

# CS 105 - Survey of Computer Science

Spring 2007 - Very Basic Visual Basic Procedures

This handout **very briefly** describes how to download course code from the course website into a Visual Basic (VB) project, and the syntax and semantics of procedures in Visual Basic. Reserved words are shown in **bold**. Syntactic categories are represented using `< category >`. Actual code is shown in **Courier** font.

## Downloading and Installing Existing Visual Basic Code Files

For this course, VB programs will consist of two files named `Form1.vb` and `Form1.Designer.vb`. `Form1.vb` is the source file containing the programming written handler code. `Form1.Designer.vb` is a source file generated by the form designer that contains the code that creates the actual GUI.

Whenever you are asked to download a VB program to use for this course, you should do the following steps:

1. Create a new VB project, noting the location of the project folder
2. Browse to the course website. Right-click on a link to the file to be downloaded. Select Save Target As (or Save Link As). A dialog box should pop up asking where to download the file to. Browse to the project folder. (There will already be files named `Form1.vb` and `Form1.Designer.vb` that will be overwritten by the downloaded files.) Click Save and OK to replacing them.
3. When you go back to VB, it will notify you that files in the project have been changed externally. Click Yes to reloading the files.

## Procedures and Functions

VB calls a procedure that does not compute a value that is returned as the result of calling the procedure a *subroutine*. A subroutine definition has the following syntax.

```
Private Sub <procedure name> (<parameter list>)  
    <body>  
End Sub
```

When a subroutine is called, the statements of the body are executed until the end of the subroutine is reached or an **Exit Sub** statement is executed. Note that all the handler procedures are subroutines.

The VB function definition has the following syntax

```
Private Function <function name> (<parameter list>) As < data type >  
    <body>  
End Sub
```

where the data type is the type of the returned value. In VB there are two ways to return a value. For this course, we will only use the return statement that has the following syntax.

```
Return <expression>
```

When a function is called, the statements of the body are executed until the return statement is executed. When the return statement is executed the function ends and the value of the expression becomes the value of the function call.

## Parameters

VB supports both parameters that are passed by value and parameters that are passed by reference. In a parameter list, those passed by value are declared as follows

```
ByVal <identifier> As < data type >
```

Parameters passed by reference are declared as follows:

```
ByRef <identifier> As < data type >
```

## Example Subroutine

Here is the code for a Sort subroutine developed in class on 3/23/07 that uses the insertion sort algorithm to sort an array of strings. It is to be added to the file `Form1.vb` available for download from the course website under that date.

```
Private Sub Sort(ByRef List As String())
    Dim N, J As Integer
    Dim PivotEntry As String
    Dim Done As Boolean

    N = 1 ' start with 2nd element
    While N < List.GetLength(0) 'there are still elements in list
        PivotEntry = List(N) 'move pivot to a temporary place
        J = N 'start at the pivot entry
        done = False 'have not found the right spot, yet
        While J > 0 And Not Done
            If List(J - 1) > PivotEntry Then
                'entry to the left is larger, so slide element down one
                List(J) = List(J - 1)
                J = J - 1
            Else
                Done = True 'found the hole
            End If
        End While
        List(J) = PivotEntry 'put the pivot in the hole
        N = N + 1 'do the next element
    End While
End Sub
```