# General Procedures

- Sub Procedures, Part I
- Sub Procedures, Part II
- Function Procedures

# Sub Procedures, Part I

- Sub Procedures
- Variables and Expressions as Arguments
- Calling Other Sub Procedures

# Devices for modularity

- Visual Basic has two devices for breaking problems into smaller pieces:
  - Sub procedures
  - Function procedures

# Sub Procedures

- Perform one or more related tasks
- General syntax

```
Sub ProcedureName()
   statements
End Sub
```

# Calling a Sub procedure

- The statement that invokes a Sub procedure is also referred to as a **call statement**.
- A call statement looks like this:

  *ProcedureName()*

# Naming Sub procedures

- The rules for naming Sub procedures are the same as the rules for naming variables.

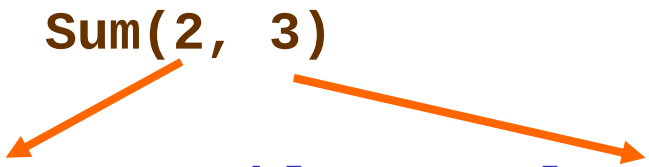
# Lab sheet 4.2: Code

```
lstBox.Items.Clear()
ExplainPurpose()
lstBox.Items.Add("")

Sub ExplainPurpose()
  lstBox.Items.Add("Program displays a sentence")
  lstBox.Items.Add("identifying a sum.")
End Sub
```

# Passing Values

- You can send values to a Sub procedure




```
Sub Sum(ByVal num1 As Double, ByVal num2 As Double)
  lstBox.Items.Add("The sum of " & num1 & " and " _
          & num2 & " is " & (num1 + num2) & ".")
End Sub
```

- In the Sum Sub procedure, 2 will be stored in *num1* and 3 will be stored in *num2*

# Arguments and Parameters

- 

```
Sum(2, 3)
```

arguments

parameters

```
Sub Sum(ByVal num1 As Double, ByVal num2 As Double)
```

displayed automatically

# Several Calling Statements

```
ExplainPurpose()
Sum(2, 3)
Sum(4, 6)
Sum(7, 8)
```

***Output:***

```
Program displays a sentence identifying a sum.
The sum of 2 and 3 is 5.
The sum of 4 and 6 is 10
The sum of 7 and 8 is 15.
```