# Visual Basic for Applications

## Programming

Damiano SOMENZI

School of Economics and Management

Advanced Computer Skills

damiano.somenzi@unibz.it

Week 2

# Outline

1. Function
    - Function Declaration
    - User Defined Functions

## Why Functions?

```
Function Procedure getCalories (food, quantity) {
   food = “bread” → CALORIES = 0.01 * quantity
   food = “egg” → CALORIES = 0.02 * quantity
   food = “lettuce” → CALORIES = 0.03 * quantity
   food = “not in the list ” → CALORIES = 0
Return CALORIES }


       ......
    Read food, quantity
    Calories = getCalories(food, quantity)
    write Calories
       ......
    Read food, quantity
    Calories = getCalories(food, quantity)
    write Calories
       ......
```

“Read *food, quantity*” reads the food and the quantity entered by the user

“Write Calories” displays the food calories in a window

# Programming Language

## Function Procedure

### More examples

| **Function** | **Expression** |
|---|---|
| $f(x,y) \rightarrow \frac{x}{y}$ | $v = 2 * f(3.2, 5.6)$ |
| $f(x) \rightarrow x^{-1}$ | $f(4) = 0.25$ |
| netPrice(price, VAT) → $\frac{price}{(1+VAT)}$ | Total = netPrice(12000, 0.21) + SubTot |
| EuroToPound(euro) → $euro * 0.80$ | Balance = EuroToPound(80000) + 500 |
| today() → ”Today is ” & Date() | MsgBox (today()) |

# Outline

1. Function
   - Function Declaration
   - User Defined Functions

# Programming Language

## Function Procedure

### Function Procedure

- A FUNCTION procedure is a series of *Visual Basic for Applicatons* statements enclosed by **`Function`** ..... **`End Function`**
- A **function** returns a value
- A **function** can take arguments, such as *literal values*, *variables* or *expressions* that are passed to it by a calling procedure
- A **function** can be used as if it is one of Excel's functions

### Function Declaration

The basic syntax is

```
Function name [(arglist)] [As type]
    [Specification]
    [statements]
    [name = expression]
End Function
```

# Programming Language

## Function Procedure

### Specification

The function declaration should include a specification:

- It generally describes the input requirements (i.e. $k$ positive integer, $n$ positive integer not equal to zero)
- It describes the characteristics of the returned value (i.e. proportion of observations, two decimal places)

Typically the specification follows the header and is written as a comment

- `Rem` **comment**
- ’ **comment**

# Programming Language

## Function Procedure

### Examples

-
  ```
  Function NextInteger (ByVal i As Integer) As Integer
  'Given any integer number, this is returned incremented by 1
  .....
  End Function
  ```

-
  ```
  Function ReturnAWord () As String
  Rem It simply returns a text
  .....
  End Function
  ```

# Programming Language

## Function Procedure

### Function Declaration

The header of the function includes:

- **name** (*required*): the name of the function
- **arglist** (optional): the list of arguments within parentheses that are passed to it when it is called. Multiple variables are separated by commas.

  The **arglist** argument has the typical syntax:

  ```
  ByVal varname As type, ByVal varname As type,....
  ```

  - **ByVal** means that the function access a copy of the variable (we always use this option)
  - **type** identifies data type of the argument that should be passed to the function

  The list of arguments declares ... these values are required for the computation ... the real values should be the same number and should be of the same type as specified in arglist ...
- **type** (optional): the data type of the value returned by the function procedure (Boolean, Integer, Double, etc.). *A good practice is to specify the returned data type*

## Function Declaration

The **body** of the function, between the key words `Function` ... `End Function` generally includes:

- **statements**, any group of statements to be executed within the function procedure

  ... list of tasks that should be accomplished ...

- **expression**, the returned value of the function

  ... the result of the computation...

# Programming Language

## Calling Function

### Calling Function

- A function generally is combined in an *expression*. It is called writing the function name, followed by the **real argument list** (.. data for a specific computation ..) in parentheses (round brackets)
- Real argument list could be made up by literals, variables or more general expressions, all matching the required arguments
- To save the returned value for more computations, the expression must be assigned to a variable

```
p1 = proportion (214, 910)
```

# Programming Language

## Operators, Functions, Statements

### Summary

| **Type** | Name | **Symbol** |
|---|---|---|
| Operators | Addition<br>Reminder | +<br>*Mod* |
| String Operator | Concatenation | & |
| Mathematical Function | Round<br>Square Root | `Round`(*expression*, [*decimal*])<br>`Sqr`(*number*) |
| String Function | Upper Case<br>Left<br>Right | `UCase`(*string*)<br>`Left`(*string*, *length*)<br>`Right`(*string*, *length*) |
| Date Function | Now<br>Difference | `Now`()<br>`DateDiff`(*interval*, *date1*, *date2*) |

## Exercise

The goal of this exercise is to familiarize yourself calling functions, hence for each question write an expression that yields a suitable answer. Further assign the expression to a variable (declare explicitly the variable)

1. Let `discount(ByVal p As Double, ByVal d As Single) As Double` be the function that returns the net price for a price $p$ and a discount rate $d$. Compute the saved amount when the price is equal to 12000 and the discount rate is equal to 25%
2. Let `character(ByVal name As String) As Integer` be the function that returns the number of occurred “s”. Report in a sentence whether (TRUE or FALSE) in the *name* “Weierstrass” occur more “s” characters than in the *name* “Descartes”
3. Let `circumference(ByVal r As Double) As Double` be the function that returns the circumference of a circle having radius $r$ . Report in a sentence the circumference of a circle having radius equal to 2.50, then of a circle having a double radius, and finally of a circle having a triple radius

## Solution

1. 
   ```
   Dim np As Double
   np = 12000 - discount(12000,0.25)
   ```
2. 
   ```
   Dim s As String
   s = "In Weierstrass occur more s than in Descartes -> "
   s = s & character("Weierstrass") > character("Descartes")
   ```
3. 
   ```
   Dim s As String
   s = "radius = 2.5 -> " & circumference(2.50)
   s = s & "radius = 2*2.5 -> " & circumference(2*2.50)
   s = s & "radius = 3*2.5 -> " & circumference(3*2.50)
   ```

# Function

## Exercise

### Exercise 1

We need a function that for every given code creates a new code having exactly 10 characters, hence if the given code is shorter than 10 it should be added leading zeros

```
Function coding(code As String) As String

    ’ the function creates and returns a 10 characters code
    ’ when the code is shorter than 10, leading zeros are added

    Dim zeros As String
    zeros = "0000000000"
    coding = Right(zeros & code, 10)
End Function
```

## Exercise 2

Given two integer numbers *a* and *b*, we define *a* “bigger” than *b* if and only if *a* is greater than two times *b*.

Write a function that checks if the number *x* is “bigger” than *y* and returns TRUE or FALSE.

The function works properly when *x* or *y* are negative? In case not modify your function (*tip*: consider only absolute values)

```
Function bigger(a As Integer, b As Integer) As Boolean

    Rem the function determines if a is bigger than b applying
    Rem the definition of "bigger"
    Rem a and b can be any integer number
    Rem the returned value is TRUE or FALSE

    bigger = (Abs(a) > 2 * Abs(b))
End Function
```

# Function

## Exercise 3

We need a function that for a two characters code is able to encrypt and return it.
This function should implement a simple encryption algorithm.
For example could be helpful the following:

- Given a character and an integer number (key), the character is modified shifting its ASCII coding by the key applying the formula char(mod(ASCII(character) + key), 255)
- The corresponding VBA expression is

  ```
  Chr((Asc(code) + key) Mod 255)
  ```
- The two encrypted characters make up in the reverse order the encrypted code

For example for thee code “DS” and the key 20, the character “D” becomes “X”, “S” becomes “g”, hence the encrypted code is “gX”

# Function

## Exercise

### Solution

```
Function encryption(ByVal code As String, ByVal key As Integer) As String

    Rem the function gets a code (made up by two characters) and an integer key
    Rem each character is encrypted shifting the character by the key
    Rem the function creates the encrypted code concatenating the resulting
    Rem encrypted characters in the reverse order

    Dim c1 As String
    Dim c2 As String
    c1 = Chr((Asc(Left(code, 1)) + key) Mod 255)
    c2 = Chr((Asc(Right(code, 1)) + key) Mod 255)
    encryption = c2 & c1
End Function
```

# Function

## Exercise 4

Typically we label a specific action of a computation (i.e. insertion, deletion, updating of data) with a time stamp, this it allows to order chronologically the actions of an execution.
We need a function that simply returns a time stamp.
The time stamp could be created as follows:

```
timestamp = T + seconds passed from first January 2000 to now
```

```
Function timestamp() As String

    ’ the function creates and returns a label:
    ’ at the character "T" is concatenated
    ’ the seconds passed from 1/1/2000 to now

    Dim t As String
    t = "T" & DateDiff("s", #1/1/2000#, Now())
    timestamp = t
End Function
```

# Function

## Exercise 5

Typically for security reasons the credit card number is displayed hiding the last four digits, replacing them with asterisk, hence we need a function that hides the last four digits of a credit card number (16 digits)

```
Function hide(ByVal ccn As String) As String

    ’ the function replaces the last 4 digits of a
    ’ credit card number with asterisk
    ’ the credit card number has 16 digits

    Dim h As String
    h = Left(ccn, 12) & "****"
    hide = h
End Function
```

# Function

## Exercise 6

We need a function to determine the pounds currency given euro currency. It's clear that the function requires euro amount and the currency rate.
We assume to know the daily exchange rate 1 GBP = X €

```
Function exchange(ByVal amount As Double, ByVal rate As Double) As Double

    Rem for an amount of money in Euro currency the function
    Rem determines the corresponding amount in GBP currency
    Rem rate -> 1 GBP = rate Euro

    Dim p As Double
    p = amount / rate
    exchange = Round(p, 2)
End Function
```

# Function

## Exercise 7

Consider the task to determine the ratio of ”Advanced Compute Skills” exam failures at each session. The ratio is equal to the number of failures (nf) over the total number (nt) of examinations multiplied by 100. Only two decimal places should be considered

```
Function failed(ByVal nf As Integer, ByVal nt As Integer) As Single

    ’ the function computes the ratio of the exam failures
    ’ when the number of failures
    ’ and the total number of examinations are provided
    ’ the ratio is returned with two decimal places
    ’ the total number of examinations shall be > 0

    failed = Round(nf / nt, 2)
End Function
```

# Outline

1. Function
   - Function Declaration
   - User Defined Functions

# About User Defined Functions

## User Defined Functions

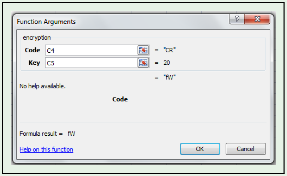
- Excel provides the user with a large collection of ready–made functions
- Sometimes you need a function that does a particular job, but any suitable function is available in the list
- In this case you can create your “User Defined” function using Visual Basic for Applications programming language
- A “User Defined” Function remain in a code module attached to a workbook

## Accessing Custom Functions

- If a workbook has a VBA code module attached to it that contains custom functions, those functions can be easily addressed within the same workbook
- You use the function name as if it were one of Excel's built–in functions
- Alternatively you find the functions listed in the **Insert Function Wizard**, within the category **User Defined**
- You have to specify the required arguments (cell reference or value)
- Arguments are typically stored within worksheet cells

## Example: function encryption

```
Function encryption(ByVal code As String, ByVal key As Integer) As String

    Rem the function gets a code (made up by two characters) and an integer key
    Rem each character is encrypted shifting the character by the key
    Rem the function creates the encrypted code concatenating the resulting
    Rem encrypted characters in the reverse order

    Dim c1 As String
    Dim c2 As String
    c1 = Chr((Asc(Left(code, 1)) + key) Mod 255)
    c2 = Chr((Asc(Right(code, 1)) + key) Mod 255)
    encryption = c2 & c1
End Function
```


Function Arguments

encryption

**Code** C4 = "CR"

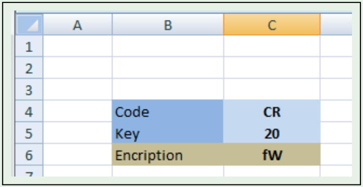
**Key** C5 = 20

= "fW"

No help available.

**Code**

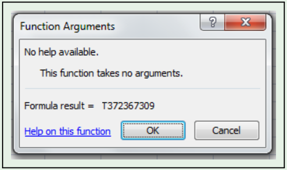
Formula result = fW



| | A | B | C |
|---|---|---|---|
| 1 | | | |
| 2 | | | |
| 3 | | | |
| 4 | | Code | CR |
| 5 | | Key | 20 |
| 6 | | Encription | fW |
| 7 | | | |

## Example: function timestamp

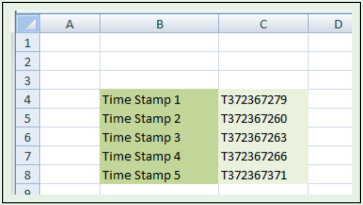
```
Function timestamp() As String

    ' the function creates and returns a label:
    ' at the character "T" is concatenated
    ' the seconds passed from 1/1/2000 to now

    Dim t As String
    t = "T" & DateDiff("s", #1/1/2000#, Now())
    timestamp = t
End Function
```


Function Arguments

No help available.

This function takes no arguments.

Formula result = T372367309



|  | A | B | C | D |
|---|---|---|---|---|
| 1 |  |  |  |  |
| 2 |  |  |  |  |
| 3 |  |  |  |  |
| 4 |  | Time Stamp 1 | T372367279 |  |
| 5 |  | Time Stamp 2 | T372367260 |  |
| 6 |  | Time Stamp 3 | T372367263 |  |
| 7 |  | Time Stamp 4 | T372367266 |  |
| 8 |  | Time Stamp 5 | T372367371 |  |