

COMP1022Q  
Introduction to Computing with Excel VBA

# VBA Basics

Gibson Lam and David Rossiter

# Outcomes

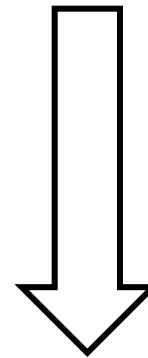
- After completing this presentation, you are expected to be able to:
  1. Use the MsgBox command to show a message box using VBA code
  2. Understand the idea of using variables
  3. Write VBA code to process variables
  4. Use the InputBox command to ask for input from the user

# VBA Code Goes In Subroutines

- Previously you learned that you can write VBA code in a macro
- (Usually, a macro is not very big, and usually a macro is triggered by a key combination)
- It doesn't matter if you are making a macro or something else, the code goes inside a subroutine
- When you run the VBA code inside the subroutine the code is executed from top to bottom

Sub Hello()

...code...



...code...

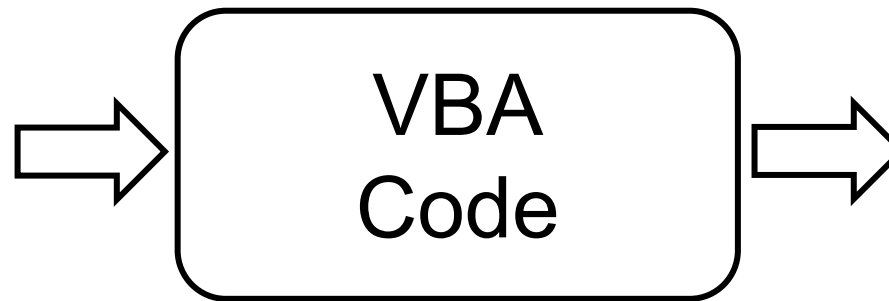
End Sub

*Inside a  
subroutine  
the code  
runs from  
top to  
bottom*

# Input and Output in This Presentation

## Inputs

*Inputs can  
be entered  
in a small  
window  
(`InputBox`)*

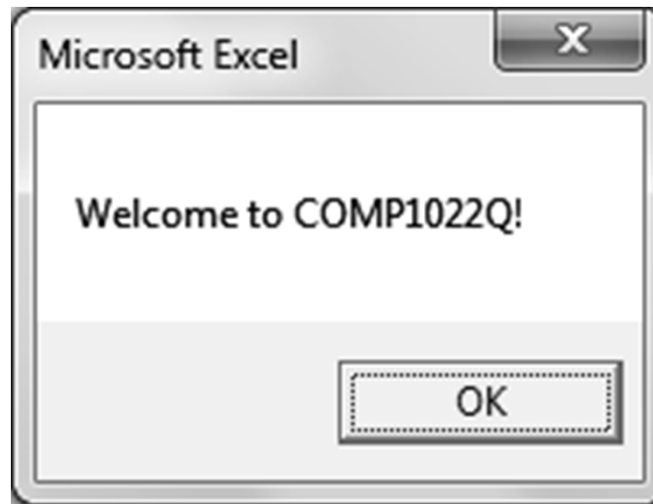


## Outputs

*Results can  
be shown  
in a small  
window  
(`MsgBox`)*

# Using MsgBox

- MsgBox is a useful command to show a few words in a small window
- Here is an example:

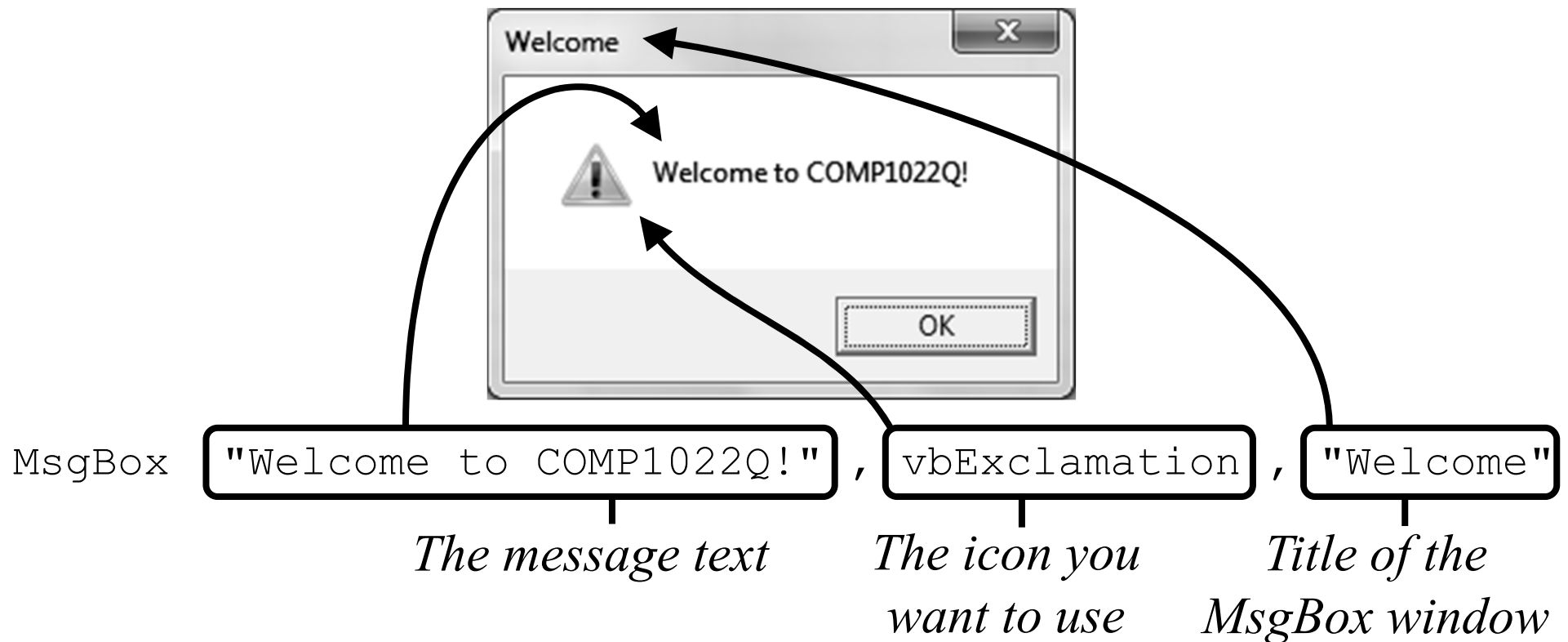


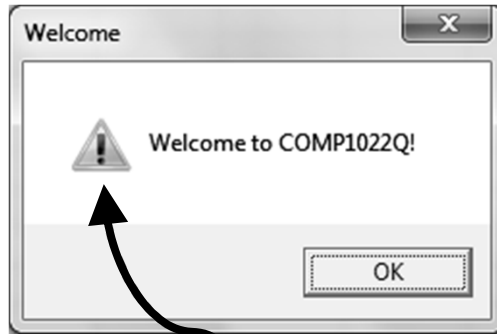
- Here is the VBA code which generates the message:

```
MsgBox "Welcome to COMP1022Q!"
```





# Changing the MsgBox Appearance

- If you want to, you can change the title and icon of a message box
- Here is an example:





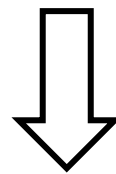
# Some Icons You Can Use

What you see	The name you write in the code
	<code>vbExclamation</code>
	<code>vbCritical</code>
	<code>vbInformation</code>
	<code>vbQuestion</code>

# Using the Underscore ‘\_’

- Sometimes a line of VBA code can be quite long and that might make it hard to read
- You can break down a line of code using an underscore ‘\_’ like this:

```
MsgBox "Welcome to COMP1022Q!", vbInformation, "Welcome"
```



*Use an underscore to put one  
line of code into more than 1 line*

```
MsgBox "Welcome to COMP1022Q!", _  
      vbInformation, "Welcome"
```



# More MsgBox Examples

MsgBox "I lost my mobile phone! :(" , \_

vbCritical , \_

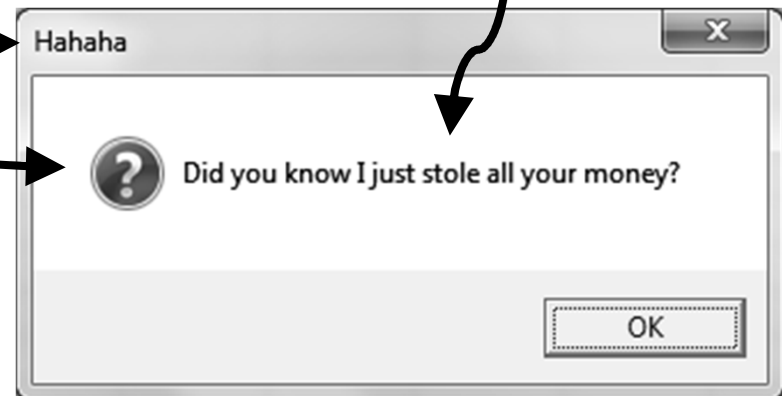
"Oh no!"



MsgBox "Did you know I just stole all your money?" , \_

vbQuestion , \_

"Hahaha"



# Variables

- You can think of a variable as a box
- There are different types of box which hold different things
- For example, you can use a box to hold a number or a piece of text
- You can create as many variables as you like in VBA code but each variable must be given a unique name

<i>David</i>
Name

# Making and Using a Variable

- For example, you can make a variable called *Name* which holds a string (a piece of text) using this code:

```
Dim Name As String
```

- You can use the variable to hold some text using this code:

```
Name = "David"
```

*David*

Name

- You can then show the content of the variable using a message box:

```
MsgBox Name
```



# Things Similar to Cell Formulas

- Previously you learned things you can do in cell formulas
- You can do the same things in VBA (sometimes you have to type it a bit differently)
- We will have a quick look at the following things:
  - Arithmetic + - \* / ^
  - String concatenation
  - Comparing things
  - Using logic

# Arithmetic

- These have the same effect that we saw before in cell formulas

^	Power
* and /	Multiplication and division
+ and -	Addition and subtraction

## String Concatenation

&	Concatenation (=gluing) two things together e.g. "Strong" & "Typhoon" gives you "StrongTyphoon"
---	---

# Comparing Things

- Like we saw in cell formulas, these return a result of *True* or *False* after comparing two things

=	Equal to
<>	Not equal to
<	Smaller than
<=	Smaller than or equal to
>	Larger than
>=	Larger than or equal to

# Using Logic

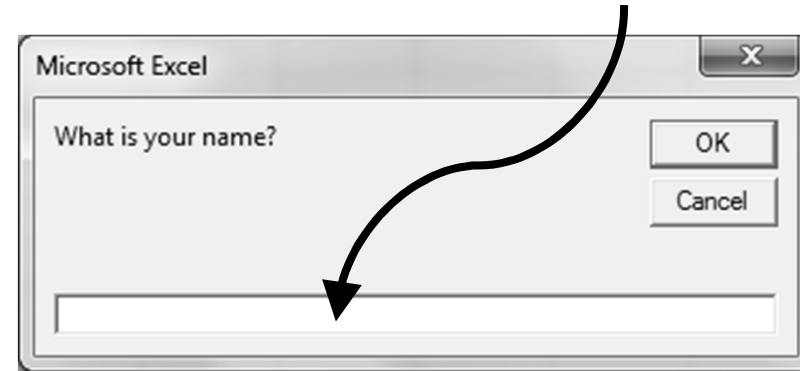
- And, Or and Not give the same results that we saw when we looked at them in cell formulas

And	And returns <i>True</i> if the inputs are <i>True</i> ; otherwise it returns <i>False</i>
Or	Or returns <i>False</i> if the inputs are <i>False</i> ; otherwise it returns <i>True</i>
Not	Not returns <i>False</i> if the input is <i>True</i> , and it returns <i>True</i> if the input is <i>False</i>

# Using InputBox

*Input the text here*

- InputBox displays a message box where you can enter a value or a message



- The user types something, and presses OK
- After that, we need to store what the user has typed, then we can do something with it, like this:

*We store the  
result of the  
input box in  
a variable*

```
Dim Name As String
```

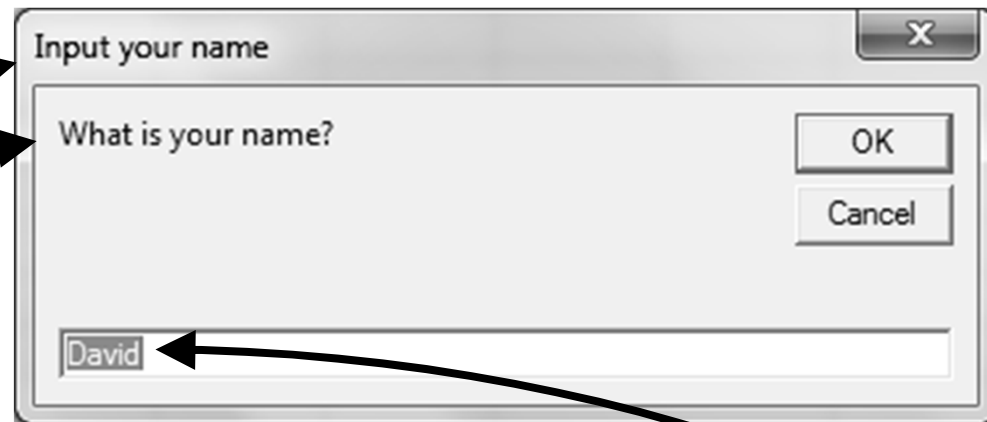
*We store the  
result of the  
input box in  
a variable* → Name = InputBox("What is your name?")



# Changing the InputBox Appearance

- You can change the title and default text
- The default text is shown when the input box appears, before the user does anything

- For example:



*In this example, the default text is 'David'*

- The VBA code:

```
Dim Name As String
```

```
Name = InputBox( "What is your name?", _  
                  "Input your name", "David" )
```

*Default text*

# Using Both InputBox and MsgBox

- We have looked at `MsgBox` and `InputBox`
- The following example uses both
- First, it asks for a name using an input box
- Then it shows a conclusion using a message box

# Example of InputBox and MsgBox

```
Sub Example()
```



*This VBA code is created as  
a macro called Example()*

```
Dim Name As String
```

```
Name = InputBox("What is your name?", _  
                "Input your name", "David")
```

*Putting strings  
together*

*The default text*

```
MsgBox Name & " is the best name!", , _
```

**"Conclusion"**

*The title of the  
MsgBox window*

```
End Sub
```

*In this example we don't  
ask for any particular  
icon, so none is shown*

# Example Execution

- When the macro is triggered Excel will show the input box

