SMU
SINGAPORE MANAGEMENT
UNIVERSITY

QF627

(AY2018, Term 2)

SMU
SINGAPORE MANAGEMENT
UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

About The Course

1. Basics of MATLAB and Review of Python (2 lessons)
2. Data Manipulation and Visualization in Python and MATLAB (3 lessons)
3. Scientific Tools in Python and MATLAB (3 lessons)
   Equation Solving, Optimization, Numerical Differentiation, Numerical Integration, Interpolation, Linear Algebra, Regression, Statistical Tests, Random Number Generation and Monte Carlo Simulation, etc.

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Assessments

✓ **Class Participation: 20%**

= 8×2% (Attendance, S01-S08) + 4% (Peer Evaluation, S0108)

✓ **In-Class Exercises: 15%**

= 3×5% (Group Work, S0102, S0305, S0608)

✓ **Homework Assignments: 15%**

= 3 ×5% (Individual Work, S0102, S0305, S0608)

✓ **Final Exam: 50%, closed-book, 2 hours**

= 30% (Python Only) + 20% (Python or MATLAB**)

001

About eLearn

✓ Content (-> Learning Materials)

✓ Discussions (After-Class Q&A)

✓ Assignments (-> I.C.E & H.W.)

✓ Peer Evaluation

Sessions 01+02

MATLAB Basics

(in comparison to Python)

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

Learning Outcomes and Problems to be revisited in these two sessions:

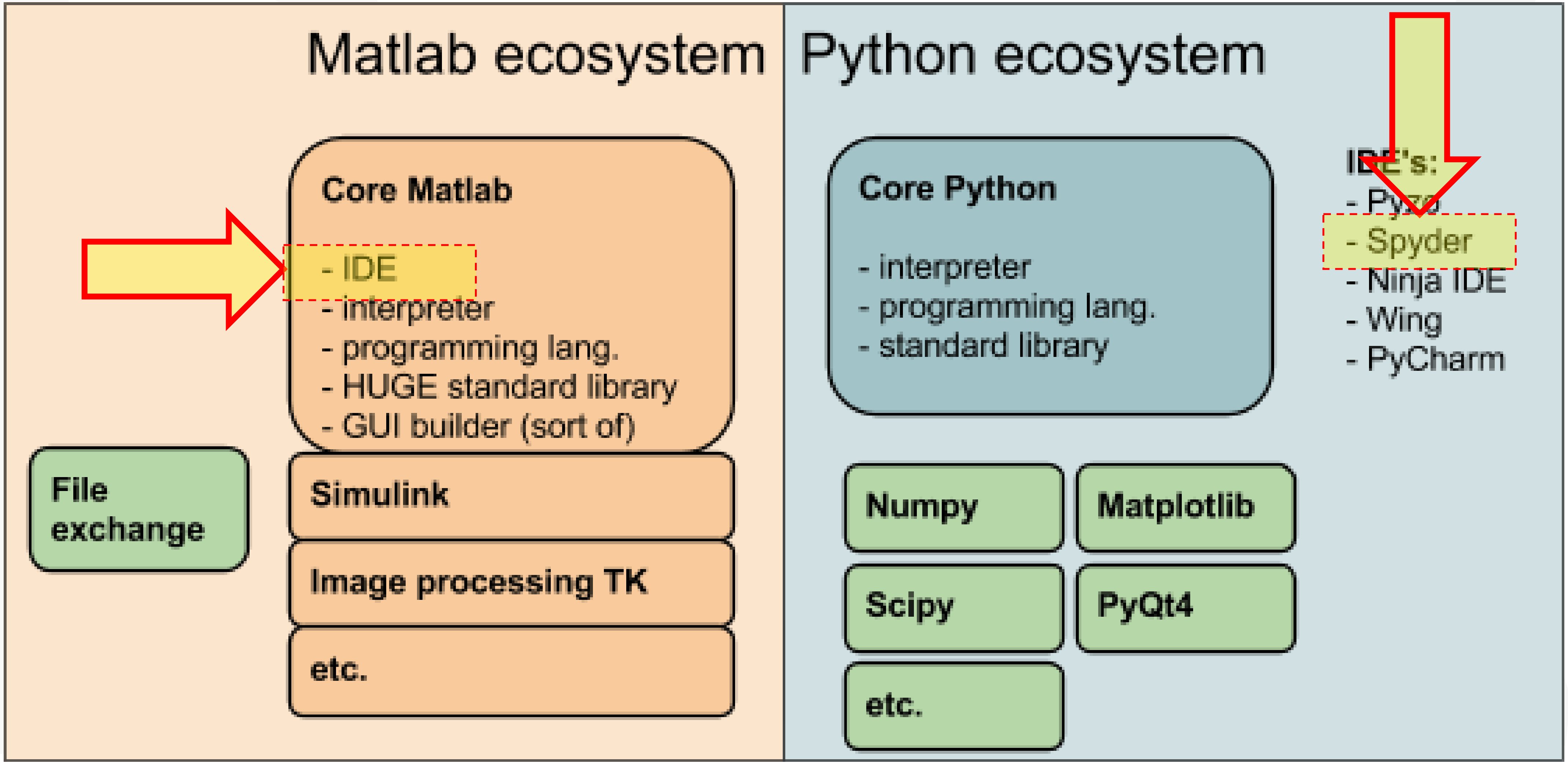
1. **HDB Loan Calculator (GUI)**
   - □ Literals and Data Types (Numbers, Strings, Function Handles etc.)
   - □ Arithmetic Operators and Operator Precedence
   - □ Data Structures (Arrays, Structure, Cell Arrays, etc.)
   - □ Expressions
   - □ Variables and Assignments Basics
   - □ Mathematical Functions
   - □ User Defined Functions Basics
   - □ Others (Comments, Indentation, Line Joining, Concatenation, Semicolon, Colon, Some Built-in Functions etc.)
2. **Income Tax Calculator**
   - □ Logical Operators, Comparison Operators
   - □ Flow Control (if statements)
3. **Pandigital Formula**
   - □ Flow Control (loops)
4. **Sudoku Solver**
   - □ More on Functions (recursive function)
5. **European Call Option Object (incl. Methods Value, Vega and Implied Volatility)**
   - □ Class

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

MATLAB vs. Python



Matlab ecosystem

Core Matlab

- IDE
- interpreter
- programming lang.
- HUGE standard library
- GUI builder (sort of)

File exchange

Simulink

Image processing TK

etc.

Python ecosystem

Core Python

- interpreter
- programming lang.
- standard library

IDE's:
- Pyzo
- Spyder
- Ninja IDE
- Wing
- PyCharm

Numpy

Matplotlib

Scipy

PyQt4

etc.

(Source: http://www.pyzo.org/python_vs_matlab.html)

SMU SINGAPORE MANAGEMENT UNIVERSITY

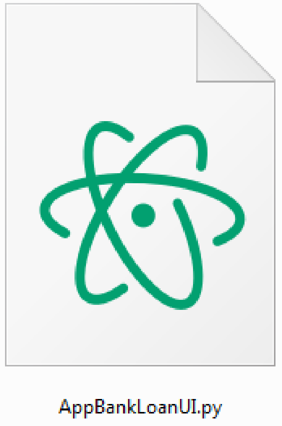
QF666
Programming and
Computational
Finance

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



AppBankLoanUI.py

HDBLoanCalculator.ui

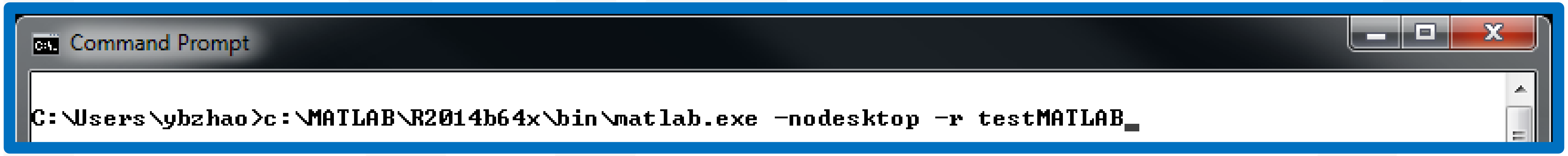
How to run a Python Program?

How to run a MATLAB Program?

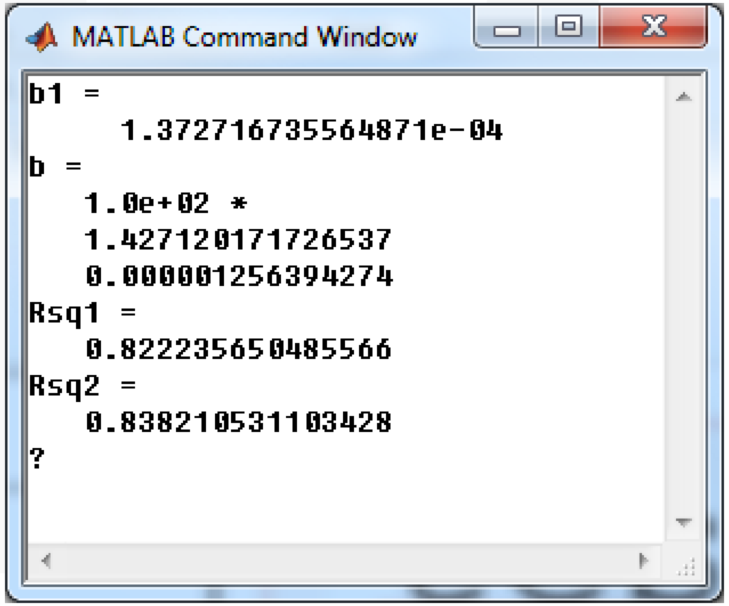
testMATLAB.m

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

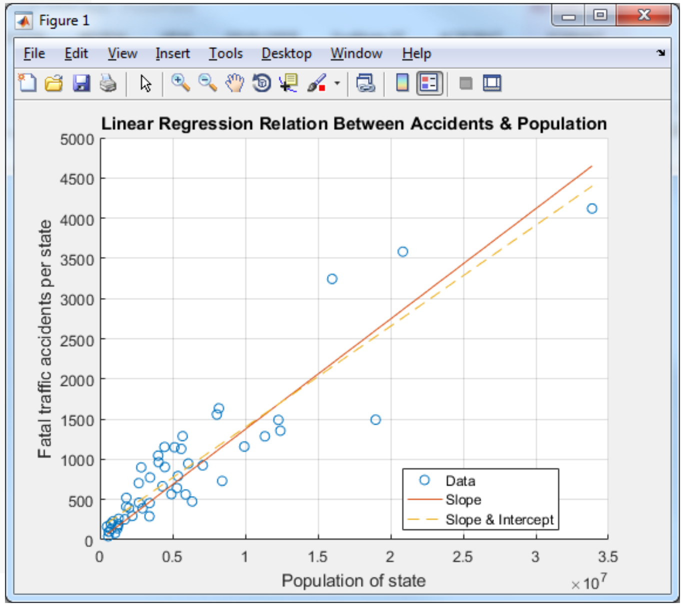
testMATLAB.m



```
C:\Users\ybzhao>c:\MATLAB\R2014b64x\bin\matlab.exe -nodesktop -r testMATLAB_
```



```
b1 =
     1.372716735564871e-04
b  =
   1.0e+02 *
   1.427120171726537
   0.000001256394274
Rsq1 =
   0.822235650485566
Rsq2 =
   0.838210531103428
?
```



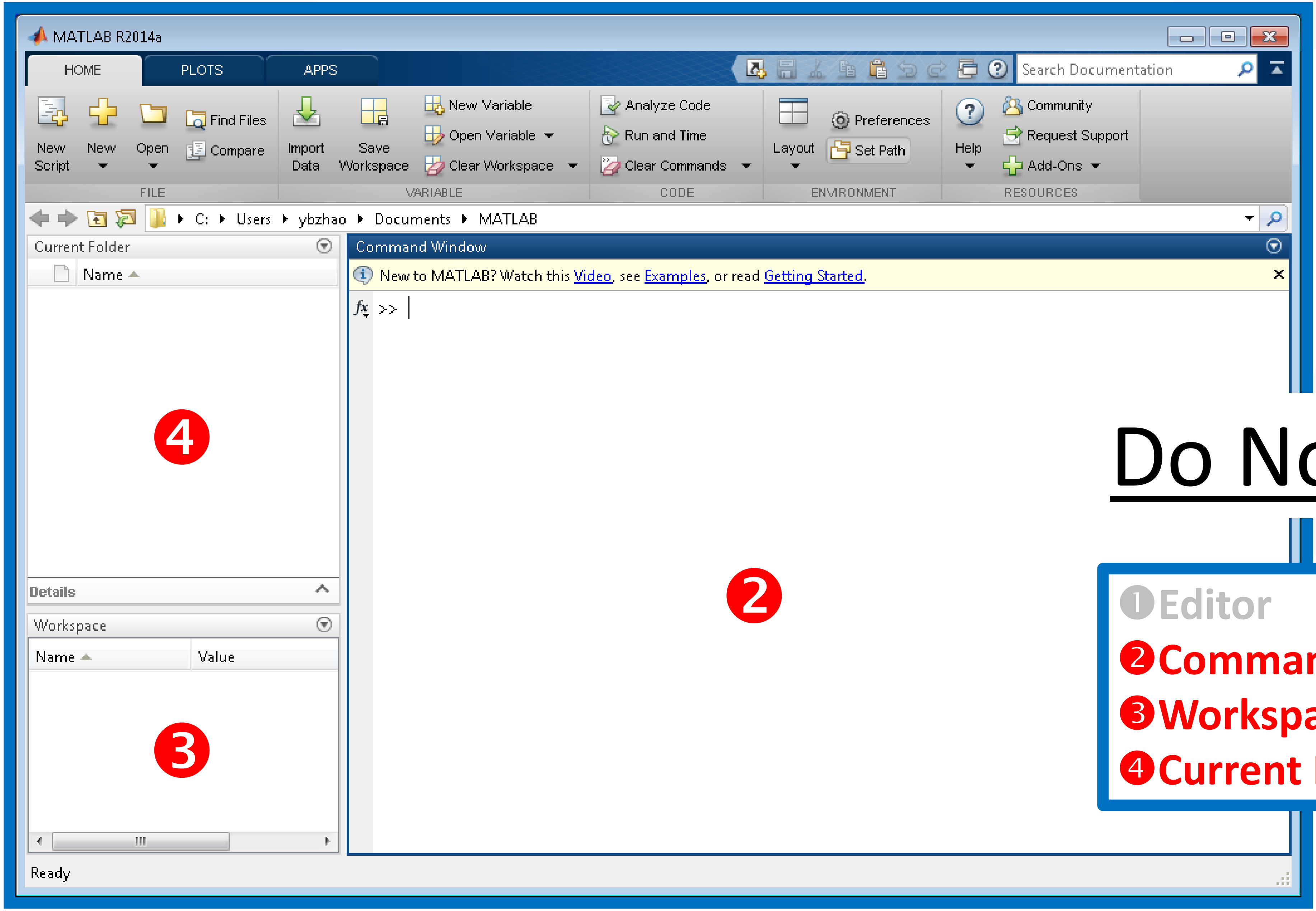
(Dr. Z: This is not very friendly.)

⇨ IDE

002

Do Not Test

MATLAB Default Desktop Layout

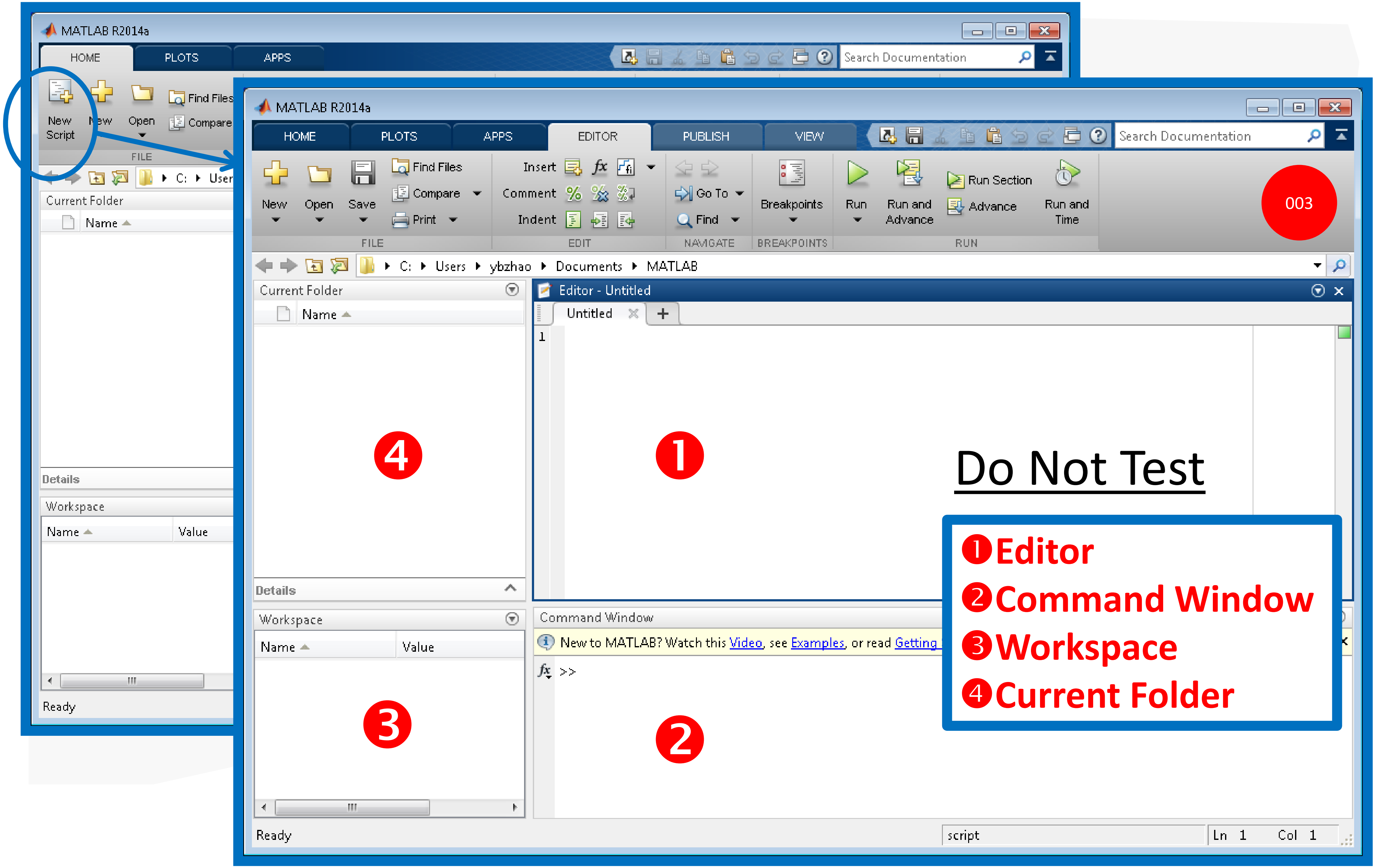


Do Not Test

➊ Editor
➋ Command Window
➌ Workspace
➍ Current Folder

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance



New to MATLAB? Watch this Video, see Examples, or read Getting

```
>>
```

Do Not Test

❶Editor
❷Command Window
❸Workspace
❹Current Folder

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

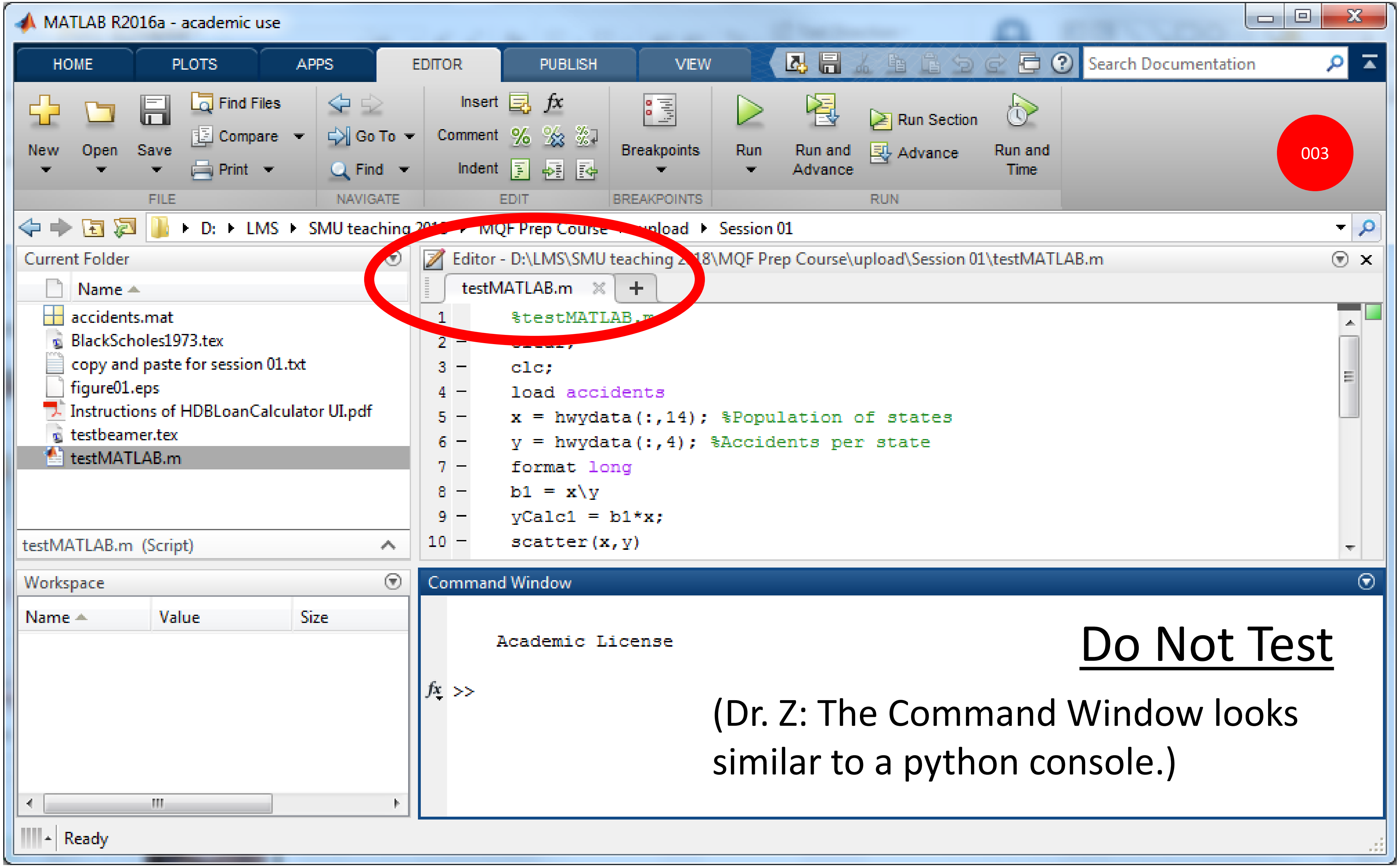
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

Slide 012/160

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
%testMATLAB.m
clear;
clc;
load accidents
x = hwydata(:,14); %Population of states
y = hwydata(:,4); %Accidents per state
format long
b1 = x\y
yCalc1 = b1*x;
scatter(x,y)
```

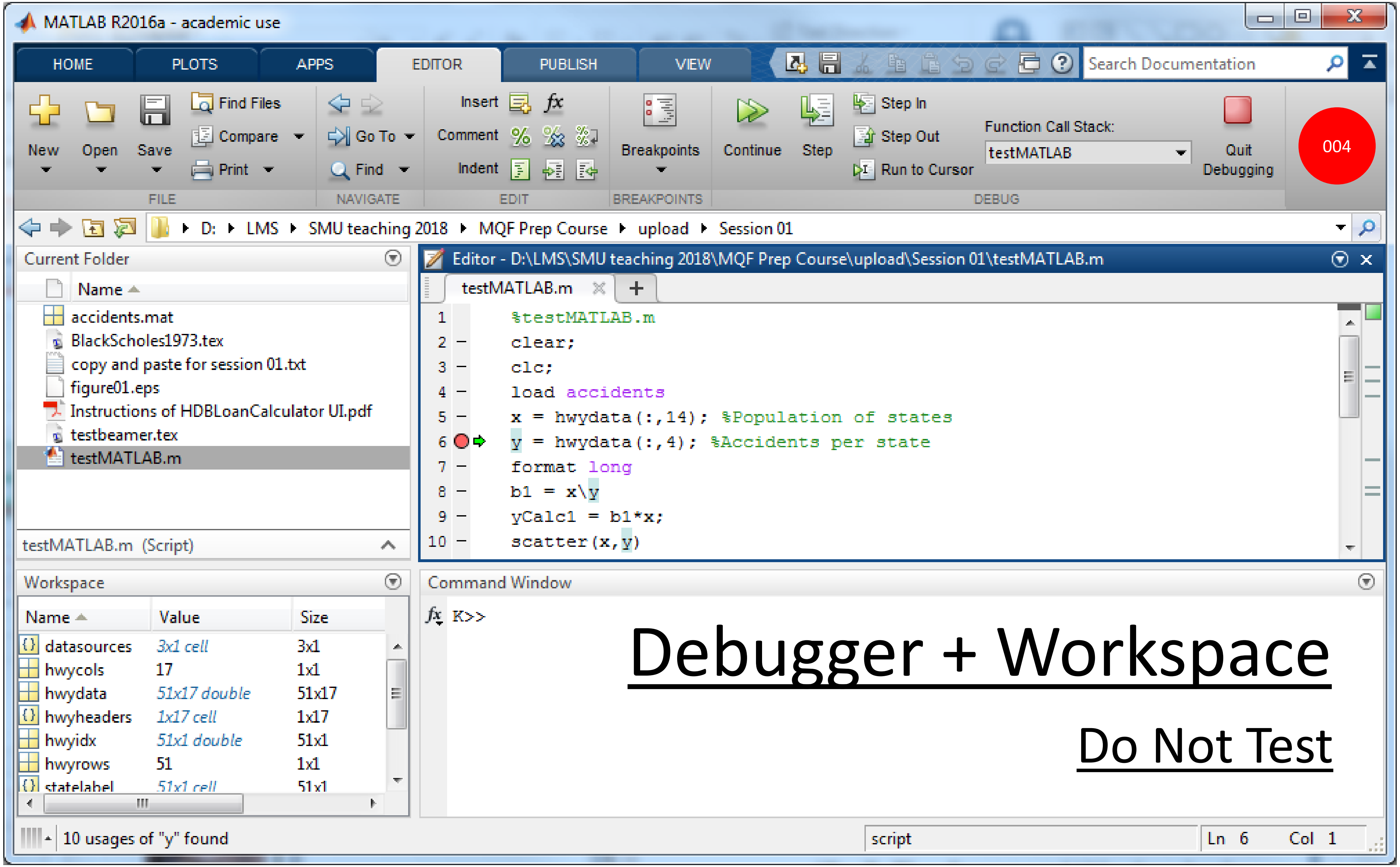
```
Academic License

fx >>
```

Do Not Test

(Dr. Z: The Command Window looks similar to a python console.)

Debugger + Workspace

Do Not Test



Current Folder:

- accidents.mat
- BlackScholes1973.tex
- copy and paste for session 01.txt
- figure01.eps
- Instructions of HDBLoanCalculator UI.pdf
- testbeamer.tex
- testMATLAB.m

testMATLAB.m (Script)

```
%testMATLAB.m
clear;
clc;
load accidents
x = hwydata(:,14); %Population of states
y = hwydata(:,4); %Accidents per state
format long
b1 = x\y
yCalc1 = b1*x;
scatter(x,y)
```

Name	Value	Size
datasources	*3x1 cell*	3x1
hwycols	17	1x1
hwydata	*51x17 double*	51x17
hwyheaders	*1x17 cell*	1x17
hwyidx	*51x1 double*	51x1
hwyrows	51	1x1
statelabel	*51x1 cell*	51x1

```
K>>
```

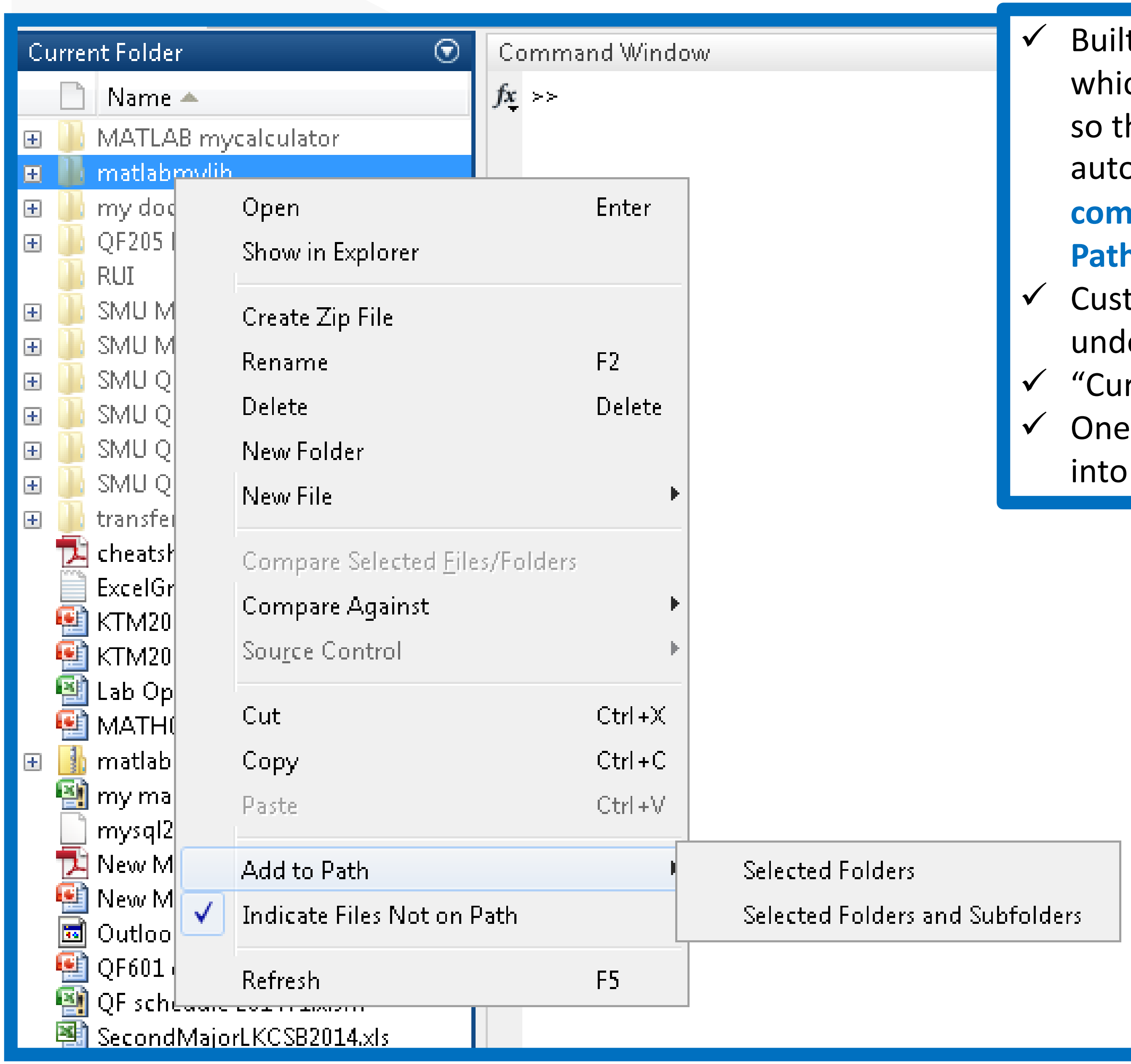
10 usages of "y" found | script | Ln 6 Col 1

SINGAPORE MANAGEMENT UNIVERSITY

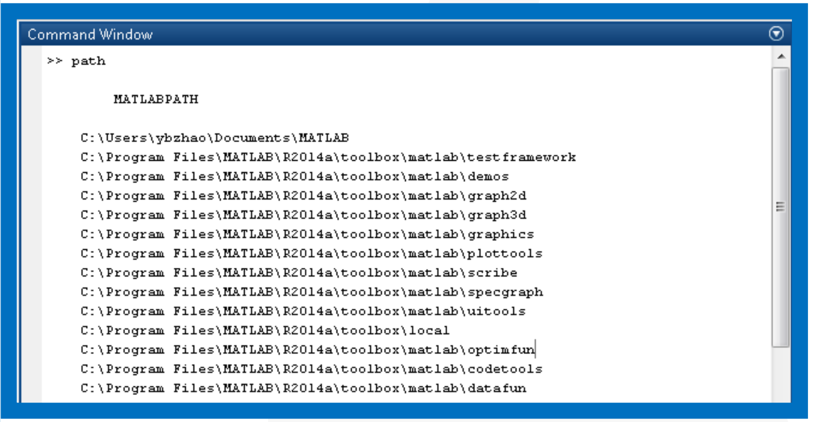
QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

Change Current Folder or Add to Path?

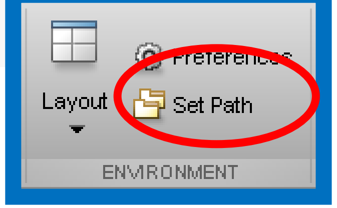


- ✓ Built-in library functions are stored in some folders which have been included in the “searching paths” so that whenever they are invoked, they can be automatically found. (One can **type “`path`” in command window to view them or click the “Set Path” button.**)
- ✓ Custom/User-defined functions have to be put under the “searching paths”.
- ✓ “Current Folder “ is under the “searching paths”.
- ✓ One can use “Add to Path” to include more folders into the searching paths.



```
>> path

        MATLABPATH

    C:\Users\ybzhao\Documents\MATLAB
    C:\Program Files\MATLAB\R2014a\toolbox\matlab\testframework
    C:\Program Files\MATLAB\R2014a\toolbox\matlab\demos
    C:\Program Files\MATLAB\R2014a\toolbox\matlab\graph2d
    C:\Program Files\MATLAB\R2014a\toolbox\matlab\graph3d
    C:\Program Files\MATLAB\R2014a\toolbox\matlab\graphics
    C:\Program Files\MATLAB\R2014a\toolbox\matlab\plottools
    C:\Program Files\MATLAB\R2014a\toolbox\matlab\scribe
    C:\Program Files\MATLAB\R2014a\toolbox\matlab\specgraph
    C:\Program Files\MATLAB\R2014a\toolbox\matlab\uitools
    C:\Program Files\MATLAB\R2014a\toolbox\local
    C:\Program Files\MATLAB\R2014a\toolbox\matlab\optimfun
    C:\Program Files\MATLAB\R2014a\toolbox\matlab\codetools
    C:\Program Files\MATLAB\R2014a\toolbox\matlab\datafun
```

Do Not Test

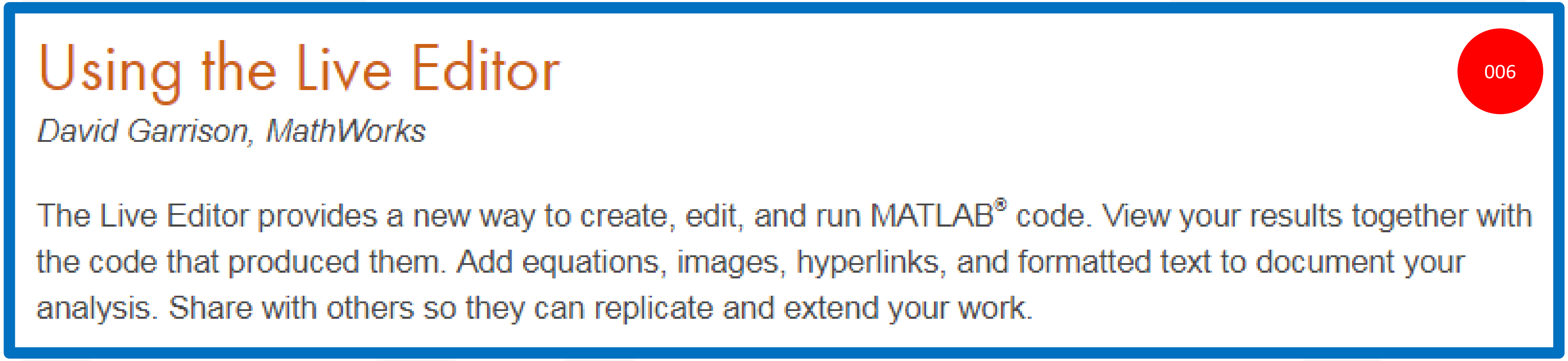


SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

https://www.mathworks.com/products/matlab/live-editor.html



Using the Live Editor

David Garrison, MathWorks

The Live Editor provides a new way to create, edit, and run MATLAB® code. View your results together with the code that produced them. Add equations, images, hyperlinks, and formatted text to document your analysis. Share with others so they can replicate and extend your work.

006

This is similar to the Jupyter Notebook (or Jupyter Lab).

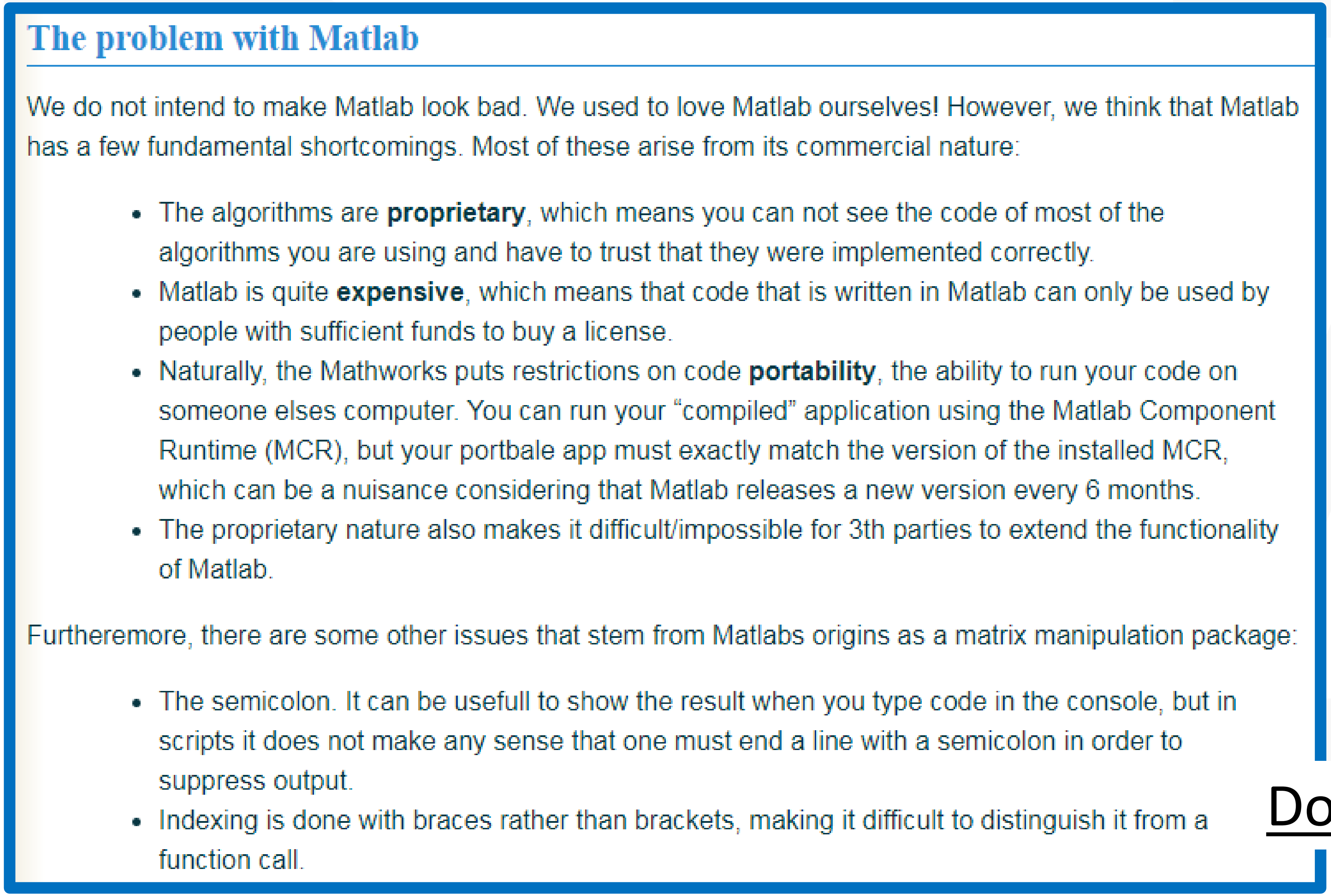
Do Not Test

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

http://www.pyzo.org/python_vs_matlab.html



The problem with Matlab

We do not intend to make Matlab look bad. We used to love Matlab ourselves! However, we think that Matlab has a few fundamental shortcomings. Most of these arise from its commercial nature:

- The algorithms are **proprietary**, which means you can not see the code of most of the algorithms you are using and have to trust that they were implemented correctly.
- Matlab is quite **expensive**, which means that code that is written in Matlab can only be used by people with sufficient funds to buy a license.
- Naturally, the Mathworks puts restrictions on code **portability**, the ability to run your code on someone elses computer. You can run your “compiled” application using the Matlab Component Runtime (MCR), but your portbale app must exactly match the version of the installed MCR, which can be a nuisance considering that Matlab releases a new version every 6 months.
- The proprietary nature also makes it difficult/impossible for 3th parties to extend the functionality of Matlab.

Furtheremore, there are some other issues that stem from Matlabs origins as a matrix manipulation package:

- The semicolon. It can be usefull to show the result when you type code in the console, but in scripts it does not make any sense that one must end a line with a semicolon in order to suppress output.
- Indexing is done with braces rather than brackets, making it difficult to distinguish it from a function call.

Do Not Test

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

MATLAB uses braces for indexing rather than brackets. This makes it difficult to distinguish it from a function call.

```
x(1), f(2)
```

007

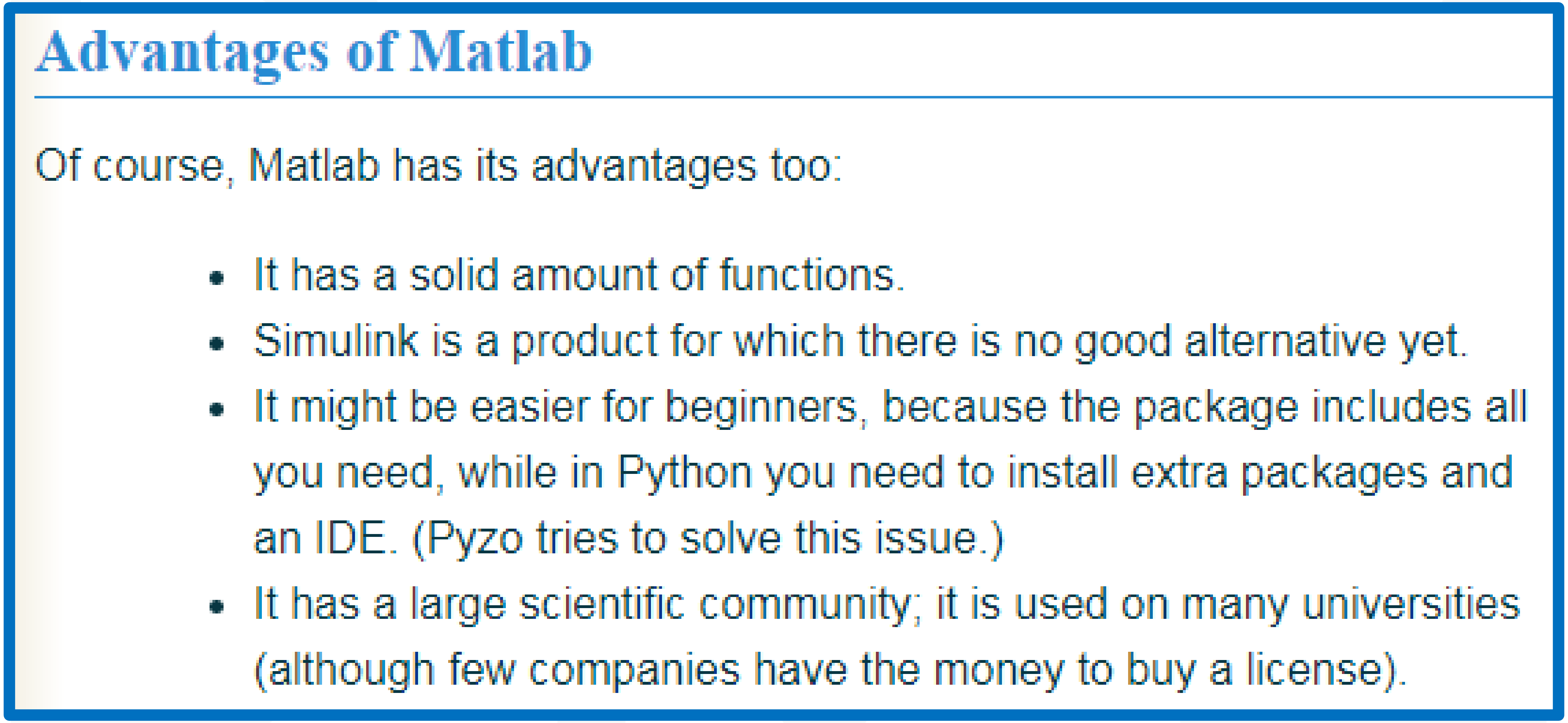
Python uses brackets for indexing, braces for function calls.

```
x[1], f(2)
```

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

http://www.pyzo.org/python_vs_matlab.html



Advantages of Matlab

Of course, Matlab has its advantages too:

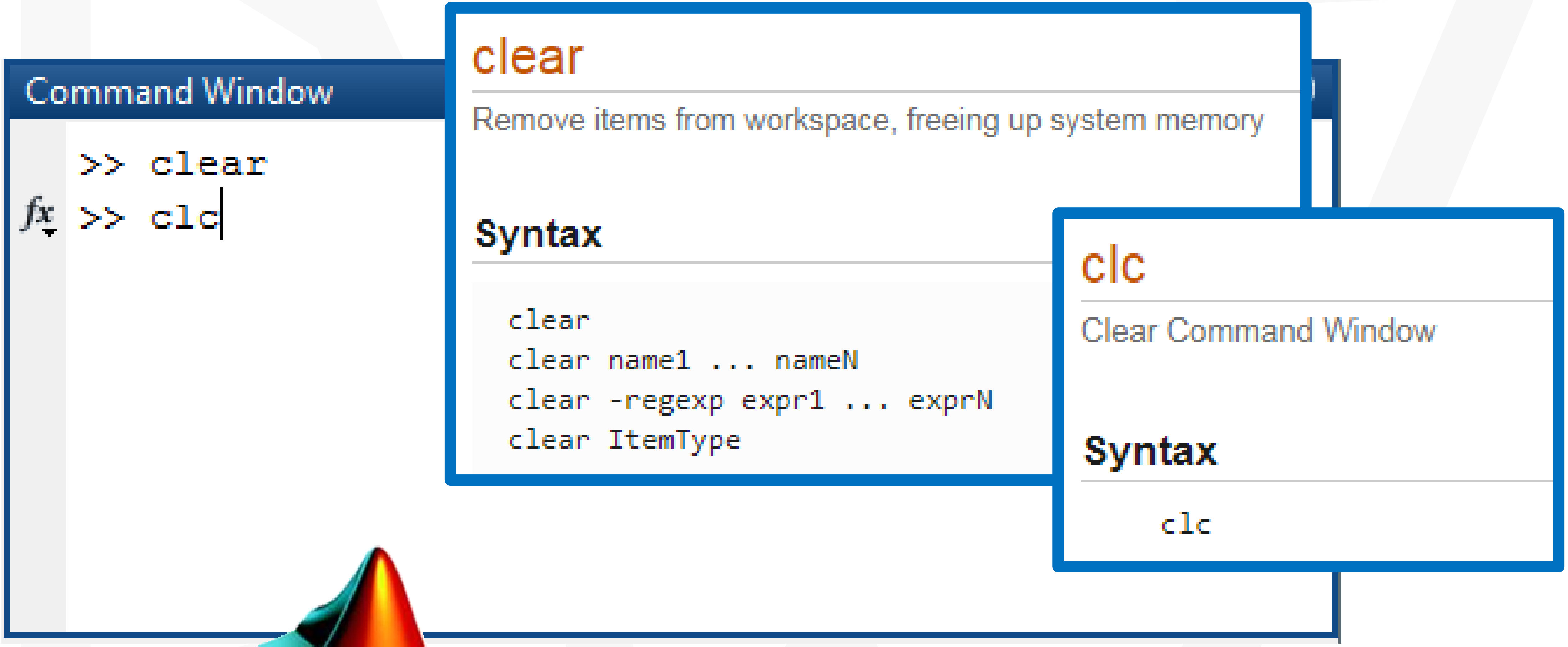
- It has a solid amount of functions.
- Simulink is a product for which there is no good alternative yet.
- It might be easier for beginners, because the package includes all you need, while in Python you need to install extra packages and an IDE. (Pyzo tries to solve this issue.)
- It has a large scientific community; it is used on many universities (although few companies have the money to buy a license).

Do Not Test

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

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

008 009



Command Window

```
>> clear
>> clc
```

clear

Remove items from workspace, freeing up system memory

Syntax

```
clear
clear name1 ... nameN
clear -regexp expr1 ... exprN
clear ItemType
```

clc

Clear Command Window

Syntax

```
clc
```

MATLAB

Dr. Z: Python does not have these two commands.

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

SMU
SINGAPORE MANAGEMENT UNIVERSITY

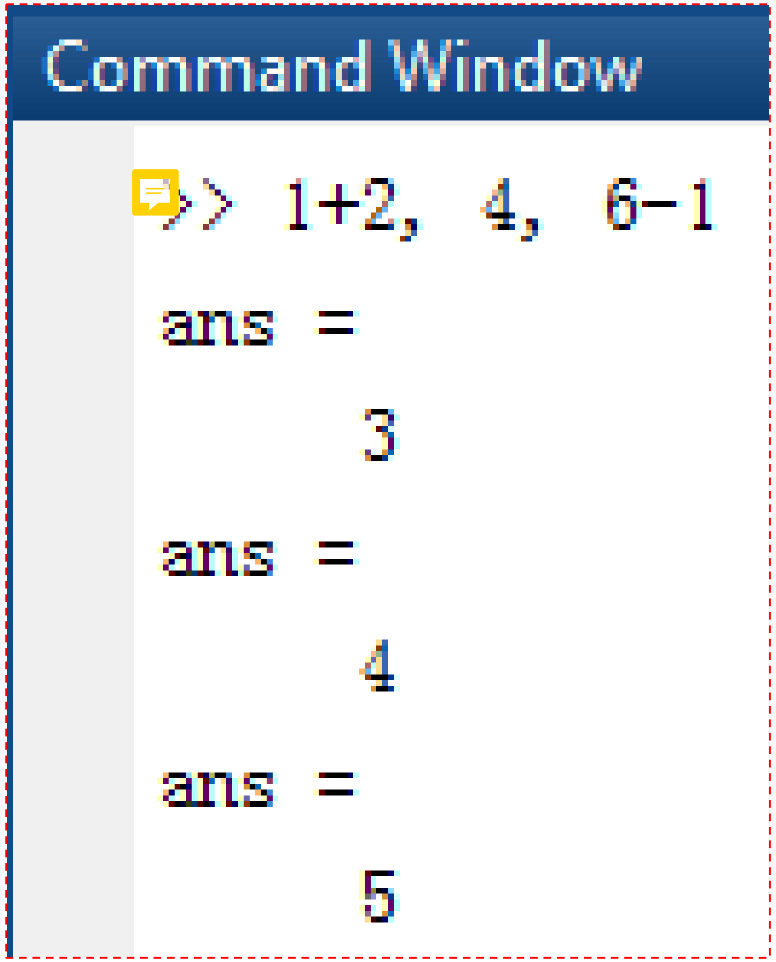
QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

Comma-Separated List

010

https://www.mathworks.com/help/matlab/matlab_prog/comma-separated-lists.html



```
>> 1+2, 4, 6-1

ans =

     3

ans =

     4

ans =

     5
```

The MATLAB® software returns each value individually.

Such a list, by itself, is not very useful. But when used with large and more complex data structures like MATLAB **structures** and **cell arrays**, the comma-separated list can enable you to simplify your MATLAB code.

Common uses for comma-separated lists are:

- ✓ Constructing Arrays
- ✓ Displaying Arrays
- ✓ Concatenation
- ✓ Function Call Arguments
- ✓ Function Return Values

(Dr. Z: No hurry. We'll explore them at a later time.)

Slide
023/160

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

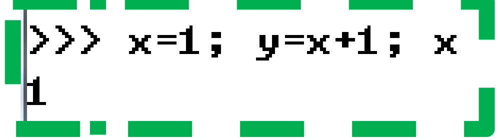
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

011

(the semicolon)

Python:

In Python, we seldom use **semicolons**. Though it is not advised, we can use **semicolons** to write multiple statements on the same line.



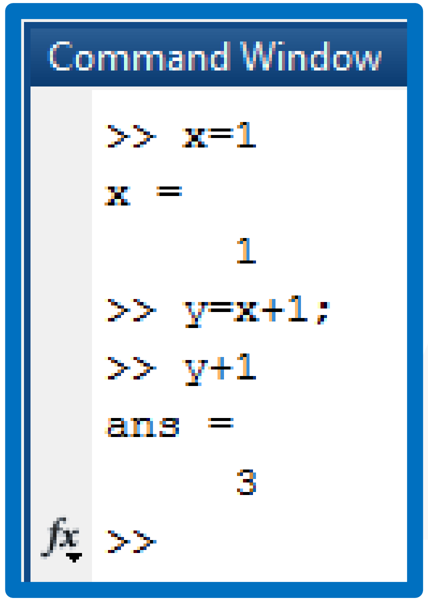
```
>>> x=1; y=x+1; x
1
```

Python:

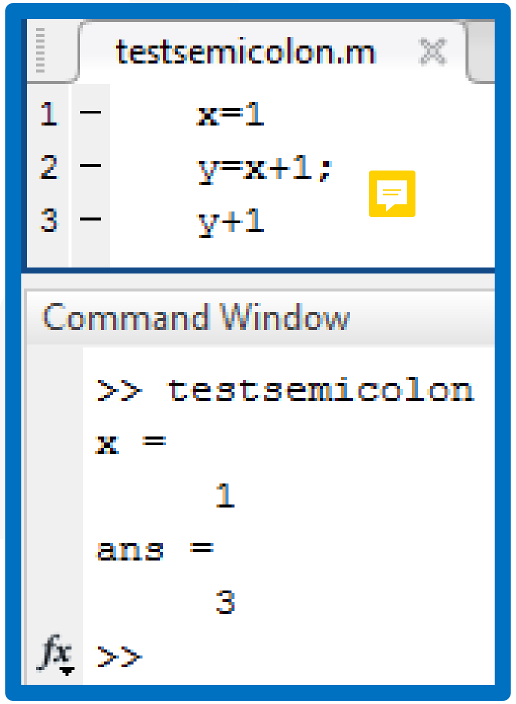
In the interactive mode, Python will write the value on the screen for the expression typed.

MATLAB:

Many MATLAB commands (incl. assignment) will output something in the Command window. **Semicolons** are used to suppress output.



```
>> x=1
x =
     1
>> y=x+1;
>> y+1
ans =
     3
>> 
```



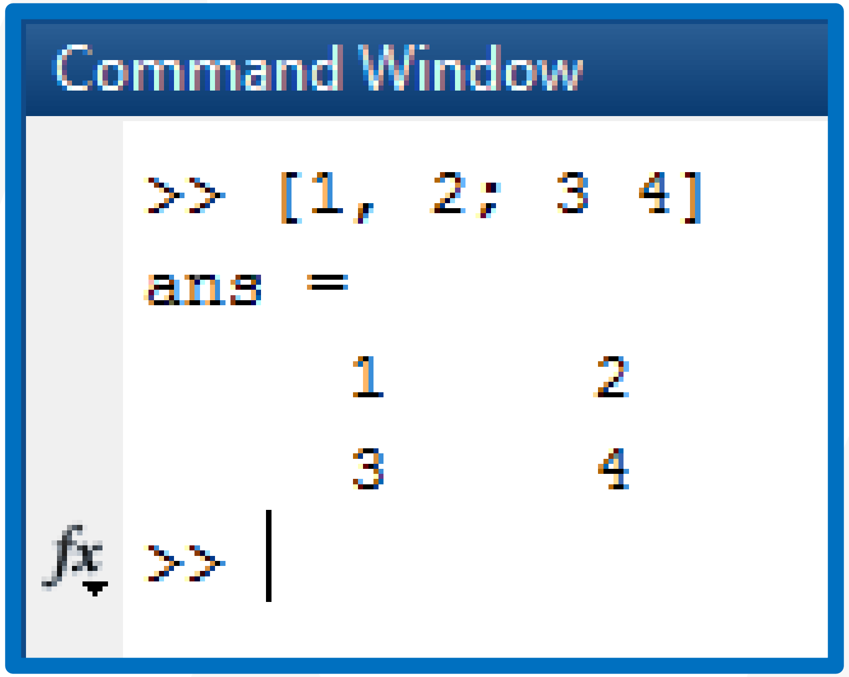
```
x=1
y=x+1;
y+1
```

```
>> testsemicolon
x =
     1
ans =
     3
>> 
```

https://www.mathworks.com/help/matlab/matlab_prog/matlab-operators-and-special-characters.html

In MATLAB, the other use of the semicolon is to signify end of row in 2D arrays.

(the semicolon)



```
>> [1, 2; 3 4]
ans =
     1     2
     3     4
>> |
```

012

013

Elements in the row can be separated by using either commas or spaces.

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Application 1: HDB Loan Calculator (GUI)

$$P = \frac{\frac{r}{12}(PV)}{1 - \left(1 + \frac{r}{12}\right)^{-12t}}$$

Given:

$r = 2.6\% \rightarrow 0.026$

$PV = 800{,}000$

$t = 25$

Compute: P

- ❑ Numeric Literals
- ❑ Arithmetic Operators
- ❑ Operator Precedence
- ❑ Variables
- ❑ Simple Assignment
- ❑ Built-in Mathematical Functions
- ❑ Built-in Function to convert between a number and a string.

014

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

MATLAB Variable Names

https://www.mathworks.com/help/matlab/matlab_prog/variable-names.html

A valid variable name

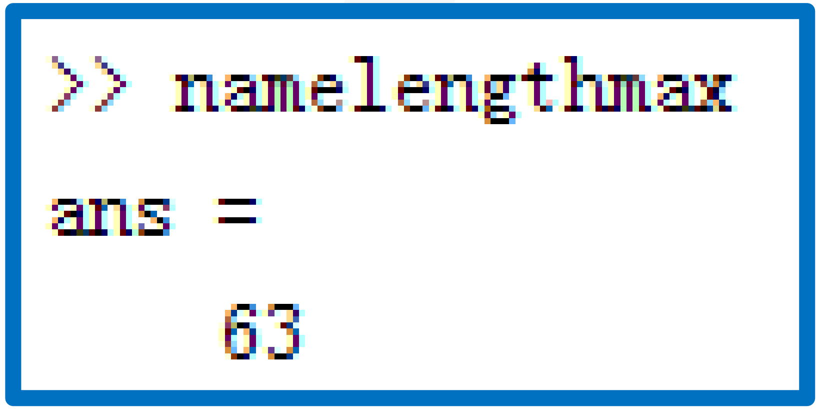
- ✓ starts with a letter,
- ✓ followed by letters, digits, or underscores.

015

MATLAB® is case sensitive, so A and a are not the same variable. The maximum length of a variable name is the value that the `namelengthmax` command returns.

016

017



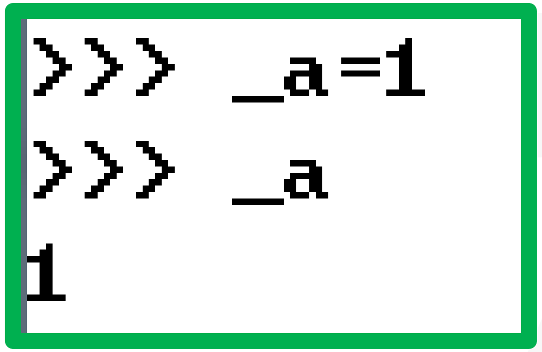
```
>> namelengthmax

ans =

    63
```

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

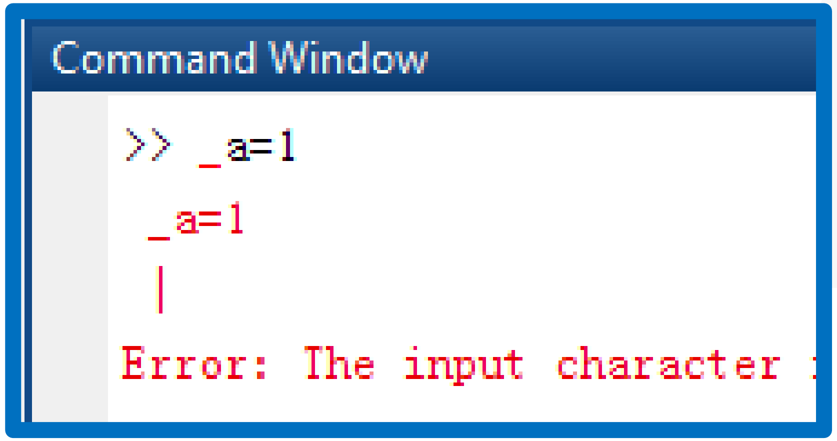
MATLAB:



```
>>> _a=1
>>> _a
1
```

018 Q: Can I use _a as a variable name?

Variable Names



```
>> _a=1
 _a=1
 |
Error: The input character
```

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Arithmetic Operators

+, -, *, /, ^

019

+, -, *, /, **

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

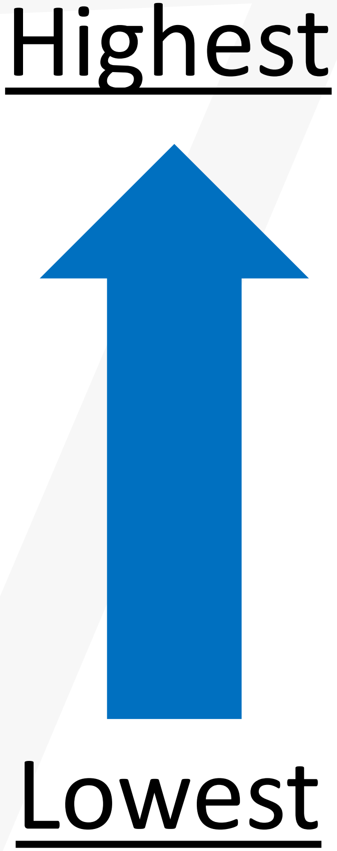
SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

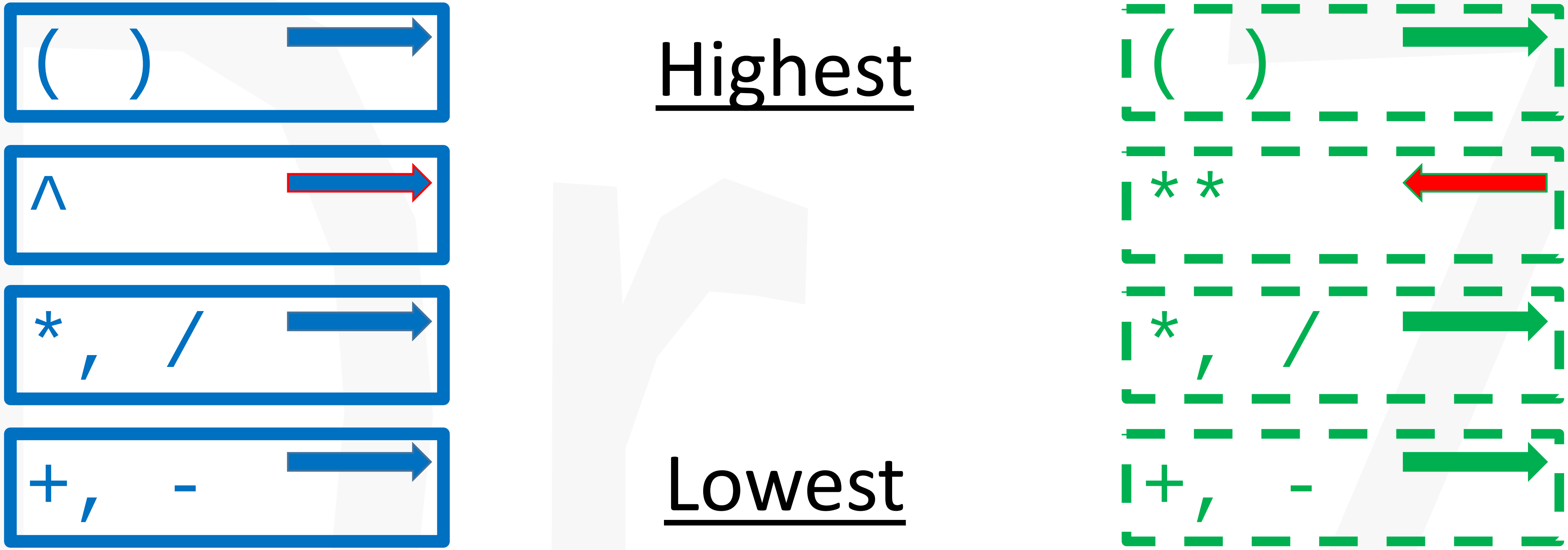
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Operator Precedence

1. Parentheses ()
2. Transpose (.'), power (.^), complex conjugate transpose ('), matrix power (^)
3. Power with unary minus (.^-), unary plus (.^+), or logical negation (.^~) as well as matrix power with unary minus (^-), unary plus (^+), or logical negation (^~).
4. Unary plus (+), unary minus (-), logical negation (~)
5. Multiplication (.*), right division (./), left division (.\), matrix multiplication (*), matrix right division (/), matrix left division (\)
6. Addition (+), subtraction (-)
7. Colon operator (:)
8. Less than (<), less than or equal to (<=), greater than (>), greater than or equal to (>=), equal to (==), not equal to (~=)
9. Element-wise AND (&)
10. Element-wise OR (|)
11. Short-circuit AND (&&)
12. Short-circuit OR (||)



Although most operators work from left to right, the operators (^-), (.^-), (^+), (.^+), (^~), and (.^~) work from second from the right to left. **It is recommended that you use parentheses to explicitly specify the intended precedence of statements containing these operator combinations.**



( )

^

*, /

+, -

Highest

Lowest

( )

**

*, /

+, -

4^(3^2)

$$4^{3^2}$$

4**3**2

```
>>> 4**3**2
262144
```

021

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

Python

```
PV=800000
t=25
r=2.6/100
P=(r/12*PV)/(1-(1+r/12)**(-12*t))
print(P)
```

✓ ** ⇨ ^ (sometimes needs extra braces)

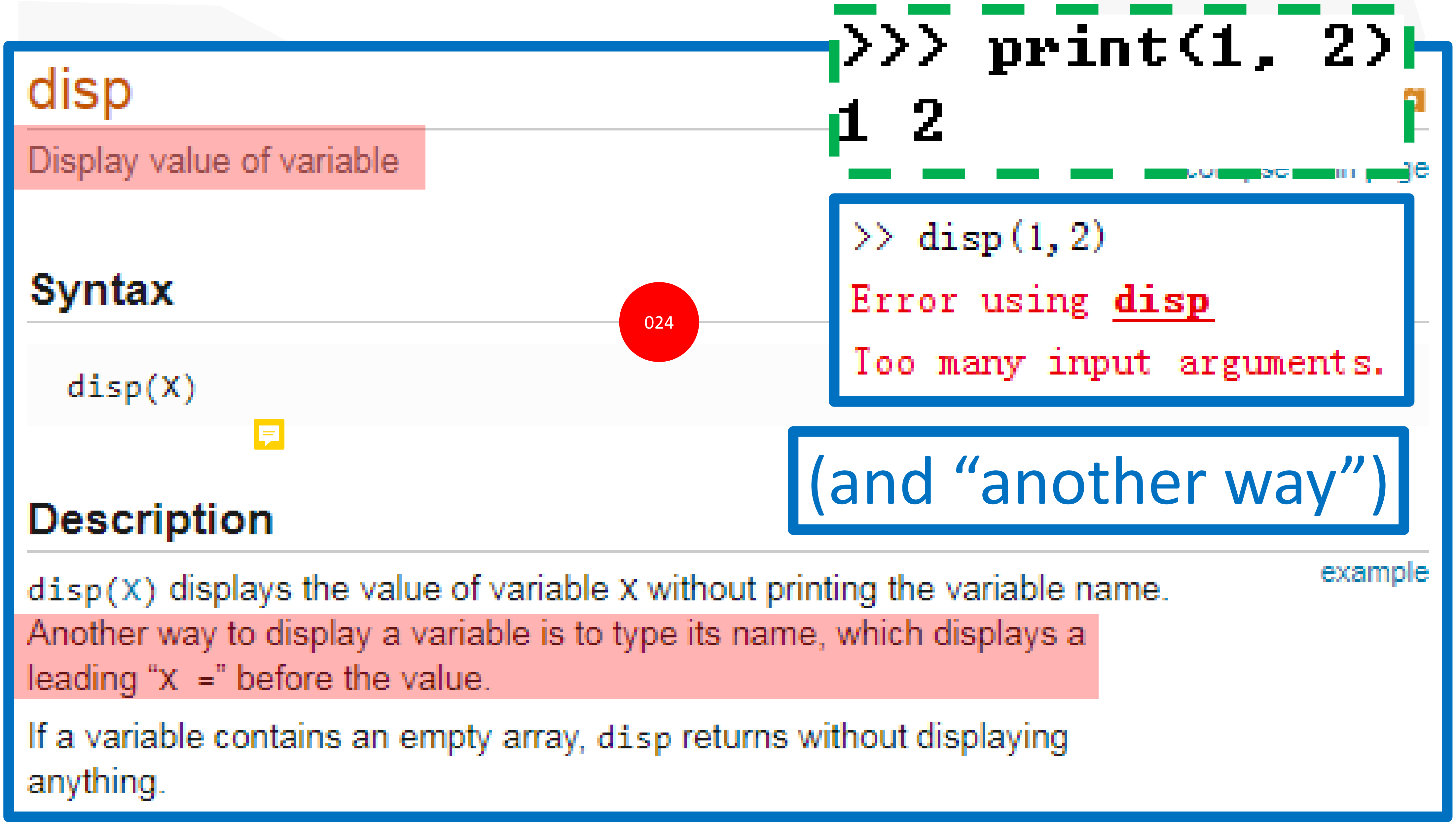
? print ⇨ disp

022 023

MATLAB

```
PV=800000;
t=25;
r=2.6/100;
P=(r/12*PV)/(1-(1+r/12)^(-12*t));
disp(P);
```

026



disp

Display value of variable

```
>>> print(1, 2)
1 2
```

```
>> disp(1,2)
Error using disp
Too many input arguments.
```

(and "another way")

Syntax

```
disp(X)
```

024

Description

example

`disp(X)` displays the value of variable X without printing the variable name. Another way to display a variable is to type its name, which displays a leading "X =" before the value.

If a variable contains an empty array, `disp` returns without displaying anything.

SMU SINGAPORE MANAGEMENT UNIVERSITY

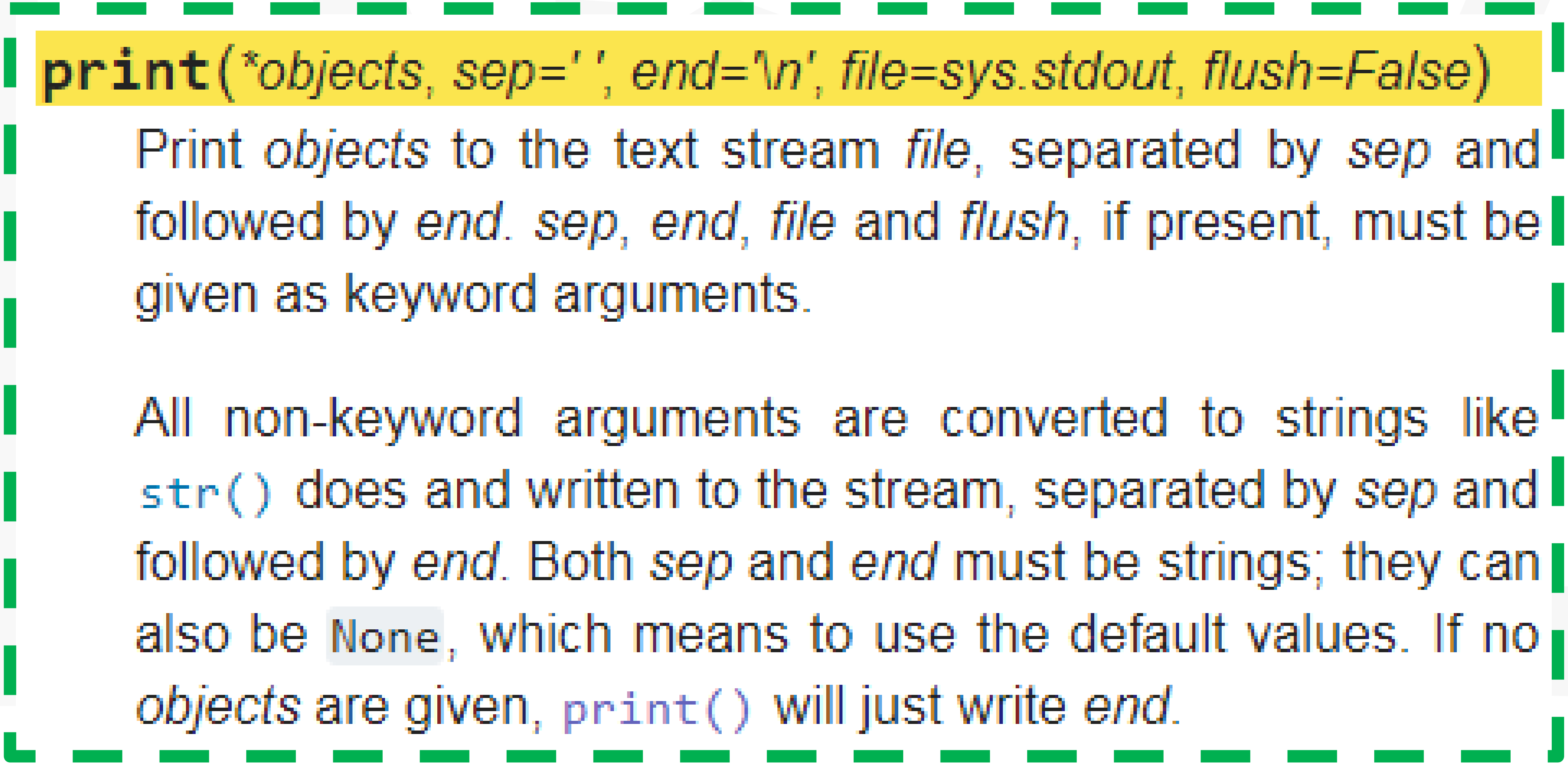
QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Review (Python): `print` function

`sep=' '`

(Dr. Z: Python is OOP. The `str` function will produce different results according to the object it applies to.)



print(**objects*, *sep=' '*, *end='\n'*, *file=sys.stdout*, *flush=False*)

Print *objects* to the text stream *file*, separated by *sep* and followed by *end*. *sep*, *end*, *file* and *flush*, if present, must be given as keyword arguments.

All non-keyword arguments are converted to strings like `str()` does and written to the stream, separated by *sep* and followed by *end*. Both *sep* and *end* must be strings; they can also be `None`, which means to use the default values. If no *objects* are given, `print()` will just write *end*.

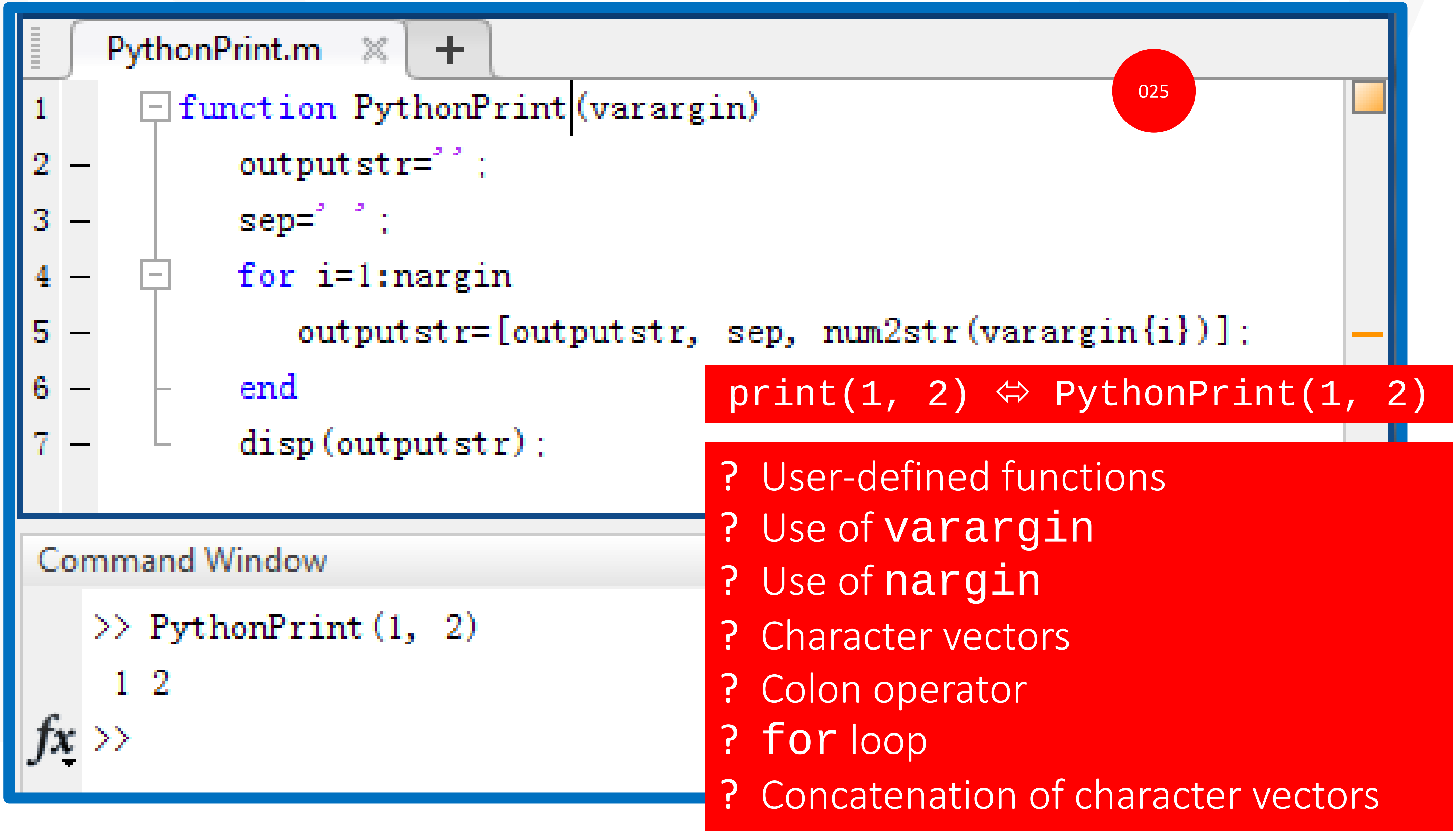
SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance

A temporary MATLAB solution:

(Dr. Z: `PythonPrint.m` can be downloaded from eLearn.)



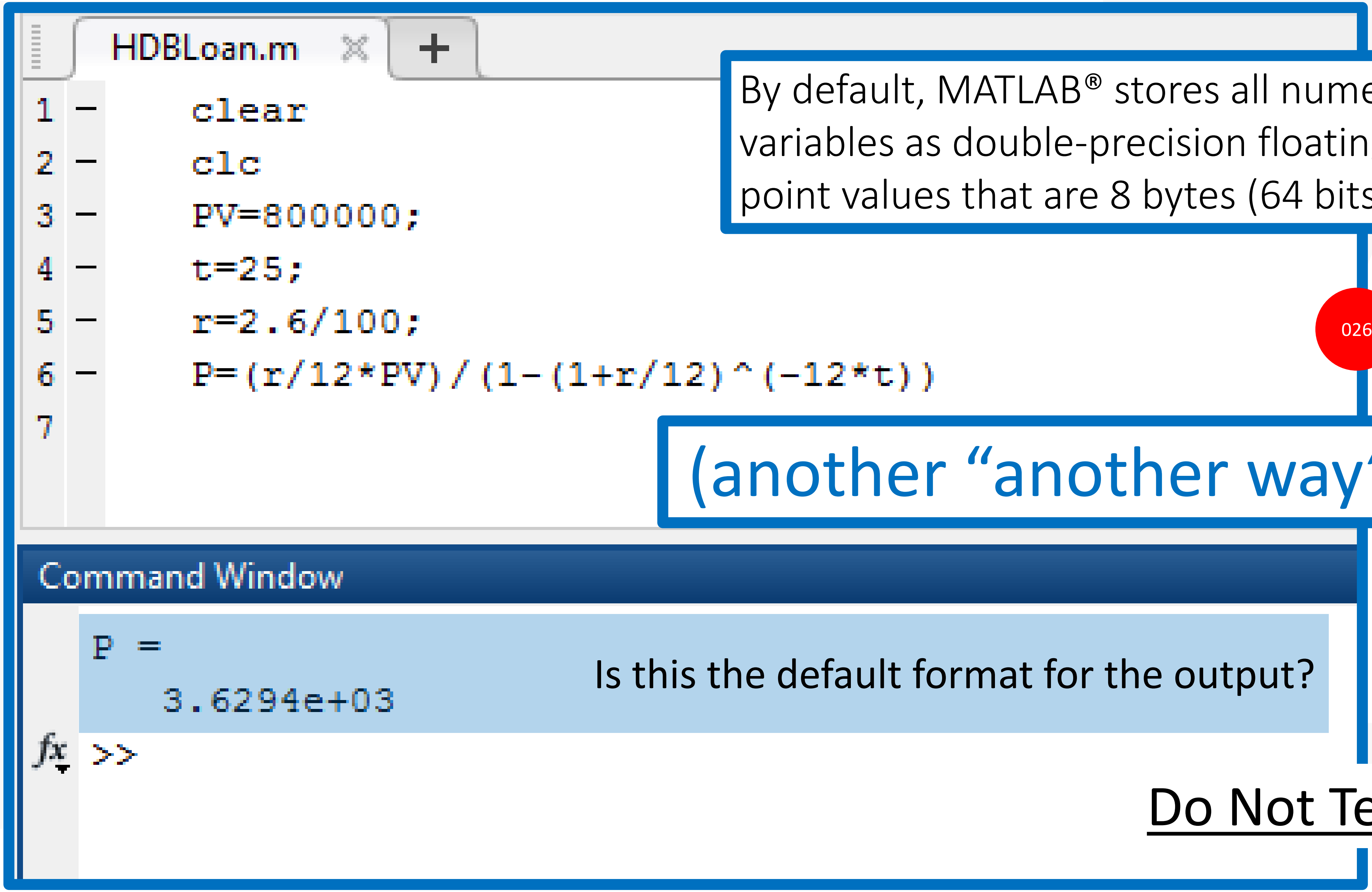
```
function PythonPrint(varargin)
    outputstr='';
    sep=' ';
    for i=1:nargin
        outputstr=[outputstr, sep, num2str(varargin{i})];
    end
    disp(outputstr);
```

```
>> PythonPrint(1, 2)
 1 2
>>
```

025

`print(1, 2)` ⇔ `PythonPrint(1, 2)`

? User-defined functions
? Use of `varargin`
? Use of `nargin`
? Character vectors
? Colon operator
? `for` loop
? Concatenation of character vectors



```
clear
clc
PV=800000;
t=25;
r=2.6/100;
P=(r/12*PV)/(1-(1+r/12)^(-12*t))
```

```
P =

   3.6294e+03

>>
```

By default, MATLAB® stores all numeric variables as double-precision floating-point values that are 8 bytes (64 bits).

026

(another “another way”)

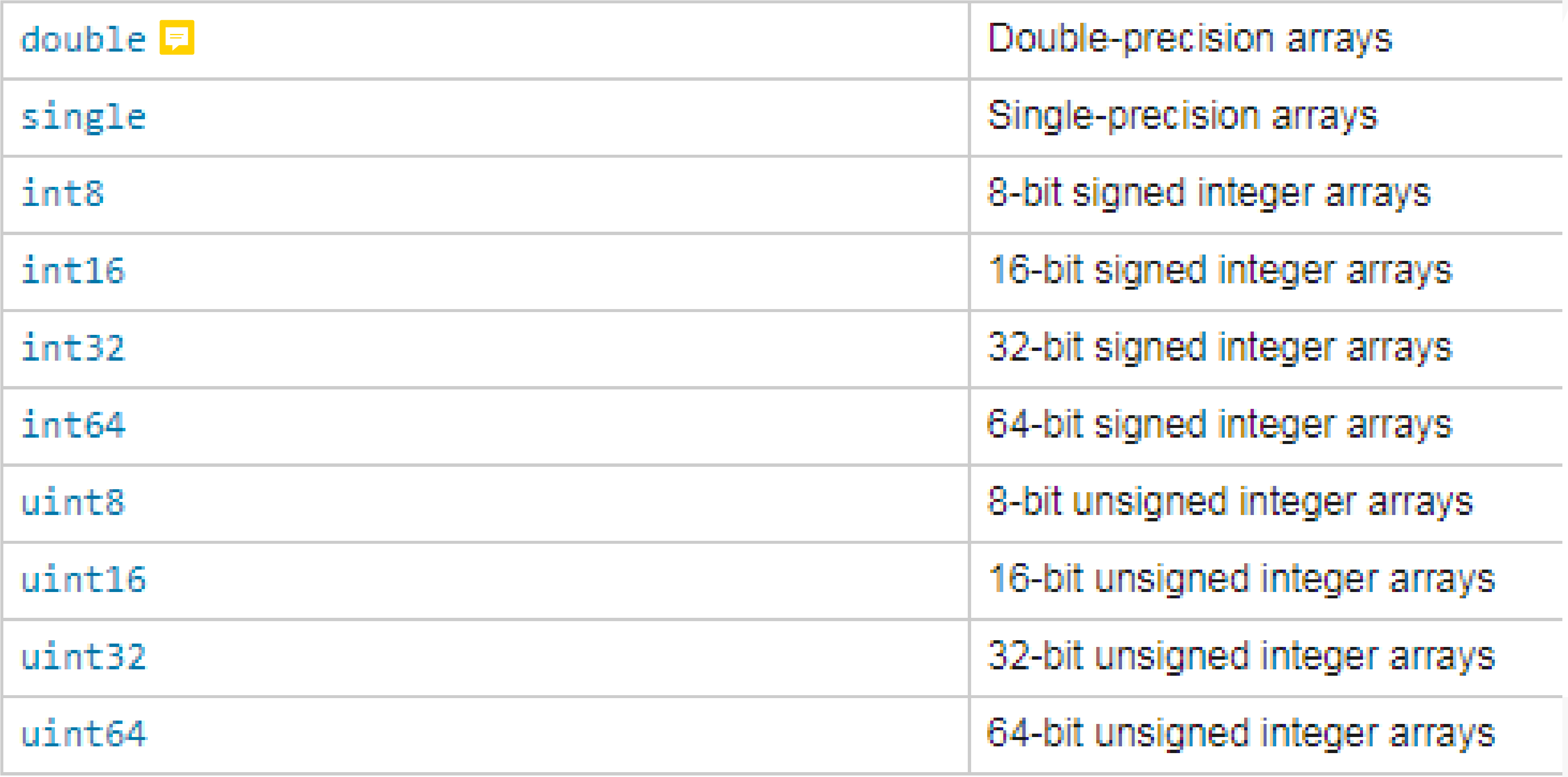
Is this the default format for the output?

Do Not Test

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

https://www.mathworks.com/help/matlab/numeric-types.html



double	Double-precision arrays
single	Single-precision arrays
int8	8-bit signed integer arrays
int16	16-bit signed integer arrays
int32	32-bit signed integer arrays
int64	64-bit signed integer arrays
uint8	8-bit unsigned integer arrays
uint16	16-bit unsigned integer arrays
uint32	32-bit unsigned integer arrays
uint64	64-bit unsigned integer arrays

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

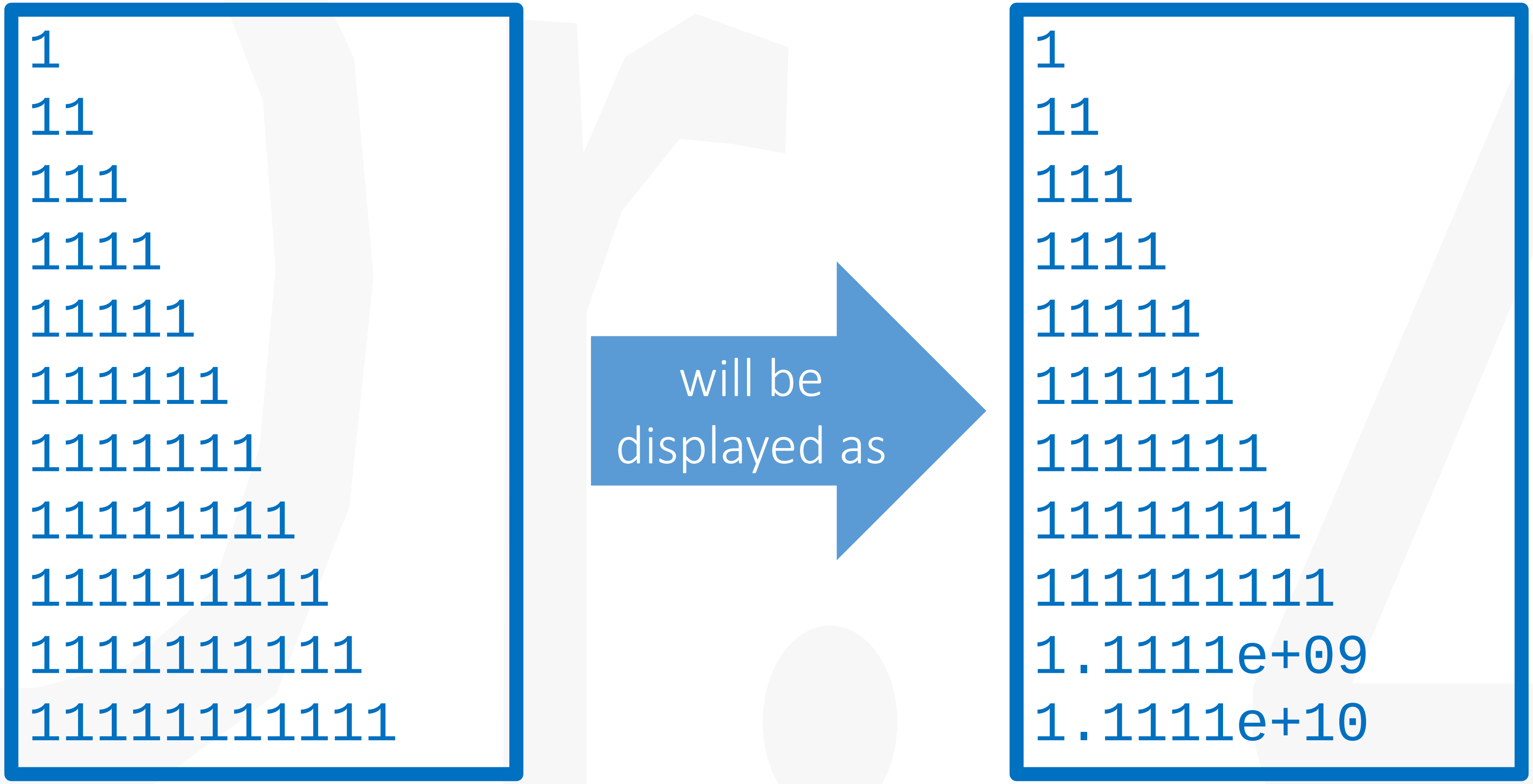
Do Not Test

https://in.mathworks.com/help/matlab/ref/format.html

Do Not Test

Default Format: `format short`

027



```
1
11
111
1111
11111
111111
1111111
11111111
111111111
1111111111
11111111111
```

will be displayed as

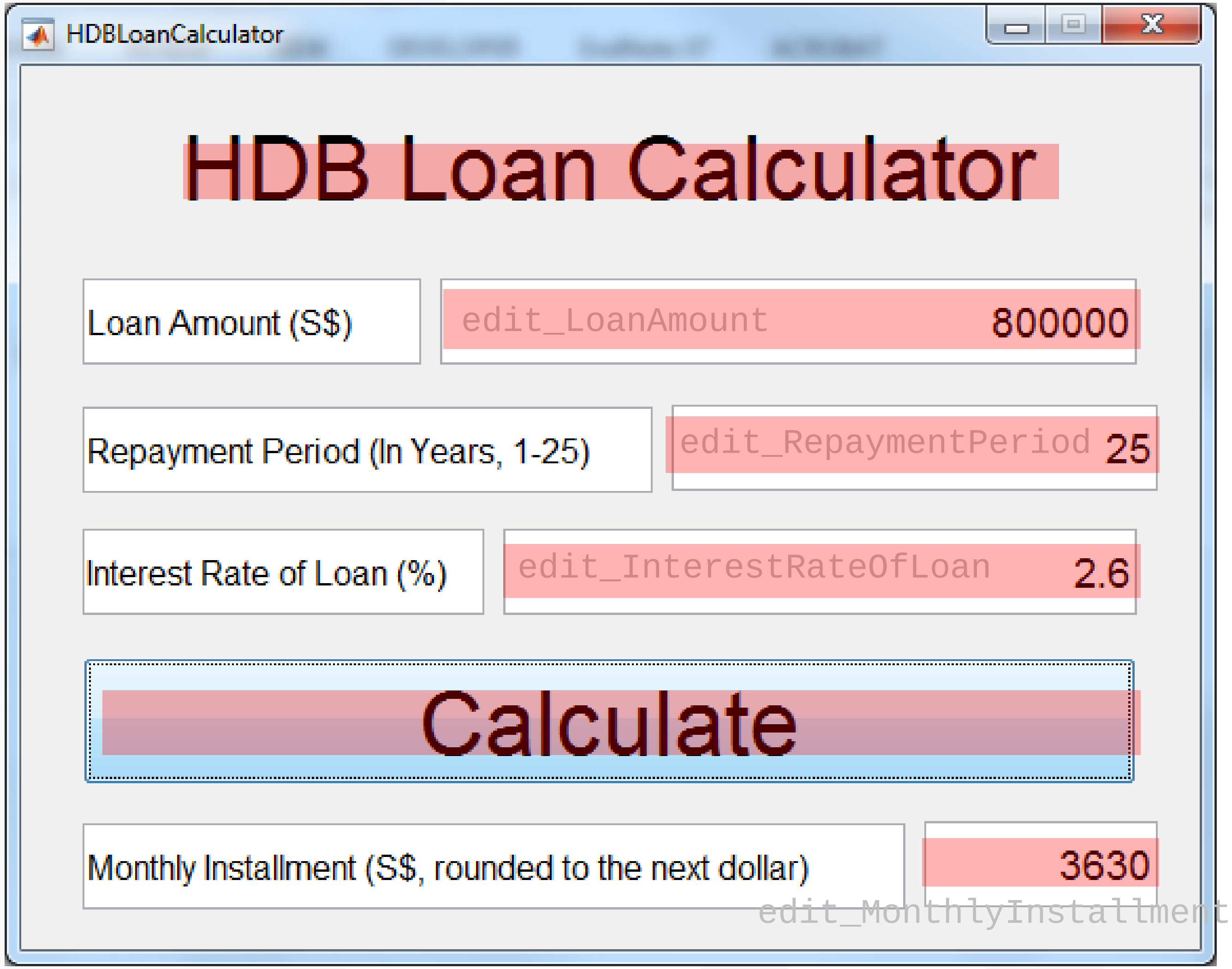
```
1
11
111
1111
11111
111111
1111111
11111111
111111111
1.1111e+09
1.1111e+10
```

4 digits after the decimal point

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance



HDBLoanCalculator

HDB Loan Calculator

Loan Amount (S$)	edit_LoanAmount 800000
Repayment Period (In Years, 1-25)	edit_RepaymentPeriod 25
Interest Rate of Loan (%)	edit_InterestRateOfLoan 2.6

Calculate

Monthly Installment (S$, rounded to the next dollar)	3630

edit_MonthlyInstallment

Python PyQt5 objectName

(see demonstration)

028

029

MATLAB GUIDE Tag

Homework Q1

Do Not Test

Do Not Test

Type “guide” in Command Window, and ...

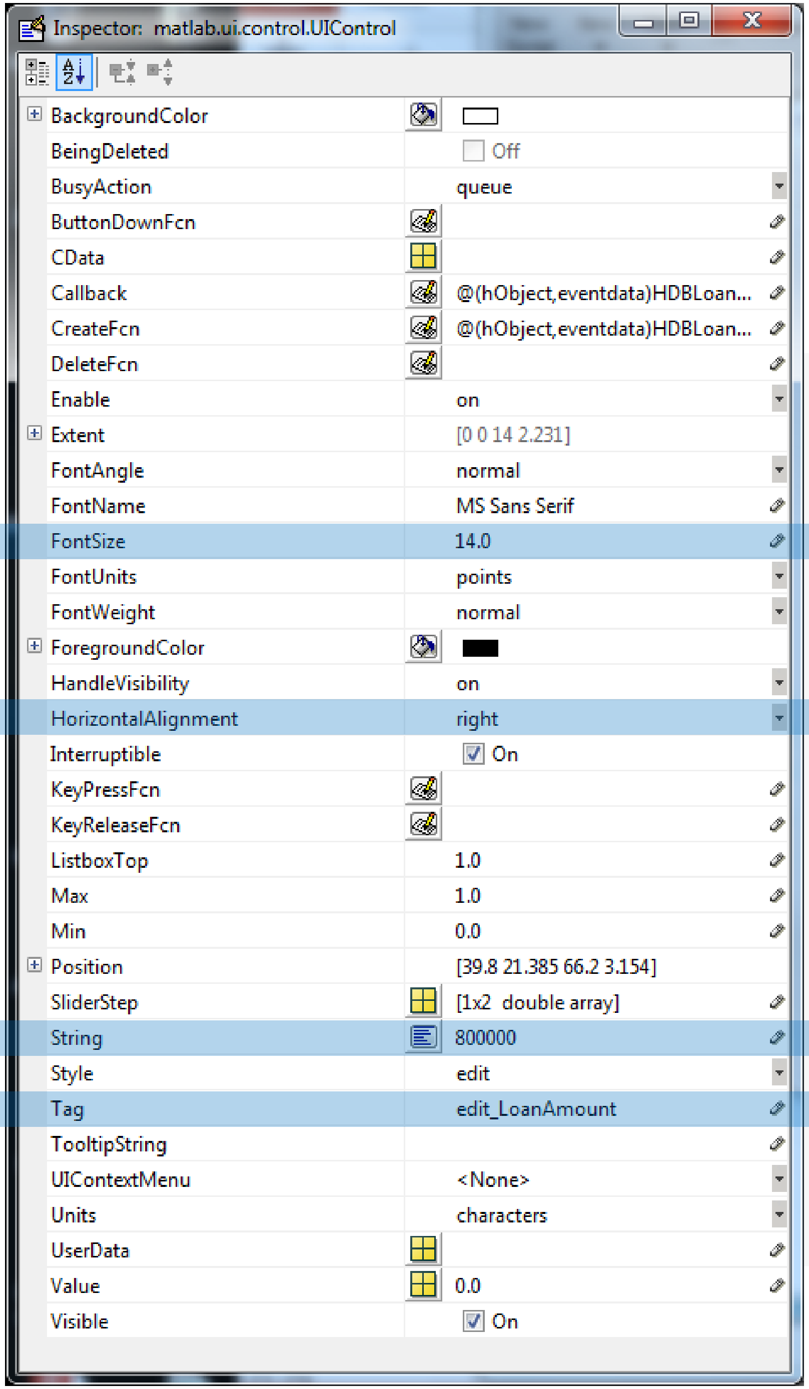
Important Tag names:

- ✓ **`edit_LoanAmount`** (see demonstration on the next slide)
- ✓ `edit_RepaymentPeriod`
- ✓ `edit_InterestRateOfLoan`
- ✓ `edit_MonthlyInstallment`

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



Inspector: matlab.ui.control.UIControl

BackgroundColor	□
BeingDeleted	☐ Off
BusyAction	queue
ButtonDownFcn	
CData	
Callback	@(hObject,eventdata)HDBLoan...
CreateFcn	@(hObject,eventdata)HDBLoan...
DeleteFcn	
Enable	on
Extent	[0 0 14 2.231]
FontAngle	normal
FontName	MS Sans Serif
FontSize	14.0
FontUnits	points
FontWeight	normal
ForegroundColor	■
HandleVisibility	on
HorizontalAlignment	right
Interruptible	☑ On
KeyPressFcn	
KeyReleaseFcn	
ListboxTop	1.0
Max	1.0
Min	0.0
Position	[39.8 21.385 66.2 3.154]
SliderStep	[1x2 double array]
String	800000
Style	edit
Tag	edit_LoanAmount
TooltipString	
UIContextMenu	<None>
Units	characters
UserData	
Value	0.0
Visible	☑ On

1. Change **FontSize**

2. Change **HorizontalAlignment**

3. Change **String**

4. Change **Tag** to

   `edit_LoanAmount`

029

Do Not Test

```
% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

PV=str2num(handles.edit_LoanAmount.String);
t=str2num(handles.edit_RepaymentPeriod.String);
r=str2num(handles.edit_InterestRateOfLoan.String)/100;
P=(r/12*PV)/(1-(1+r/12)^(-12*t));
handles.edit_MonthlyInstallment.String = num2str(ceil(P));
```

Do Not Test

- □ handles.__tag__.String
- □ str2num, num2str
- □ ceil
- □ +, -, *, /, ^, ()

Do Not Test

029

030

031

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Mathematical Functions

https://www.mathworks.com/help/symbolic/mathematical-functions.html

$\ln(x)$ ⇨ `log(x)`

$\log_{10}(x)$ ⇨ `log10(x)`

$\log_{2}(x)$ ⇨ `log2(x)`

e^{x} ⇨ `exp(x)`

$\sin(x)$ ⇨ `sin(x)`

$\cos(x)$ ⇨ `cos(x)`

$|x|$ ⇨ `abs(x)`

$\tan(x)$ ⇨ `tan(x)`

$\cot(x)$ ⇨ `cot(x)`

$\sec(x)$ ⇨ `sec(x)`

$\mathrm{asin}(x)$ ⇨ `asin(x)`

$\mathrm{acos}(x)$ ⇨ `acos(x)`

$\lfloor x \rfloor$ ⇨ `floor(x)`

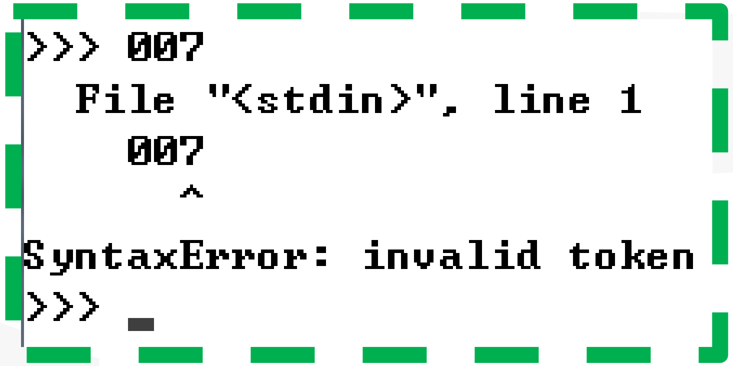
$\lceil x \rceil$ ⇨ `ceil(x)`

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

MATLAB:

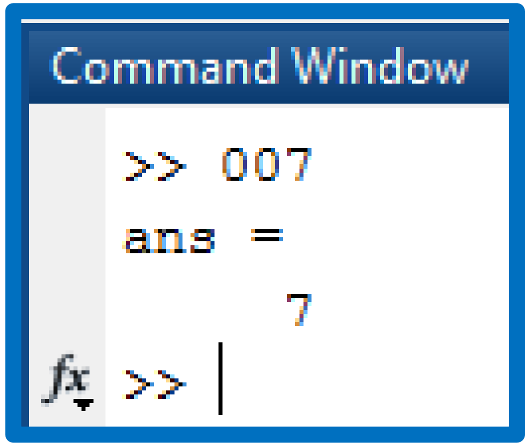


```
>>> 007
  File "<stdin>", line 1
    007
      ^
SyntaxError: invalid token
>>> _
```

033

Q: Can I use 007?

Integer Literals



```
>> 007

ans =

     7

>> |
```

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

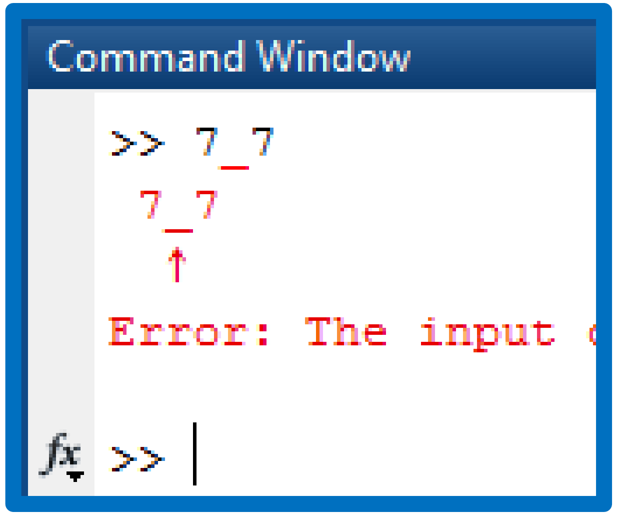
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

MATLAB:

```
>>> 7_7
77
```

034

Q: Can I use 7_7?



```
>> 7_7
 7_7
  ↑
Error: The input 

>> 
```

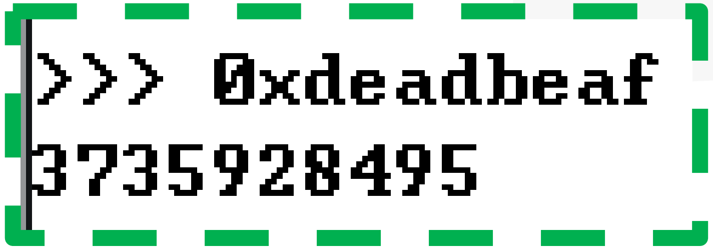
Integer Literals

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

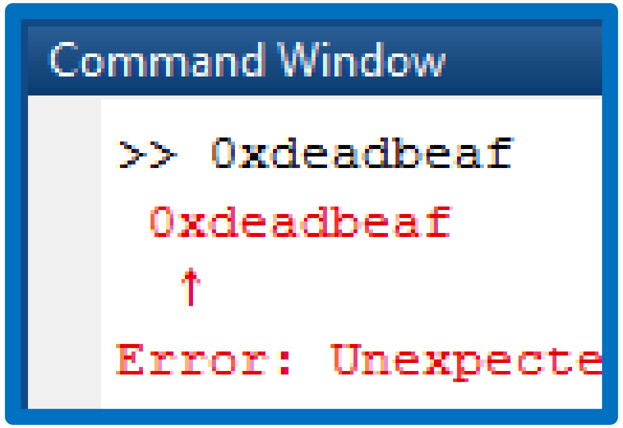
MATLAB:



```
>>> 0xdeadbeaf
3735928495
```

035

Q: Can I use `0xdeadbeaf`?



```
>> 0xdeadbeaf
 0xdeadbeaf
   ↑
Error: Unexpecte
```

Integer Literals

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

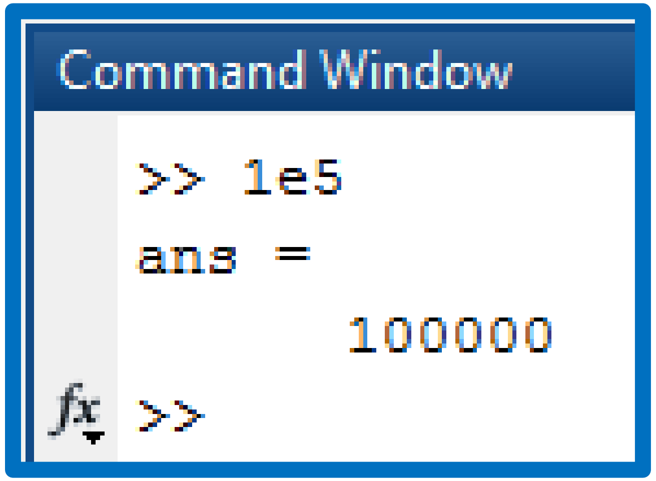
MATLAB:

```
>>> 1e5
100000.0
```

036

Q: Can I use 1e5?

Float Literals



```
>> 1e5
ans =
       100000
>>
```

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

MATLAB String and Character Arrays

https://www.mathworks.com/help/matlab/matlab_prog/creating-character-arrays.html

□ **Character Vector**: A sequence of characters enclosed in **single** quotation marks.

037

□ **Character Array**: Character vectors stored as rows in a 2D array. Vectors need to be of the same number of characters.

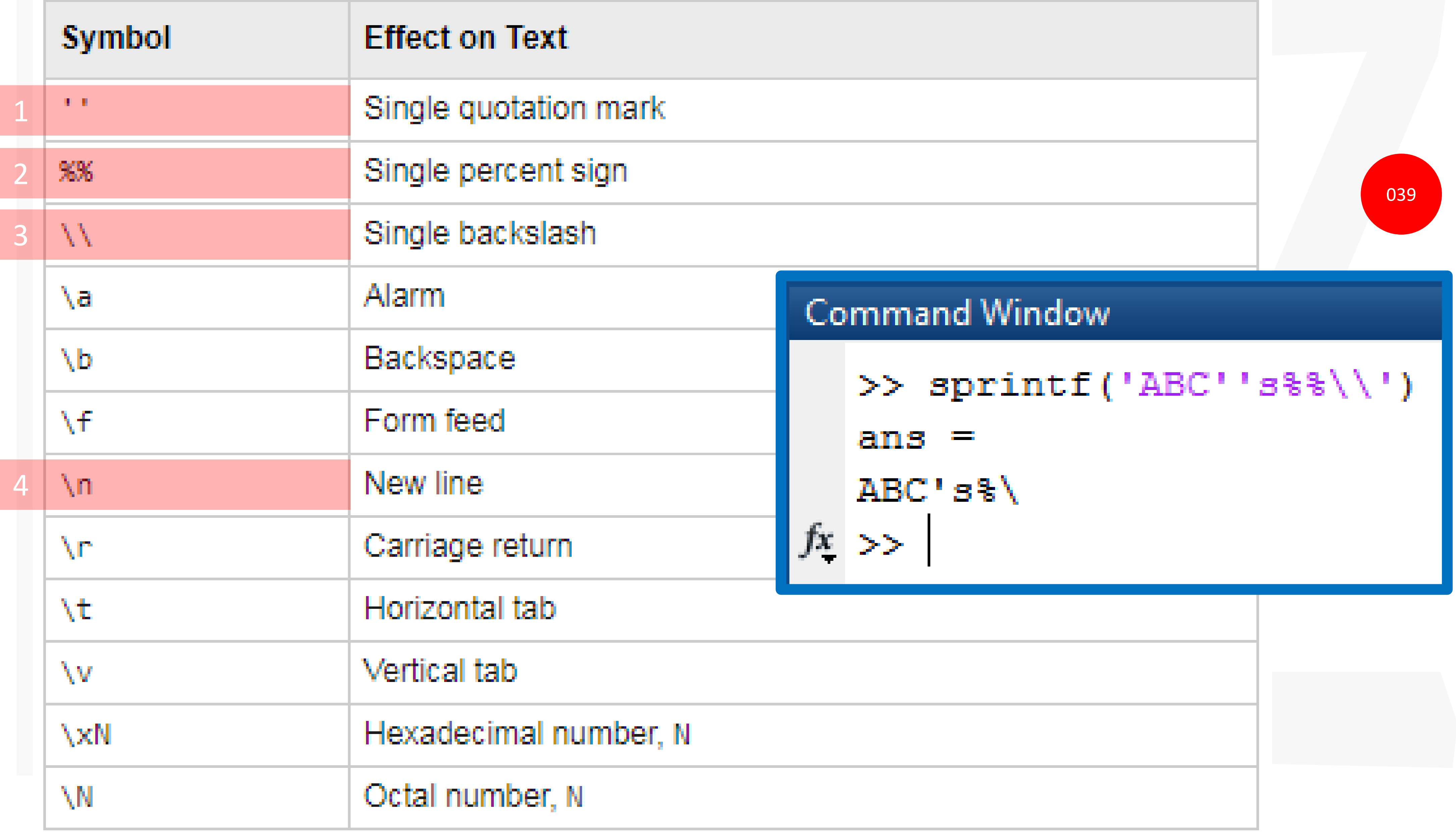
038

□ **String**: Starting in R2017a, we can create a string using **double** quotes. (New built-in function: `string`, `strings`, `split`, etc.)

□ **String Array**: Create a string array using the [ ] operator, which is similar to storing character vectors in a cell array using the { } operator. (Dr. Z: This is a bit awkward.)

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

https://www.mathworks.com/help/matlab/matlab_prog/matlab-operators-and-special-characters.html



	Symbol	Effect on Text
1	''	Single quotation mark
2	%%	Single percent sign
3	\\	Single backslash
	\a	Alarm
	\b	Backspace
	\f	Form feed
4	\n	New line
	\r	Carriage return
	\t	Horizontal tab
	\v	Vertical tab
	\xN	Hexadecimal number, N
	\N	Octal number, N

Command Window

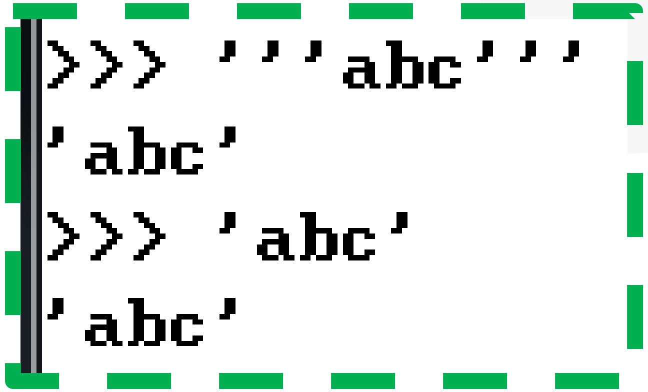
```
>> sprintf('ABC''s%%\\')
ans =
ABC's%\
>> 
```

039

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

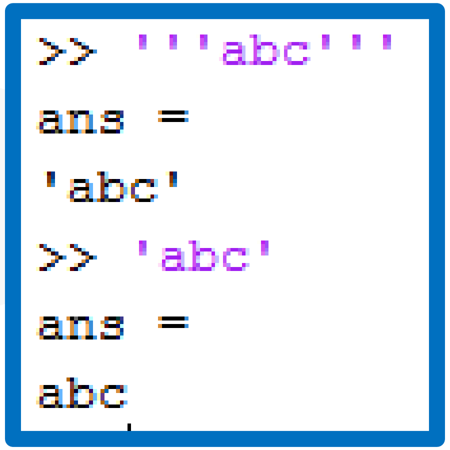
MATLAB:



```
>>> '''abc'''
'abc'
>>> 'abc'
'abc'
```

Q: Can I use triple-quotes to create strings?

040



```
>> '''abc'''
ans =
'abc'
>> 'abc'
ans =
abc
```

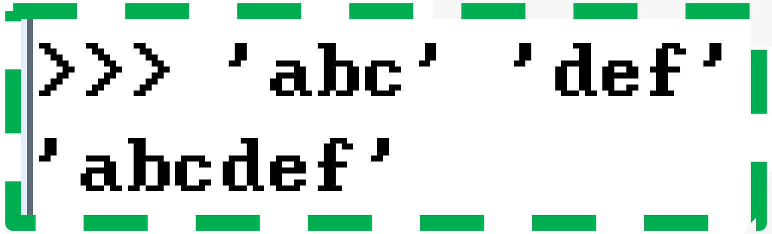
String Literals

SMU
SINGAPORE MANAGEMENT
UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

MATLAB:

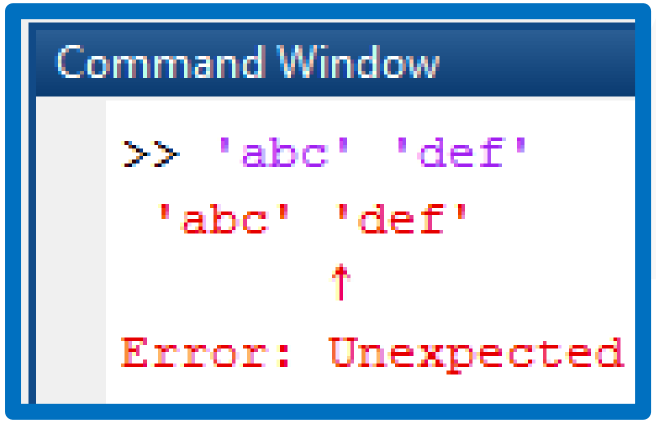


```
>>> 'abc' 'def'
'abcdef'
```

041

Q: Will two adjacent string literals be concatenated ?

String Concatenation



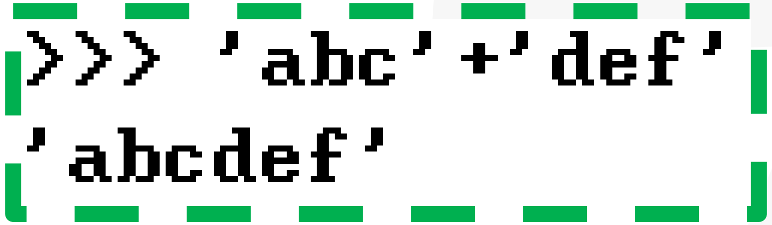
```
>> 'abc' 'def'
 'abc' 'def'
       ↑
Error: Unexpected
```

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

MATLAB:

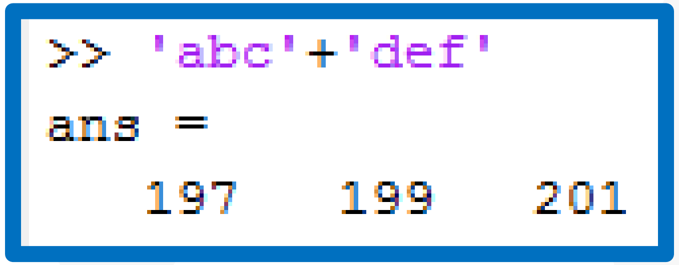


```
>>> 'abc'+'def'
'abcdef'
```

042

Q: Does the sum of two string literals mean concatenation?

String Concatenation



```
>> 'abc'+'def'
ans =
   197   199   201
```

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

SMU
SINGAPORE MANAGEMENT UNIVERSITY

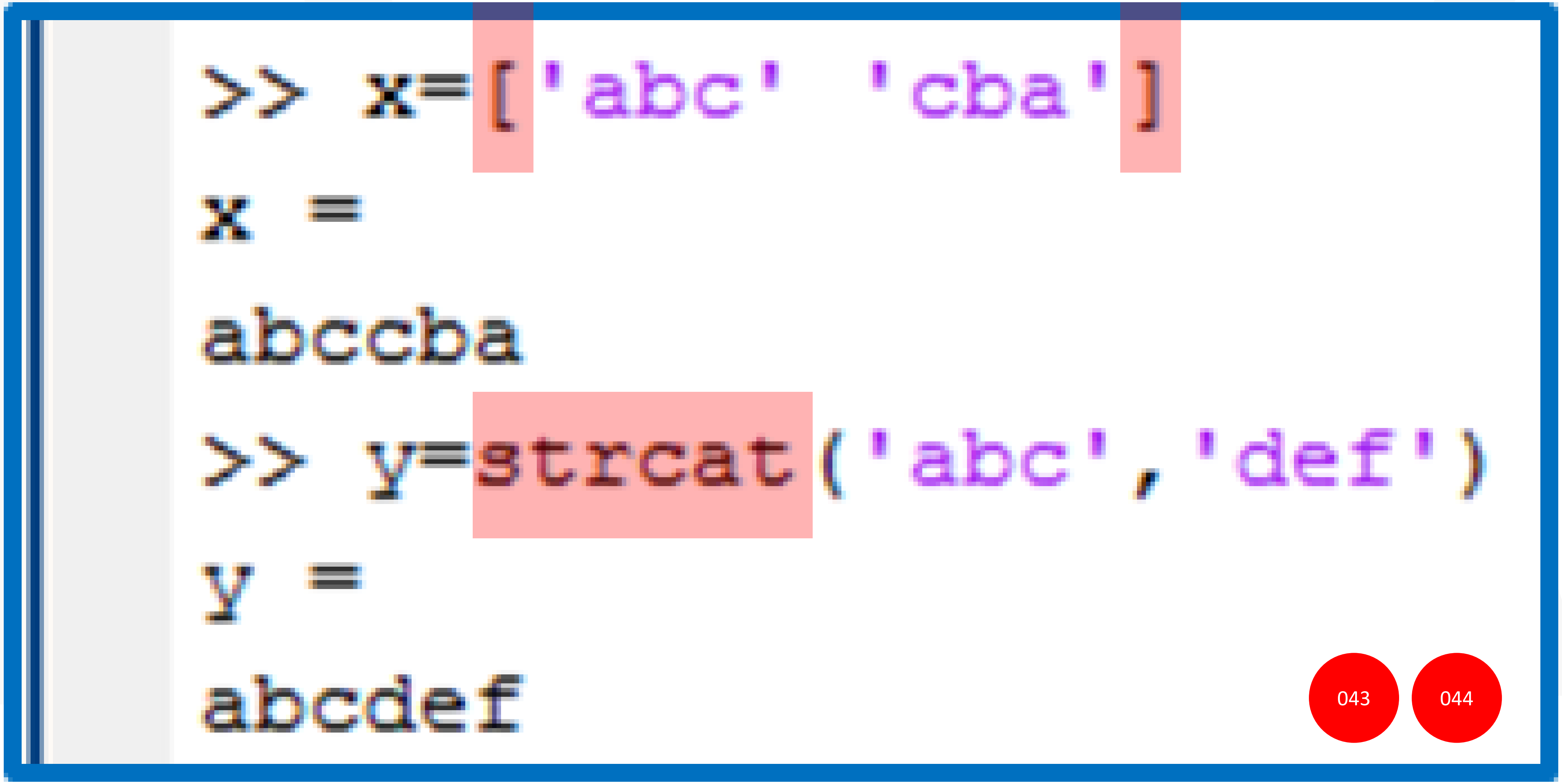
QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

MATLAB Basic String Operations:

- □ H-Concatenation using the [ ] operator 043
- □ H-Concatenation using the `strcat` function 044
- □ **Indexing and slicing (★★★★★)**
- □ Comparison using the `strcmp` function 045
- □ Find the first occurrence of one string in another. (Dr. Z: Note that we need this operation in the Sudoku Solver.) 046
- □ Others: ______________________

Character Vector Concatenation using [ ] operator and `strcat`



```
>> x=['abc' 'cba']

x =

abcba

>> y=strcat('abc','def')

y =

abcdef
```

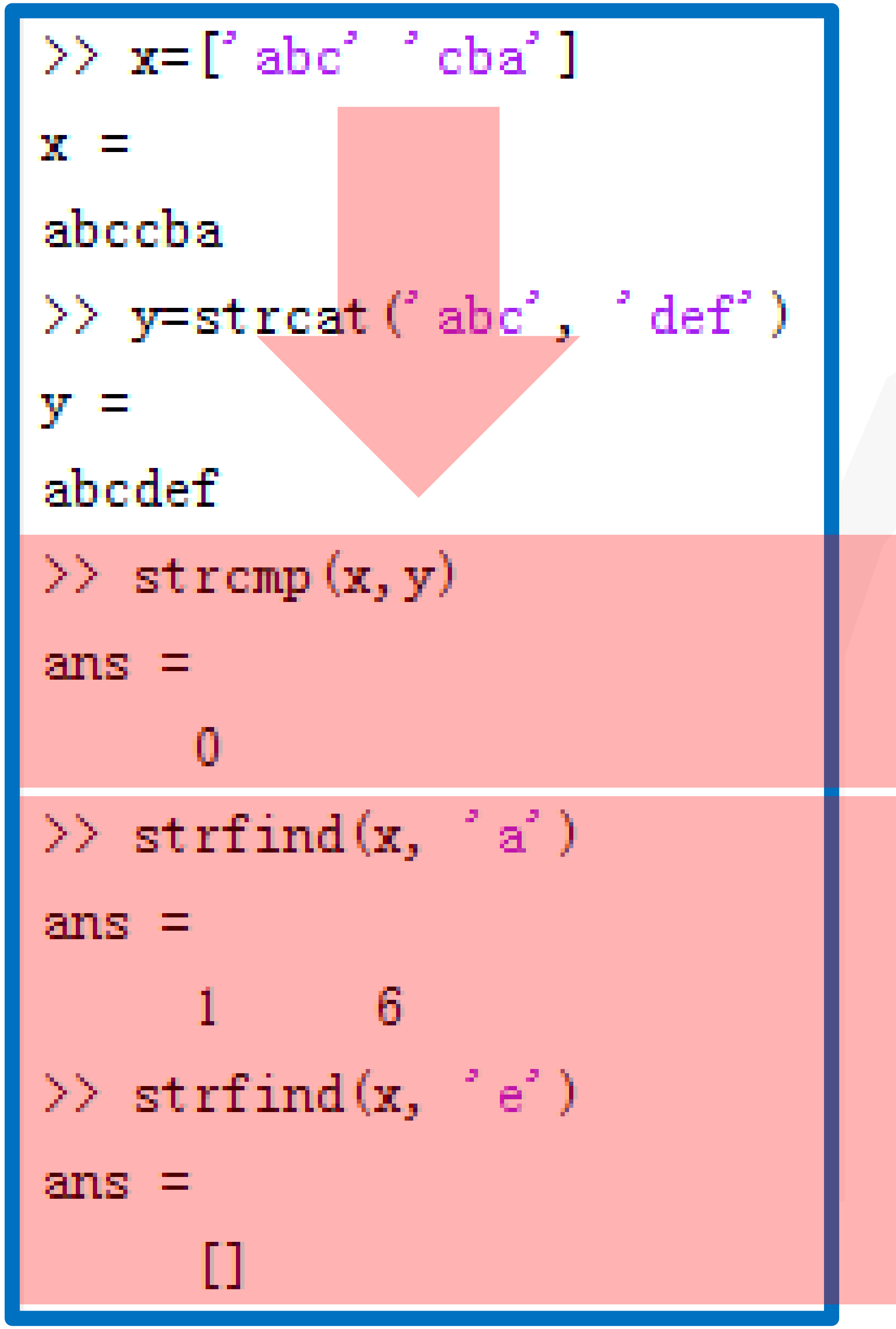
SMU
SINGAPORE MANAGEMENT UNIVERSITY

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



```
>> x=['abc' 'cba']

x =

abccba

>> y=strcat('abc', 'def')

y =

abcdef

>> strcmp(x,y)

ans =

     0

>> strfind(x, 'a')

ans =

     1     6

>> strfind(x, 'e')

ans =

     []
```

- ✓ strcmp
- ✓ strfind

045 046

0 ?

1 6 ?

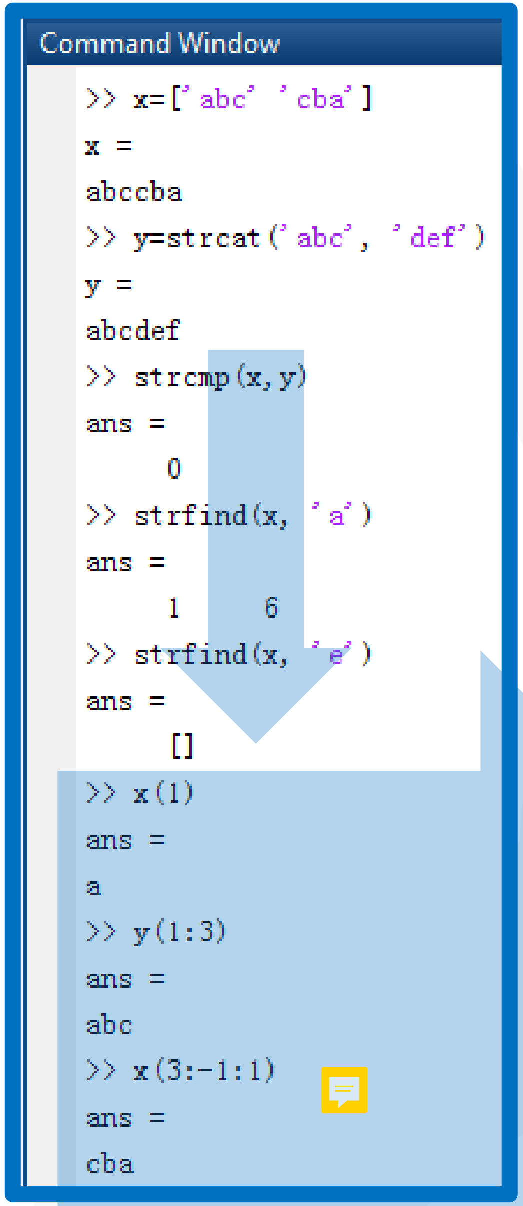
[] ?

Slide
055/160

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance



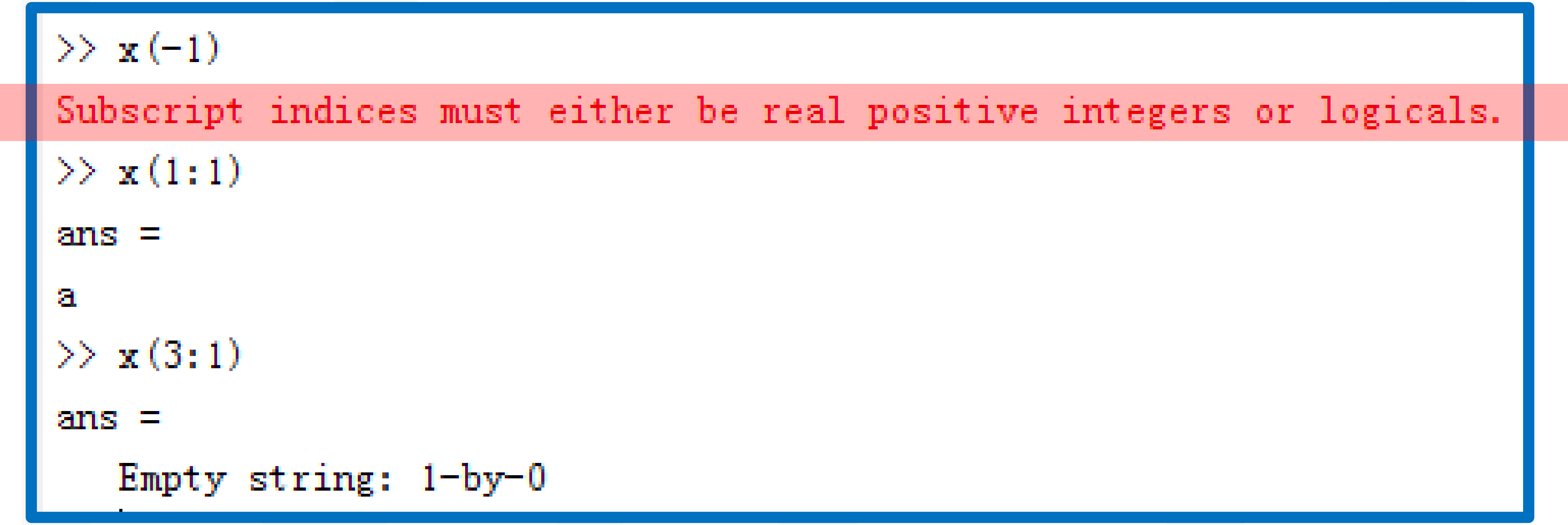
```
>> x=['abc' 'cba']
x =
abccba
>> y=strcat('abc', 'def')
y =
abcdef
>> strcmp(x,y)
ans =
     0
>> strfind(x, 'a')
ans =
     1     6
>> strfind(x, 'e')
ans =
     []
>> x(1)
ans =
a
>> y(1:3)
ans =
abc
>> x(3:-1:1)
ans =
cba
```

047 048 049 050 051 052 053

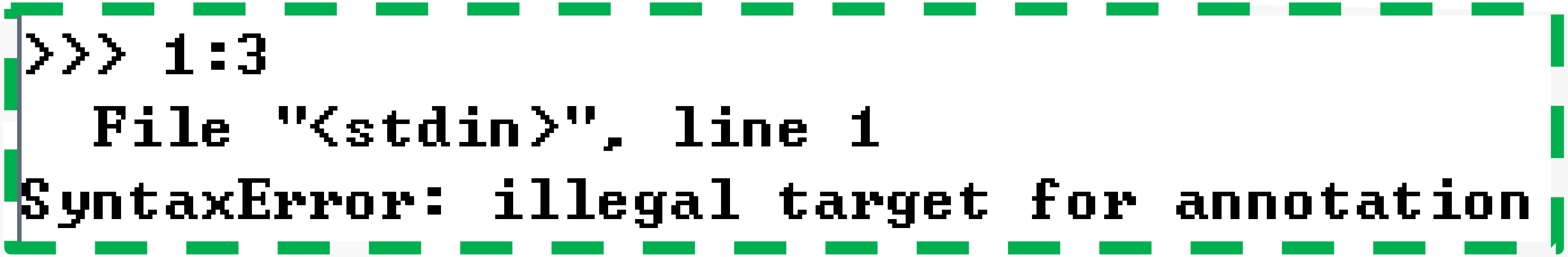
- ✓Indexing and Slicing use braces.
- ✓The index of the first element is 1.
- ✓Negative index is not allowed.
- ✓Slicing "start:end" **includes** "end".
- ✓Extended slicing format: "start:step:end"
- ✓`x(1:1)` is not empty
- ✓`x(3:1)` is empty
- ✓1<=start, end<=length

✓ Negative index is not allowed.

049 052 053

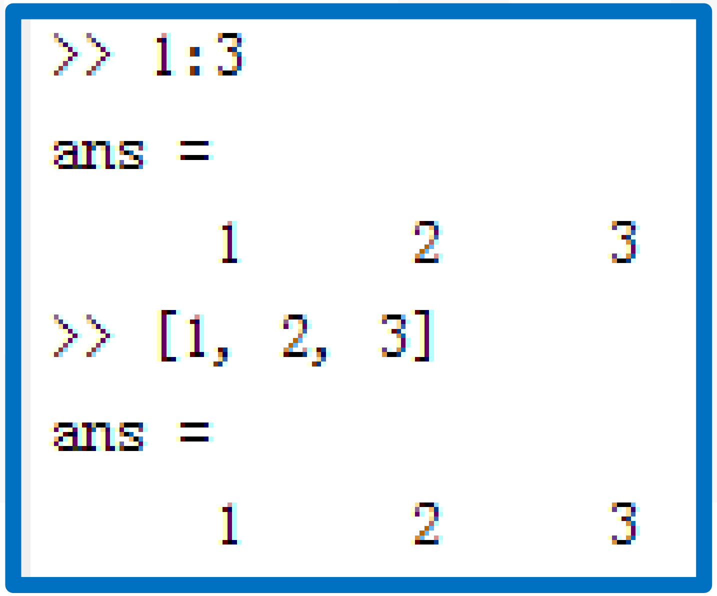


```
>> x(-1)
Subscript indices must either be real positive integers or logicals.
>> x(1:1)

ans =

a

>> x(3:1)

ans =

   Empty string: 1-by-0
```



```
>>> 1:3
  File "<stdin>", line 1
SyntaxError: illegal target for annotation
```

1:3



```
>> 1:3

ans =

     1     2     3

>> [1, 2, 3]

ans =

     1     2     3
```

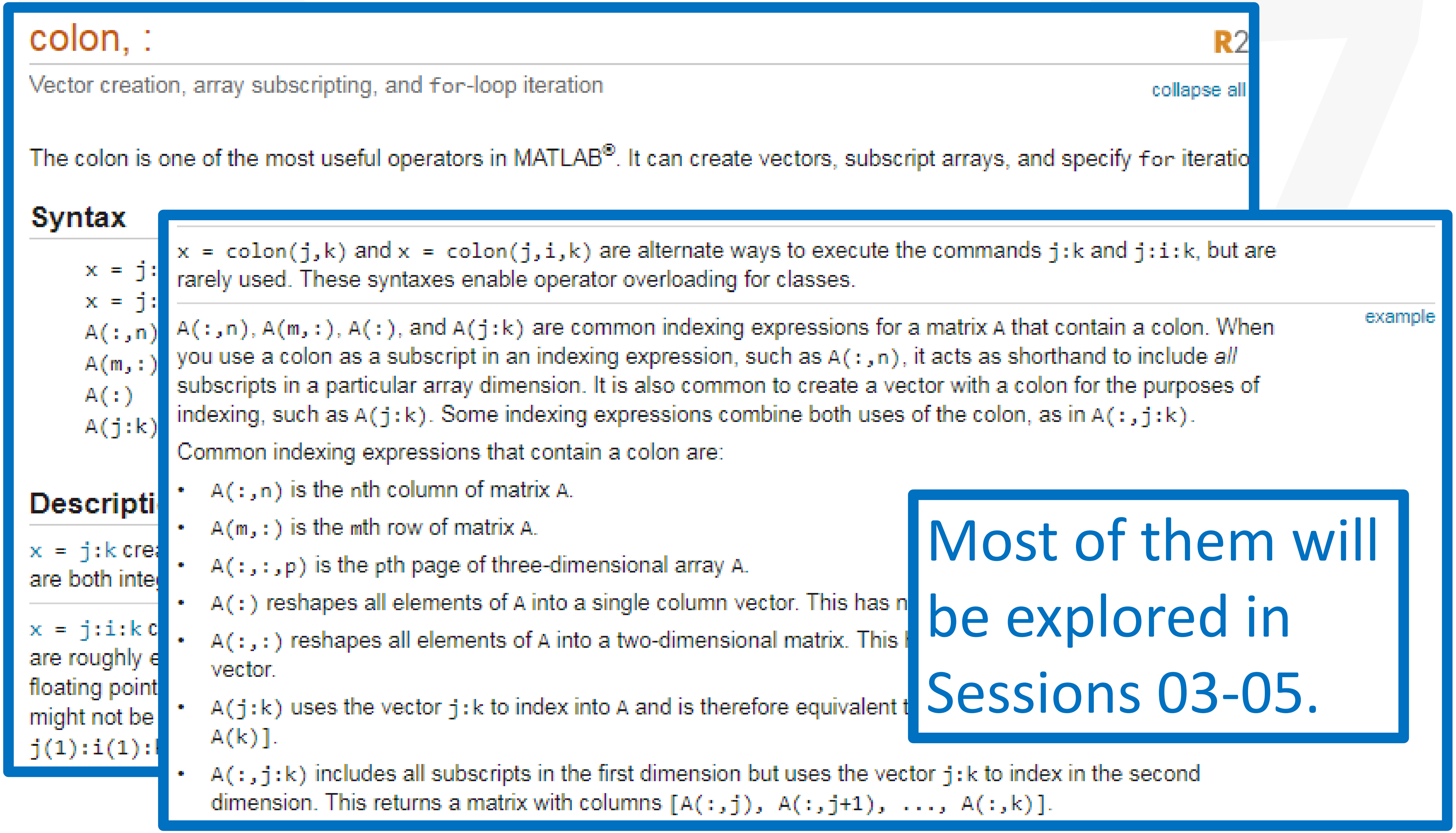
054

1:3 ⇔ [1, 2, 3]

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

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



colon, :

R2

Vector creation, array subscripting, and for-loop iteration

collapse all

The colon is one of the most useful operators in MATLAB®. It can create vectors, subscript arrays, and specify for iteratio

Syntax

```
x = j:
x = j:
A(:,n)
A(m,:)
A(:)
A(j:k)
```

Descripti

x = j:k crea
are both inte

x = j:i:k c
are roughly e
floating point
might not be
j(1):i(1):

x = colon(j,k) and x = colon(j,i,k) are alternate ways to execute the commands j:k and j:i:k, but are rarely used. These syntaxes enable operator overloading for classes.

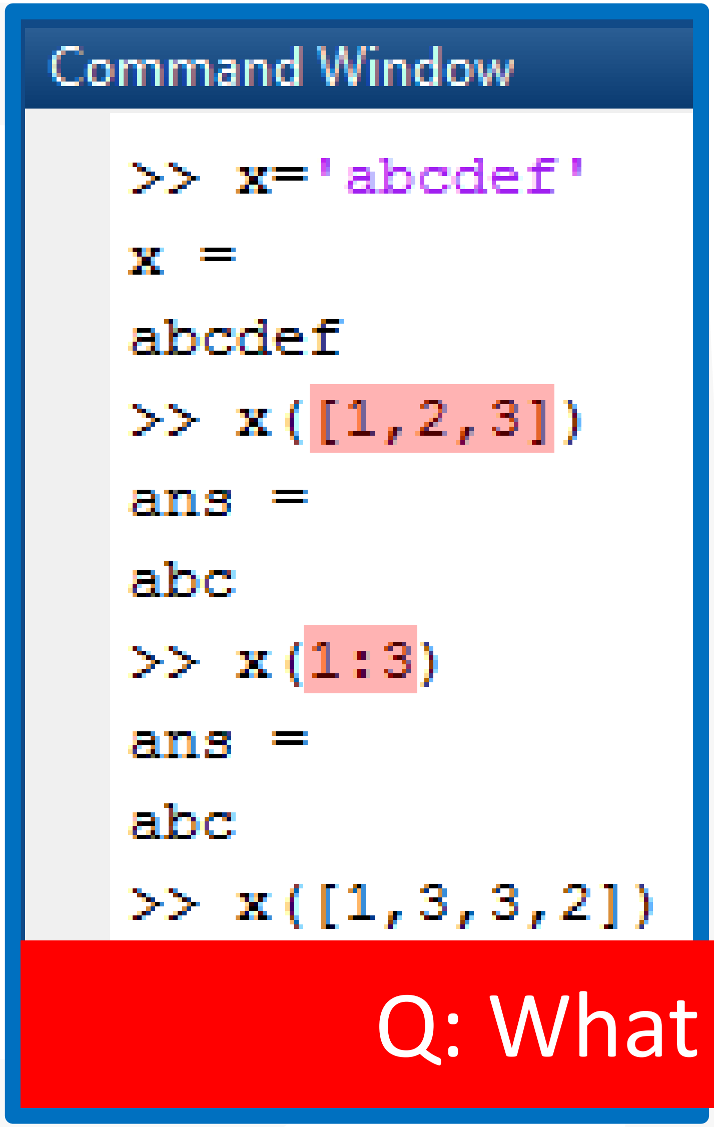
example

A(:,n), A(m,:), A(:), and A(j:k) are common indexing expressions for a matrix A that contain a colon. When you use a colon as a subscript in an indexing expression, such as A(:,n), it acts as shorthand to include *all* subscripts in a particular array dimension. It is also common to create a vector with a colon for the purposes of indexing, such as A(j:k). Some indexing expressions combine both uses of the colon, as in A(:,j:k).

Common indexing expressions that contain a colon are:

- A(:,n) is the nth column of matrix A.
- A(m,:) is the mth row of matrix A.
- A(:,:,p) is the pth page of three-dimensional array A.
- A(:) reshapes all elements of A into a single column vector. This has n
- A(:,:) reshapes all elements of A into a two-dimensional matrix. This
vector.
- A(j:k) uses the vector j:k to index into A and is therefore equivalent t
A(k)].
- A(:,j:k) includes all subscripts in the first dimension but uses the vector j:k to index in the second dimension. This returns a matrix with columns [A(:,j), A(:,j+1), ..., A(:,k)].

Most of them will be explored in Sessions 03-05.

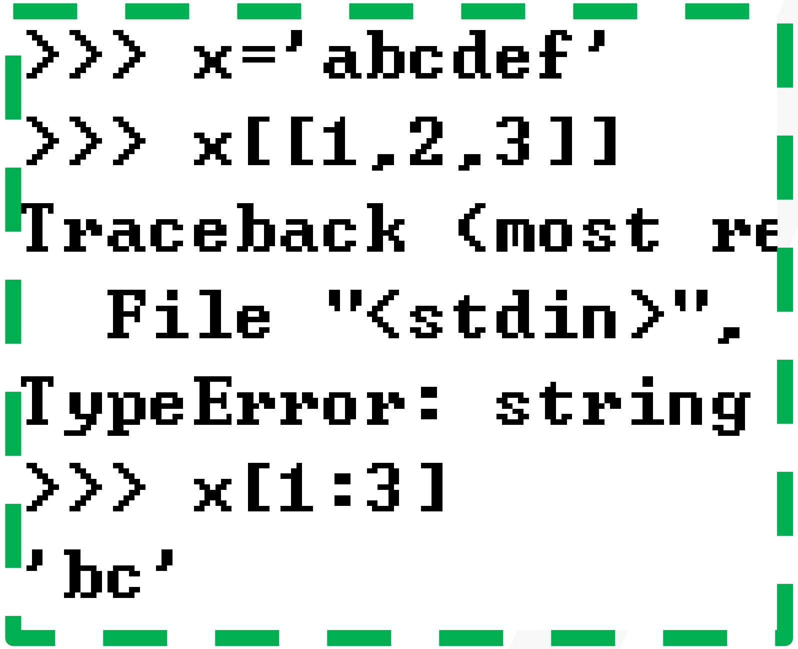


```
>> x='abcdef'
x =
abcdef
>> x([1,2,3])
ans =
abc
>> x(1:3)
ans =
abc
>> x([1,3,3,2])
```

Python Core is not vectorization-ready.

055

056



```
>>> x='abcdef'
>>> x[[1,2,3]]
Traceback (most re
  File "<stdin>",
TypeError: string
>>> x[1:3]
'bc'
```

Q: What is the output?

Q: How to create a 10-letter random “word” using the letters from a given word?

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance

QF666
Programming and
Computational
Finance

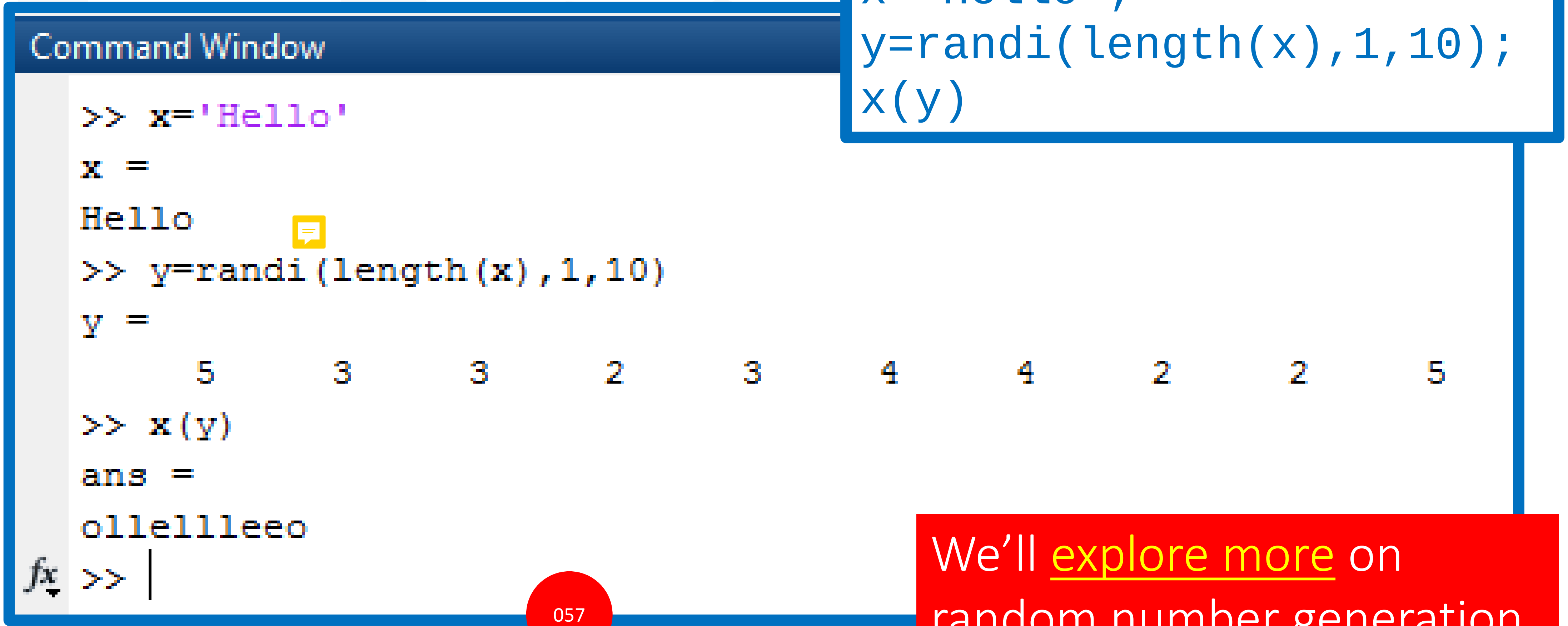
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Q: How to create a 10-letter random “word” using the letters from a given word, say “Hello”?

In MATLAB, this problem is solved through the following steps:

1. Let `x='Hello'`
2. Generate a `10`-element array of random integers from `1` to `5` (i.e. the length of `x`) inclusive and name it `y`
3. The answer of the problem: `x(y)`

```
clear
clc
x='Hello';
y=randi(length(x),1,10);
x(y)
```



```
>> x='Hello'
x =
Hello
>> y=randi(length(x),1,10)
y =
     5     3     3     2     3     4     4     2     2     5
>> x(y)
ans =
ollellleeo
>> 
```

We'll explore more on random number generation in a subsequent session.

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

Q: How to create a 10-letter random “word” using the letters from a given word, say “Hello”?

Use Python

Please pretend you have not learned Numpy yet.

Noticing that Python does not have the indexing feature in MATLAB as shown in the previous slide, we need a different approach:

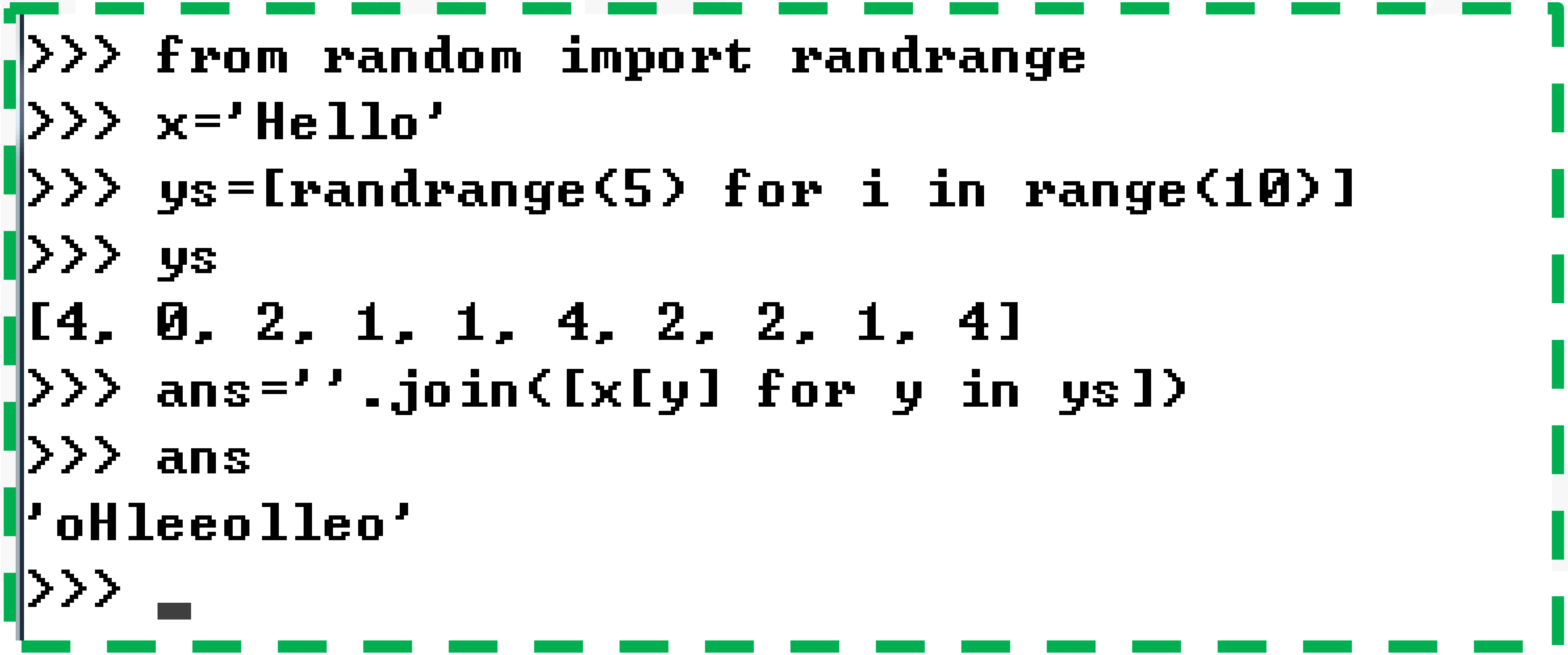
1. Let `x='Hello'`
2. Use list comprehension to generate a list of `10` random integers from `0` to `4` (i.e. `len(x)-1`) inclusive and name it `ys`. (see `random.randrange`)
3. Use list comprehension to get a list of single letters from `x` at locations respectively equal to the random integers in `ys`.
4. Use string join method to combine all letters to get the answer.

[Hint: 2 and 3 can be combined if you know how to use `random.choice`.)

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

```
from random import randrange
x='Hello'
ys=[randrange(5) for i in range(10)]
ans=''.join([x[y] for y in ys])
ans
```



```
>>> from random import randrange
>>> x='Hello'
>>> ys=[randrange(5) for i in range(10)]
>>> ys
[4, 0, 2, 1, 1, 4, 2, 2, 1, 4]
>>> ans=''.join([x[y] for y in ys])
>>> ans
'oHleeolleo'
>>> 
```

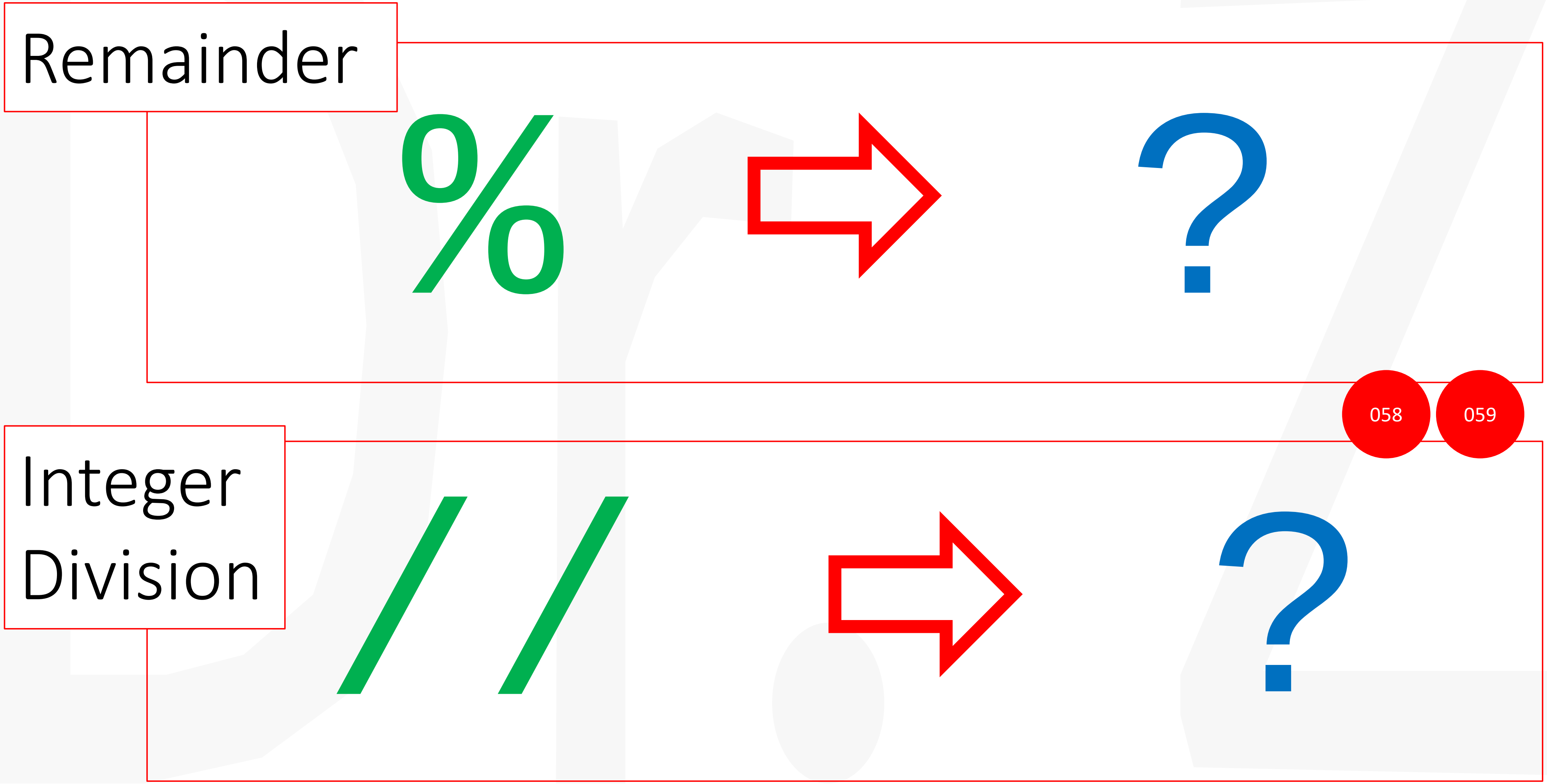
Any Question?

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

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

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



Remainder

% → ?

Integer Division

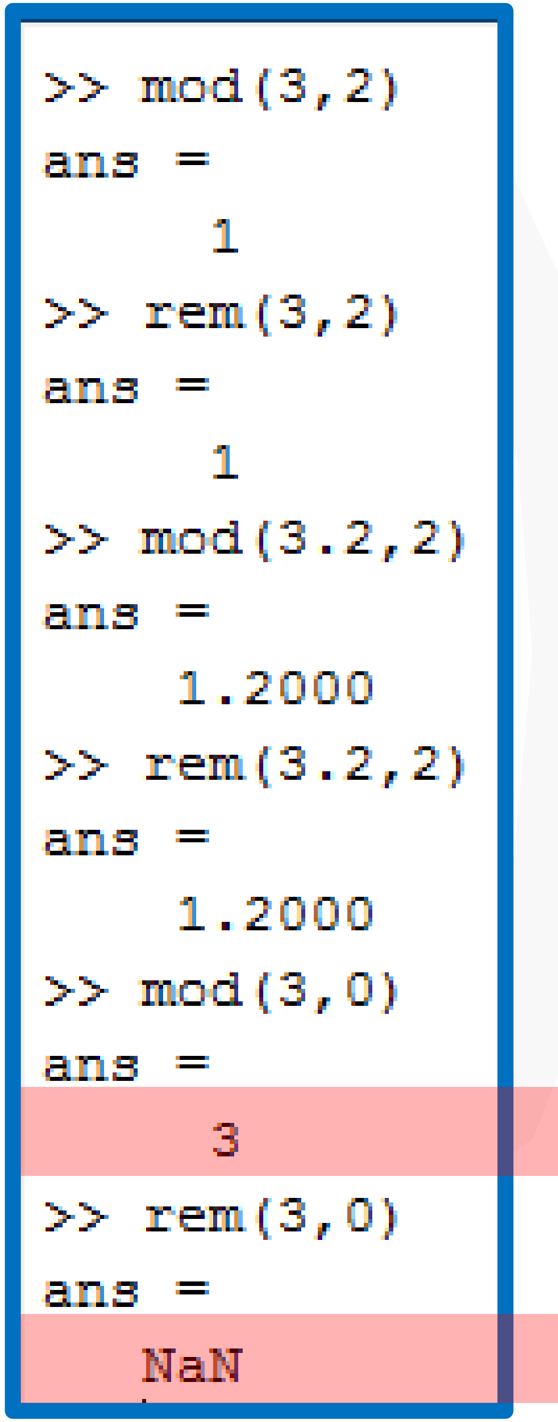
// → ?

(Dr. Z: Remember? We use them in the Sudoku Solver.)

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
>> mod(3,2)
ans =
     1
>> rem(3,2)
ans =
     1
>> mod(3.2,2)
ans =
    1.2000
>> rem(3.2,2)
ans =
    1.2000
>> mod(3,0)
ans =
     3
>> rem(3,0)
ans =
   NaN
```

Remainder

058

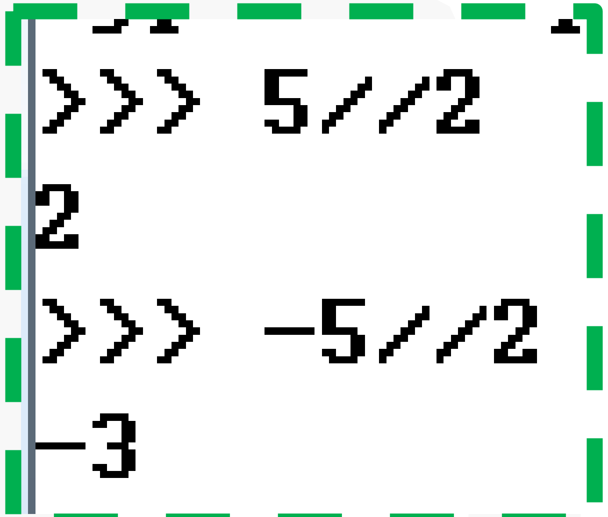
mod vs. rem

Do Not Test

QF666 Programming and Computational Finance

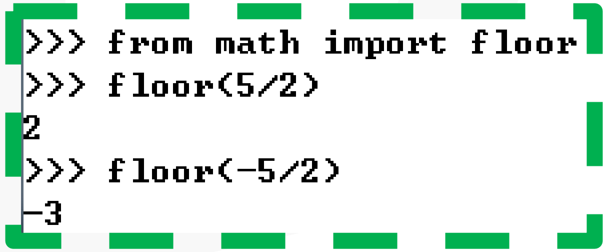
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Integer Division



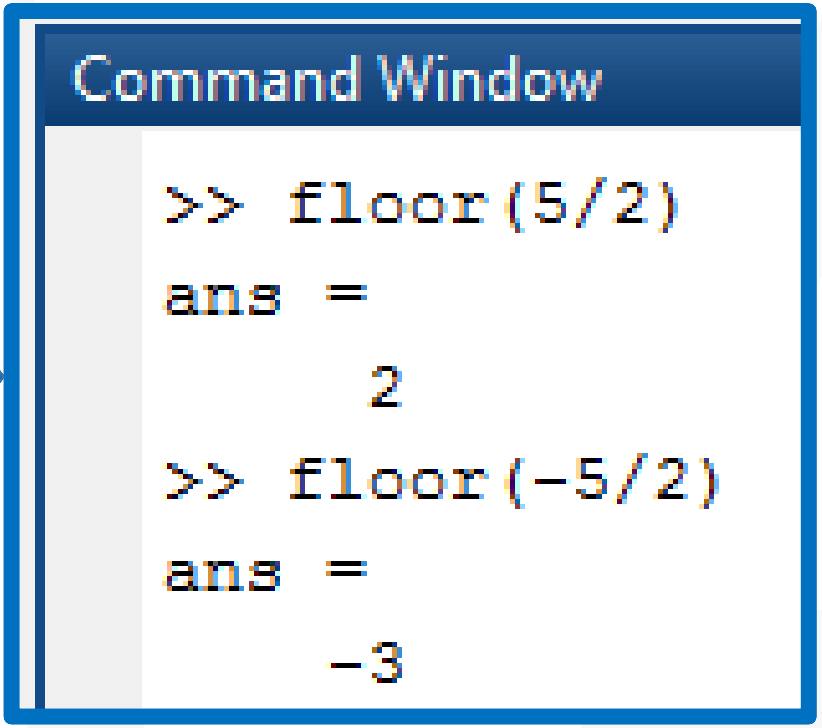
```
>>> 5//2
2
>>> -5//2
-3
```

⇔ floor



```
>>> from math import floor
>>> floor(5/2)
2
>>> floor(-5/2)
-3
```

(Dr. Z: By default, MATLAB treat every number as a floating point number with double precision.)



```
>> floor(5/2)
ans =
     2
>> floor(-5/2)
ans =
    -3
```

(Dr. Z: If the float numbers give us trouble in the Sudoku Solver, we'll learn how to use integers.)

059

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Main Components in a Programming Language

Python	Python
☑ Literals (int, float, complex, str, etc.)	□ Boolean Operations
☑ Arithmetic Operators	□ Built-in Functions: enumerate, min, max
☑ Expressions and Operator Precedence	□ lambda expression (anonymous function)
☑ Variables/Identifiers	□ Built-in Functions: filter, map, next, zip
☑ Scripts	□ Truth Value Testing (and Built-in Function bool)
□ Line Joining	□ Set/Dictionary Operations and Methods
☑ Assignment Statements	□ Augmented Assignment
□ Comments	□ Comparison Operators
□ Indentation	□ Control Flow (if Statements)
□ User defined functions	□ Control Flow (for Statements, while Statements)
☑ String Concatenation (operation)	□ break and continue statements and else Clause
☑ String Indexing and Slicing (operation)	□ Implementation of Big Sigma
☑ String Methods	□ List/Set/Dictionary Comprehensions
□ Lists Operations and Methods (deep copy)	□ More on Assignments and Functions
□ Tuple, Range, Set and Dictionary	□ Classes
□ Membership Test	□ Error Handling

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

MATLAB (Explicit) Line Joining

060

061

(Dr. Z: There’s no implicit line joining.)

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

MATLAB Comments

%

062

(Dr. Z: In a string, it is the start of a format specifier, e.g. `%d` and `%f`.)

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

MATLAB does not require “Indentation”. Code blocks have “end”.

063

(Dr. Z: Use “indentation” to help us to “view” the blocks. The “Editor” has a “Smart Indent” feature.)

SMU
SINGAPORE MANAGEMENT
UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

User-Defined Functions

M-File Functions

https://www.mathworks.com/help/matlab/matlab_prog/create-functions-in-files.html
https://www.mathworks.com/help/matlab/matlab_prog/local-functions.html

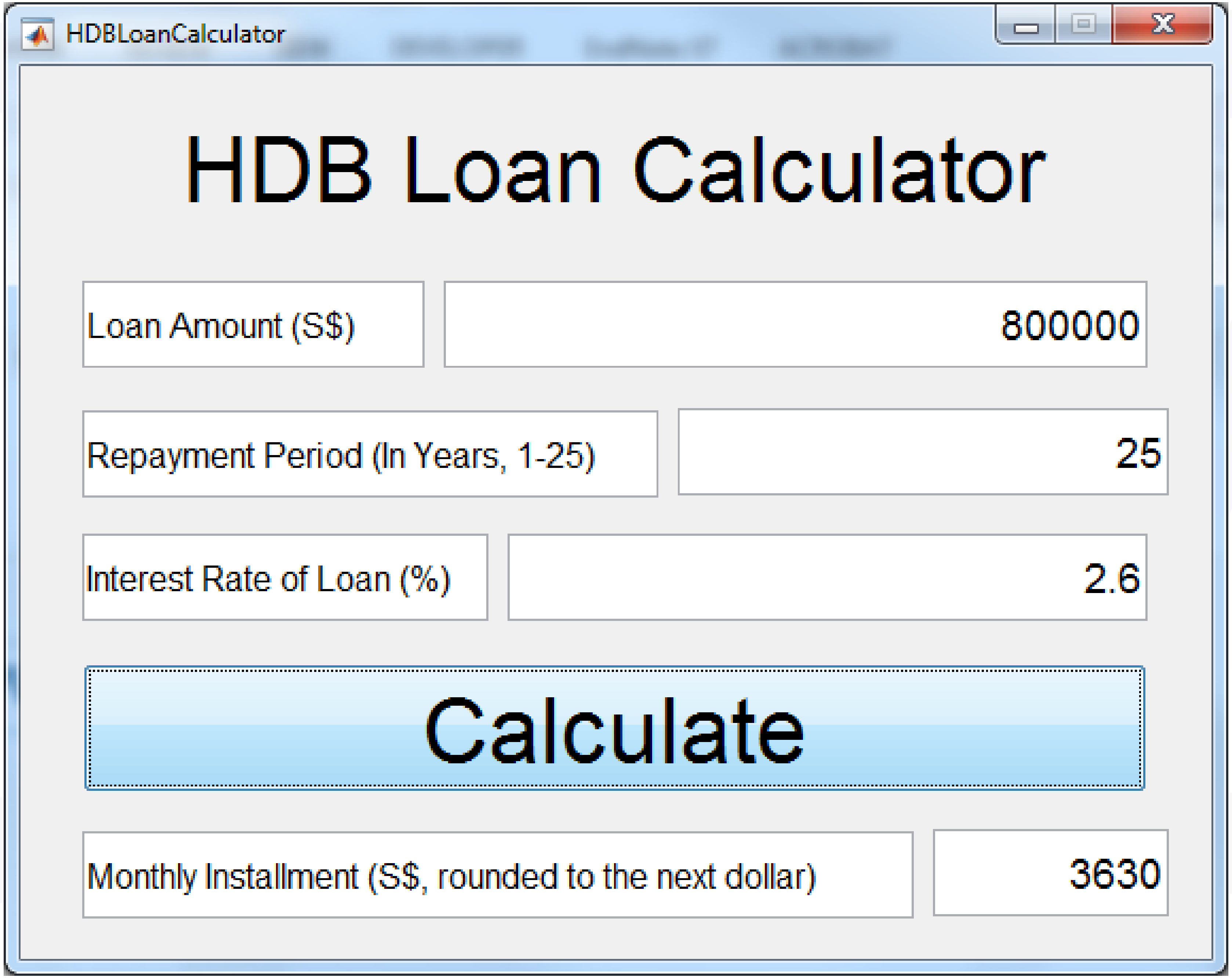
- In MATLAB, we need save the function definition in an M-file (i.e. a .m file), better to use the function name as the file name. (Python will laugh at it. Hahaha…) 064

- Only the first function (the main function) defined in the file is visible to others. Additional functions defined within the same file are only for internal use which are called local functions (or subfunctions). 065

```
P=(r/12*PV)/(1-(1+r/12)^(-12*t));
```

```
function [P]=funP(PV, r, t)
    P=(r/12*PV)/(1-(1+r/12)^(-12*t));
end
```

We need to save the above code as **funP.m** (as "suggested") in a folder that MATLAB can find, say in the current directory.



HDB Loan Calculator

Loan Amount (S$)	800000
Repayment Period (In Years, 1-25)	25
Interest Rate of Loan (%)	2.6
Monthly Installment (S$, rounded to the next dollar)	3630

SMU
SINGAPORE MANAGEMENT
UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

```
% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

PV=str2num(handles.edit_LoanAmount.String);
t=str2num(handles.edit_RepaymentPeriod.String);
r=str2num(handles.edit_InterestRateOfLoan.String)/100;
P=funP(PV, r, t);
handles.edit_MonthlyInstallment.String = num2str(ceil(P));
```

Note that, in MATLAB, we do not need to `import` anything.

Homework Q1

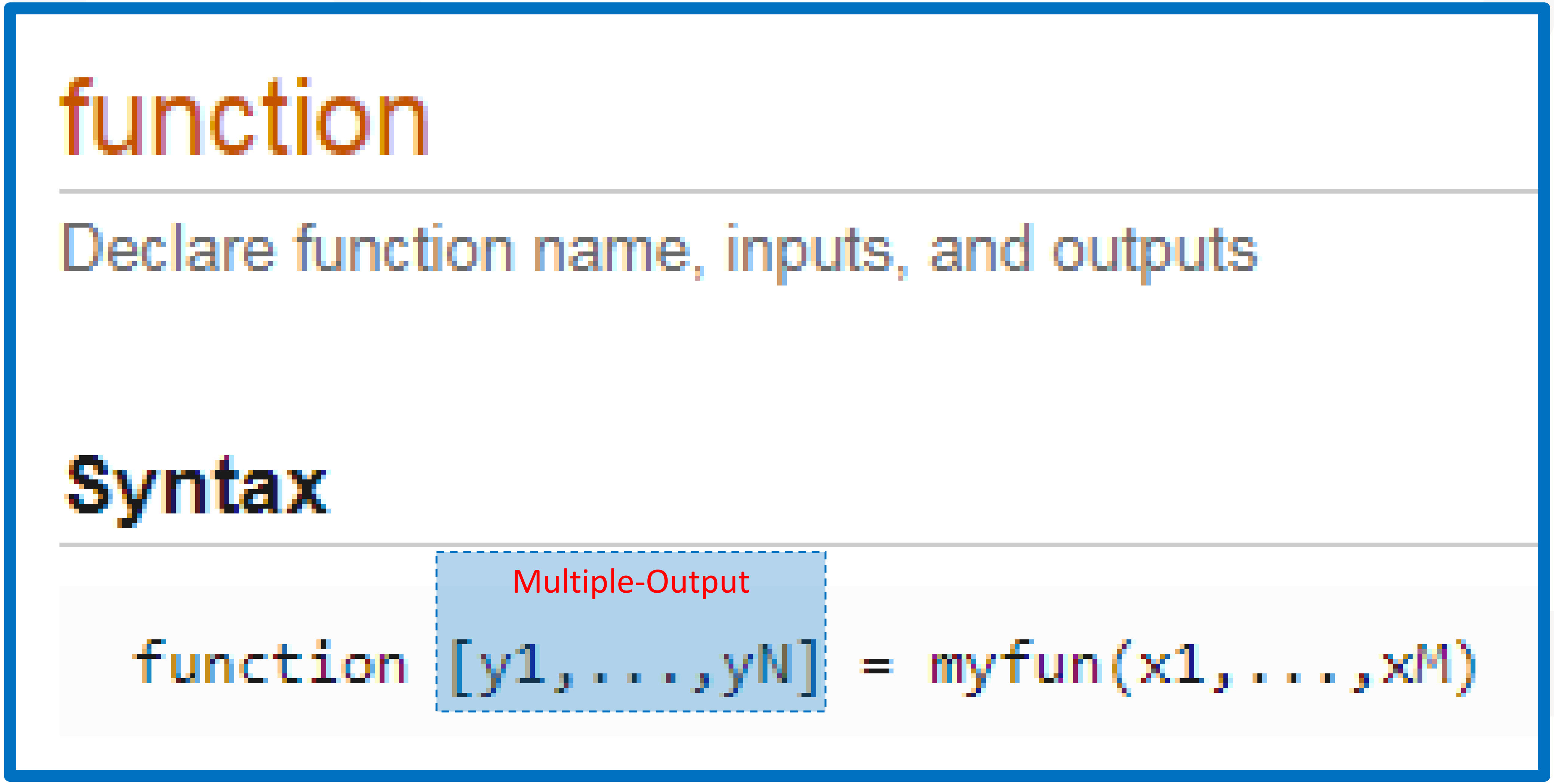
SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

MATLAB functions

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



function

Declare function name, inputs, and outputs

Syntax

Multiple-Output

```
function [y1,...,yN] = myfun(x1,...,xM)
```

```
def funP(PV, r, t):
    P=(r/12*PV)/(1-(1+r/12)**(-12*t))
    return P
```

Python Function

⇩

MATLAB Function

```
function [P]=funP(PV, r, t)
    P=(r/12*PV)/(1-(1+r/12)^(-12*t));
end
```

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

```
from scipy.stats import norm
from math import *
def BS_EuroCallPut(S,K,r,q,sigma,T,t):
   d1=(log(S/K)+(r-q+sigma**2/2.)*(T-t))/(sigma*sqrt(T-t))
   d2=d1-sigma*sqrt(T-t)
   c=S*exp(-q*(T-t))*norm.cdf(d1)-K*exp(-r*(T-t))*norm.cdf(d2)
   p=K*exp(-r*(T-t))*norm.cdf(-d2)-S*exp(-q*(T-t))*norm.cdf(-d1)
   return [c, p]

r=BS_EuroCallPut(50,50,0.04,0.01,0.4,0.5,0)
print(r)
```

In-Class Exercise

(See sample answer on next slide.)

068

Hints:

scipy.stats.norm.cdf ⇔ normcdf

def ⇔ M-File function

retrun [c, p] ⇔ Output [c, p]=

log, sqrt, exp ⇔ log, sqrt, exp

** ⇔ ^

print ⇔ disp

Sample Answer to In-Class Exercise 68

068

BS_EuroCallPut.m

```
function [c,p]=BS_EuroCallPut(S,K,r,q,sigma,T,t)
d1=(log(S/K)+(r-q+sigma^2/2)*(T-t))/(sigma*sqrt(T-t));
d2=d1-sigma*sqrt(T-t);
c=S*exp(-q*(T-t))*normcdf(d1)-K*exp(-r*(T-t))*normcdf(d2);
p=K*exp(-r*(T-t))*normcdf(-d2)-S*exp(-q*(T-t))*normcdf(-d1);
end
```

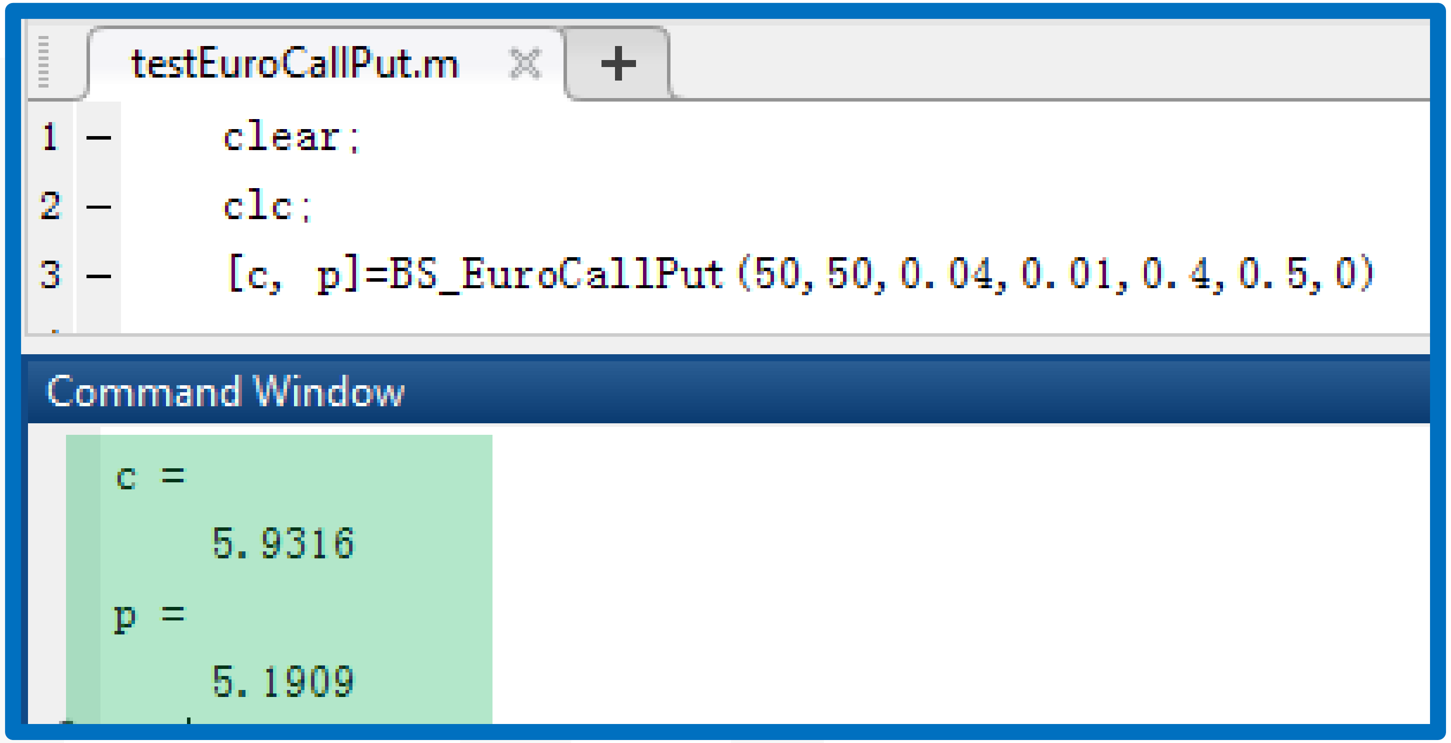
```
clear;
clc;
[c, p]=BS_EuroCallPut(50,50,0.04,0.01,0.4,0.5,0)
```

(Dr. Z: BTW, this is the another "another way" to display the results computed.)

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

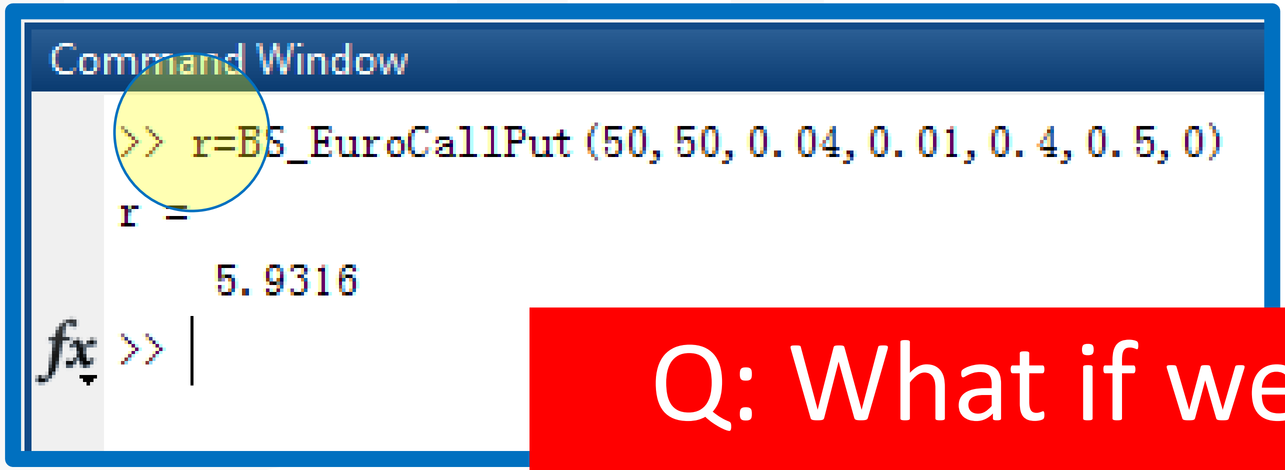


testEuroCallPut.m

```
clear;
clc;
[c, p]=BS_EuroCallPut(50,50,0.04,0.01,0.4,0.5,0)
```

Command Window

```
c =

    5.9316

p =

    5.1909
```



Command Window

```
>> r=BS_EuroCallPut(50,50,0.04,0.01,0.4,0.5,0)

r =

    5.9316

>> 
```

069

Q: What if we only prepare one variable for the output?

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

MATLAB function does not have ~~keyword arguments~~.

070

MATLAB function does not have ~~keyword-only parameters~~.

071

MATLAB function with default arguments' values is not easy.

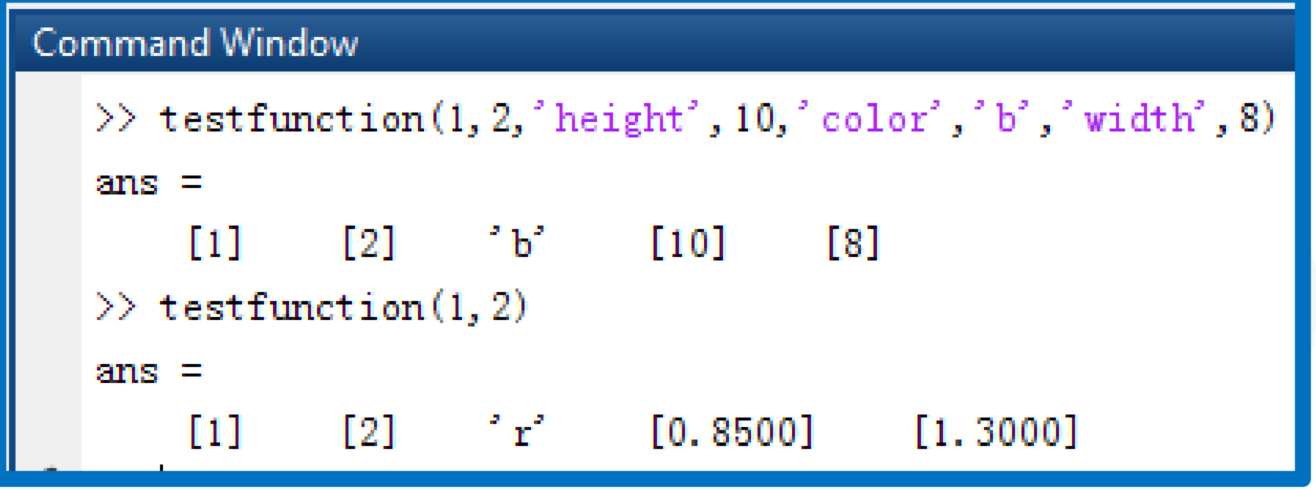
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

In MATLAB, we only consider functions with **required arguments** and **optional arguments** (corresponding to parameters with default values).

```
function r=testfunction(a, b, varargin)
names={'color', 'height', 'width'};
v={'r', 0.85, 1.3};
n=length(varargin)/2;
for i=1:n
   k=find(strcmp(names, varargin{2*i-1}));
   v{k}=varargin{2*i};
end
r=[a, b, v];
```

⇦ Cell Arrays

⇦ for loops



Command Window

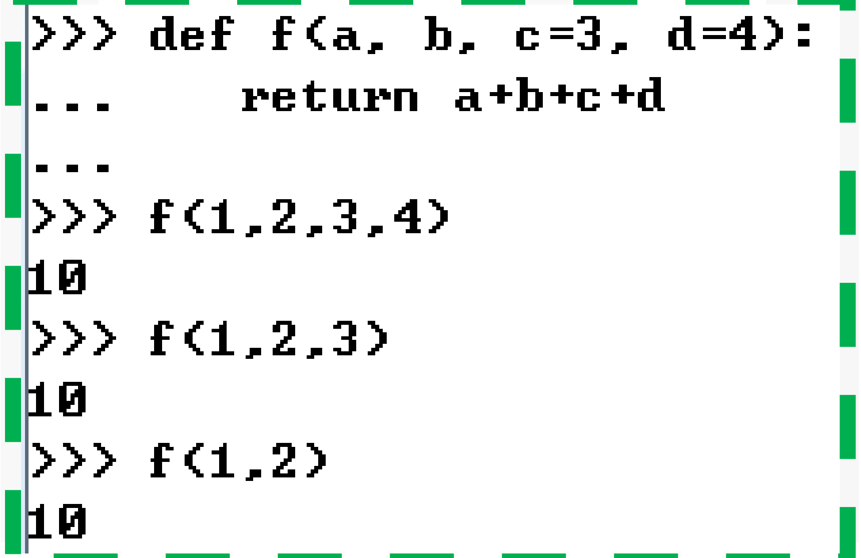
```
>> testfunction(1,2,'height',10,'color','b','width',8)

ans =

    [1]    [2]    'b'    [10]    [8]

>> testfunction(1,2)

ans =

    [1]    [2]    'r'    [0.8500]    [1.3000]
```

Do Not Test

⇨ slide 104

A simple case (function parameters with default values):

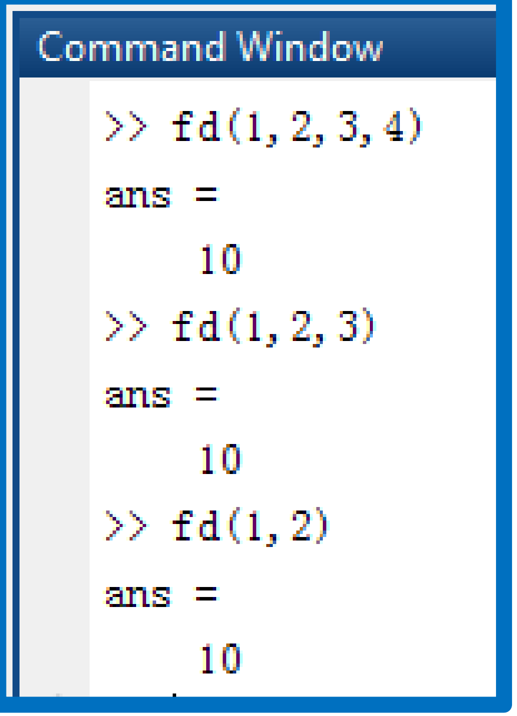
```
def f(a, b, c=3, d=4):
    return a+b+c+d

print(f(1,2,3,4))
print(f(1,2,3))
print(f(1,2))
```



```
>>> def f(a, b, c=3, d=4):
...     return a+b+c+d
...
>>> f(1,2,3,4)
10
>>> f(1,2,3)
10
>>> f(1,2)
10
```

MATLAB nargin

```
function r=fd(a, b, c, d)
   if nargin==3
      d=4;
   elseif nargin==2
      c=3;
      d=4;
   end
   r=a+b+c+d;
```



```
>> fd(1,2,3,4)
ans =
    10
>> fd(1,2,3)
ans =
    10
>> fd(1,2)
ans =
    10
```

Do Not Test

Anonymous Functions

https://www.mathworks.com/help/matlab/matlab_prog/anonymous-functions.html

```
f=lambda x,y: x+y
```

⇨ function object

072

```
f=@(x,y) x+y;
f(1,2)
```

anonymous function

⇨ function handle

⇨ call the function with x=1, y=2

```
f={@(x) x+1, ...
   @(x,y) x+y};
```

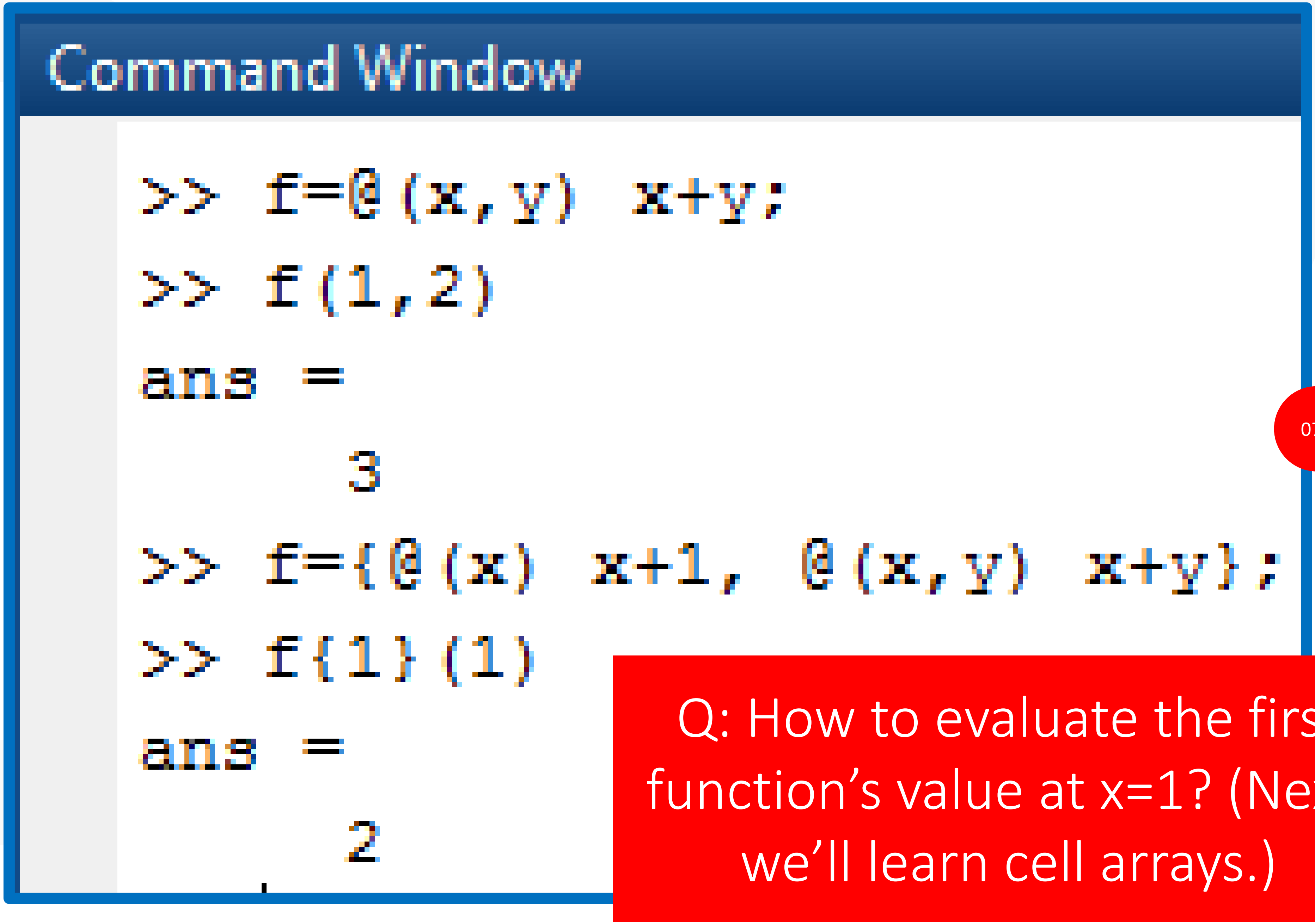
⇨ function handle cell array

Q: How to call the first function with x=1?

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



Command Window

```
>> f=@(x,y) x+y;
>> f(1,2)

ans =

     3

>> f={@(x) x+1, @(x,y) x+y};
>> f{1}(1)

ans =

     2
```

073

Q: How to evaluate the first function's value at x=1? (Next, we'll learn cell arrays.)

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

String

List

Tuple

Range

Set

Dictionary

Class

☑ Character Vector: 'ab'

Array: [1,2]

Cell Array: {1,'a'}

Structure: x.a

Class: x.a

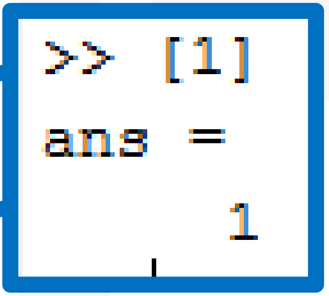
a sequence of character vectors ⇨ concatenated into one

☑ `x=['ab', 'cde']`

074

a sequence of numbers

☑ `x=[1, 2]`



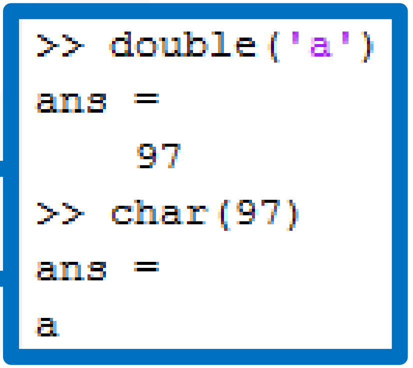
```
>> [1]
ans =
     1
```

(Dr. Z: We'll study 2D arrays and cell arrays in Sessions 03-04.)

075

a sequence of mixed items

☒ `x=[1, 'ab']` ⇔ `x=[char(1), {'ab'}]`



```
>> double('a')
ans =
    97
>> char(97)
ans =
a
```

Numbers are automatically converted to characters when concatenated with character vectors.

076

a sequence of mixed items

☑ `x={1, 'ab'}` ⇔ `x=[{1}, {'ab'}]`

077

a cell array ⇔ an array of cells

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

MATLAB does not have nested arrays.

078

`[[1,2],[3,4]]` ⇨ `[1,2,3,4]`

MATLAB has nested cell arrays.

079

☑ `x={{1,2},{3,4}}`

Cell Array Indexing: `x(1)` and `x{1}`

Cell Array Slicing: `x(1:2)` and `x{1:2}`

(See examples on subsequent slides.)

```
x={{1,2},{3,4}}
```

⇔ `x=[{{1,2}},{{3,4}}]`

☑ `x(1)` ⇨ `{{1,2}}`

080

the first cell

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

080

x={{1,2},{3,4}}

☑ x{1}⇨{1,2}

content in the first cell

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

080

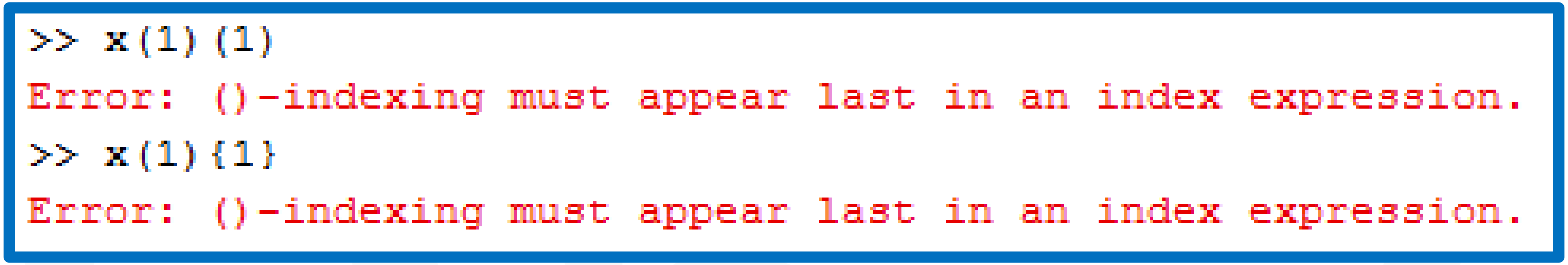
```
x={{1,2},{3,4}}
```

```
(A) x(1)(1)

(B) x(1){1}

(C) x{1}(1)

(D) x{1}{1}

(E) Others:___________
```

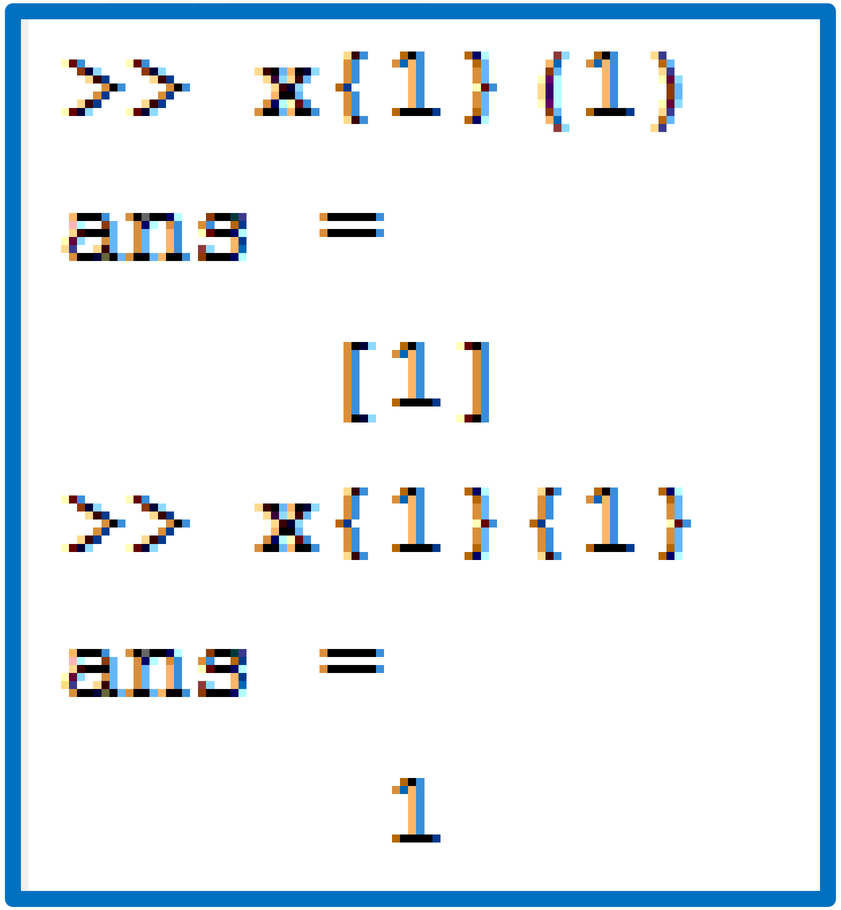
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance



```
>> x(1)(1)
Error: ()-indexing must appear last in an index expression.
>> x(1){1}
Error: ()-indexing must appear last in an index expression.
```



```
>> x{1}(1)
ans =

    [1]
>> x{1}{1}
ans =

     1
```

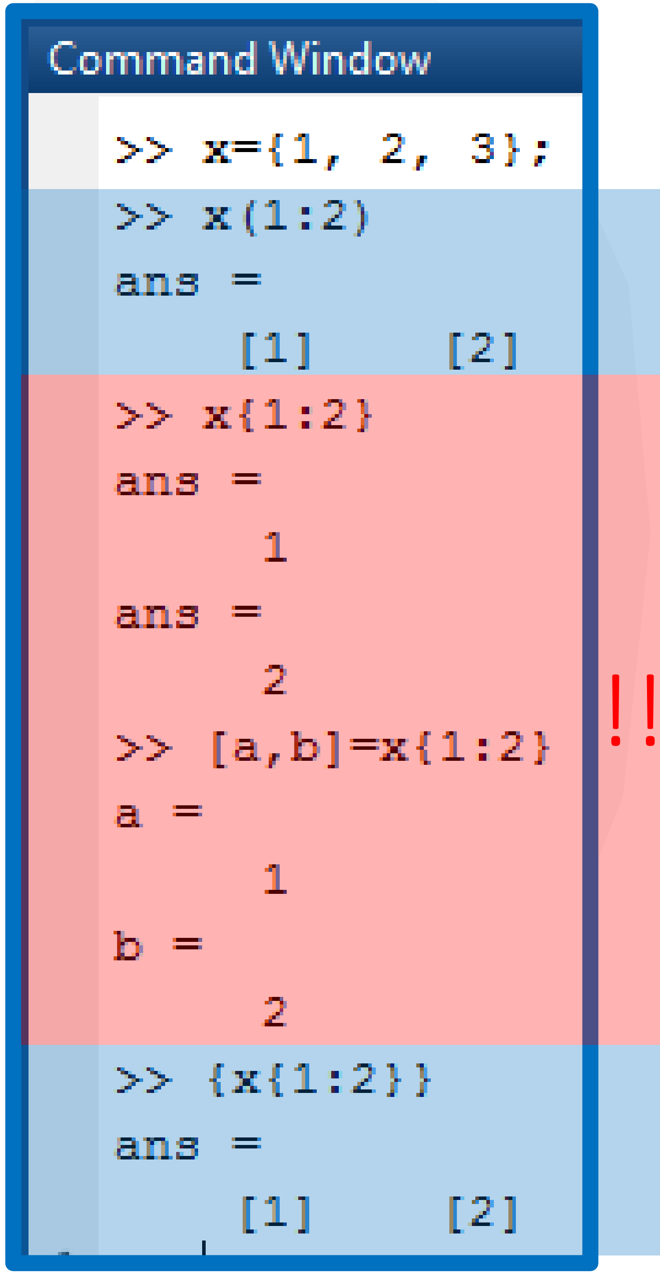
080

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

https://www.mathworks.com/help/matlab/matlab_prog/comma-separated-lists.html



```
>> x={1, 2, 3};
>> x(1:2)
ans =
      [1]      [2]
>> x{1:2}
ans =
       1
ans =
       2
>> [a,b]=x{1:2}
a =
       1
b =
       2
>> {x{1:2}}
ans =
      [1]      [2]
```

!!!!!!!!!

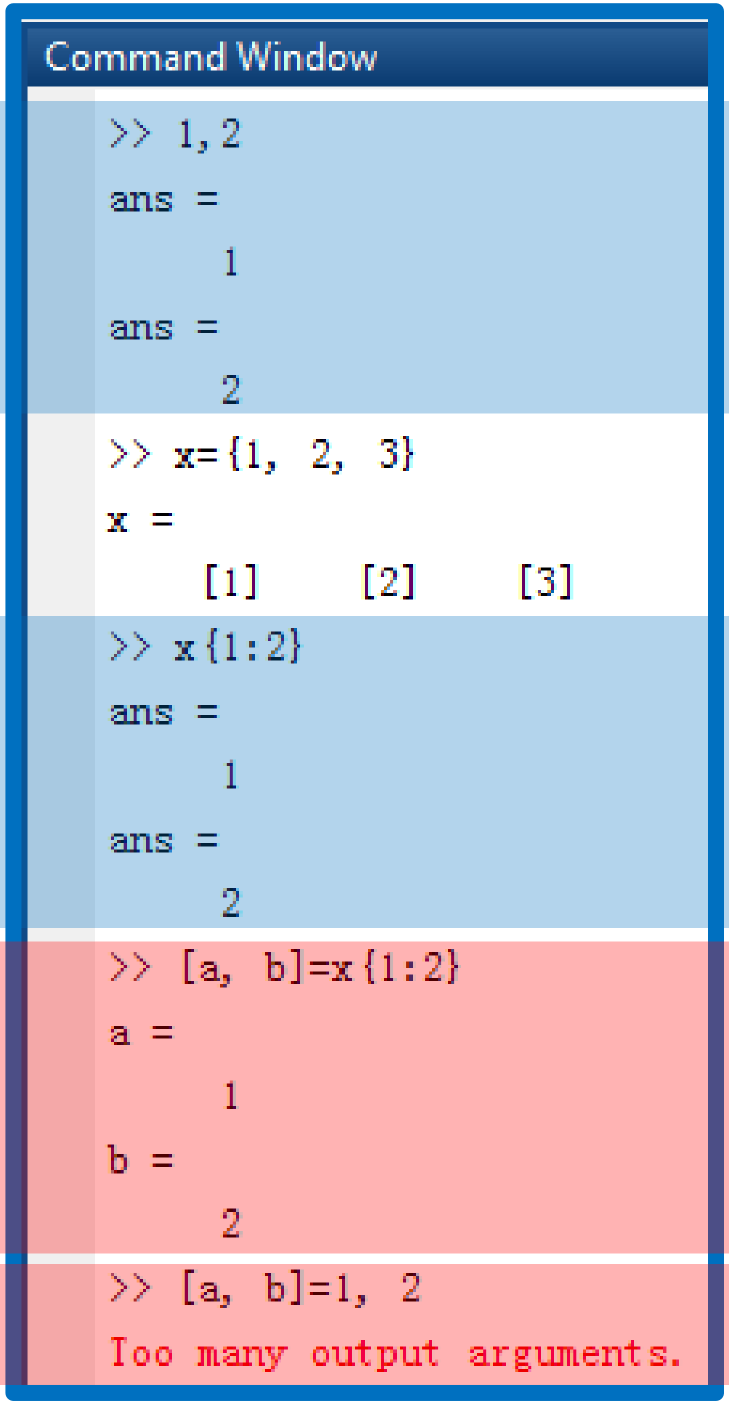
081

For a cell array x, x{1:2} creates a <u>comma-separated list</u>.

Extracting multiple elements from a cell array yields a comma-separated list.

082

For a cell array x,
x(1:2) ⇔ {x{1:2}}



```
>> 1,2

ans =

     1

ans =

     2

>> x={1, 2, 3}

x =

    [1]    [2]    [3]

>> x{1:2}

ans =

     1

ans =

     2

>> [a, b]=x{1:2}

a =

     1

b =

     2

>> [a, b]=1, 2
Too many output arguments.
```

A Comma-Separated List

A Comma-Separated List

Assignment

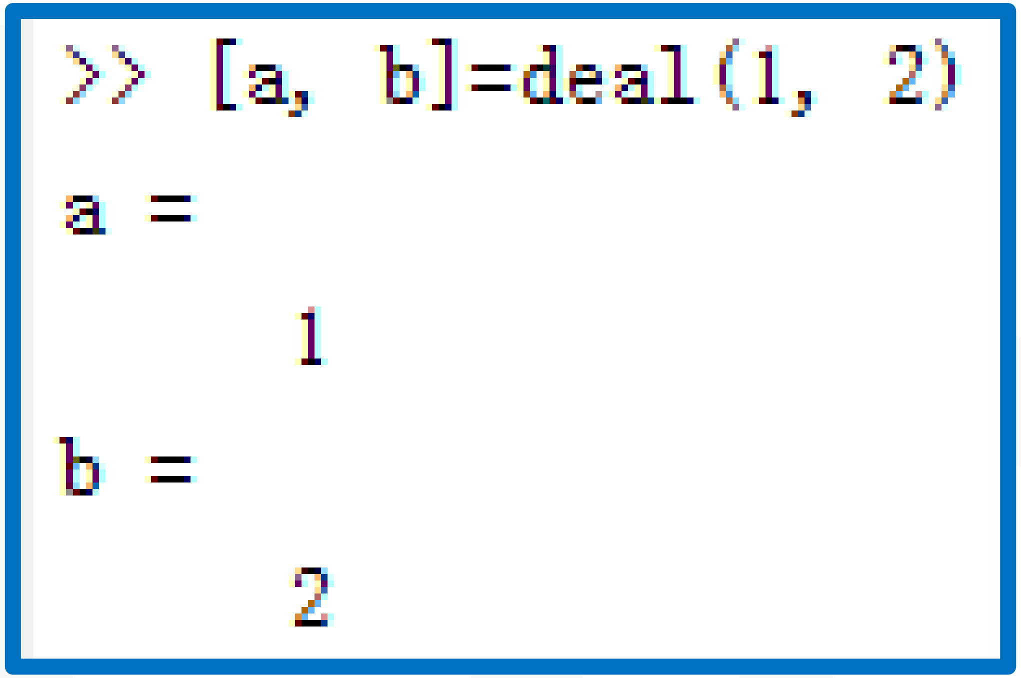
!!!Too many output arguments!!!

SMU
SINGAPORE MANAGEMENT UNIVERSITY

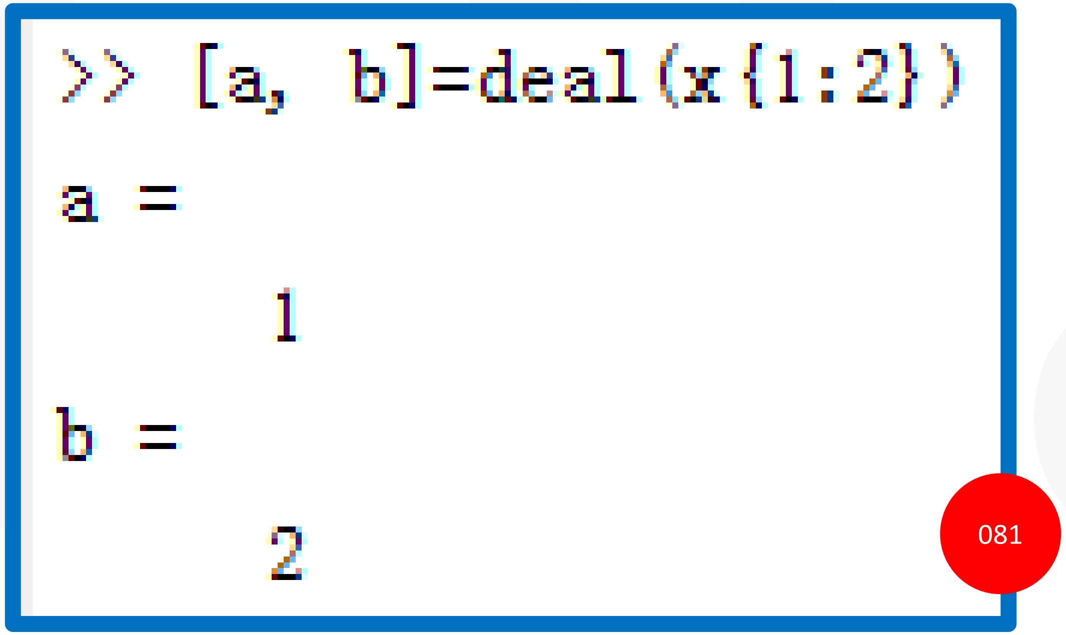
QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

Assignment



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

     1

b =

     2
```

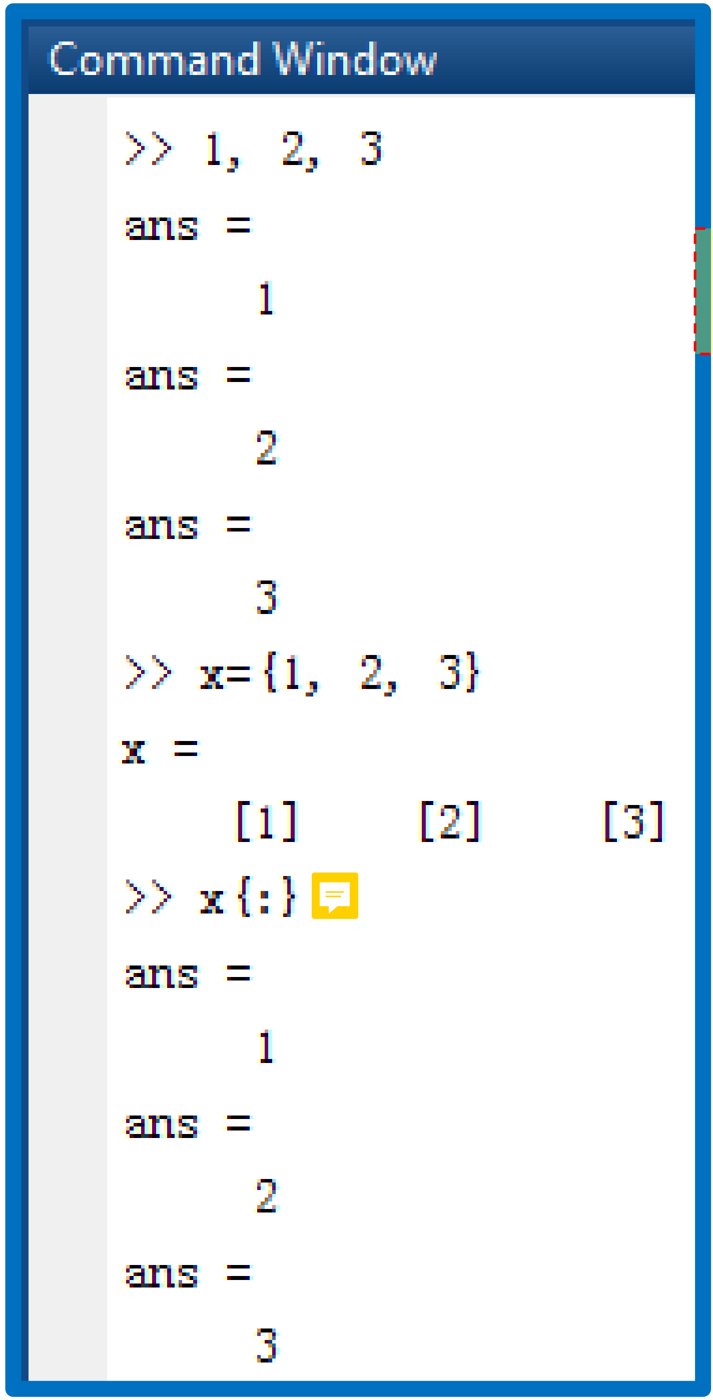


```
>> [a, b]=deal(x{1:2})
a =

     1

b =

     2
```

SMU
SINGAPORE MANAGEMENT UNIVERSITY

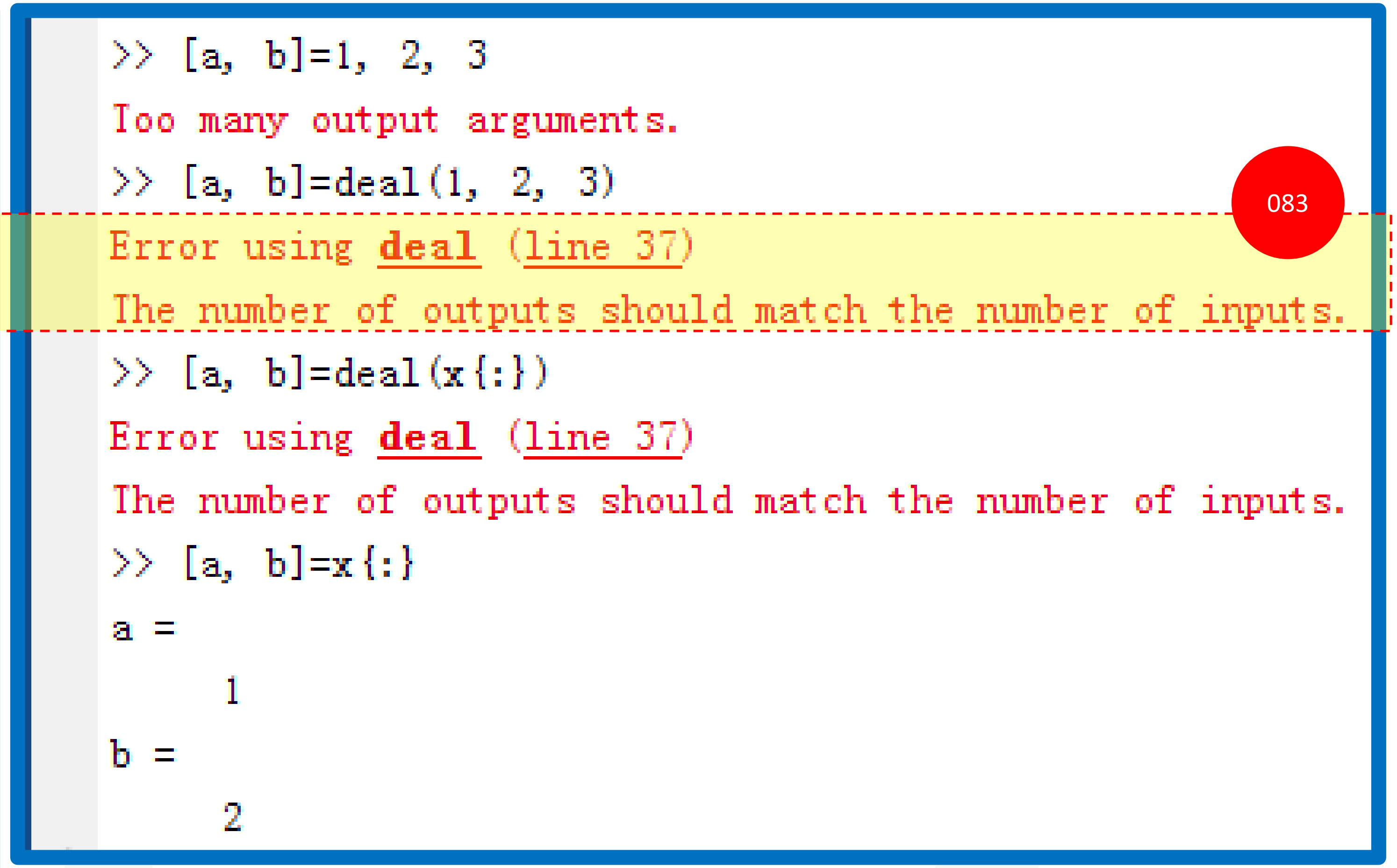
QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance



Command Window

```
>> 1, 2, 3
ans =
     1
ans =
     2
ans =
     3
>> x={1, 2, 3}
x =
    [1]    [2]    [3]
>> x{:}
ans =
     1
ans =
     2
ans =
     3
```



```
>> [a, b]=1, 2, 3
Too many output arguments.
>> [a, b]=deal(1, 2, 3)
Error using deal (line 37)
The number of outputs should match the number of inputs.
>> [a, b]=deal(x{:})
Error using deal (line 37)
The number of outputs should match the number of inputs.
>> [a, b]=x{:}
a =
     1
b =
     2
```

083

Q: What is the output?

Q: "deal" or not?

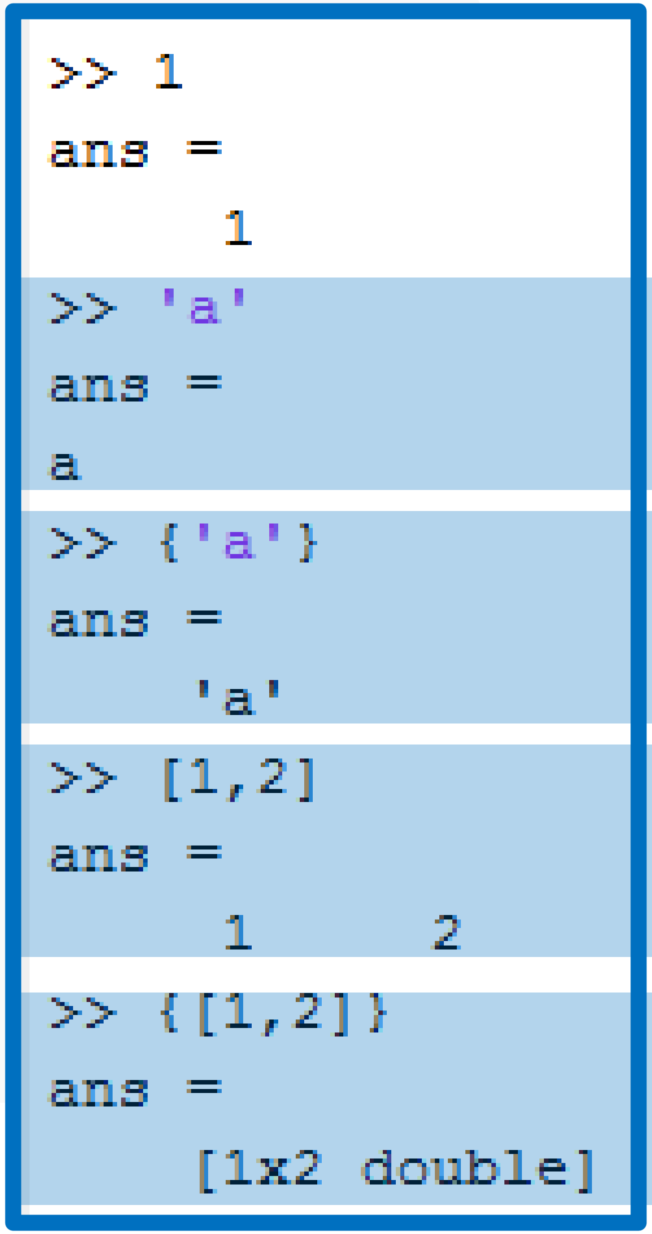
SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Format of the value displayed

084 085 086 087



```
>> 1
ans =
     1
>> 'a'
ans =
a
>> {'a'}
ans =
    'a'
>> [1,2]
ans =
     1     2
>> {[1,2]}
ans =
    [1x2 double]
```

The character vector does not show single-quotation marks.

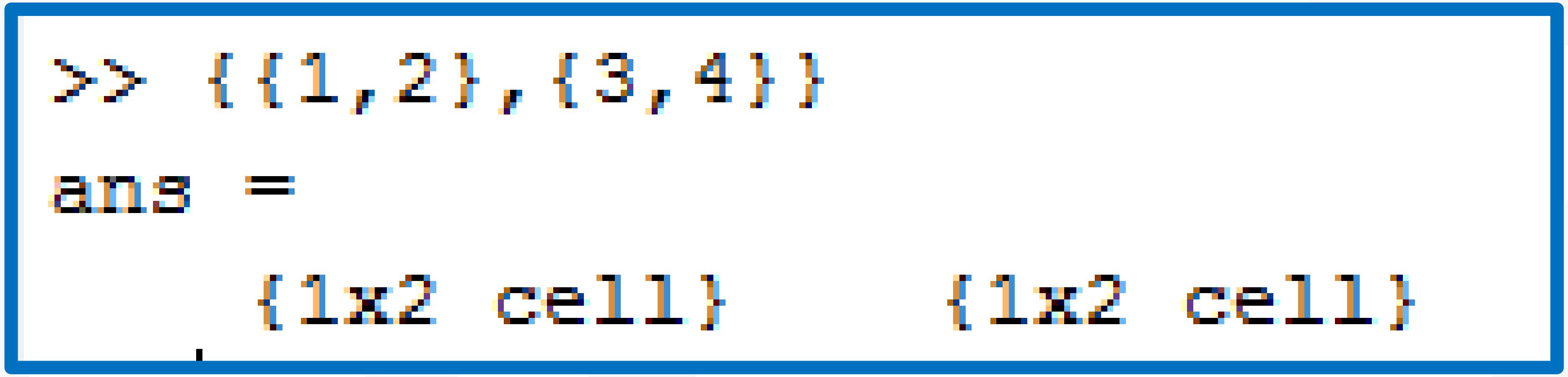
We see single-quotation marks when the character vector is in a cell array.

Array of numbers does not show brackets.

We see brackets when the array of numbers is in a cell array.

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
>> {{1,2},{3,4}}
ans =
    {1x2 cell}    {1x2 cell}
```

What is displayed?

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Slide
100/160

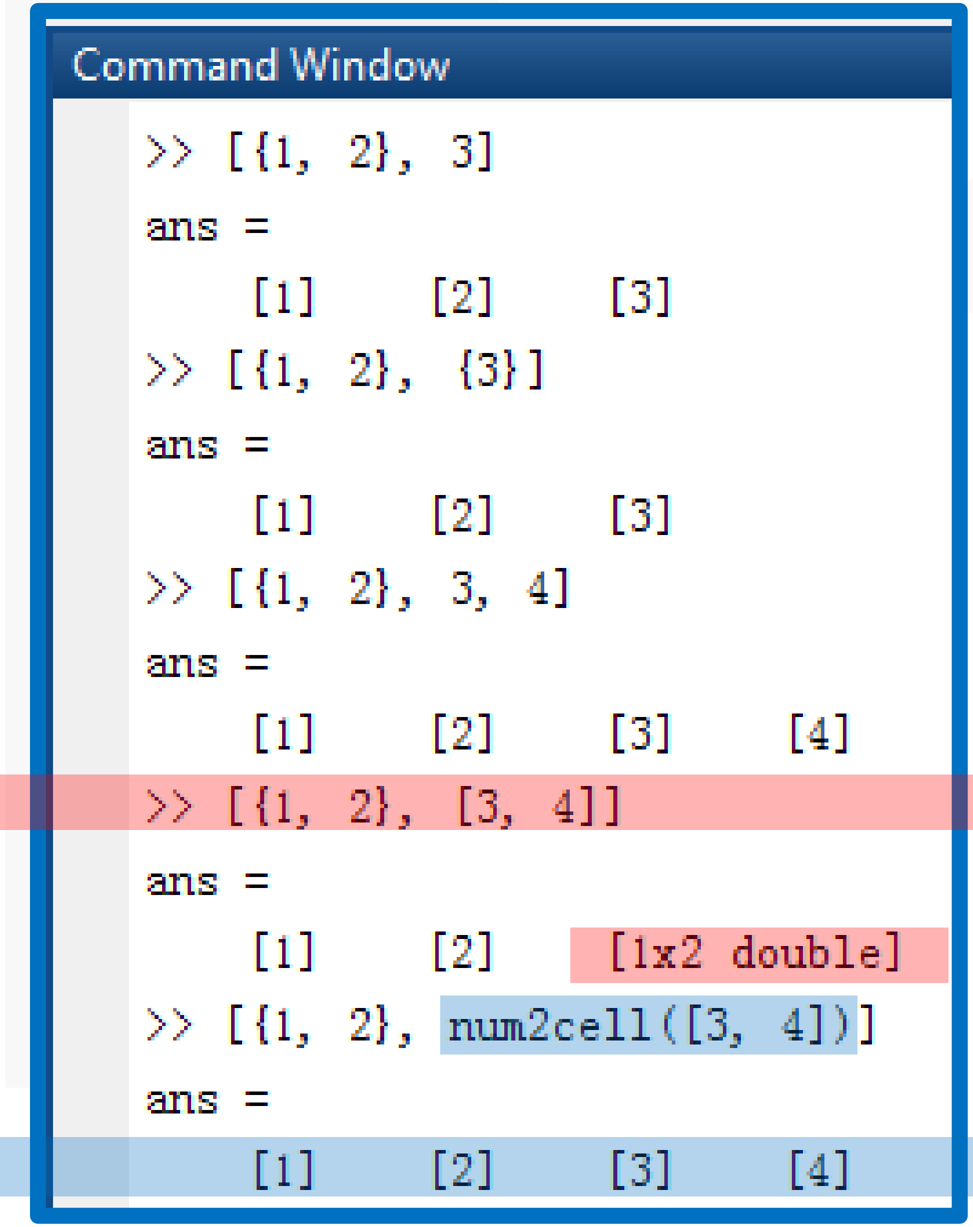
SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance

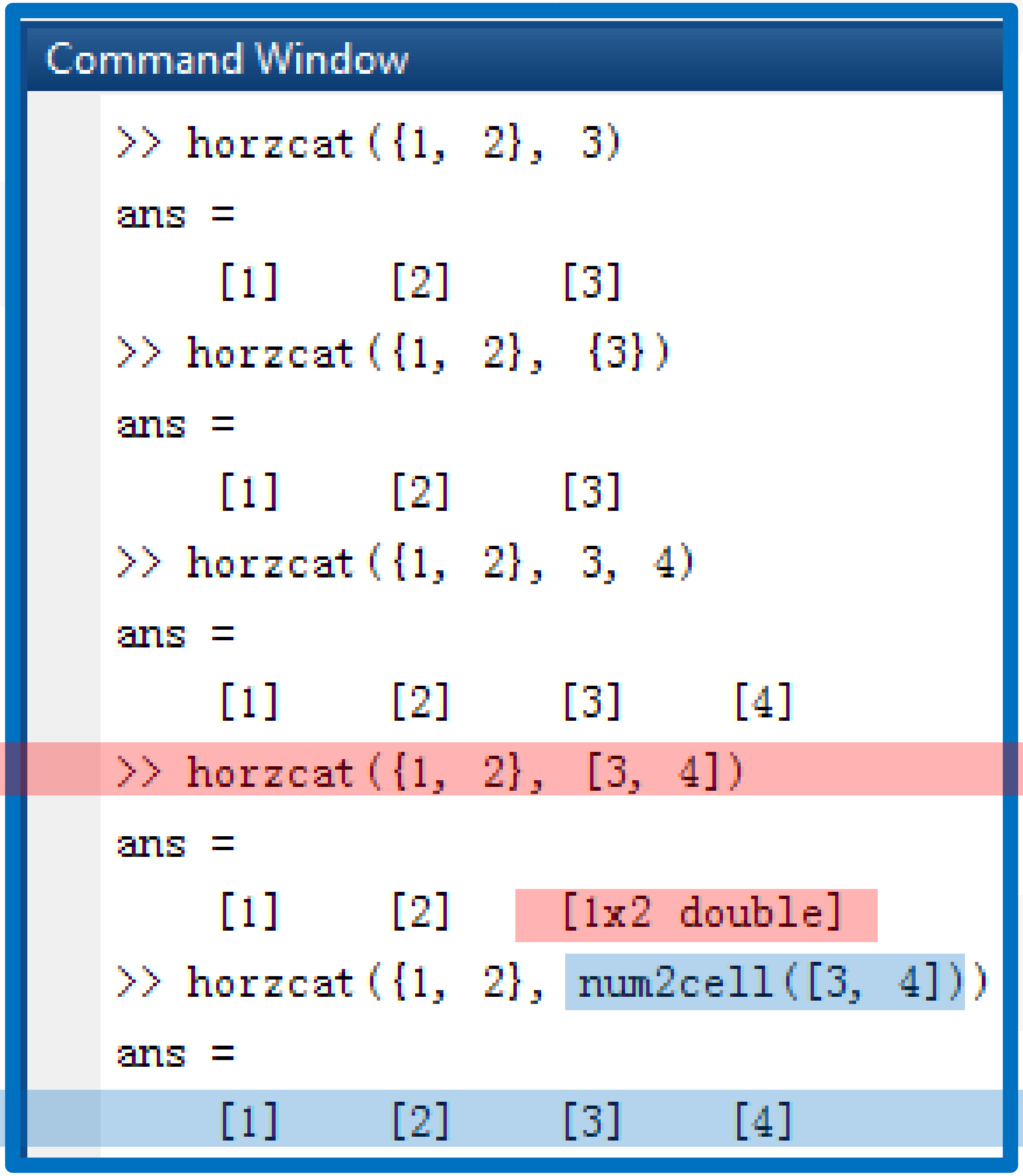
Add items to a cell array: [ ] and horzcat

089



Command Window

```
>> [{1, 2}, 3]
ans =
    [1]    [2]    [3]
>> [{1, 2}, {3}]
ans =
    [1]    [2]    [3]
>> [{1, 2}, 3, 4]
ans =
    [1]    [2]    [3]    [4]
>> [{1, 2}, [3, 4]]
ans =
    [1]    [2]    [1x2 double]
>> [{1, 2}, num2cell([3, 4])]
ans =
    [1]    [2]    [3]    [4]
```



Command Window

```
>> horzcat({1, 2}, 3)
ans =
    [1]    [2]    [3]
>> horzcat({1, 2}, {3})
ans =
    [1]    [2]    [3]
>> horzcat({1, 2}, 3, 4)
ans =
    [1]    [2]    [3]    [4]
>> horzcat({1, 2}, [3, 4])
ans =
    [1]    [2]    [1x2 double]
>> horzcat({1, 2}, num2cell([3, 4]))
ans =
    [1]    [2]    [3]    [4]
```

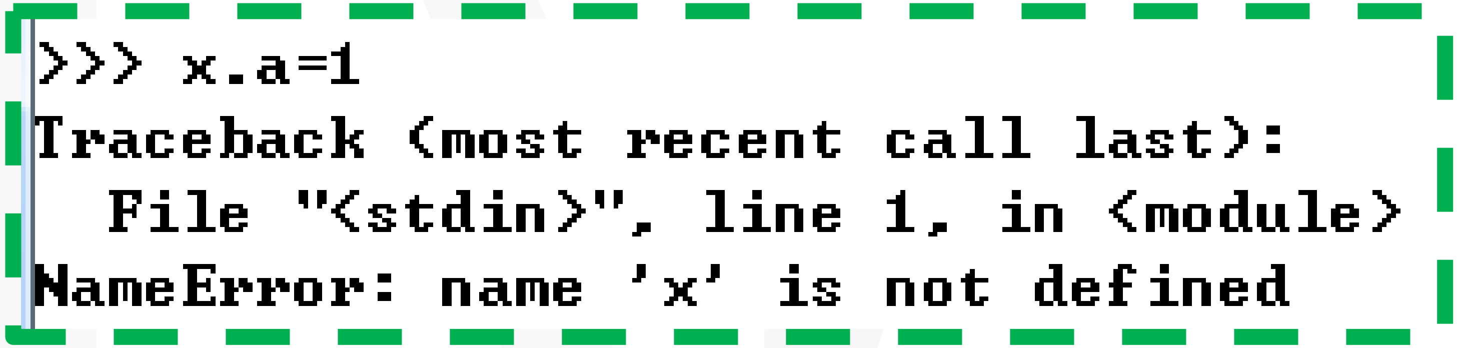
SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

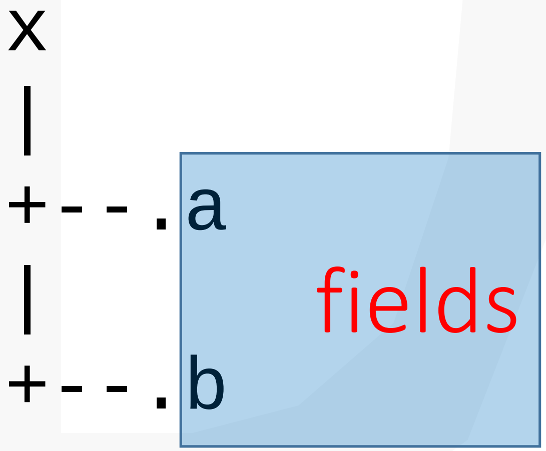
Structures/Structure Arrays (1)

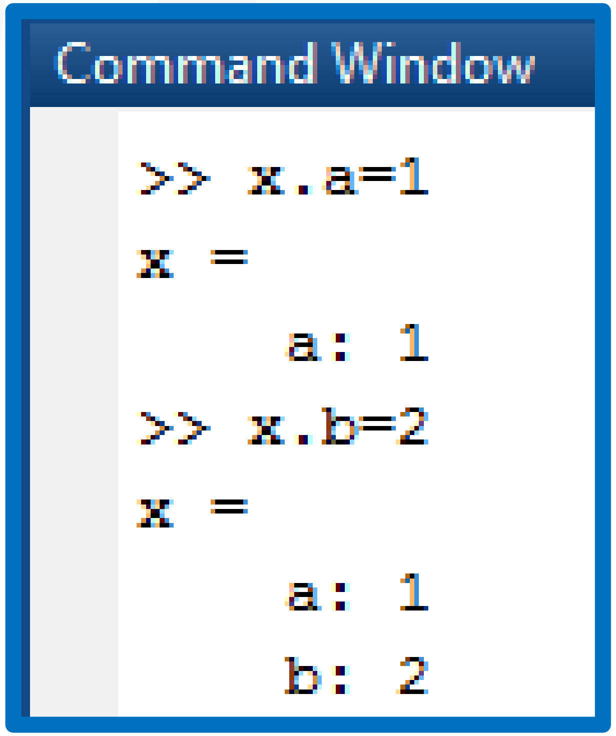
https://www.mathworks.com/help/matlab/matlab_prog/create-a-structure-array.html



```
>>> x.a=1
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'x' is not defined
```

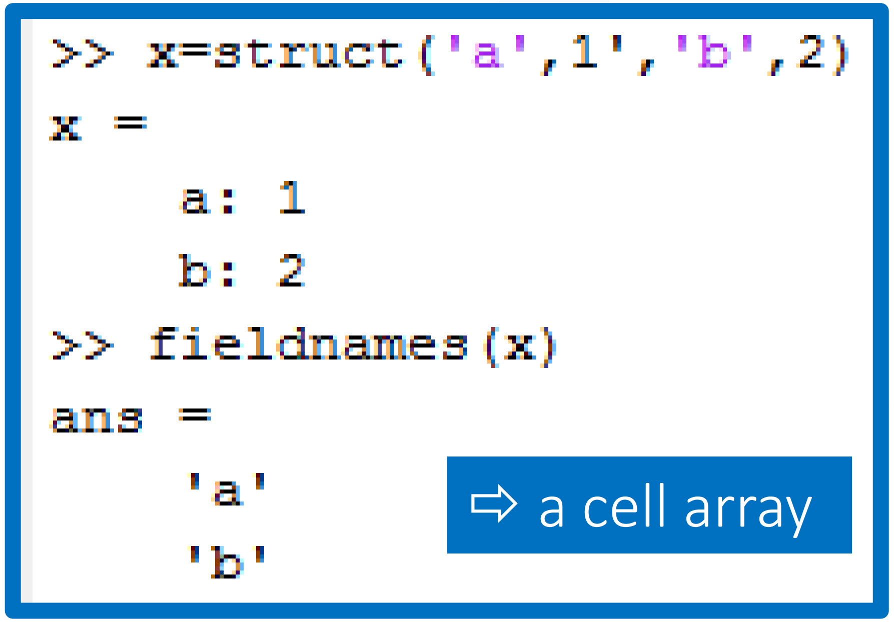
090





```
>> x.a=1
x =
      a: 1
>> x.b=2
x =
      a: 1
      b: 2
```

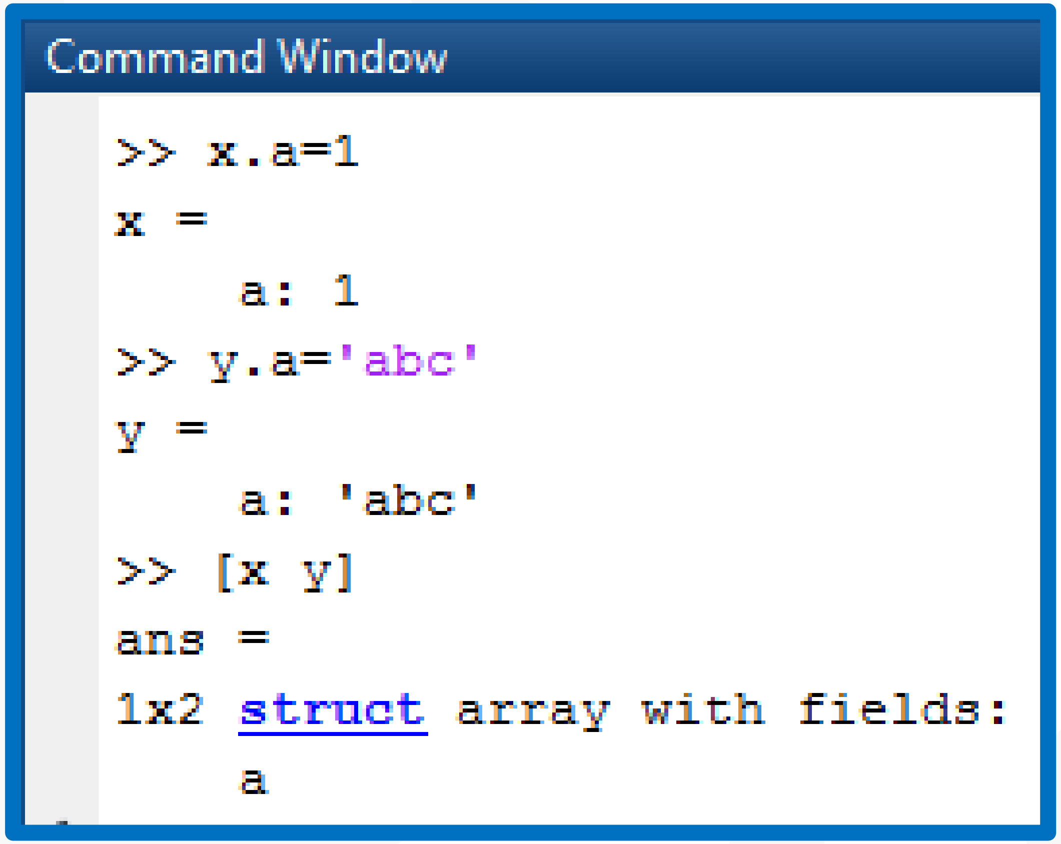
⇔ x=struct('a',1, 'b',2)



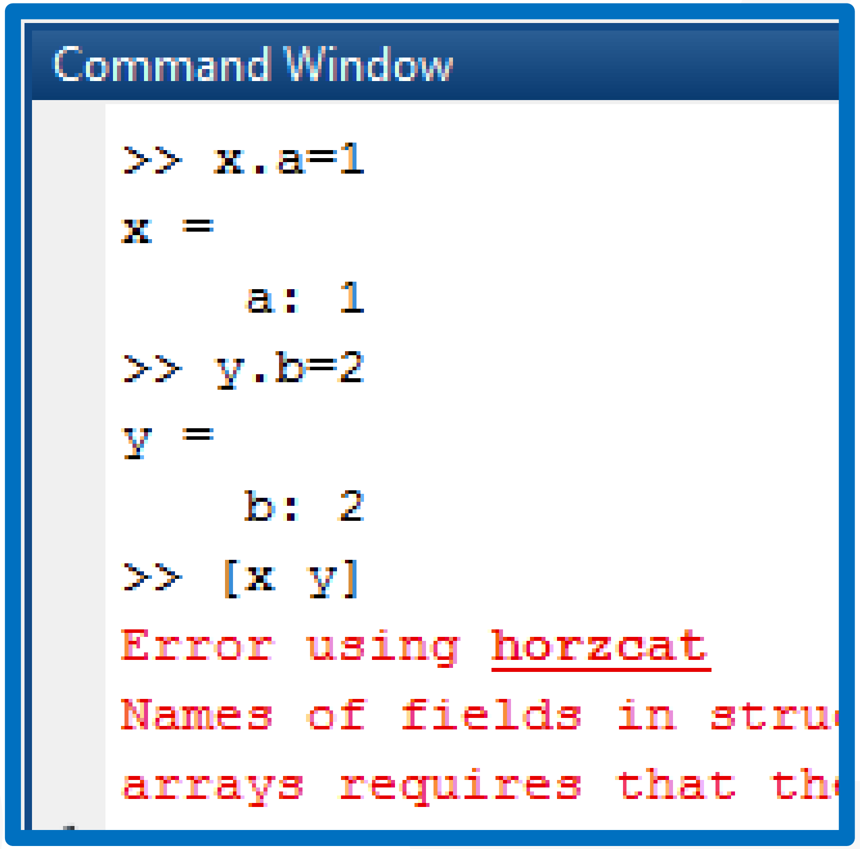
```
>> x=struct('a',1','b',2)
x =
    a: 1
    b: 2
>> fieldnames(x)
ans =
    'a'
    'b'
```

⇨ a cell array

Structures/Structure Arrays (2)



```
>> x.a=1
x =
     a: 1
>> y.a='abc'
y =
     a: 'abc'
>> [x y]
ans =
1x2 struct array with fields:
     a
```

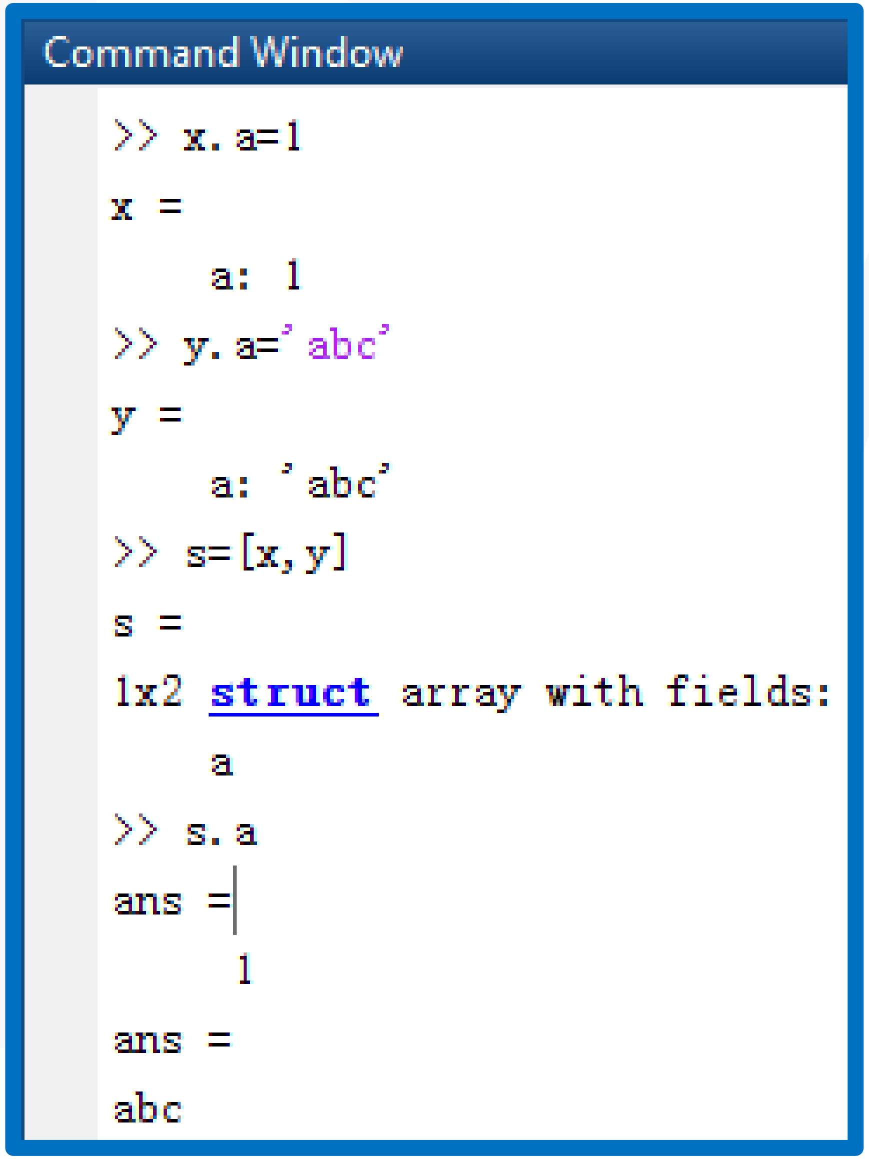


```
>> x.a=1
x =
     a: 1
>> y.b=2
y =
     b: 2
>> [x y]
Error using horzcat
Names of fields in stru
arrays requires that th
```

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Structures/Structure Arrays (3)



```
>> x.a=1
x =
    a: 1
>> y.a='abc'
y =
    a: 'abc'
>> s=[x,y]
s =
1x2 struct array with fields:
    a
>> s.a
ans =
     1
ans =
abc
```

For a structure array, extracting a field of the structure yields a comma-separated list.

`s.a ⇔ s(1).a, s(2).a`

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

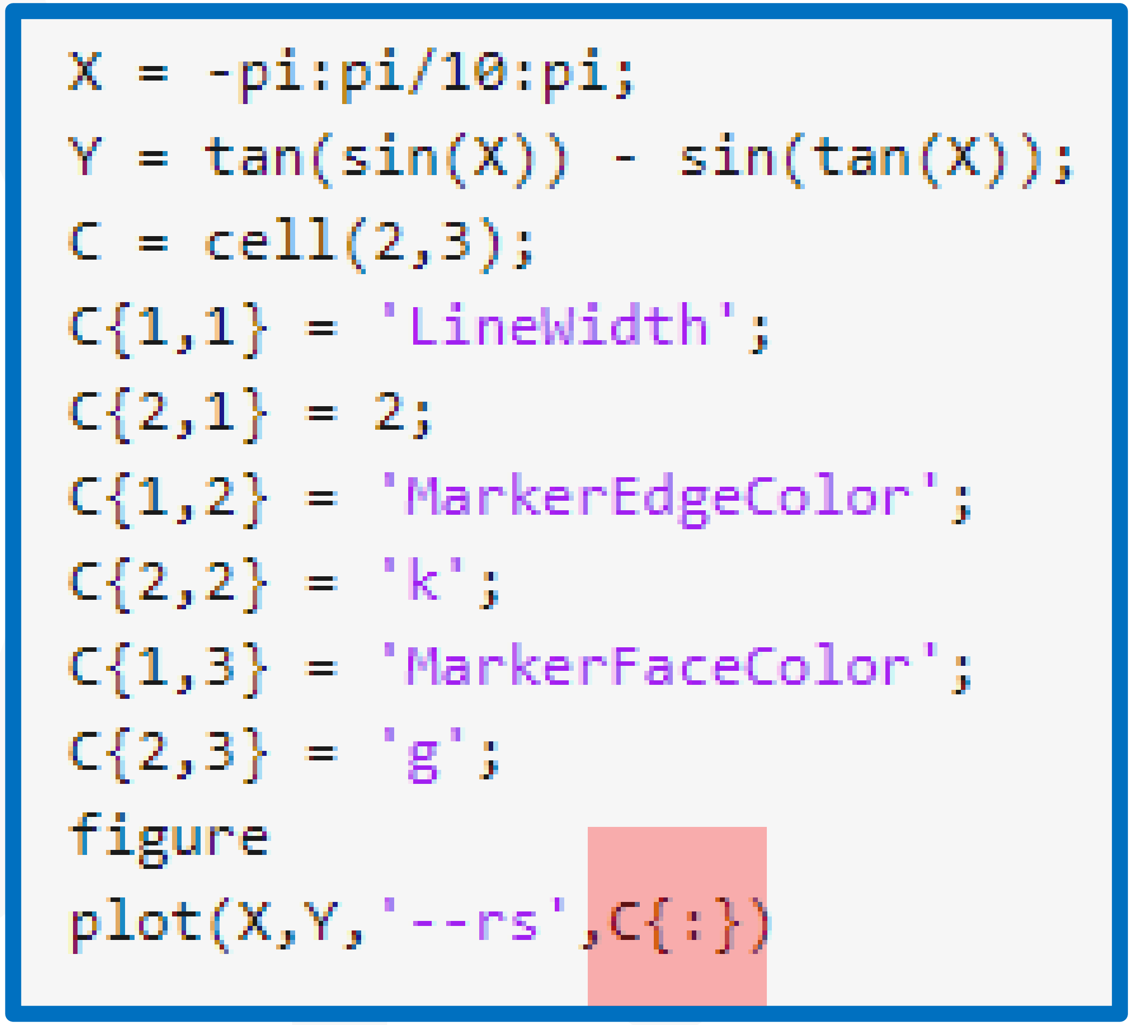
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance

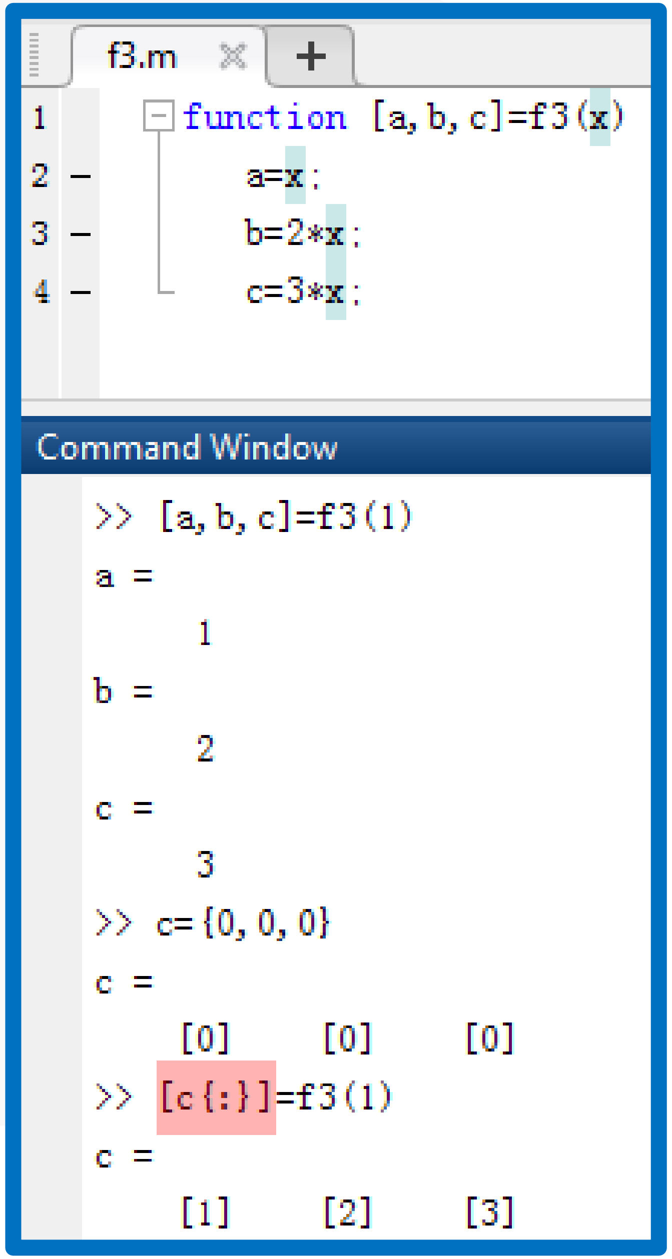
Comma-Separate Lists as Function Call Arguments



```
X = -pi:pi/10:pi;
Y = tan(sin(X)) - sin(tan(X));
C = cell(2,3);
C{1,1} = 'LineWidth';
C{2,1} = 2;
C{1,2} = 'MarkerEdgeColor';
C{2,2} = 'k';
C{1,3} = 'MarkerFaceColor';
C{2,3} = 'g';
figure
plot(X,Y,'--rs',C{:})
```

⇦ slide 83

Comma-Separate Lists as Function Outputs

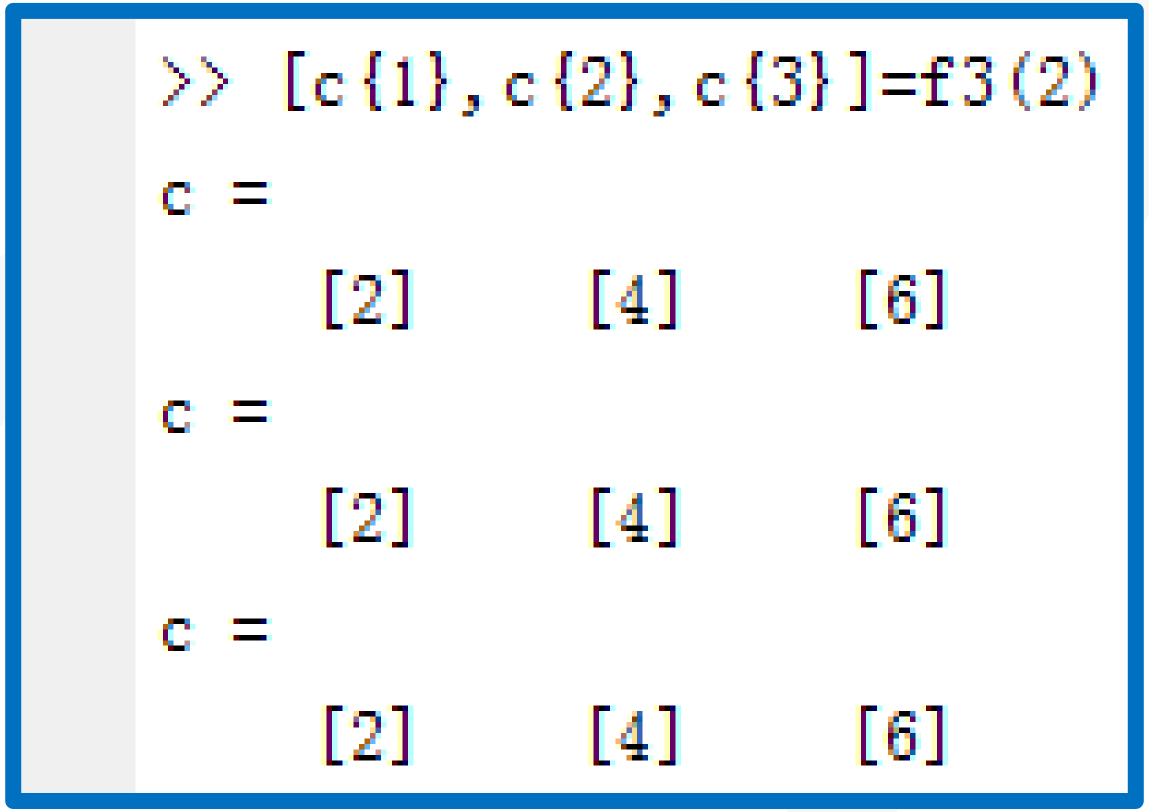


f3.m

```
function [a, b, c]=f3(x)
    a=x;
    b=2*x;
    c=3*x;
```

Command Window

```
>> [a, b, c]=f3(1)
a =
     1
b =
     2
c =
     3
>> c={0, 0, 0}
c =
    [0]    [0]    [0]
>> [c{:}]=f3(1)
c =
    [1]    [2]    [3]
```



```
>> [c{1},c{2},c{3}]=f3(2)
c =
    [2]    [4]    [6]
c =
    [2]    [4]    [6]
c =
    [2]    [4]    [6]
```

(Dr. Z: We notice a small difference in the display.)

(Dr. Z: This should also work on a field of a structure array, right?)

QF666 Programming and Computational Finance

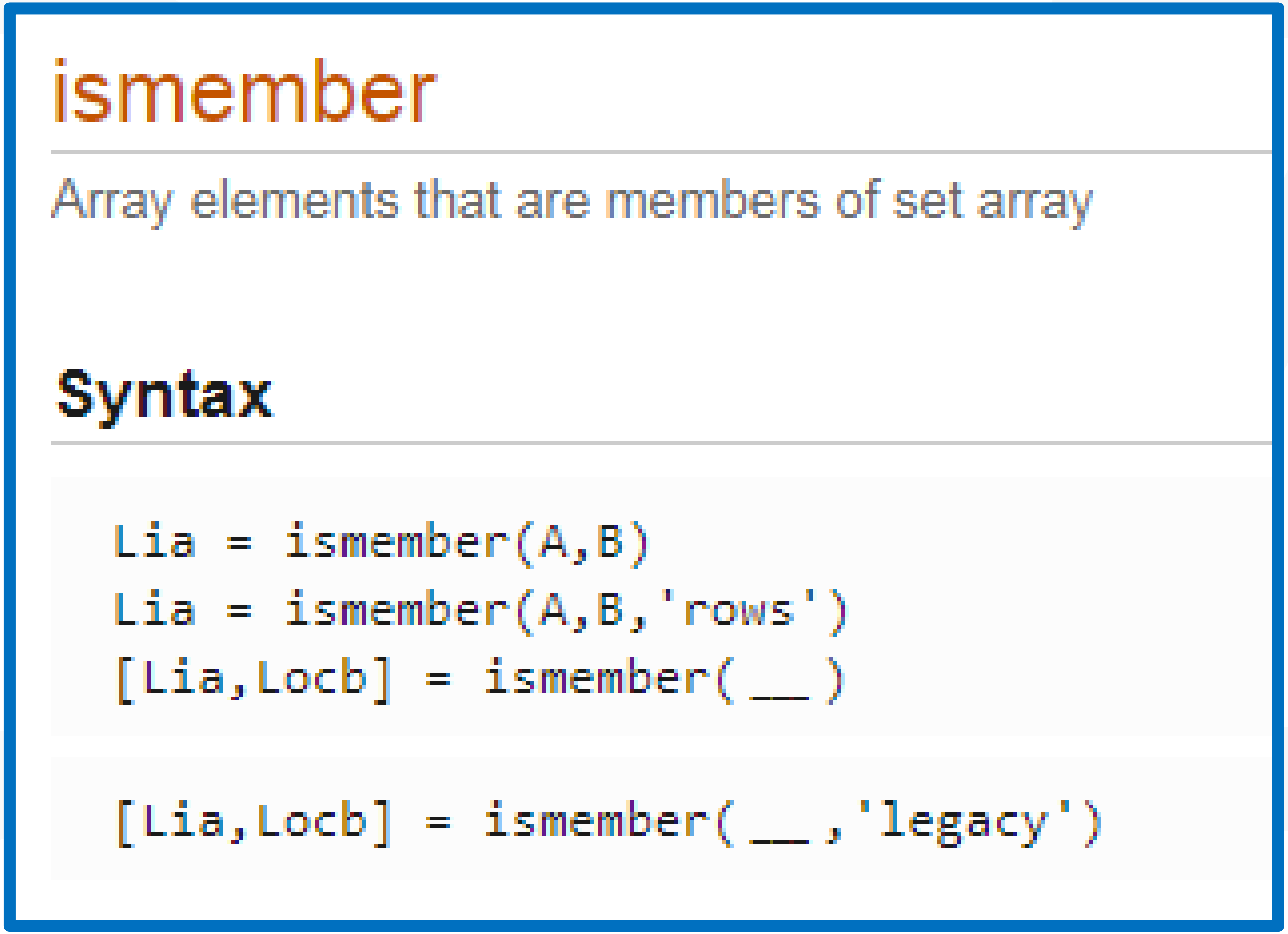
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

MATLAB does not have “Membership Test Operators”. There is a MATAB built-in function for membership test. And this function can do more.

093

`ismember`

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



ismember

Array elements that are members of set array

Syntax

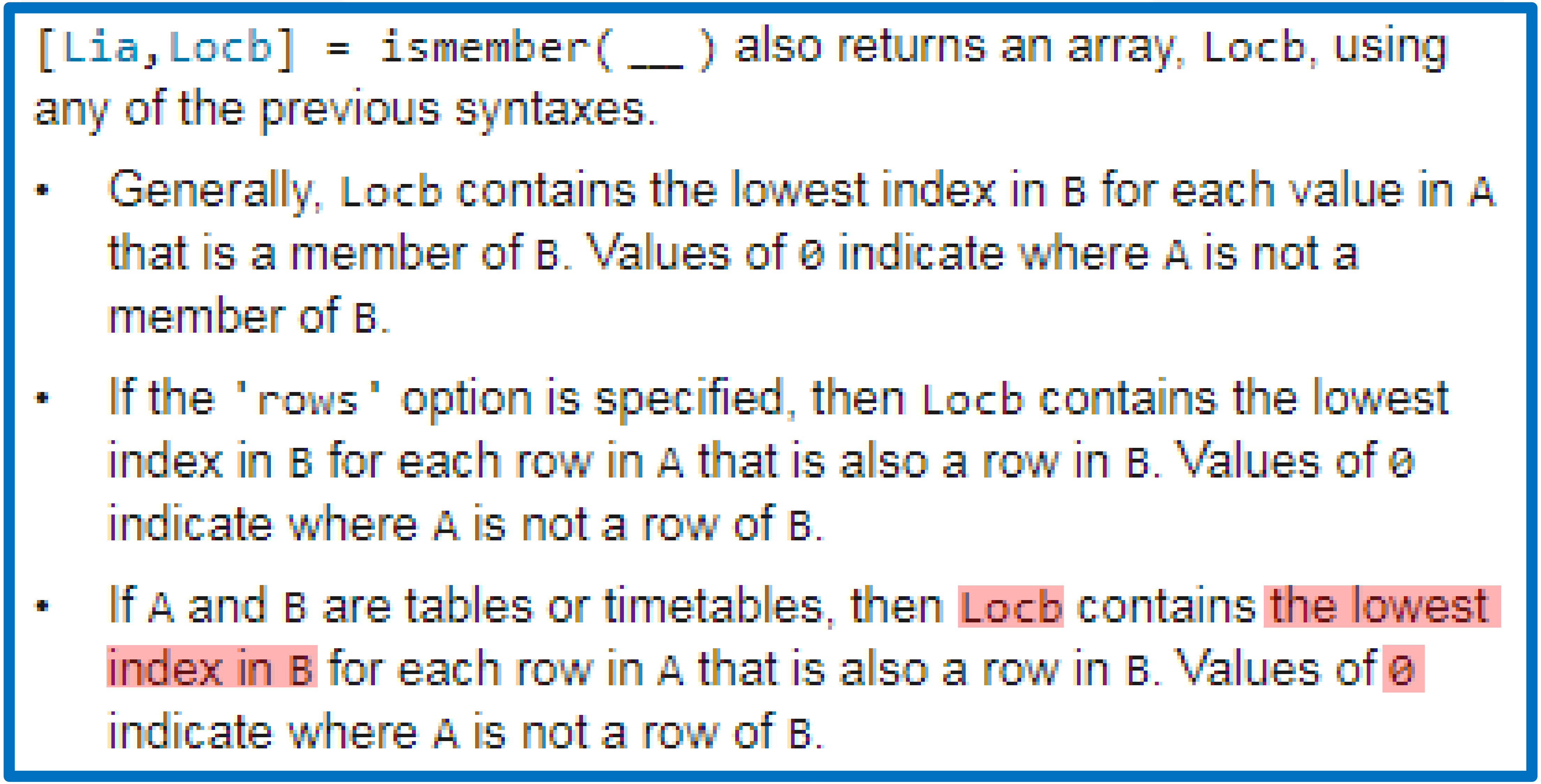
```
Lia = ismember(A,B)
Lia = ismember(A,B,'rows')
[Lia,Locb] = ismember(___)

[Lia,Locb] = ismember(___,'legacy')
```

093

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



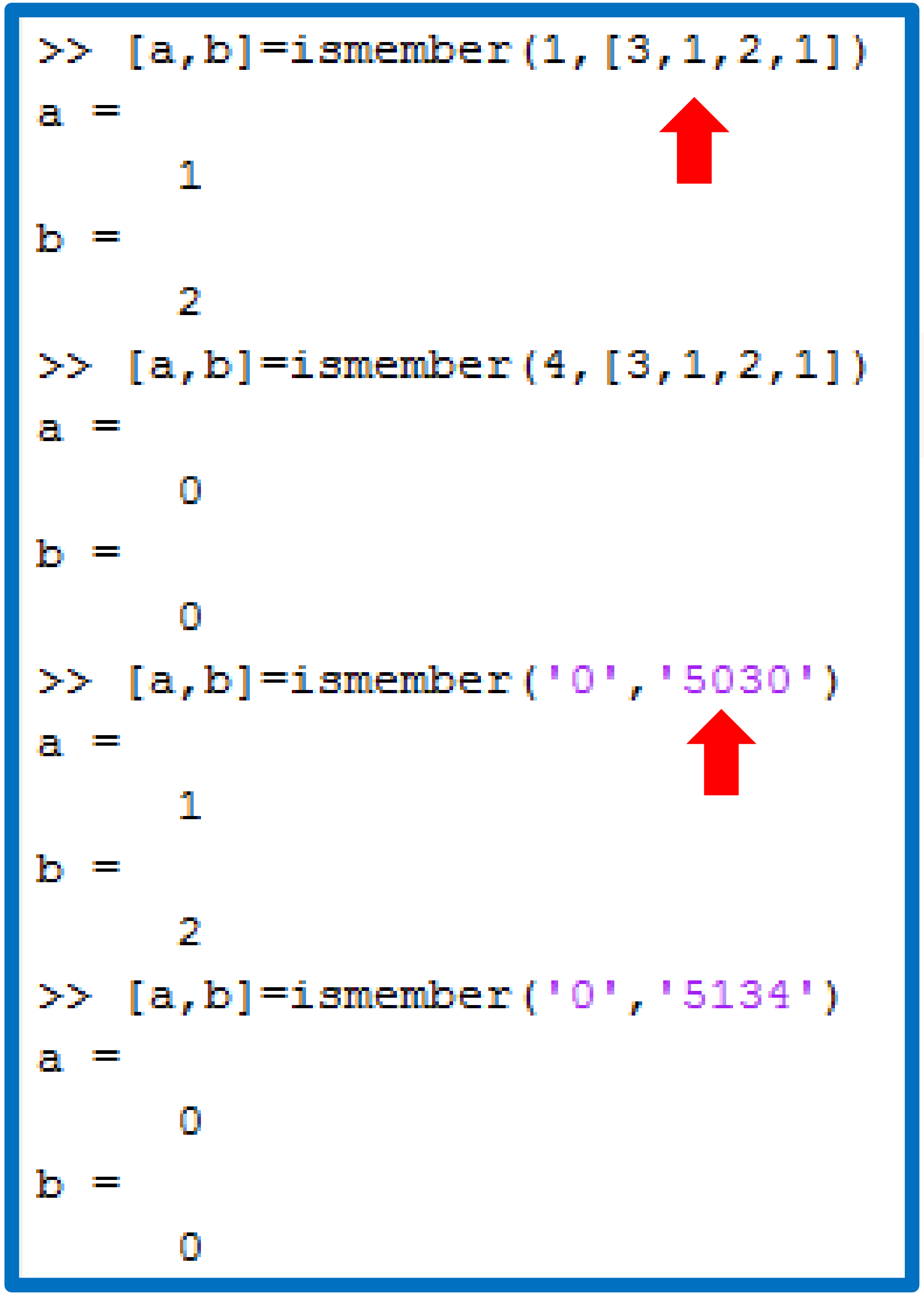
[Lia,Locb] = ismember(__) also returns an array, Locb, using any of the previous syntaxes.

- Generally, Locb contains the lowest index in B for each value in A that is a member of B. Values of 0 indicate where A is not a member of B.
- If the 'rows' option is specified, then Locb contains the lowest index in B for each row in A that is also a row in B. Values of 0 indicate where A is not a row of B.
- If A and B are tables or timetables, then Locb contains the lowest index in B for each row in A that is also a row in B. Values of 0 indicate where A is not a row of B.

SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
>> [a,b]=ismember(1,[3,1,2,1])
a =
     1
b =
     2
>> [a,b]=ismember(4,[3,1,2,1])
a =
     0
b =
     0
>> [a,b]=ismember('0','5030')
a =
     1
b =
     2
>> [a,b]=ismember('0','5134')
a =
     0
b =
     0
```

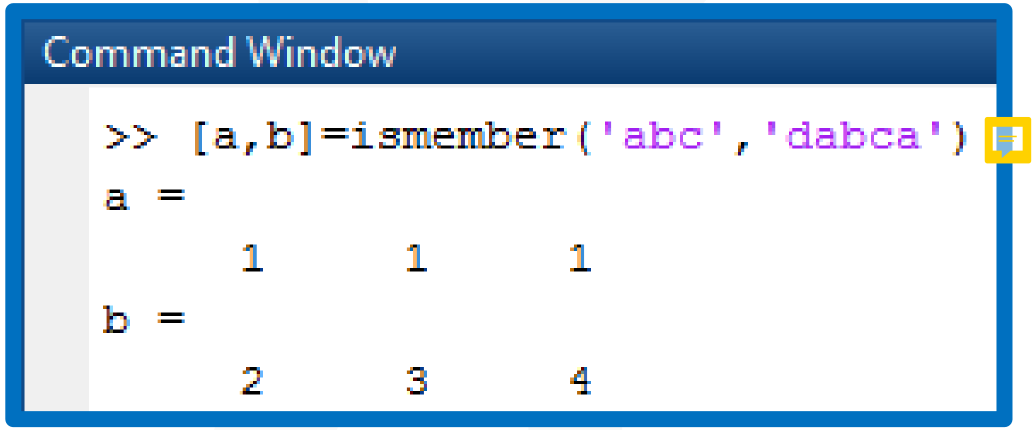
MATLAB does not show “`True`” or “`False`”.

Instead, MATLAB uses `1` and `0`.

Q: Is `'abc'` a substring of `'dabca'`? (MATLAB 2016a)

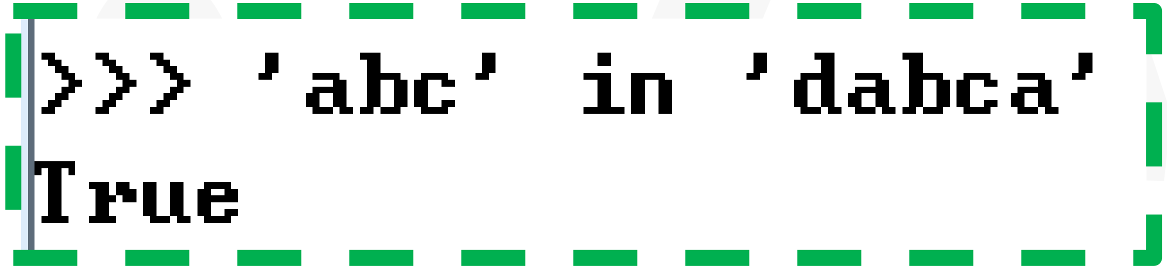
☑ `strfind` can help, but may need two steps.

☒ `ismember` does not help.



```
>> [a,b]=ismember('abc','dabca')
a =
     1     1     1
b =
     2     3     4
```

095



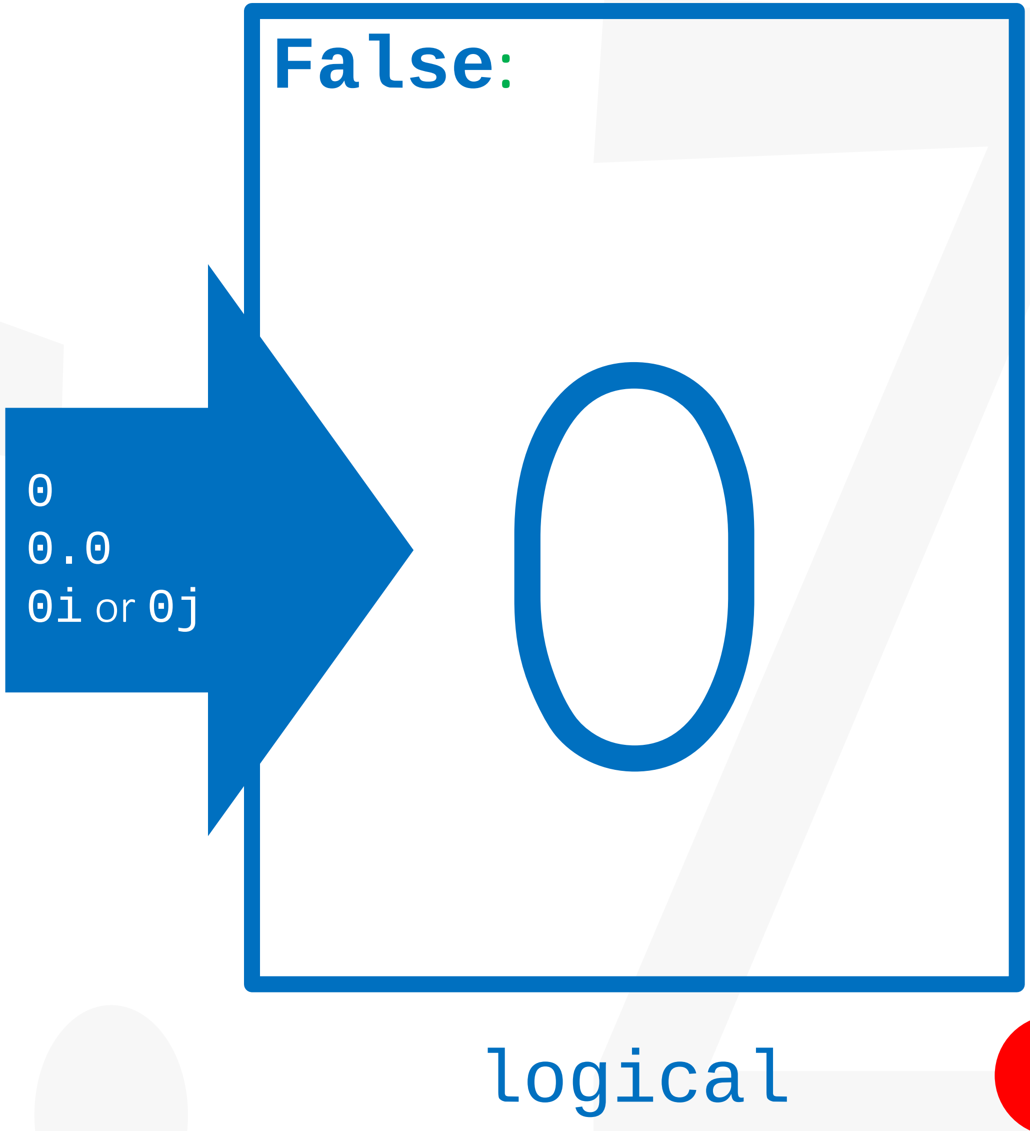
```
>>> 'abc' in 'dabca'
True
```

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Truth Value Testing

False:

☑ None
☑ False
☑ 0
☑ 0.0
☑ 0j
☑ ''
☑ ()
☑ []
☑ {}
☑ range(0)



0
0.0
0i or 0j

False:

0

logical

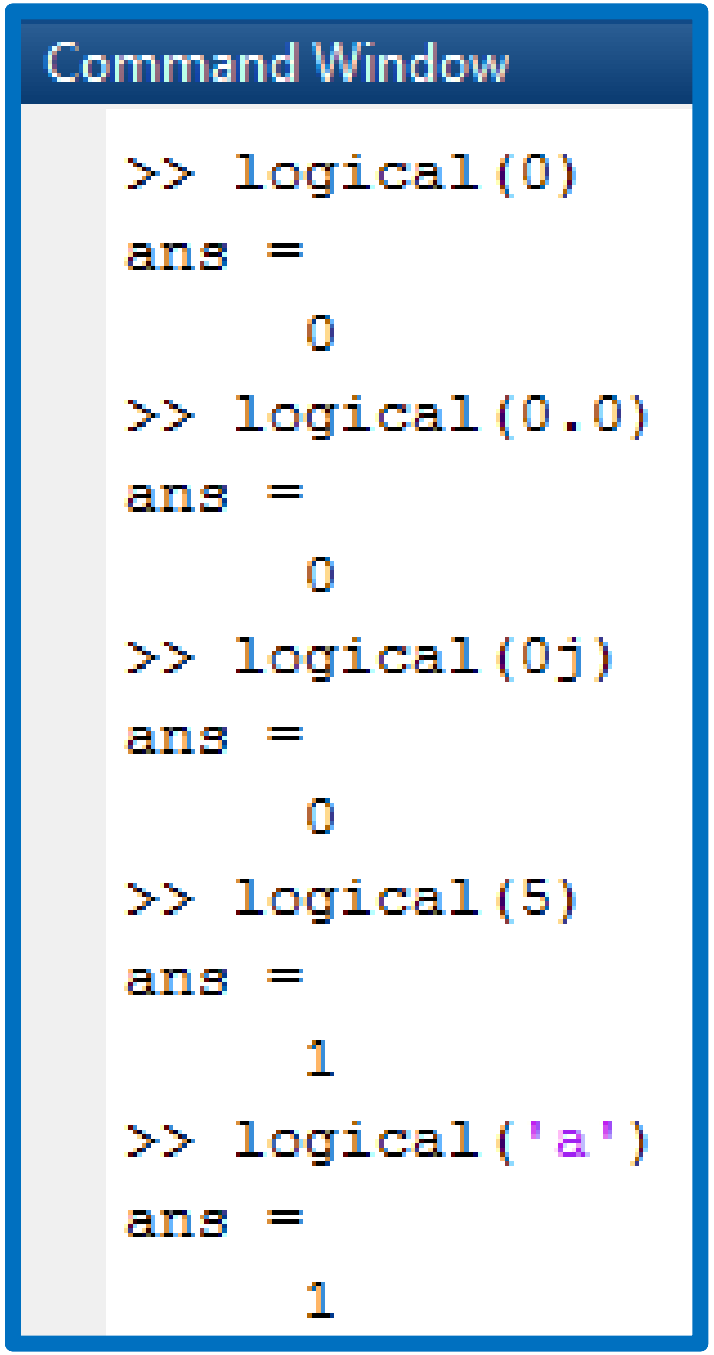
096

(Dr. Z: Some values are neither True nor False. See next slide.)

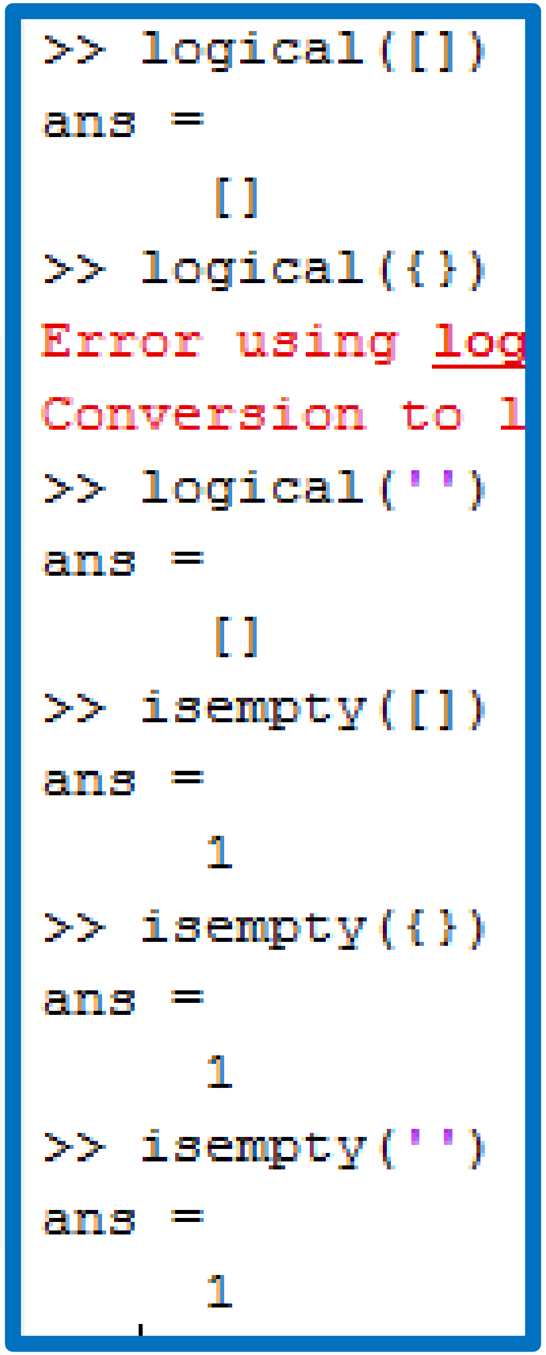
SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
>> logical(0)
ans =
     0
>> logical(0.0)
ans =
     0
>> logical(0j)
ans =
     0
>> logical(5)
ans =
     1
>> logical('a')
ans =
     1
```



```
>> logical([])
ans =
     []
>> logical({})
Error using log
Conversion to l
>> logical('')
ans =
     []
>> isempty([])
ans =
     1
>> isempty({})
ans =
     1
>> isempty('')
ans =
     1
```

097

isempty

(Dr. Z: Use `isempty` on an empty string, an empty array and an empty cell for a truth falue. The empty string and empty array are neither True or False. Empty cell array is even more complicated.)

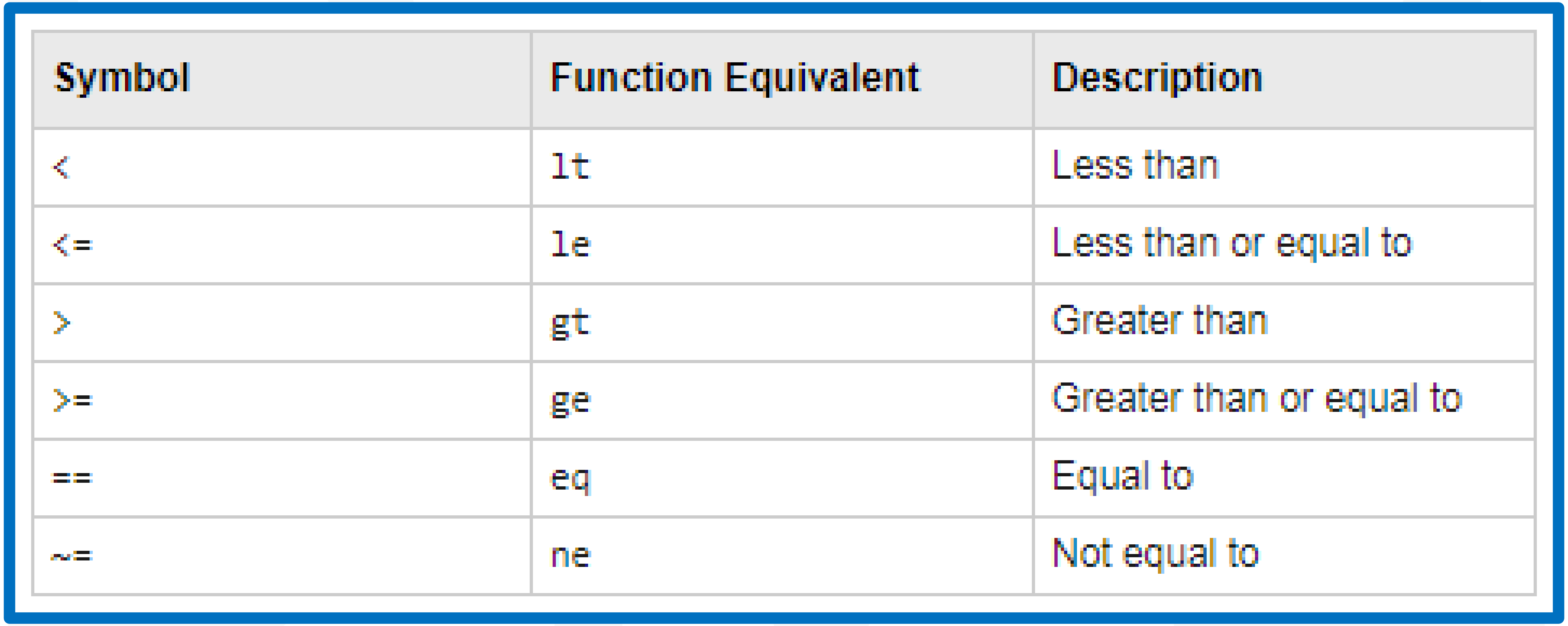
SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

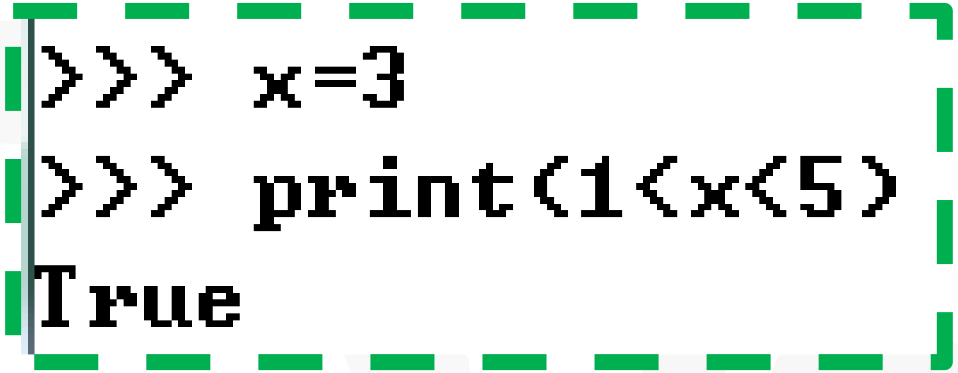
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

Comparison (Relational) Operators

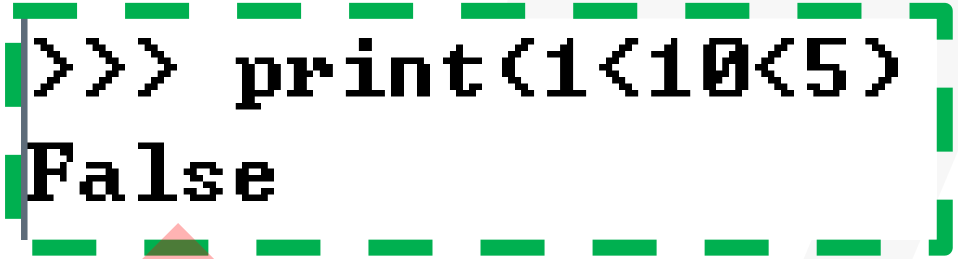
https://www.mathworks.com/help/matlab/matlab_prog/array-comparison-with-relational-operators.html



Symbol	Function Equivalent	Description
`<`	`lt`	Less than
`<=`	`le`	Less than or equal to
`>`	`gt`	Greater than
`>=`	`ge`	Greater than or equal to
`==`	`eq`	Equal to
`~=`	`ne`	Not equal to



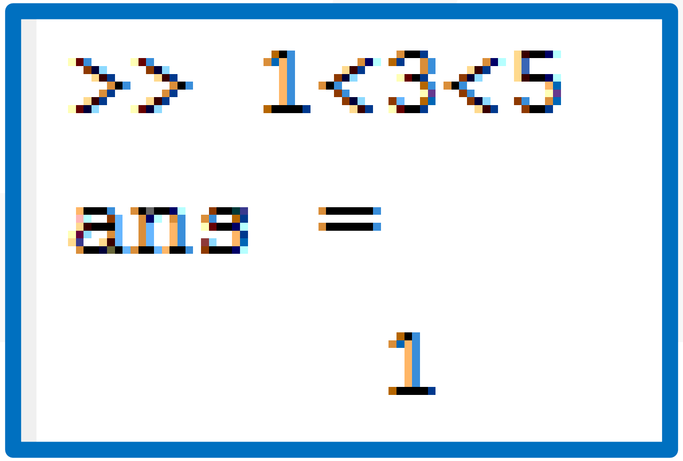
```
>>> x=3
>>> print(1<x<5)
True
```



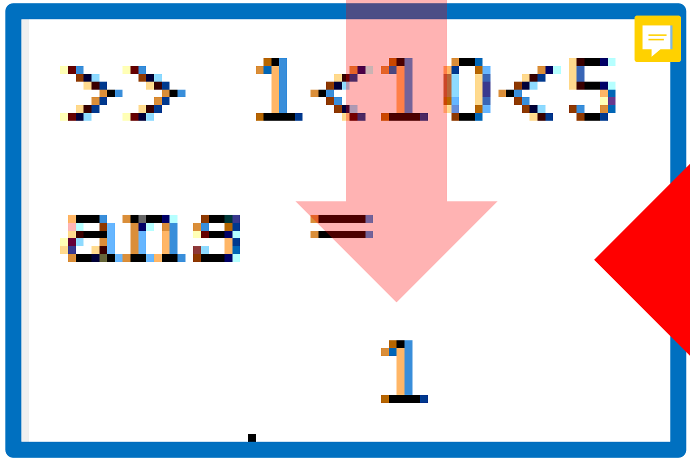
```
>>> print(1<10<5)
False
```

MATLAB does not support "Chained Comparison".

098



```
>> 1<3<5
ans =
     1
```



```
>> 1<10<5
ans =
     1
```

099

Dr. Z: Let me explain.

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Inf, NaN in a Comparison

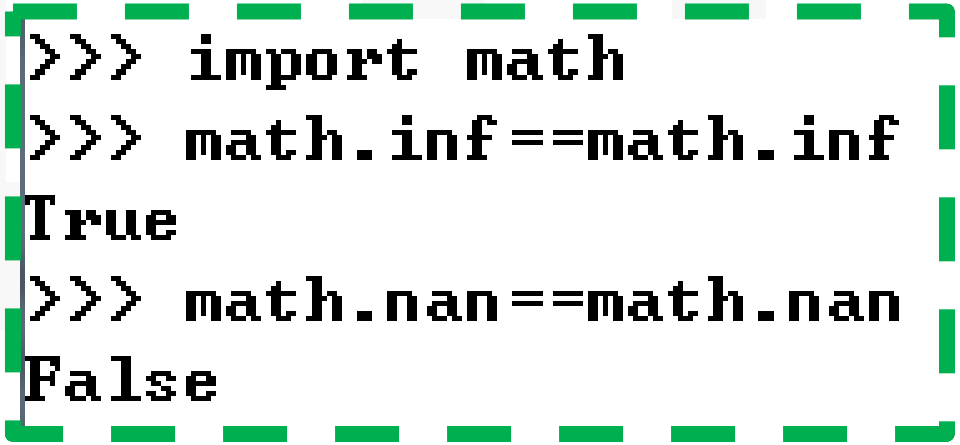
- ❑ `Inf` values are equal to other `Inf` values.
- ❑ NaN values are not equal to any other numeric value, including other NaN values.

`math.inf`

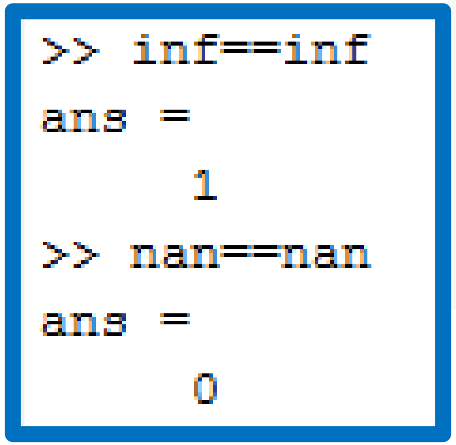
`math.nan`

Inf or inf

NaN or nan



```
>>> import math
>>> math.inf==math.inf
True
>>> math.nan==math.nan
False
```



```
>> inf==inf
ans =
     1
>> nan==nan
ans =
     0
```

100

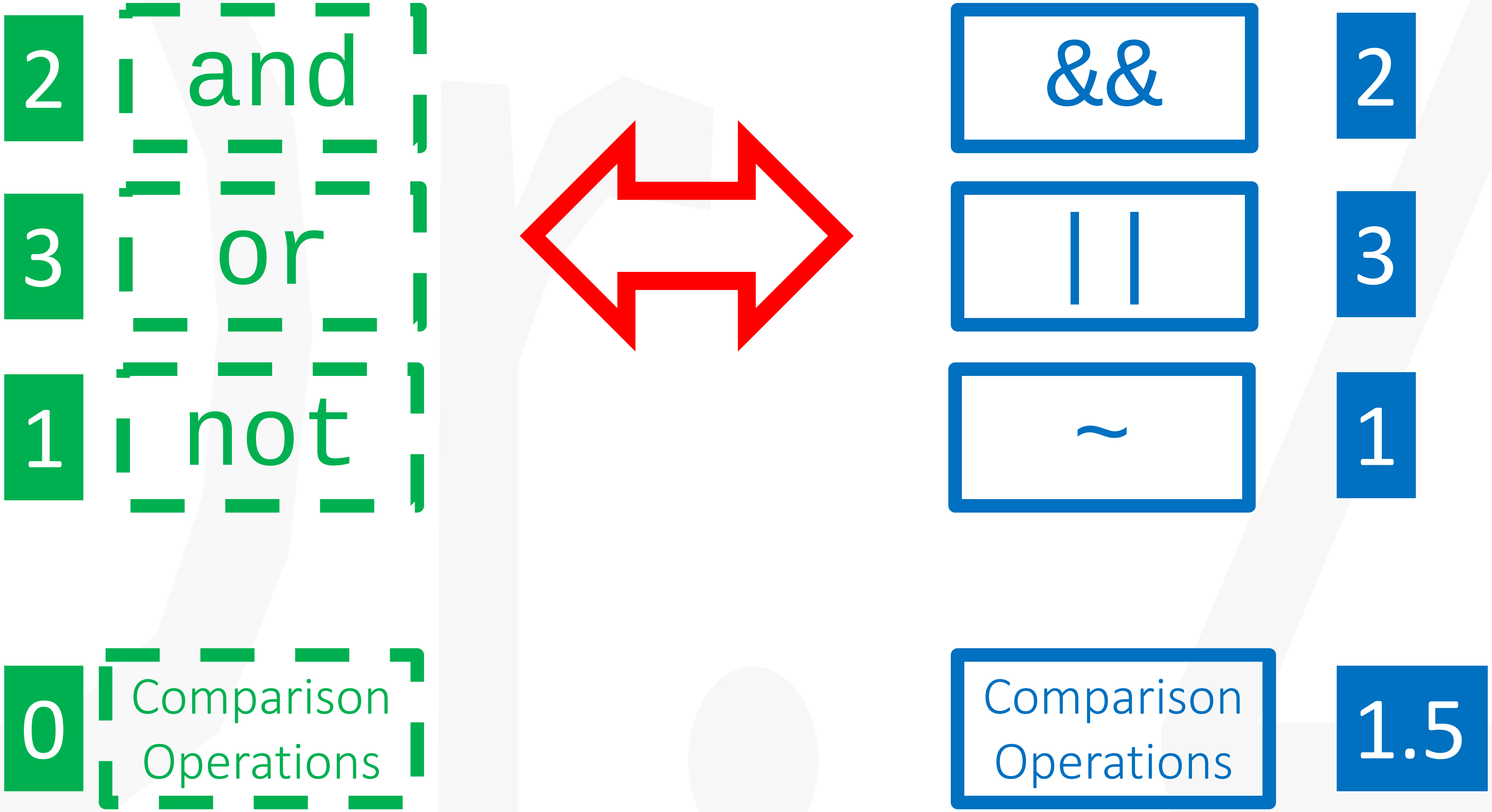
101

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

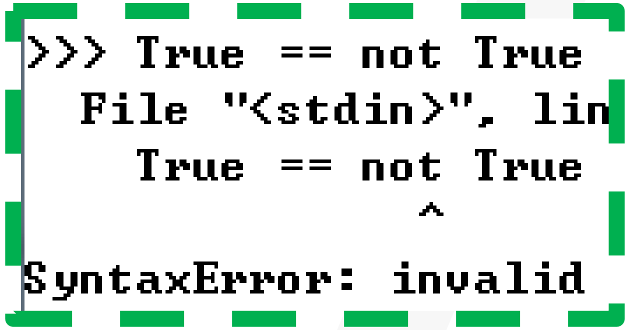
Boolean (Logical) Operations



2	and	&&	2
3	or	||	3
1	not	~	1
0	Comparison Operations	Comparison Operations	1.5

☒ a == not b

⇔ (a == not) b

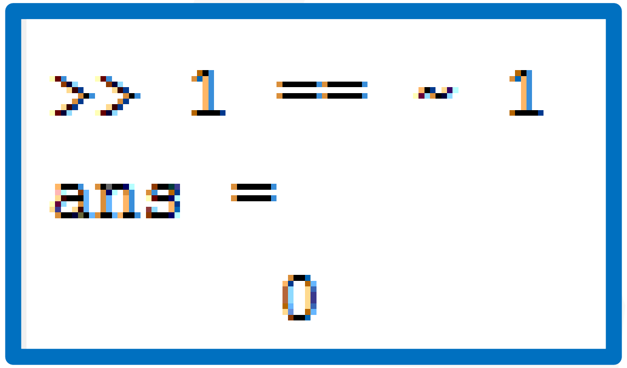


```
>>> True == not True
  File "<stdin>", lin
    True == not True
                ^
SyntaxError: invalid
```

☑ a == ~ b

⇔ a == (~ b)

102



```
>> 1 == ~ 1
ans =
     0
```

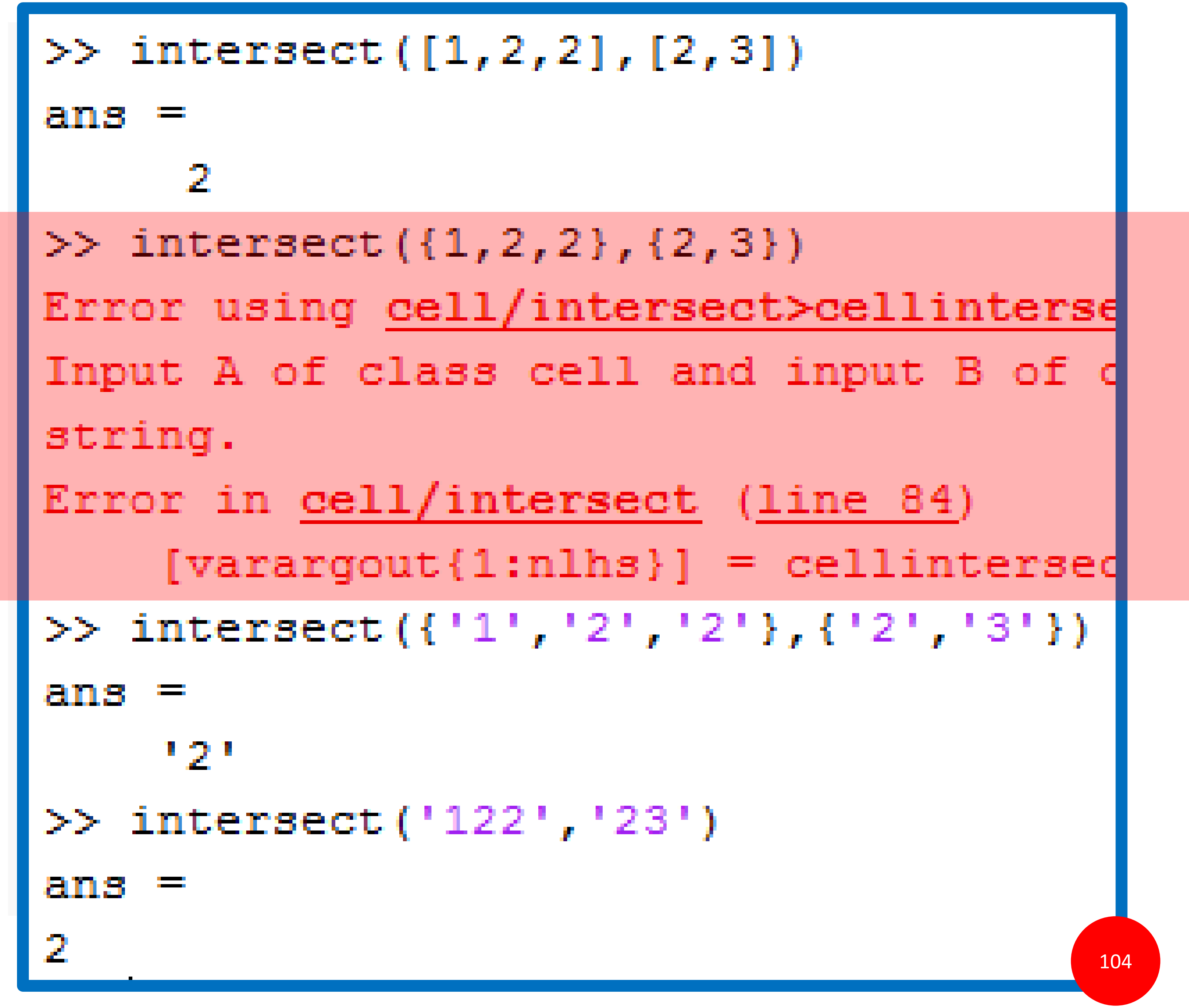
QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

MATLAB does not have “augmented assignment”.

MATLAB does not have set or dictionary. However, MATLAB has some functions for set operations.

`intersect` `ismember` `setdiff` `union` `unique`



```
>> intersect([1,2,2],[2,3])
ans =
     2
>> intersect({1,2,2},{2,3})
Error using cell/intersect>cellinterse
Input A of class cell and input B of c
string.
Error in cell/intersect (line 84)
    [varargout{1:nlhs}] = cellintersec
>> intersect({'1','2','2'},{'2','3'})
ans =
    '2'
>> intersect('122','23')
ans =
2
```

intersect

ismember

setdiff

union

unique

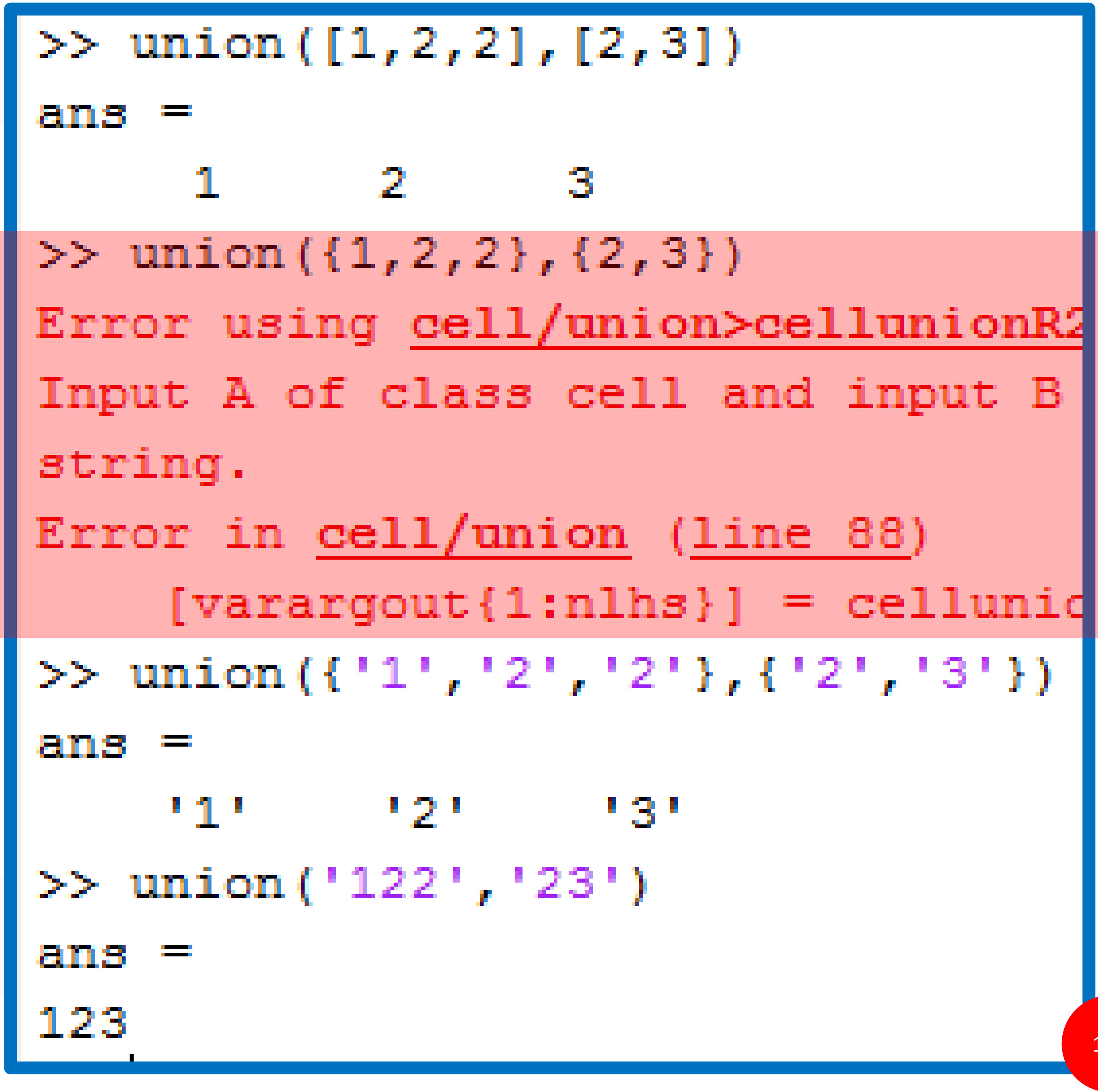
QF666
Programming and
Computational
Finance

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



```
>> union([1,2,2],[2,3])
ans =
     1     2     3
>> union({1,2,2},{2,3})
Error using cell/union>cellunionR2
Input A of class cell and input B
string.
Error in cell/union (line 88)
       [varargout{1:nlhs}] = cellunic
>> union({'1','2','2'},{'2','3'})
ans =
    '1'    '2'    '3'
>> union('122','23')
ans =
123
```

intersect

ismember

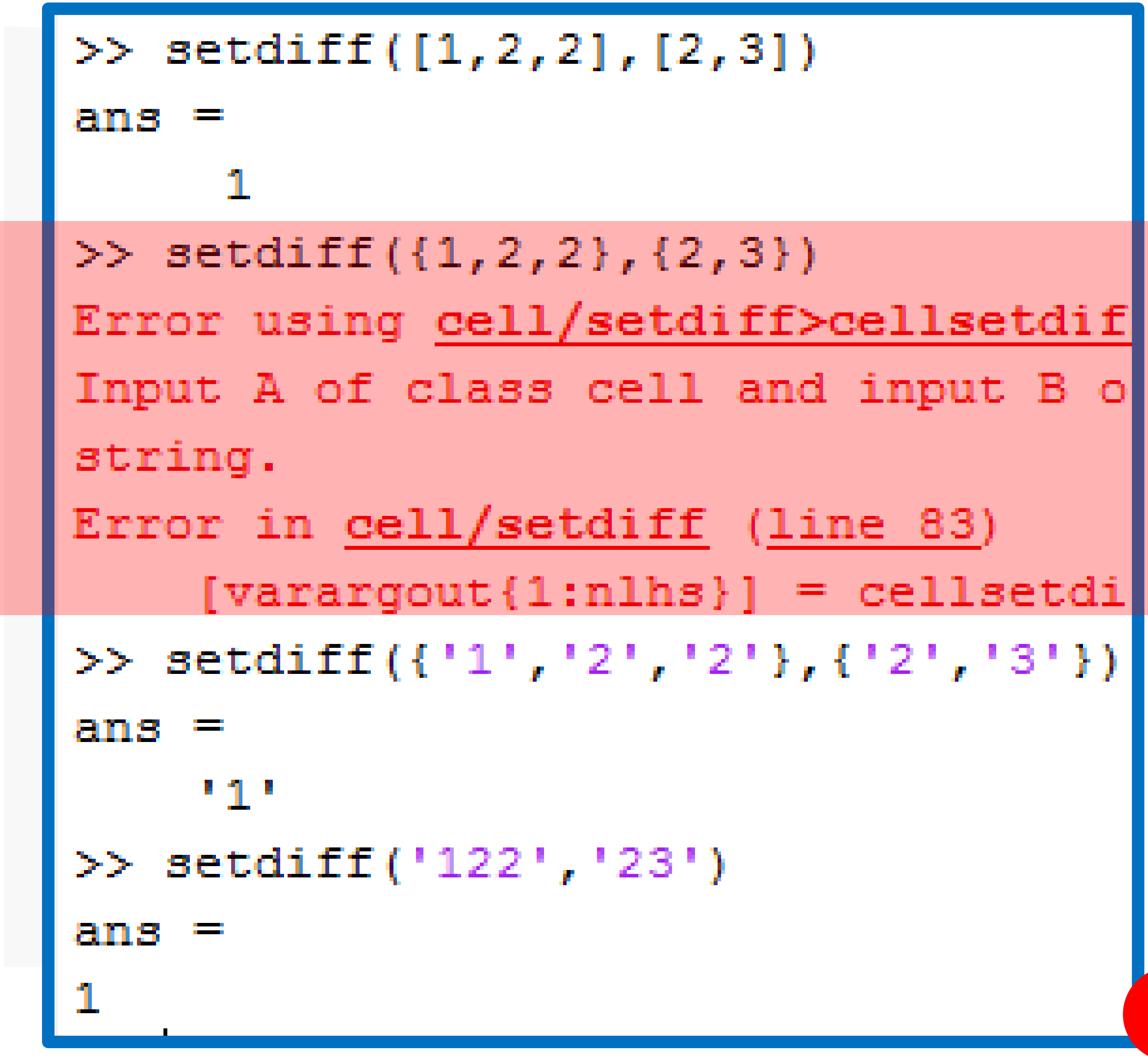
setdiff

union

unique

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
>> setdiff([1,2,2],[2,3])
ans =
     1
>> setdiff({1,2,2},{2,3})
Error using cell/setdiff>cellsetdif
Input A of class cell and input B o
string.
Error in cell/setdiff (line 83)
    [varargout{1:nlhs}] = cellsetdi
>> setdiff({'1','2','2'},{'2','3'})
ans =
    '1'
>> setdiff('122','23')
ans =
1
```

intersect

ismember

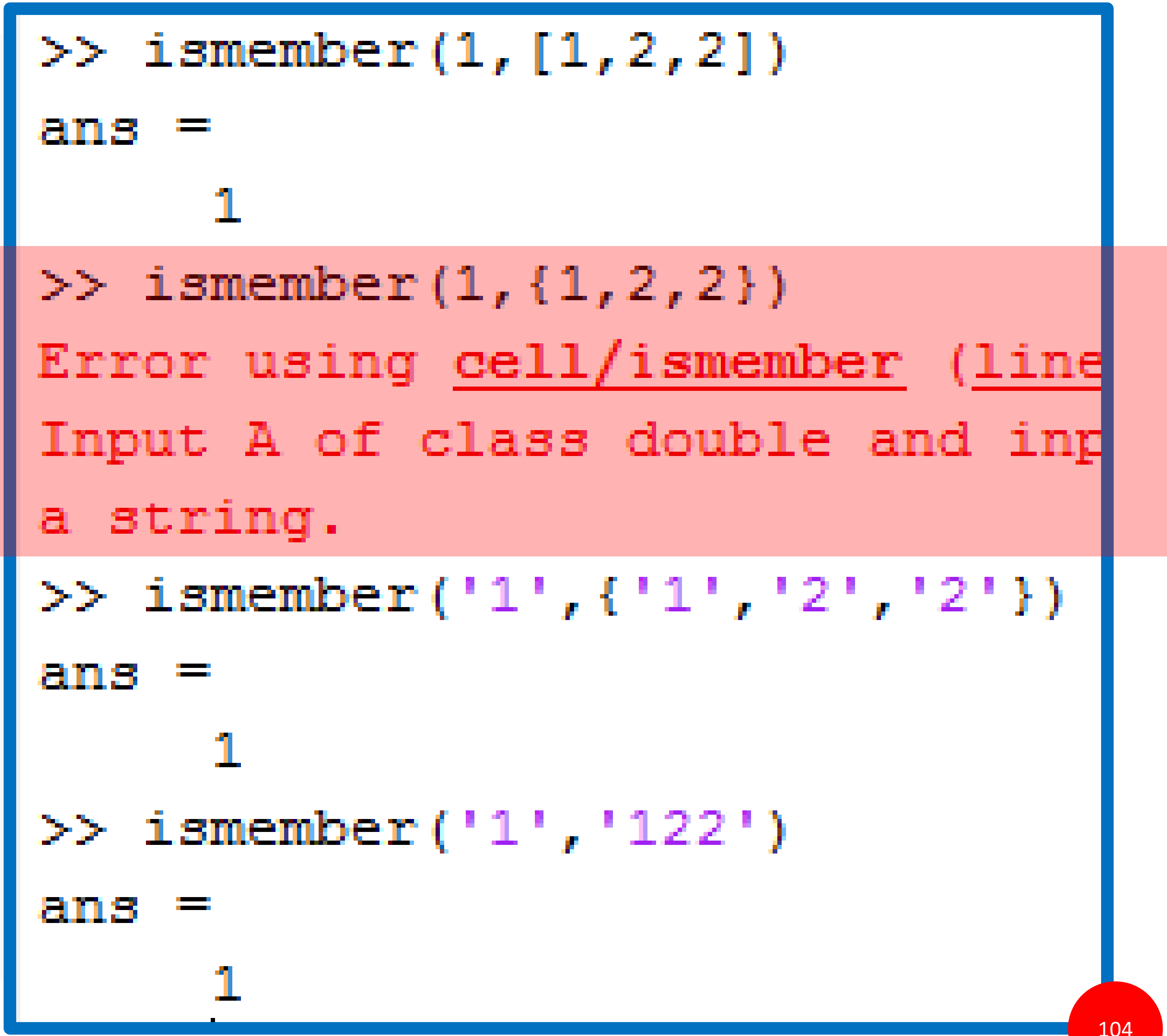
setdiff

union

unique

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



```
>> ismember(1,[1,2,2])
ans =

     1

>> ismember(1,{1,2,2})
Error using cell/ismember (line
Input A of class double and inp
a string.
>> ismember('1',{'1','2','2'})
ans =

     1

>> ismember('1','122')
ans =

     1
```

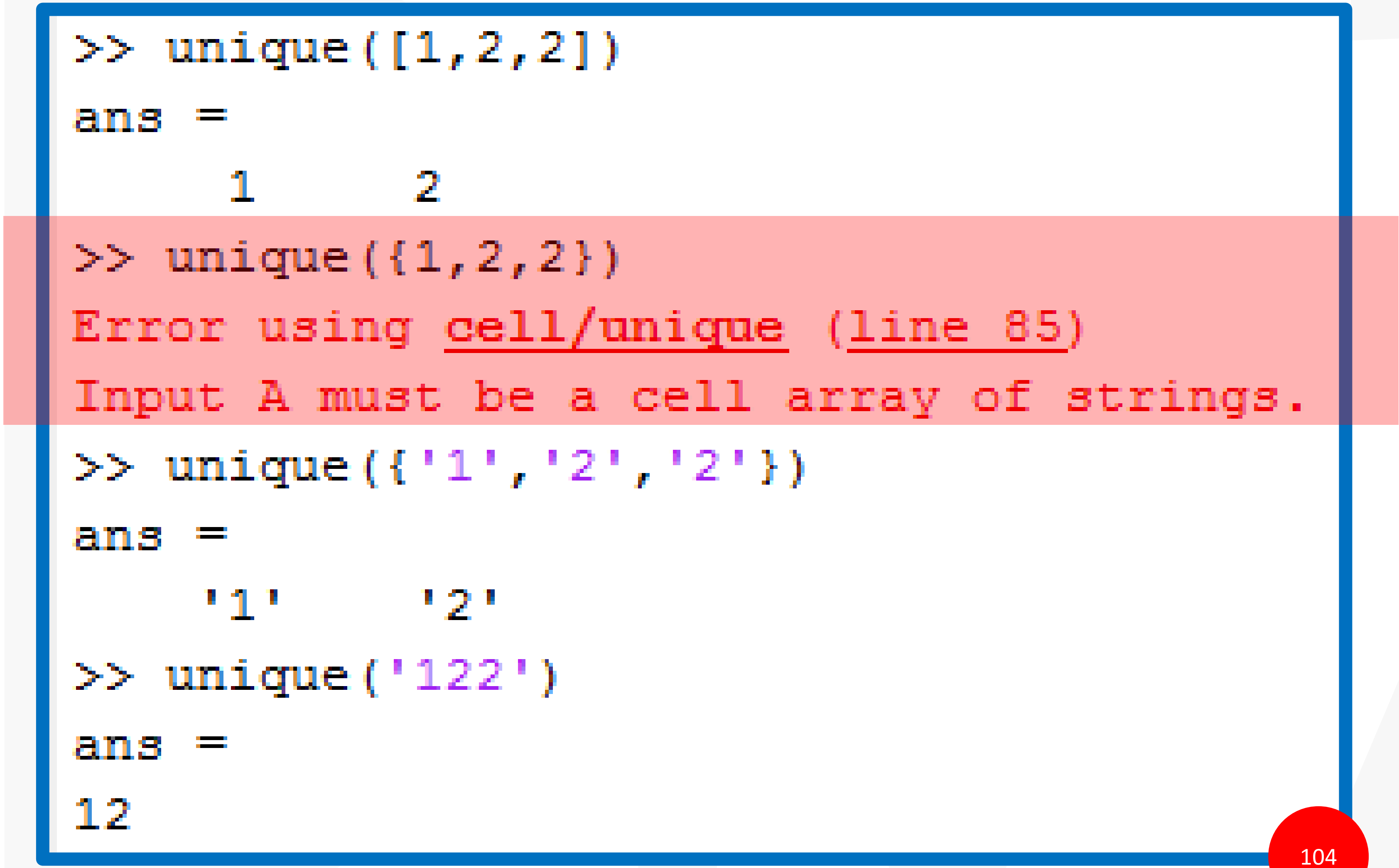
intersect

ismember

setdiff

union

unique



```
>> unique([1,2,2])
ans =
     1     2
>> unique({1,2,2})
Error using cell/unique (line 85)
Input A must be a cell array of strings.
>> unique({'1','2','2'})
ans =
    '1'    '2'
>> unique('122')
ans =
12
```

intersect

ismember

setdiff

union

unique

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

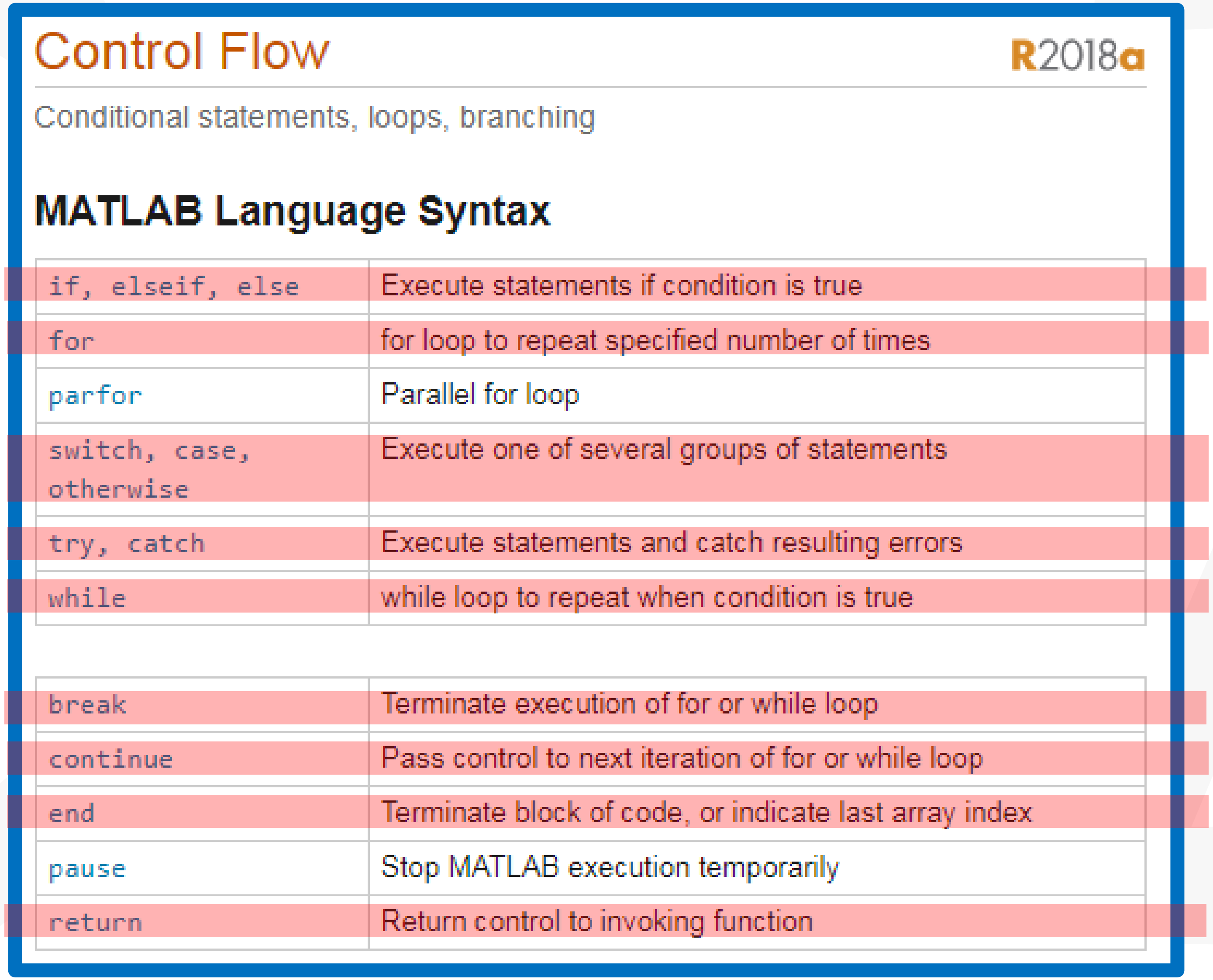
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

https://www.mathworks.com/help/matlab/control-flow.html



Control Flow

R2018a

Conditional statements, loops, branching

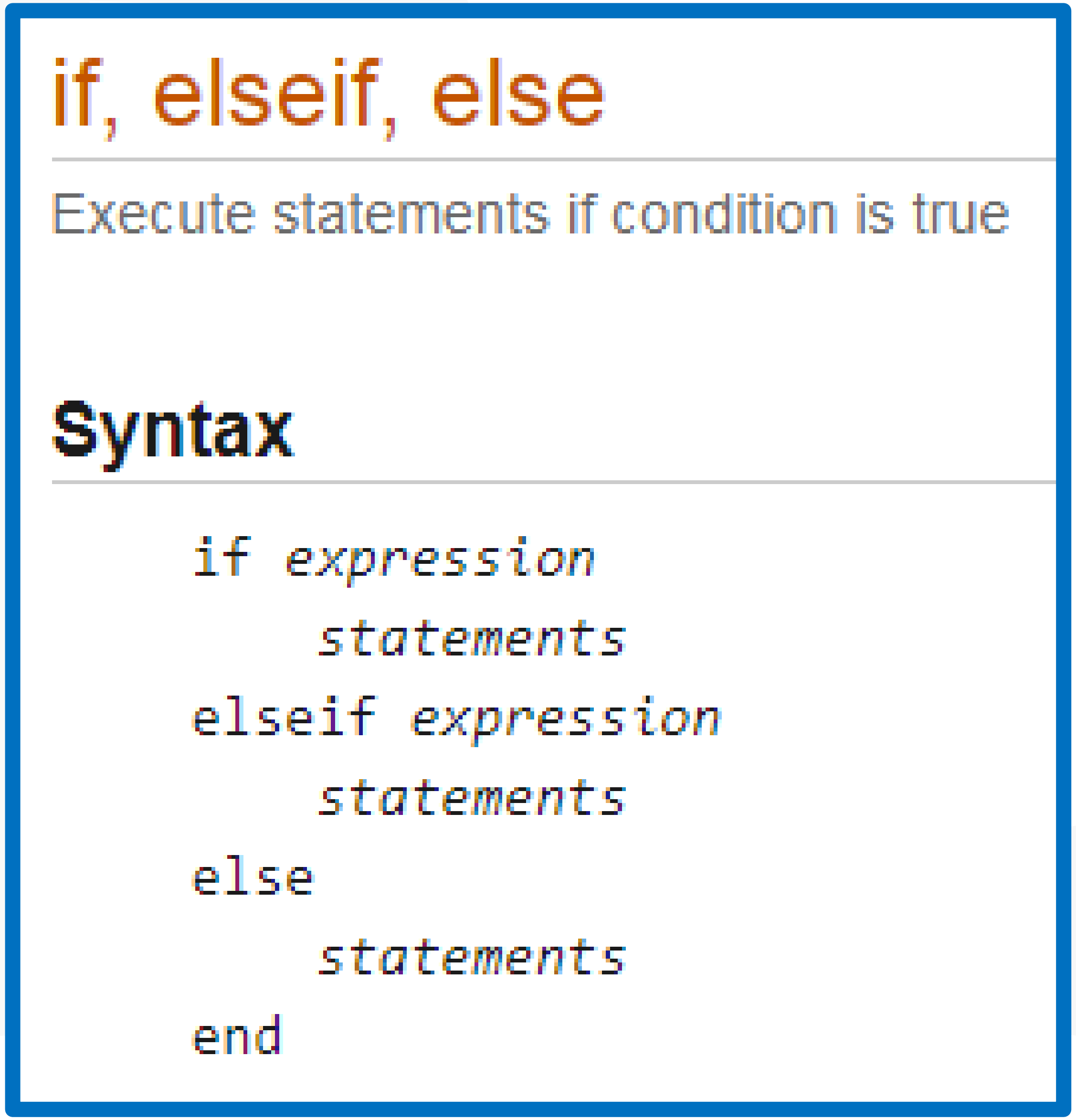
MATLAB Language Syntax

`if, elseif, else`	Execute statements if condition is true
`for`	for loop to repeat specified number of times
`parfor`	Parallel for loop
`switch, case, otherwise`	Execute one of several groups of statements
`try, catch`	Execute statements and catch resulting errors
`while`	while loop to repeat when condition is true

`break`	Terminate execution of for or while loop
`continue`	Pass control to next iteration of for or while loop
`end`	Terminate block of code, or indicate last array index
`pause`	Stop MATLAB execution temporarily
`return`	Return control to invoking function

MATLAB has `return` statement. But this `return` is not that `return`.

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



if, elseif, else

Execute statements if condition is true

Syntax

```
if expression
    statements
elseif expression
    statements
else
    statements
end
```

106

```
if expression:
    statements
elif expression:
    statements
else:
    statements
```

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

2. Income Tax Calculator (ver. 1)

```
x=150_000
if 0<=x<20_000:
    y=0
elif 20_000<=x<30_000:
    y=0+0.02*(x-20_000)
elif 30_000<=x<40_000:
    y=200+0.035*(x-30_000)
elif 40_000<=x<80_000:
    y=550+0.07*(x-40_000)
elif 80_000<=x<120_000:
    y=3_350+0.115*(x-80_000)
elif 120_000<=x<160_000:
    y=7_950+0.15*(x-120_000)
elif 160_000<=x<200_000:
    y=13_950+0.18*(x-160_000)
elif 200_000<=x<240_000:
    y=21_150+0.19*(x-200_000)
elif 240_000<=x<280_000:
    y=28_750+0.195*(x-240_000)
elif 280_000<=x<320_000:
    y=36_550+0.2*(x-280_000)
else:
    y=44_550+0.22*(x-320_000)
print(y)
```

A<=B<C ⇔ A<=B and B<C

and ⇔ &&

elif ⇔ elseif

if ⇔ if...end

print ⇔ disp

In-Class Exercise

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance

2. Income Tax Calculator (ver. 2)

```
bi=[0, 200, 550, 3350, 7950, 13950, 21150, 28750, 36550, 44550]
mi=[2.0, 3.5, 7.0, 11.5, 15.0, 18.0, 19.0, 19.5, 20.0, 22.0]
xi=[20000, 30000, 40000, 80000, 120000, 160000, 200000, 240000, 280000, 320000]
x=400_000
if x>xi[-1]:
    i=len(xi)-1
else:
    i=next(filter(lambda w: w[1]>x, enumerate(xi)))[0]-1

if i==-1:
    y=0
else:
    y=bi[i]+mi[i]/100*(x-xi[i])
print(y)
```

⇨i=len(xi)-1

⇨i=-1,0,…,len(xi)-2

In-Class Exercise

Homework Q2

```
i=next(filter(lambda w: w[1]>x, enumerate(xi)))[0]-1
```

Find the position of the first number in a list, xi, which is greater than x.

```
c=find(xi>x);
i=c(1)-1;
```

[i] ⇔ (i) or (i+1)

[-1] ⇔ (end)

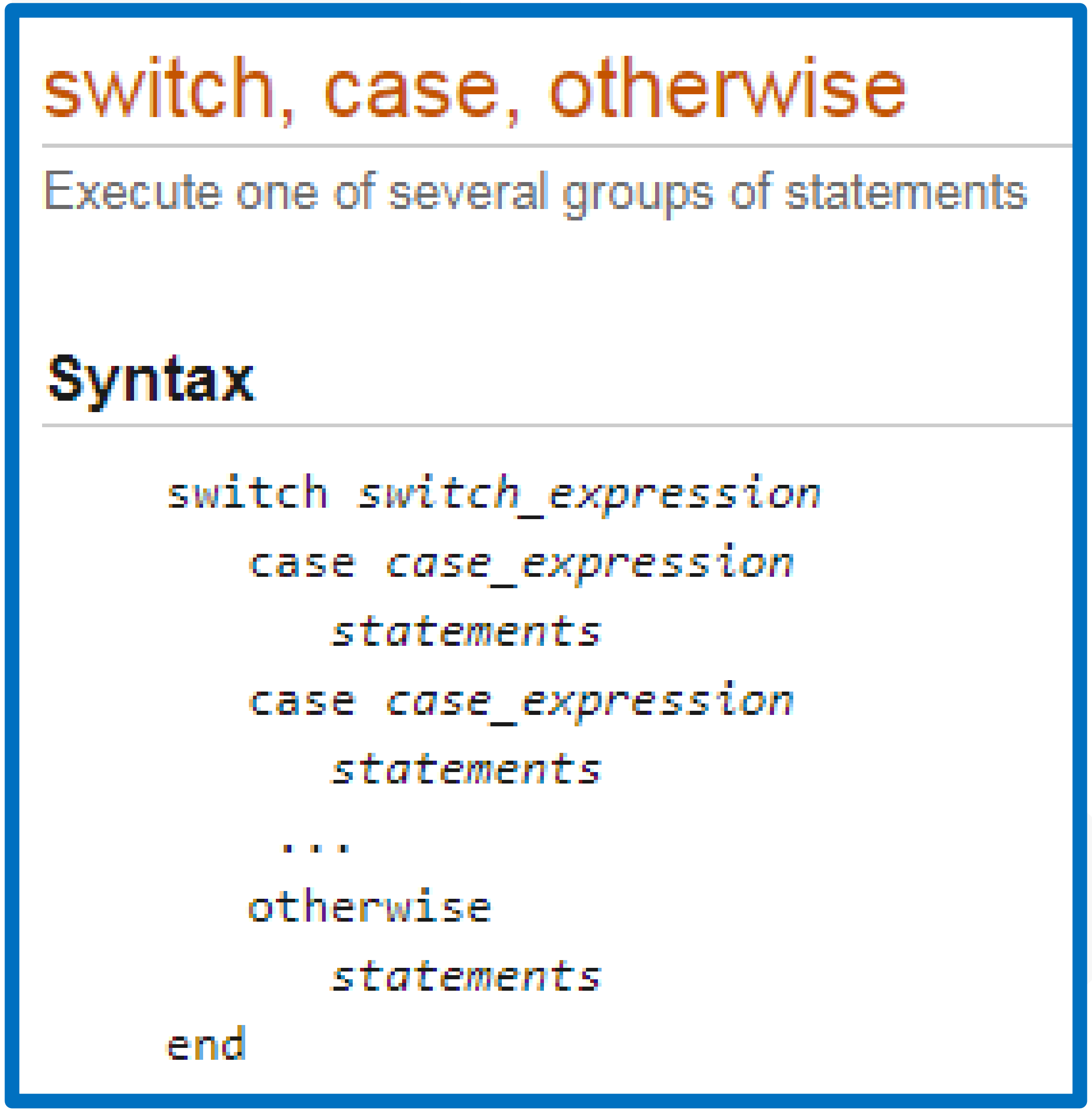
len() ⇔ length()

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

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



switch, case, otherwise

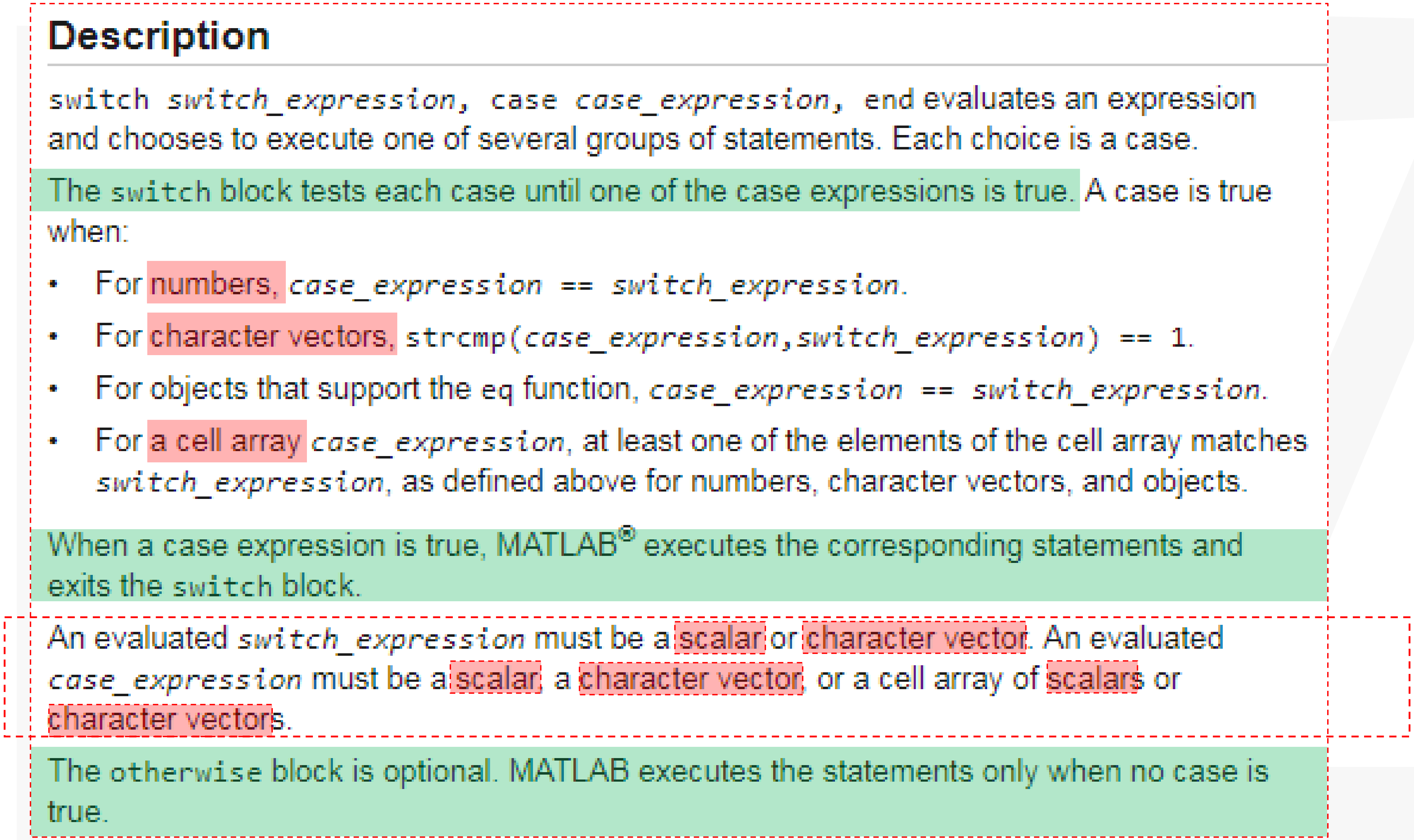
Execute one of several groups of statements

Syntax

```
switch switch_expression
   case case_expression
      statements
   case case_expression
      statements
    ...
   otherwise
      statements
end
```

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance



Description

`switch switch_expression, case case_expression, end` evaluates an expression and chooses to execute one of several groups of statements. Each choice is a case.

The `switch` block tests each case until one of the case expressions is true. A case is true when:

- For numbers, `case_expression == switch_expression`.
- For character vectors, `strcmp(case_expression,switch_expression) == 1`.
- For objects that support the `eq` function, `case_expression == switch_expression`.
- For a cell array `case_expression`, at least one of the elements of the cell array matches `switch_expression`, as defined above for numbers, character vectors, and objects.

When a case expression is true, MATLAB® executes the corresponding statements and exits the `switch` block.

An evaluated `switch_expression` must be a scalar or character vector. An evaluated `case_expression` must be a scalar, a character vector, or a cell array of scalars or character vectors.

The `otherwise` block is optional. MATLAB executes the statements only when no case is true.

Scalars: 1x1 matrices; **Vectors:** 1xn or nx1 matrices; **Matrices:** mxn, where m, n >=2

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

isscalar, isvector, ismatrix, isempty

A	isscalar(A)	isvector(A)	ismatrix(A)	isempty(A)
2	1 (true)	1 (true)	1 (true)	0 (false)
[2, 2]	0 (false)	1 (true)	1 (true)	0 (false)
[2; 2]	0 (false)	1 (true)	1 (true)	0 (false)
[2 2;2 2]	0 (false)	0 (false)	1 (true)	0 (false)
[]	0 (false)	0 (false)	1 (true)	1 (true)
'a'	1 (true)	1 (true)	1 (true)	0 (false)
'abc'	0 (false)	1 (true)	1 (true)	0 (false)
''	0 (false)	0 (false)	1 (true)	1 (true)

- A character vector is a vector, and therefore is a matrix.
- A single character is a scalar, and therefore is a vector and is a matrix.

For Numbers

```
n = input('Enter a number: ');

switch n
    case -1
        disp('negative one')
    case 0
        disp('zero')
    case 1
        disp('positive one')
    otherwise
        disp('other value')
end
```

```
n = input('Enter a number: ');

if n==-1
    disp('negative one')
elseif n==0
    disp('zero')
elseif n==1
    disp('positive one')
else
    disp('other value')
end
```

Do Not Test

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

For Character Vectors

```
s=input('Enter a string:','s');

switch n
    case 'Good'
        disp('Good')
    case 'Bad'
        disp('Bad')
    otherwise
        disp('others')
end
```

⇔

```
s=input('Enter a string:','s');

if strcmp(s, 'Good')
    disp('Good')
elseif strcmp(s, 'Bad')
    disp('Bad')
else
    disp('others')
end
```

Do Not Test

For a Cell Array

```
n = input('Enter a number:');

switch n
    case {-1, 1}
        disp('abs=1')
    case 0
        disp('abs=0')
    case {-2, 2}
        disp('abs=2')
    otherwise
        disp('abs=other')
end
```

⇔

```
n = input('Enter a number:');

if n==-1 || n==1
    disp('abs=1')
elseif n==0
    disp('abs=0')
elseif n==-2 || n==2
    disp('abs=2')
else
    disp('abs=other')
end
```

Do Not Test

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

A Challenge

```
n = input('Enter a number:');

switch 1
    case 0<=n && n<50
        disp('F')
    case n>=80
        disp('A')
    otherwise
        disp('others')
end
```

⇔

```
n = input('Enter a number:');

if 0<=n && n<50
    disp('F')
elseif n>=80
    disp('A')
else
    disp('others')
end
```

Do Not Test

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

110

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

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

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

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

(Dr. Z: Python uses iterables in a for-loop. MATLAB uses functions to help to loop over items of a data structure.)

arrayfun, cellfun and structfun (map)

✓ arrayfun

Apply function to each element of an array.

❑ **B=arrayfun(func, A):** B(i)=func(A(i)) → row array

❑ **B=arrayfun(func, A1, …, An):** B(i)=func(A1(i),…,An(i)) → row array

❑ **B=arrayfun(__,'UniformOutput',false)** → cell array

❑ **[B1,…,Bm]=arrayfun(___)**

✓ cellfun

Apply function to each cell in a cell array

❑ **A=cellfun(func, C):** A(i)=func(C{i}) → row array

❑ **A=cellfun(func, C1, …, Cn):** A(i)=func(C1{i},…,Cn{i}) → row array

❑ **A=cellfun(__, 'UniformOutput',false)** → cell array

❑ **[A1,…,Am]=cellfun(___)**

✓ structfun

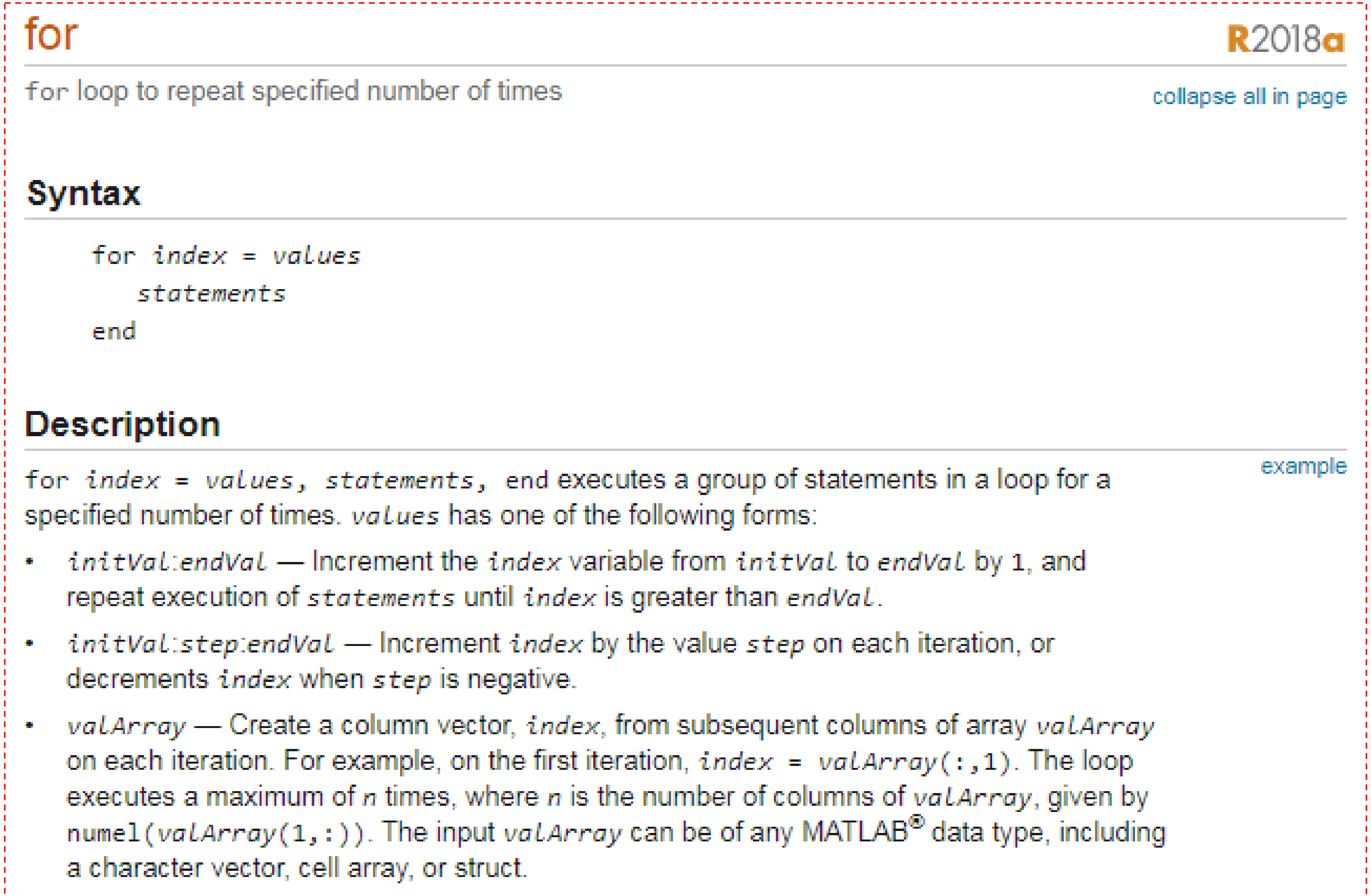
Apply function to each field of a scalar structure

❑ **A = structfun(func,S)** → column array

❑ **A = structfun(func,S, 'UniformOutput',false)** → cell array

❑ **[A1,...,Am] = structfun(___)**

Loop Controls (1): for loop



for

R2018a

for loop to repeat specified number of times

collapse all in page

Syntax

```
for index = values
   statements
end
```

Description

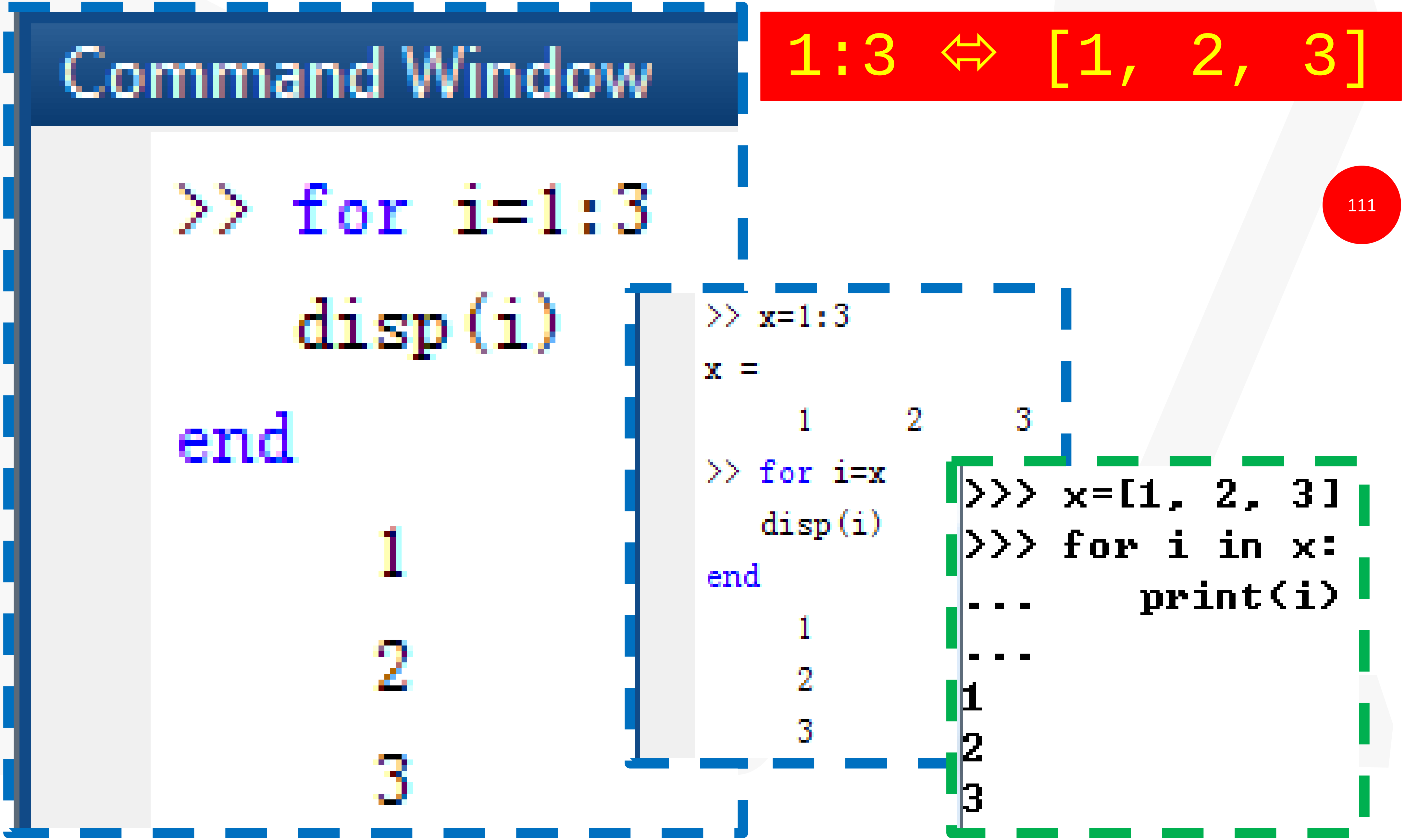
example

`for index = values, statements, end` executes a group of statements in a loop for a specified number of times. *values* has one of the following forms:

- *initVal*:*endVal* — Increment the *index* variable from *initVal* to *endVal* by `1`, and repeat execution of *statements* until *index* is greater than *endVal*.
- *initVal*:*step*:*endVal* — Increment *index* by the value *step* on each iteration, or decrements *index* when *step* is negative.
- *valArray* — Create a column vector, *index*, from subsequent columns of array *valArray* on each iteration. For example, on the first iteration, `index = valArray(:,1)`. The loop executes a maximum of *n* times, where *n* is the number of columns of *valArray*, given by `numel(valArray(1,:))`. The input *valArray* can be of any MATLAB® data type, including a character vector, cell array, or struct.

1:3 ⇔ [1, 2, 3]

111



Command Window

```
>> for i=1:3
     disp(i)
end

     1

     2

     3
```

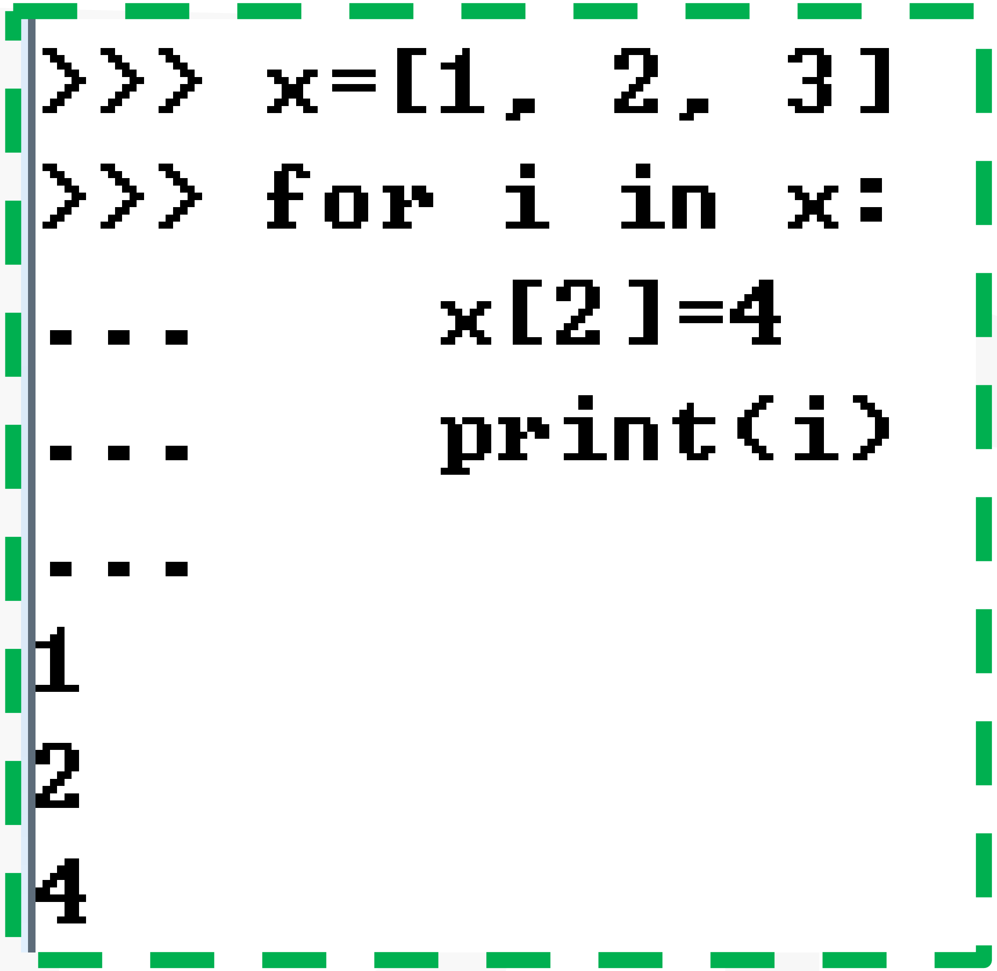
```
>> x=1:3
x =
     1     2     3
>> for i=x
    disp(i)
end
     1
     2
     3
```

```
>>> x=[1, 2, 3]
>>> for i in x:
...     print(i)
...
1
2
3
```

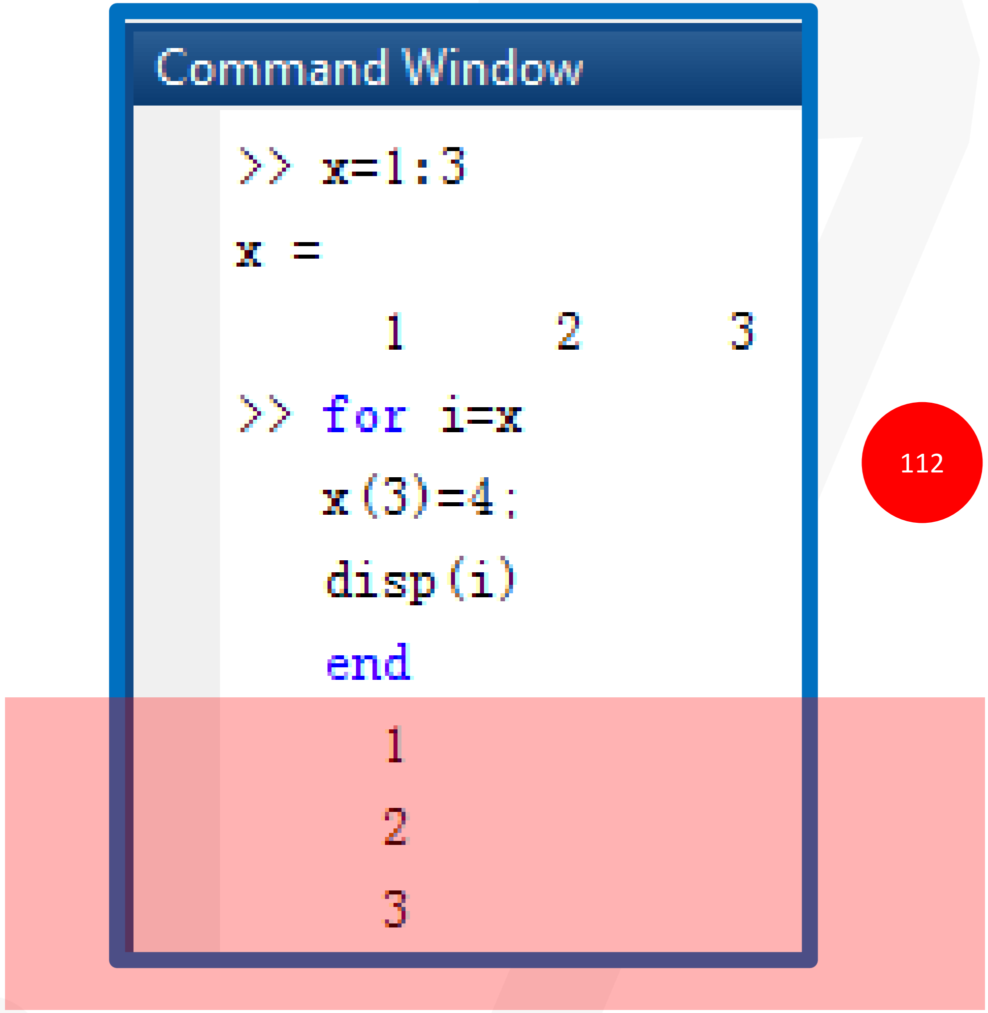
SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance



```
>>> x=[1, 2, 3]
>>> for i in x:
...     x[2]=4
...     print(i)
...
1
2
4
```



```
>> x=1:3
x =
     1     2     3
>> for i=x
   x(3)=4;
   disp(i)
   end
     1
     2
     3
```

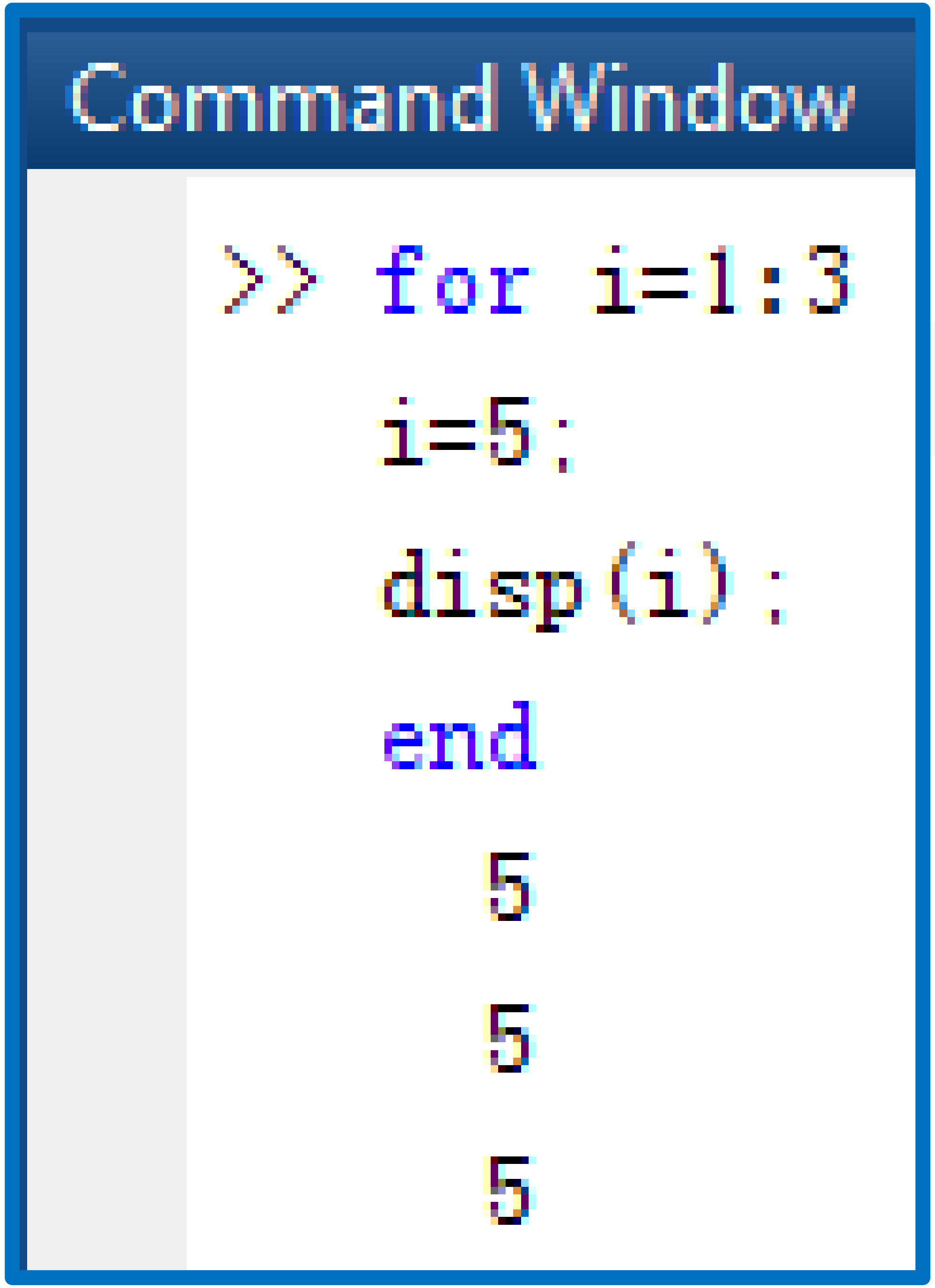
112

(Discussion Time)

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

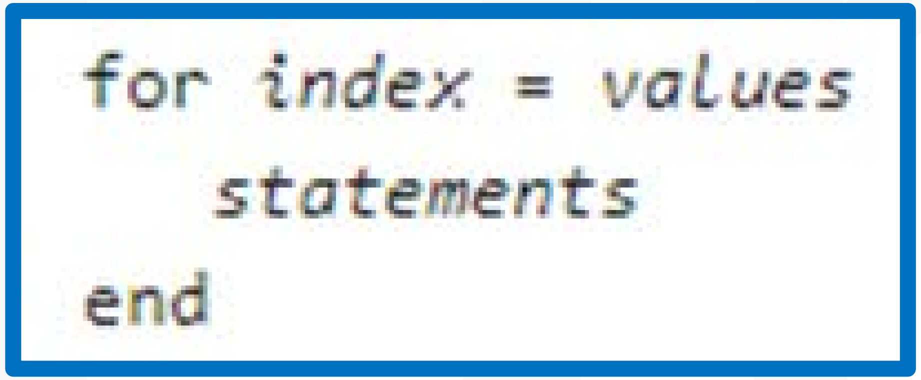
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance



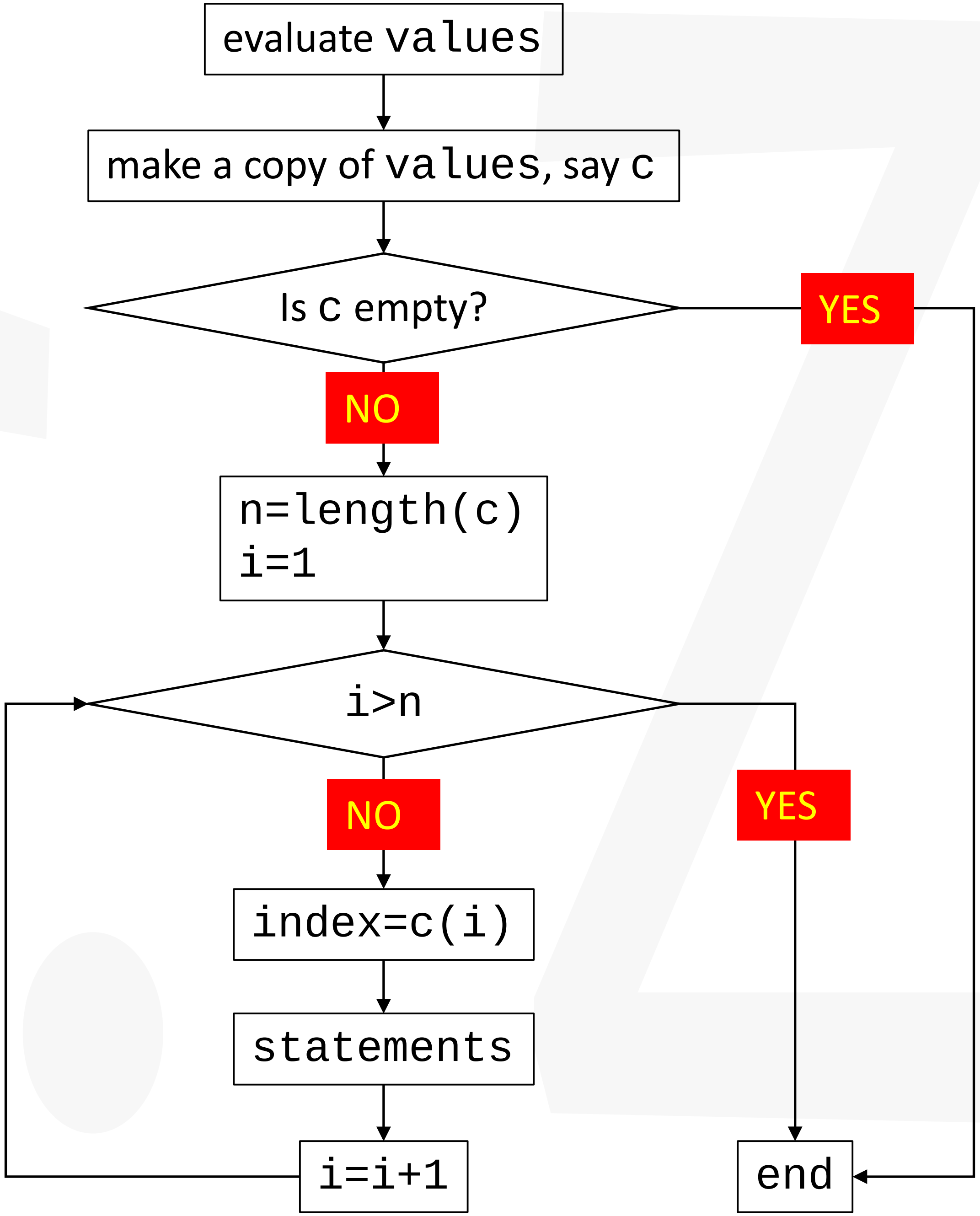
```
>> for i=1:3
i=5;
disp(i);
end
     5

     5

     5
```

112

(Discussion Time)



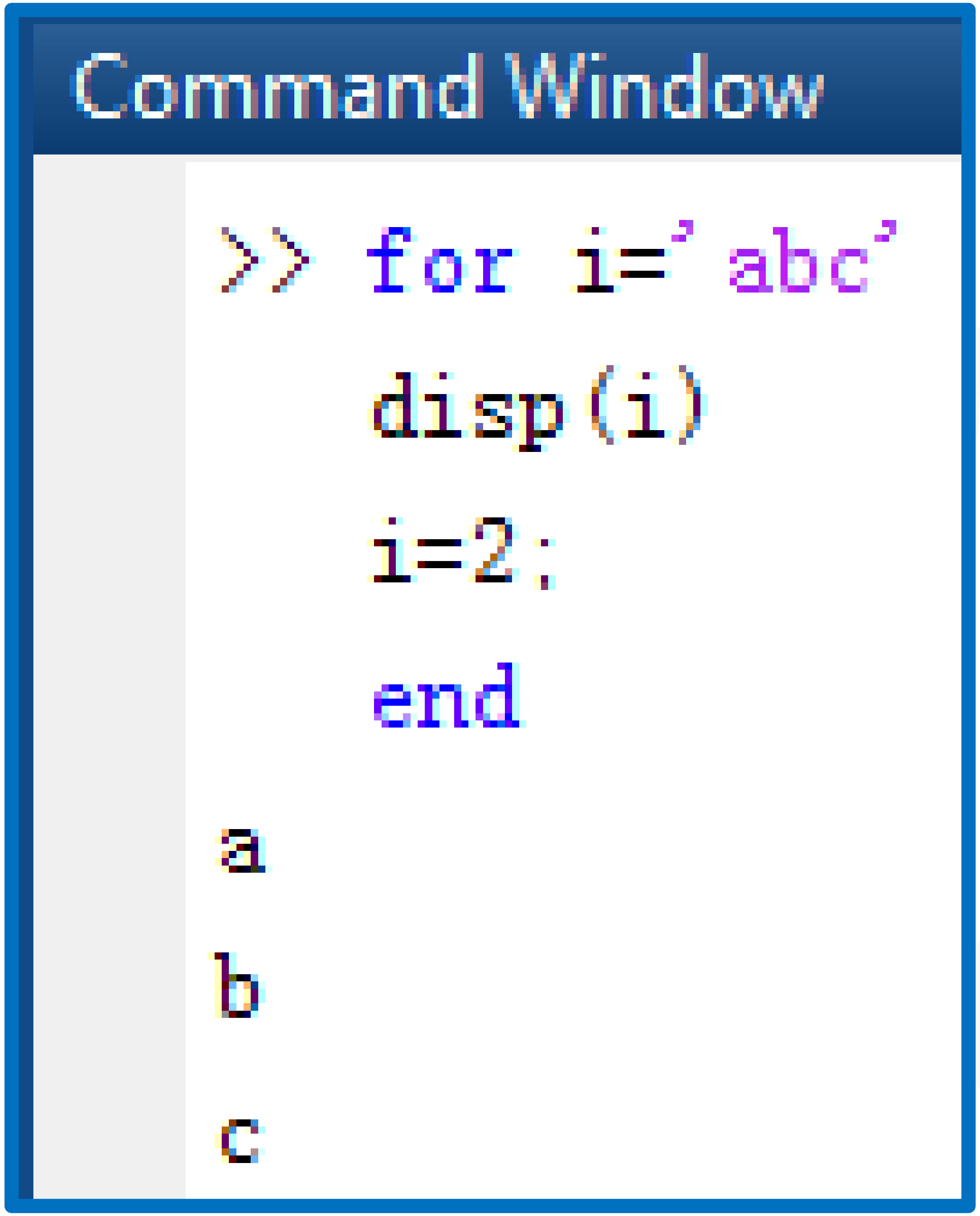
```
for index = values
    statements
end
```



QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Slide
142/160



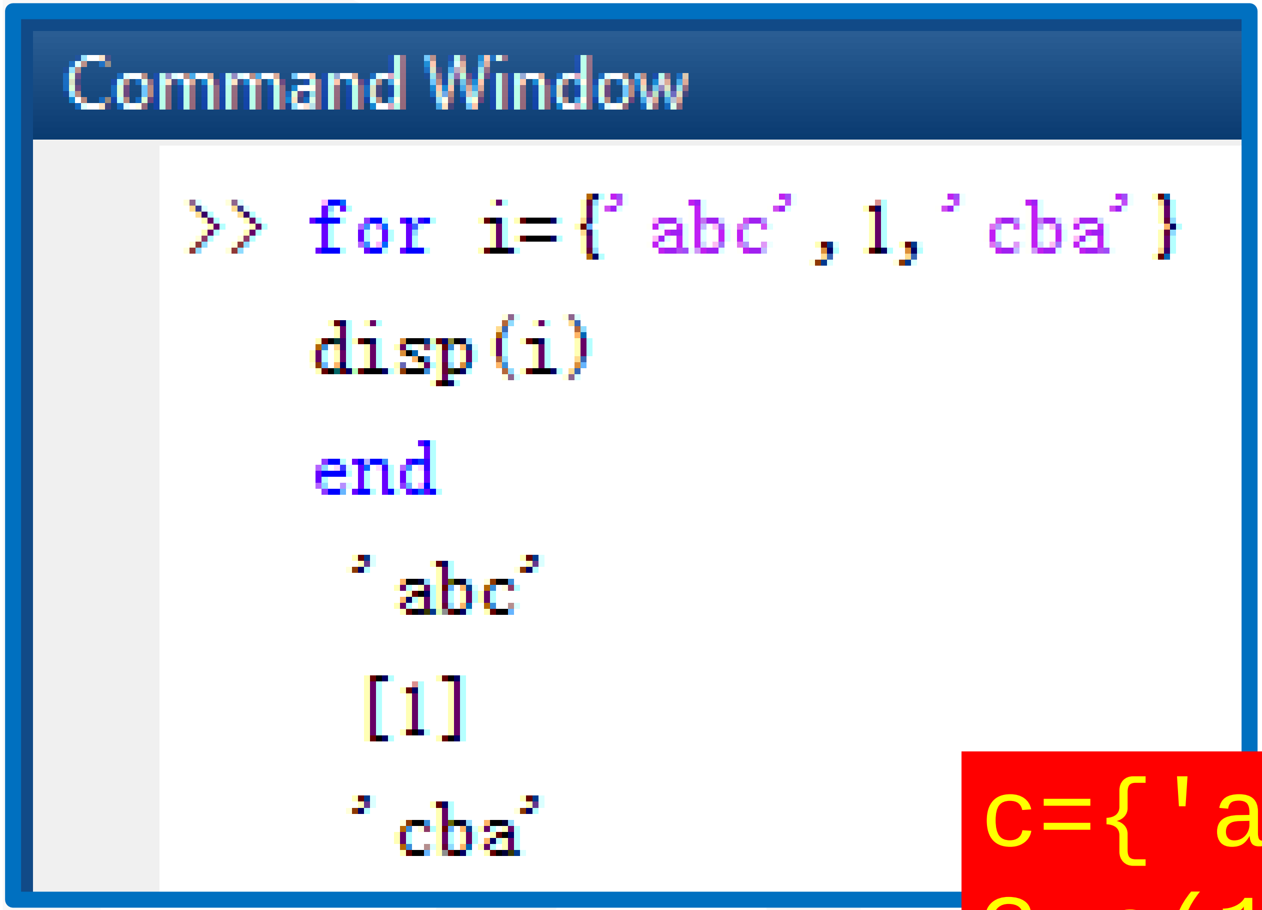
```
>> for i='abc'
    disp(i)
    i=2;
    end

a

b

c
```

112

SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance



```
>> for i={'abc',1,'cba'}
   disp(i)
   end
    'abc'
    [1]
    'cba'
```

```
c={'abc', 1, 'cba'}
? c(1)
? c(2)
? c(3)
```

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

3. Pandigital Formula

```
year = 2017
o=['+','-','*','/','']  # or o='+-*/_'
for o1 in o:
    for o2 in o:
        for o3 in o:
            for o4 in o:
                for o5 in o:
                    for o6 in o:
                        for o7 in o:
                            for o8 in o:
                                s=('1'+ o1 +
                                   '2'+ o2 +
                                   '3'+ o3 +
                                   '4'+ o4 +
                                   '5'+ o5 +
                                   '6'+ o6 +
                                   '7'+ o7 +
                                   '8'+ o8 + '9')
                                if eval(s)==year:
                                    print(s + '=', year, sep='')
print('Done!')
```

```
['a','b'] ⇔ {'a','b'}
```

```
for o1 in o:
⇕
for o1=o … end
```

```
'a' + 'b' ⇔ ['a','b']
```

```
eval(s) ⇔ eval(s)
```

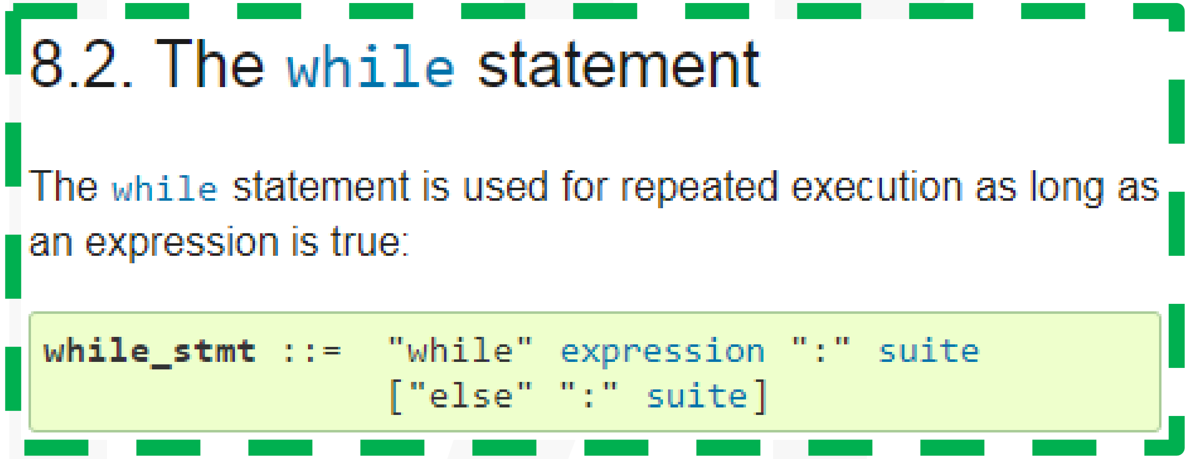
```
print('a', 2, sep='')
⇕
disp(['a', num2str(2)])
```

In-Class Exercise

15 min

Homework Q3

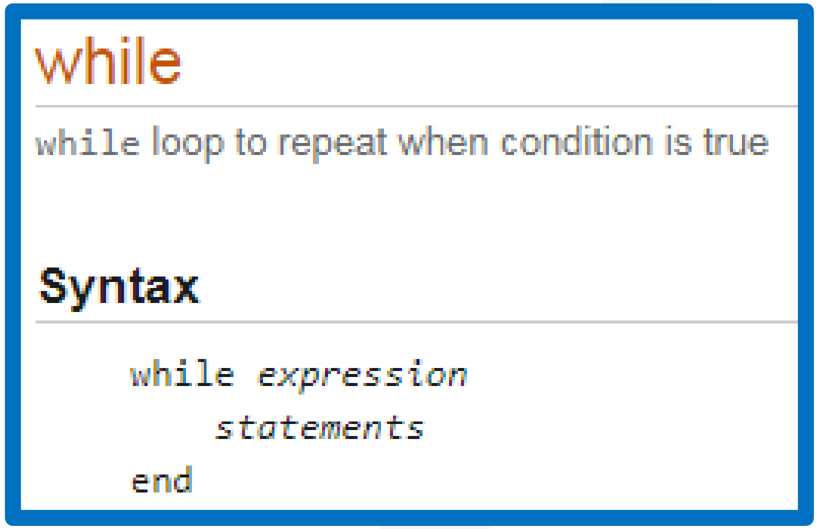
Loop Controls (2): while loop



8.2. The while statement

The while statement is used for repeated execution as long as an expression is true:

```
while_stmt ::=  "while" expression ":" suite
                ["else" ":" suite]
```



while

while loop to repeat when condition is true

Syntax

```
    while expression
        statements
    end
```

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

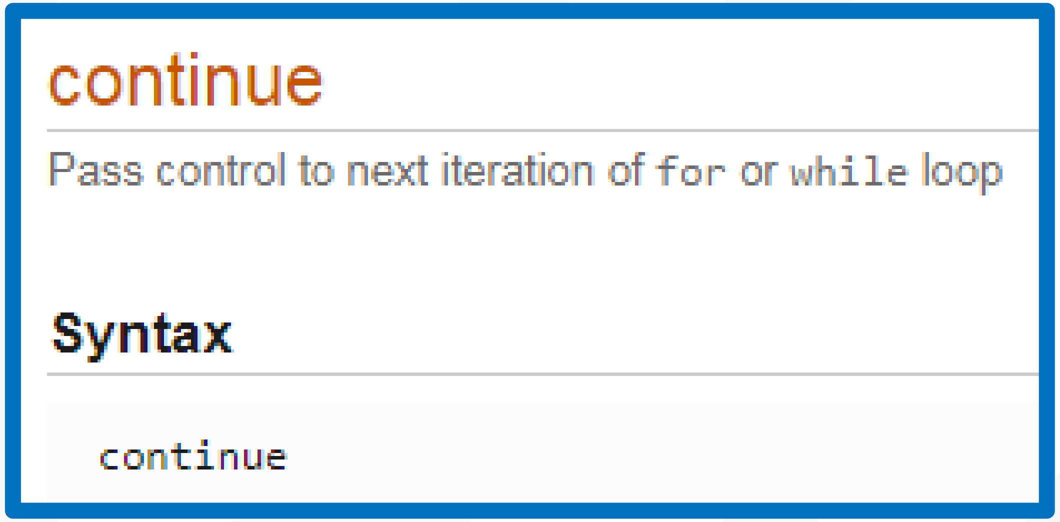
MATLAB `for` loop and `while` loop do not have the “`else`” clause.

break and continue

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

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

115

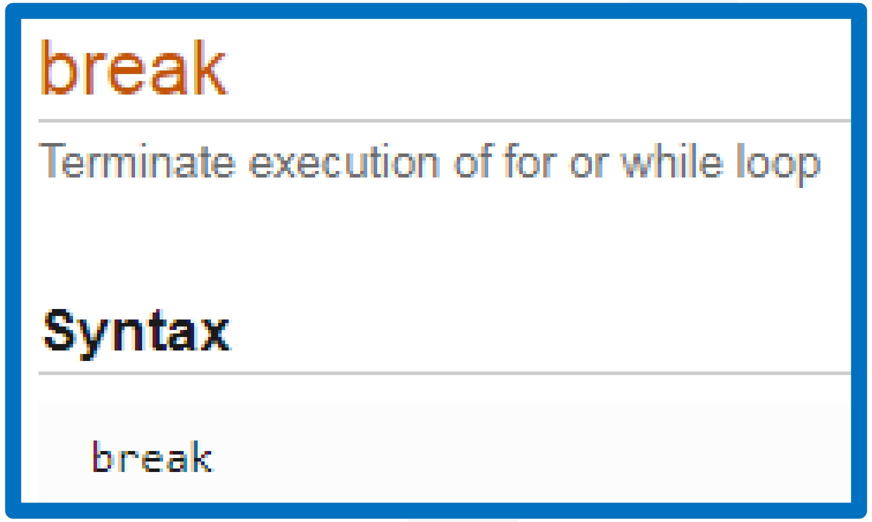


continue

Pass control to next iteration of `for` or `while` loop

Syntax

```
continue
```



break

Terminate execution of for or while loop

Syntax

```
break
```

(Dr. Z: It is much easier than Python.)

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

116

MATLAB, as all other programming languages, allows “recursive function definition”.

4. Sudoku Solver

```
def same_row(i,j): return (i//9)==(j//9)
def same_col(i,j): return (i%9)==(j%9)
def same_block(i,j): return ((i//27)==(j//27)) and (((i%9)//3)==((j%9)//3))
def r(s):
    i=s.find('0')
    if i==-1:
        print(s)
    else:
        excluded_numbers={s[j] for j in range(81) if same_row(i,j)
                                                  or same_col(i,j)
                                                  or same_block(i,j)}
        for m in set('123456789')-excluded_numbers:
            r(s[:i]+m+s[i+1:])

s=('390060807' + '020030050' + '000005096' +
   '900502400' + '000000000' + '003907002' +
   '810600000' + '030050080' + '502090043')

print(s)
print(r(s))
```

Homework Q4

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

`find`, indexing, range(81)⇨ i,j∈{0,1,...,80} ⇔`strfind`, 1:81, indexing ⇨ i,j∈{1,2,...,81}

`same_row(i,j), same_col(i,j), same_block(i,j)`
⇳
`same_row(i-1,j-1), same_col(i-1,j-1), same_block(i-1,j-1)`

function `def` ⇔ M-File function

`i=s.find('0')` ⇔ `c=strfind(s,'0')` and if not empty use `i=c(1)`

`r==-1` ⇔ `isempty(r)`

`if…else…` ⇔ `if…else…end`

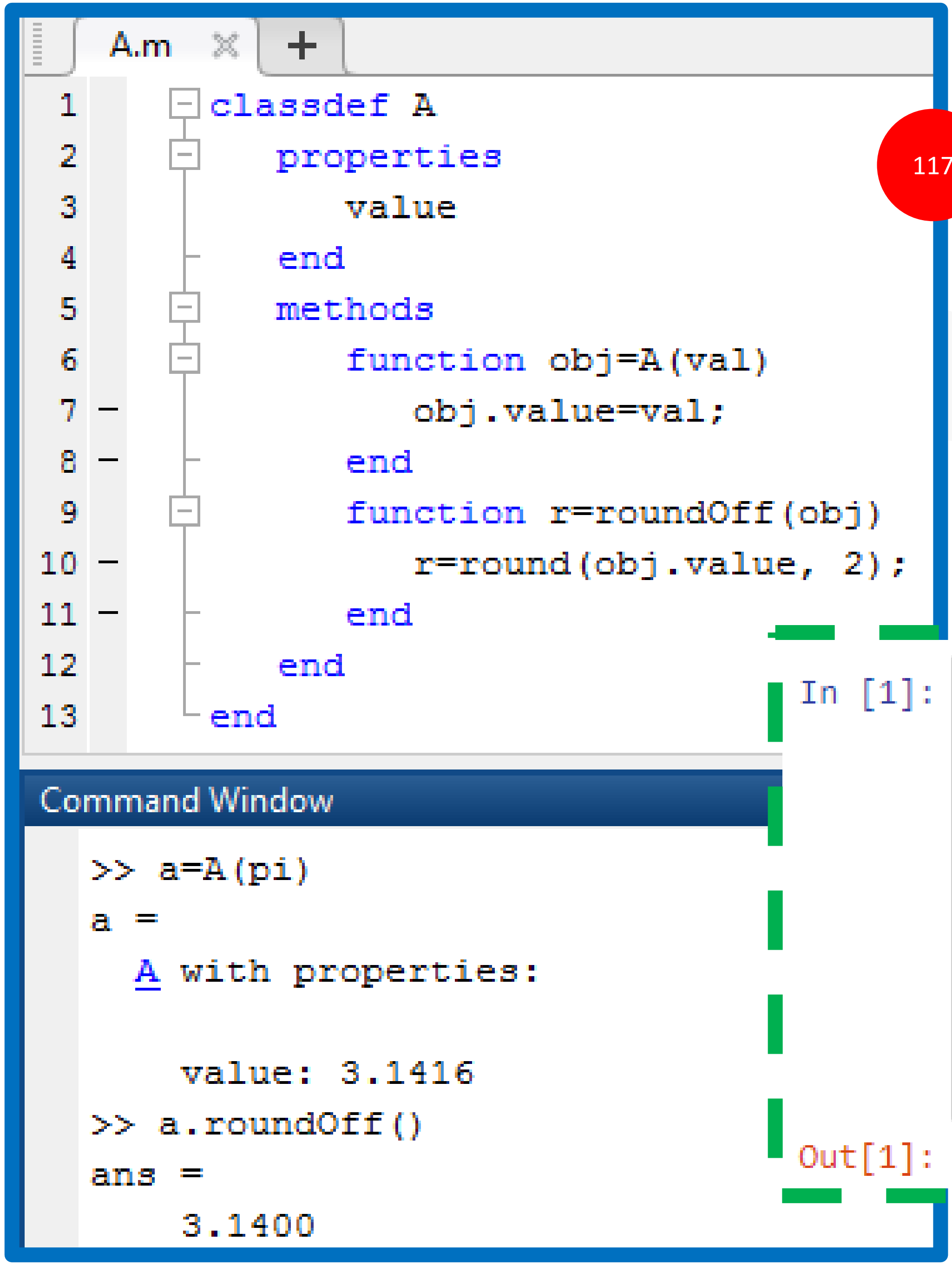
set comprehension ⇔ loop + add item + set method `unique`

set difference ⇔ set method `setdiff`

string concatenation using + ⇔ string concatenation using [ ]

`and`, `or` ⇔ &&, | |

`a%b`, `a//b` ⇔ `mod(a,b)`, `floor(a/b)`



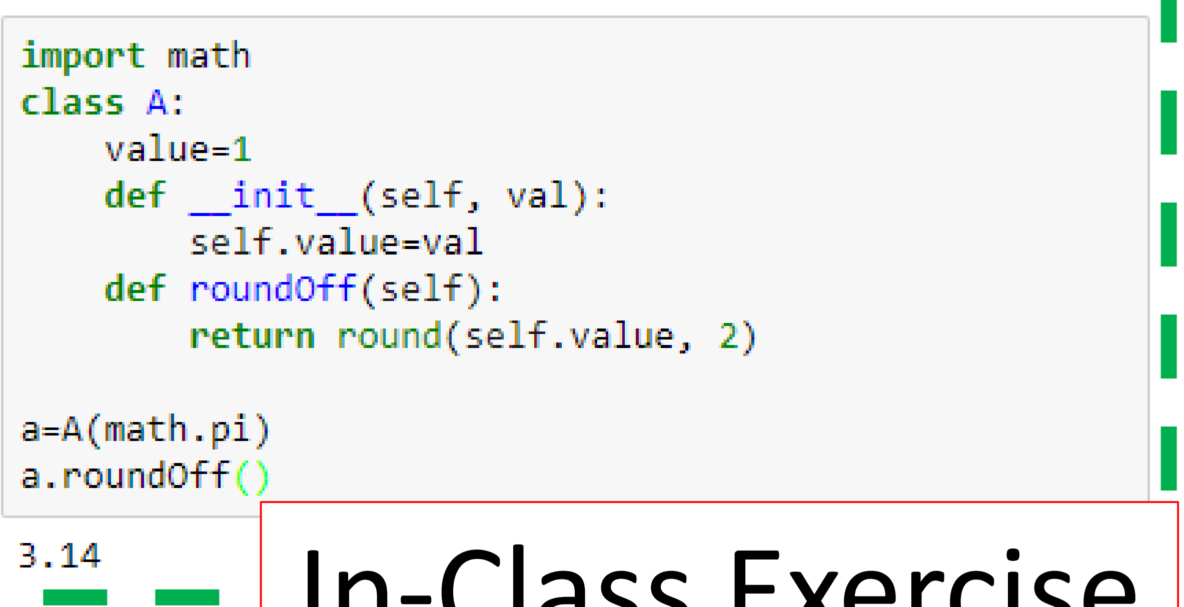
```
classdef A
    properties
        value
    end
    methods
        function obj=A(val)
            obj.value=val;
        end
        function r=roundOff(obj)
            r=round(obj.value, 2);
        end
    end
end
```

```
>> a=A(pi)
a =
  A with properties:

    value: 3.1416
>> a.roundOff()
ans =
    3.1400
```

117

class

All properties (corresponding to Python’s class attributes and instance attributes) have to be class properties. They are not “shared”. We cannot add extra properties to an “instance” dynamically. (hahaha…You know what it means. Right?)



```
In [1]: import math
        class A:
            value=1
            def __init__(self, val):
                self.value=val
            def roundOff(self):
                return round(self.value, 2)

        a=A(math.pi)
        a.roundOff()

Out[1]: 3.14
```

In-Class Exercise

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

MATLAB does not have class objects as Python's. The class definition is used to define instance properties/methods.

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

5. European Call Option Object

Convert the following Python code to MATLAB. (Class to Class, Method to Method, etc.)

Homework Q5

```
from math import log, sqrt, exp
from scipy import stats
class call_option(object):
    def __init__(self, S0, K, T, r, sigma):
        self.S0 = float(S0)
        self.K = K
        self.T = T
        self.r = r
        self.sigma = sigma
    def value(self):
        d1 = ((log(self.S0 / self.K) + (self.r + 0.5 * self.sigma ** 2) * self.T)
            / (self.sigma * sqrt(self.T)))
        d2 = ((log(self.S0 / self.K) + (self.r - 0.5 * self.sigma ** 2) * self.T)
            / (self.sigma * sqrt(self.T)))
        value = (self.S0 * stats.norm.cdf(d1, 0.0, 1.0)
            - self.K * exp(-self.r * self.T) * stats.norm.cdf(d2, 0.0, 1.0))
        return value
    def vega(self):
        d1 = ((log(self.S0 / self.K) + (self.r + 0.5 * self.sigma ** 2) * self.T)
            / (self.sigma * sqrt(self.T)))
        vega = self.S0 * stats.norm.cdf(d1, 0.0, 1.0) * sqrt(self.T)
        return vega
    def imp_vol(self, C0, sigma_est=0.2, it=100):
        option = call_option(self.S0, self.K, self.T, self.r, sigma_est)
        for i in range(it):
            option.sigma -= (option.value() - C0) / option.vega()
        return option.sigma

o=call_option(100., 105., 1.0, 0.05, 0.2)
print(o.value())
print(o.vega())
print(o.imp_vol(C0=value))
```

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Some Hints:

log, **, sqrt ⇨ log, ^, sqrt

stats.norm.cdfqrt ⇨ normcdf

data/function attributes ⇨ properties, methods

self ⇨ obj

__init__() ⇨ obj=ClassName()

Return Values ⇨ Return Values

Line Joining ⇨ Line Joining

Default Value ⇨ Default Value

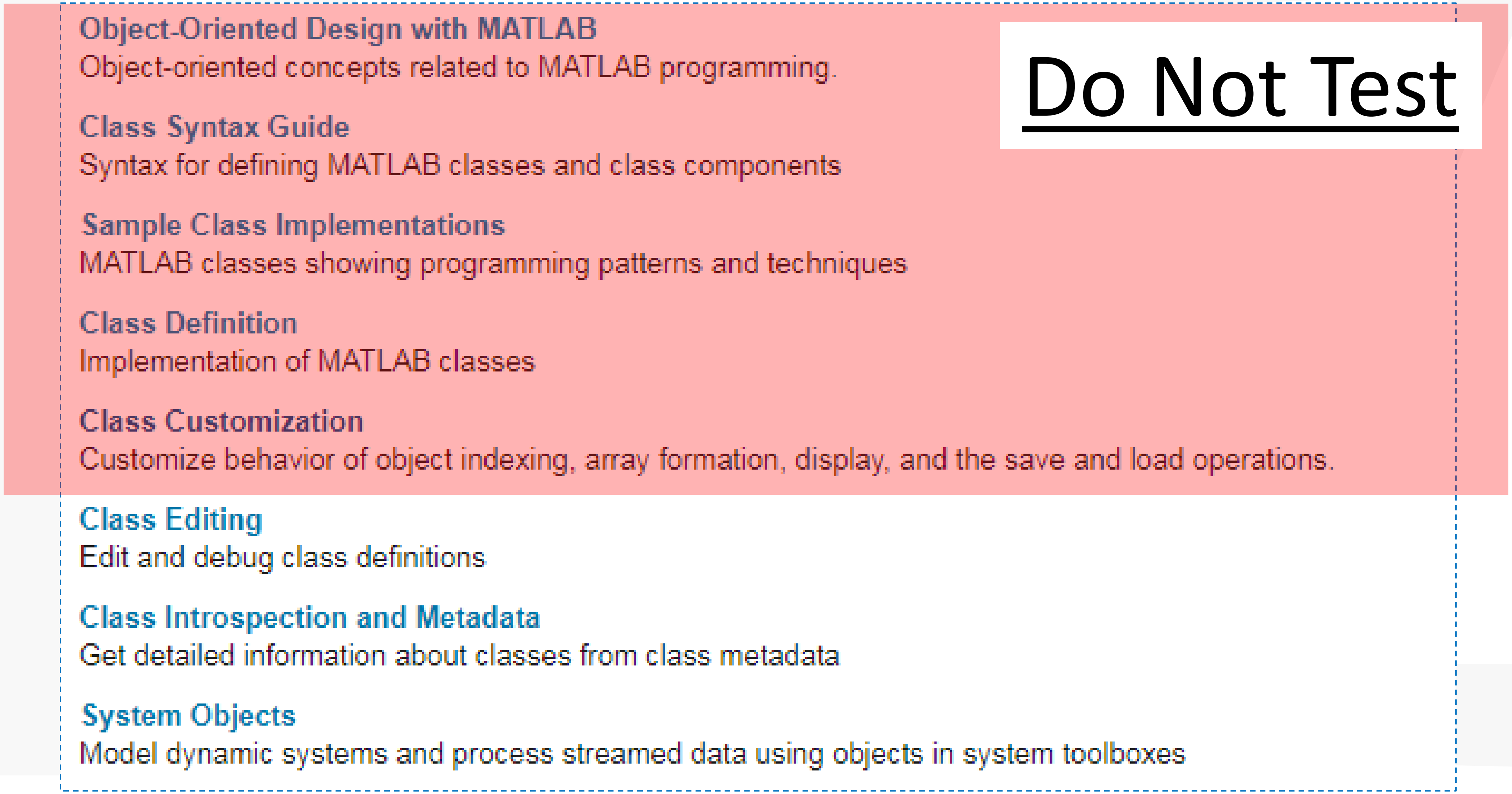
Code Block Indentation ⇨ ”end”

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

https://www.mathworks.com/help/matlab/object-oriented-programming.html



Do Not Test

Object-Oriented Design with MATLAB
Object-oriented concepts related to MATLAB programming.

Class Syntax Guide
Syntax for defining MATLAB classes and class components

Sample Class Implementations
MATLAB classes showing programming patterns and techniques

Class Definition
Implementation of MATLAB classes

Class Customization
Customize behavior of object indexing, array formation, display, and the save and load operations.

Class Editing
Edit and debug class definitions

Class Introspection and Metadata
Get detailed information about classes from class metadata

System Objects
Model dynamic systems and process streamed data using objects in system toolboxes

QF666 Programming and Computational Finance

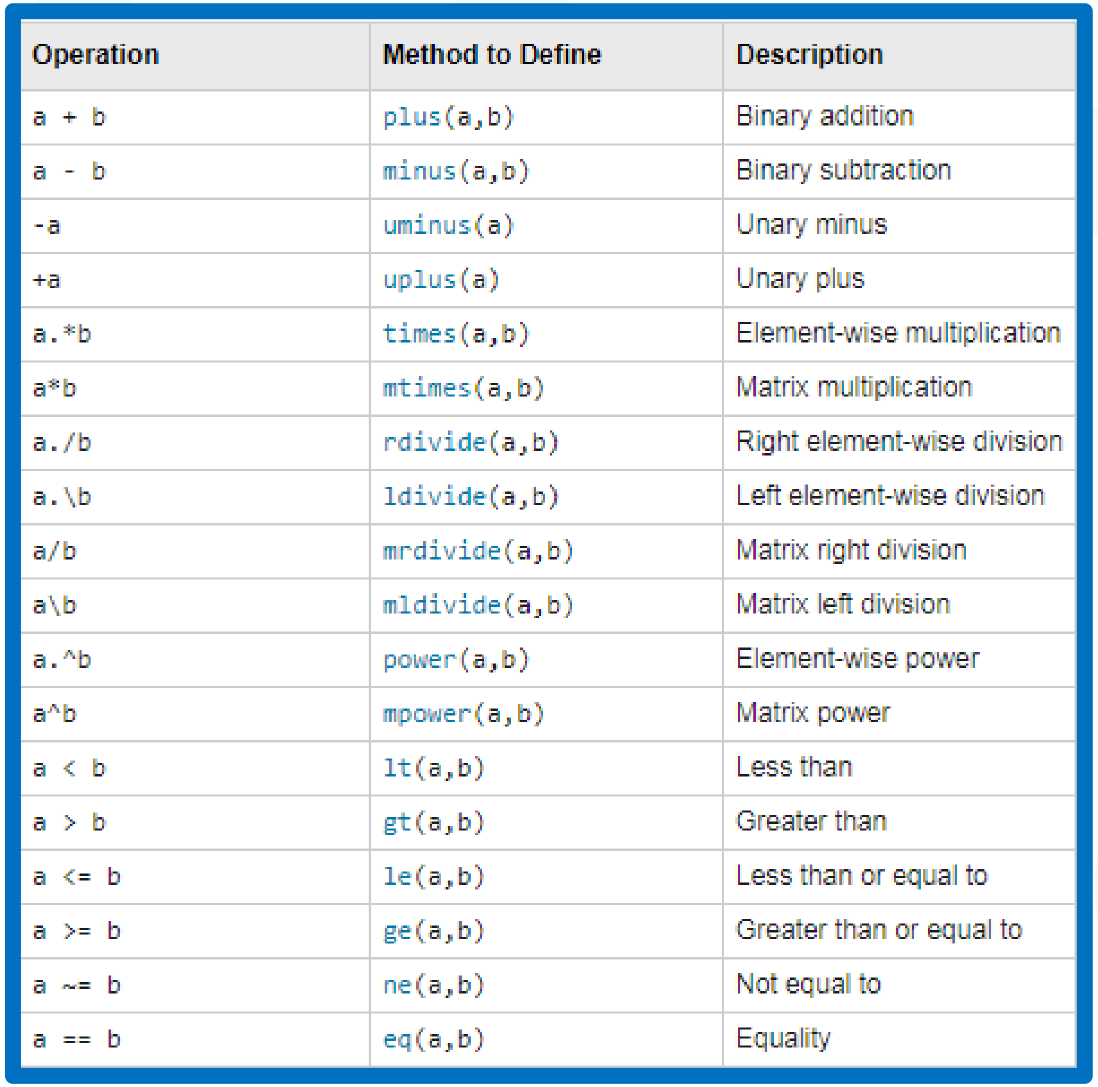
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

Operator Overloading

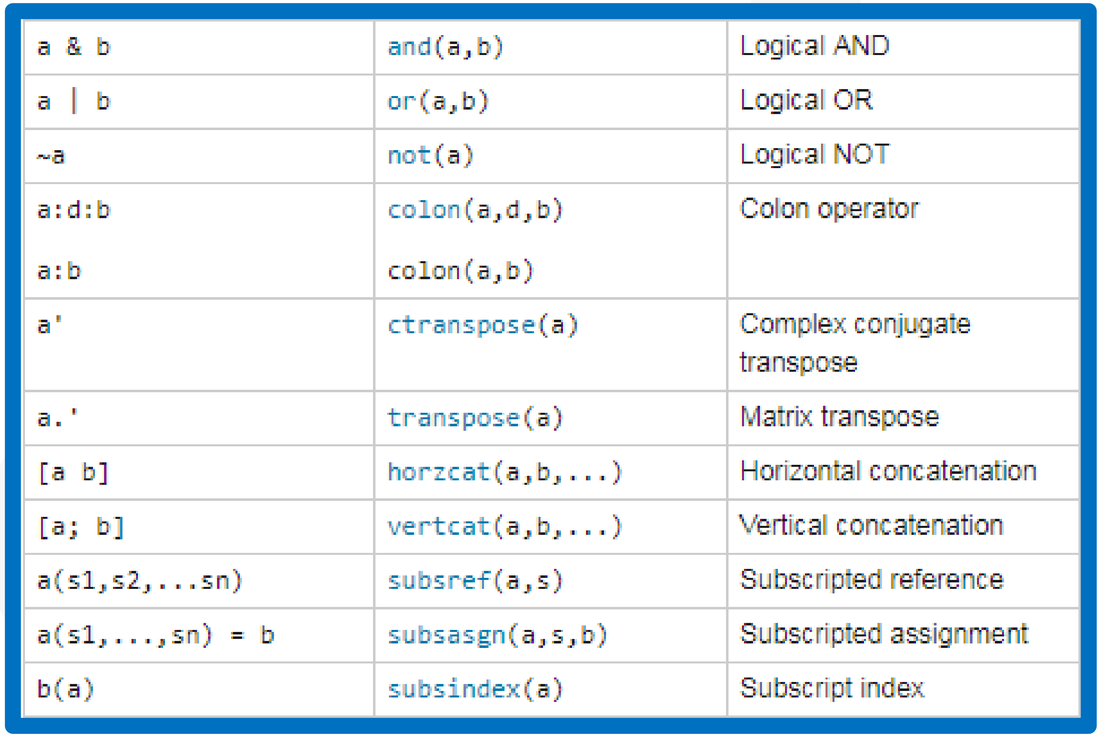
https://www.mathworks.com/help/matlab/matlab_oop/implementing-operators-for-your-class.html

Method to Define

Do Not Test



Operation	Method to Define	Description
`a + b`	`plus(a,b)`	Binary addition
`a - b`	`minus(a,b)`	Binary subtraction
`-a`	`uminus(a)`	Unary minus
`+a`	`uplus(a)`	Unary plus
`a.*b`	`times(a,b)`	Element-wise multiplication
`a*b`	`mtimes(a,b)`	Matrix multiplication
`a./b`	`rdivide(a,b)`	Right element-wise division
`a.\b`	`ldivide(a,b)`	Left element-wise division
`a/b`	`mrdivide(a,b)`	Matrix right division
`a\b`	`mldivide(a,b)`	Matrix left division
`a.^b`	`power(a,b)`	Element-wise power
`a^b`	`mpower(a,b)`	Matrix power
`a < b`	`lt(a,b)`	Less than
`a > b`	`gt(a,b)`	Greater than
`a <= b`	`le(a,b)`	Less than or equal to
`a >= b`	`ge(a,b)`	Greater than or equal to
`a ~= b`	`ne(a,b)`	Not equal to
`a == b`	`eq(a,b)`	Equality



`a & b`	`and(a,b)`	Logical AND
`a | b`	`or(a,b)`	Logical OR
`~a`	`not(a)`	Logical NOT

`a:d:b`
`a:b`
	`colon(a,d,b)`
	`colon(a,b)`
		Colon operator

`a'`	`ctranspose(a)`	Complex conjugate transpose
`a.'`	`transpose(a)`	Matrix transpose
`[a b]`	`horzcat(a,b,...)`	Horizontal concatenation
`[a; b]`	`vertcat(a,b,...)`	Vertical concatenation
`a(s1,s2,...sn)`	`subsref(a,s)`	Subscripted reference
`a(s1,...,sn) = b`	`subsasgn(a,s,b)`	Subscripted assignment
`b(a)`	`subsindex(a)`	Subscript index

- ✓ Base Classes

https://fr.mathworks.com/help/matlab/matlab_oop/subclass-syntax.html

- ✓ Private Attributes

https://www.mathworks.com/help/matlab/matlab_oop/property-attributes.html

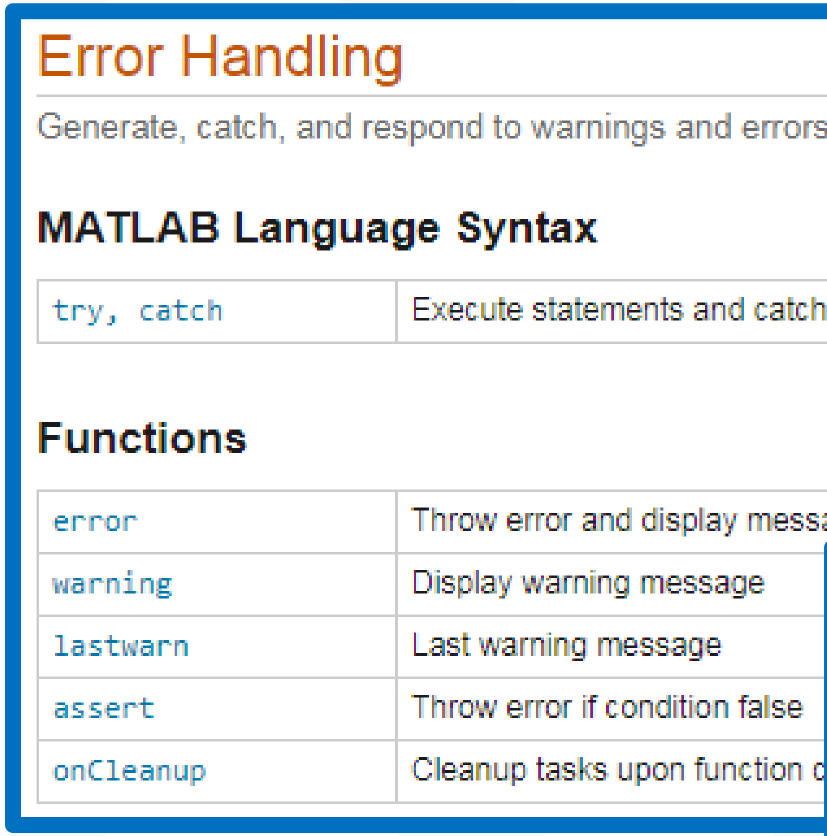
- ✓ Static Methods

https://www.mathworks.com/help/matlab/matlab_oop/method-attributes.html

Do Not Test

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

MATLAB Error Handling



Error Handling

R2018a

Generate, catch, and respond to warnings and errors

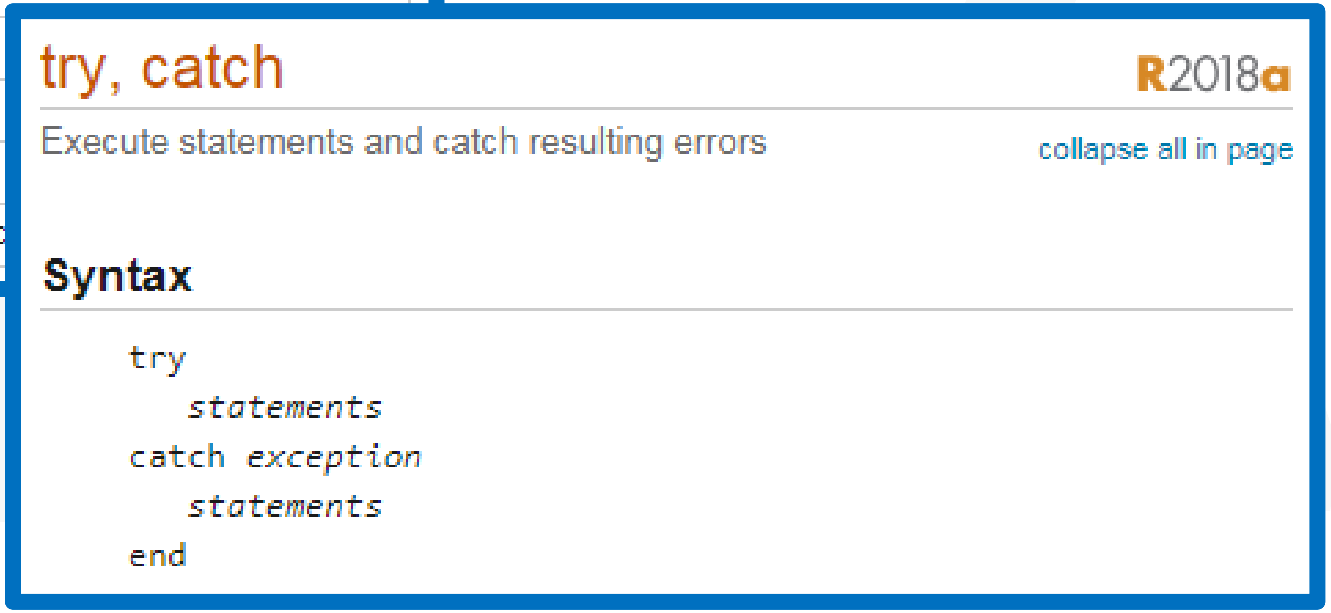
MATLAB Language Syntax

`try, catch`	Execute statements and catch resulting errors

Functions

`error`	Throw error and display message
`warning`	Display warning message
`lastwarn`	Last warning message
`assert`	Throw error if condition false
`onCleanup`	Cleanup tasks upon function c

Do Not Test



try, catch

R2018a

Execute statements and catch resulting errors

Syntax

```
try
   statements
catch exception
   statements
end
```

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

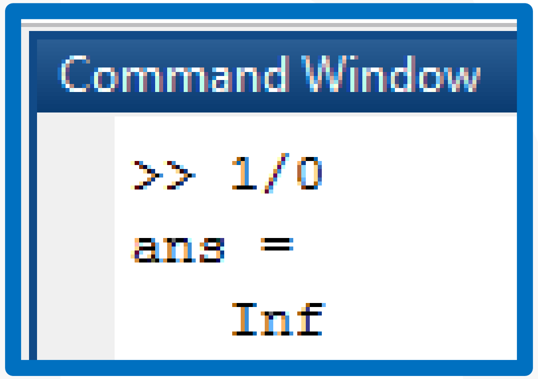
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

SMU SINGAPORE MANAGEMENT UNIVERSITY

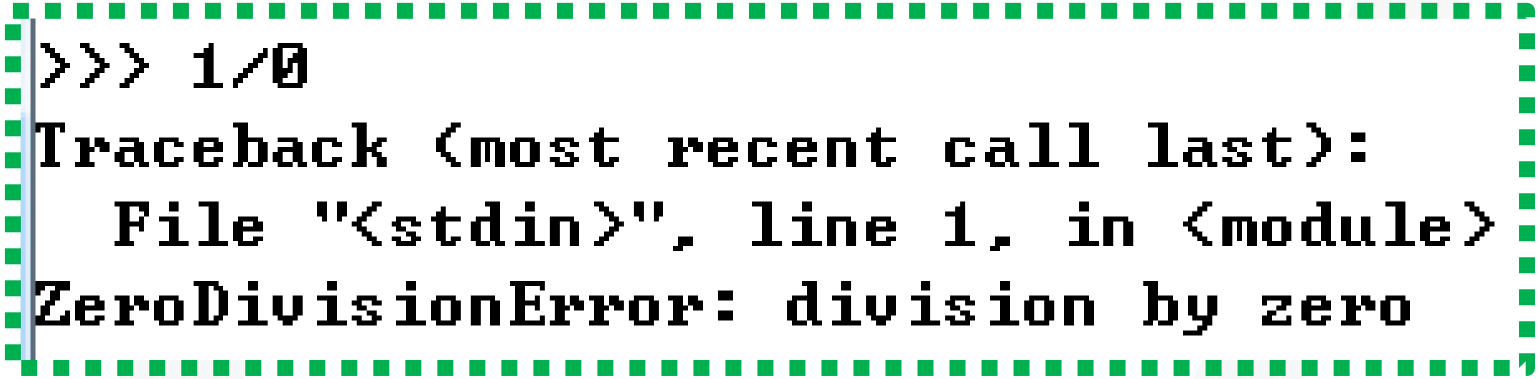
QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Do Not Test

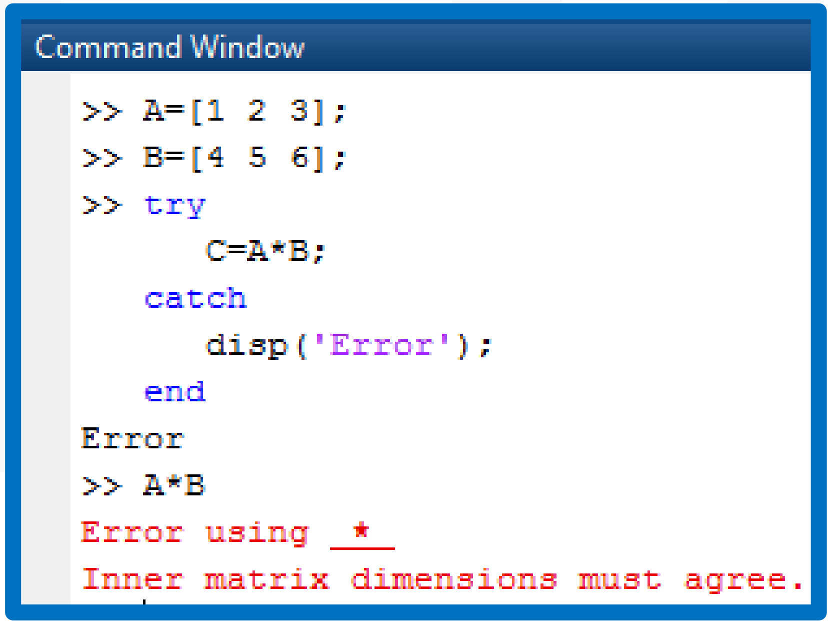


```
>> 1/0
ans =
    Inf
```

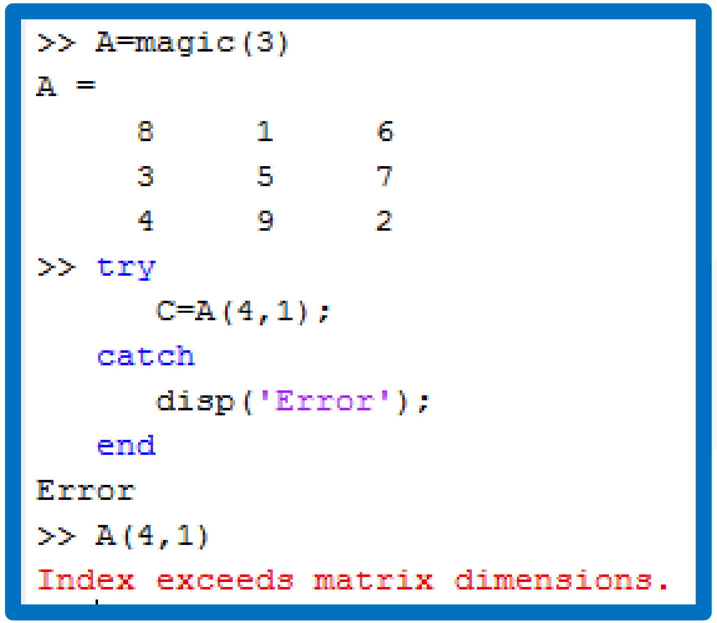


```
>>> 1/0
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ZeroDivisionError: division by zero
```

In MATLAB, 1/0 will not raise any erro.



```
>> A=[1 2 3];
>> B=[4 5 6];
>> try
       C=A*B;
   catch
       disp('Error');
   end
Error
>> A*B
Error using  *
Inner matrix dimensions must agree.
```



```
>> A=magic(3)
A =
     8     1     6
     3     5     7
     4     9     2
>> try
       C=A(4,1);
   catch
       disp('Error');
   end
Error
>> A(4,1)
Index exceeds matrix dimensions.
```

...

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

end

Next, …