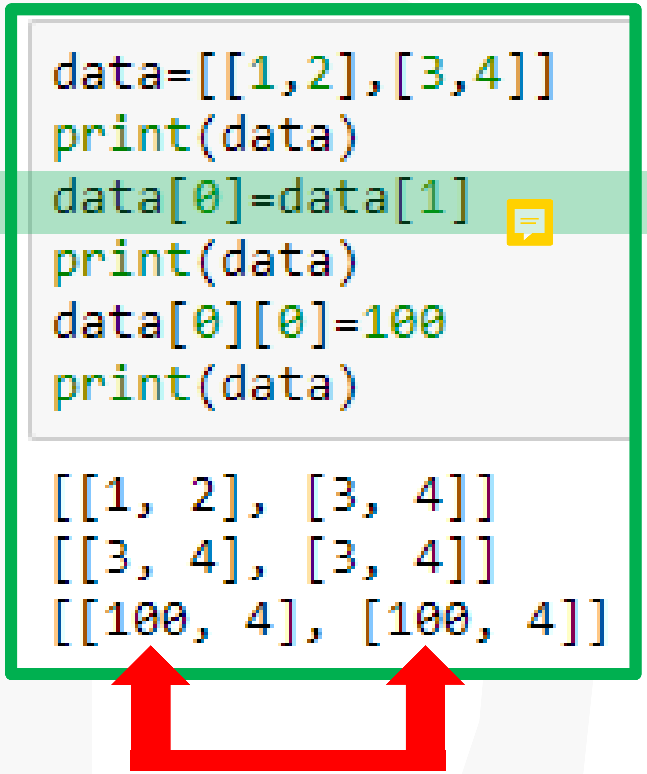
```
temp=data['b']
data['b']=data['c']
data['c']=temp
```

Dr. Z’s interpretation on the assignment to a view

Slide
232/307

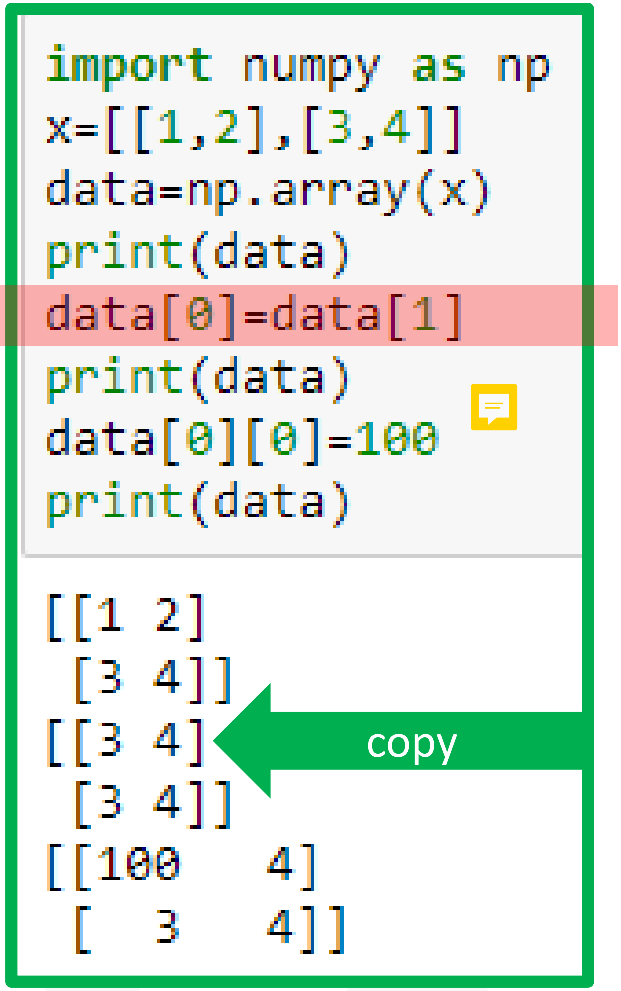
QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



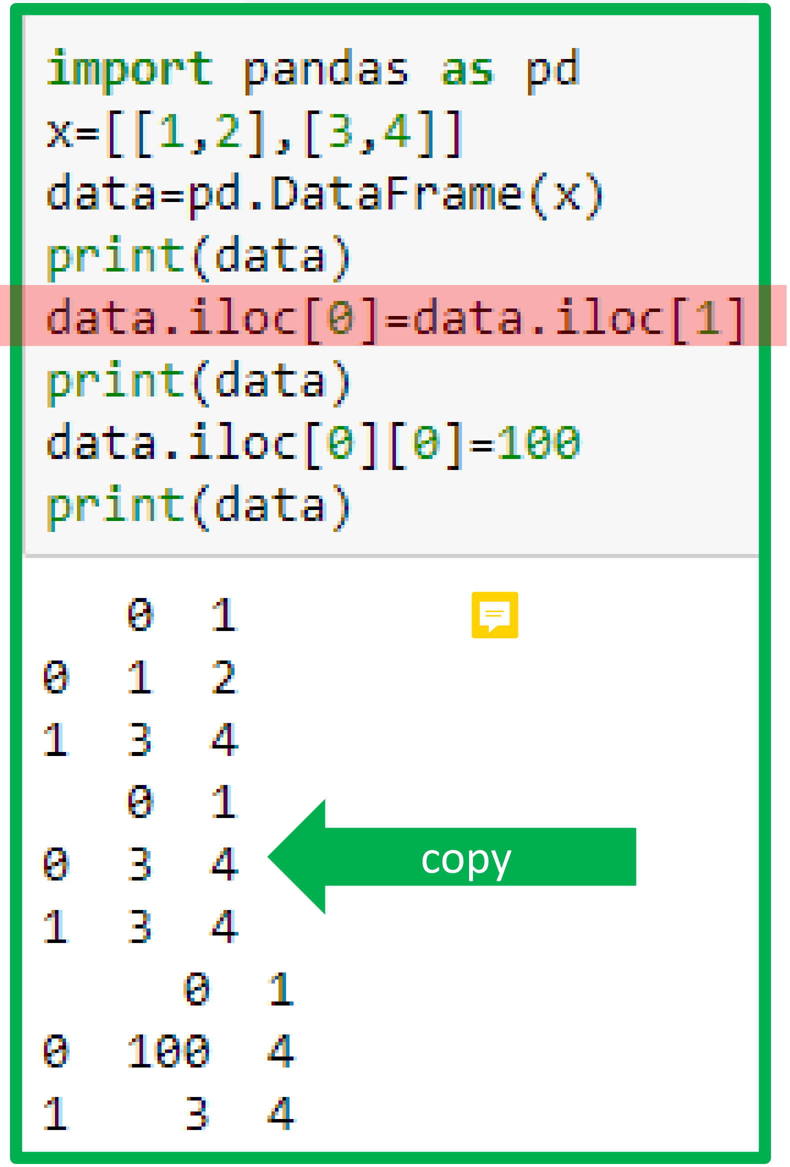
```
data=[[1,2],[3,4]]
print(data)
data[0]=data[1]
print(data)
data[0][0]=100
print(data)
```

```
[[1, 2], [3, 4]]
[[3, 4], [3, 4]]
[[100, 4], [100, 4]]
```



```
import numpy as np
x=[[1,2],[3,4]]
data=np.array(x)
print(data)
data[0]=data[1]
print(data)
data[0][0]=100
print(data)
```

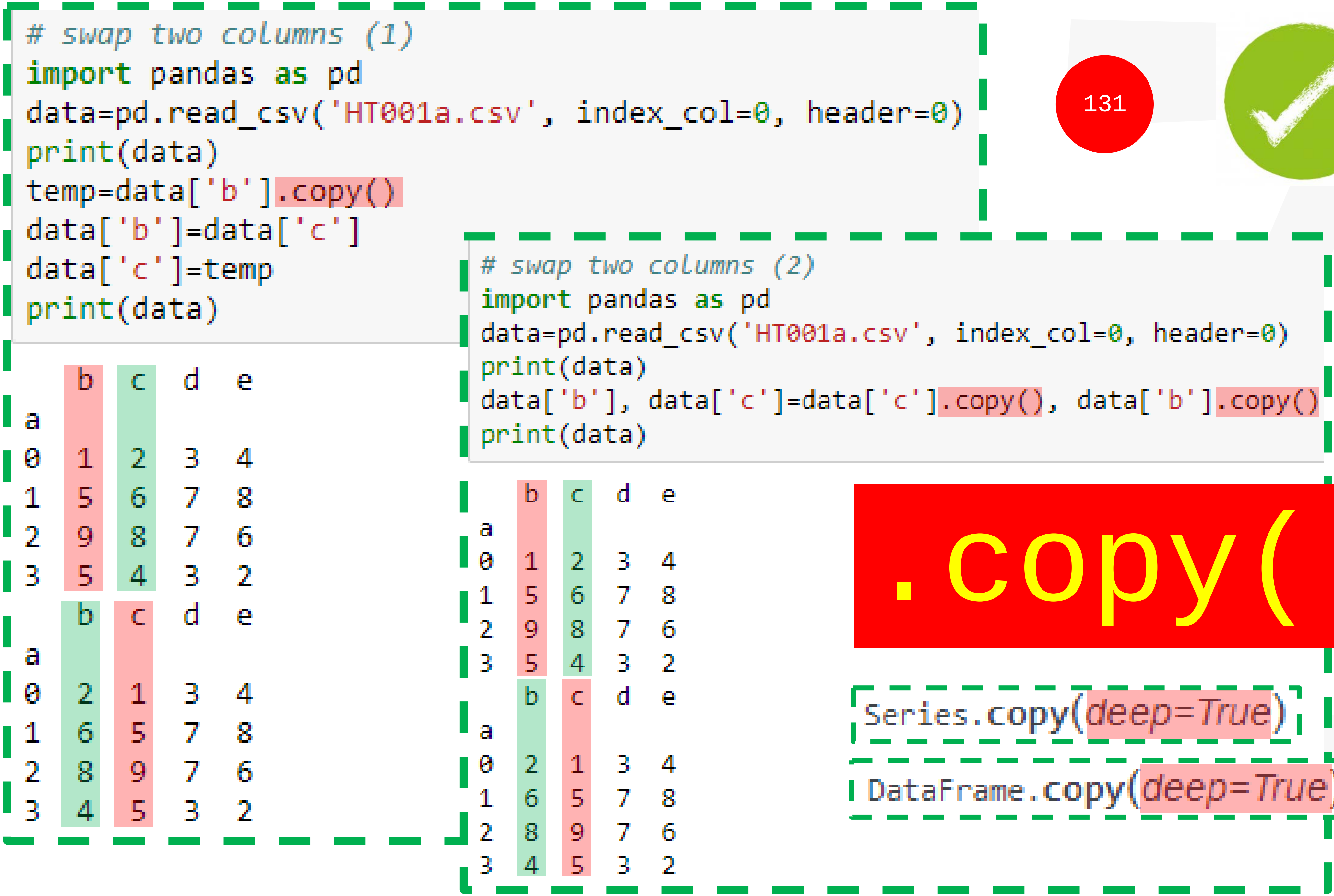
```
[[1 2]
 [3 4]]
[[3 4]
 [3 4]]
[[100   4]
 [  3   4]]
```



```
import pandas as pd
x=[[1,2],[3,4]]
data=pd.DataFrame(x)
print(data)
data.iloc[0]=data.iloc[1]
print(data)
data.iloc[0][0]=100
print(data)
```

```
   0  1
0  1  2
1  3  4
   0  1
0  3  4
1  3  4
     0  1
0  100  4
1    3  4
```

131



```
# swap two columns (1)
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
temp=data['b'].copy()
data['b']=data['c']
data['c']=temp
print(data)
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
   b  c  d  e
a
0  2  1  3  4
1  6  5  7  8
2  8  9  7  6
3  4  5  3  2
```

```
# swap two columns (2)
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
data['b'], data['c']=data['c'].copy(), data['b'].copy()
print(data)
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
   b  c  d  e
a
0  2  1  3  4
1  6  5  7  8
2  8  9  7  6
3  4  5  3  2
```

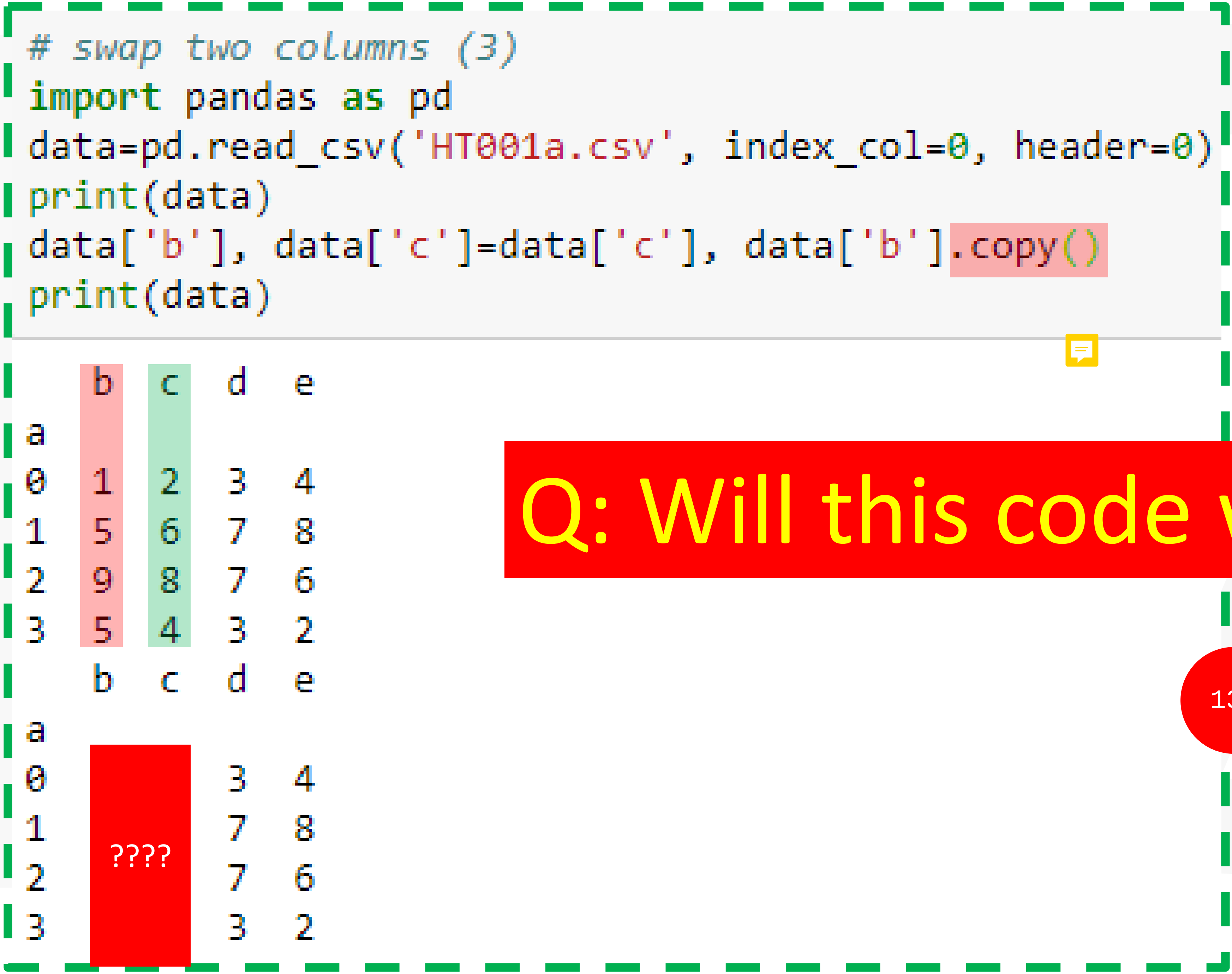
.copy()

Series.copy(deep=True)

DataFrame.copy(deep=True)

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
# swap two columns (3)
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
data['b'], data['c']=data['c'], data['b'].copy()
print(data)
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
   b  c  d  e
a
0        3  4
1        7  8
2  ????  7  6
3        3  2
```

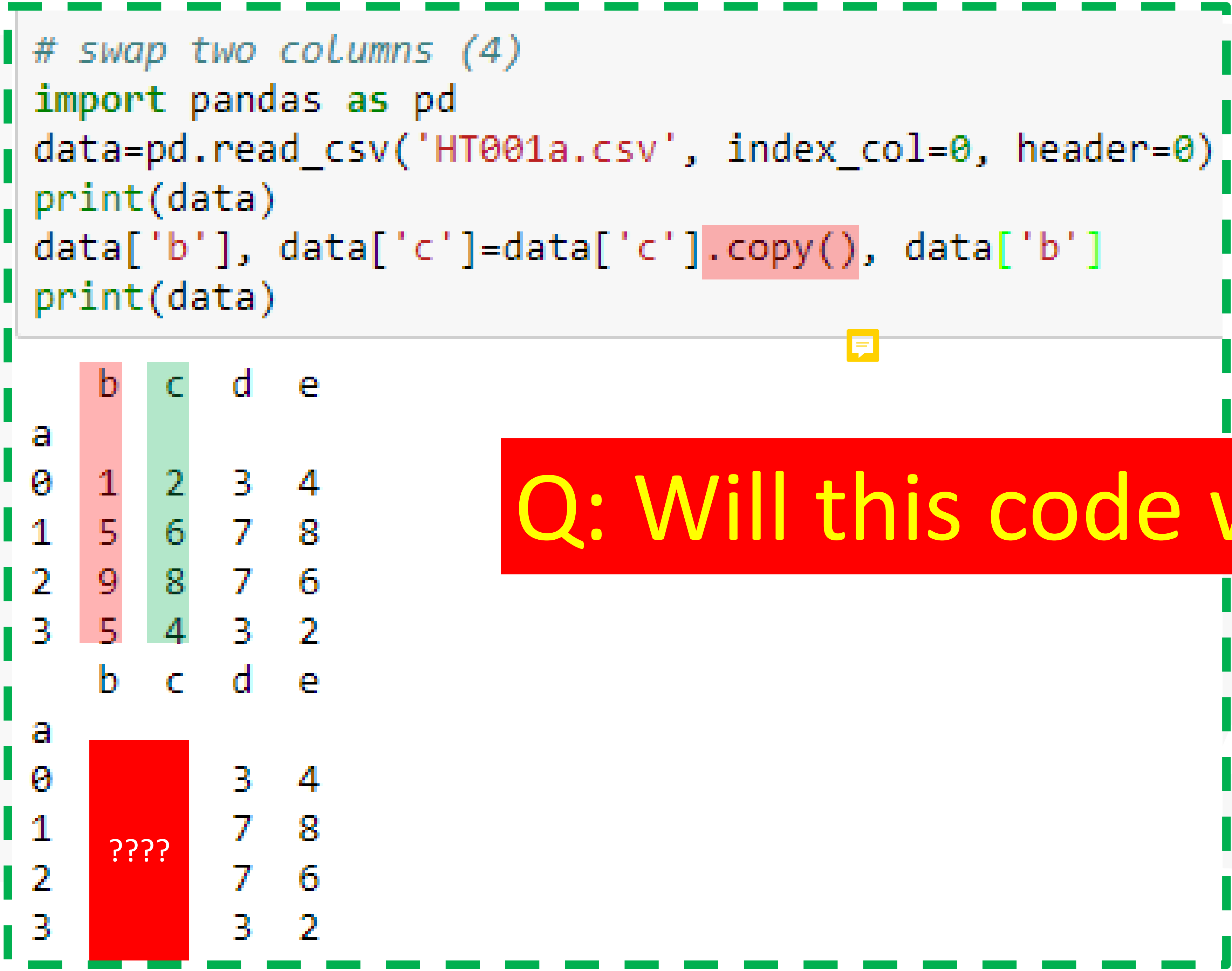
Q: Will this code work?

131

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
# swap two columns (4)
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
data['b'], data['c']=data['c'].copy(), data['b']
print(data)
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
   b  c  d  e
a
0        3  4
1  ????  7  8
2        7  6
3        3  2
```

Q: Will this code work?

SMU
SINGAPORE MANAGEMENT UNIVERSITY

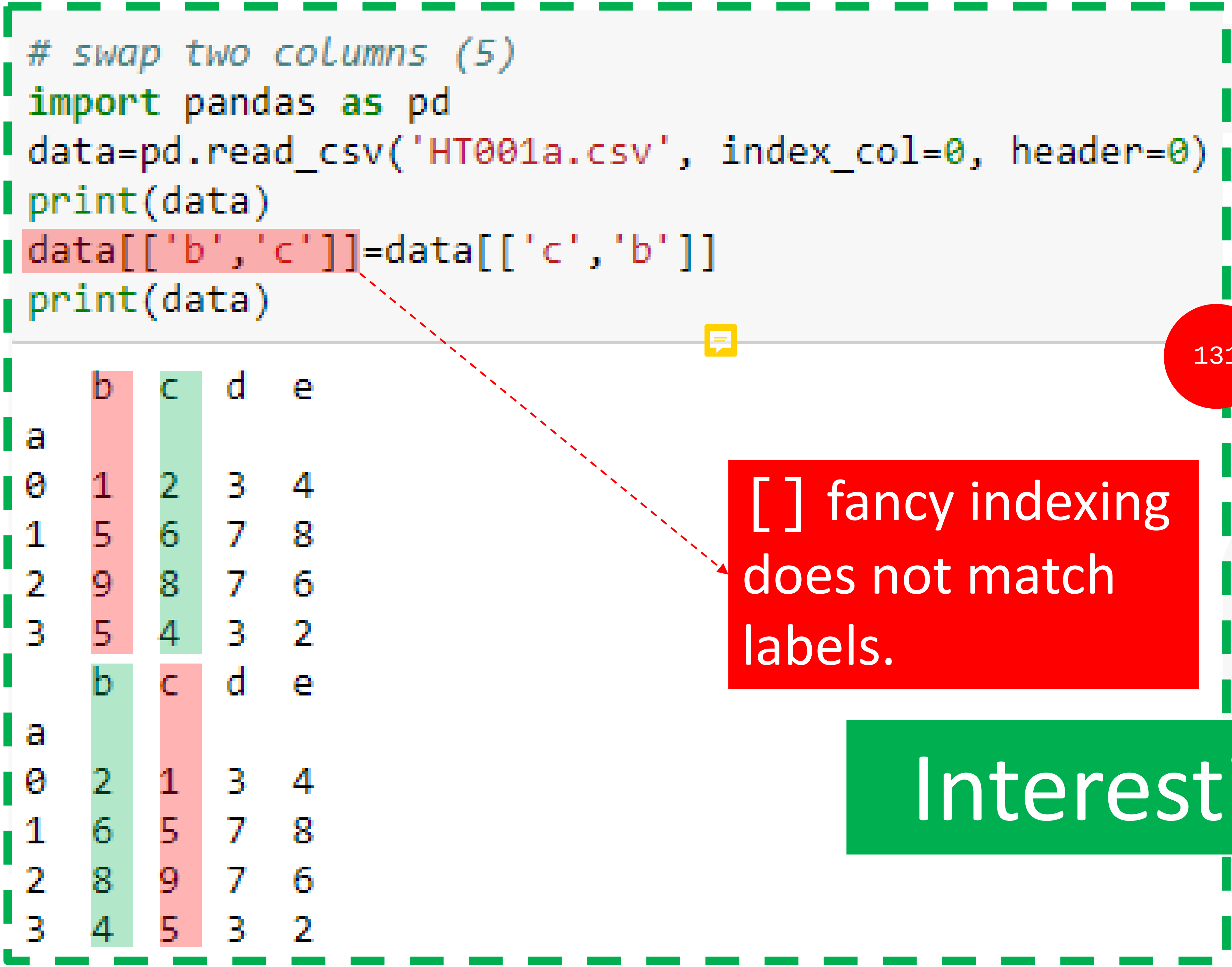
QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance



```
# swap two columns (5)
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
data[['b','c']]=data[['c','b']]
print(data)
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
   b  c  d  e
a
0  2  1  3  4
1  6  5  7  8
2  8  9  7  6
3  4  5  3  2
```

131

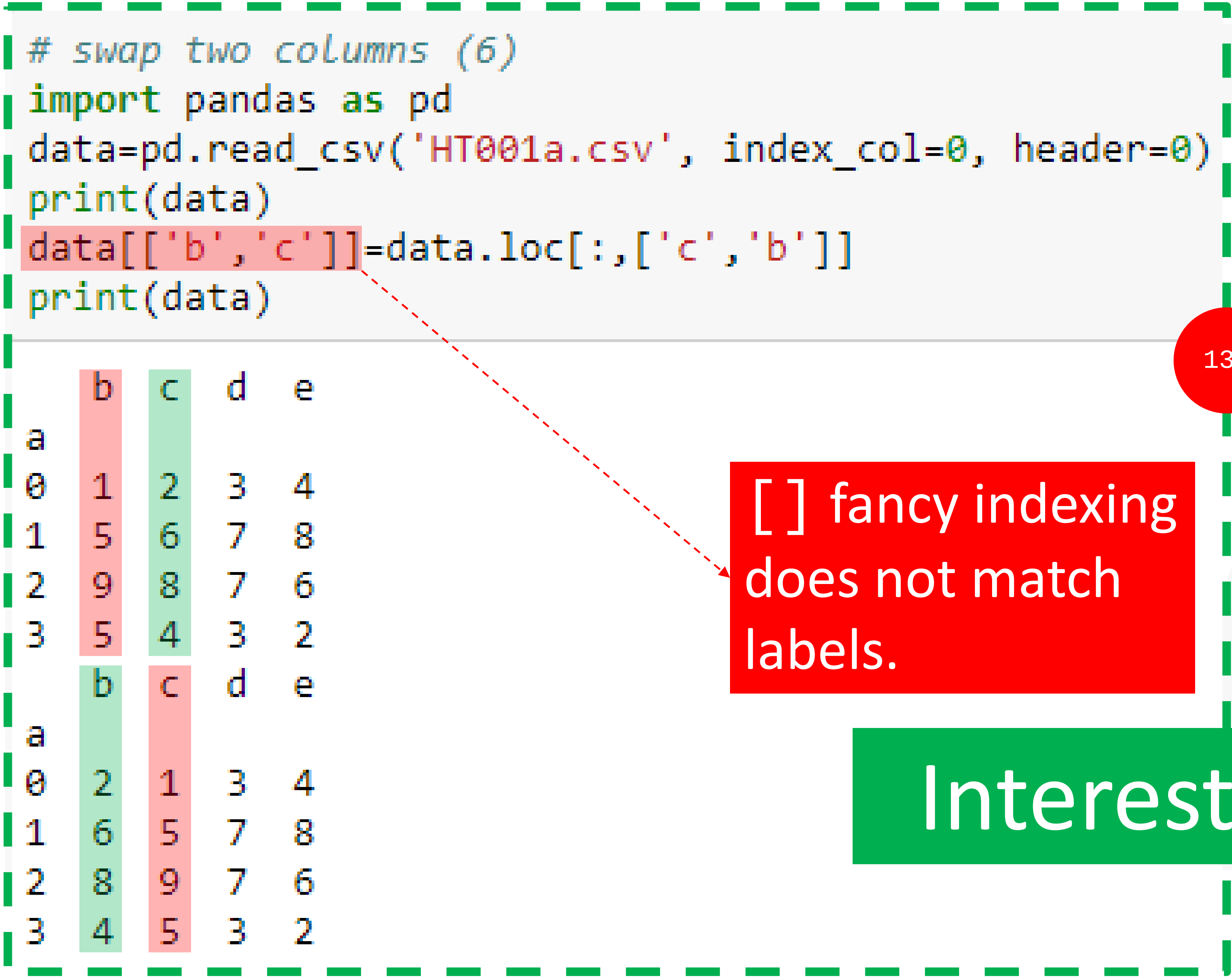
[ ] fancy indexing does not match labels.

Interesting...

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance



```
# swap two columns (6)
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
data[['b','c']]=data.loc[:,['c','b']]
print(data)
```

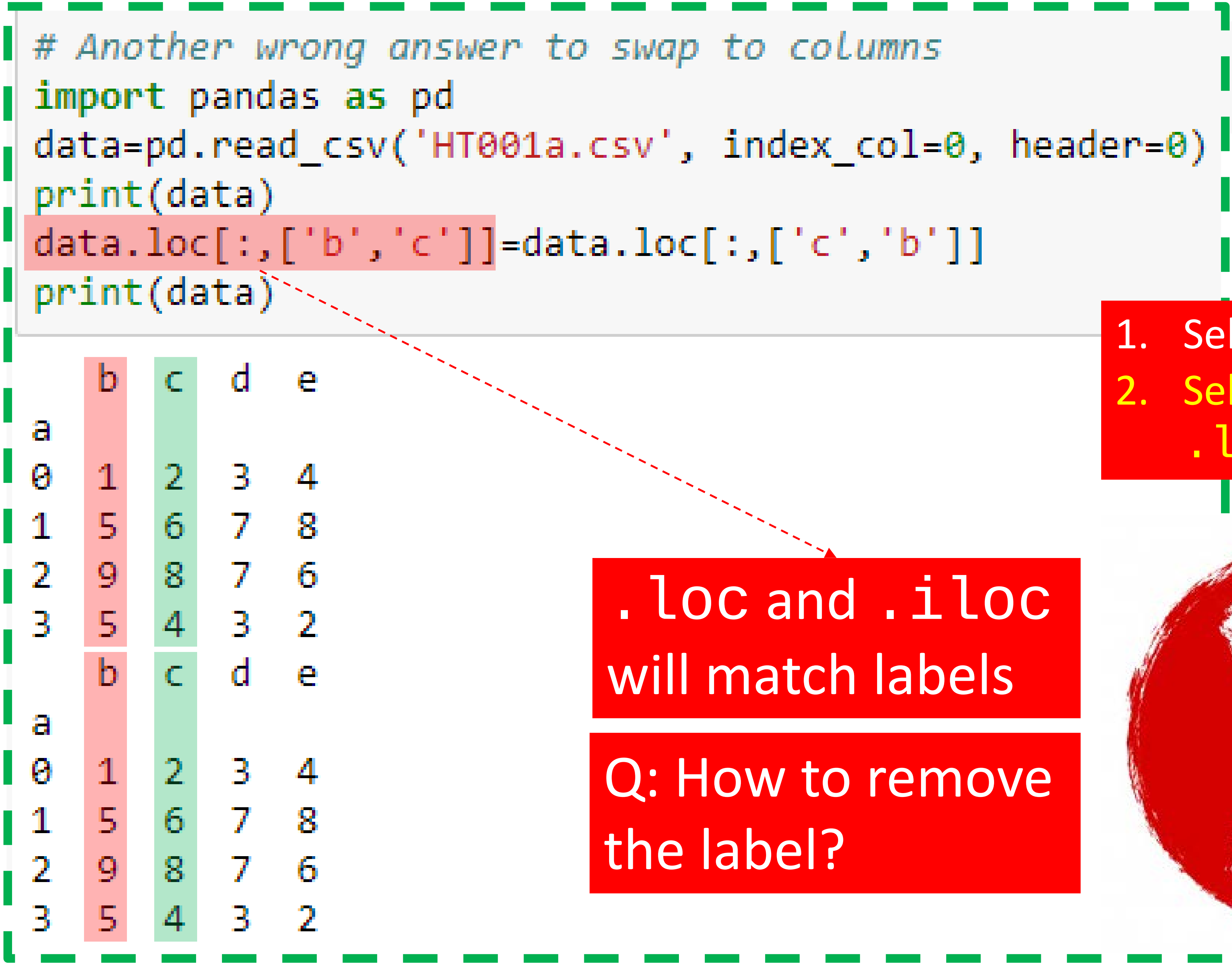
```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
   b  c  d  e
a
0  2  1  3  4
1  6  5  7  8
2  8  9  7  6
3  4  5  3  2
```

131

[ ] fancy indexing does not match labels.

✓

Interesting...



```
# Another wrong answer to swap to columns
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
data.loc[:,['b','c']]=data.loc[:,['c','b']]
print(data)
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
```

1. Select data via [ ]
2. Select data via .loc or .iloc

.loc and .iloc will match labels

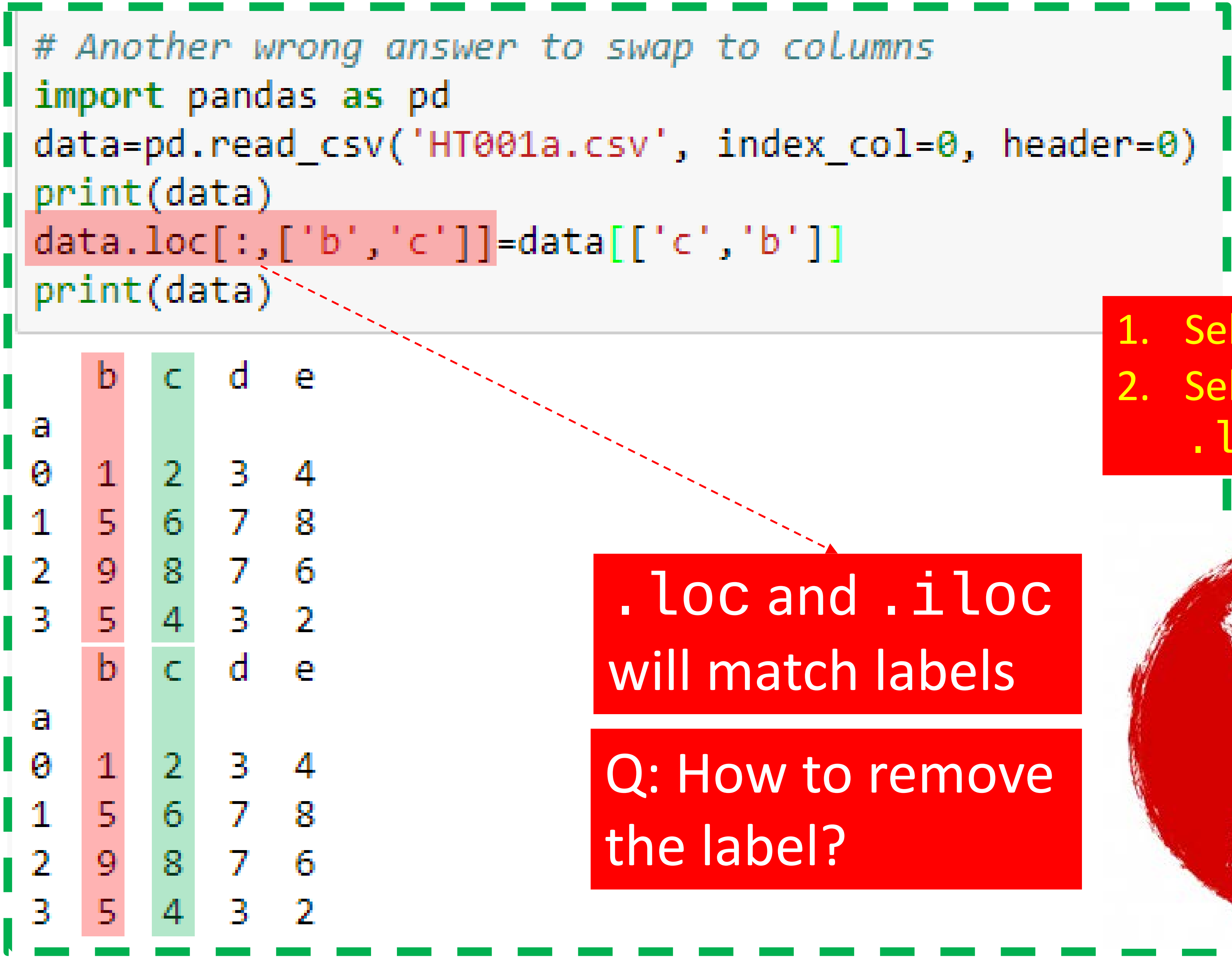
Q: How to remove the label?

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
# Another wrong answer to swap to columns
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
data.loc[:,['b','c']]=data[['c','b']]
print(data)
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
```

1. Select data via [ ]
2. Select data via .loc or .iloc

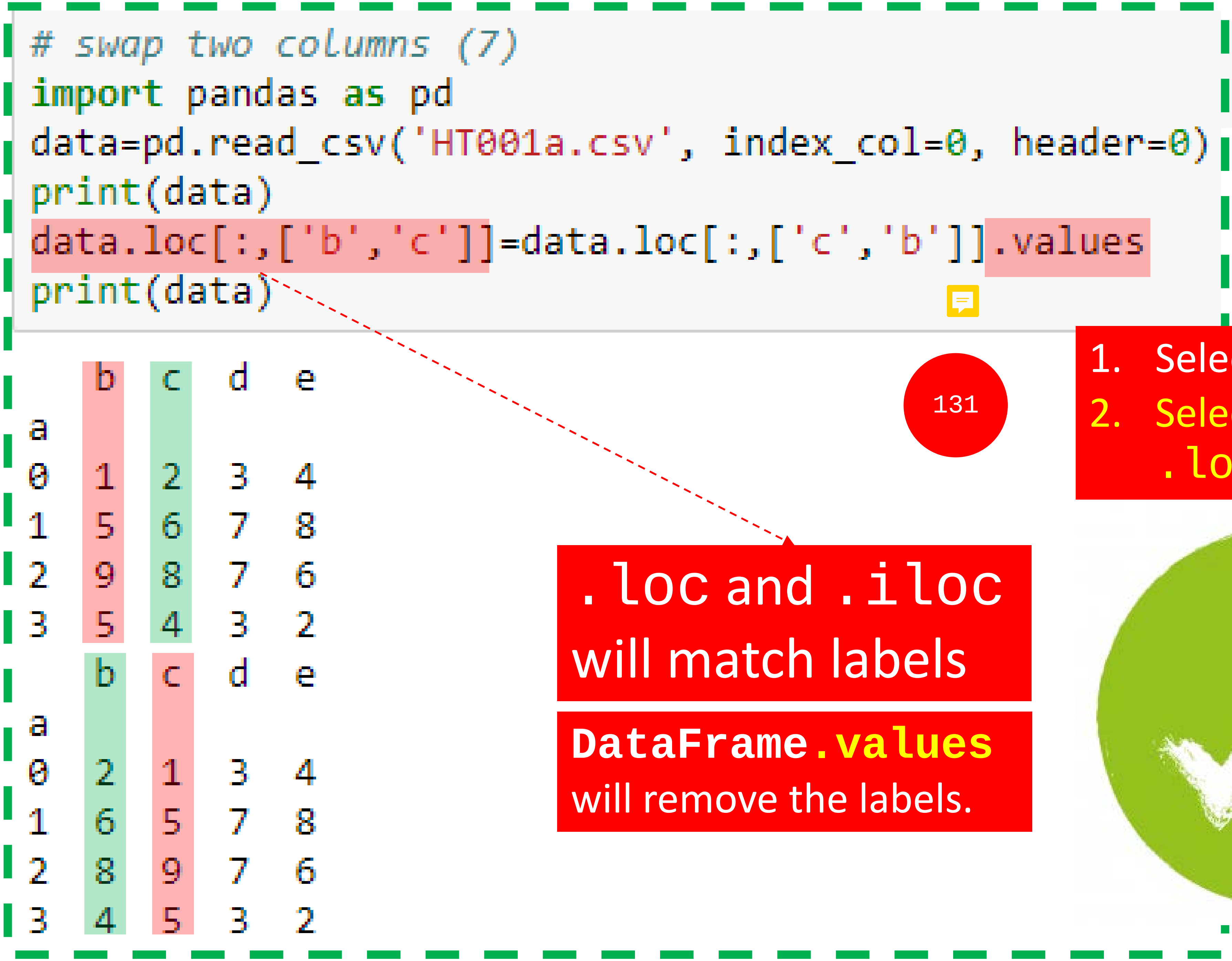
.loc and .iloc will match labels

Q: How to remove the label?

Slide
240/307

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
# swap two columns (7)
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
data.loc[:,['b','c']]=data.loc[:,['c','b']].values
print(data)
```

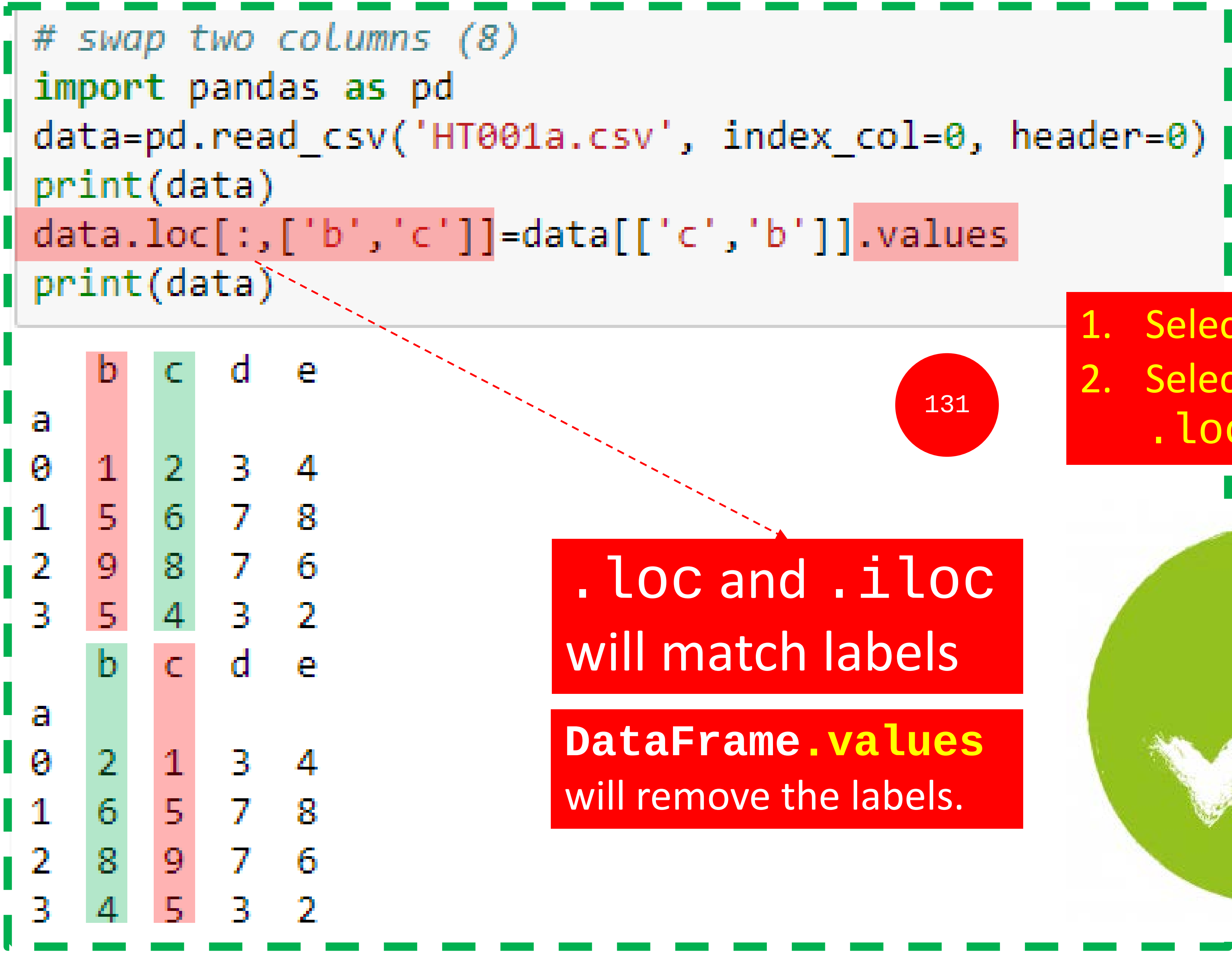
```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
   b  c  d  e
a
0  2  1  3  4
1  6  5  7  8
2  8  9  7  6
3  4  5  3  2
```

131

1. Select data via [ ]
2. Select data via `.loc` or `.iloc`

`.loc` and `.iloc` will match labels

`DataFrame.values` will remove the labels.



```
# swap two columns (8)
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
data.loc[:,['b','c']]=data[['c','b']].values
print(data)
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
   b  c  d  e
a
0  2  1  3  4
1  6  5  7  8
2  8  9  7  6
3  4  5  3  2
```

131

1. Select data via [ ]
2. Select data via `.loc` or `.iloc`

`.loc` and `.iloc` will match labels

`DataFrame.values` will remove the labels.

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Summary:

Use one command to swap two **columns** using column labels in `data`.

```
1. data['b'], data['c']=data['c'].copy(), data['b'].copy()
2. data['b'], data['c']=data['c'], data['b'].copy()
3. data[['b','c']]=data[['c','b']].copy()
4. data[['b','c']]=data.loc[:,['c','b']].copy()
5. data.loc[:,['b','c']]=data.loc[:,['c','b']].values.copy()
6. data.loc[:,['b','c']]=data[['c','b']].values.copy()
```

131

Dr. Z: ⇨Better to use `.copy()`.

How about rows?

Use one command to swap two **rows** in data.

132 133

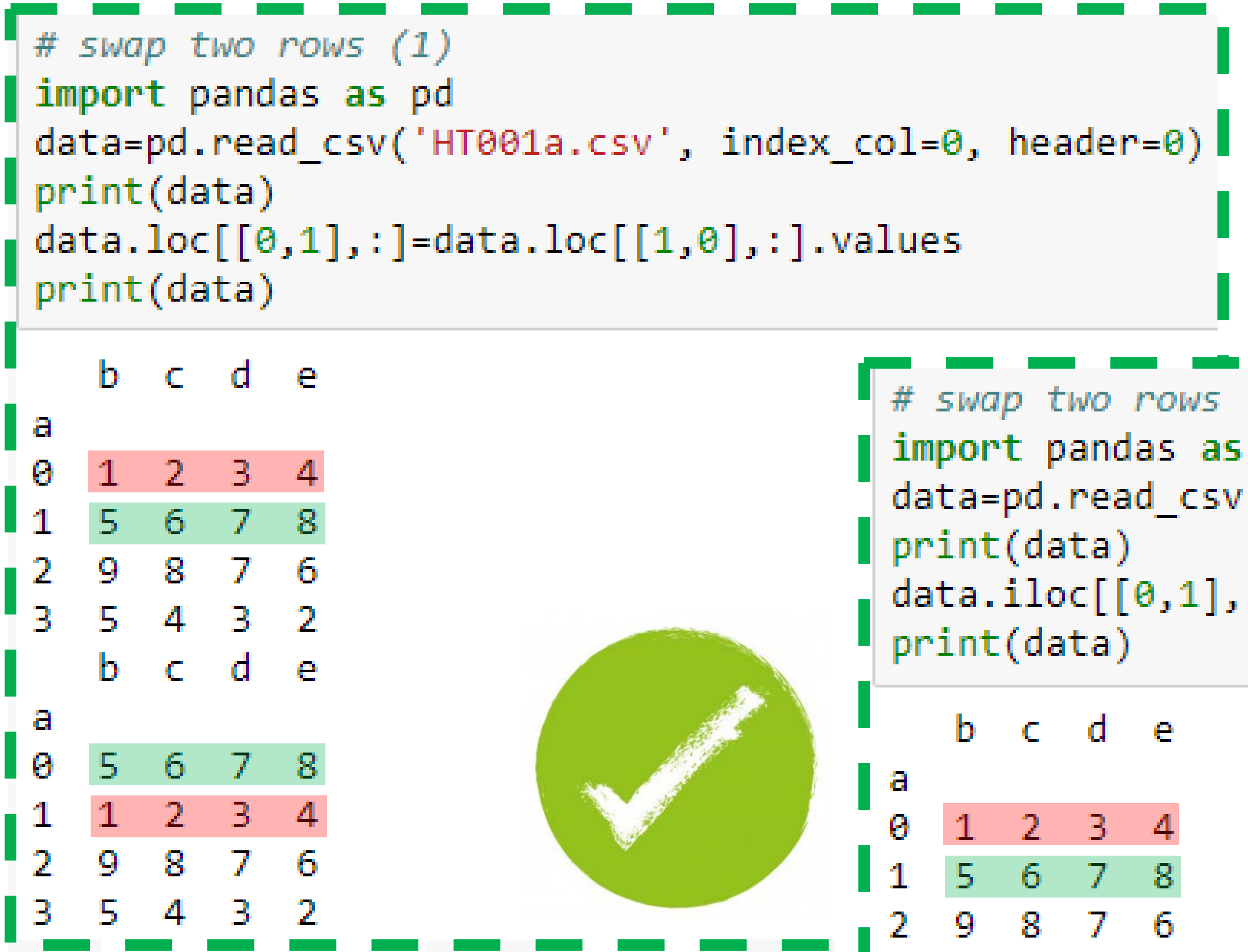
1. **Label-based indexing/slicing**

```
data.loc[[0,1],:]=data.loc[[1,0],:].values.copy()
```

2. **Position-based indexing/slicing**

```
data.iloc[[0,1],:]=data.iloc[[1,0],:].values.copy()
```

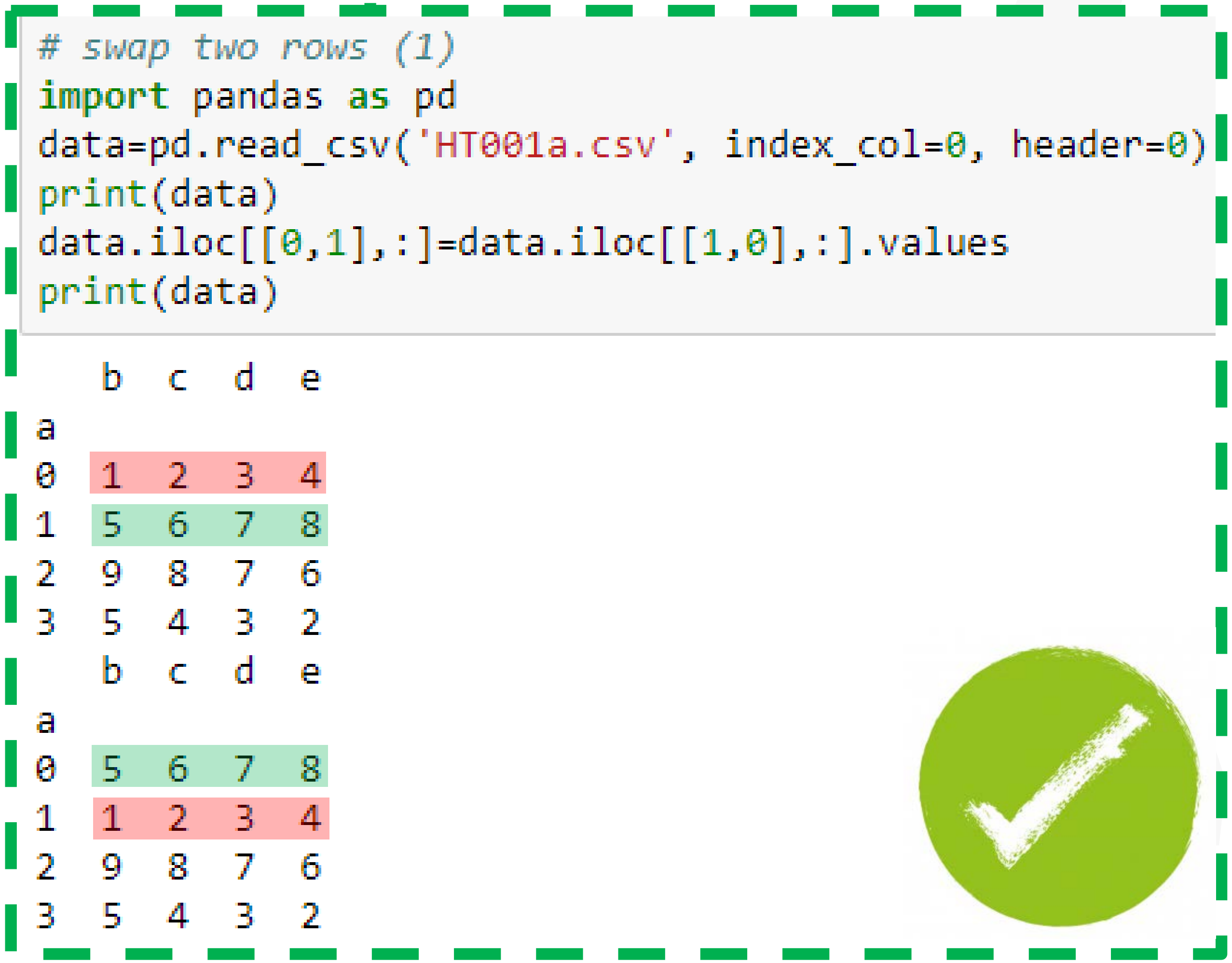
Dr. Z: ⇨Better to use **.copy()**.



```
# swap two rows (1)
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
data.loc[[0,1],:]=data.loc[[1,0],:].values
print(data)
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
   b  c  d  e
a
0  5  6  7  8
1  1  2  3  4
2  9  8  7  6
3  5  4  3  2
```

132



```
# swap two rows (1)
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
data.iloc[[0,1],:]=data.iloc[[1,0],:].values
print(data)
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
   b  c  d  e
a
0  5  6  7  8
1  1  2  3  4
2  9  8  7  6
3  5  4  3  2
```

133

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

HT007: Swap two rows/columns in **data**

✓ label-based indexing

Use one command

```
✓ data(:,{'b','c'})=data(:,{'c','b'})
✓ data{:,{'b','c'}}=data{:,{'c','b'}}
```

134

```
× data(:,{'b','c'})=data{:,{'c','b'}}
```

Right hand side of an assignment into a table must be another table or a cell array.

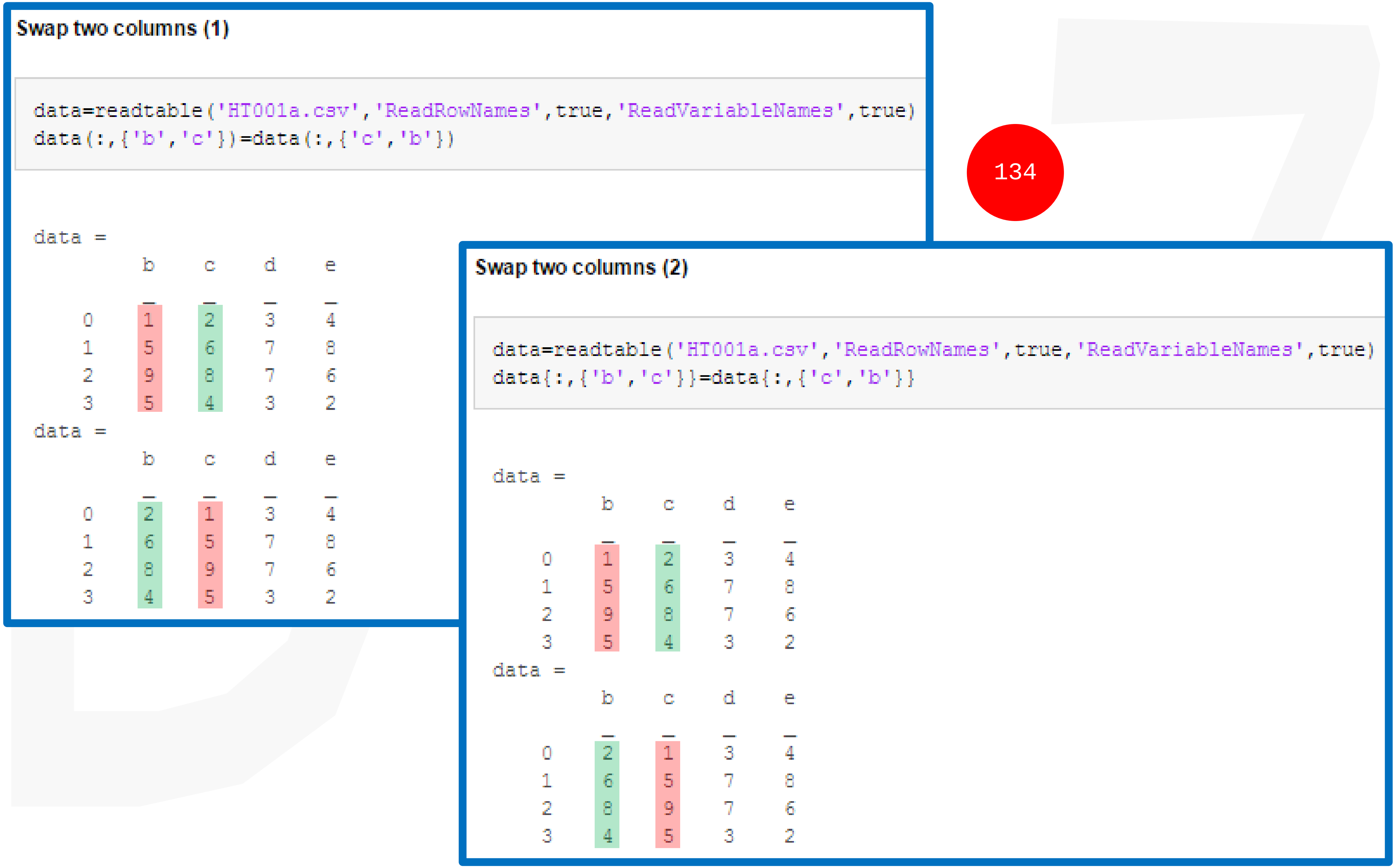
```
× data{:,{'b','c'}}=data(:,{'c','b'})
```

The following error occurred converting from table to double:

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



Swap two columns (1)

```
data=readtable('HT001a.csv','ReadRowNames',true,'ReadVariableNames',true)
data(:,{'b','c'})=data(:,{'c','b'})
```

```
data = 
          b    c    d    e
          _    _    _    _
    0     1    2    3    4
    1     5    6    7    8
    2     9    8    7    6
    3     5    4    3    2
data = 
          b    c    d    e
          _    _    _    _
    0     2    1    3    4
    1     6    5    7    8
    2     8    9    7    6
    3     4    5    3    2
```

Swap two columns (2)

```
data=readtable('HT001a.csv','ReadRowNames',true,'ReadVariableNames',true)
data{:,{'b','c'}}=data{:,{'c','b'}}
```

```
data = 
          b    c    d    e
          _    _    _    _
    0     1    2    3    4
    1     5    6    7    8
    2     9    8    7    6
    3     5    4    3    2
data = 
          b    c    d    e
          _    _    _    _
    0     2    1    3    4
    1     6    5    7    8
    2     8    9    7    6
    3     4    5    3    2
```

134

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

How about rows?

✓ label-based indexing

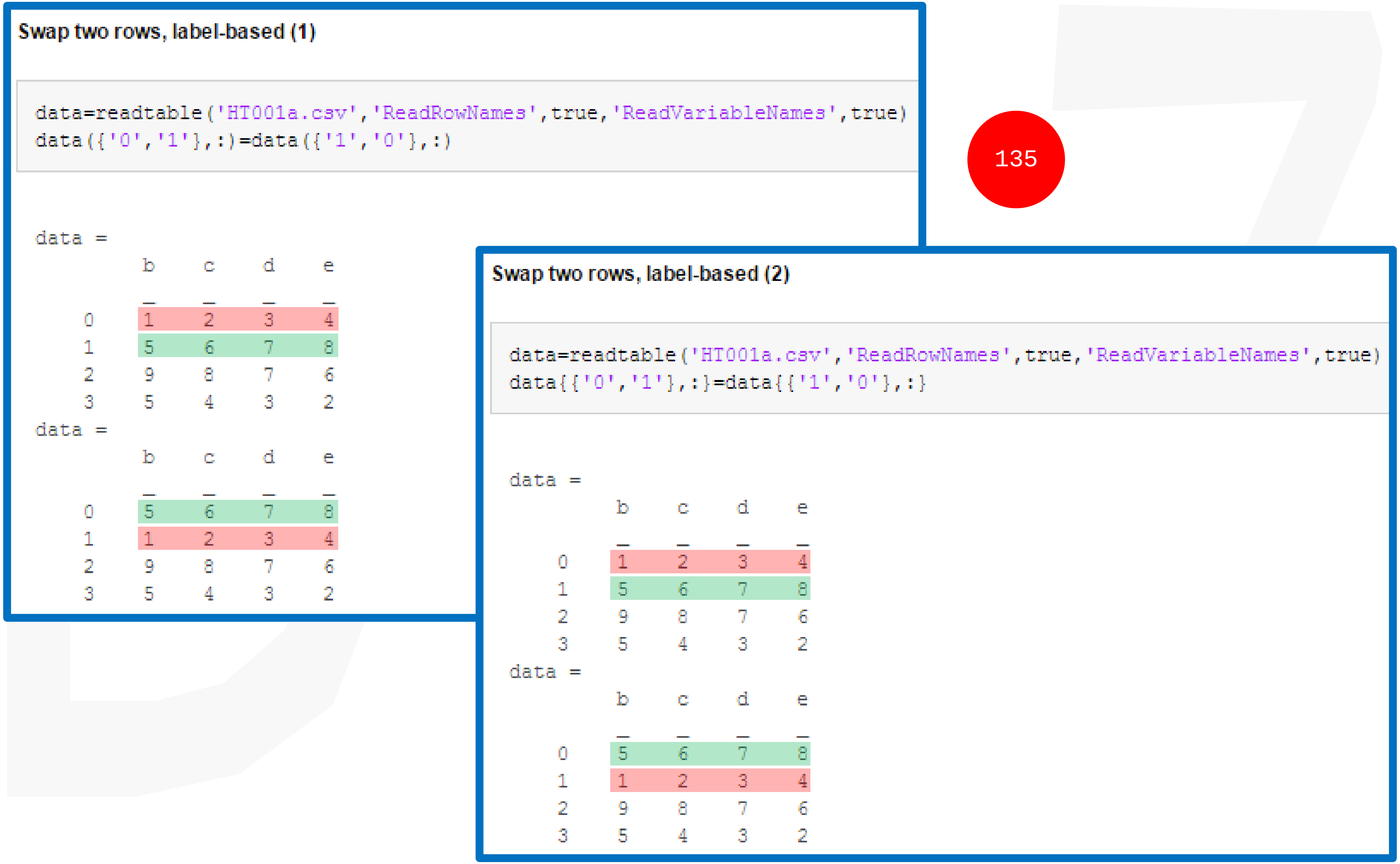
Use one command

```
✓ data({'0','1'},:)=data({'1','0'},:)
✓ data{{'0','1'},:}=data{{'1','0'},:}
```

135

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



Swap two rows, label-based (1)

```
data=readtable('HT001a.csv','ReadRowNames',true,'ReadVariableNames',true)
data({'0','1'},:)=data({'1','0'},:)
```

```
data =
          b    c    d    e

          _    _    _    _
    0     1    2    3    4
    1     5    6    7    8
    2     9    8    7    6
    3     5    4    3    2
data =
          b    c    d    e

          _    _    _    _
    0     5    6    7    8
    1     1    2    3    4
    2     9    8    7    6
    3     5    4    3    2
```

135

Swap two rows, label-based (2)

```
data=readtable('HT001a.csv','ReadRowNames',true,'ReadVariableNames',true)
data{{'0','1'},:}=data{{'1','0'},:}
```

```
data =
          b    c    d    e

          _    _    _    _
    0     1    2    3    4
    1     5    6    7    8
    2     9    8    7    6
    3     5    4    3    2
data =
          b    c    d    e

          _    _    _    _
    0     5    6    7    8
    1     1    2    3    4
    2     9    8    7    6
    3     5    4    3    2
```

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

How about using position-based indexing?

- Swap two columns

✓ `data(:,[1,2])=data(:,[2,1])`
✓ `data{:,[1,2]}=data{:,[2,1]}`

136

- Swap two rows

✓ `data([1,2],:)=data([2,1],:)`
✓ `data{[1,2],:}=data{[2,1],:}`

137

Use one command

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

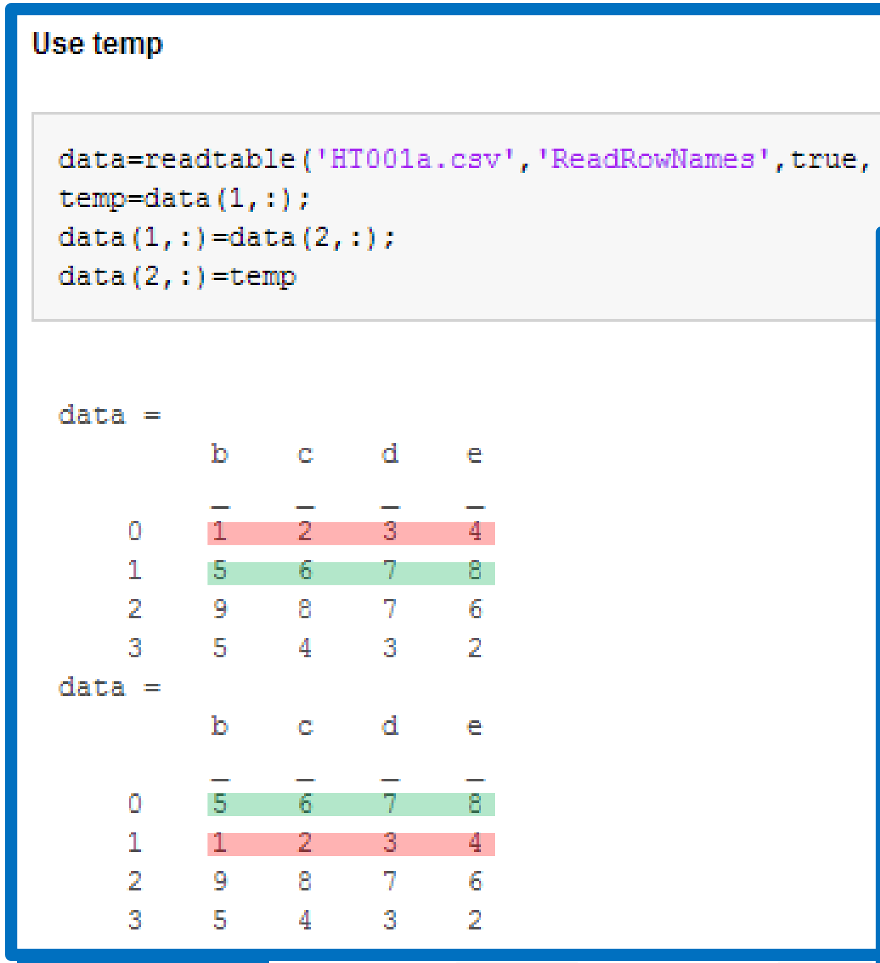
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Slide
250/307

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance



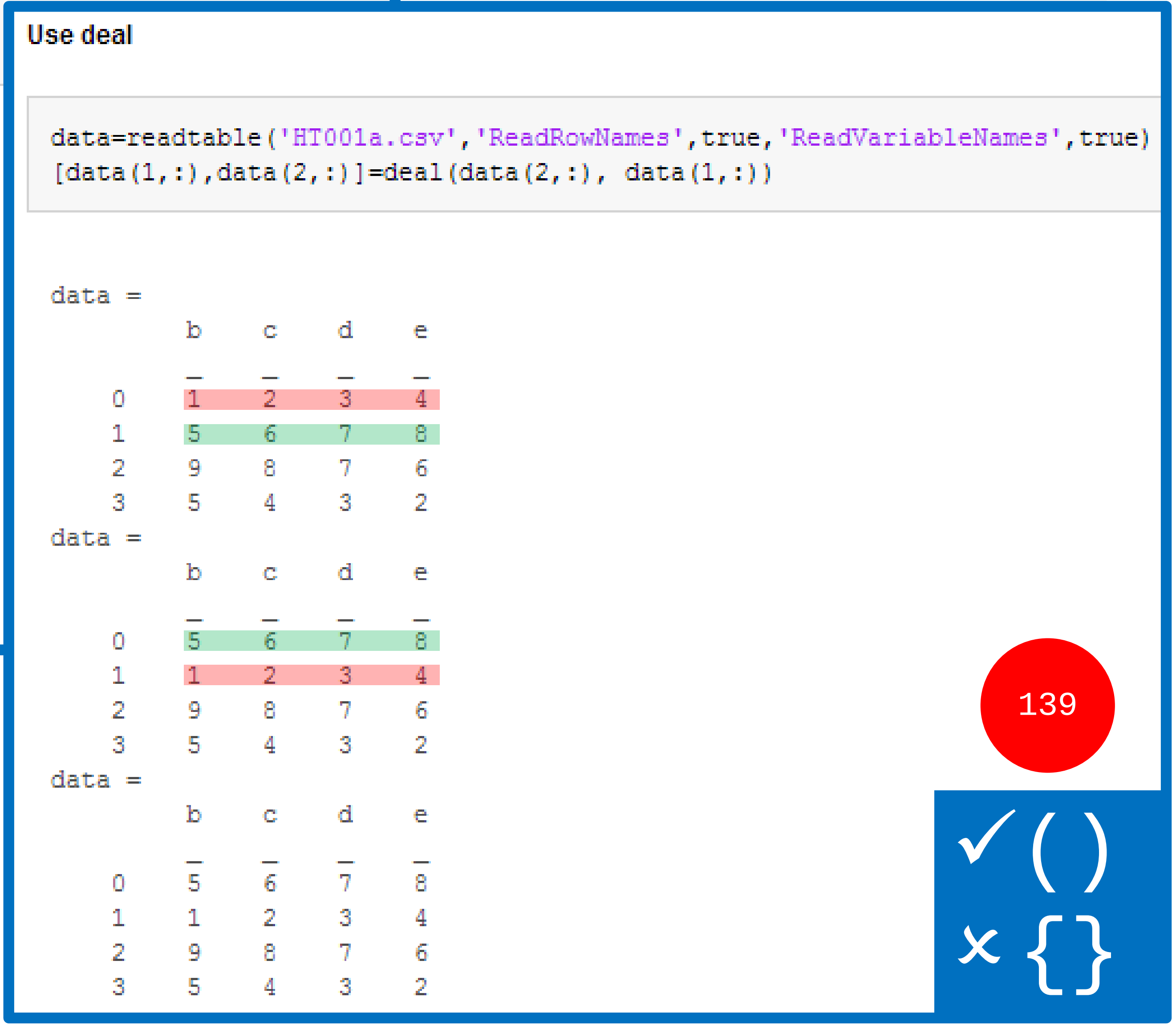
Use temp

```
data=readtable('HT001a.csv','ReadRowNames',true,'ReadVariableNames',true)
temp=data(1,:);
data(1,:)=data(2,:);
data(2,:)=temp
```

```
data =
          b    c    d    e
          _    _    _    _
     0    1    2    3    4
     1    5    6    7    8
     2    9    8    7    6
     3    5    4    3    2
data =
          b    c    d    e
          _    _    _    _
     0    5    6    7    8
     1    1    2    3    4
     2    9    8    7    6
     3    5    4    3    2
```

✓ ( )
✓ {}

138



Use deal

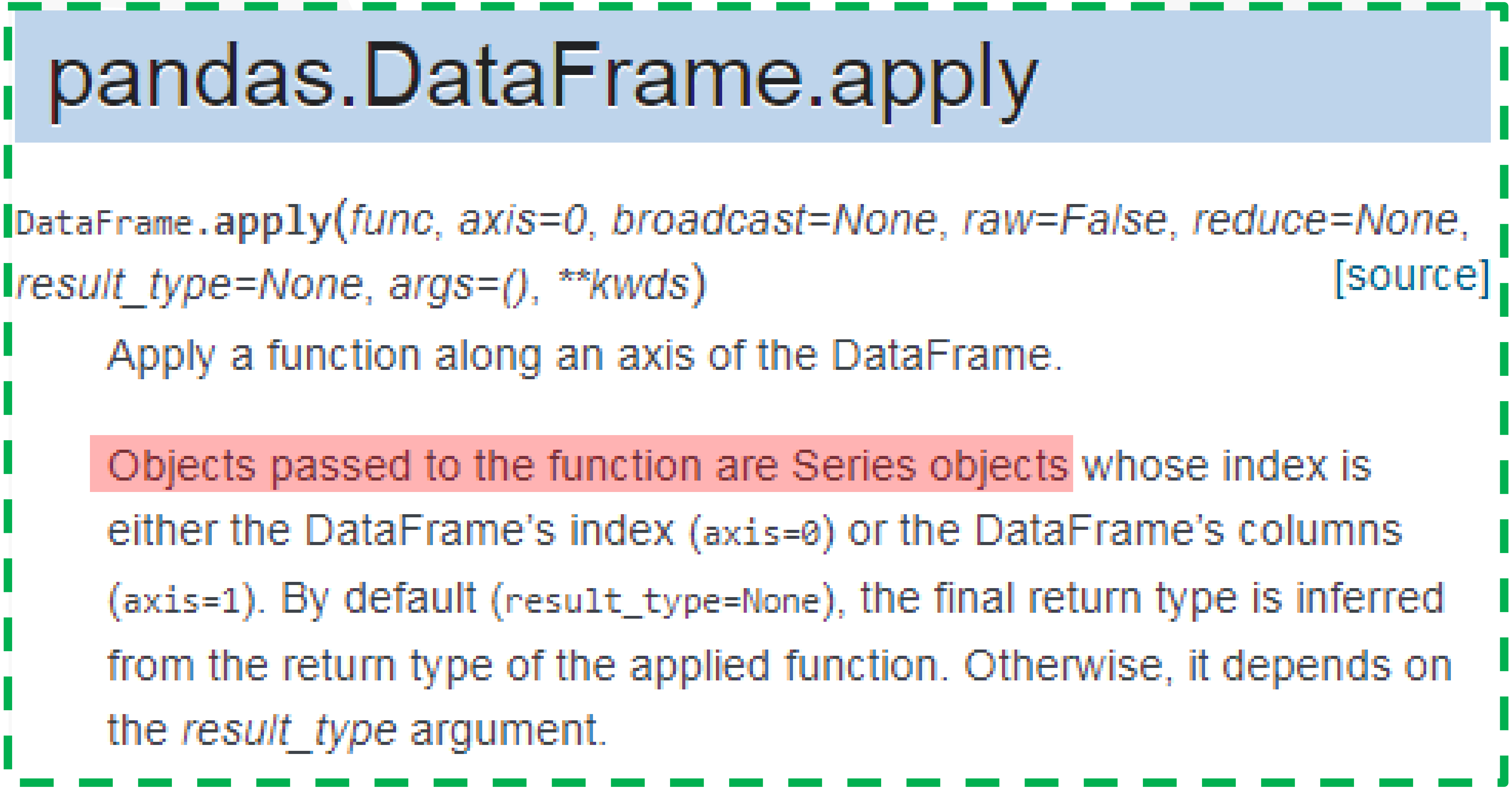
```
data=readtable('HT001a.csv','ReadRowNames',true,'ReadVariableNames',true)
[data(1,:),data(2,:)]=deal(data(2,:), data(1,:))
```

```
data =
          b    c    d    e
          _    _    _    _
     0    1    2    3    4
     1    5    6    7    8
     2    9    8    7    6
     3    5    4    3    2
data =
          b    c    d    e
          _    _    _    _
     0    5    6    7    8
     1    1    2    3    4
     2    9    8    7    6
     3    5    4    3    2
data =
          b    c    d    e
          _    _    _    _
     0    5    6    7    8
     1    1    2    3    4
     2    9    8    7    6
     3    5    4    3    2
```

139

✓ ( )
× {}

HT008: Apply a function to each row/column of `data`



pandas.DataFrame.apply

`DataFrame.apply`(*func*, *axis=0*, *broadcast=None*, *raw=False*, *reduce=None*, *result_type=None*, *args=()*, ***kwds*) [source]

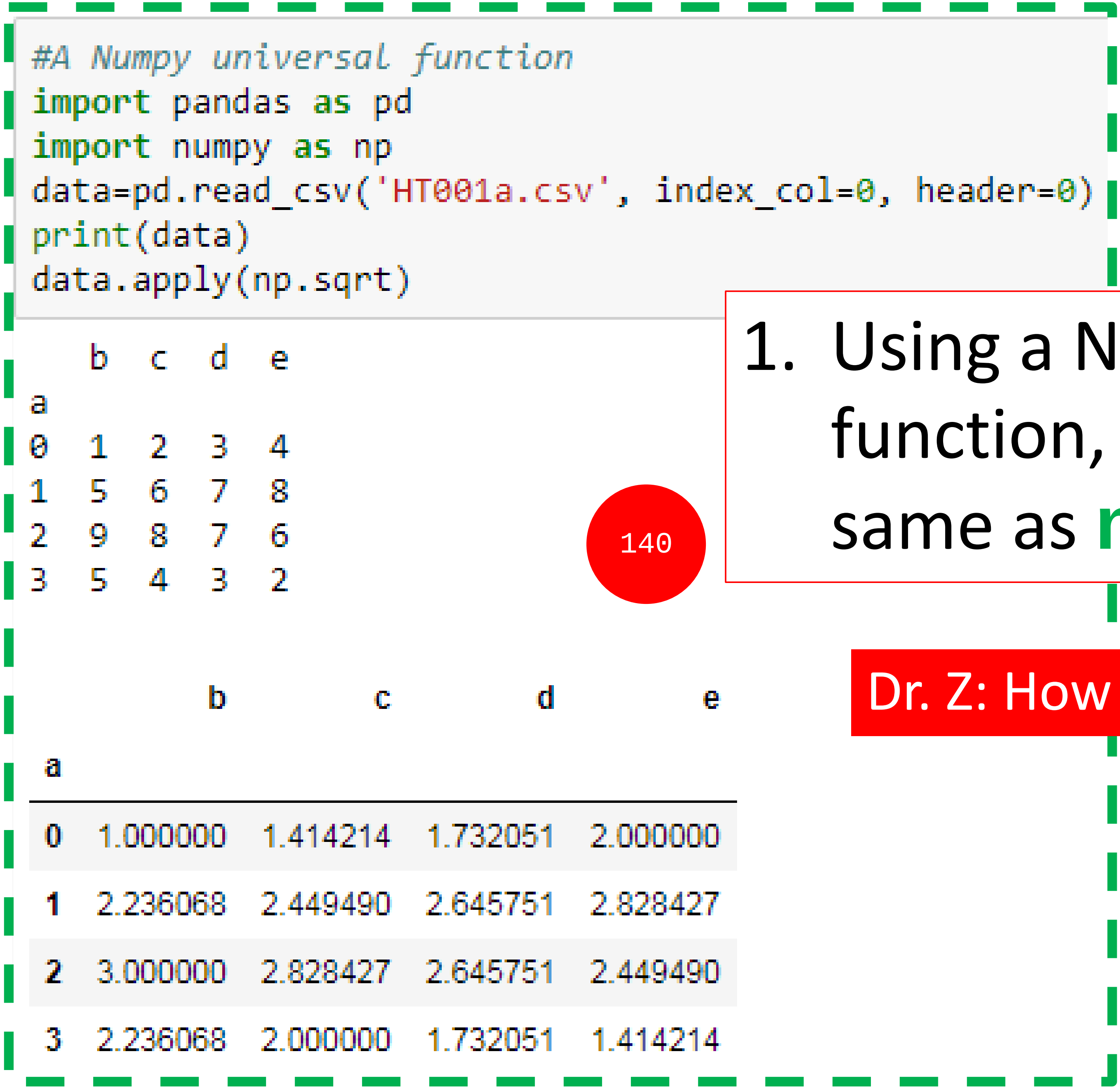
Apply a function along an axis of the DataFrame.

Objects passed to the function are Series objects whose index is either the DataFrame's index (`axis=0`) or the DataFrame's columns (`axis=1`). By default (`result_type=None`), the final return type is inferred from the return type of the applied function. Otherwise, it depends on the *result_type* argument.

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
#A Numpy universal function
import pandas as pd
import numpy as np
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
data.apply(np.sqrt)
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
```

	b	c	d	e
a				
0	1.000000	1.414214	1.732051	2.000000
1	2.236068	2.449490	2.645751	2.828427
2	3.000000	2.828427	2.645751	2.449490
3	2.236068	2.000000	1.732051	1.414214

140

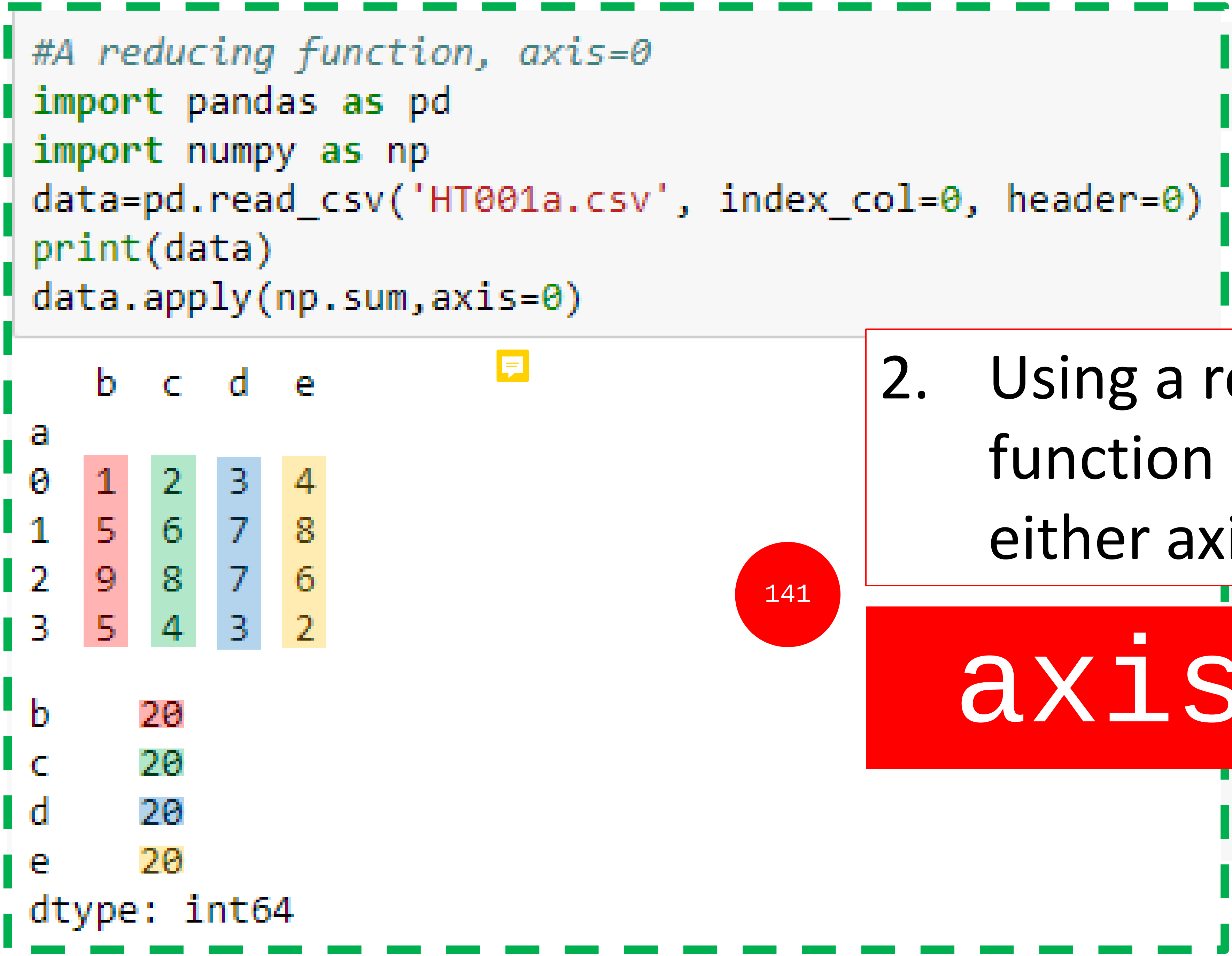
1. Using a Numpy universal function, **fun**, (in the case same as **np.fun(data)**)

Dr. Z: How about math.sqrt?

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
#A reducing function, axis=0
import pandas as pd
import numpy as np
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
data.apply(np.sum,axis=0)
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2

b    20
c    20
d    20
e    20
dtype: int64
```

141

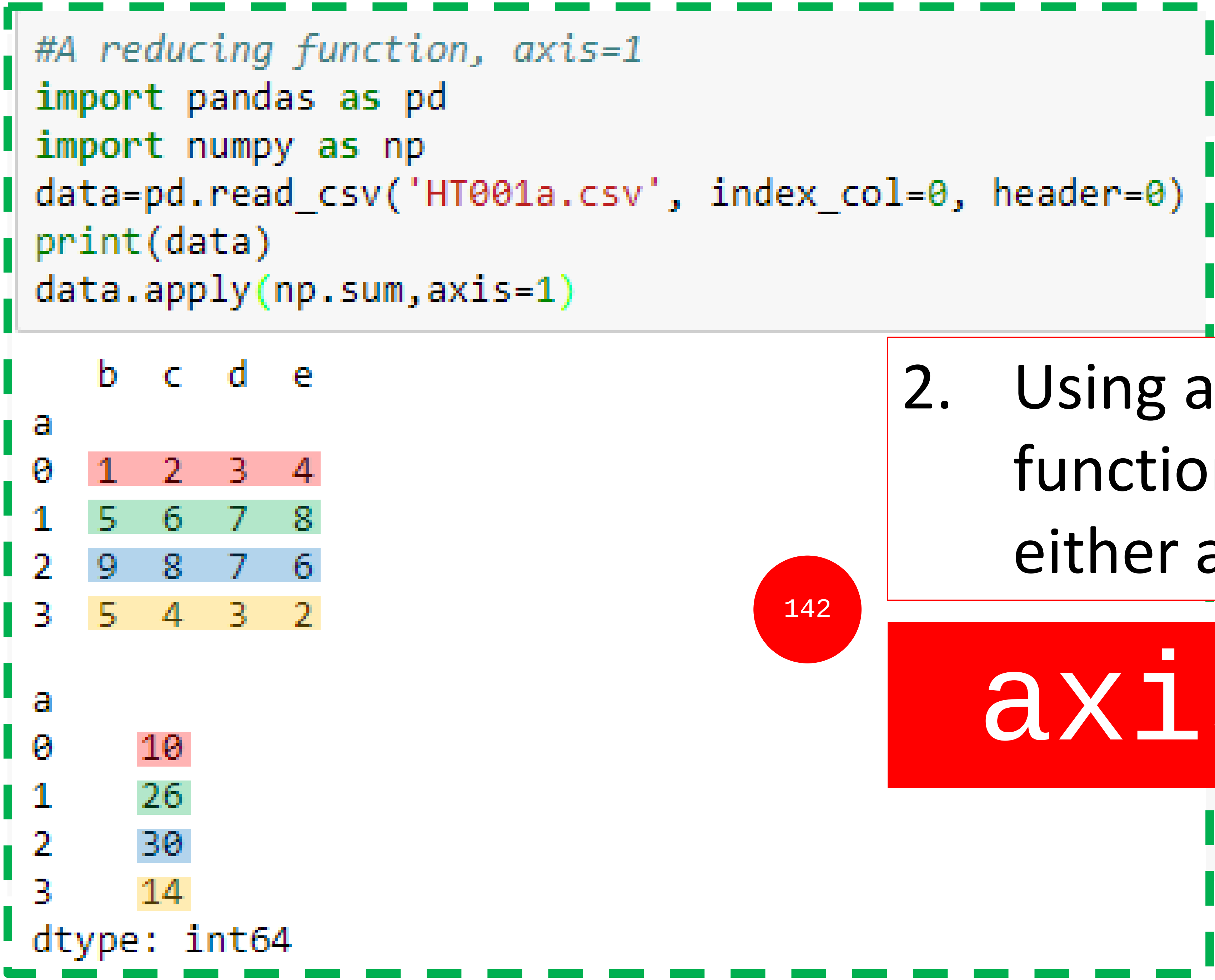
2. Using a reducing function on either axis

axis=0

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
#A reducing function, axis=1
import pandas as pd
import numpy as np
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
data.apply(np.sum,axis=1)
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2

a
0    10
1    26
2    30
3    14
dtype: int64
```

142

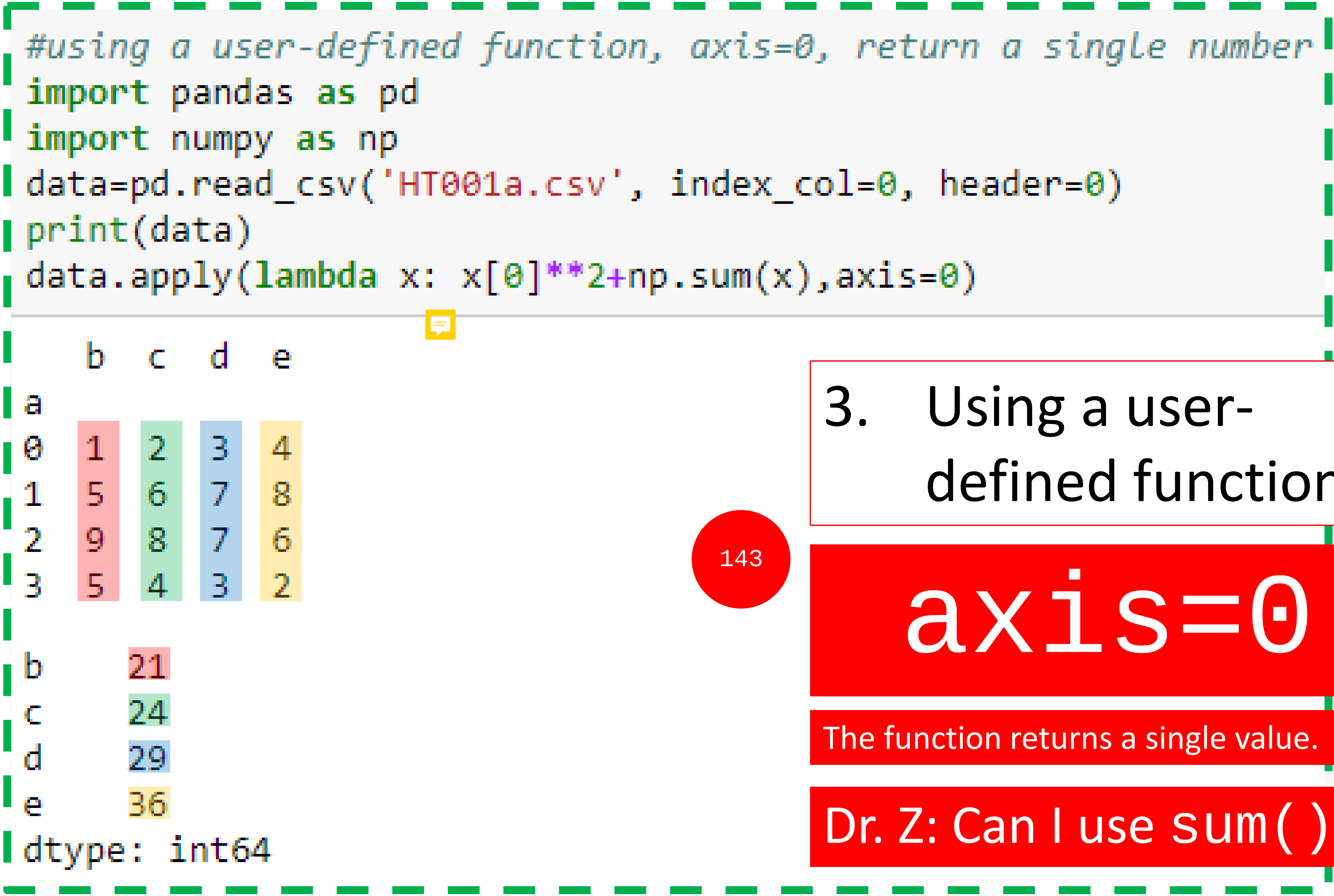
2. Using a reducing function on either axis

axis=1

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance



```
#using a user-defined function, axis=0, return a single number
import pandas as pd
import numpy as np
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
data.apply(lambda x: x[0]**2+np.sum(x),axis=0)
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2

b    21
c    24
d    29
e    36
dtype: int64
```

143

3. Using a user-defined function

axis=0

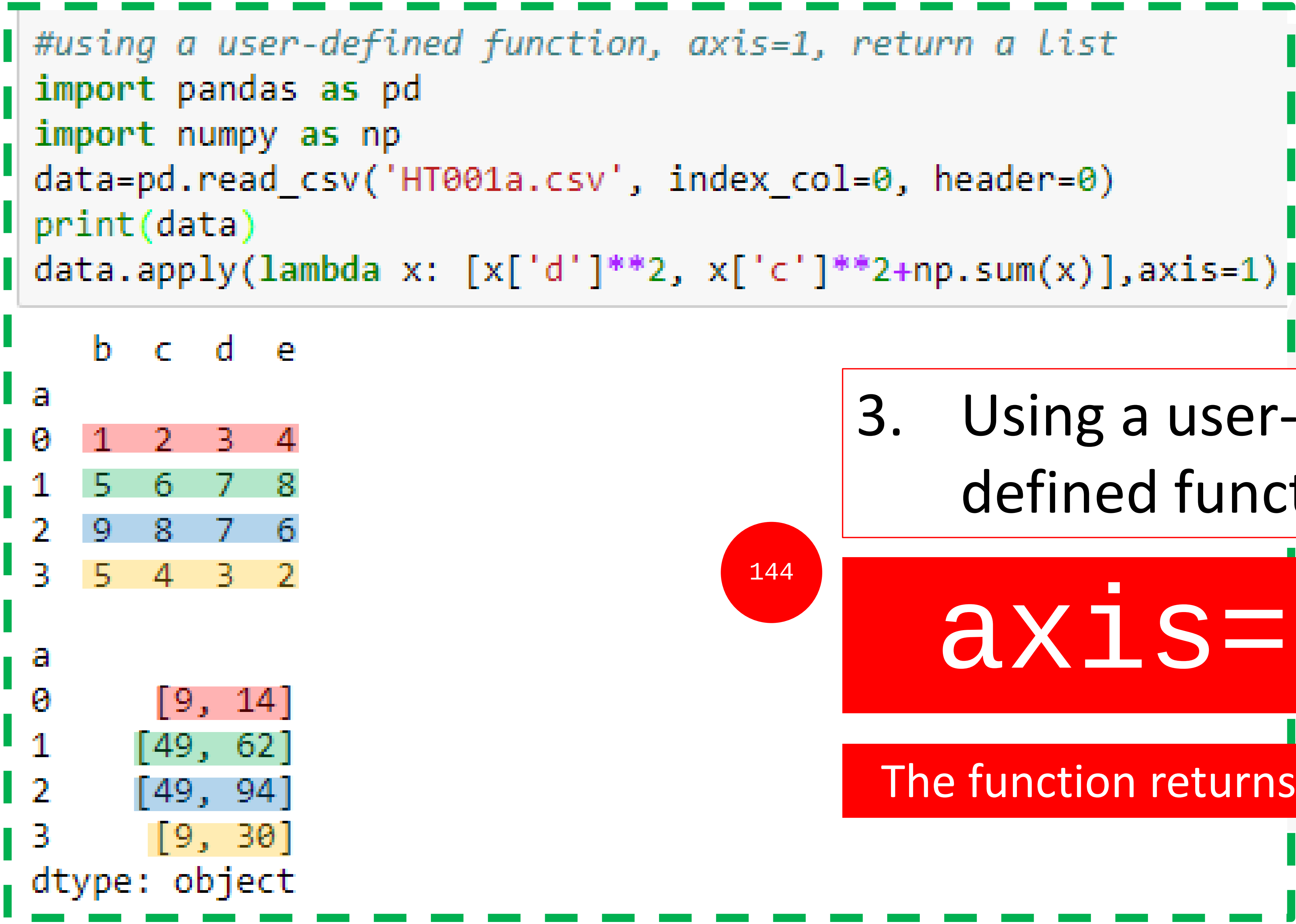
The function returns a single value.

Dr. Z: Can I use sum( )?

Slide
256/307

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
#using a user-defined function, axis=1, return a list
import pandas as pd
import numpy as np
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
data.apply(lambda x: [x['d']**2, x['c']**2+np.sum(x)],axis=1)
```

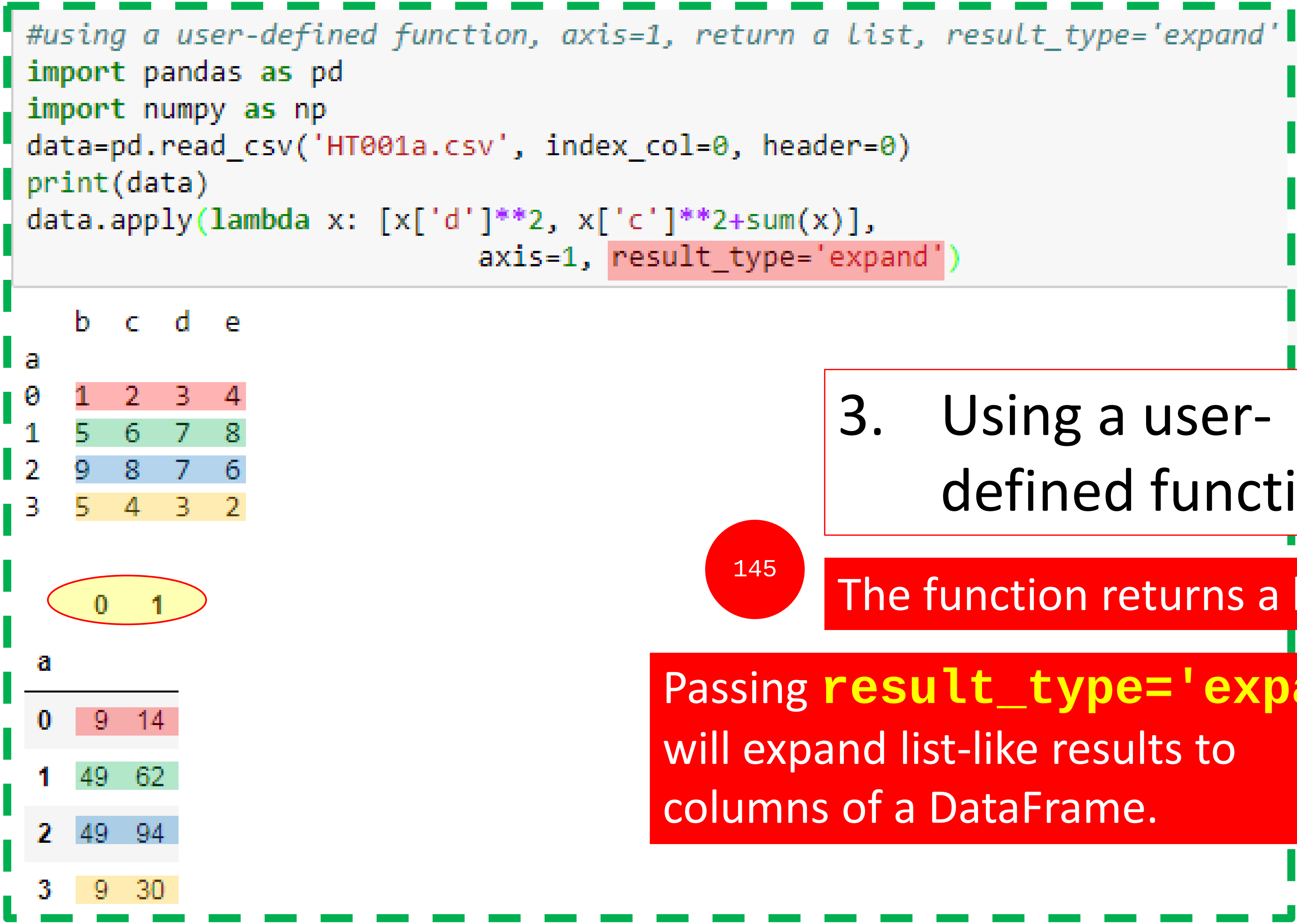
```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2

a
0     [9, 14]
1    [49, 62]
2    [49, 94]
3     [9, 30]
dtype: object
```

3. Using a user-defined function

144

axis=1

The function returns a list.



```
#using a user-defined function, axis=1, return a list, result_type='expand'
import pandas as pd
import numpy as np
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
data.apply(lambda x: [x['d']**2, x['c']**2+sum(x)],
                                axis=1, result_type='expand')
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
```

```
    0   1
a
0   9  14
1  49  62
2  49  94
3   9  30
```

3. Using a user-defined function

145

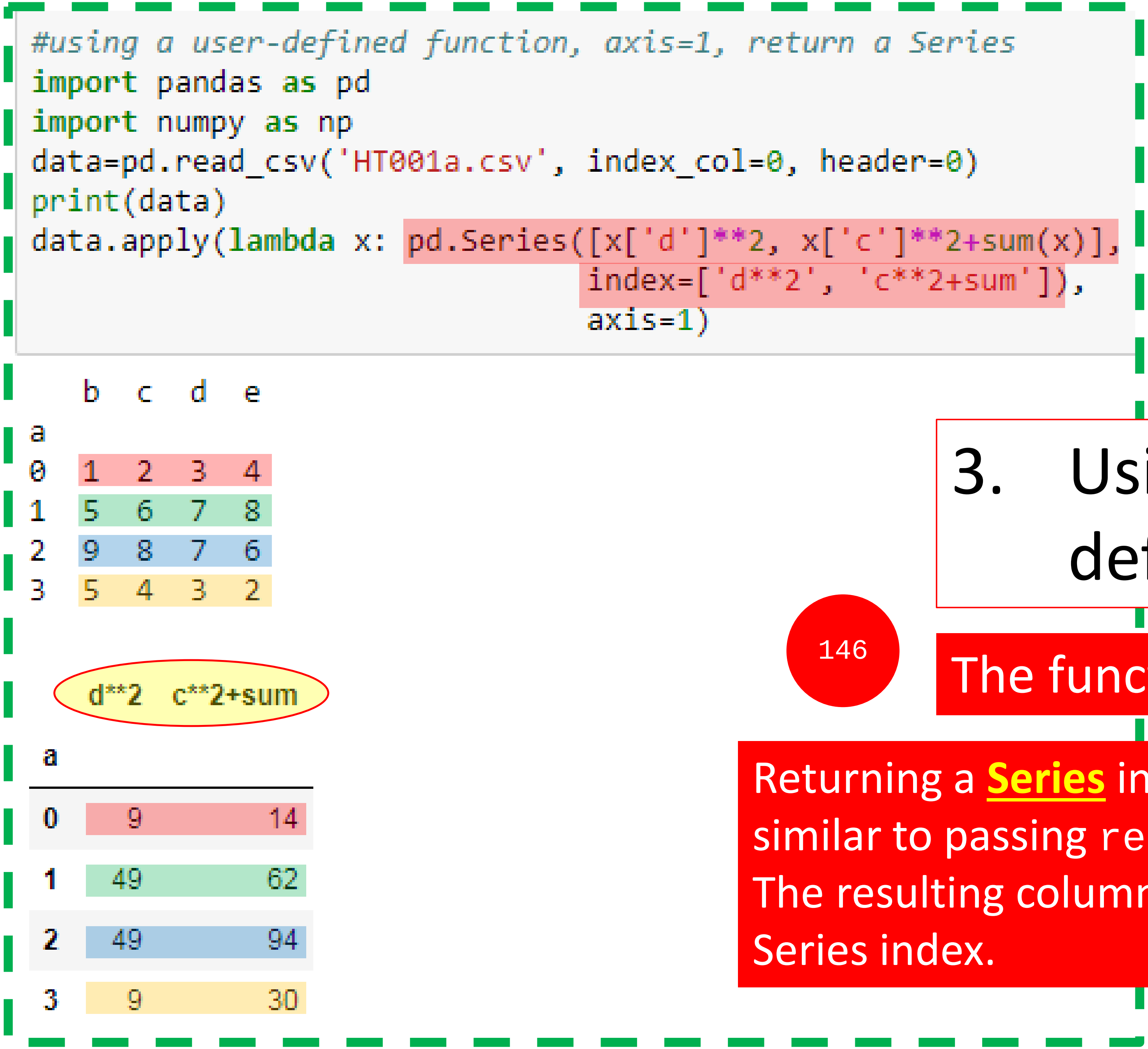
The function returns a list.

Passing **result_type='expand'** will expand list-like results to columns of a DataFrame.

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance



```
#using a user-defined function, axis=1, return a Series
import pandas as pd
import numpy as np
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
data.apply(lambda x: pd.Series([x['d']**2, x['c']**2+sum(x)],
                               index=['d**2', 'c**2+sum']),
                               axis=1)
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
```

```
   d**2  c**2+sum
a
0     9        14
1    49        62
2    49        94
3     9        30
```

3. Using a user-defined function

146

The function returns a list.

Returning a **Series** inside the function is similar to passing `result_type='expand'`. The resulting column names will be the Series index.

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

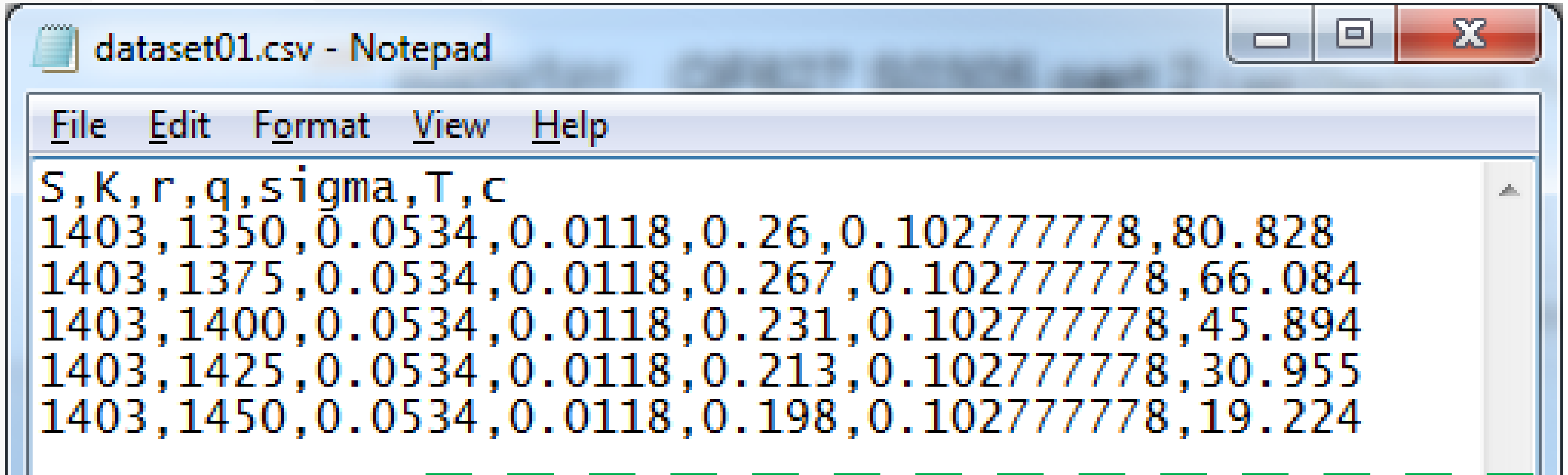
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

SMU SINGAPORE MANAGEMENT UNIVERSITY

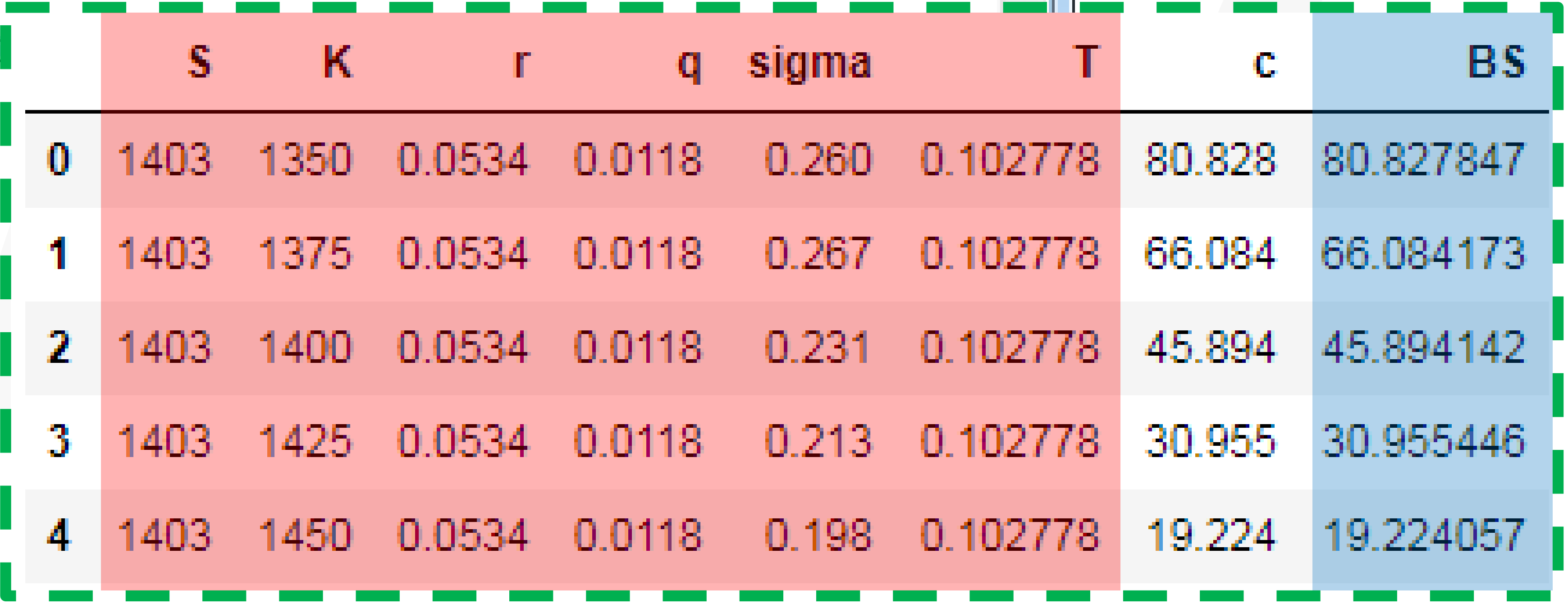
QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Homework Question:



```
S,K,r,q,sigma,T,c
1403,1350,0.0534,0.0118,0.26,0.102777778,80.828
1403,1375,0.0534,0.0118,0.267,0.102777778,66.084
1403,1400,0.0534,0.0118,0.231,0.102777778,45.894
1403,1425,0.0534,0.0118,0.213,0.102777778,30.955
1403,1450,0.0534,0.0118,0.198,0.102777778,19.224
```



	S	K	r	q	sigma	T	c	BS
0	1403	1350	0.0534	0.0118	0.260	0.102778	80.828	80.827847
1	1403	1375	0.0534	0.0118	0.267	0.102778	66.084	66.084173
2	1403	1400	0.0534	0.0118	0.231	0.102778	45.894	45.894142
3	1403	1425	0.0534	0.0118	0.213	0.102778	30.955	30.955446
4	1403	1450	0.0534	0.0118	0.198	0.102778	19.224	19.224057

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

1. **Use one command** with the Pandas library function **`pandas.read_csv`** to load data from the CSV file, `dataset01.csv`, using the first row as column names. Name the data as **`data`**.
2. Define a function, **`option_BS`**, which computes and returns the European call option price using the following formula:

$$c = S \cdot e^{-q \cdot T} \cdot \Phi(d_1) - K \cdot e^{-r \cdot T} \cdot \Phi(d_2)$$

where

$$d_1 = \frac{\ln\left(\frac{S}{K}\right) + \left(r - q + \frac{\sigma^2}{2}\right)T}{\sigma\sqrt{T}} \text{ and } d_2 = d_1 - \sigma\sqrt{T}$$

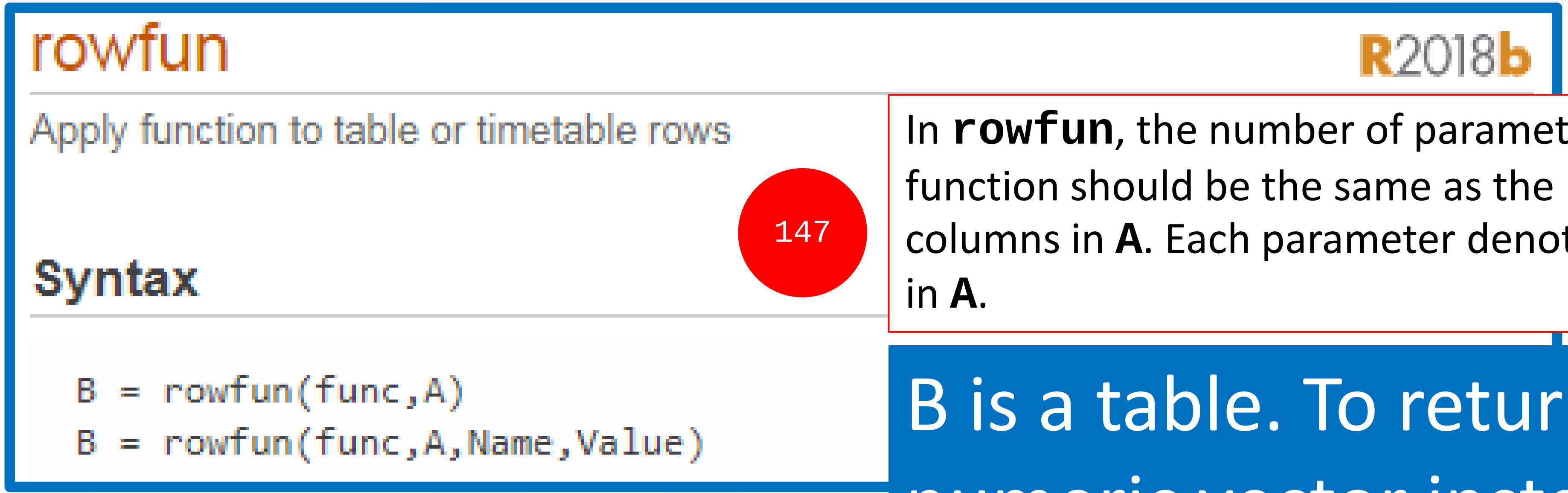
3. **Use one command** with the Pandas library function **`pandas.DataFrame.apply`** to compute the European call option price for each row of **`data`** and add the results to **`data`** as a new column, and name this column as **BS**.

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

HT008: Apply a function to each row/column of **`data`**



rowfun

R2018b

Apply function to table or timetable rows

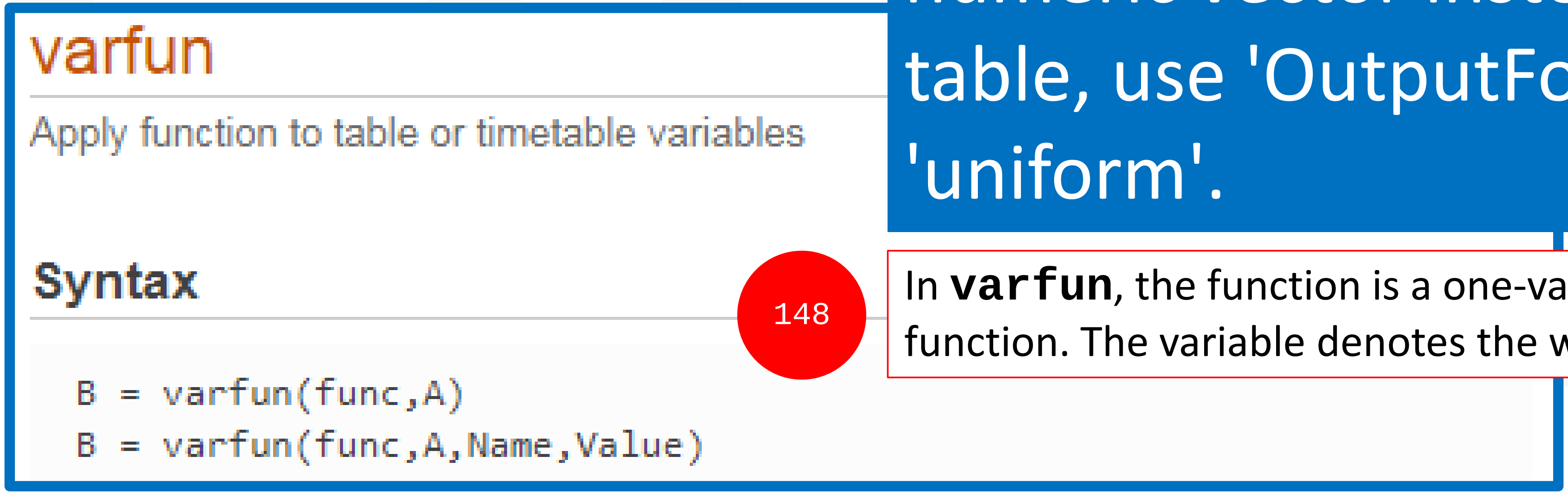
Syntax

```
B = rowfun(func,A)
B = rowfun(func,A,Name,Value)
```

147

In **`rowfun`**, the number of parameters in the function should be the same as the number of columns in **A**. Each parameter denotes a column in **A**.

B is a table. To return a numeric vector instead of a table, use 'OutputFormat', 'uniform'.



varfun

Apply function to table or timetable variables

Syntax

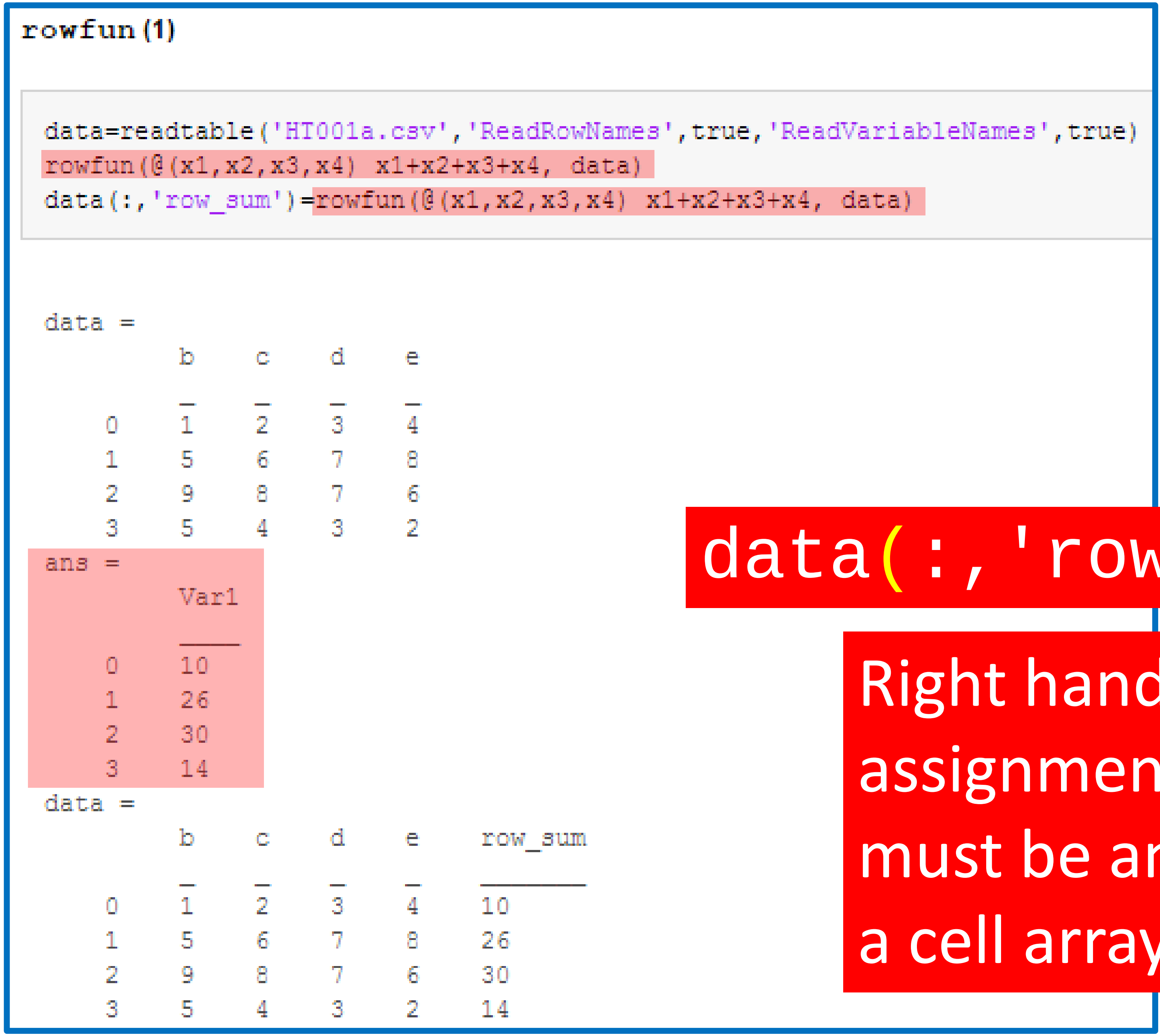
```
B = varfun(func,A)
B = varfun(func,A,Name,Value)
```

148

In **`varfun`**, the function is a one-variable function. The variable denotes the whole column.

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



rowfun (1)

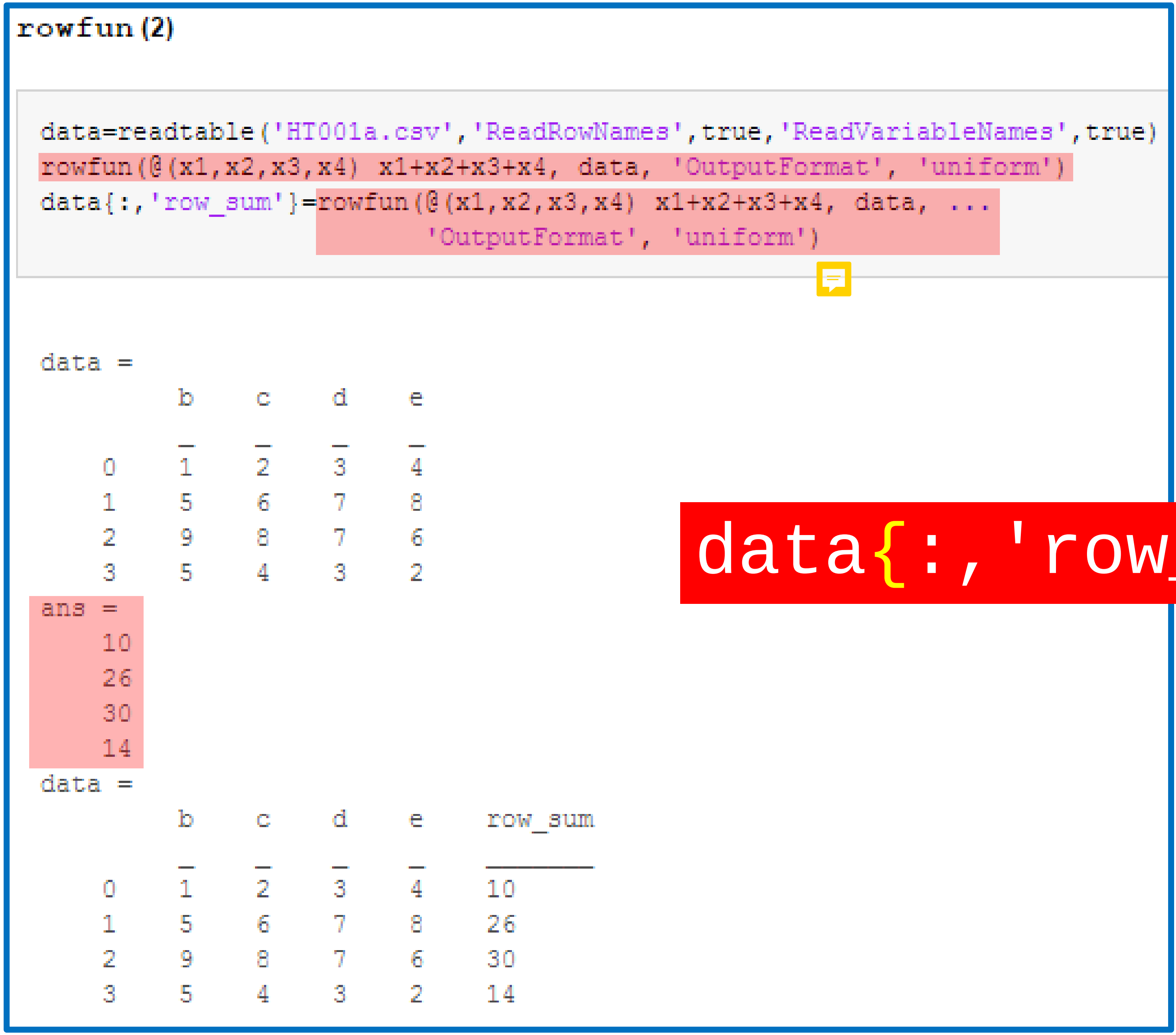
```
data=readtable('HT001a.csv','ReadRowNames',true,'ReadVariableNames',true)
rowfun(@(x1,x2,x3,x4) x1+x2+x3+x4, data)
data(:,'row_sum')=rowfun(@(x1,x2,x3,x4) x1+x2+x3+x4, data)
```

```
data =
             b    c    d    e

             _    _    _    _
        0    1    2    3    4
        1    5    6    7    8
        2    9    8    7    6
        3    5    4    3    2
ans =
             Var1

             ____
        0    10
        1    26
        2    30
        3    14
data =
             b    c    d    e    row_sum

             _    _    _    _    _______
        0    1    2    3    4    10
        1    5    6    7    8    26
        2    9    8    7    6    30
        3    5    4    3    2    14
```

147

`data(:,'row_sum')=`

Right hand side of an assignment into a table must be another table or a cell array.



rowfun (2)

```
data=readtable('HT001a.csv','ReadRowNames',true,'ReadVariableNames',true)
rowfun(@(x1,x2,x3,x4) x1+x2+x3+x4, data, 'OutputFormat', 'uniform')
data{:,'row_sum'}=rowfun(@(x1,x2,x3,x4) x1+x2+x3+x4, data, ...
                  'OutputFormat', 'uniform')
```

```
data =
         b    c    d    e

         _    _    _    _
    0    1    2    3    4
    1    5    6    7    8
    2    9    8    7    6
    3    5    4    3    2
ans =
   10
   26
   30
   14
data =
         b    c    d    e    row_sum

         _    _    _    _    _______
    0    1    2    3    4    10
    1    5    6    7    8    26
    2    9    8    7    6    30
    3    5    4    3    2    14
```

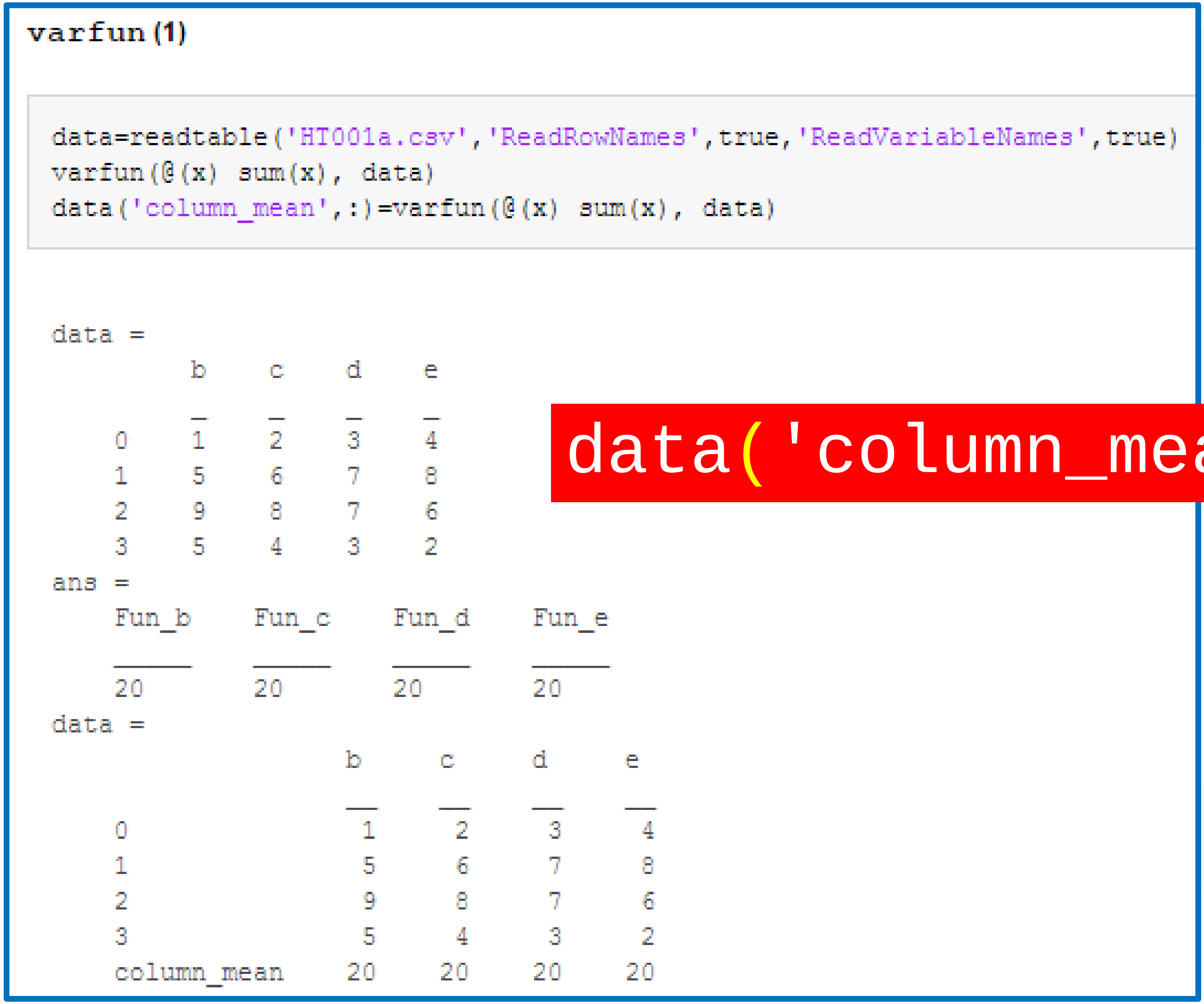
147

data{:,'row_sum'}=

SMU
SINGAPORE MANAGEMENT
UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



varfun (1)

```
data=readtable('HT001a.csv','ReadRowNames',true,'ReadVariableNames',true)
varfun(@(x) sum(x), data)
data('column_mean',:)=varfun(@(x) sum(x), data)
```

```
data =
              b    c    d    e

              _    _    _    _
    0         1    2    3    4
    1         5    6    7    8
    2         9    8    7    6
    3         5    4    3    2
ans =
    Fun_b    Fun_c    Fun_d    Fun_e

    _____    _____    _____    _____
    20       20       20       20
data =
                   b     c     d     e

                   __    __    __    __
    0               1     2     3     4
    1               5     6     7     8
    2               9     8     7     6
    3               5     4     3     2
    column_mean    20    20    20    20
```

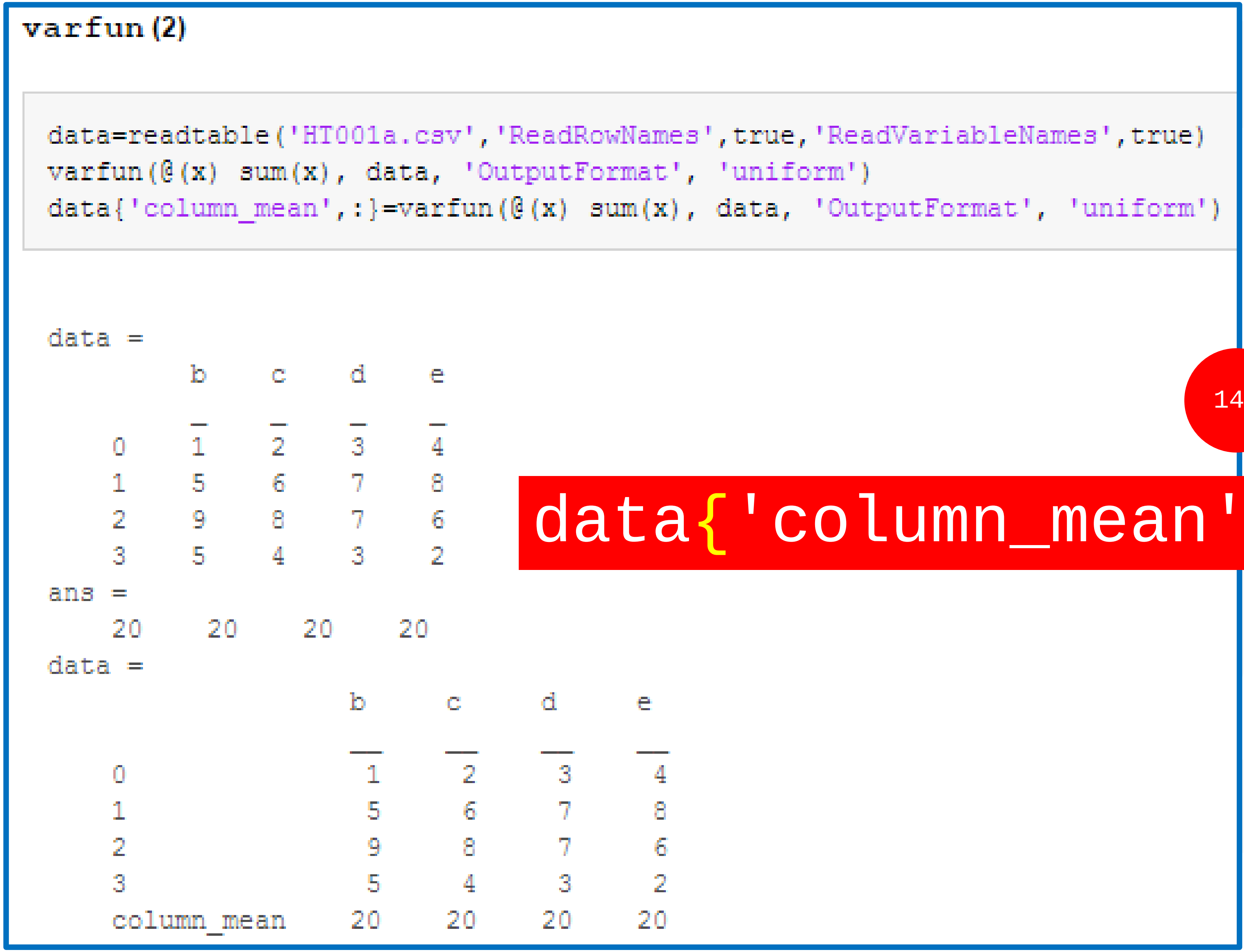
148

data('column_mean',:)=

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance



varfun(2)

```
data=readtable('HT001a.csv','ReadRowNames',true,'ReadVariableNames',true)
varfun(@(x) sum(x), data, 'OutputFormat', 'uniform')
data{'column_mean',:}=varfun(@(x) sum(x), data, 'OutputFormat', 'uniform')
```

```
data =
            b    c    d    e
            _    _    _    _
    0       1    2    3    4
    1       5    6    7    8
    2       9    8    7    6
    3       5    4    3    2
ans =
    20     20     20     20
data =
                    b      c      d      e
                   ___    ___    ___    ___
    0               1      2      3      4
    1               5      6      7      8
    2               9      8      7      6
    3               5      4      3      2
    column_mean    20     20     20     20
```

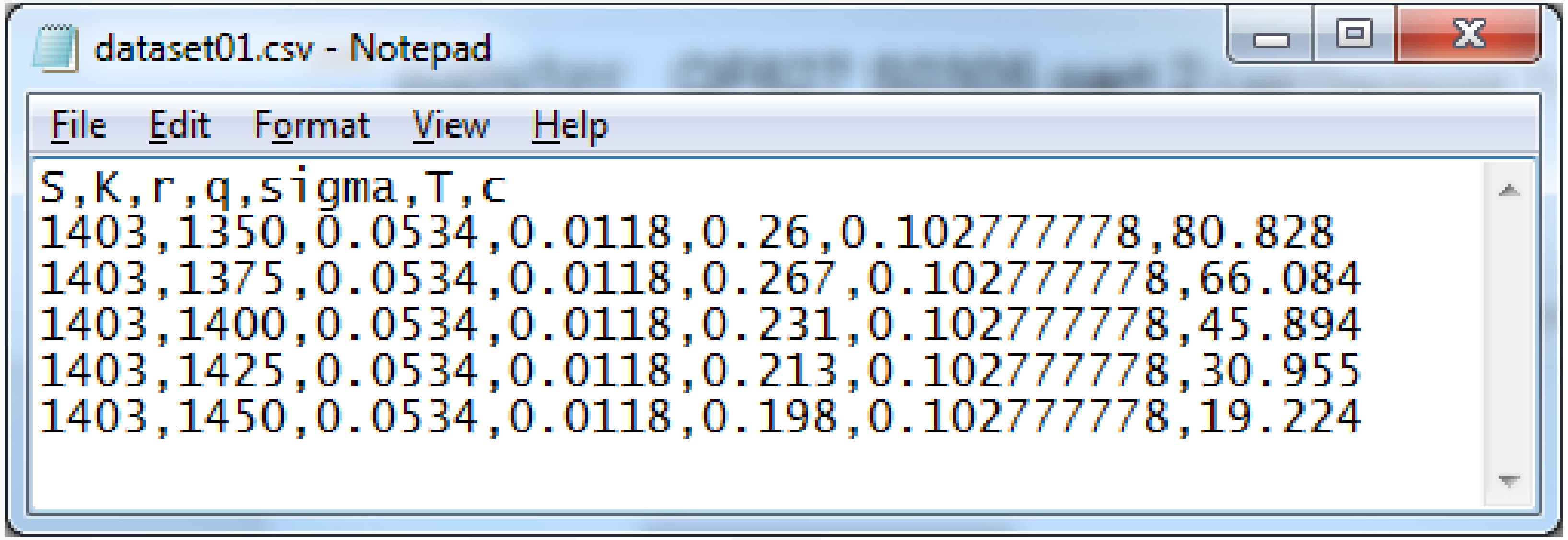
148

data{'column_mean',:}=

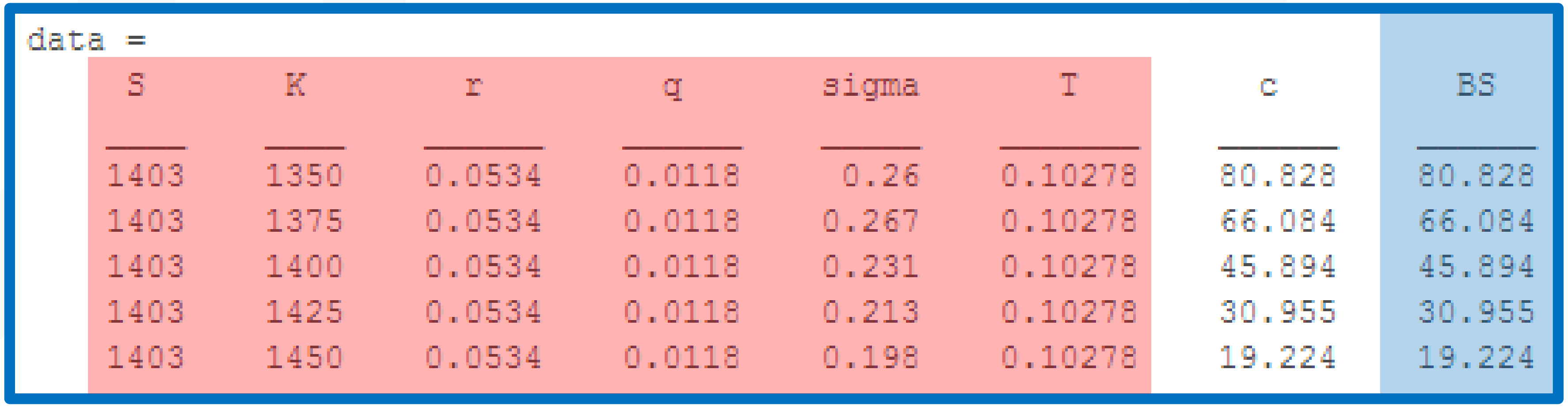
QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Homework Question:



```
S,K,r,q,sigma,T,c
1403,1350,0.0534,0.0118,0.26,0.102777778,80.828
1403,1375,0.0534,0.0118,0.267,0.102777778,66.084
1403,1400,0.0534,0.0118,0.231,0.102777778,45.894
1403,1425,0.0534,0.0118,0.213,0.102777778,30.955
1403,1450,0.0534,0.0118,0.198,0.102777778,19.224
```



```
data =
     S       K        r         q       sigma       T           c         BS
    ____    ____    ______    ______    _____    _______    ______    ______
    1403    1350    0.0534    0.0118     0.26    0.10278    80.828    80.828
    1403    1375    0.0534    0.0118    0.267    0.10278    66.084    66.084
    1403    1400    0.0534    0.0118    0.231    0.10278    45.894    45.894
    1403    1425    0.0534    0.0118    0.213    0.10278    30.955    30.955
    1403    1450    0.0534    0.0118    0.198    0.10278    19.224    19.224
```

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance

1. Load data from the CSV file, `dataset01.csv`, using the first row as column names. Name the data as **`data`**.
2. Define a function, **`option_BS`**, which computes and returns the European call option price using the following formula:

$$c = S \cdot e^{-q \cdot T} \cdot \Phi(d_1) - K \cdot e^{-r \cdot T} \cdot \Phi(d_2)$$

where

$$d_1 = \frac{\ln\left(\frac{S}{K}\right) + \left(r - q + \frac{\sigma^2}{2}\right)T}{\sigma\sqrt{T}} \text{ and } d_2 = d_1 - \sigma\sqrt{T}$$

3. **Use one command** to compute the European call option price for each row of **`data`** and add the results to **`data`** as a new column, and name this column as **BS**.

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

HT009: Basic operations on two rows/columns of **`data`**

1. row(s) *op* row(s) (with the same label)
2. row(s) *op* row(s) (with different labels)
3. column(s) *op* column(s) (with the same label)
4. column(s) *op* column(s) (with different labels)
5. row(s) *op* column(s) (??? What operation???)

op: +, -, *, /, ** (or ^)

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

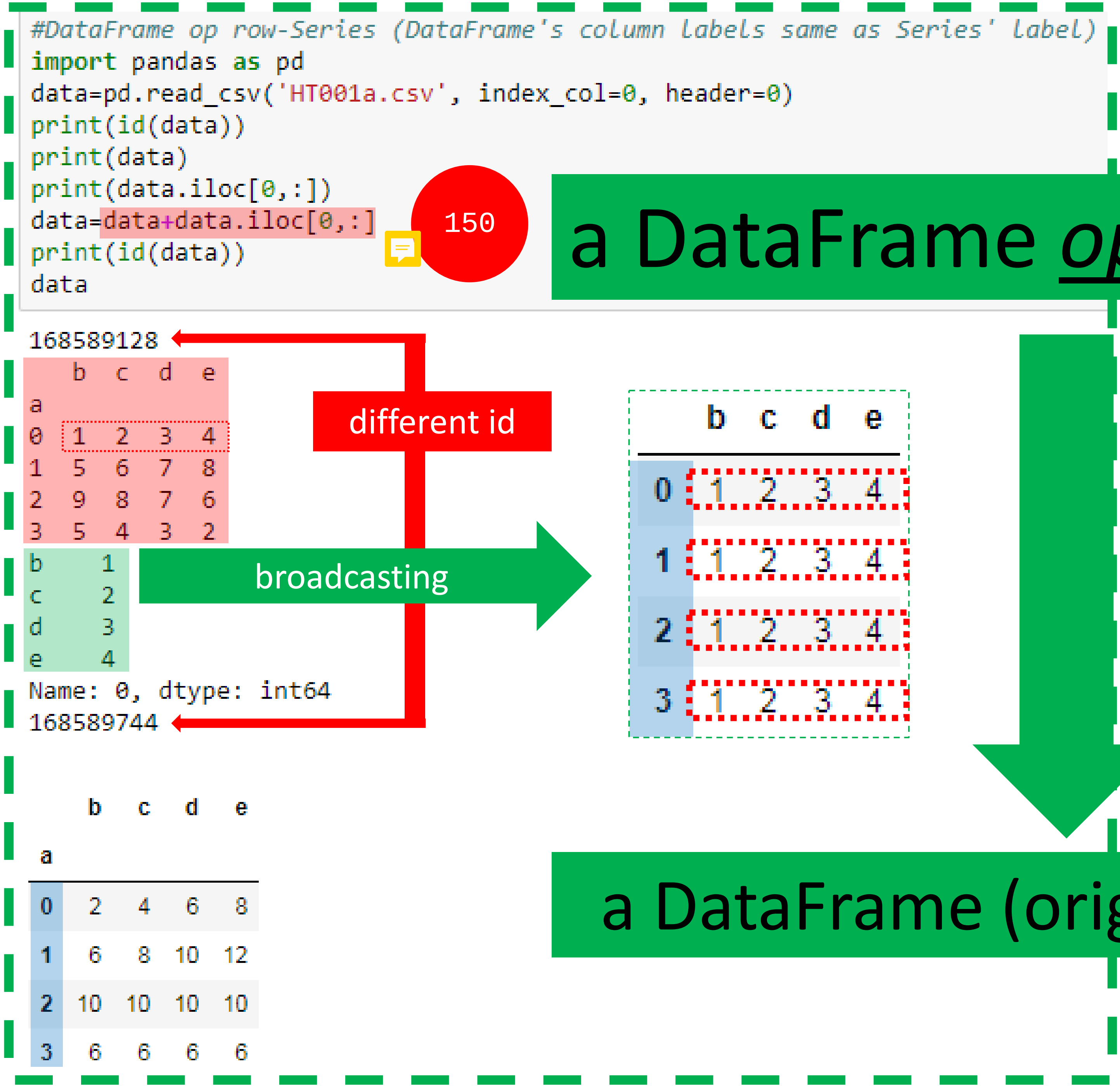
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

HT009: Basic operations on two rows/columns of **data**

- ✓ A row/column of a DataFrame can be a DataFrame, a Series, a Numpy 2D array or a Numpy 1D array.
- ✓ Rows/Columns of a DataFrame can be a DataFrame or a Numpy 2D array.

	DataFrame	Series	Numpy 2D array	Numpy 1D array
DataFrame	element-wise, aligned by labels	broadcasting, align DataFrame's column labels and Series' labels	element-wise, size must agree	use array as a row, broadcasting, element-wise, **size** must agree
Series	▨	element-wise, aligned by labels	N.A.	element-wise, size must agree
Numpy 2D array	▨	▨	broadcasting, element-wise, size must agree	broadcasting
Numpy 1D array	▨	▨	▨	element-wise, size must agree

Slide
270/307



```
#DataFrame op row-Series (DataFrame's column labels same as Series' label)
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(id(data))
print(data)
print(data.iloc[0,:])
data=data+data.iloc[0,:]
print(id(data))
data
```

150

```
168589128
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
b    1
c    2
d    3
e    4
Name: 0, dtype: int64
168589744
```

different id

broadcasting

	b	c	d	e
0	1	2	3	4
1	1	2	3	4
2	1	2	3	4
3	1	2	3	4

	b	c	d	e
a				
0	2	4	6	8
1	6	8	10	12
2	10	10	10	10
3	6	6	6	6

data=

a DataFrame *op* a row-Series

(row)

(N,)⇨(1,N)

Use one command to add the first row to every row in **data**.

a DataFrame (original row labels)

QF666
Programming and Computational Finance

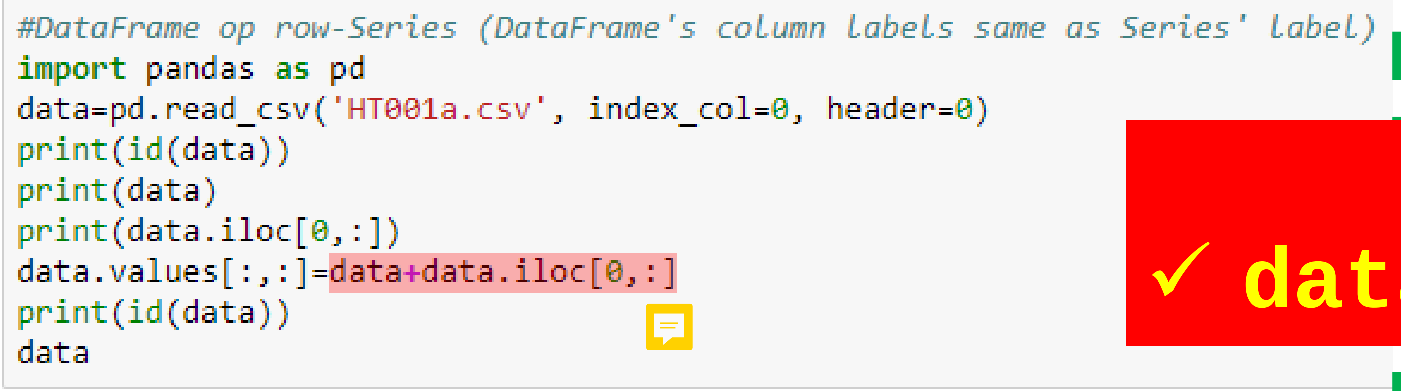
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

Slide
271/307

SMU
SINGAPORE MANAGEMENT UNIVERSITY

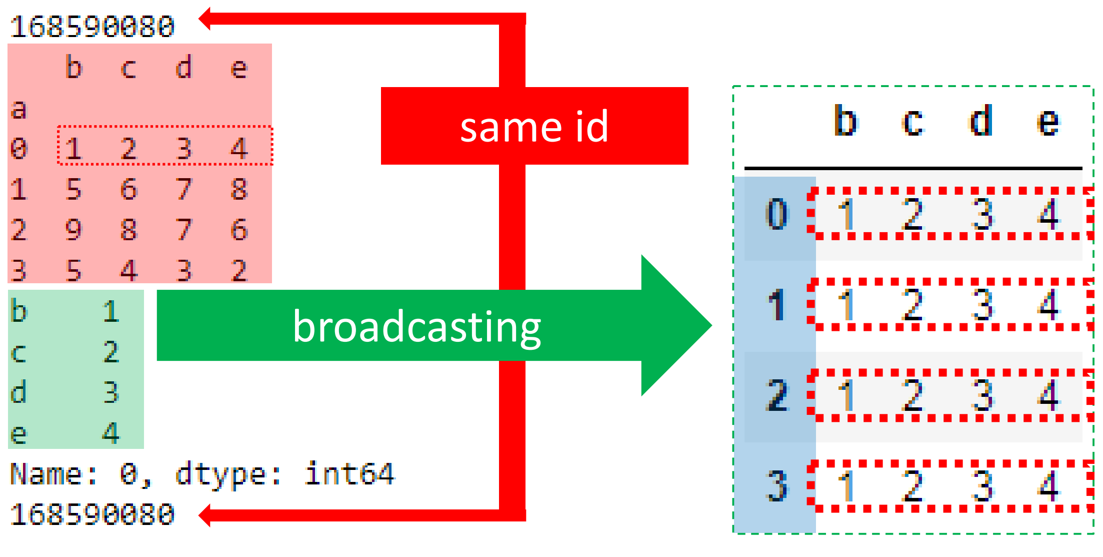
QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



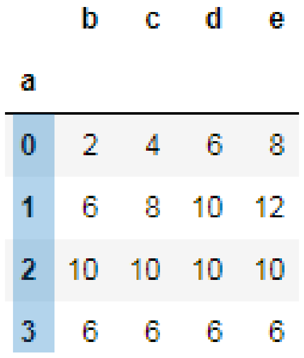
```
#DataFrame op row-Series (DataFrame's column labels same as Series' label)
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(id(data))
print(data)
print(data.iloc[0,:])
data.values[:,:]=data+data.iloc[0,:]
print(id(data))
data
```

× data.values=
✓ data.values[:,:]=



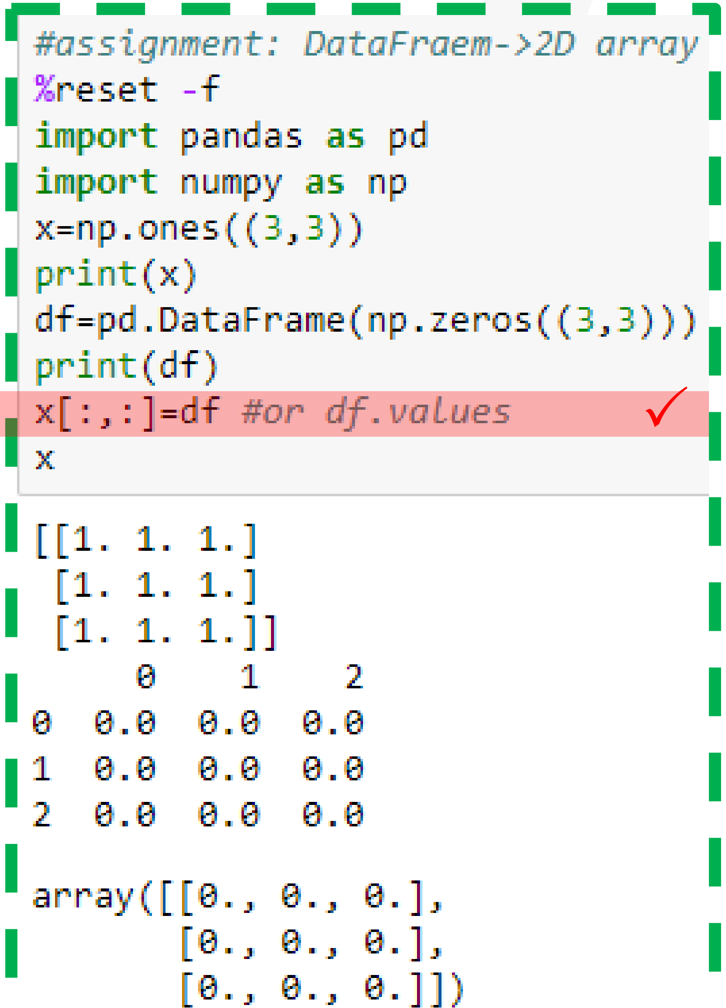
```
168590080
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
b    1
c    2
d    3
e    4
Name: 0, dtype: int64
168590080
```

	b	c	d	e
0	1	2	3	4
1	1	2	3	4
2	1	2	3	4
3	1	2	3	4



	b	c	d	e
a				
0	2	4	6	8
1	6	8	10	12
2	10	10	10	10
3	6	6	6	6

Use one command to add the first row to every row in **data**.



```
#assignment: DataFraem->2D array
%reset -f
import pandas as pd
import numpy as np
x=np.ones((3,3))
print(x)
df=pd.DataFrame(np.zeros((3,3)))
print(df)
x[:,:]=df #or df.values ✓
x
```

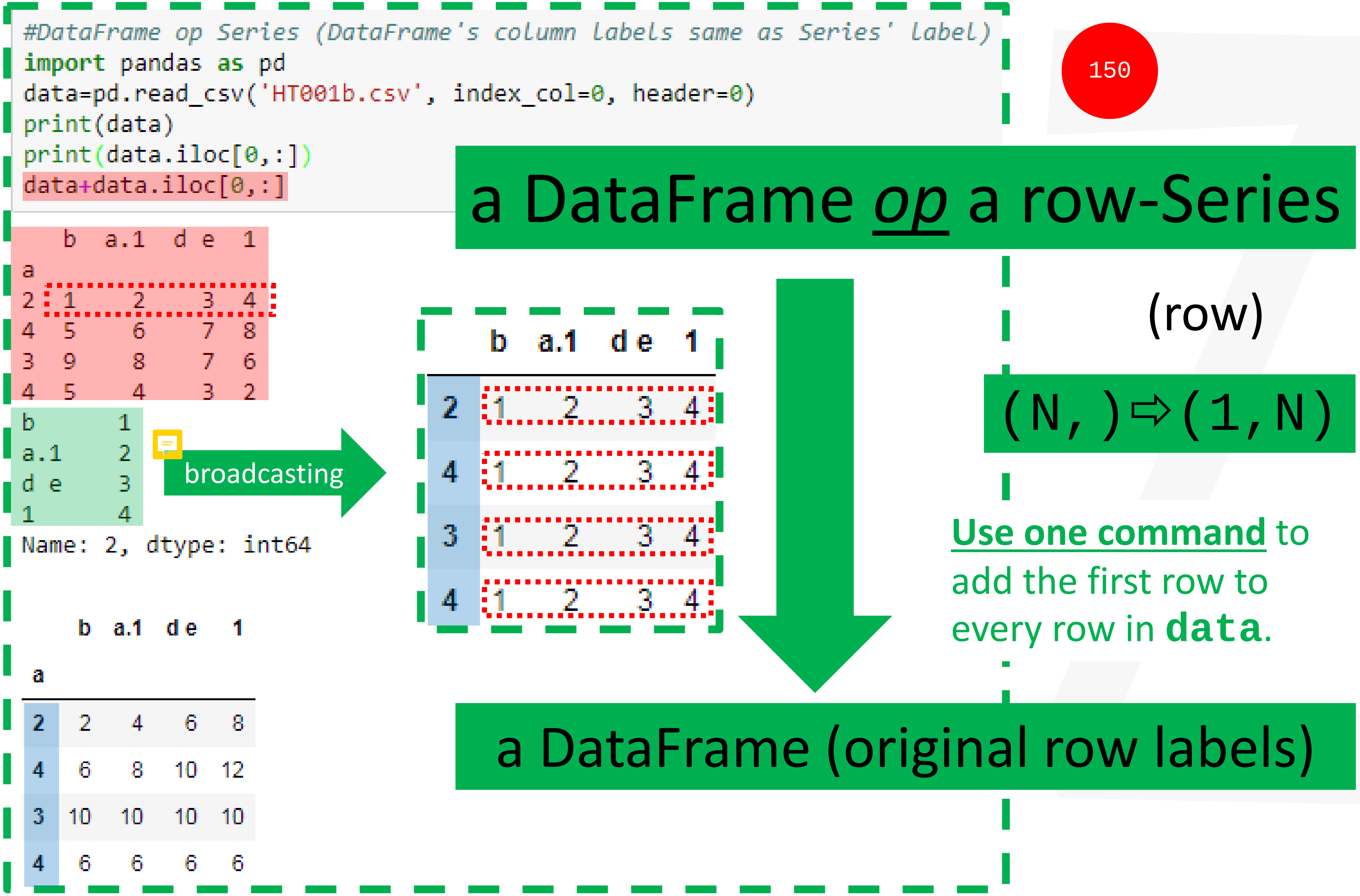
```
[[1. 1. 1.]
 [1. 1. 1.]
 [1. 1. 1.]]
     0    1    2
0  0.0  0.0  0.0
1  0.0  0.0  0.0
2  0.0  0.0  0.0

array([[0., 0., 0.],
       [0., 0., 0.],
       [0., 0., 0.]])
```

Slide
272/307

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



150

```
#DataFrame op Series (DataFrame's column labels same as Series' label)
import pandas as pd
data=pd.read_csv('HT001b.csv', index_col=0, header=0)
print(data)
print(data.iloc[0,:])
data+data.iloc[0,:]
```

```
   b  a.1  d e  1
a
2  1    2    3  4
4  5    6    7  8
3  9    8    7  6
4  5    4    3  2
b      1
a.1    2
d e    3
1      4
Name: 2, dtype: int64
```

a	b	a.1	d e	1
2	2	4	6	8
4	6	8	10	12
3	10	10	10	10
4	6	6	6	6

broadcasting

	b	a.1	d e	1
2	1	2	3	4
4	1	2	3	4
3	1	2	3	4
4	1	2	3	4

a DataFrame *op* a row-Series

(row)

(N,)⇨(1,N)

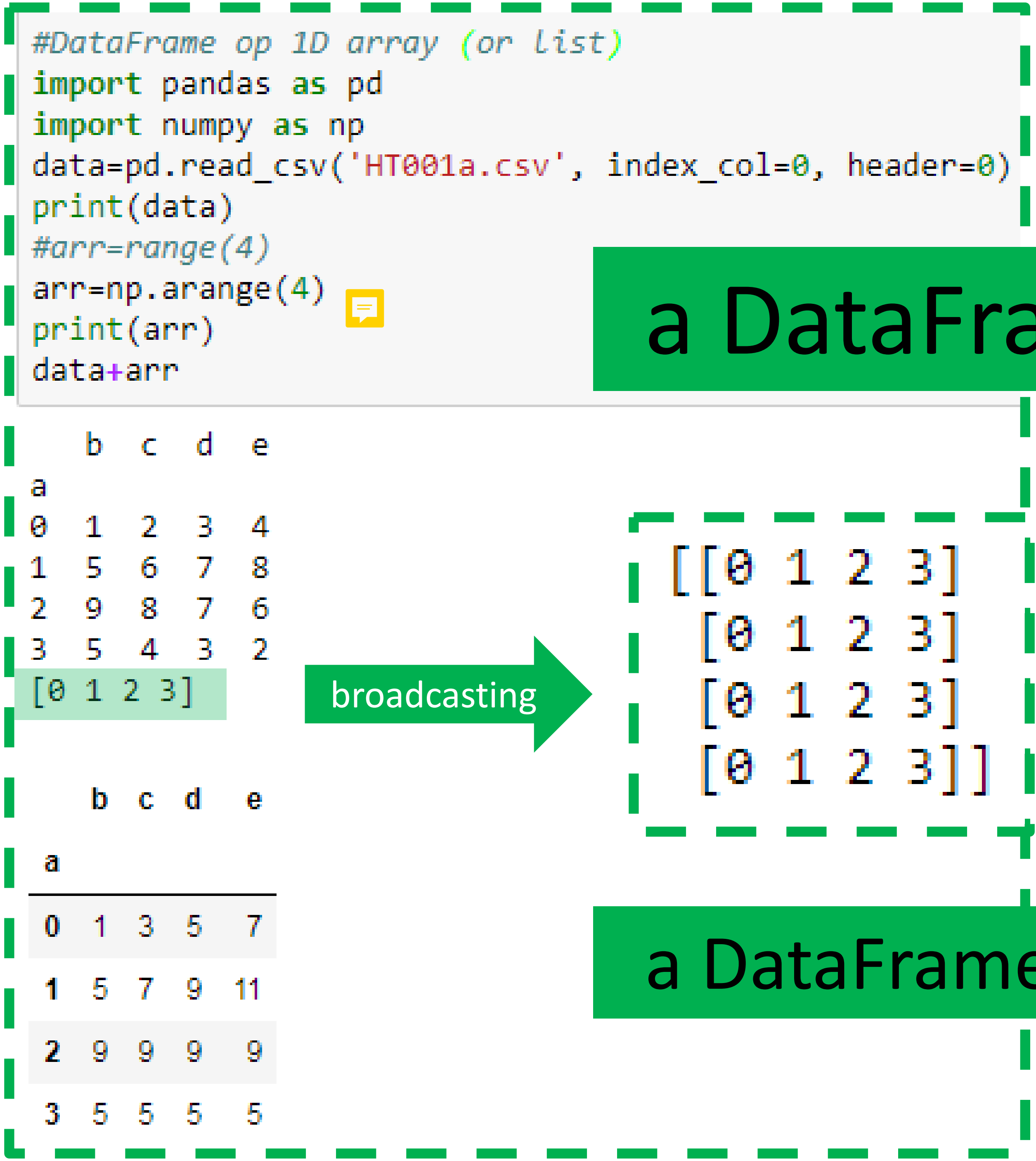
a DataFrame (original row labels)

Use one command to add the first row to every row in **data**.

a DataFrame *op* 1D array

(or list)

151



```
#DataFrame op 1D array (or list)
import pandas as pd
import numpy as np
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
#arr=range(4)
arr=np.arange(4)
print(arr)
data+arr
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
[0 1 2 3]
```

broadcasting

```
[[0 1 2 3]
 [0 1 2 3]
 [0 1 2 3]
 [0 1 2 3]]
```

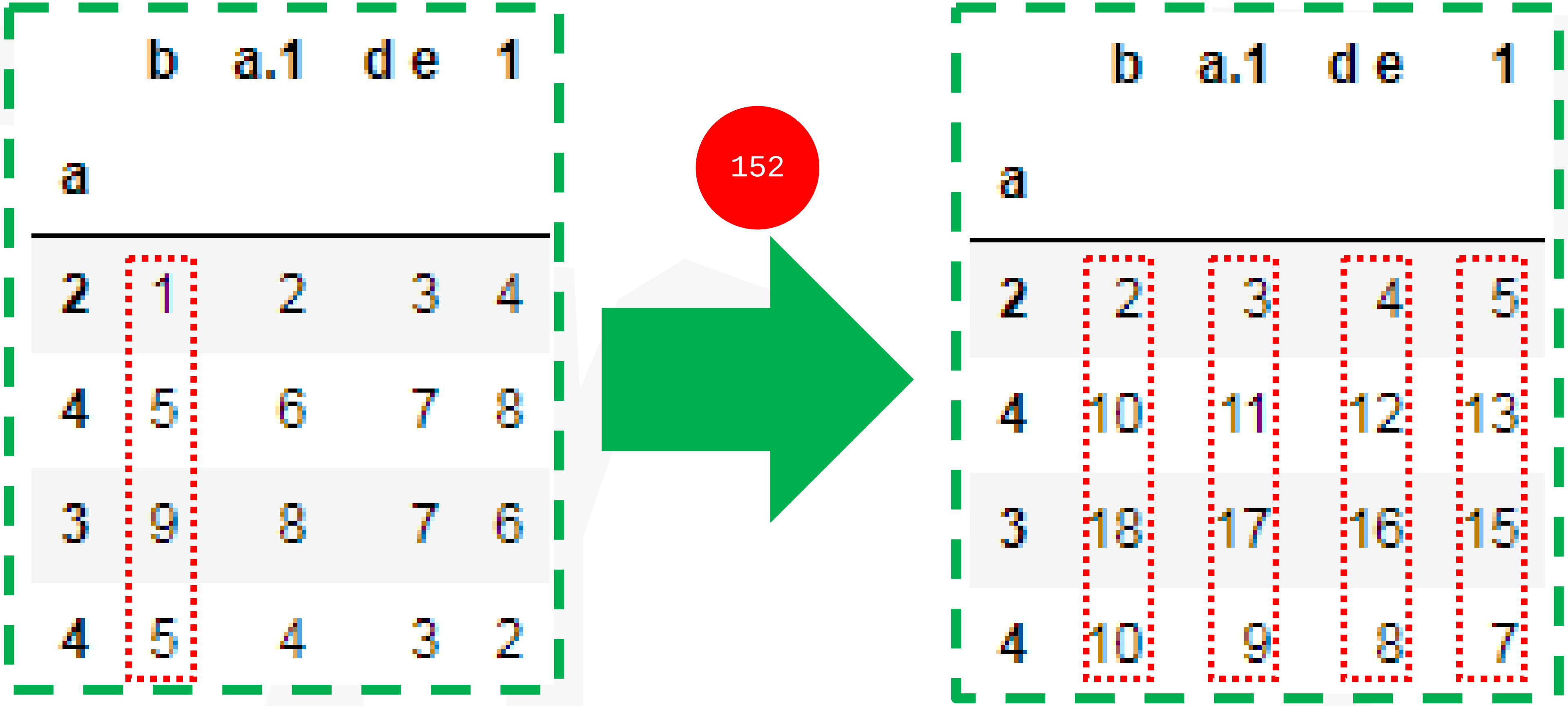
	b	c	d	e
a				
0	1	3	5	7
1	5	7	9	11
2	9	9	9	9
3	5	5	5	5

a DataFrame (original row labels)

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



a	b	a.1	d e	1
2	1	2	3	4
4	5	6	7	8
3	9	8	7	6
4	5	4	3	2

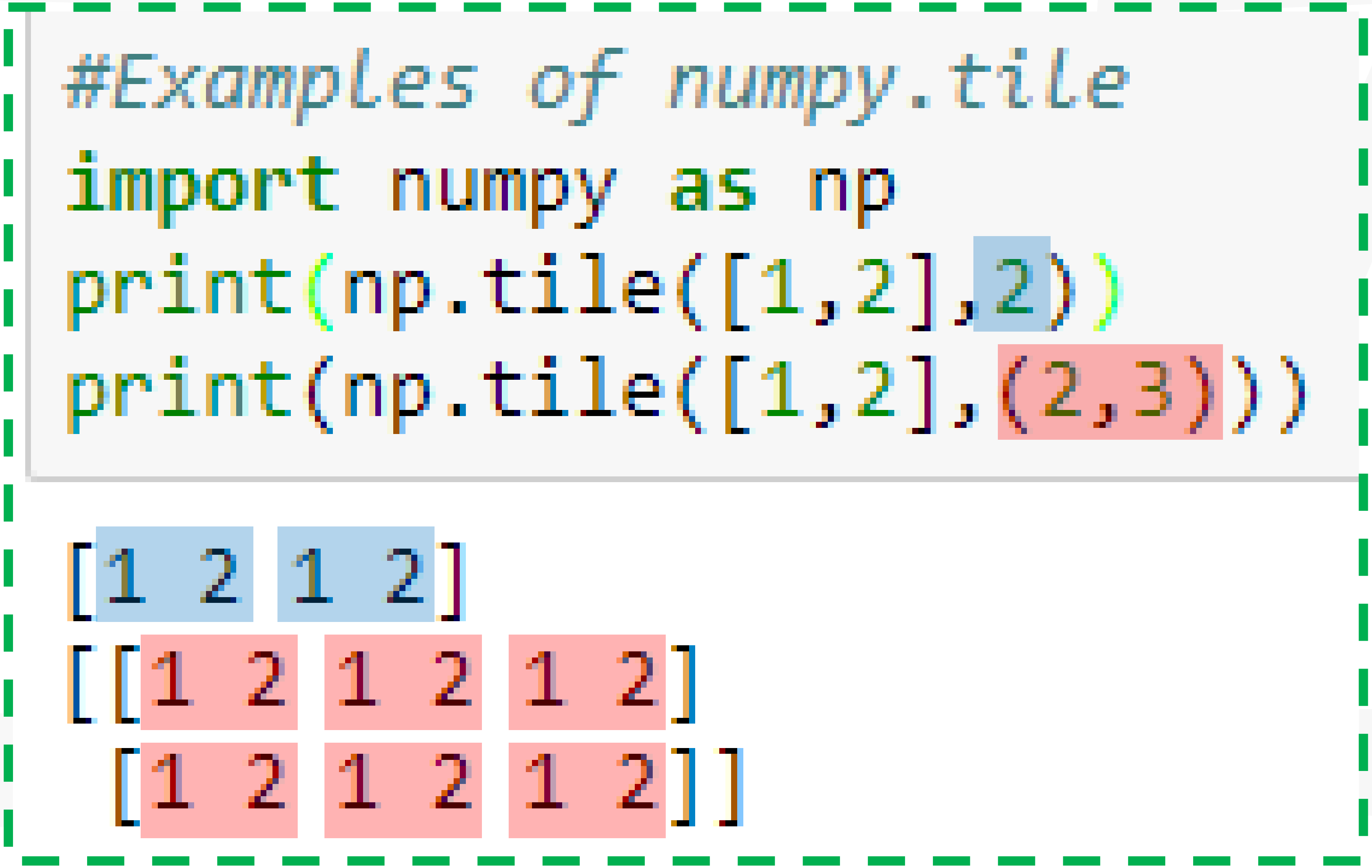
a	b	a.1	d e	1
2	2	3	4	5
4	10	11	12	13
3	18	17	16	15
4	10	9	8	7

Use one command to add the first column to every column in **data**.

numpy.tile

```
×  data+data.iloc[:,0]
?  data+data.iloc[:,0].values
×  data+data.iloc[:,0].values.reshape(4,1)
✓  data+np.tile(data.iloc[:,0].values.reshape(4,1),4)
✓  data+np.tile(data.iloc[:,[0]].values,4)
✓  data.apply(lambda x: x+data.iloc[:,0].values, axis=0)
```

https://docs.scipy.org/doc/numpy-1.15.0/reference/generated/numpy.tile.html



```
#Examples of numpy.tile
import numpy as np
print(np.tile([1,2],2))
print(np.tile([1,2],(2,3)))
```

```
[1 2 1 2]
[[1 2 1 2 1 2]
 [1 2 1 2 1 2]]
```

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

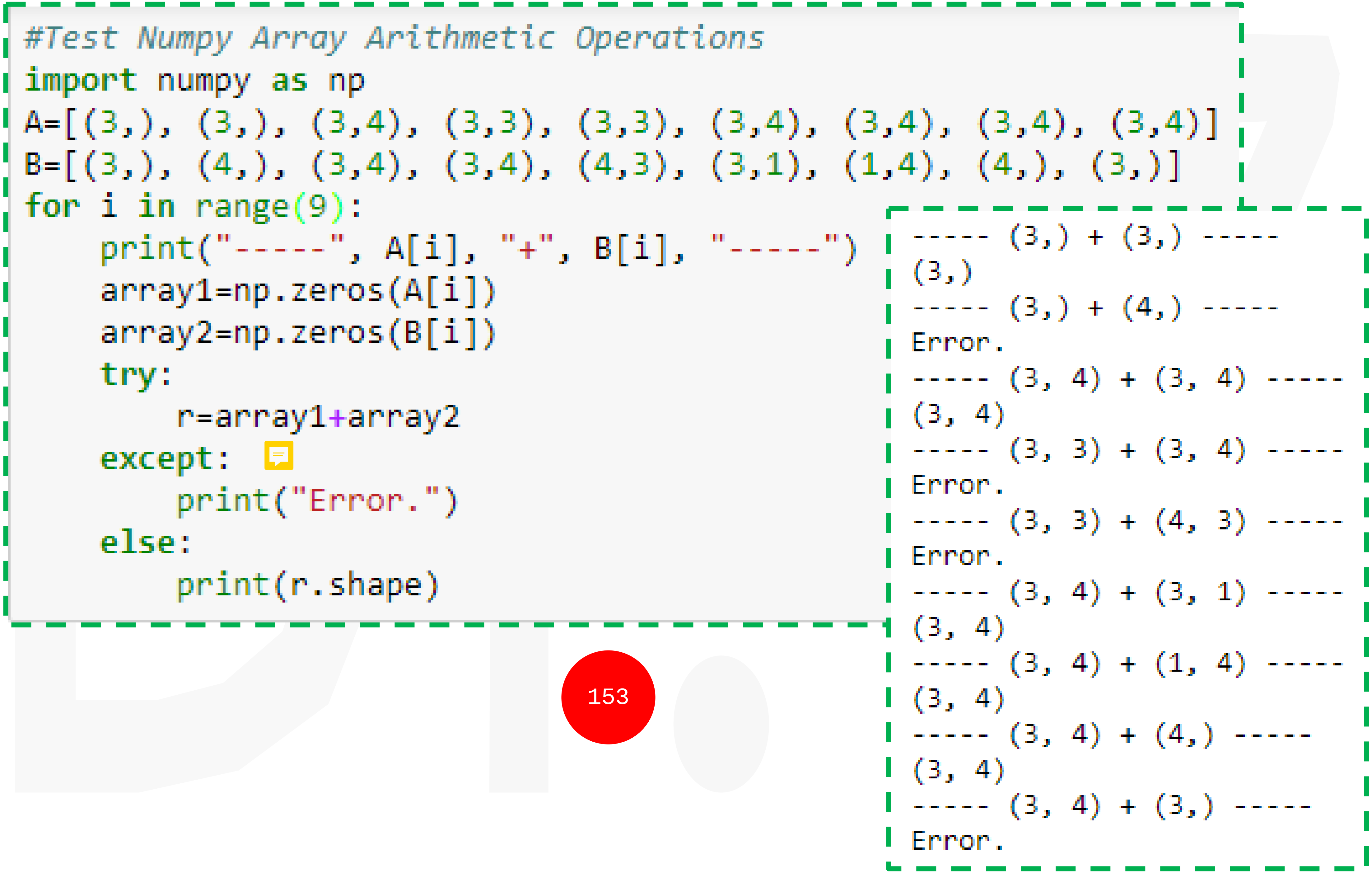
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Numpy Array Arithmetic Operations

$1 \le M_1 < M_2; 1 \le N_1 < N_2$

153

- ✓ $(M_1,)$ 1D array *op* $(M_1,)$ 1D array ⇨ $(M_1,)$ 1D array
- × $(M_1,)$ 1D array *op* $(M_2,)$ 1D array
- ✓ (M_1, N_1) 2D array *op* (M_1, N_1) 2D array ⇨ (M_1, N_1) 2D array
- × (M_1, N_1) 2D array *op* (M_1, N_2) 2D array
- × (M_1, N_1) 2D array *op* (M_2, N_1) 2D array
- ✓ $(M_1, 1)$ 2D array *op* $(1, N_1)$ 2D array ⇨ (M_1, N_1) 2D array
- ✓ (M_1, N_1) 2D array *op* $(M_1, 1)$ 2D array ⇨ (M_1, N_1) 2D array
- ✓ (M_1, N_1) 2D array *op* $(1, N_1)$ 2D array ⇨ (M_1, N_1) 2D array
- ✓ (M_1, N_1) 2D array *op* $(N_1,)$ 1D array ⇨ (M_1, N_1) 2D array
- × (M_1, N_2) 2D array *op* $(M_1,)$ 1D array



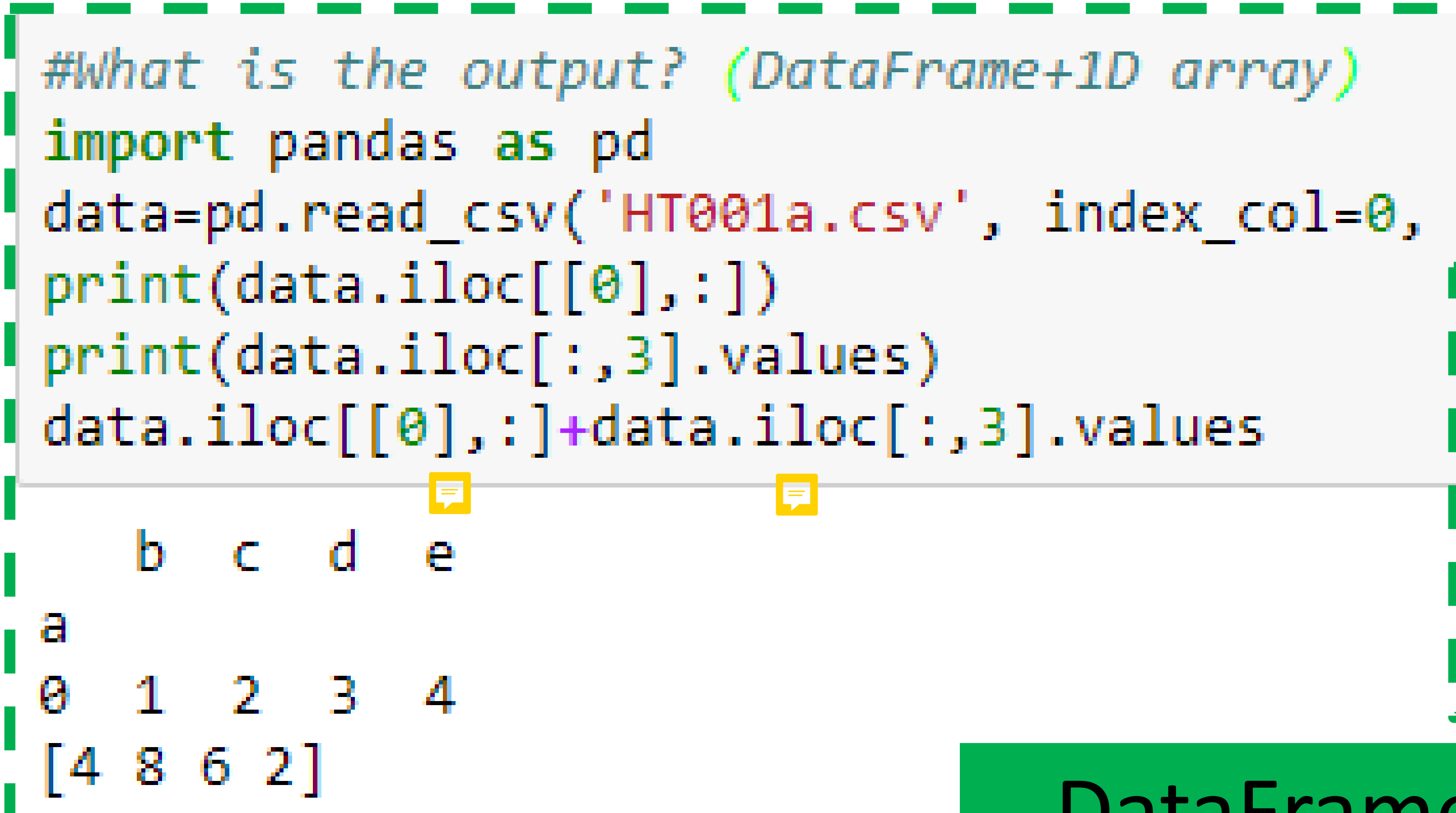
```
#Test Numpy Array Arithmetic Operations
import numpy as np
A=[(3,), (3,), (3,4), (3,3), (3,3), (3,4), (3,4), (3,4), (3,4)]
B=[(3,), (4,), (3,4), (3,4), (4,3), (3,1), (1,4), (4,), (3,)]
for i in range(9):
    print("-----", A[i], "+", B[i], "-----")
    array1=np.zeros(A[i])
    array2=np.zeros(B[i])
    try:
        r=array1+array2
    except:
        print("Error.")
    else:
        print(r.shape)
```

```
----- (3,) + (3,) -----
(3,)
----- (3,) + (4,) -----
Error.
----- (3, 4) + (3, 4) -----
(3, 4)
----- (3, 3) + (3, 4) -----
Error.
----- (3, 3) + (4, 3) -----
Error.
----- (3, 4) + (3, 1) -----
(3, 4)
----- (3, 4) + (1, 4) -----
(3, 4)
----- (3, 4) + (4,) -----
(3, 4)
----- (3, 4) + (3,) -----
Error.
```

SMU
SINGAPORE MANAGEMENT UNIVERSITY

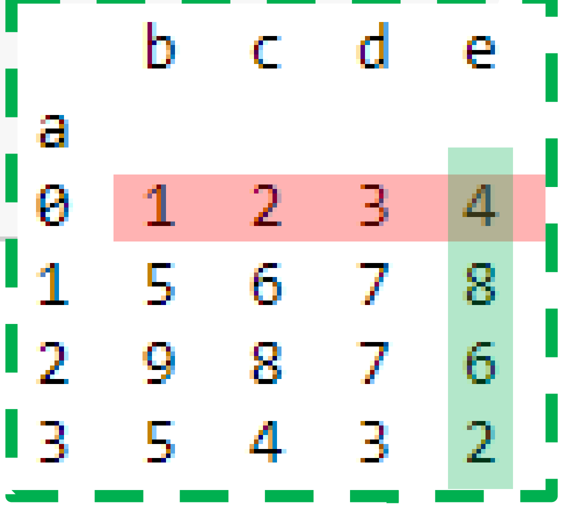
QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
#What is the output? (DataFrame+1D array)
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data.iloc[[0],:])
print(data.iloc[:,3].values)
data.iloc[[0],:]+data.iloc[:,3].values
```

```
   b  c  d  e
a
0  1  2  3  4
[4 8 6 2]
```



```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
```

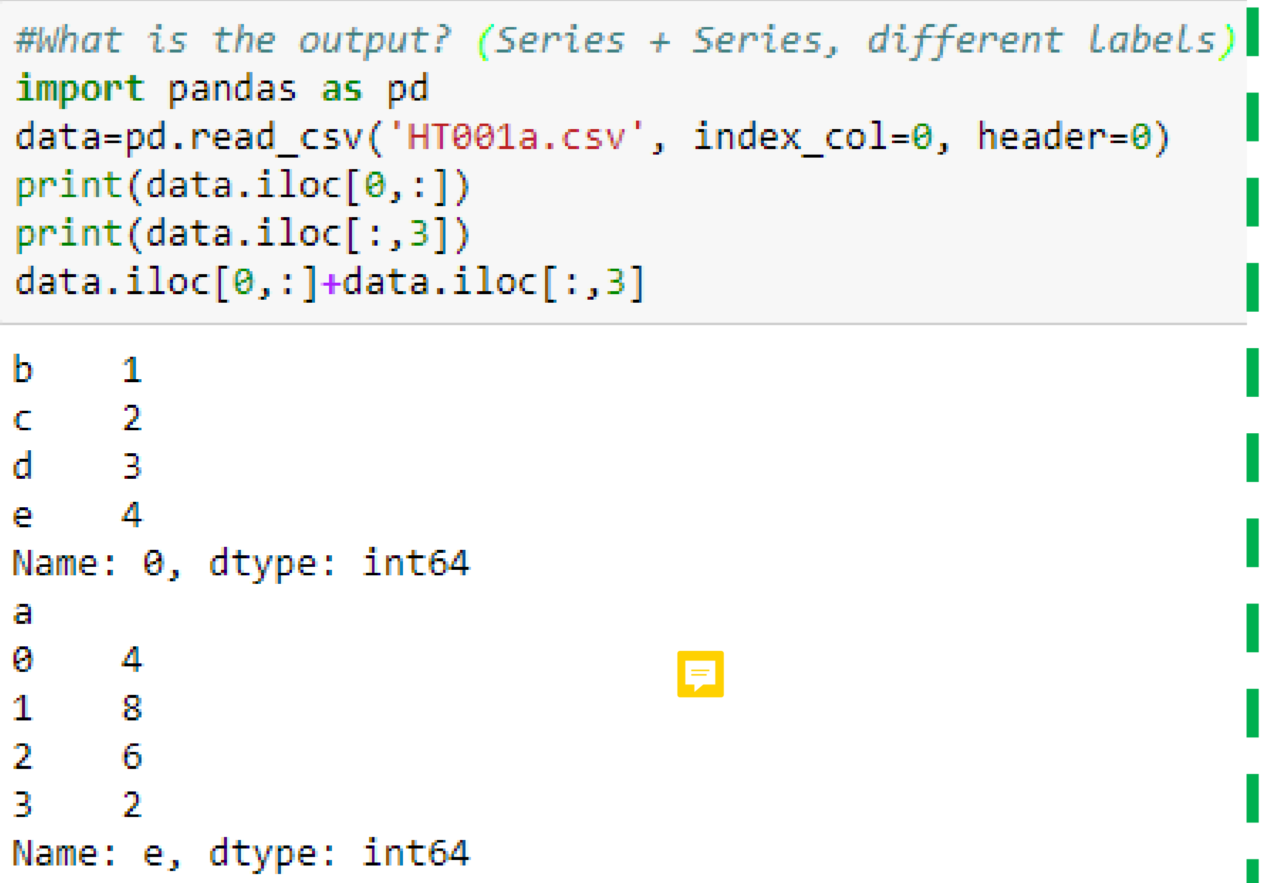
DataFrame+1D array

What is the output?

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

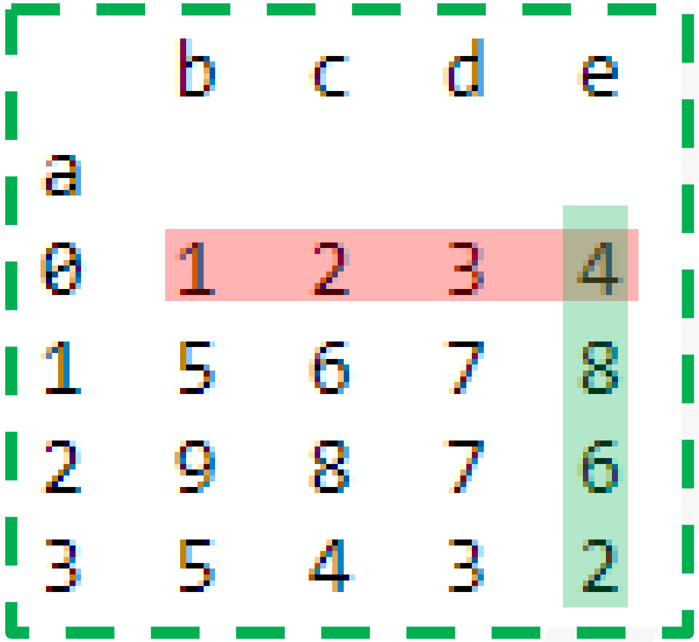
Slide
279/307



```
#What is the output? (Series + Series, different labels)
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data.iloc[0,:])
print(data.iloc[:,3])
data.iloc[0,:]+data.iloc[:,3]
```

```
b    1
c    2
d    3
e    4
Name: 0, dtype: int64
a
0    4
1    8
2    6
3    2
Name: e, dtype: int64
```

What is the output?



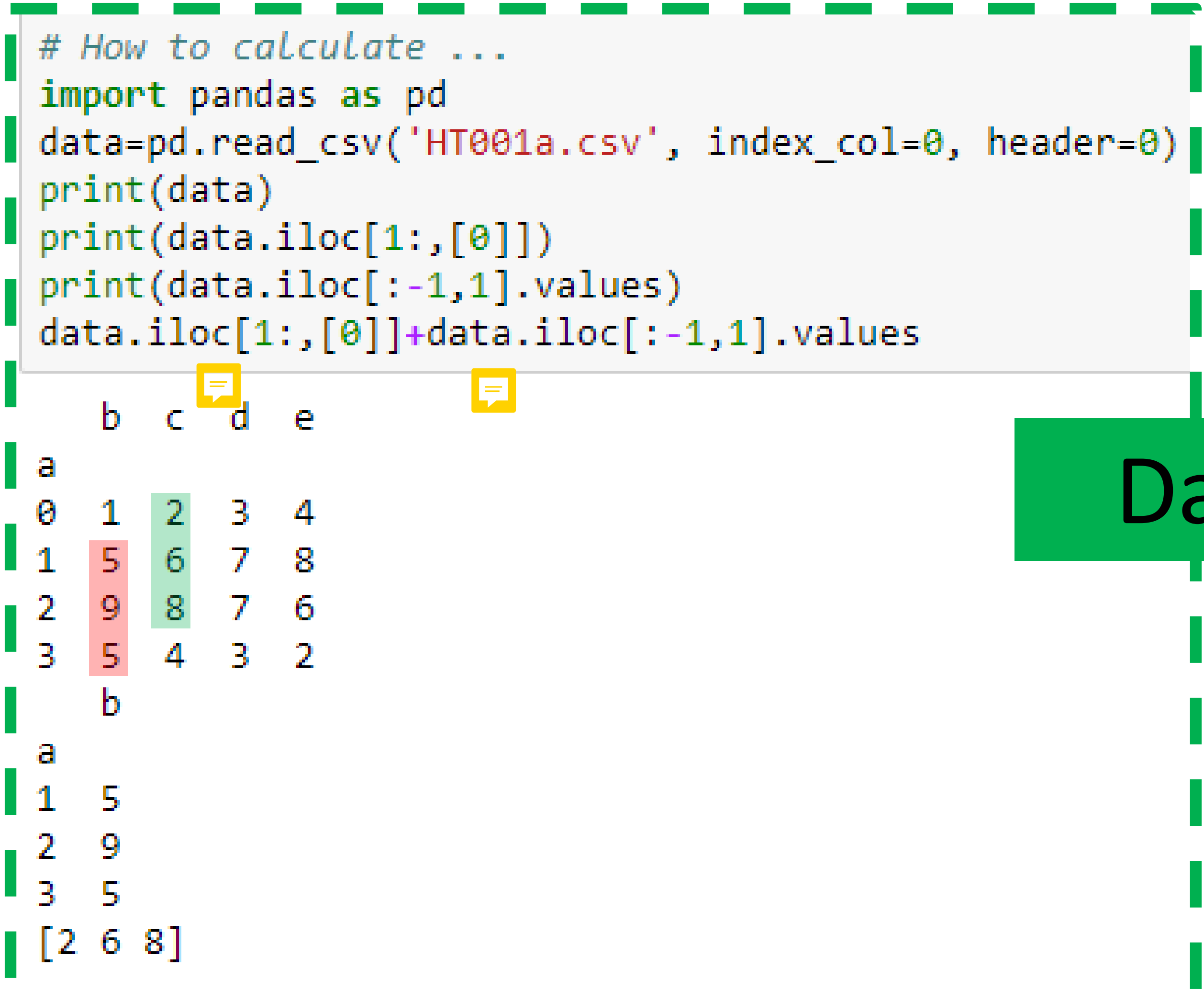
	b	c	d	e
a				
0	1	2	3	4
1	5	6	7	8
2	9	8	7	6
3	5	4	3	2

Series+Series

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

DataFrame+1D array



```python
# How to calculate ...
import pandas as pd
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
print(data.iloc[1:,[0]])
print(data.iloc[:-1,1].values)
data.iloc[1:,[0]]+data.iloc[:-1,1].values
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
   b
a
1  5
2  9
3  5
[2 6 8]
```

What is the output?

calculate ⇨ use array (label free)

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Matrix Multiplication using Numpy 1D/2D arrays

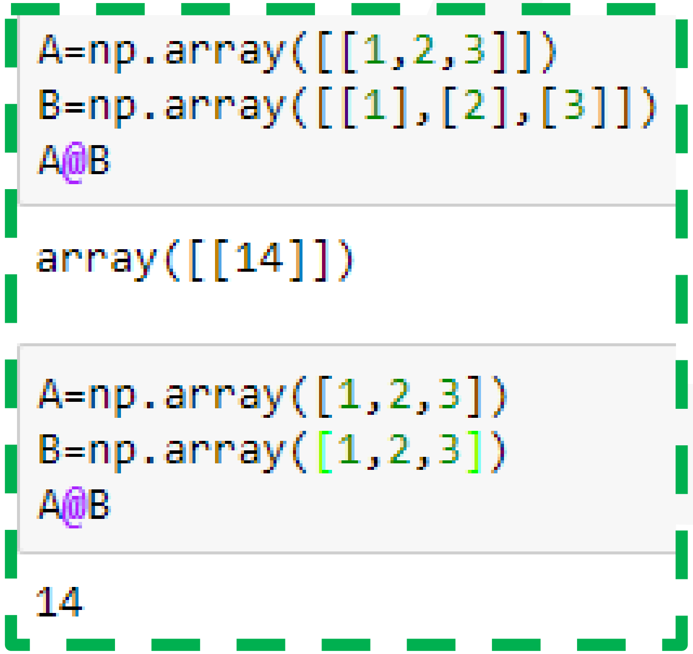
(M, N) 2D array
or
(N,) 1D array used
as (1,N) 2D array

@

(N, P) 2D array
or
(N,) 1D array used
as (N,1) 2D array

A	B	A @ B
(M, N)	(N, P)	(M, P)
(M, N)	(N, 1)	(M, 1)
(M, N)	(N,)	(M,)
(1, N)	(N, P)	(1, P)
(N,)	(N, P)	(P,)
(1, N)	(N, 1)	(1, 1)
(N,)	(N,)	()

154



```
A=np.array([[1,2,3]])
B=np.array([[1],[2],[3]])
A@B
```

```
array([[14]])
```

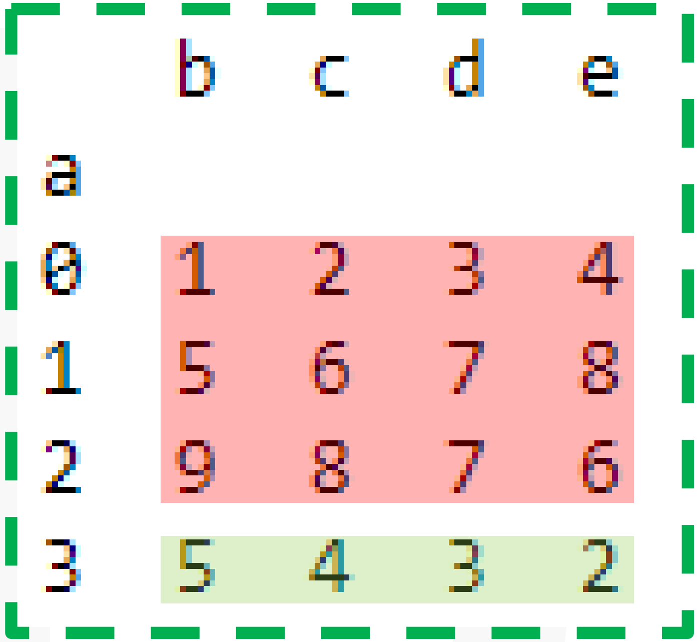
```
A=np.array([1,2,3])
B=np.array([1,2,3])
A@B
```

```
14
```

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
```

$$\begin{pmatrix}1&2&3&4\\5&6&7&8\\9&8&7&6\end{pmatrix} \times \begin{pmatrix}5\\4\\3\\2\end{pmatrix}$$

(3,4) (4,1)

155

Use one command to compute the matrix multiplication, using the first 3 rows in **data** as the first matrix, and using the last row (without finding the number of rows) as the second 1-column matrix.

```
✓ data.values[:3] @ data.values[-1]
✓ data.values[:3] @ data.values[-1:].T
✓ data.values[:3] @ data.values[[-1]].T
```

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

HT009: Basic operations on two rows/columns of `data`

✓ MATLAB tables do not support arithmetic operations.

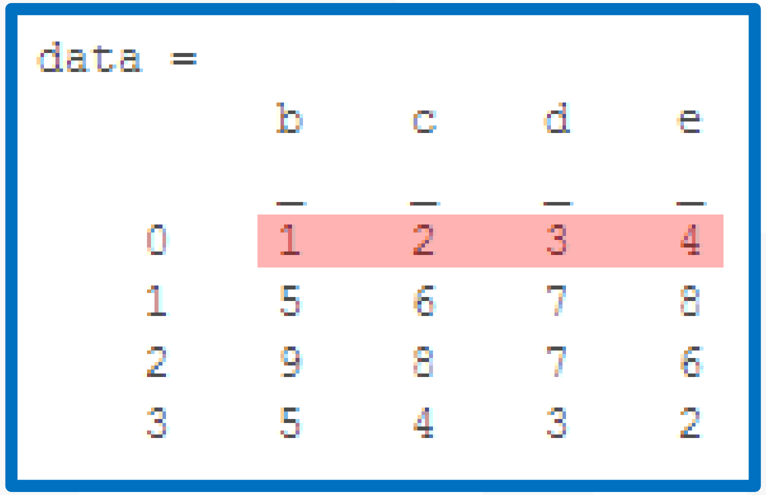
HAHAHA....
HAHAHA....

✓ Using the dot syntax (⇨1-column 2D array) or **{}**-indexing, we obtain MATLAB arrays.

- ✓ 1D arrays are row matrices (or 1-row 2D arrays).
- ✓ Matrix dimensions must agree or one is a scalar.
- ✓ Manual broadcasting

Wow~~~

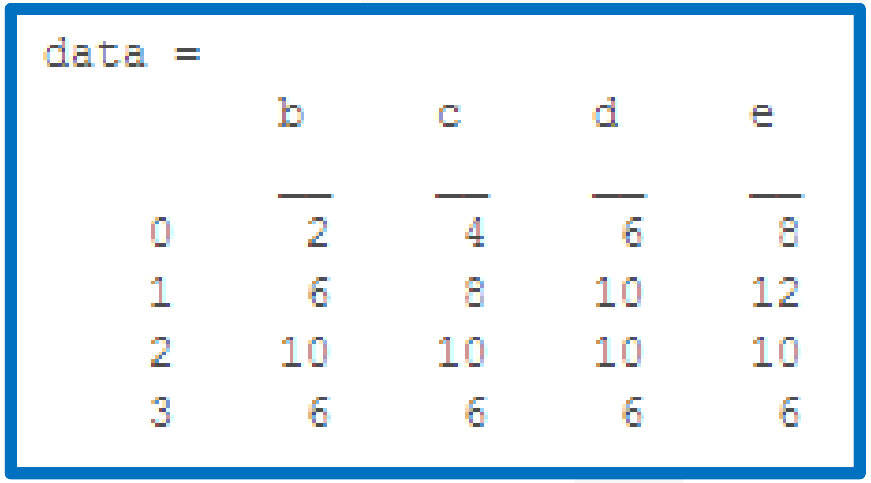
156



```
data =

         b    c    d    e
         _    _    _    _

    0    1    2    3    4
    1    5    6    7    8
    2    9    8    7    6
    3    5    4    3    2
```

1. Add the first row to every row.
2. “help repmat”
3. “help size”

Use one command



```
data =

          b     c     d     e
         __    __    __    __

    0     2     4     6     8
    1     6     8    10    12
    2    10    10    10    10
    3     6     6     6     6
```

157

```
data{:,:}=data{:,:}+repmat(data{1,:},size(data,1),1)
```

numpy.tile(A, (M, N)) ⇔ **repmat(A, M, N)**

or **repmat(A, [M, N])**

https://www.mathworks.com/help/matlab/ref/repmat.html

https://www.mathworks.com/help/matlab/ref/size.html

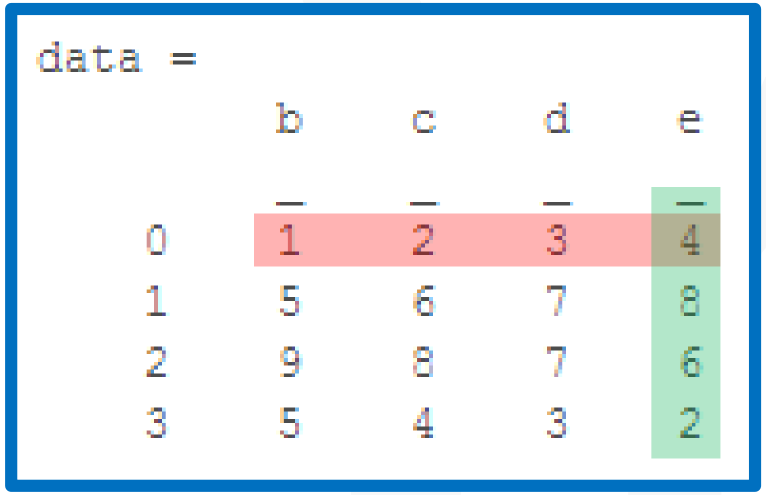
SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

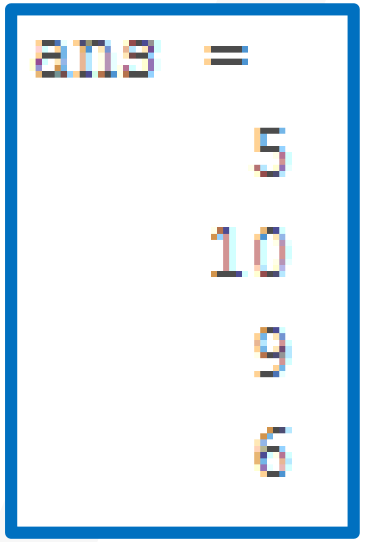
QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
data =
            b    c    d    e
            _    _    _    _
      0     1    2    3    4
      1     5    6    7    8
      2     9    8    7    6
      3     5    4    3    2
```

- ✓ Use one command to add the first row and last column (without using the size of **`data`**) of **`data`** elementwise and return the result in a column array.
- ✓ “help transpose”
- ✓ “help end”



```
ans =
      5
     10
      9
      6
```

158

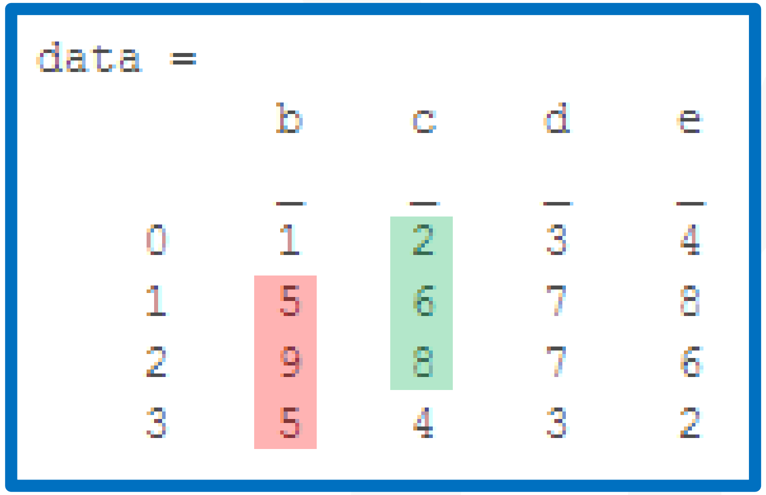
```
transpose(data{1,:})+data{:,end}
```

SMU
SINGAPORE MANAGEMENT UNIVERSITY

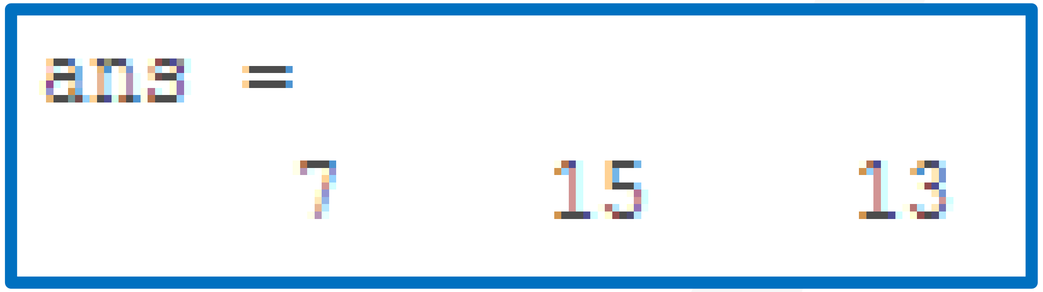
QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

- ✓ Use one command to add elements in the first column from the second row to the last row (without using the size of **`data`**) and elements in the second column from the first row to the second to the last row (without finding the size of **`data`**) element wise and return the result in a row array.



```
data =
          b    c    d    e
          _    _    _    _
     0    1    2    3    4
     1    5    6    7    8
     2    9    8    7    6
     3    5    4    3    2
```



```
ans =

     7     15     13
```

159

`transpose(data{2:end,1}+data{1:end-1,2})`

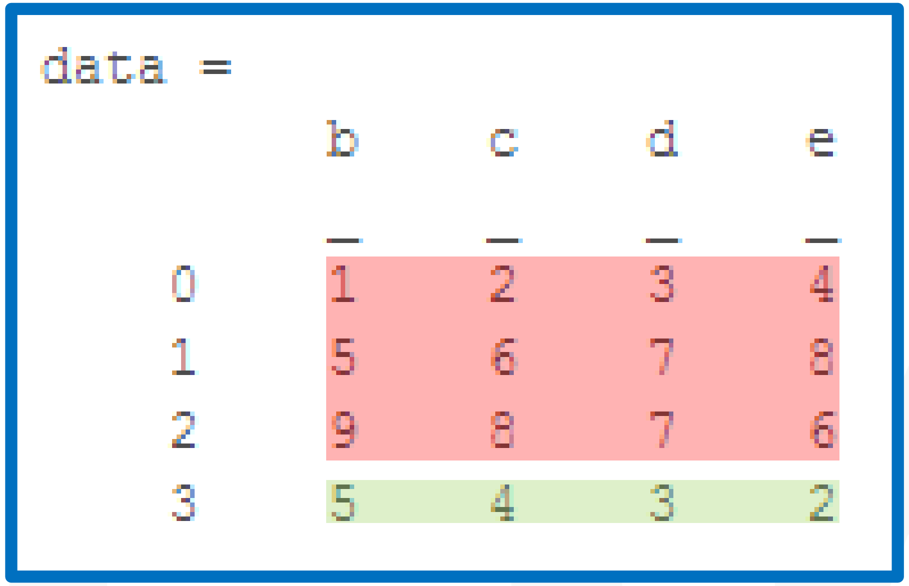
MATLAB Operators

X*Y	: matrix multiplication
X.*Y	: element-wise multiplication
X^y	: matrix power
X.^Y	: element-wise power
X/Y	: matrix right division
X./Y	: element-wise divide

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
data =

          b    c    d    e

          _    _    _    _
    0     1    2    3    4
    1     5    6    7    8
    2     9    8    7    6
    3     5    4    3    2
```

$$\underset{(3,4)}{\begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 8 & 7 & 6 \end{pmatrix}} \times \underset{(4,1)}{\begin{pmatrix} 5 \\ 4 \\ 3 \\ 2 \end{pmatrix}}$$

Use one command to compute the matrix multiplication, using the first 3 rows in **`data`** as the first matrix, and using the last row (without using the size of **`data`**) as the second 1-column matrix.

161

```
data{1:3, :}*transpose(data{end,:})
```

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

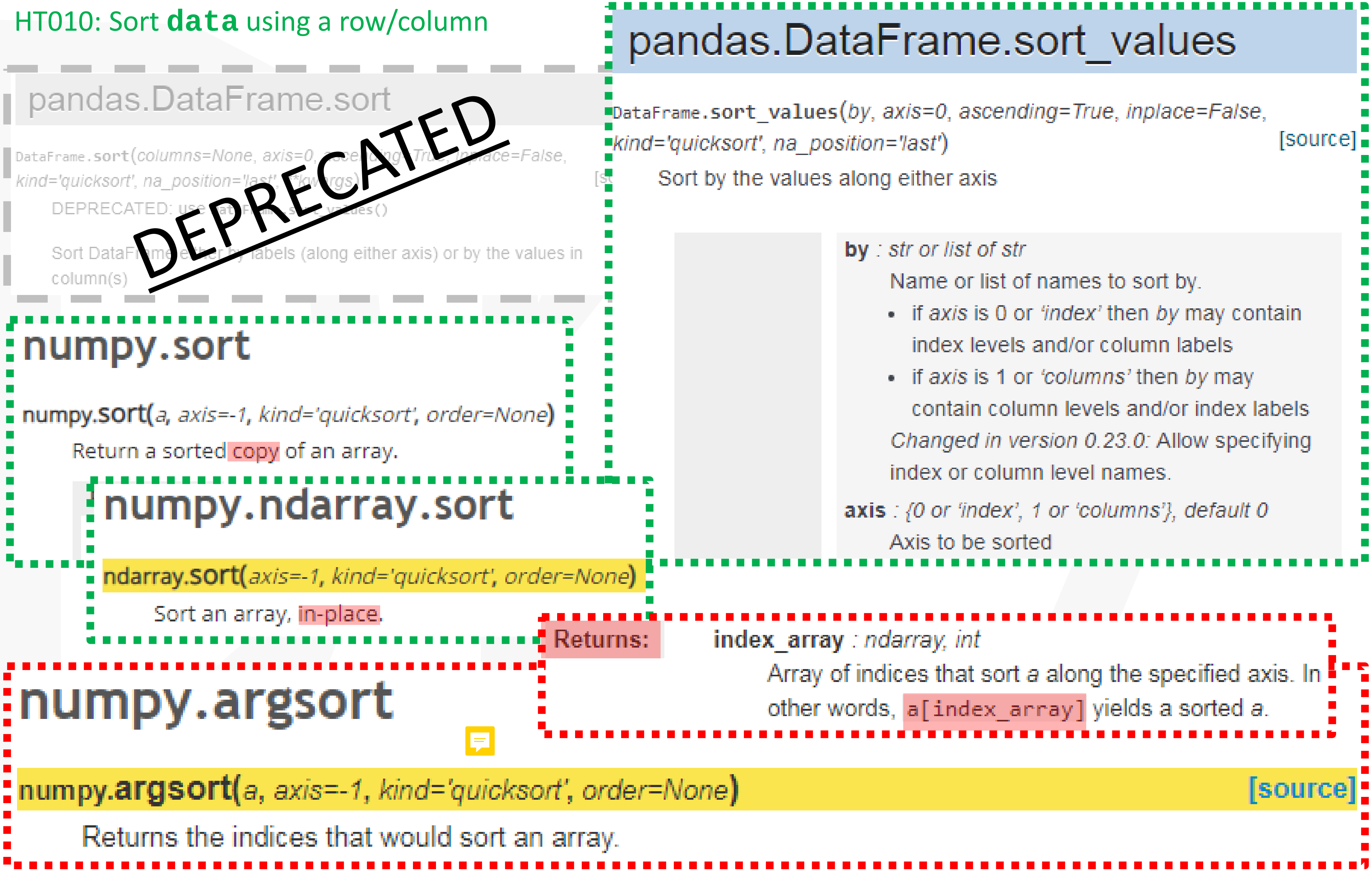
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Slide
289/307

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

HT010: Sort **data** using a row/column



pandas.DataFrame.sort

DataFrame.sort(*columns=None, axis=0, ascending=True, inplace=False, kind='quicksort', na_position='last', **kwargs*) [source]

DEPRECATED: use DataFrame.sort_values()

Sort DataFrame either by labels (along either axis) or by the values in column(s)

DEPRECATED

pandas.DataFrame.sort_values

DataFrame.**sort_values**(*by, axis=0, ascending=True, inplace=False, kind='quicksort', na_position='last'*) [source]

Sort by the values along either axis

by : *str or list of str*

Name or list of names to sort by.

- if *axis* is 0 or *'index'* then *by* may contain index levels and/or column labels
- if *axis* is 1 or *'columns'* then *by* may contain column levels and/or index labels

Changed in version 0.23.0: Allow specifying index or column level names.

axis : *{0 or 'index', 1 or 'columns'}, default 0*

Axis to be sorted

numpy.sort

numpy.sort(*a, axis=-1, kind='quicksort', order=None*)

Return a sorted copy of an array.

numpy.ndarray.sort

ndarray.sort(*axis=-1, kind='quicksort', order=None*)

Sort an array, in-place.

numpy.argsort

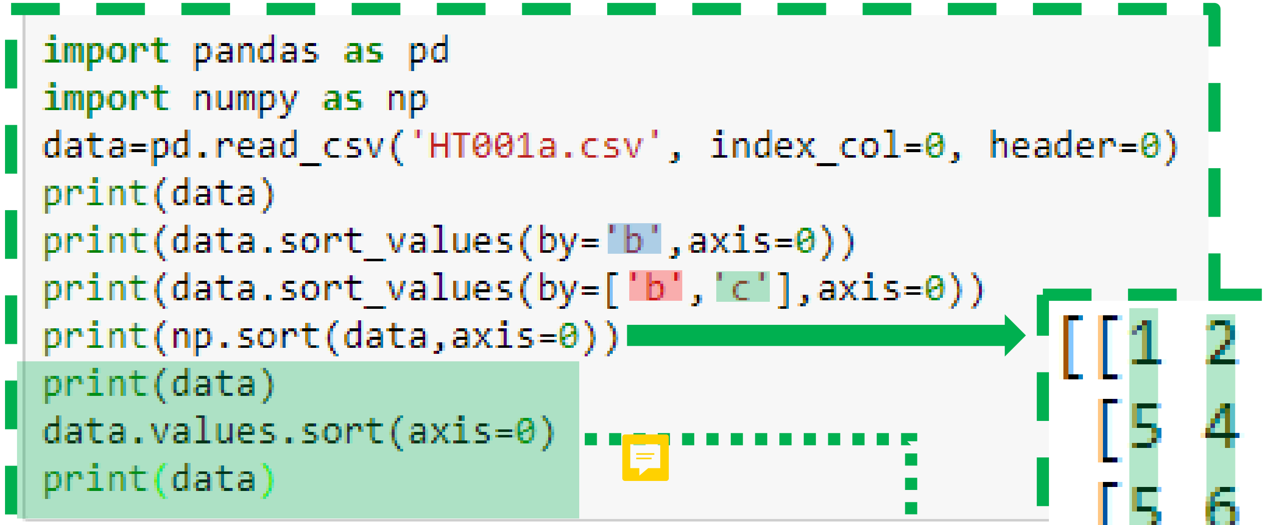
numpy.argsort(*a, axis=-1, kind='quicksort', order=None*) [source]

Returns the indices that would sort an array.

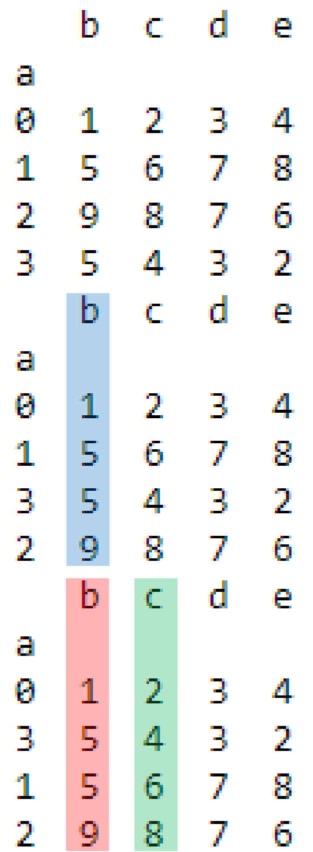
Returns: **index_array** : *ndarray, int*

Array of indices that sort *a* along the specified axis. In other words, `a[index_array]` yields a sorted *a*.

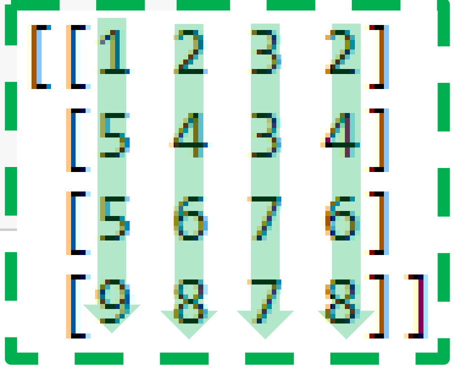
Slide 290/307



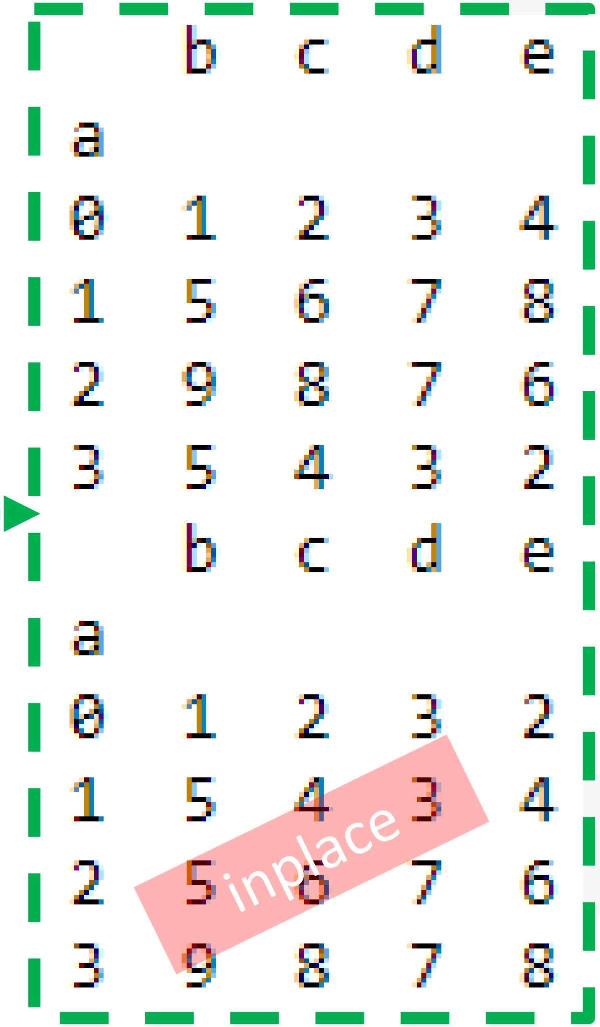
```
import pandas as pd
import numpy as np
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
print(data.sort_values(by='b',axis=0))
print(data.sort_values(by=['b','c'],axis=0))
print(np.sort(data,axis=0))
print(data)
data.values.sort(axis=0)
print(data)
```



```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
3  5  4  3  2
2  9  8  7  6
   b  c  d  e
a
0  1  2  3  4
3  5  4  3  2
1  5  6  7  8
2  9  8  7  6
```



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



```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
   b  c  d  e
a
0  1  2  3  2
1  5  4  3  4
2  5  6  7  6
3  9  8  7  8
```

- ❑ **np.sort** sort **every** row/colum.
- ❑ **np.ndarray.sort** sort, **inplace**, **every** row/colum.

pandas.DataFrame.sort_values:
Sort **data** using rows/columns.

Use one command to sort rows in ascending order using the 3rd column?

162

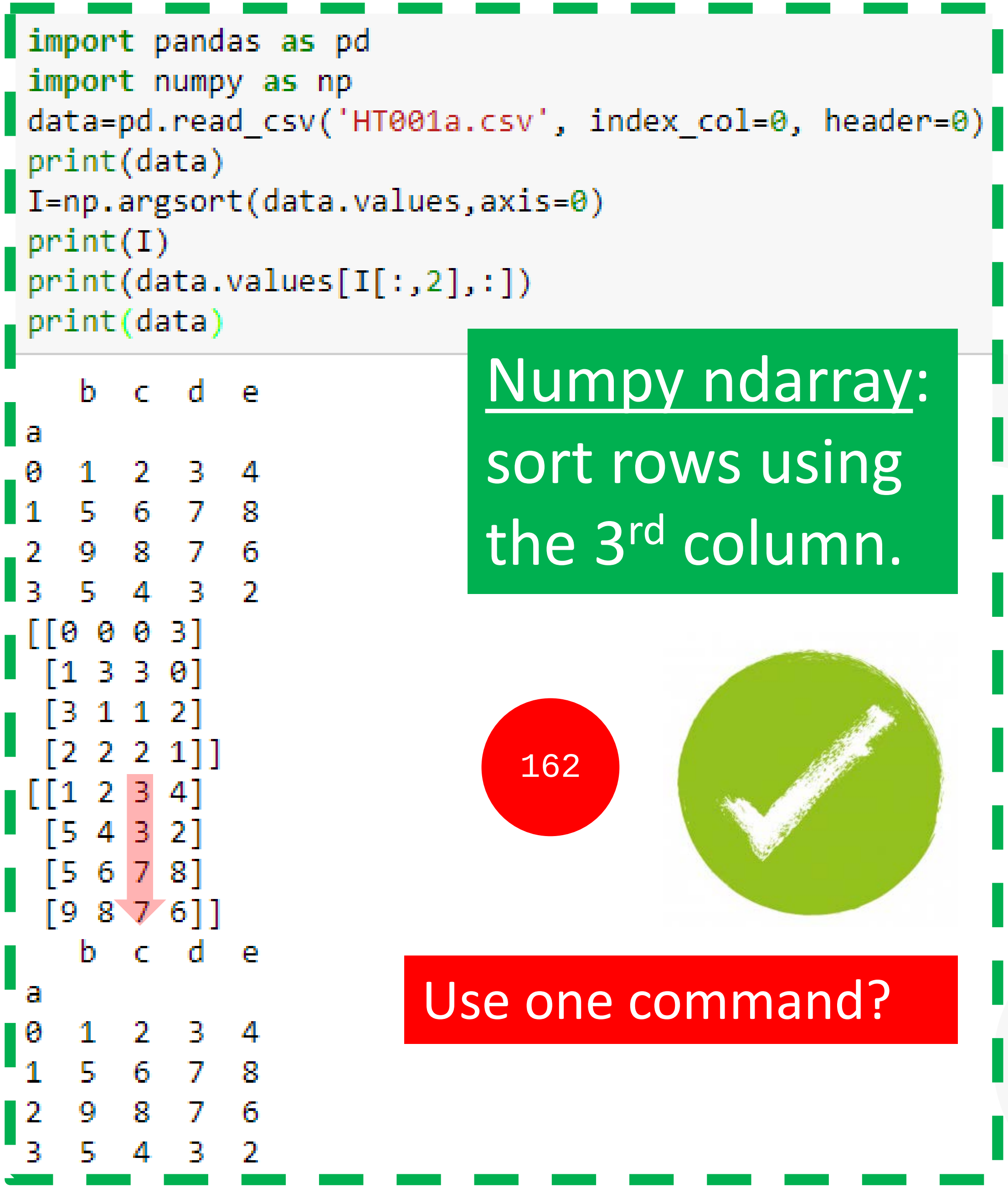
QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



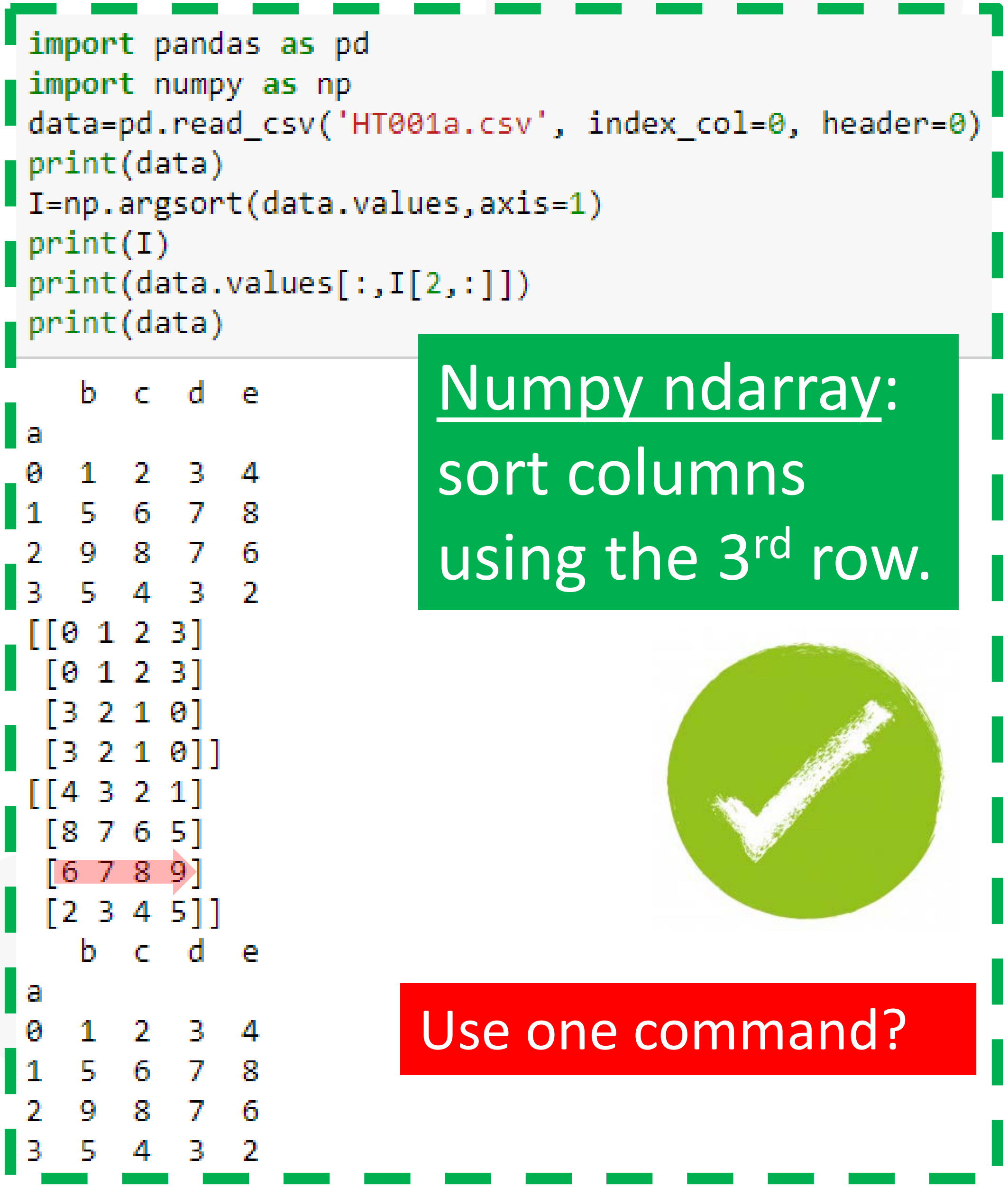
```
import pandas as pd
import numpy as np
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
I=np.argsort(data.values,axis=0)
print(I)
print(data.values[I[:,2],:])
print(data)
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
[[0 0 0 3]
 [1 3 3 0]
 [3 1 1 2]
 [2 2 2 1]]
[[1 2 3 4]
 [5 4 3 2]
 [5 6 7 8]
 [9 8 7 6]]
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
```

Numpy ndarray: sort rows using the 3rd column.

162

Use one command?



```
import pandas as pd
import numpy as np
data=pd.read_csv('HT001a.csv', index_col=0, header=0)
print(data)
I=np.argsort(data.values,axis=1)
print(I)
print(data.values[:,I[2,:]])
print(data)
```

```
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
[[0 1 2 3]
 [0 1 2 3]
 [3 2 1 0]
 [3 2 1 0]]
[[4 3 2 1]
 [8 7 6 5]
 [6 7 8 9]
 [2 3 4 5]]
   b  c  d  e
a
0  1  2  3  4
1  5  6  7  8
2  9  8  7  6
3  5  4  3  2
```

Numpy ndarray: sort columns using the 3rd row.

Use one command?

HT010: Sort `data` using a row/column

✓ `sort` (total row/column sort)

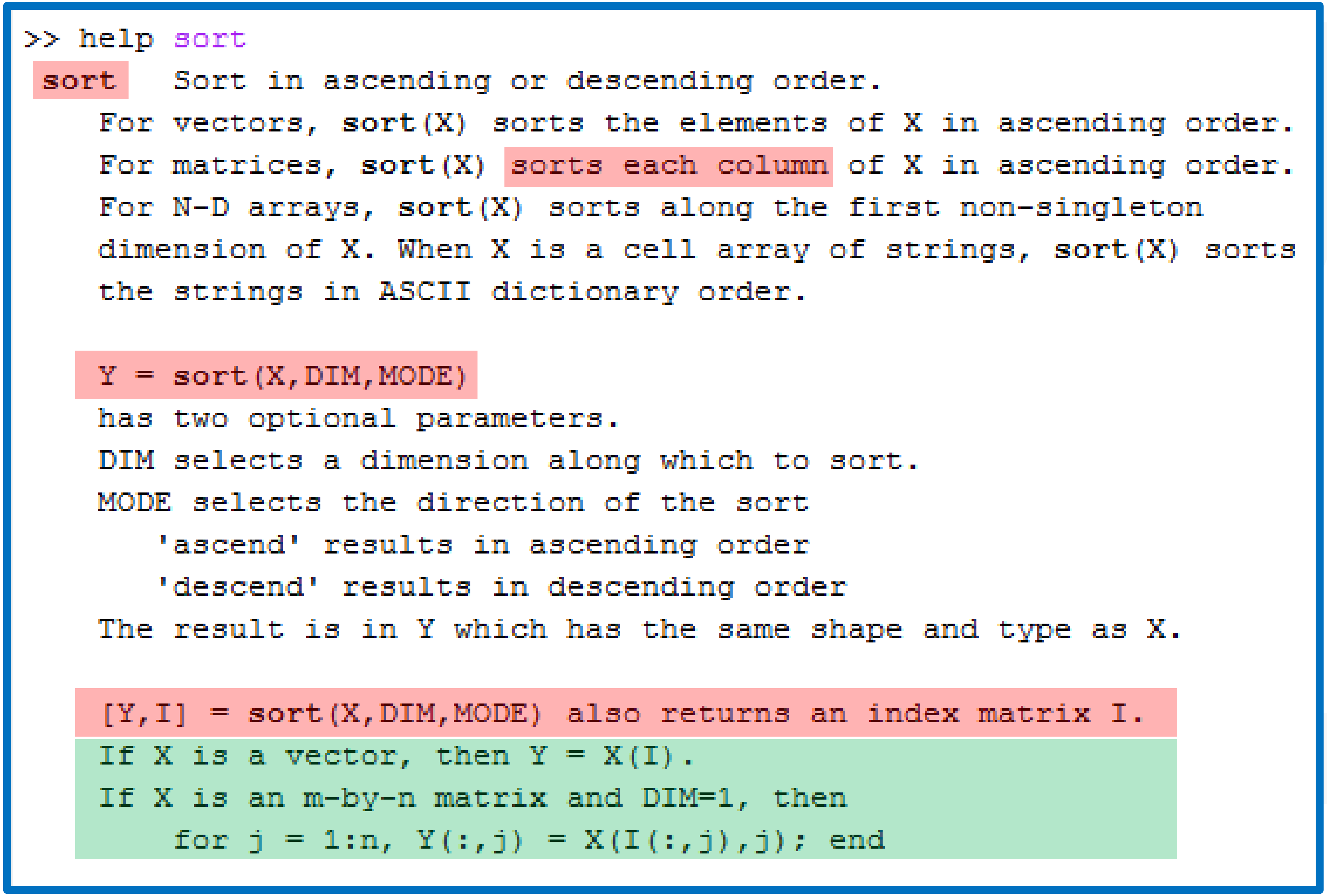
✓ `sortrows` (sort rows)

`sort` and `sortrows` are functions on matrices. `sort` does not work on table, `sortrows` works on table.

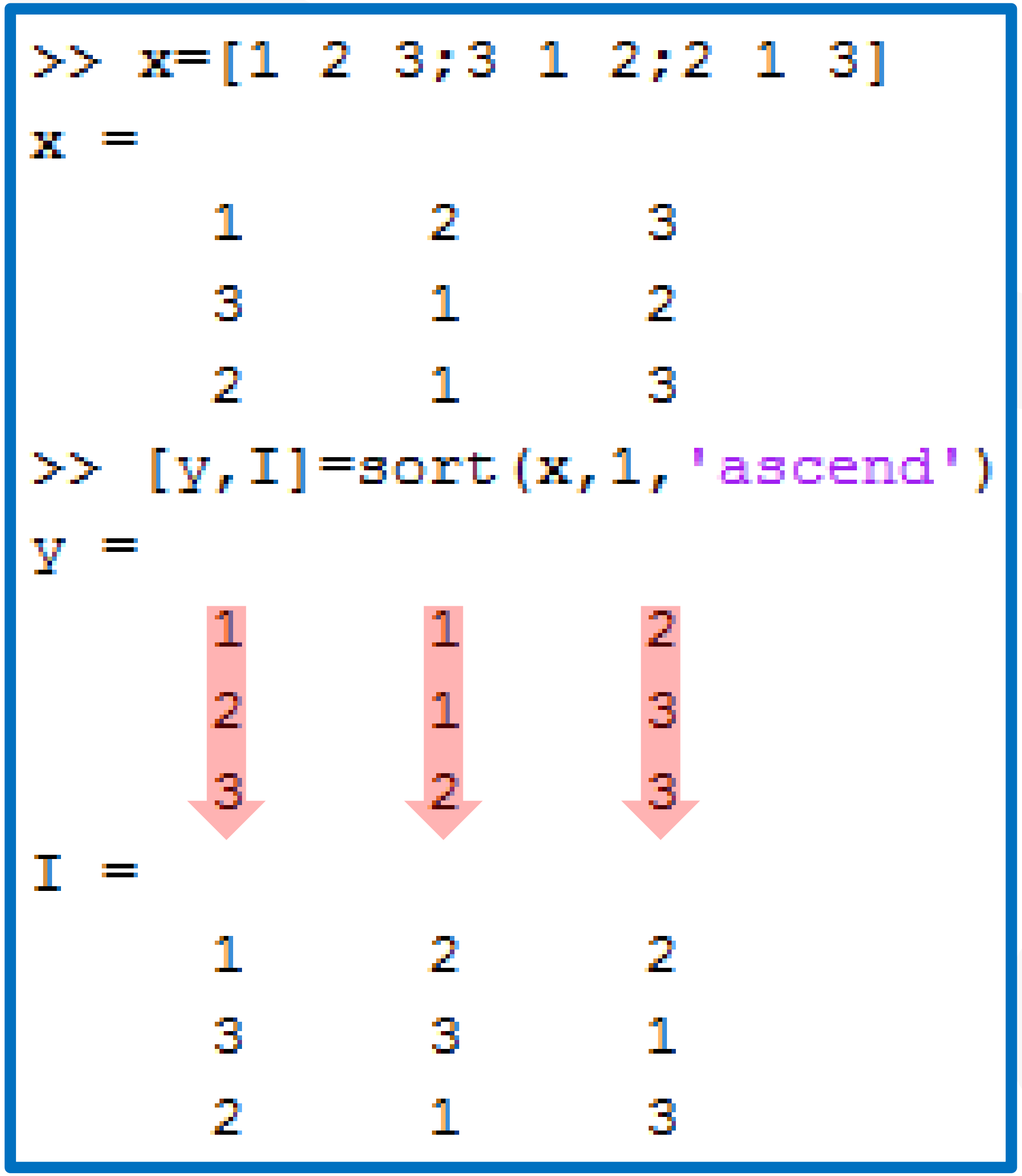
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

QF666
Programming and
Computational
Finance

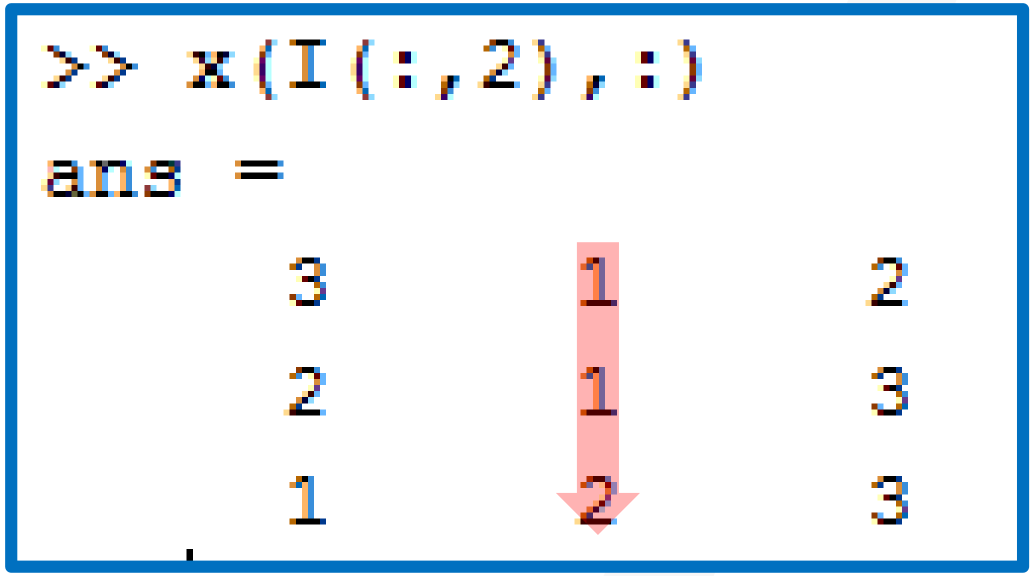
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
>> help sort
 sort    Sort in ascending or descending order.
    For vectors, sort(X) sorts the elements of X in ascending order.
    For matrices, sort(X) sorts each column of X in ascending order.
    For N-D arrays, sort(X) sorts along the first non-singleton
    dimension of X. When X is a cell array of strings, sort(X) sorts
    the strings in ASCII dictionary order.

    Y = sort(X,DIM,MODE)
    has two optional parameters.
    DIM selects a dimension along which to sort.
    MODE selects the direction of the sort
       'ascend' results in ascending order
       'descend' results in descending order
    The result is in Y which has the same shape and type as X.

    [Y,I] = sort(X,DIM,MODE) also returns an index matrix I.
    If X is a vector, then Y = X(I).
    If X is an m-by-n matrix and DIM=1, then
        for j = 1:n, Y(:,j) = X(I(:,j),j); end
```



```
>> x=[1 2 3;3 1 2;2 1 3]
x =
     1     2     3
     3     1     2
     2     1     3
>> [y,I]=sort(x,1,'ascend')
y =
     1     1     2
     2     1     3
     3     2     3
I =
     1     2     2
     3     3     1
     2     1     3
```



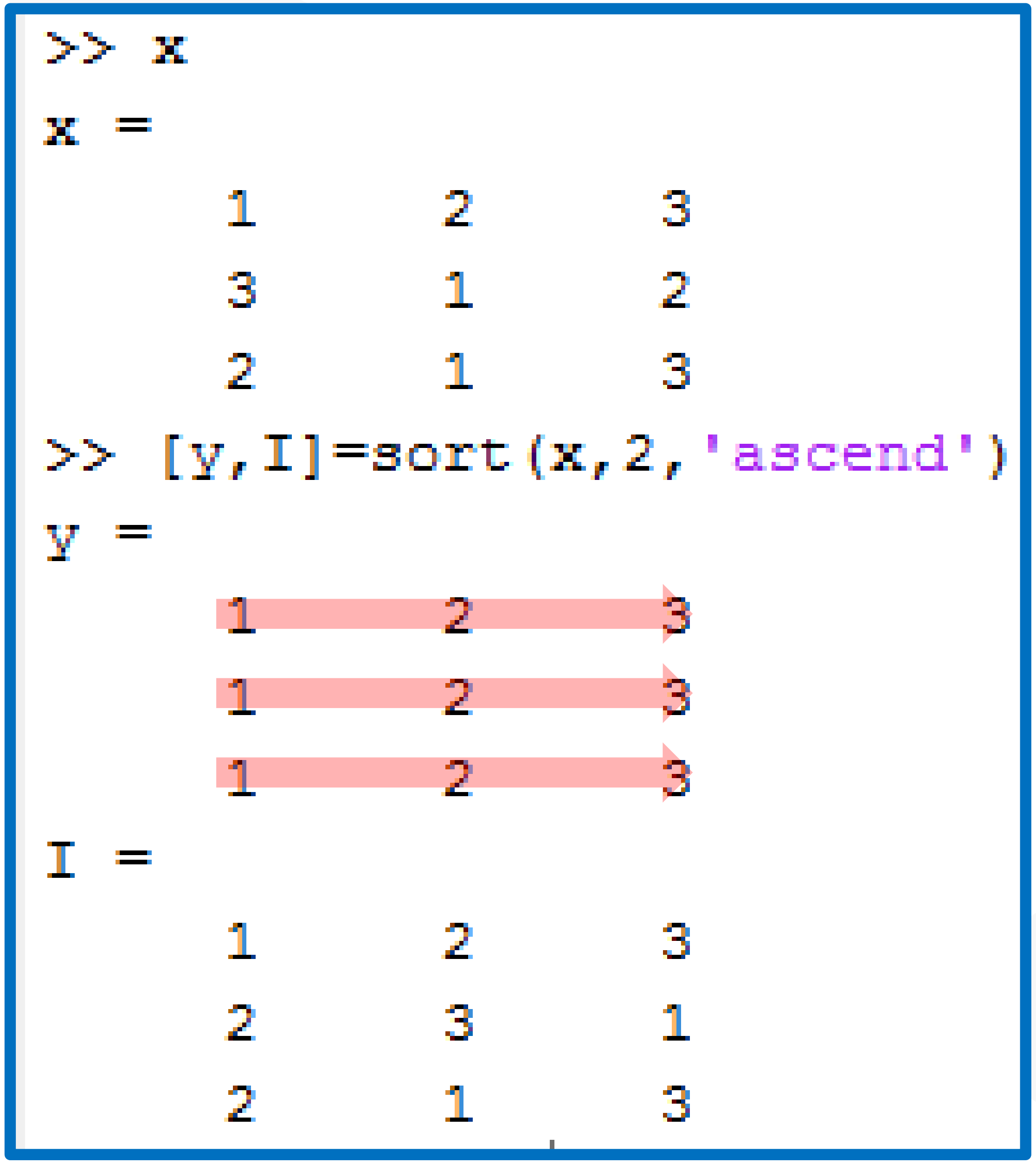
```
>> x(I(:,2),:)
ans =
     3     1     2
     2     1     3
     1     2     3
```

`x(I(:,2),:)` sort **`x`** using the second column.

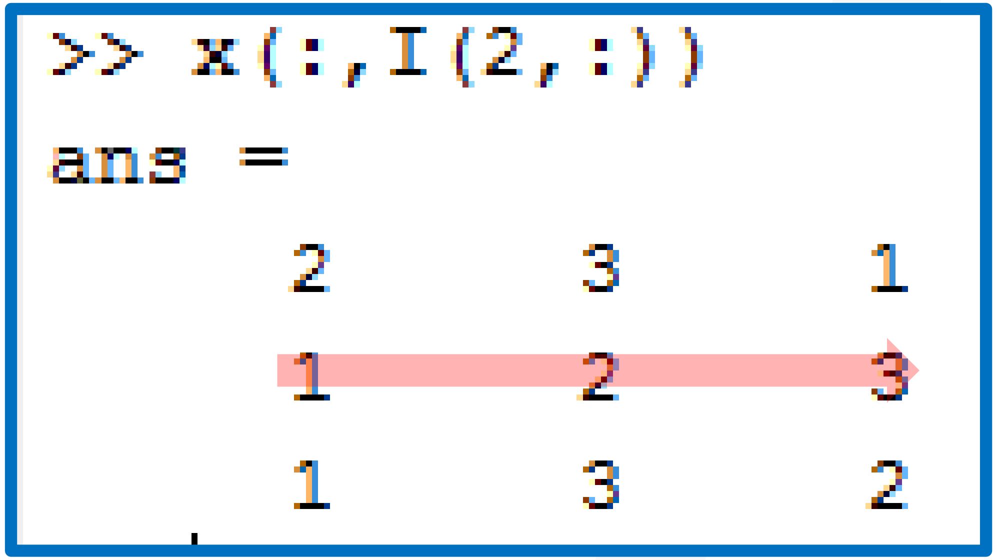
SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
>> x

x =

     1     2     3
     3     1     2
     2     1     3

>> [y,I]=sort(x,2,'ascend')

y =

     1     2     3
     1     2     3
     1     2     3

I =

     1     2     3
     2     3     1
     2     1     3
```



```
>> x(:,I(2,:))

ans =

     2     3     1
     1     2     3
     1     3     2
```

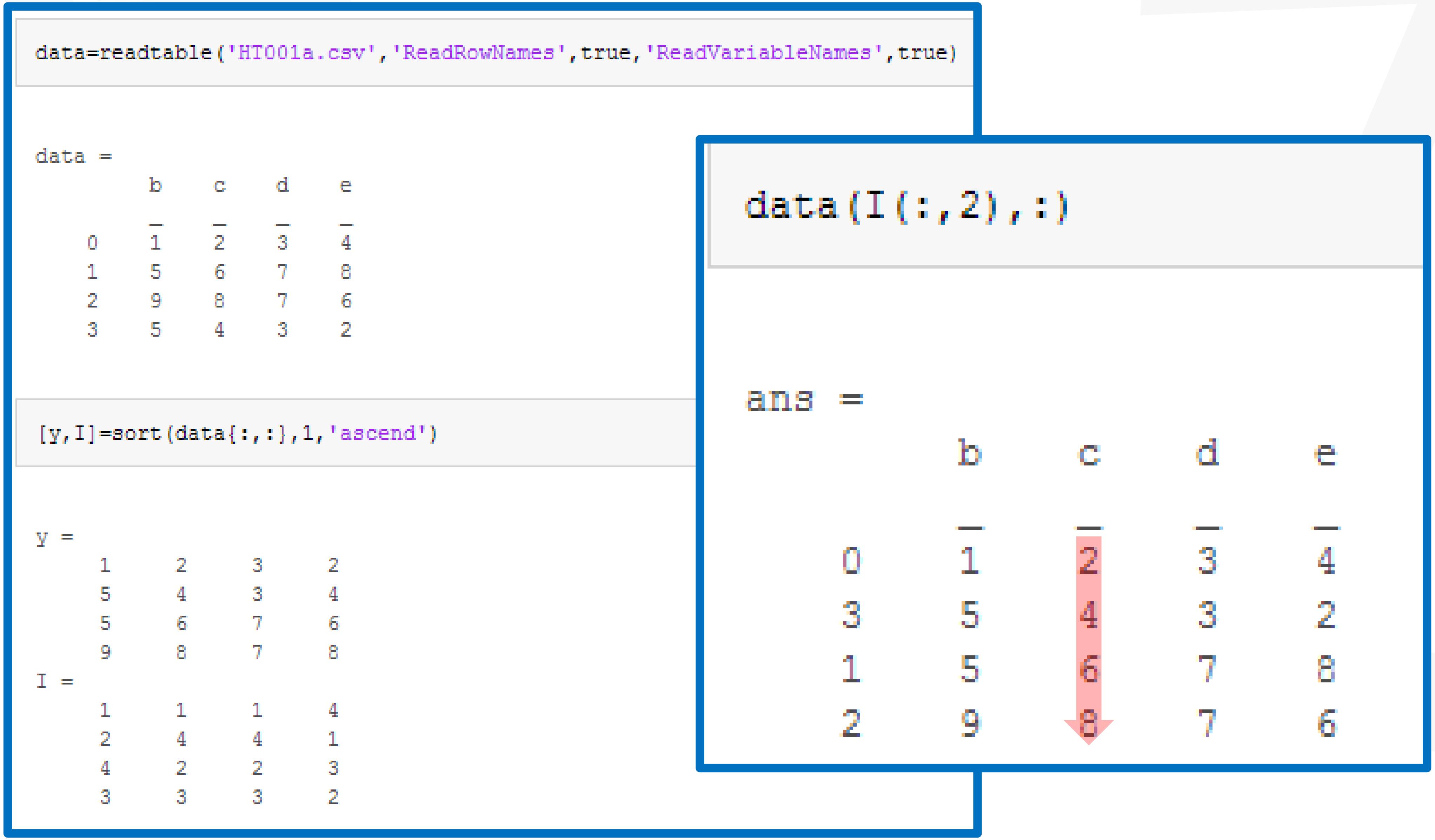
`x(:,I(2,:))` sort `x` using the second row.

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

⇨How to use **sort** to sort **data**?



```
data=readtable('HT001a.csv','ReadRowNames',true,'ReadVariableNames',true)
```

```
data =
             b      c      d      e
             _      _      _      _
      0      1      2      3      4
      1      5      6      7      8
      2      9      8      7      6
      3      5      4      3      2
```

```
[y,I]=sort(data{:,:},1,'ascend')
```

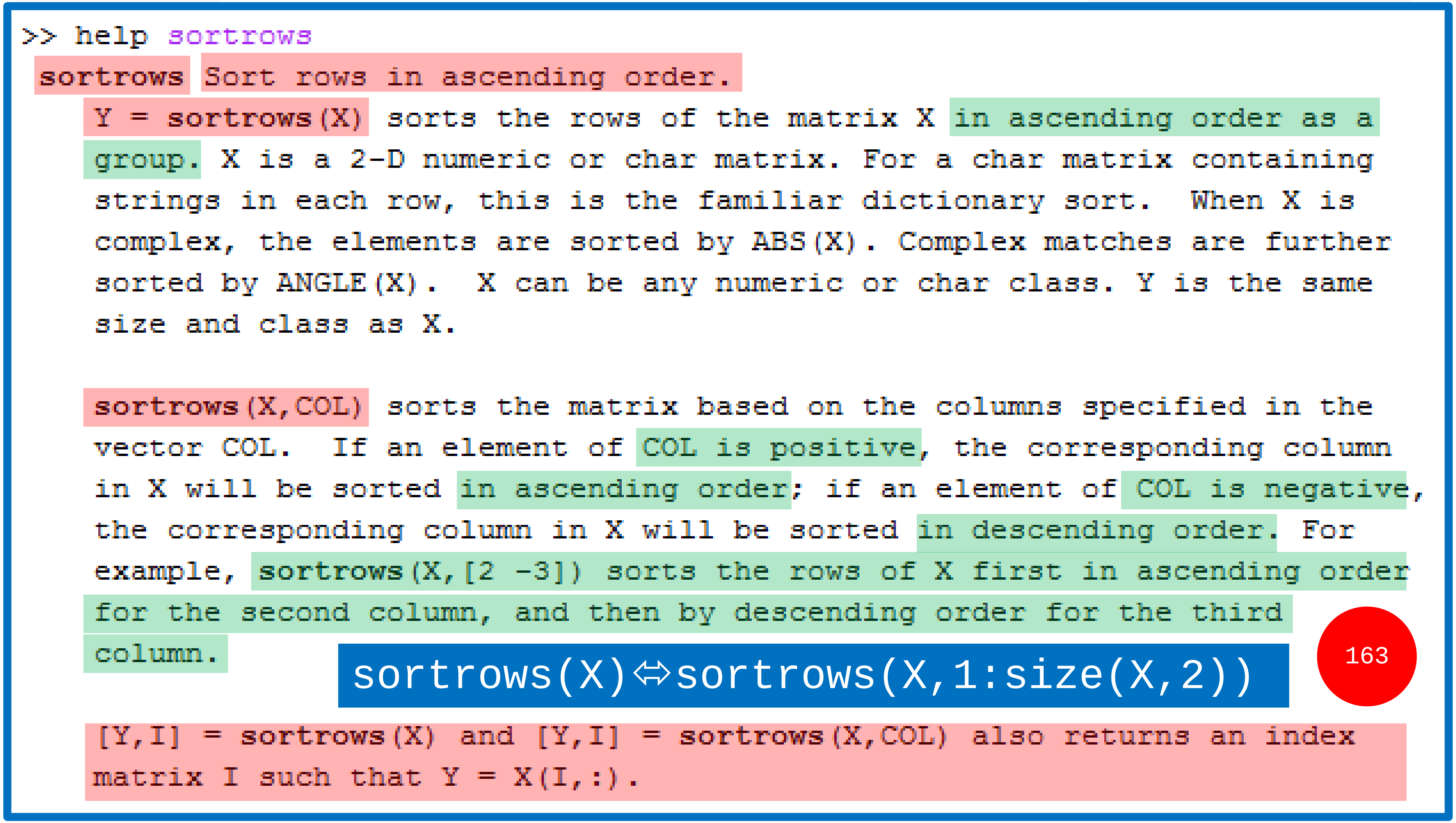
```
y =
      1      2      3      2
      5      4      3      4
      5      6      7      6
      9      8      7      8
I =
      1      1      1      4
      2      4      4      1
      4      2      2      3
      3      3      3      2
```

```
data(I(:,2),:)
```

```
ans =
             b      c      d      e
             _      _      _      _
      0      1      2      3      4
      3      5      4      3      2
      1      5      6      7      8
      2      9      8      7      6
```

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
>> help sortrows
 sortrows Sort rows in ascending order.
    Y = sortrows(X) sorts the rows of the matrix X in ascending order as a
    group. X is a 2-D numeric or char matrix. For a char matrix containing
    strings in each row, this is the familiar dictionary sort.  When X is
    complex, the elements are sorted by ABS(X). Complex matches are further
    sorted by ANGLE(X).  X can be any numeric or char class. Y is the same
    size and class as X.

    sortrows(X,COL) sorts the matrix based on the columns specified in the
    vector COL.  If an element of COL is positive, the corresponding column
    in X will be sorted in ascending order; if an element of COL is negative,
    the corresponding column in X will be sorted in descending order. For
    example, sortrows(X,[2 -3]) sorts the rows of X first in ascending order
    for the second column, and then by descending order for the third
    column.

    [Y,I] = sortrows(X) and [Y,I] = sortrows(X,COL) also returns an index
    matrix I such that Y = X(I,:).
```

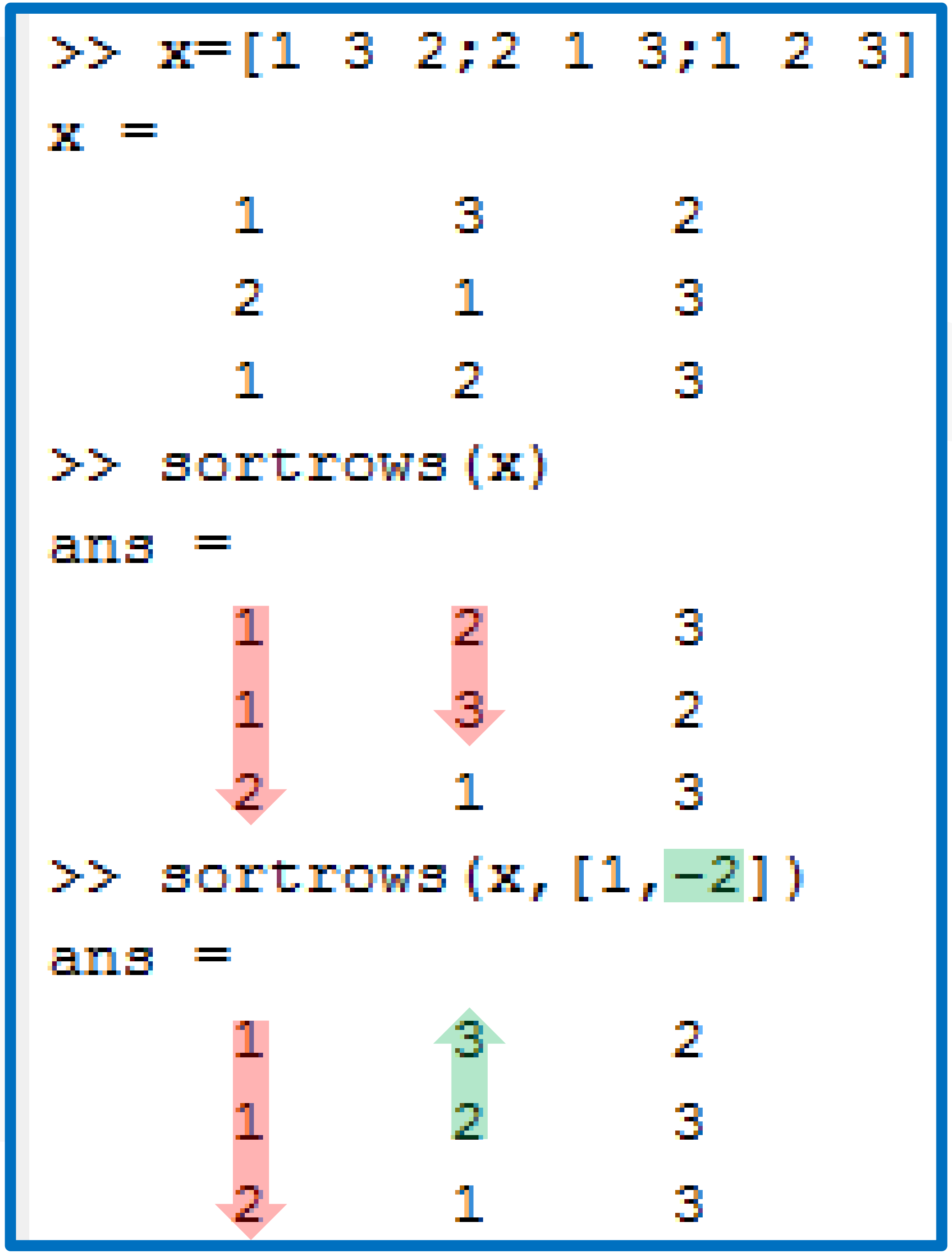
sortrows(X)⇔sortrows(X,1:size(X,2))

163

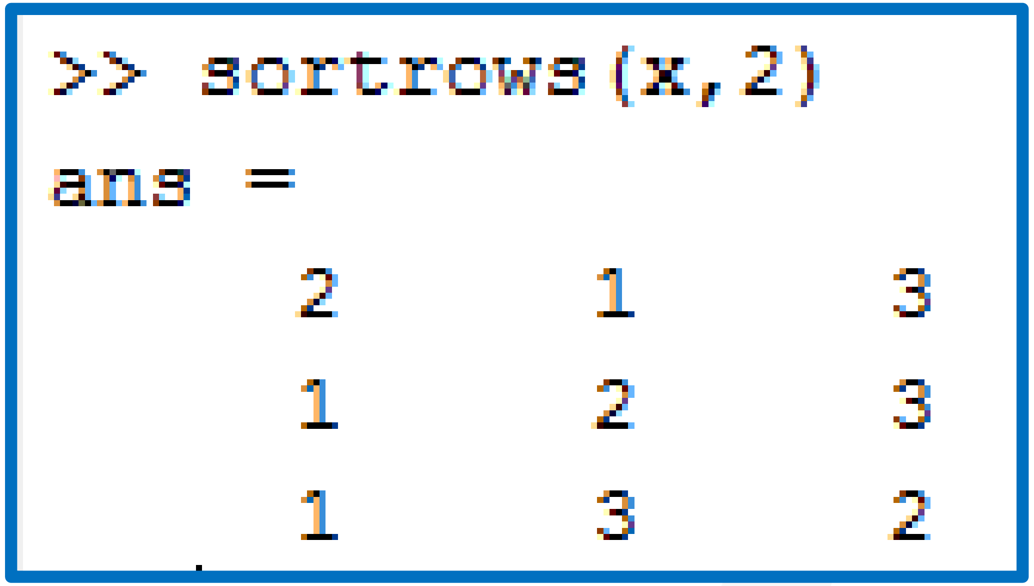
SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
>> x=[1 3 2;2 1 3;1 2 3]
x =
     1     3     2
     2     1     3
     1     2     3
>> sortrows(x)
ans =
     1     2     3
     1     3     2
     2     1     3
>> sortrows(x,[1,-2])
ans =
     1     3     2
     1     2     3
     2     1     3
```



```
>> sortrows(x,2)
ans =
     2     1     3
     1     2     3
     1     3     2
```

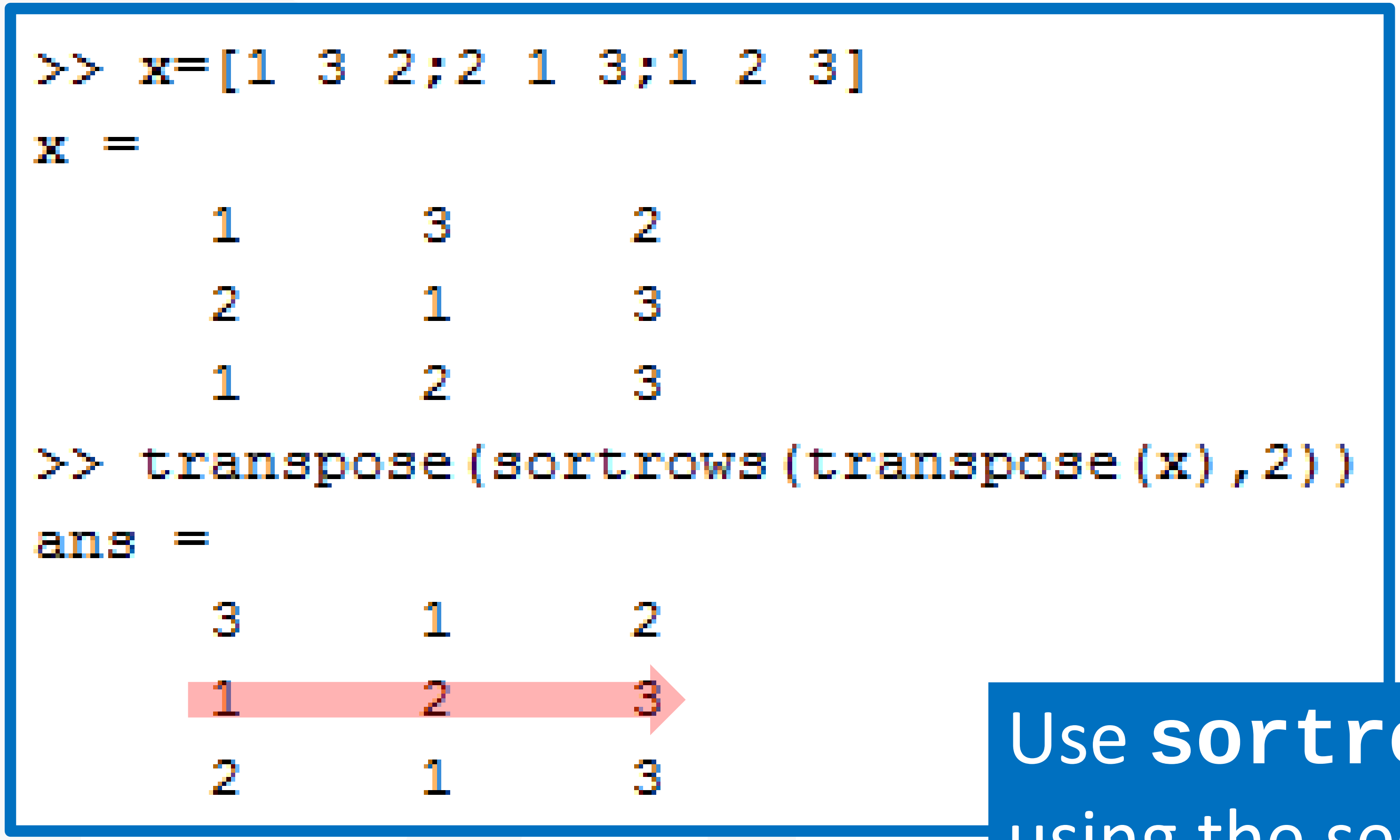
`sortrows(x,2)` sort `x` using the second column.

Q: How to use `sortrows` to sort `x` using the second row?
[Hint: `transpose(x)`]

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
>> x=[1 3 2;2 1 3;1 2 3]

x =

     1     3     2
     2     1     3
     1     2     3

>> transpose(sortrows(transpose(x),2))

ans =

     3     1     2
     1     2     3
     2     1     3
```

Use **`sortrows`** to sort **`x`** using the second row?

QF666 Programming and Computational Finance

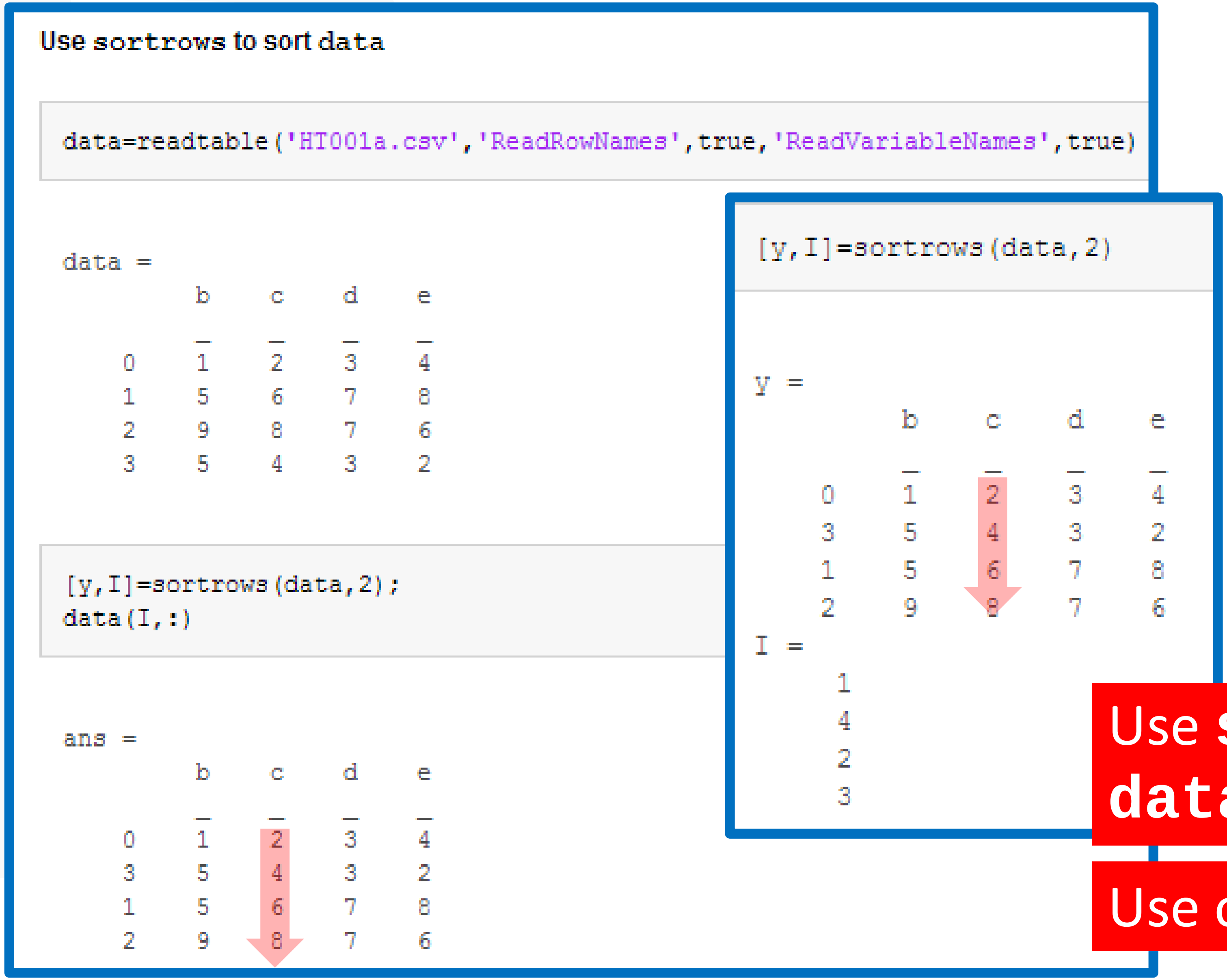
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Slide
300/307

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

⇨How to use **sortrows** to sort **data**?



Use sortrows to sort data

```
data=readtable('HT001a.csv','ReadRowNames',true,'ReadVariableNames',true)
```

```
data =
             b     c     d     e

             _     _     _     _
       0     1     2     3     4
       1     5     6     7     8
       2     9     8     7     6
       3     5     4     3     2
```

```
[y,I]=sortrows(data,2);
data(I,:)
```

```
ans =
             b     c     d     e

             _     _     _     _
       0     1     2     3     4
       3     5     4     3     2
       1     5     9     7     8
       2     9     8     7     6
```

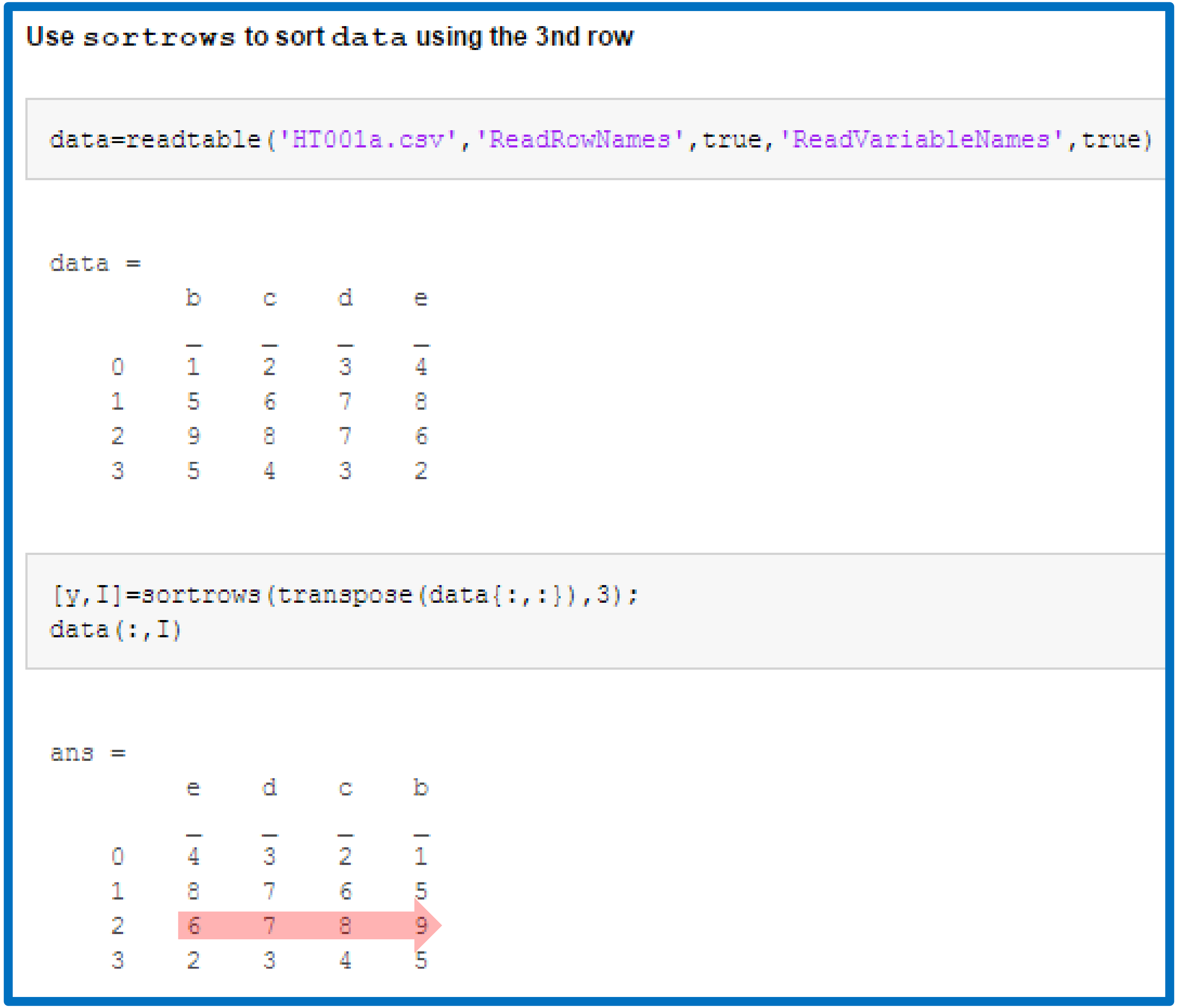
```
[y,I]=sortrows(data,2)
```

```
y =
             b     c     d     e

             _     _     _     _
       0     1     2     3     4
       3     5     4     3     2
       1     5     6     7     8
       2     9     8     7     6
I =
        1
        4
        2
        3
```

164

Use **sortrows** to sort **data** using a row?

Use one command?



Use sortrows to sort data using the 3nd row

```
data=readtable('HT001a.csv','ReadRowNames',true,'ReadVariableNames',true)
```

```
data =
                b    c    d    e

                _    _    _    _
        0       1    2    3    4
        1       5    6    7    8
        2       9    8    7    6
        3       5    4    3    2
```

```
[y,I]=sortrows(transpose(data{:,:}),3);
data(:,I)
```

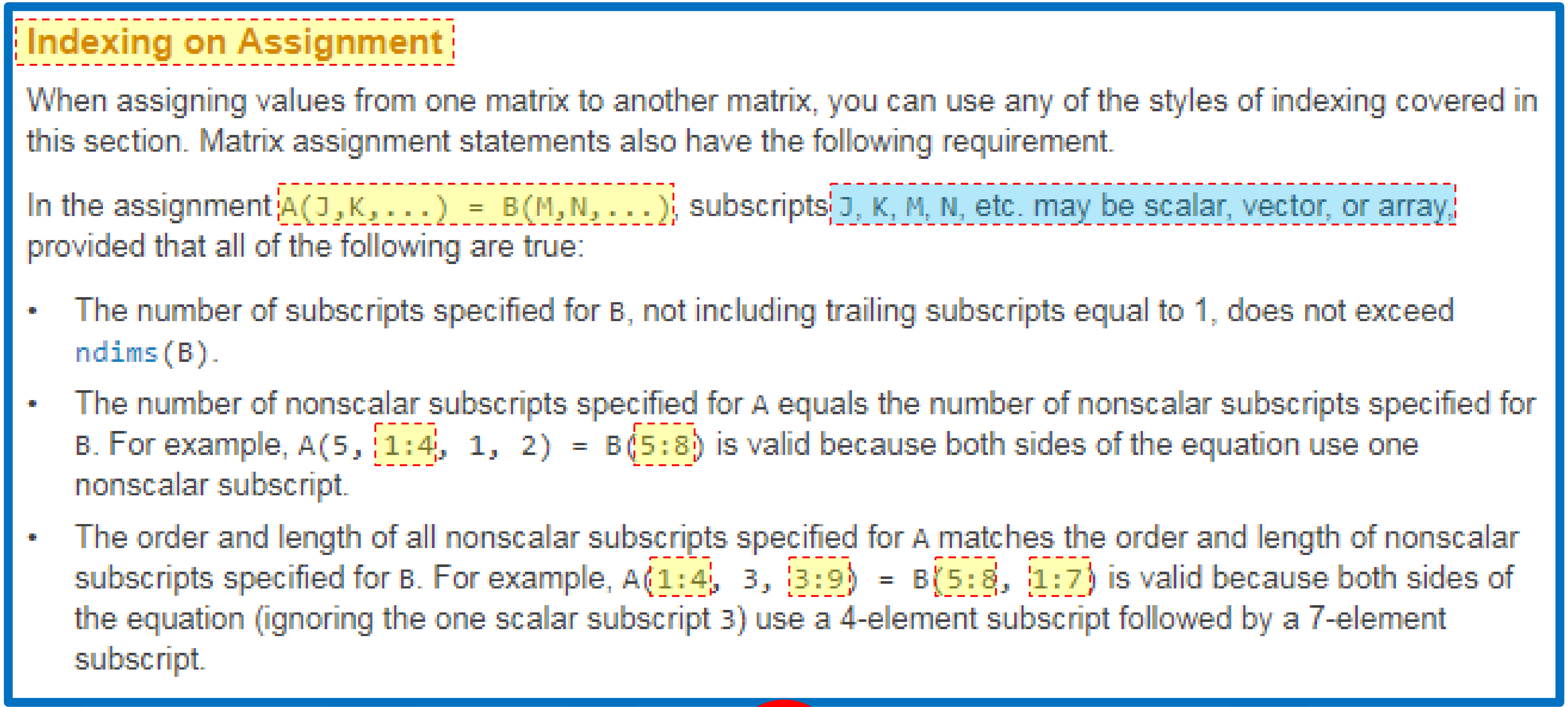
```
ans =
                e    d    c    b

                _    _    _    _
        0       4    3    2    1
        1       8    7    6    5
        2       6    7    8    9
        3       2    3    4    5
```

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

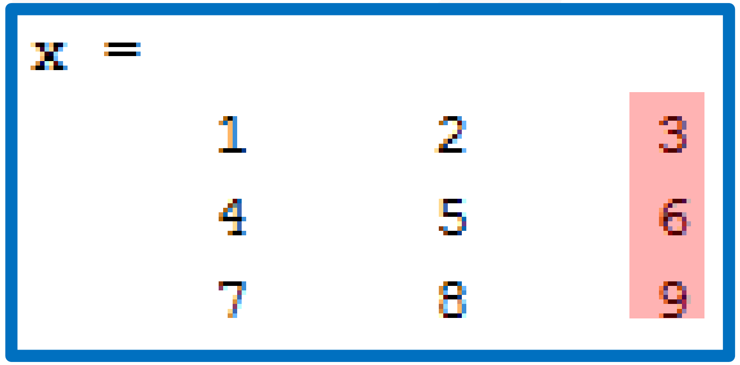


Indexing on Assignment

When assigning values from one matrix to another matrix, you can use any of the styles of indexing covered in this section. Matrix assignment statements also have the following requirement.

In the assignment `A(J,K,...) = B(M,N,...)`, subscripts `J`, `K`, `M`, `N`, etc. may be scalar, vector, or array, provided that all of the following are true:

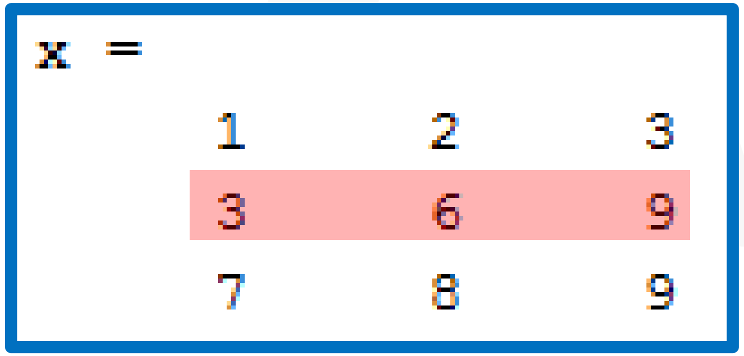
- The number of subscripts specified for `B`, not including trailing subscripts equal to 1, does not exceed `ndims(B)`.
- The number of nonscalar subscripts specified for `A` equals the number of nonscalar subscripts specified for `B`. For example, `A(5, 1:4, 1, 2) = B(5:8)` is valid because both sides of the equation use one nonscalar subscript.
- The order and length of all nonscalar subscripts specified for `A` matches the order and length of nonscalar subscripts specified for `B`. For example, `A(1:4, 3, 3:9) = B(5:8, 1:7)` is valid because both sides of the equation (ignoring the one scalar subscript `3`) use a 4-element subscript followed by a 7-element subscript.



```
x =

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

166

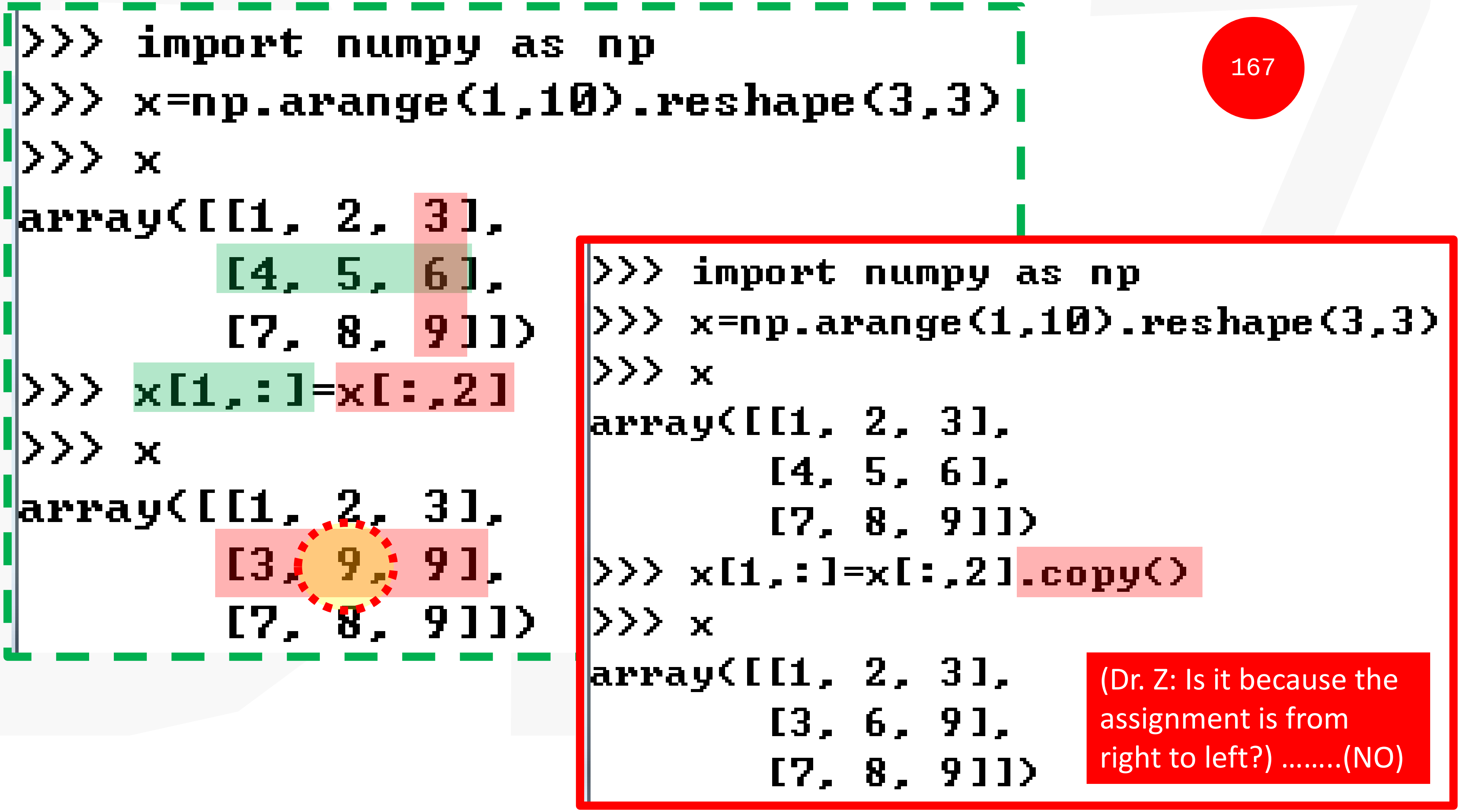
x(2,:)=x(:,3)



```
x =

     1     2     3
     3     6     9
     7     8     9
```

Caution: Numpy Array (Slicings are Views)

167



```
>>> import numpy as np
>>> x=np.arange(1,10).reshape(3,3)
>>> x
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]])
>>> x[1,:]=x[:,2]
>>> x
array([[1, 2, 3],
       [3, 9, 9],
       [7, 8, 9]])
```

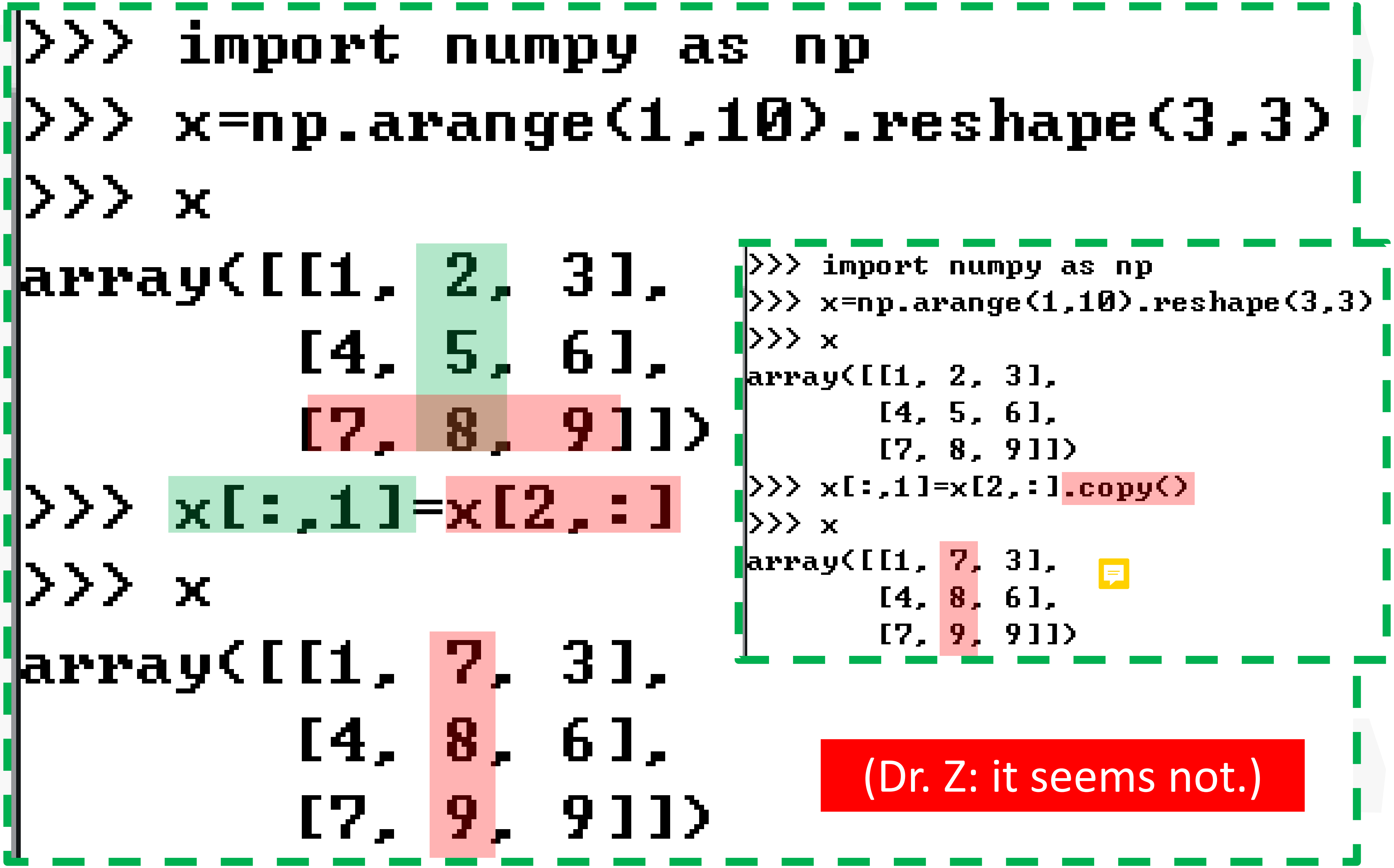
```
>>> import numpy as np
>>> x=np.arange(1,10).reshape(3,3)
>>> x
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]])
>>> x[1,:]=x[:,2].copy()
>>> x
array([[1, 2, 3],
       [3, 6, 9],
       [7, 8, 9]])
```

(Dr. Z: Is it because the assignment is from right to left?) ....(NO)

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



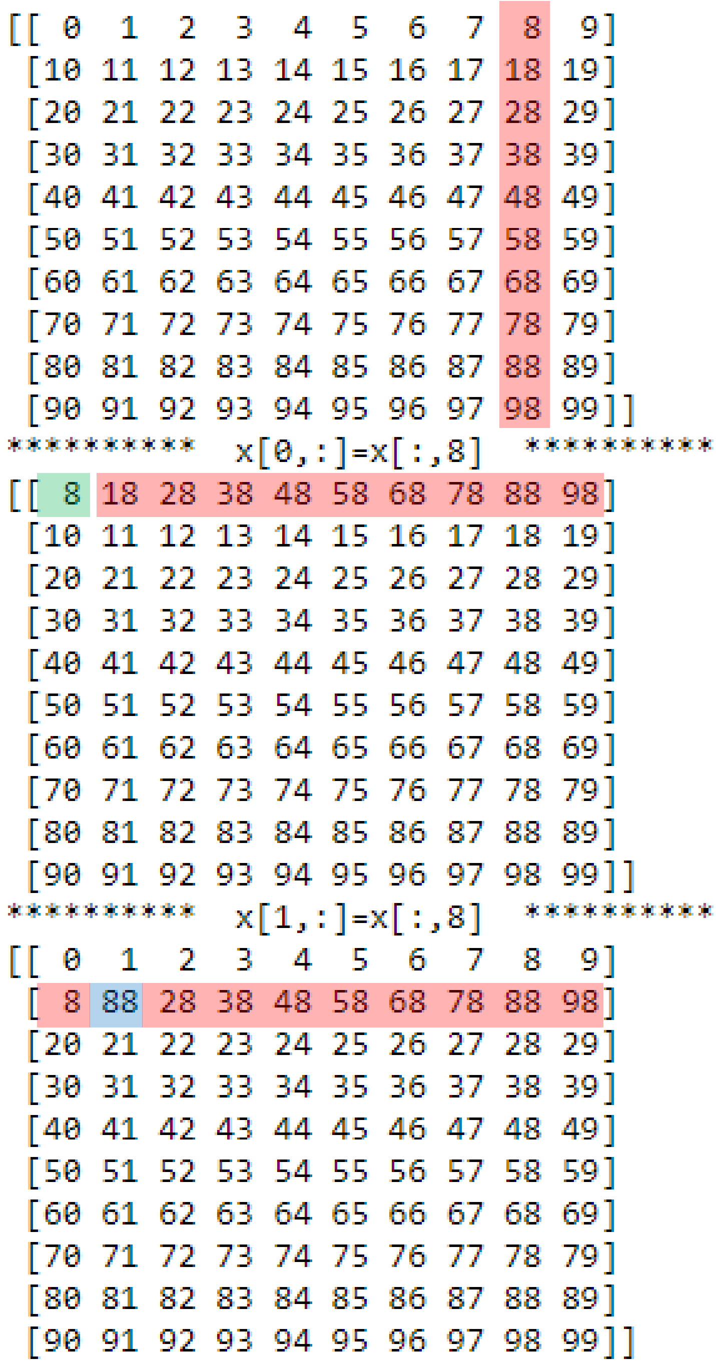
```
>>> import numpy as np
>>> x=np.arange(1,10).reshape(3,3)
>>> x
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]])
>>> x[:,1]=x[2,:]
>>> x
array([[1, 7, 3],
       [4, 8, 6],
       [7, 9, 9]])
```

```
>>> import numpy as np
>>> x=np.arange(1,10).reshape(3,3)
>>> x
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]])
>>> x[:,1]=x[2,:].copy()
>>> x
array([[1, 7, 3],
       [4, 8, 6],
       [7, 9, 9]])
```

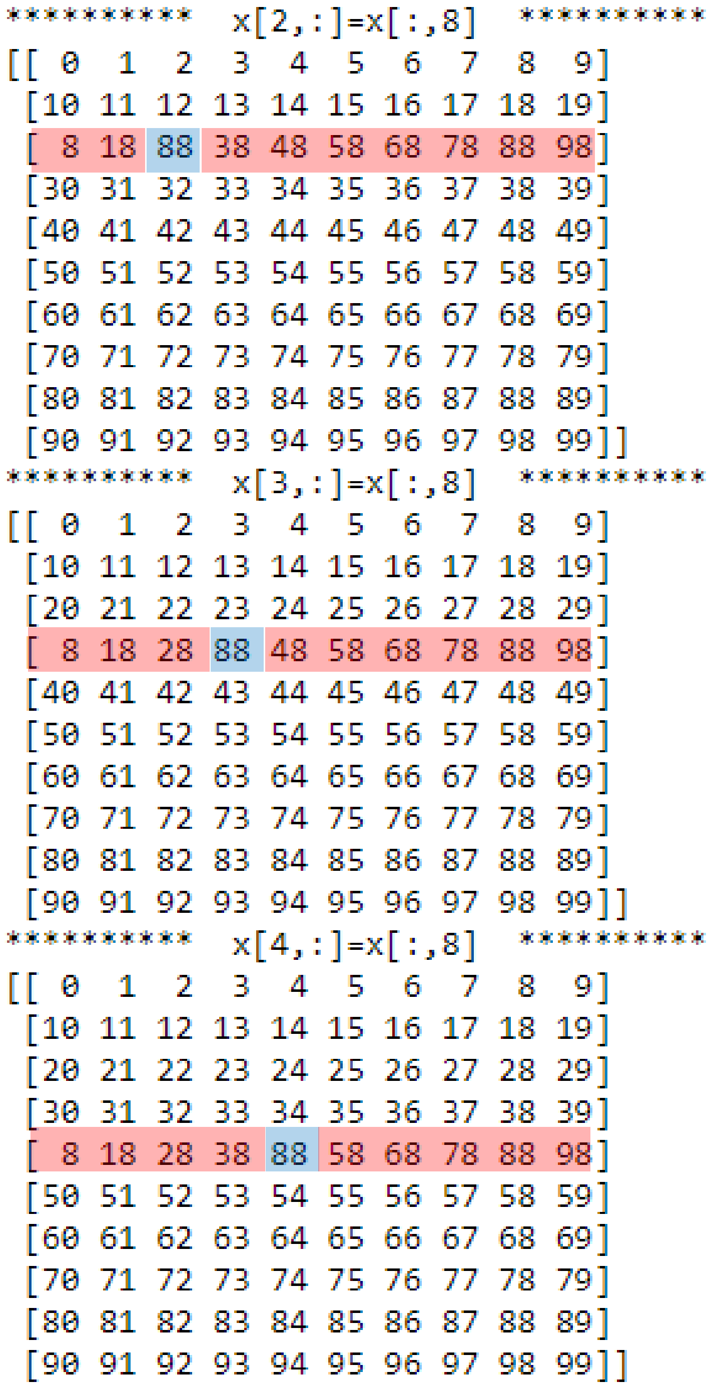
(Dr. Z: it seems not.)

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
[[ 0  1  2  3  4  5  6  7  8  9]
 [10 11 12 13 14 15 16 17 18 19]
 [20 21 22 23 24 25 26 27 28 29]
 [30 31 32 33 34 35 36 37 38 39]
 [40 41 42 43 44 45 46 47 48 49]
 [50 51 52 53 54 55 56 57 58 59]
 [60 61 62 63 64 65 66 67 68 69]
 [70 71 72 73 74 75 76 77 78 79]
 [80 81 82 83 84 85 86 87 88 89]
 [90 91 92 93 94 95 96 97 98 99]]
**********  x[0,:]=x[:,8]  **********
[[ 8 18 28 38 48 58 68 78 88 98]
 [10 11 12 13 14 15 16 17 18 19]
 [20 21 22 23 24 25 26 27 28 29]
 [30 31 32 33 34 35 36 37 38 39]
 [40 41 42 43 44 45 46 47 48 49]
 [50 51 52 53 54 55 56 57 58 59]
 [60 61 62 63 64 65 66 67 68 69]
 [70 71 72 73 74 75 76 77 78 79]
 [80 81 82 83 84 85 86 87 88 89]
 [90 91 92 93 94 95 96 97 98 99]]
**********  x[1,:]=x[:,8]  **********
[[ 0  1  2  3  4  5  6  7  8  9]
 [ 8 88 28 38 48 58 68 78 88 98]
 [20 21 22 23 24 25 26 27 28 29]
 [30 31 32 33 34 35 36 37 38 39]
 [40 41 42 43 44 45 46 47 48 49]
 [50 51 52 53 54 55 56 57 58 59]
 [60 61 62 63 64 65 66 67 68 69]
 [70 71 72 73 74 75 76 77 78 79]
 [80 81 82 83 84 85 86 87 88 89]
 [90 91 92 93 94 95 96 97 98 99]]
```

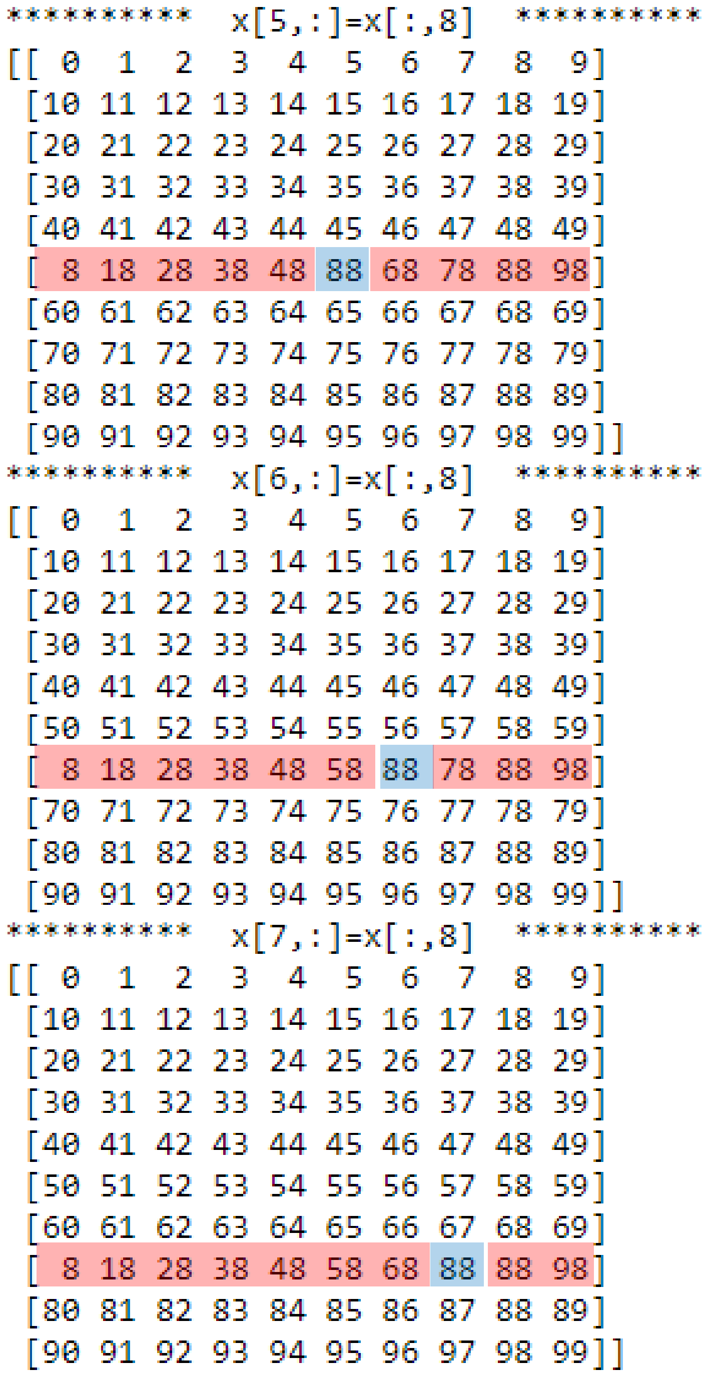


```
**********  x[2,:]=x[:,8]  **********
[[ 0  1  2  3  4  5  6  7  8  9]
 [10 11 12 13 14 15 16 17 18 19]
 [ 8 18 88 38 48 58 68 78 88 98]
 [30 31 32 33 34 35 36 37 38 39]
 [40 41 42 43 44 45 46 47 48 49]
 [50 51 52 53 54 55 56 57 58 59]
 [60 61 62 63 64 65 66 67 68 69]
 [70 71 72 73 74 75 76 77 78 79]
 [80 81 82 83 84 85 86 87 88 89]
 [90 91 92 93 94 95 96 97 98 99]]
**********  x[3,:]=x[:,8]  **********
[[ 0  1  2  3  4  5  6  7  8  9]
 [10 11 12 13 14 15 16 17 18 19]
 [20 21 22 23 24 25 26 27 28 29]
 [ 8 18 28 88 48 58 68 78 88 98]
 [40 41 42 43 44 45 46 47 48 49]
 [50 51 52 53 54 55 56 57 58 59]
 [60 61 62 63 64 65 66 67 68 69]
 [70 71 72 73 74 75 76 77 78 79]
 [80 81 82 83 84 85 86 87 88 89]
 [90 91 92 93 94 95 96 97 98 99]]
**********  x[4,:]=x[:,8]  **********
[[ 0  1  2  3  4  5  6  7  8  9]
 [10 11 12 13 14 15 16 17 18 19]
 [20 21 22 23 24 25 26 27 28 29]
 [30 31 32 33 34 35 36 37 38 39]
 [ 8 18 28 38 88 58 68 78 88 98]
 [50 51 52 53 54 55 56 57 58 59]
 [60 61 62 63 64 65 66 67 68 69]
 [70 71 72 73 74 75 76 77 78 79]
 [80 81 82 83 84 85 86 87 88 89]
 [90 91 92 93 94 95 96 97 98 99]]
```

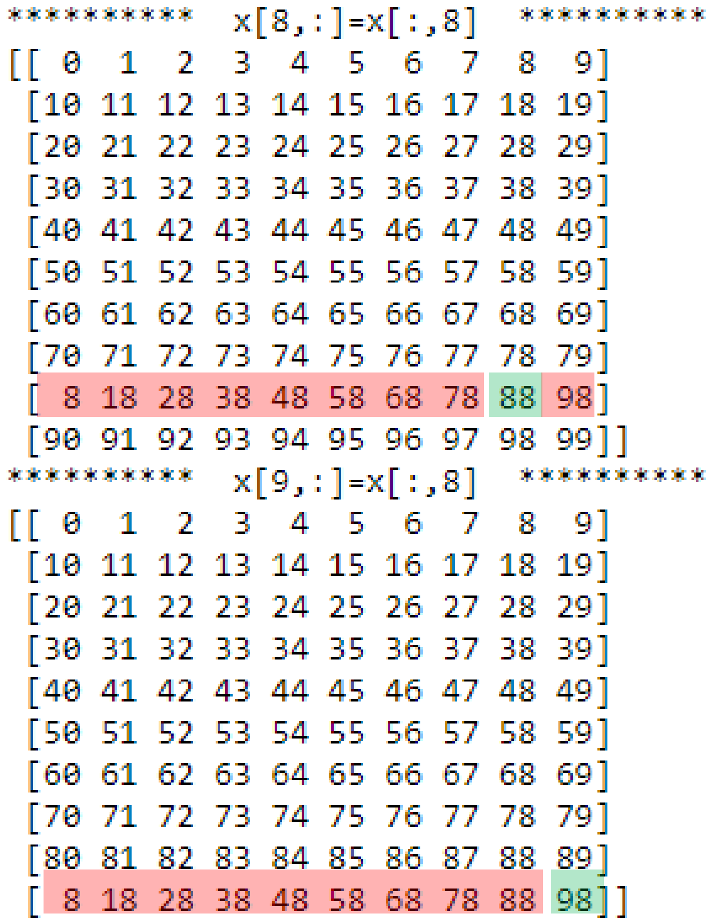
QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Slide
306/307



```
**********   x[5,:]=x[:,8]   **********
[[ 0  1  2  3  4  5  6  7  8  9]
 [10 11 12 13 14 15 16 17 18 19]
 [20 21 22 23 24 25 26 27 28 29]
 [30 31 32 33 34 35 36 37 38 39]
 [40 41 42 43 44 45 46 47 48 49]
 [ 8 18 28 38 48 88 68 78 88 98]
 [60 61 62 63 64 65 66 67 68 69]
 [70 71 72 73 74 75 76 77 78 79]
 [80 81 82 83 84 85 86 87 88 89]
 [90 91 92 93 94 95 96 97 98 99]]
**********   x[6,:]=x[:,8]   **********
[[ 0  1  2  3  4  5  6  7  8  9]
 [10 11 12 13 14 15 16 17 18 19]
 [20 21 22 23 24 25 26 27 28 29]
 [30 31 32 33 34 35 36 37 38 39]
 [40 41 42 43 44 45 46 47 48 49]
 [50 51 52 53 54 55 56 57 58 59]
 [ 8 18 28 38 48 58 88 78 88 98]
 [70 71 72 73 74 75 76 77 78 79]
 [80 81 82 83 84 85 86 87 88 89]
 [90 91 92 93 94 95 96 97 98 99]]
**********   x[7,:]=x[:,8]   **********
[[ 0  1  2  3  4  5  6  7  8  9]
 [10 11 12 13 14 15 16 17 18 19]
 [20 21 22 23 24 25 26 27 28 29]
 [30 31 32 33 34 35 36 37 38 39]
 [40 41 42 43 44 45 46 47 48 49]
 [50 51 52 53 54 55 56 57 58 59]
 [60 61 62 63 64 65 66 67 68 69]
 [ 8 18 28 38 48 58 68 88 88 98]
 [80 81 82 83 84 85 86 87 88 89]
 [90 91 92 93 94 95 96 97 98 99]]
```



```
**********   x[8,:]=x[:,8]   **********
[[ 0  1  2  3  4  5  6  7  8  9]
 [10 11 12 13 14 15 16 17 18 19]
 [20 21 22 23 24 25 26 27 28 29]
 [30 31 32 33 34 35 36 37 38 39]
 [40 41 42 43 44 45 46 47 48 49]
 [50 51 52 53 54 55 56 57 58 59]
 [60 61 62 63 64 65 66 67 68 69]
 [70 71 72 73 74 75 76 77 78 79]
 [ 8 18 28 38 48 58 68 78 88 98]
 [90 91 92 93 94 95 96 97 98 99]]
**********   x[9,:]=x[:,8]   **********
[[ 0  1  2  3  4  5  6  7  8  9]
 [10 11 12 13 14 15 16 17 18 19]
 [20 21 22 23 24 25 26 27 28 29]
 [30 31 32 33 34 35 36 37 38 39]
 [40 41 42 43 44 45 46 47 48 49]
 [50 51 52 53 54 55 56 57 58 59]
 [60 61 62 63 64 65 66 67 68 69]
 [70 71 72 73 74 75 76 77 78 79]
 [80 81 82 83 84 85 86 87 88 89]
 [ 8 18 28 38 48 58 68 78 88 98]]
```

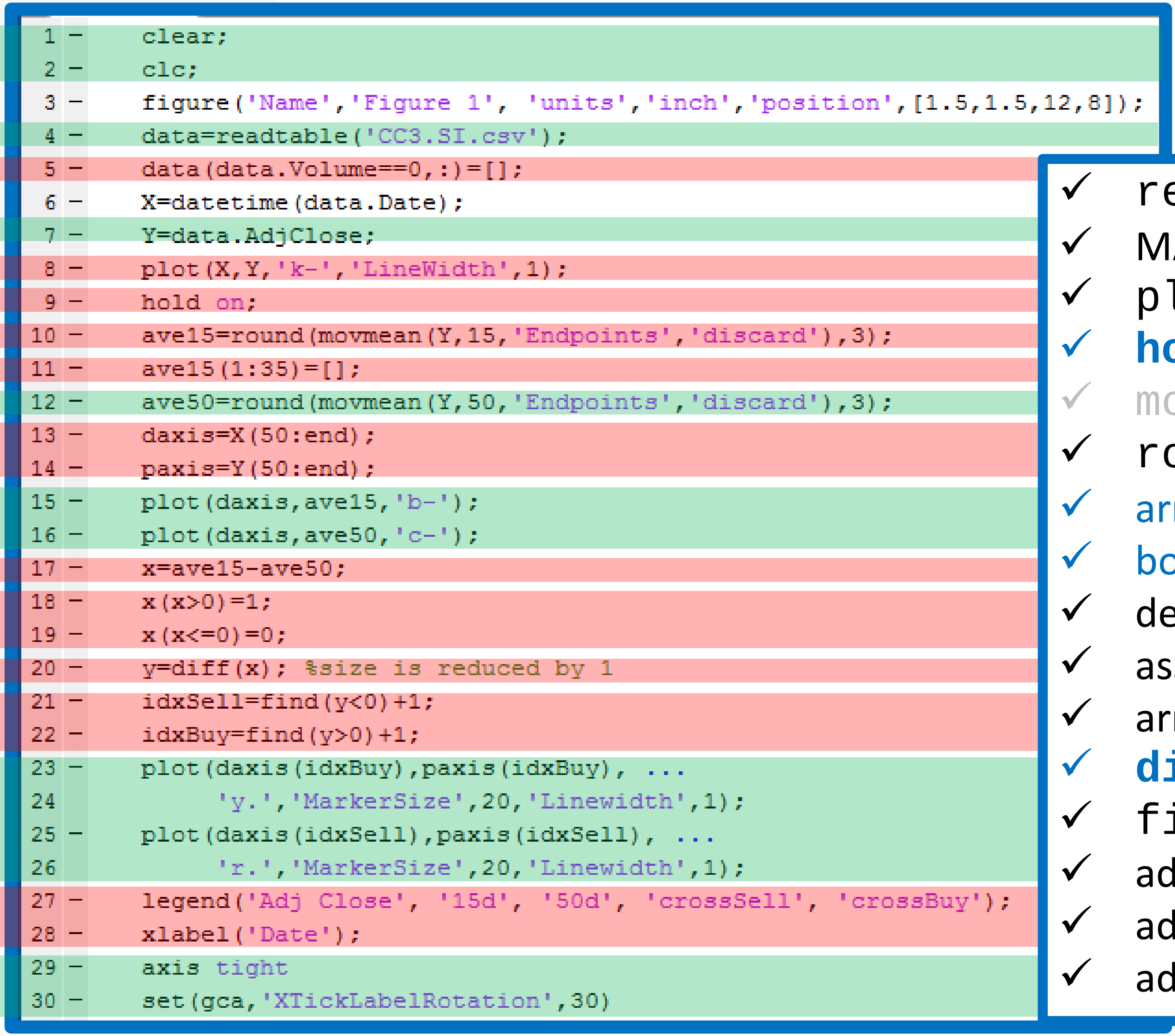
SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Dissection of the MATLAB code:

168



```
*    clear;
     clc;
     figure('Name','Figure 1', 'units','inch','position',[1.5,1.5,12,8]);
*    data=readtable('CC3.SI.csv');
1    data(data.Volume==0,:)=[];
     X=datetime(data.Date);
*    Y=data.AdjClose;
2    plot(X,Y,'k-','LineWidth',1);
3    hold on;
4    ave15=round(movmean(Y,15,'Endpoints','discard'),3);
5    ave15(1:35)=[];
*    ave50=round(movmean(Y,50,'Endpoints','discard'),3);
6    daxis=X(50:end);
     paxis=Y(50:end);
*    plot(daxis,ave15,'b-');
     plot(daxis,ave50,'c-');
7    x=ave15-ave50;
8    x(x>0)=1;
     x(x<=0)=0;
9    y=diff(x); %size is reduced by 1
10   idxSell=find(y<0)+1;
     idxBuy=find(y>0)+1;
*    plot(daxis(idxBuy),paxis(idxBuy), ...
             'y.','MarkerSize',20,'Linewidth',1);
     plot(daxis(idxSell),paxis(idxSell), ...
             'r.','MarkerSize',20,'Linewidth',1);
11   legend('Adj Close', '15d', '50d', 'crossSell', 'crossBuy');
     xlabel('Date');
*    axis tight
     set(gca,'XTickLabelRotation',30)
```

- ✓ `readtable`
- ✓ MATLAB table data selection
- ✓ `plot`
- ✓ **`hold on`**
- ✓ `movmean (D.N.T.)`
- ✓ `round`
- ✓ array comparison operations
- ✓ boolean index/logical array
- ✓ delete elements
- ✓ assignment
- ✓ array arithmetic operations
- ✓ **`diff`**
- ✓ `find`
- ✓ add `legend`
- ✓ add `xlabel` and `ylabel`
- ✓ add `title`

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance