

12-Sub Procedure and Function Procedure:

Most computer programs that solve real-world problems are much larger than those presented in the first few chapters. Experience has shown that the best way to develop and maintain a large program is to construct it from smaller pieces each of which is more manageable than the original program. This technique is called divide and conquer. This chapter describes many key features that facilitate the design, implementation, operation and maintenance of large programs.

Functions and Subroutines are programs designed for specific task, and could be called from the main program or from sub-procedures without pre definition or declaration. Users are allowed to call in any number of times which save the main program space, since it avoids repetition of code these subroutines could be designed by user or could be previously built. The concepts and descriptions are summarized in the following table.

Item	Subroutine	Function
Code	Sub Name (arguments) Statements End Sub	Function Name (arguments) Statements End Function
Remark	<ul style="list-style-type: none"> • Need call statement • Return values by arguments • Return many values (arguments) • Used for Input/output, condition treatment • Could be used with out arguments. 	<ul style="list-style-type: none"> • Used in arithmetic statement • Return value by its name • Return one value Used for arithmetic's or conversion of variable type.
Call Statement	Call Name(value1,value2,,,,)	Z=name(value1)
Exit statement	Exit Sub	Exit Function

12.1 Sub Procedures

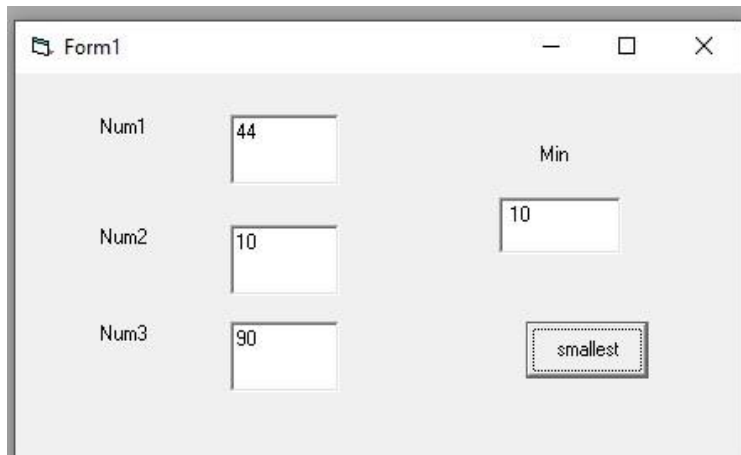
Sub procedure are created with the add procedure dialog (displayed when add procedure is selected from the tools menu). The add procedure menu item is grayed unless the code window is visible.

Example 1: Write a code program to read three integer numbers. Using a define sub procedure (Minimum) to determine the smallest of three integers. Display the smallest value in textbox.

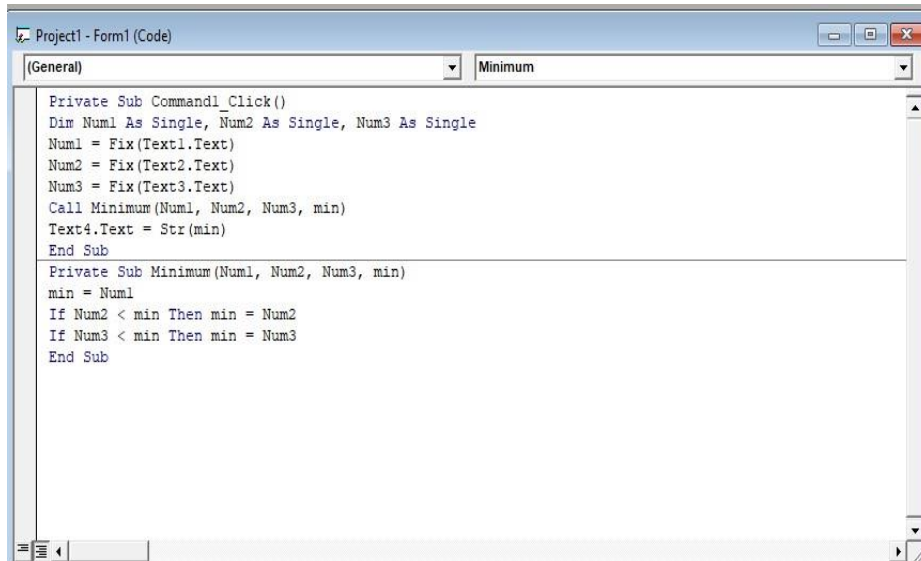
Solution:

```
Private Sub Command1_Click()  
Dim Num1 As Single, Num2 As Single, Num3 As Single  
Num1 = Fix(Text1.Text)  
Num2 = Fix(Text2.Text)  
Num3 = Fix(Text3.Text)  
Call Minimum(Num1, Num2, Num3, min)  
Text4.Text = Str(min)  
End Sub
```

```
Private Sub Minimum(Num1, Num2, Num3, min)  
min = Num1  
If Num2 < min Then min = Num2  
If Num3 < min Then min = Num3  
End Sub
```



The screenshot shows a Windows-style window titled "Form1". Inside the window, there are three labels on the left: "Num1", "Num2", and "Num3". Each label is followed by a text box containing a number: "44", "10", and "90" respectively. To the right of these, there is a label "Min" above a text box containing the number "10". At the bottom right of the form, there is a button with the text "smallest".



```

Project1 - Form1 (Code)
[General] Minimum
Private Sub Command1_Click()
    Dim Num1 As Single, Num2 As Single, Num3 As Single
    Num1 = Fix(Text1.Text)
    Num2 = Fix(Text2.Text)
    Num3 = Fix(Text3.Text)
    Call Minimum(Num1, Num2, Num3, min)
    Text4.Text = Str(min)
End Sub
Private Sub Minimum(Num1, Num2, Num3, min)
    min = Num1
    If Num2 < min Then min = Num2
    If Num3 < min Then min = Num3
End Sub

```

Example 2: Write a code program to read a one dimension array A (10). Using a define sub procedure (Sort) to Sort (increasing) the array A. Display the new array into picturebox.

Solution:

```

Private Sub Command1_Click()
    Dim A(10) As Single
    For I = 1 To 10
        A(I) = Val(List1.List(I - 1))
    Next I
    Call Sort(A, 10)
    For I = 1 To 10
        Picture1.Print A(I)
    Next I
End Sub

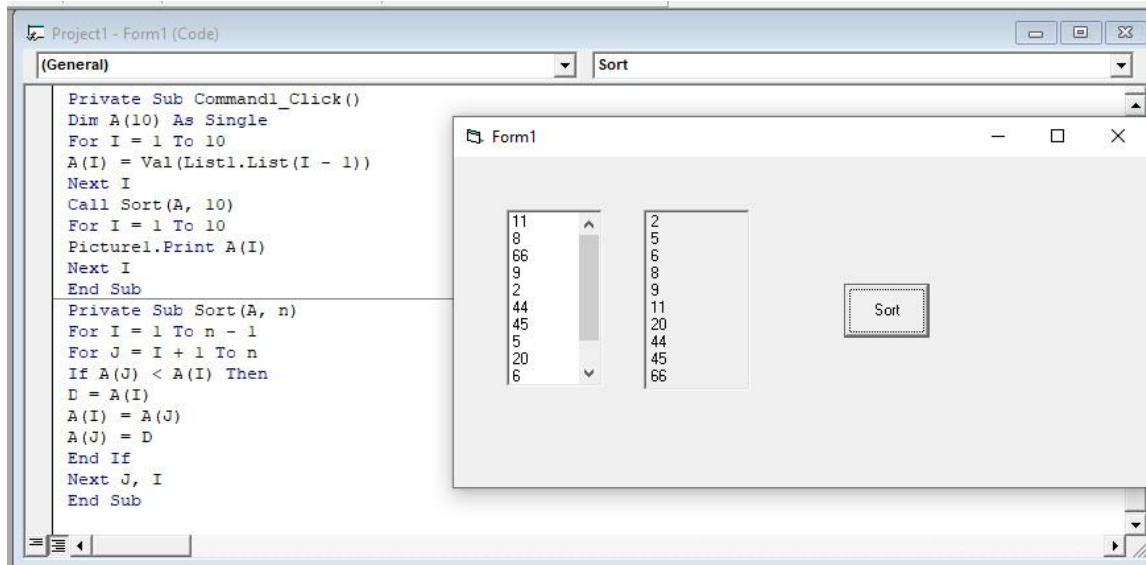
```

```

Private Sub Sort(A, n)
    For I = 1 To n - 1
        For J = I + 1 To n
            If A(J) < A(I) Then
                D = A(I)
                A(I) = A(J)
                A(J) = D
            End If
        Next J
    Next I
End Sub

```

```
Next J, I
End Sub
```



2. Function Procedures: function procedures and sub procedures share the same characteristics, with one important difference- function procedures return a value (i.g., give a value back) to the caller, whereas sub procedures do not. Fact implicitly returns variant.

Fact could also have been created by typing the function procedure directly into the code window. The line

```
Private Function Fact()
```

is the function procedure header. The header contains the keyword function, the function name and parentheses. The declarations and statements that the programmer will insert between the header and End Function form the function procedure body, Fact is invoked with the line.

```
Result= Fact( )
```

When a function procedure name (such as Fact) is encountered at run time, the function procedure is called, causing its body statements to execute. Consider the complete definition for Fact

```
Private Function Fact( N )
```

```
Fact=N^2
```

```
End Function
```

A function procedure return value is specified in the body by assigning a value to the function

procedure name, as in

```
Fact=N^2
```

Then returns (along with the value returned) to the calling statement

```
Result=Fact (N)
```

And the return value is assigned to variable result. Program execution then continues with the next statement after the call to Fact.

All function procedure definitions contain parentheses, the parentheses may be empty (e.g. Fact) or may contain one parameter variable declarations. Consider the following function procedure:

```
Private Function Area (s1 as single,s2 as single)
```

```
Area=s1*s2
```

```
End Function
```

Which declare two parameter variables s1, and s2. Area's return type is variant. Area is called with

the statement

```
Square=area(8.5, 7.34)
```

The value 8.5 is stored in s1 and the value 7.34 is stored in s2.

Example 4: Write a code program to read three integer numbers. Using a define sub Function (Min) to determine the smallest of three integers. Display the smallest value in textbox.

Solution:

```
Private Sub Command1_Click()
Dim Num1 As Single, Num2 As Single, Num3 As Single, Result As Single
Num1 = Fix(Text1.Text)
Num2 = Fix(Text2.Text)
Num3 = Fix(Text3.Text)
Result = MinNum1, Num2, Num3)
Text4.Text = Str(Result)
End Sub
```

```
Private Function Min(Num1, Num2, Num3)
Min = Num1
If Num2 < Min Then Min = Num2
If Num3 < Min Then Min = Num3
End Function
```

#####

Example 5: Write a code program to input the value of N. Using a define sub function fact to determine(N!). Display the result into text box.

Solution:

```
Private Sub Command1_Click()
Dim N As Single, Result As Double
N = Val(Text1.Text)
Result = Fact(N)
Text2.Text = Str(Result)
End Sub
```

```
Private Function Fact(N)
Dim I, F
F = 1
For I = 1 To N
F = F * I
Next
Fact = F
End Function
```

