

Chapter 15

Creating User-Defined Functions

❖ Learning how to create user-defined function

Function is a method that returns a value to the calling procedure. You can create user-defined function to perform certain calculations and some other tasks.

The general format of a function is as follows:

Public Function `functionName` (param As dataType,.....) As dataType

or

Private Function `functionName` (param As dataType,.....) As dataType

- * Public indicates that the function is applicable to the whole project and
- * Private indicates that the function is only applicable to a certain module or procedure.
- * param is the argument or parameter of the function that can store a value. You can specify more than one parameter, separated by commas.

Example 15.1: Cube Root Calculator

In this example, we will create a program that calculates the cube root of a number. The function code is

```
Public Function cubeRoot(ByVal myNumber As Single) As Single
    Return myNumber ^ (1 / 3)
End Function
```

The keyword *Return* is to compute the cube root and return the value to the calling procedure.

Place the function procedure in the general section of the module.

Next, design an interface and create a procedure that call the function and display the value to user.

To create the interface, place three label controls and one textbox into the form.
Rename the label and use it to display the cube root to be LblCubeRoot.
Now click on the textbox and enter the following code:

```
Private Sub TextBox1_TextChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles TextBox1.TextChanged
```

```
    LblCubeRoot.Text = cubeRoot(Val(TextBox1.Text))
```

```
End Sub
```

Press F5 to run the program and you should get the following output:

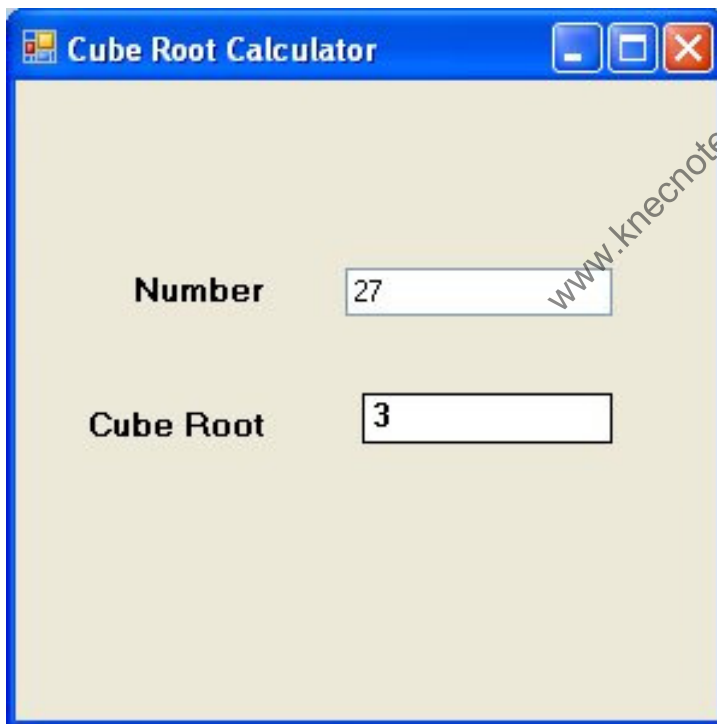


Figure 15.1: Cube Root Calculator

Example 15.2

In this example, we create a function that can convert mark to grade, a handy function to manage college examinations or tests processing. In this function, we use the Select case control structure to convert marks of different range to different grades.

```
Public Function grade(ByVal mark As Single) As String
```

```
    Select Case mark
```

```
        Case Is > 100
```

```
            Return "Invalid mark"
```

```
        Case Is >= 80
```

```
            Return "A"
```

```
        Case Is >= 70
```

```
            Return "B"
```

```
        Case Is >= 60
```

```
            Return "C"
```

```
        Case Is >= 50
```

```
            Return "D"
```

```
        Case Is >= 40
```

```
            Return "E"
```

```
        Case Is >= 0
```

```
            Return "F"
```

```
        Case Is < 0
```

```
            Return "Invalid mark"
```

```
    End Select
```

```
End Function
```

We need to design an interface for the user to enter the marks and we also need to write a procedure to call the function and display the grade on a label. To achieve the purpose, we will insert the following controls and set their properties as follows:

Control	Properties
Label1	Text: Mark ; font bold
Label2	Text:Grade ; font bold
TextBox1	Name: TxtMark
Lable3	LblGrade

We also need to write a procedure to call the function. Click on Textbox1 and enter the following code:

```
Private Sub TxtMark_TextChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles TxtMark.TextChanged
```

```
    If TxtMark.Text = "" Then
        Lbl_Grade.Text = "Enter Mark"
    Else
        Lbl_Grade.Text = grade(Val(TxtMark.Text))
    End If
```

```
End Sub
```

The procedure will compute the value entered in the textbox by the user by calling the grade () function and display the result on the label Lbl_Grade.

The output is shown in Figure 15.2:

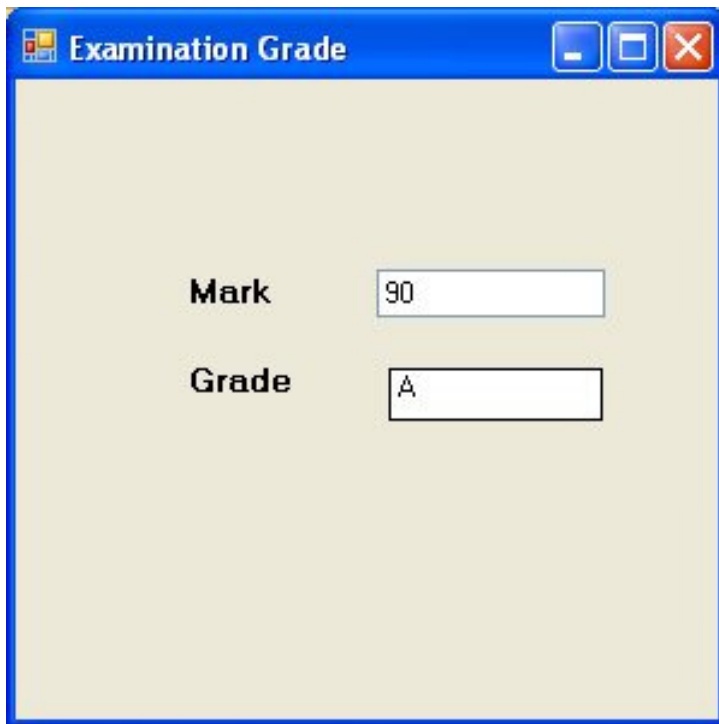


Figure 15.2

Example 15.3: BMI calculator

Many people are obese now and it could affect their health seriously. If your BMI is more than 30, you are obese. You can refer to the following range of BMI values for your weight status.

Underweight = <18.5

Normal weight = 18.5-24.9

Overweight = 25-29.9

Obesity = BMI of 30 or greater

Now we shall create a calculator in Vb2010 that can calculate the body mass index, or BMI of a person based on the body weight in kilogram and the body height in meter.

BMI can be calculated using the formula $\text{weight} / (\text{height})^2$, where weight is measured in kg and height in meter. If you only know your weight and height in lb and feet, then you need to convert them to the metric system. To build the calculator, we need to create a function that contains two parameters, namely height and weight, as follows:

Public Function BMI (ByVal height, ByVal weight)

Return Val ((weight) / (height ^ 2))

End Function

Next, design an interface that includes four labels, three of them is used for labeling height, weight and BMI and the last one is to display the value of BMI. We also inserted two text boxes to accept input of height and weight from the user. Lastly, insert a button for the user to click on in order to start the calculation process. Set the properties as follows:

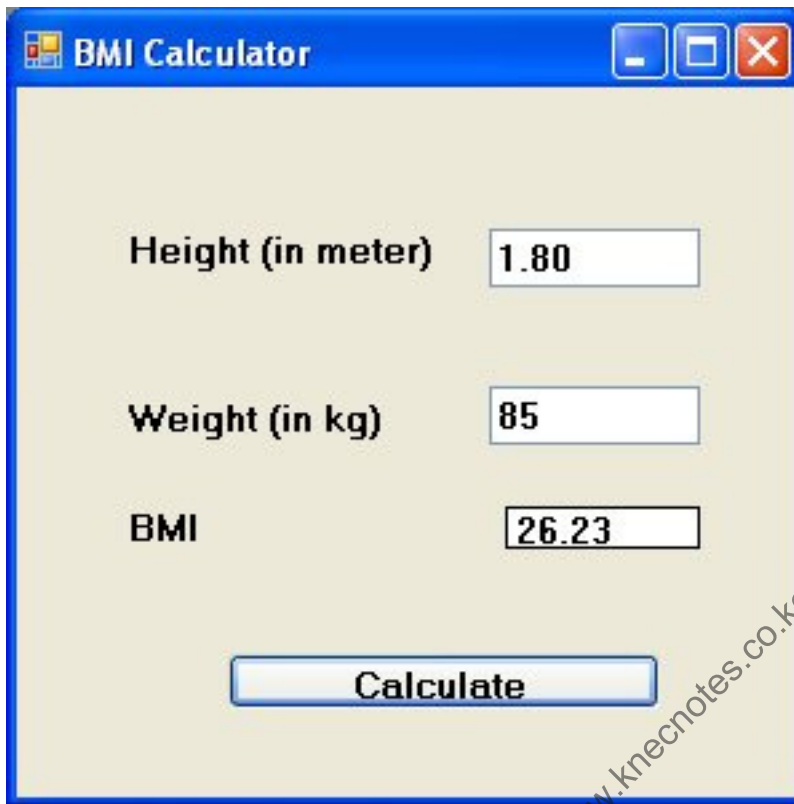
Control	Properties
Label1	Text : Height (in meter) Font : Microsoft Sans Serif, 10 pt, style=Bold
Label2	Text : Weight (in kg) Font : Microsoft Sans Serif, 10 pt, style=Bold
Label3	Text : BMI Font : Microsoft Sans Serif, 10 pt, style=Bold
Label4	Name: LblBMI Text : Blank Font : Microsoft Sans Serif, 10 pt, style=Bold
Textbox1	Name; TxtH Text : Blank Font : Microsoft Sans Serif, 10 pt, style=Bold
Textbox2	Name; TxtW Text : Blank Font : Microsoft Sans Serif, 10 pt, style=Bold

Now, click on the button and enter the following code:

```
LblBMI.Text = Format (BMI(TxtH.Text, TxtW.Text), "0.00")
```

We use the format function to configure the output value to two decimal places. This procedure will call the function BMI to perform calculation based on the values input by the user using the formula defined in the function.

The output is shown in Figure 15.3



Height (in meter)	1.80
Weight (in kg)	85
BMI	26.23

Calculate

Figure 15.3

Example 15.4: Future Value Calculator

In this example, the user can calculate the future value of a certain amount of money he has today based on the interest rate and the number of years from now, supposing he or she will invest this amount of money somewhere. The calculation is based on the compound interest rate. This reflects the time value of money.

Future value is calculated based on the following formula:

$$PV = FV \left(1 + \frac{i}{100} \right)^n$$

The function to calculate the future value involves three parameters namely the present value (PV), the interest rate (i) and the length of period (n). The function code is shown below:

```
Public Function FV(ByVal PV As Single, ByVal i As Single, ByVal n As Integer) As Double
```

```
    Return PV * (1 + i / 100) ^ n
```

```
End Function
```

The code to display the Future Value is

```
Private Sub BtnCal_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles BtnCal.Click
    LblFV.Text = FV(TxtPV.Text, TxtI.Text, TxtYear.Text).ToString("C")
```

```
End Sub
```

The screenshot shows a Windows-style application window titled "Future Value Calculator" designed by Liew Y K. The window has a light beige background and a blue border. It contains four rows of labels and text boxes: "Present Value" with "10000", "Interest Rate" with "4", "Number of Years" with "20", and "Future Value" with "\$21911.23". A "Calculate" button is positioned at the bottom center of the window.

Figure 15.4: The Future Value Calculator

Summary

In this chapter, you learned how to create user-defined functions. Among them are the cube root calculator, the examination grades calculator, the BMI calculator and the future value calculator.