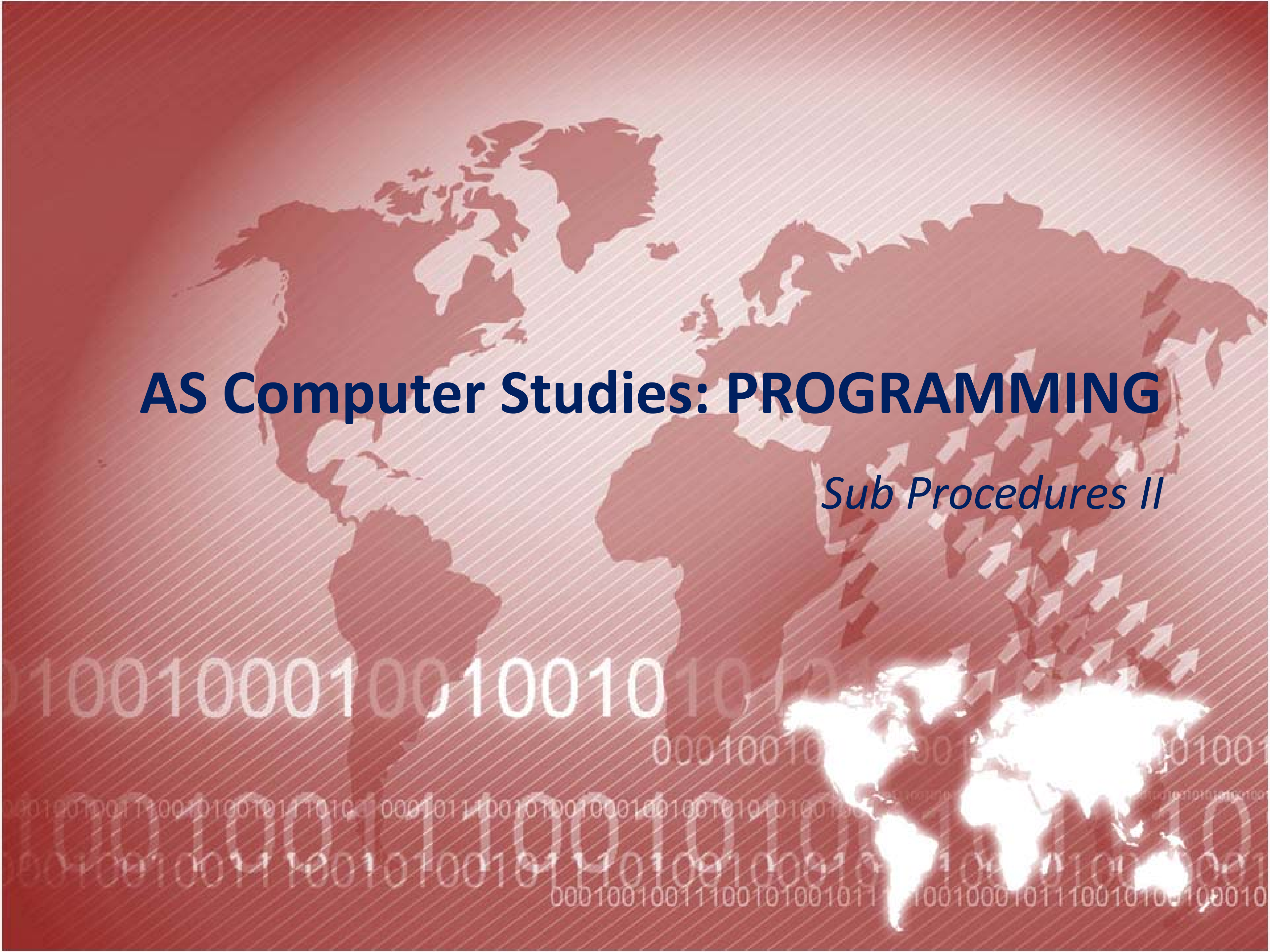


AS Computer Studies: PROGRAMMING

Sub Procedures II

Starter

- Pink booklet

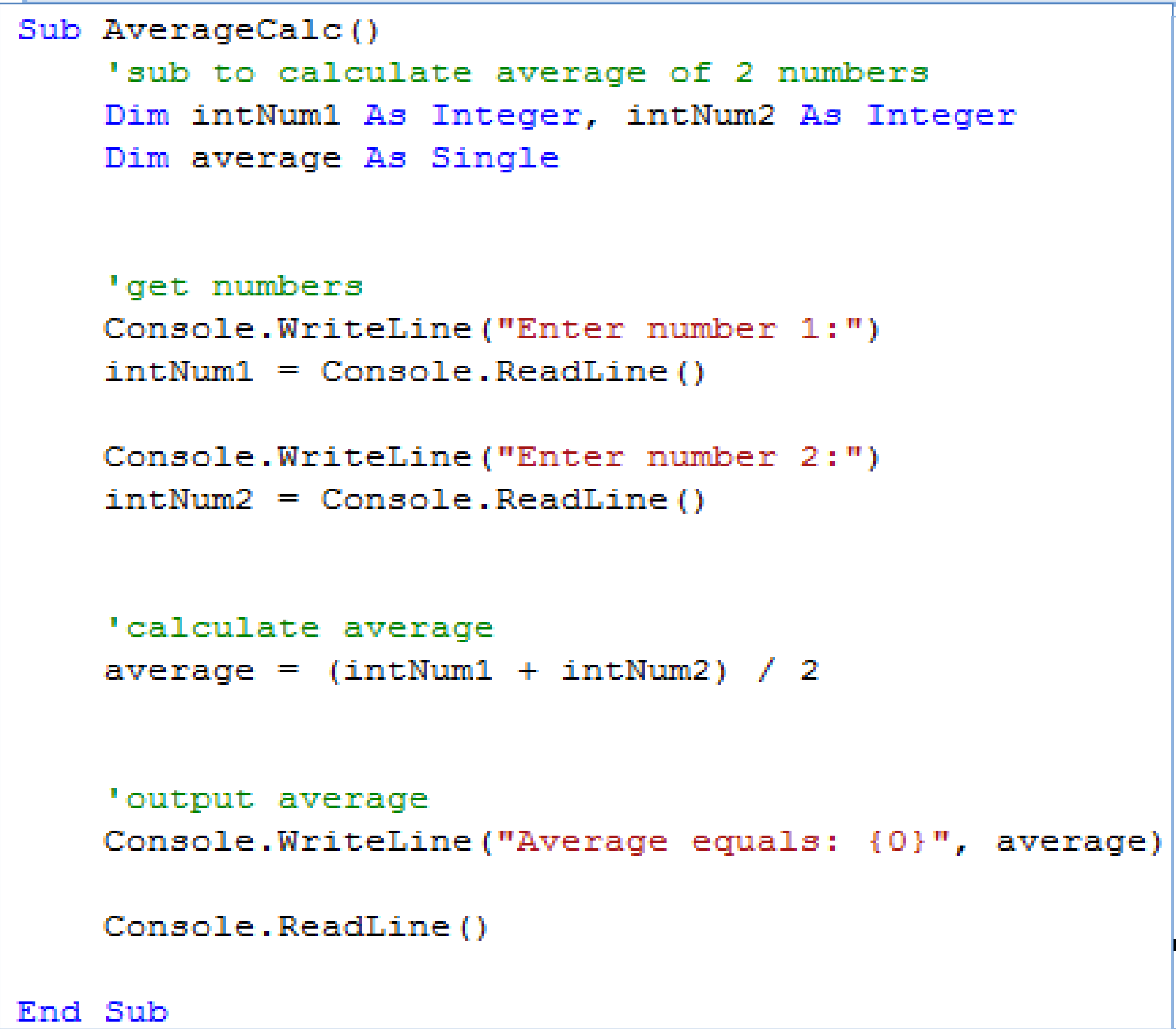
- Page 36

- Bottom left example – explain & state the output. **3 minutes**.

Objectives

- Understand the **importance of modular programming**.
- Know the **role of procedures** within programming.
- Understand the concept of **parameters**; **ByRef and ByVal**
- Use procedures within your programming.

Recap: Writing a Procedure



```
Sub AverageCalc()
    'sub to calculate average of 2 numbers
    Dim intNum1 As Integer, intNum2 As Integer
    Dim average As Single


    'get numbers
    Console.WriteLine("Enter number 1:")
    intNum1 = Console.ReadLine()

    Console.WriteLine("Enter number 2:")
    intNum2 = Console.ReadLine()


    'calculate average
    average = (intNum1 + intNum2) / 2


    'output average
    Console.WriteLine("Average equals: {0}", average)

    Console.ReadLine()

End Sub
```

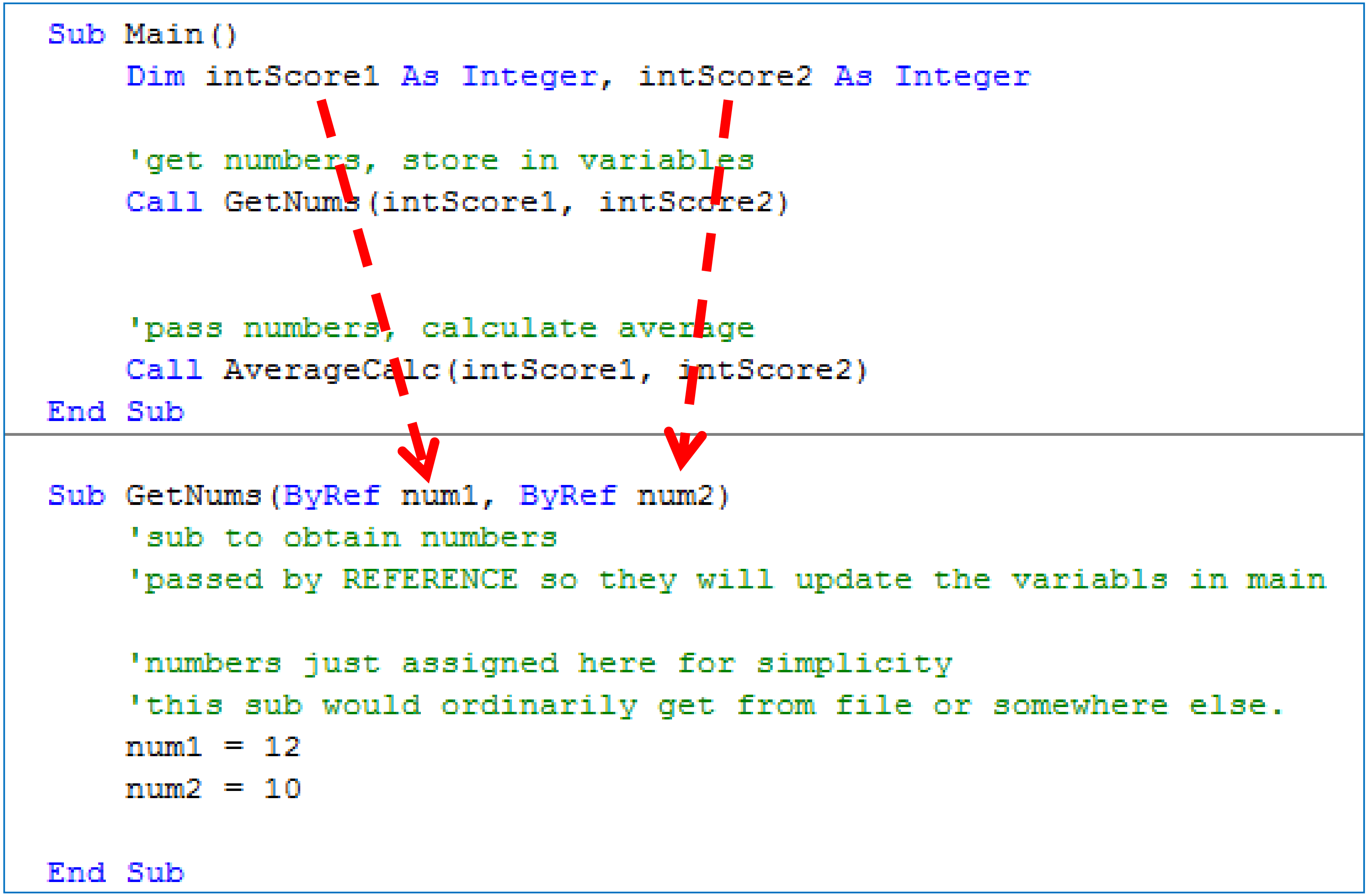
- You should always ensure that the **first thing** in the code for any procedure you write is a ***comment*** which explains what the procedure does

- Notice how this procedure is just doing **ONE** job – that is calculating the average and displaying it.

The Problem

- You request a user’s name and yearly salary in ONE procedure.
- You then need to calculate the deductions and monthly salary in **ANOTHER** procedure – where is the data to be used here coming from?
- Your first sub! What do you do!?
- There are a couple of options:
  - Global Variable Definition – **NOT** worth doing, **VERY** inefficient code.
  - Variable Passing – The **BEST** way to go.

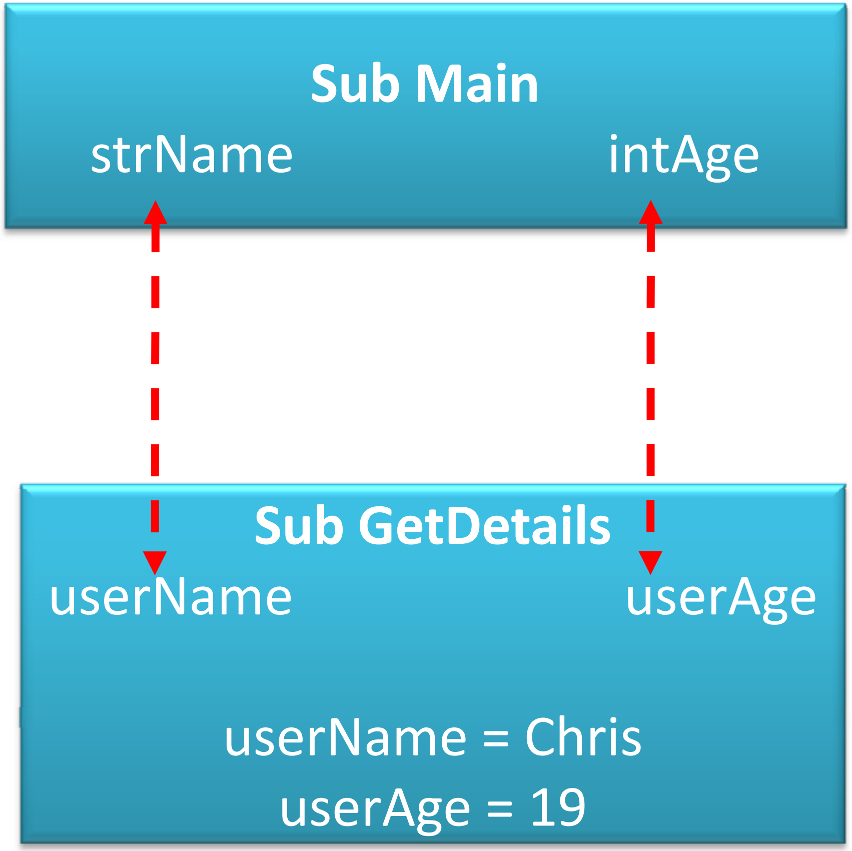
Variable Passing Example



```
Sub Main()
    Dim intScore1 As Integer, intScore2 As Integer

    'get numbers, store in variables
    Call GetNums(intScore1, intScore2)


    'pass numbers, calculate average
    Call AverageCalc(intScore1, intScore2)
End Sub

Sub GetNums(ByRef num1, ByRef num2)
    'sub to obtain numbers
    'passed by REFERENCE so they will update the variabls in main

    'numbers just assigned here for simplicity
    'this sub would ordinarily get from file or somewhere else.
    num1 = 12
    num2 = 10

End Sub
```

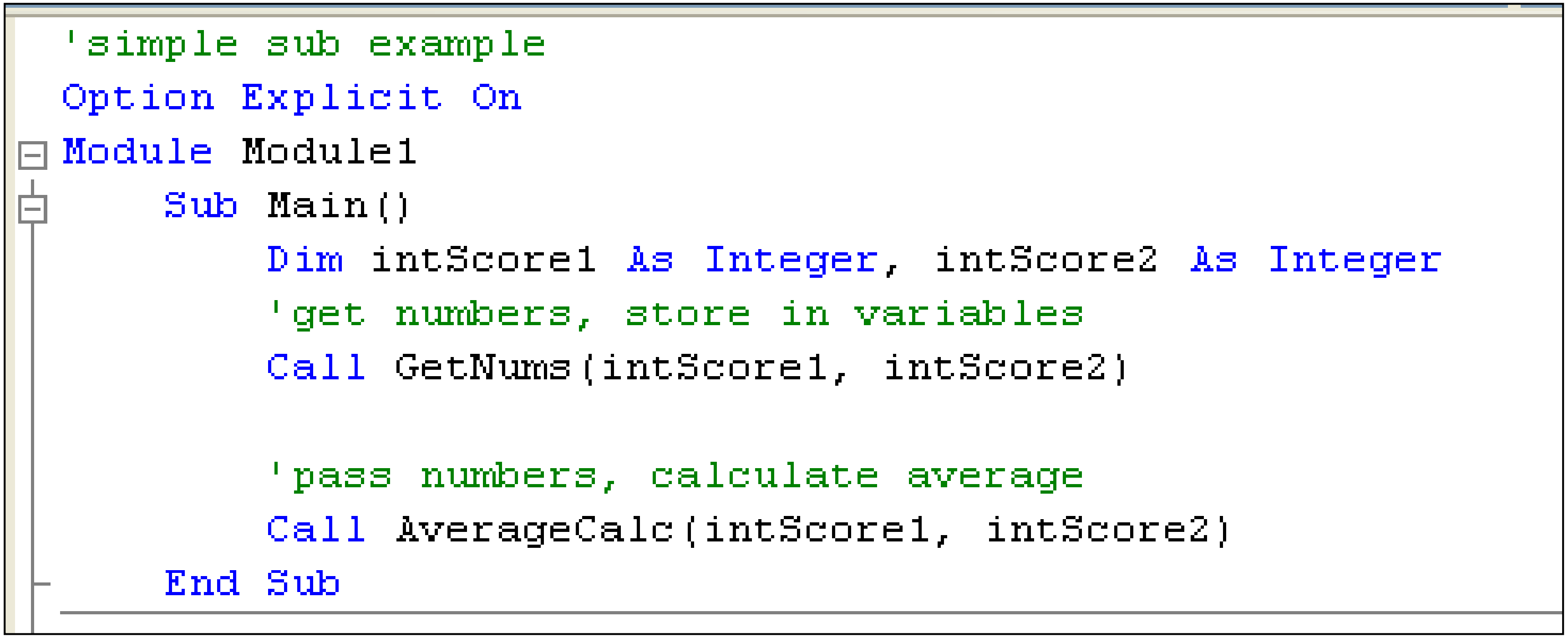
Passing by Reference (ByRef)

- Say you have two variables in one Sub.
- You can **pass a reference** (or pointer) to where they are in memory.
- This way your other subs can use them, and write directly back to the variables.
- This way, the **original calling sub will be able to use the new data**!



Data changed in the called sub (Get Details), updates the variables in Main.

Passing by Reference (ByRef) Example

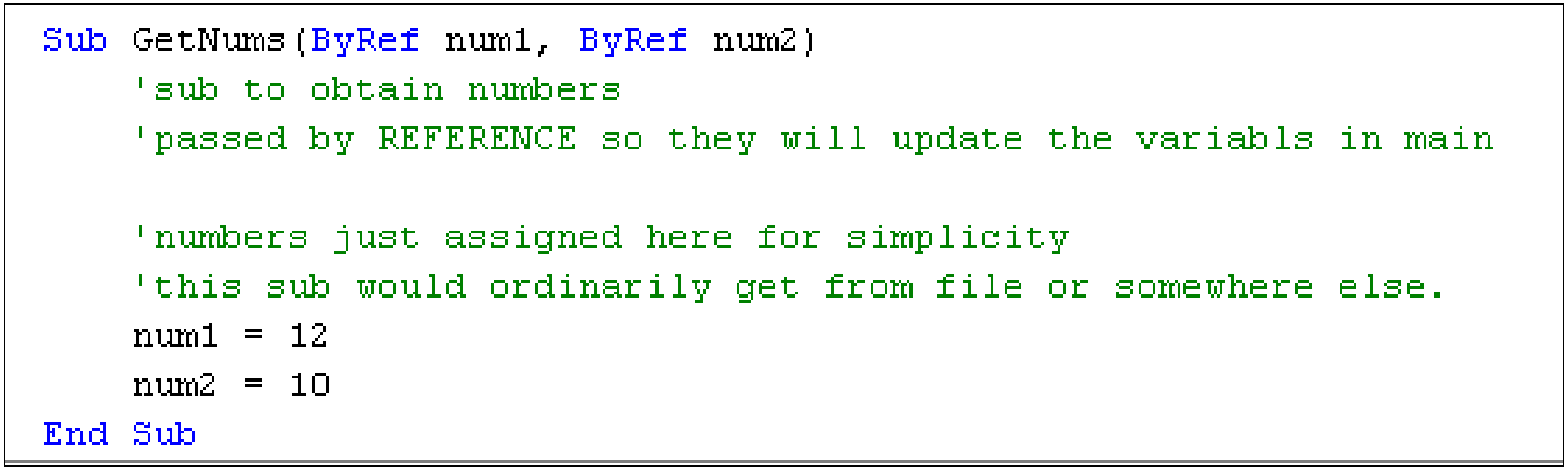


```
'simple sub example
Option Explicit On
Module Module1
    Sub Main()
        Dim intScore1 As Integer, intScore2 As Integer
        'get numbers, store in variables
        Call GetNums(intScore1, intScore2)

        'pass numbers, calculate average
        Call AverageCalc(intScore1, intScore2)
    End Sub
```

The main program body.

- Two variables declared here to store scores.
- GetNums call is passing these two variables.
- AverageCalc call is passing the same variables.

Passing by Reference (ByRef) Example

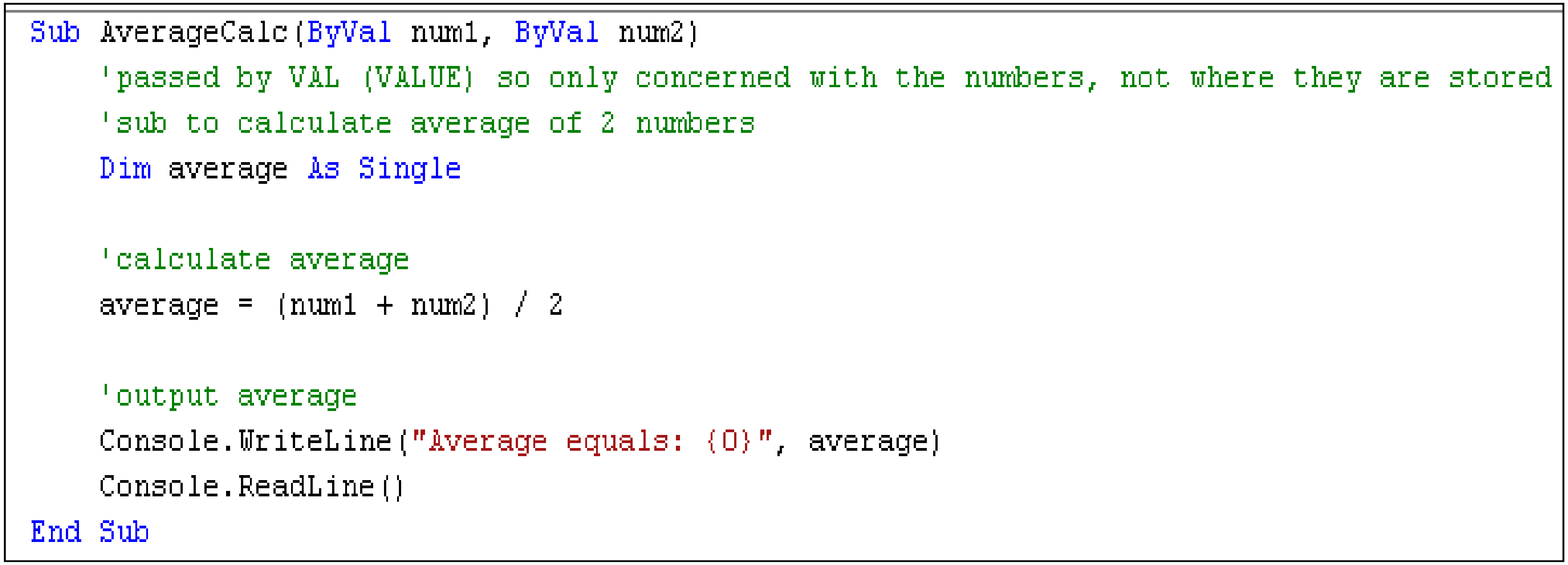


```
Sub GetNums(ByRef num1, ByRef num2)
    'sub to obtain numbers
    'passed by REFERENCE so they will update the variabls in main

    'numbers just assigned here for simplicity
    'this sub would ordinarily get from file or somewhere else.
    num1 = 12
    num2 = 10
End Sub
```

Note that:

- The **ByRef** keyword before each of the parameters
- **DIFFERENT** identifiers for the two parameters – **ALWAYS** recommended!
- Assigning values to num1 and num2 which actually are **returning these values to intScore1 and intScore2**, as they **REFERENCE** (ie REFER to) intScore1 and 2 in the main sub.

Passing by Reference (ByRef) Example



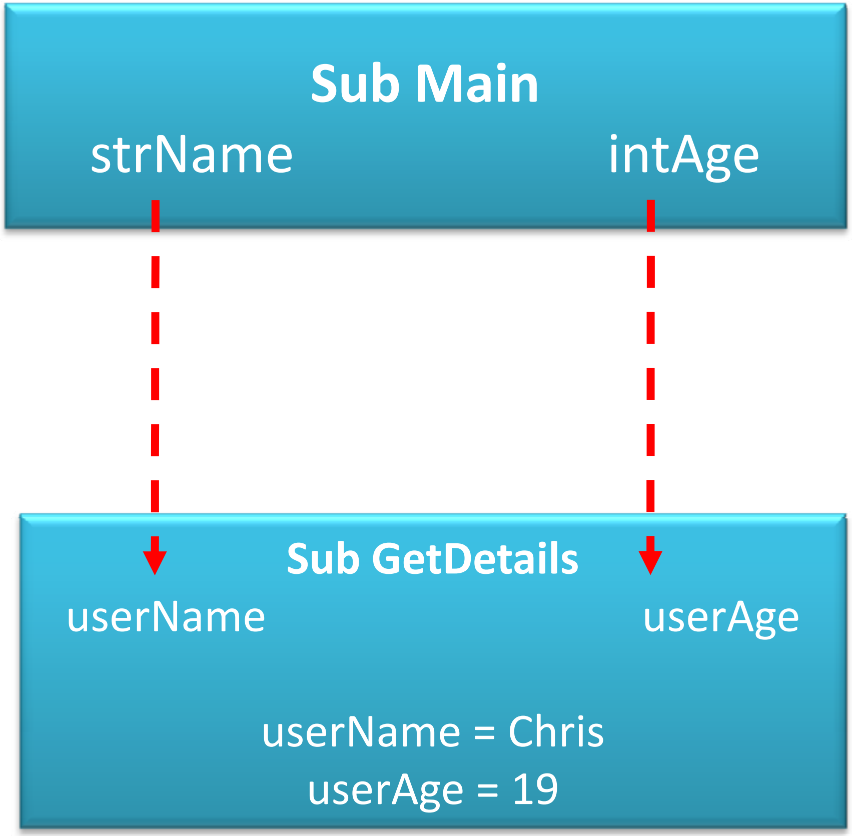
```
Sub AverageCalc(ByVal num1, ByVal num2)
    'passed by VAL (VALUE) so only concerned with the numbers, not where they are stored
    'sub to calculate average of 2 numbers
    Dim average As Single

    'calculate average
    average = (num1 + num2) / 2

    'output average
    Console.WriteLine("Average equals: {0}", average)
    Console.ReadLine()
End Sub
```

- **Notice how, the next call is to AverageCalc**
  - **passing the variables by VALUE as we don’t need to update the original variables (intScore1 and 2).**
- **They could be passed by reference too, in this case it doesn’t matter.**
- **What will happen if the first procedure will pass by value?**

Passing by Value (ByVal)

- What if you **don't want to change any variable data**? What if you just want the *contents of the data being passed to do something else* and you want to do things with the data without affecting the original variables?

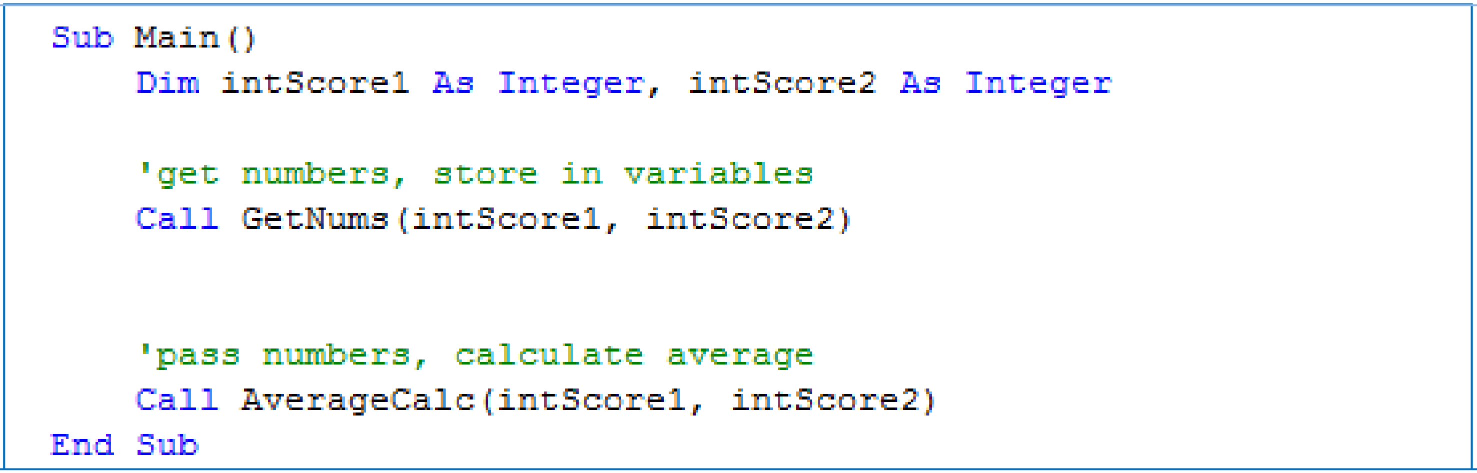
- To do this you **pass by value**. That is, you pass the **CONTENTS** of the variables, not the variables themselves.



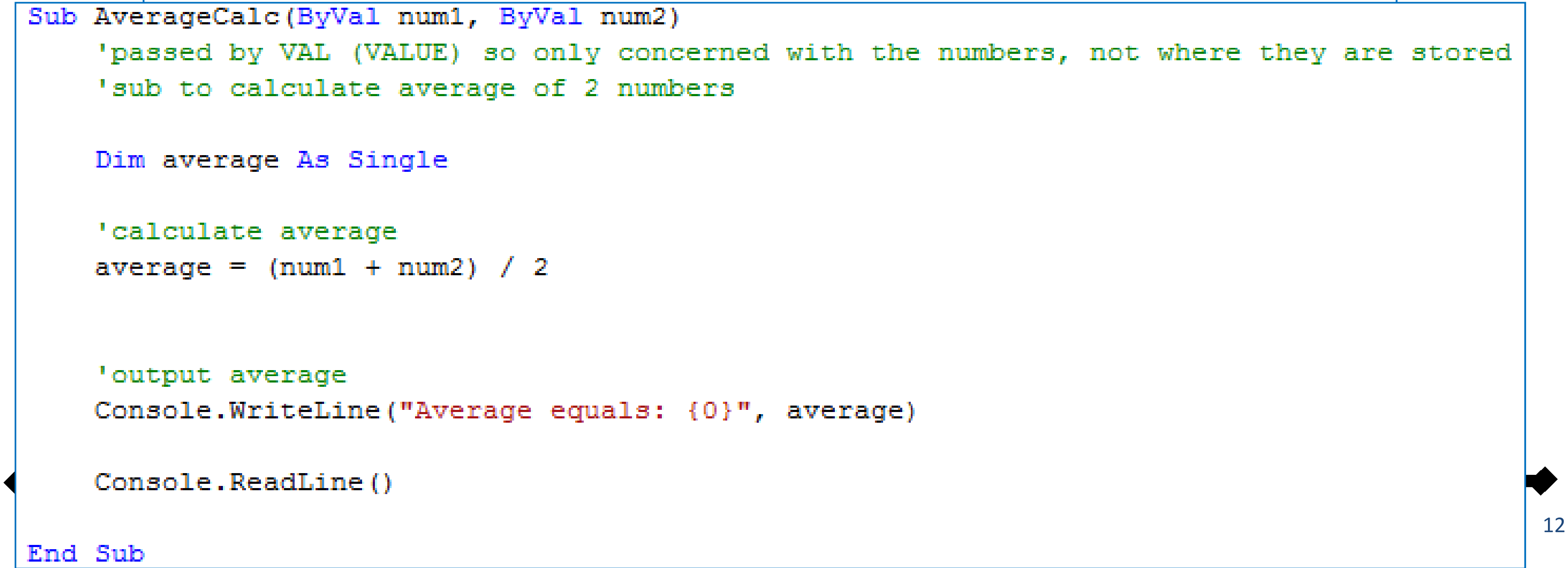
Data changed in the called sub (Get Details), does NOT update the variables in Main.

Passing by Value (ByVal) Example

- Notice how we only need to pass by value to AverageCalc as we aren’t actually changing the passed values, we are just using them.



```
Sub Main()
    Dim intScore1 As Integer, intScore2 As Integer

    'get numbers, store in variables
    Call GetNums(intScore1, intScore2)


    'pass numbers, calculate average
    Call AverageCalc(intScore1, intScore2)
End Sub
```



```
Sub AverageCalc(ByVal num1, ByVal num2)
    'passed by VAL (VALUE) so only concerned with the numbers, not where they are stored
    'sub to calculate average of 2 numbers

    Dim average As Single

    'calculate average
    average = (num1 + num2) / 2


    'output average
    Console.WriteLine("Average equals: {0}", average)

    Console.ReadLine()

End Sub
```

Tasks

1. Page 36 – task 17. Code in example 5. Note the difference if you change ByRef to ByVal (example 6).

2. Code in Example 7. Comments?

THEN:

Continue with task 18 (page 37)

- practice the use of ByVal & ByRef
- notice changes to output (if any) when switching from one to another.
- Make a note about this in the code using COMMENTS

Objectives

- Understand the **importance of modular programming**.
- Know the **role of procedures** within programming.
- Understand the concept of **parameters**; **ByRef and ByVal**
- Use procedures within your programming.

Plenary

- What is meant by modular programming?
- Why is it important to break down a program into smaller parts – what are the benefits?
- What is meant by passing by reference – someone come up and use the board to explain.
- What is meant by passing by value– someone come up and use the board to explain.