Parameters

FIT 100

Procedures allow tasks to be encapsulated for use at another time. Parameters provide a technique for providing inputs to procedures and receiving outputs from them.

FIT 100 Body Mass Computation

- The body mass index is used to determine if a person is overweight:

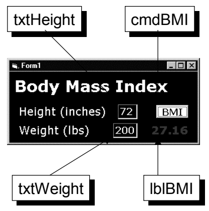
  BMI = 4.89weight/height^2
  - where the weight is in pounds, the height is in feet

- Converting it to a procedure is straightforward … so volunteer to write it, letting your friend build the GUI

```
Option Explicit
Dim weightLBS As Double
Dim heightIN As Double
Dim bodyMass As Double

Private Sub BMI()
    bodyMass = 4.89 * weightLBS / (heightIN/12)^2
End Sub
```

FIT 100 The GUI Built By A Friend

```
Private Sub cmdBMI_Click()
  Call BMI
   lblBMI.Caption = BMIndex
End Sub

Private Sub txtHeight_Change()
   BMIheight = txtHeight.Text / 12
End Sub

Private Sub txtWeight_Change()
   BMIweight = txtWeight.Text
End Sub
```



FIT 100 Incompatibility of Names

- A problem with names …

Procedure Assumes	Quantity	GUI Assumes
heightIN	height	BMIheight
weightLBS	weight	BMIweight
bodyMass	bmi	BMIndex

- Though in this case better communication might have saved this case, the need to associate different names is fundamental – it is essential in making procedures reusable.

The procedure context must use a specific set of names for inputs and outputs, while the calling context will use other names for these quantities ... Parameters associate the names in each context

FIT 100 Adding Parameters

- The body mass problems can be fixed without dieting
- Introduce parameters ...

```
Private Sub BMI(bodyMass As Double, weightLBS As Double, heightIN As Double)
   bodyMass = 4.89 * weightLBS / (heightIN/12) ^ 2
End Sub
```

- *Formal parameters* are part of the formal definition
- Formal parameters are “declared” in the parenthesized list following the procedure name
- To call the procedure, give the *actual parameters*

```
Call BMI(BMIndex, BMIweight, BMIheight)
```

```
BMIndex = 4.89 * BMIweight /(BMIheight/12)^2
```

FIT 100 Body Mass Index Program

```
Option Explicit
Dim BMIweight As Double
Dim BMIheight As Double
Dim BMIndex As Double
Private Sub BMI(bodyMass As Double, weightLBS As Double, heightIN As Double)
   bodyMass = 4.89 * weightLBS / (heightIN/12) ^ 2
End Sub
Private Sub cmdBMI_Click()
Call BMI(BMIndex, BMIweight,BMIheight)
   lblBMI.Caption = BMIndex
End Sub
Private Sub txtHeight_Change()
   BMIheight = txtHeight.Text / 12
End Sub
Private Sub txtWeight_Change()
   BMIweight = txtWeight.Text
End Sub
```

FIT 100 Formal Parameters

- The formal parameters are “declared” within the parentheses … the syntax is just like DIM statements
  - As with other variables, any names can be chosen
  - Each variable must be given a type: Integer, String, Double
- Formal parameter variables are “known” only within the procedure, i.e. they are local to a procedure
  - They never conflict with variables in the calling context
  - Different procedures could use the same formal parameter names without confusion or conflict
  - The technical term for this is “scope”: the scope of the formal parameter is local to the procedure.

FIT 100 Input vs Output

- Many programming languages (including VB6) provide several different ways of passing values back and forth between the actual and the formal parameters.
- The default in Visual Basic, and the only kind we’ll use in CSE/IMT 100, is **pass by reference**.
- Pass by reference allows information to flow in both directions.
  - Formal parameters can be used as inputs or outputs or both
  - Any changes made to a formal parameter will make a change to the corresponding actual parameter.

FIT 100 Actual Parameters

- The actual parameters must fulfill these requirements known as the formal/actual correspondence rules
  - There must be the same number of actual parameters in the call, as there are formal parameters in the proc declaration
  - The order of the parameters matters --
    - The 1st actual parameter corresponds to the 1st formal
    - The 2nd actual parameter corresponds to the 2nd formal
  - The types of the actuals must match the types of the formals
  - Any formal used as a procedure output must have a variable as an actual

```
Private Sub sample( a As String, b As String, c As String)
    a = c & b & “ay”
End Sub
```

```
Call sample( text, “N”, “Ix”)
```

FIT 100 Review -- Control Flow for Procedures

- When we call a procedure, Visual Basic jumps to the code for the procedure. It runs this code, then returns back to where the procedure was called, and continues on.

```
x = 5
Call squid()
x = x+1
```

```
Private sub squid()
    Print “hi there”
End Sub
```

FIT 100 Information Flow for Procedures

- When we call a procedure, the formal parameter temporarily becomes another name for the actual parameter.
- In other words, in Visual Basic the formal parameter temporarily becomes an *alias* for the actual parameter, for as long as the procedure is executing.
- Aliases in real life:
  - “The Sundance Kid” was an alias for Harry Longabaugh
  - Two names; one person.

FIT 100 Input parameter example

- Remember … the formal parameter becomes an alias for the actual parameter

```
x = 10
Call squid(x)

Private Sub squid(y As Integer)
   Print y
End Sub
```

The program prints 10

FIT 100 Output parameter example

```
Call squid(x)
Print x

Private Sub squid(y As Integer)
   y = 20
End Sub
```

The program prints 20

FIT 100 Both Input and Output

```
x = 10
Call squid(x)
Print x

Private Sub squid(y As Integer)
   y = 2*y
End Sub
```

The program prints 20

FIT 100 Expressions as Actual Parameters

```
x = 10
Call squid(x+5)

Private Sub squid(y As Integer)
   Print y
End Sub
```

The program prints 15

Here, Visual Basic makes an anonymous (secret) variable to hold the result of adding x and 5

FIT 100 Expressions as Parameters -- Caution

```
x = 10
Call squid(x+5)
Print x

Private Sub squid(y As Integer)
   y = 0
End Sub
```

BAD PROGRAM!! Don't do this!!

If the actual parameter is an expression, don't assign to the formal parameter! (Otherwise the result gets lost.)

FIT 100 Mini-Exercise #1

- What does the program print?

```
x = 10
Call squid(x+5)

Private Sub squid(y As Integer)
   Print y
End Sub
```

FIT 100 Mini-Exercise #1 -- Answer

- What does the program print?

```
x = 10
Call squid(x+5)

Private Sub squid(y As Integer)
   Print y
End Sub
```

The program prints 15

FIT 100 Mini-Exercise #2

- What does the program print?

```
x = 10
Call squid(x)
Print x

Private Sub squid(y As Integer)
   y = 20
End Sub
```

FIT 100 Mini-Exercise #2 -- Answer

- What does the program print?

```
x = 10
Call squid(x)
Print x

Private Sub squid(y As Integer)
   y = 20
End Sub
```

The program prints 20

FIT 100 Mini-Exercise #3

- What does the program print?

```
x = 10
Call squid(x+5)
Print x

Private Sub squid(y As Integer)
   y = 20
End Sub
```

FIT 100 Mini-Exercise #3 -- Answer

- What does the program print?

```
x = 10
Call squid(x+5)
Print x

Private Sub squid(y As Integer)
   y = 20
End Sub
```

Who knows! Who cares! This is an evil program!

(Well, OK, in our version of VB it prints 10. There won't be a question like this on any of our quizzes or final though.)

FIT 100 Mini-Exercise #4

- What does the program print?

```
x = 10
Call squid(x+5, y)
Print y

Private Sub squid(x As Integer, y As Integer)
   y = x+2
End Sub
```

FIT 100 Mini-Exercise #4 -- Answer

- What does the program print?

```
x = 10
Call squid(x+5, y)
Print y

Private Sub squid(x As Integer, y As Integer)
   y = x+2
End Sub
```

The program prints 17

FIT 100 Surgeon General's Warning!

- The "Fluency" book uses a different way of explaining parameter passing (as assignment statements into the formal parameters).
- For straightforward programs, this always gives the same results as pass by reference.
- However, for some messy cases it gives different results.
  - Ugh! We're never going to give you such programs in CSE/IMT 100 (in homework or quizzes).
  - If you go on to further study of programming, however, you will probably run into this.
  - The way described in the lecture is how it's actually done.

FIT 100 Summary

- Discussion of parameters for procedures
  - Parameters link the variables in the calling context with the variables in the procedure context
  - There is a 1-to-1 relationship between the formal parameters of the procedure definition and the actual parameters of the actual procedure call
  - The default way of passing parameters in Visual Basic is "pass by reference". The formal parameter becomes an alias for the actual parameter.