INTRODUCTION TO VISUAL BASIC 6
Table of Contents

Introduction ..................................................................................................................... 3
Starting the Visual Basic Program ................................................................................... 4
  Program 1 .................................................................................................................... 4
Explaining the Windows .................................................................................................. 5
  The Title Bar, Menu Bar and Tool Bar ........................................................................ 6
  The Form Window ....................................................................................................... 7
    Program 2 .................................................................................................................. 7
Using Code ...................................................................................................................... 8
  Program 3 ................................................................................................................... 8
The General Toolbox ..................................................................................................... 9
  Program 4 ................................................................................................................... 10
The Properties Window .................................................................................................. 11
The Project Explorer and The Form Layout Windows ................................................... 13
  Program 5 .................................................................................................................. 14
Designing the Interface .................................................................................................. 15
  The Text Box .............................................................................................................. 15
  The List Box .............................................................................................................. 15
  The Combo Box ......................................................................................................... 15
  The Option Button .................................................................................................... 15
  The Check Box .......................................................................................................... 16
  The Frame .................................................................................................................. 16
  The Timer ................................................................................................................... 16
  The Scroll Bars .......................................................................................................... 16
    Program 6 ................................................................................................................ 17
    Program 7 ................................................................................................................. 18
    Program 8 ................................................................................................................ 19
Using Colour ................................................................................................................. 19
  Program 9 .................................................................................................................. 19
  Program 10 ............................................................................................................... 20
Essential Elements ........................................................................................................ 21
  Variables ..................................................................................................................... 21
    Program 11 ............................................................................................................. 23
If Statements ................................................................................................................. 23
Introduction

Visual Basic is an event-driven programming language. Windows type applications are produced by using forms on which are placed various controls. Code can be attached to the controls to produce various results.

Visual Basic is a Windows development language and so some familiarity with the Windows environment is required. In a simple form, working with Windows involves three key concepts: windows, events and messages.

A window is a rectangular area with its own boundaries, the boundaries of which can be fixed or adjustable. The usual windows such as an Explorer window or a dialogue box are common. However command buttons, icons, option buttons, menu bars etc are also windows.

All these windows are managed by the operating system by the assignment of unique id numbers (a window handle). These are continually monitored for signs of activity or events. An event can occur through a user action such as a mouse click or key press, through a program request or as a result of another window’s action.

Whenever an event occurs, it causes a message to be sent to the operating system. The operating system will process the message and broadcast it to other windows. Each window can then take the appropriate action based on its instructions for dealing with that message. Most of this is dealt with automatically by Visual Basic as you develop your applications.

In traditional (procedural) programming, the application controls the sequence of execution of the code. Execution will start with the first line and then follow a predefined path, calling various procedures as required.

In event-driven programming (Visual Basic), there is no predetermined path – the various code sections are executed in response to events. It is the sequence of these events that determines the order of code execution. This can result in the path through the application’s code differing each time the program runs.

Because you cannot predict the sequence of events, any code must make certain assumptions when it executes. Therefore, you should structure your applications to make sure that any assumptions will always be valid; for example, by disabling a command button that starts a procedure until the entry field contains a value.
Starting the Visual Basic Program

When opening Visual Basic, the following screen will appear so that you can either:

- Choose a new project by clicking on Standard.EXE
- Choose an existing project by clicking on the Existing tab and then browsing to find the project you want.

Program 1

1. Select Standard.EXE from the new project window
2. Make sure Form1 is selected in Project Explorer.
3. In the Form1 properties, find Caption and enter ‘My First Application’ alongside. Press return. You should see the title of the form change.
4. Run the application.
5. This application will create a window, which can be resized, moved and minimised like a normal window.
6. You can quit by closing the window in the normal way.
7. Save the project and form with the name ‘First’.
Explaining the Windows

The following shows everything that will appear when starting a new project. Each of the separate parts can be opened and closed and repositioned according to your preferences.

We will now look at each part individually.
The Title Bar, Menu Bar and Tool Bar

The Title Bar tells you the name of the project you are working on.

The Menu Bar provides a series of drop down menus, some of which are also available as buttons on the tool bar.

The standard toolbar consists of the following buttons. By hovering the mouse over each button, a box will appear saying what each button is for. Complete the following using this method.

Title Bar
Menu Bar
Standard Tool Bar
Program 2

1. If it is not still open, re-open your ‘First’ project.
2. In the Toolbox, locate the icon for a label and click on it.
3. Move the mouse pointer to the Form.
4. Place the mouse pointer at the top-left corner of the label. Keeping the mouse button pressed, drag the mouse down and to the right to create the label of the size you want. Then release the button.
5. It can be resized if required by using the resize tabs.
6. Ensure the label is selected, and the Label properties should appear on the right.
7. Find the Alignment property and change it to ‘2 – Center’. This will place the caption in the centre of the label.
8. Find the BorderStyle property and change it to ‘1 – Fixed Single’
9. Find the Caption property and change it to ‘My Custom Label’.
Using Code

Not everything can be done by setting properties initially, often we want these properties to change when a particular event takes place. This is done by adding instructions to the Click event of the objects concerned. These instructions are usually referred to as code.

You can access the editor where code is added by double-clicking on the object concerned – this could be a label, a button, a form etc.

Program 3

Try the following program to display your name:
1. Select Standard.EXE from the new project window
2. Make sure Form1 is selected in Project Explorer.
3. In the Properties window change the Caption property for Form1 to Print My Name
4. Double-click anywhere on the form to display the code template which will look like this:

```
Option Explicit

Private Sub Form_Load()
  Form1.Show
  Form1.Print "John Smith"
End Sub
```

5. Where the cursor is located type the following:
   Form1.Show
   Form1.Print "John Smith"

6. You are now ready to run the program. There are two options available for this located under Run on the menu bar. These are:
   - **Start** – the program will run and then stop if a line of source code contains an error.
   - **Start with full compile** – the computer will check the code first for errors and will only run if no errors are found.

7. When you have successfully run your program, close the program in the usual way for closing a window by clicking on the X at the top right.

8. Select File/Save Project As and then save first the form and then the project file using the same name to the location specified by your tutor.

Now amend the code so that your address is displayed under your name.
The General Toolbox

The buttons in the General Toolbox allow you to add components to your form. Each button provides a different component. These all have different properties that can be set in the Properties window. Hovering the mouse over each button will display its use.
Program 4

1. Open a new project
2. Place a label on the form in the top right
3. Place a button on the form across the bottom centre. Do this by using the button icon from the toolbox.
4. In the button’s properties, change the caption to ‘Change the Label’.
5. If you place the character & in front of the C, the C will be underlined. When running the program, you can click the button with the keyboard shortcut ALT + C.
6. Double click the button and the Text editor will open with some code already entered in it. The code will read:
   
   ```vba
   Private Sub Command1_Click()
   End Sub
   ```

   End Sub
7. Code can be added in the space between these lines. This is the code that will run when the button is clicked. Clicking the button is known as an event.
8. Enter the following code in the space:
   
   ```vba
   Label1.Caption = "I've been changed"
   ```
9. Run the application and click on the button.
10. Now add the following lines of code. Run the program after the addition of each line to see what happens:
    
    ```vba
    Label1.Caption = "My new caption"
    Label1.Alignment = vbCenter
    Label1.BackColor = vbBlue
    Label1.ForeColor = vbYellow
    Label1.Font.Size = 18
    Label1.FontBold = True
    Label1.ToolTipText = "This is a label"
    ```
11. Keep experimenting with the code, changing values and colours. REMEMBER – VB is American and therefore uses American spelling for words like colour and centre.
The Properties Window

The following show the Properties Window associated with Form1. Each type of object will have its own list of properties. Normally you will only see part of the contents of this window and will have to scroll up and down to see it all. It is up to you whether you use it by categories or alphabetically. All options are available in both views. When using different objects, it is useful to change the name to something more meaningful. Good practice recommends that you prefix the name with an abbreviation to remind you of the type of object eg lb for label. Therefore a label that will contain a result could be called lbResult.
This page has been left blank for you to make notes about what the different properties do as you use them.
The Project Explorer and The Form Layout Windows

The Project Explorer window shows how the project is developing and how the different components are linked together.

The Form Layout window allows you to visually position your forms on the display screen in the way you want them to appear at run time. When you position the mouse cursor in the window it will change to a cross symbol. This will allow you to drag the forms into different positions on the screen.
Program 5

Write the following program to change a message.

1. Open a new project file
2. Make sure Form1 is selected and in the Properties window change the Caption property for Form1 to Change a Message
3. Click on the Label control in the toolbox and then use the mouse button to draw a rectangle in the upper part of the form.
4. In the label’s Name Property, change it from label1 to lblMessage
5. Set the label’s Caption property to blank
6. Change the label’s BorderStyle property from 0 to 1
7. Set the label’s Font property to 14 Bold
8. Drag a Command button from the toolbox and draw a rectangle at the bottom left of the form.
9. Change the Command Button’s Caption property to Good Message and change it’s Name Property to cmdGoodMessage
10. Place a second command button at the bottom right of the form. Change it’s Caption property to Bad Message and change it’s Name Property to cmdBadMessage
11. Double click on the Good Message command button to get the code template for its Click event.
12. Where the cursor is located, type the code line
   
   lblMessage.Caption = “I LIKE Visual Basic”

   The equal sign means store or assign the text inside the quotation marks in the Caption property of the label.
13. Repeat steps 11 and 12 for the Bad Message command button, changing the message to read “I HATE Visual Basic”.
14. Run the program, trying out the two buttons to see the effect.
15. Save the project and form files using the same name.

Now try the following:

1. Write a program that changes to title of the form to Welcome to Visual Basic when a command button is clicked
2. Place two command buttons on a form, set their captions to Hide and Show. Clicking the Show button should display a label with the message Hello. The Hide button should remove the message. Use the label’s Visible property.
3. Write a program so that the colour of the words in a text box change to red or blue when a command button is clicked. You will need to set a text box property to allow word wrapping in the text box and also assign vbRed or vbBlue to the property of the text box as appropriate.
Designing the Interface

There are many controls used in an interface. We have used two so far – the label and the command button. We will now look at some others:

The Text Box

These are used for entering and displaying data. The SetFocus property positions the cursor inside the text box.

The List Box

This control allows you to provide a list of items that can be selected from. The list usually appears on the left and the items selected are transferred to the right. A common use of this is the Query Wizard in Microsoft Access.

The most commonly used properties of the List Box are:
- List – contains all the items of the list box which are automatically numbered by Visual Basic (from 0 upwards)
- ListCount – The number of items in the list box
- ListIndex – Records the number of the item currently selected
- Sorted – to define whether or not the list is sorted
- Text – the currently selected item from the list box
- AddItem – adds an item to the list box
- Clear – empties the list box
- RemoveItem – removes single items form the list box

The Combo Box

This is a combination (combo) box as it is used to combine the features of the text box and the list box, using the same properties and methods. The main differences between combo and list boxes are:
- Some types of combo box allow you to edit the items displayed or enter an item that is not listed
- The standard combo box must be clicked to reveal its items, so taking up less space on the form.

The list of Font sizes available in many Microsoft applications is a combo box that allows you to type in a font size that is not listed.

The Option Button

This allows you to select one option form two or more. Amongst the properties for the Option button, the most commonly used are Name and caption but the most useful is Value. This holds the value True is if selected and False if not.
**The Check Box**

Similar to the Option button in that you can check its value to see if it is selected. However, you can select more than one at a time. A common use is to select a group of options to define the default values you require at that time. The check box uses 1 and 0 instead of True and False.

**The Frame**

This is a rectangular shape usually placed around other controls. For example, you might require two groups of option buttons. To make them easier to group each group would be put into a frame. The advantages of a frame are that you can reposition all the controls inside it by moving the frame only; you can show or hide all the controls by showing or hiding the frame; it allows you to choose an option from inside each frame.

**The Timer**

This control is only visible on the form when developing and is hidden at run time. It only function is to check the system clock. Two important properties are:
- **Enabled** – This holds as True or Flase. When enabled it can carry out what has been coded in the Timer event. The timer event will occur according to the interval that has been set.
- **Interval** – measured in milliseconds where a value of 1000 represents 1000 milliseconds or 1 second.

**The Scroll Bars**

The scroll bars can be horizontal or vertical and are used to increase or decrease a value. The start and finish values are represented by the two ends of the bar.

Useful properties of scroll bars are:
- **Max** – highest value represented by the right side or bottom of the scroll bar.
- **Min** – lowest value represented by the left side or top of the scroll bar
- **Value** – the number represented by the current position of the scroll box.
- **Change** – the code initiated after the user has moved the scroll box resulting in a change in the value
- **Scroll** – code that is triggered as the user moves the scroll box
Program 6

1. Start a new project
2. Change the Caption for Form1 to Using a list box
3. Drag a text box, three command buttons and a list box onto the form as shown below.

![Form diagram]

4. Set the properties for these items as recorded in the following table:

<table>
<thead>
<tr>
<th>Control</th>
<th>Property</th>
<th>Property Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text Box</td>
<td>Name</td>
<td>txtCountry</td>
</tr>
<tr>
<td></td>
<td>Text</td>
<td>Blank</td>
</tr>
<tr>
<td>Command button</td>
<td>Name</td>
<td>cmdAdd</td>
</tr>
<tr>
<td></td>
<td>Caption</td>
<td>Add</td>
</tr>
<tr>
<td>Command button</td>
<td>Name</td>
<td>cmdDelete</td>
</tr>
<tr>
<td></td>
<td>Caption</td>
<td>Delete</td>
</tr>
<tr>
<td>Command button</td>
<td>Name</td>
<td>cmdClear</td>
</tr>
<tr>
<td></td>
<td>Caption</td>
<td>Clear</td>
</tr>
<tr>
<td>List box</td>
<td>Name</td>
<td>lstCountries</td>
</tr>
<tr>
<td></td>
<td>Sorted</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>List</td>
<td>Spain</td>
</tr>
<tr>
<td></td>
<td></td>
<td>USA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Austria</td>
</tr>
</tbody>
</table>

5. Bring up the code template for the Add command button and insert the following code:

```vbnet
lstCountries.AddItem txtCountry.Text ' adds the contents of the text box to the list box
txtCountry.Text = " " 'clear the text box
txtCountry.SetFocus ' put cursor in text box ready for next country
```

6. For the delete command button, enter the following code:
ListCountries.RemoveItem ListCountries.ListIndex ‘delete the selected item from the list of ‘displayed items

7 For the clear command button, enter the following code:
ListCountries.Clear ‘ delete all items displayed in the list box

8 Run the program and try adding and removing countries from the list.

Program 7

1 Start a new project
2 Change the Form1’s Caption property to Option buttons, check boxes and frames
3 Place a Frame Control box in the top left of Form1 and change the Caption to Gender
4 Place two option buttons in the frame. Change their names to optMale and optFemale and their Captions to Male and Female appropriately. The options buttons must be created in the frame to belong to it – they cannot be dragged in from another area of Form1.
5 Try moving the frame – the option buttons should move with it.
6 Place a label below the frame with its Name set to lblSelection, a blank Caption and a BorderStyle of 1-Fixed Single.
7 Bring up the code template for the male option button and insert the following code:
   LblSelection.Caption = “You selected Male”
8 Repeat 7 for the female option button with the text “You selected Female”.
9 Run the program and check the option buttons work correctly.
10 Place another Frame on the form with the Caption Age.
11 Place three option buttons in the frame, keeping the default Names but changing the Captions to Under 20, 20 to 60 and Over 60.
12 Run the program and make sure you can select one option from each frame.
13 Place a third frame on the form and change the Caption to Replies.
14 Place three check boxes in this frame with the Captions set to Drives a car, Owns a house and Owns shares
15 Run the program and make sure you can select any number of check boxes.
Program 8

1. Open a new project
2. Drag two labels onto the form. Set the Caption of the left label to The time is.
3. Delete the Caption of the right label, set its name to lblTime and its Style property to 1 – Fixed Single.
4. Place a timer control anywhere on the form and set its Name property to tmrTimer and its Interval property to 1000 (1 second).
5. Double click the Timer to bring up the code template and enter the following:
   ```vba
   lblTimer.Caption = Time
   ```
   This will display the current time and update it every second.
6. Run the program.

Using Colour

There is a Visual Basic function called RGB (red, green, blue) which provides a colour by mixing proportions of these three primary colours. Each colour can have a value from 0 to 255, so the code

```vba
Form1.BackColor = RGB(255, 0, 0)
Form2.BackColor = RGB(0, 255, 0)
Form3.BackColor = RGB(125, 125, 125)
```

Will set Form1’s BackColor property to red, Form2’s to green while Form3 gets equal amounts of all the colours to produce grey. The numbers given to the function are called arguments or parameters.

Remember, Visual Basic is American and uses the American spellings for words such as colour.

Program 9

1. Open a new project
2. Place three labels vertically down the left side of the form with a scrollbar to the right of each one.
3. Change the Captions of the labels to Red, Green and Blue respectively.
4. Name the scroll bars hsbRed, hsbGreen and hsbBlue as appropriate.
5. Set the Max property of each scrollbar to 255 (the Min property defaults to 0).
6. Click on the View Code button in Project Explorer. From the drop down list at the top left select hsbRed.Delete the default Change event template and from the right hand drop down list select Scroll to get the correct code template.
7. Ensure the code is as follows:
   ```vba
   Private Sub hsbRed_Scroll
   End Sub
   ```
   Repeat the above for hsbBlue and hsbGreen
8. Run the program and explore the colour changes.
**Program 10**

1. Start a new project and place two text boxes, 4 labels and a button on the form as shown below:

![Form with controls](image)

2. Set the properties for these items as recorded in the following table:

<table>
<thead>
<tr>
<th>Control</th>
<th>Property</th>
<th>Property Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text Box 1</td>
<td>Name</td>
<td>txtFirst</td>
</tr>
<tr>
<td></td>
<td>Text</td>
<td>Blank</td>
</tr>
<tr>
<td>Text Box 2</td>
<td>Name</td>
<td>txtSecond</td>
</tr>
<tr>
<td></td>
<td>Text</td>
<td>Blank</td>
</tr>
<tr>
<td>Label 1</td>
<td>Name</td>
<td>lblFirst</td>
</tr>
<tr>
<td></td>
<td>Caption</td>
<td>First Number</td>
</tr>
<tr>
<td></td>
<td>Font</td>
<td>14</td>
</tr>
<tr>
<td>Label 2</td>
<td>Name</td>
<td>lblSecond</td>
</tr>
<tr>
<td></td>
<td>Caption</td>
<td>Second Number</td>
</tr>
<tr>
<td></td>
<td>Font</td>
<td>14</td>
</tr>
<tr>
<td>Label 3</td>
<td>Name</td>
<td>lblResultName</td>
</tr>
<tr>
<td></td>
<td>Caption</td>
<td>Result</td>
</tr>
<tr>
<td></td>
<td>Font</td>
<td>14</td>
</tr>
<tr>
<td>Label 4</td>
<td>Name</td>
<td>lblResult</td>
</tr>
<tr>
<td></td>
<td>Caption</td>
<td>The result shows here</td>
</tr>
<tr>
<td></td>
<td>Font</td>
<td>14</td>
</tr>
<tr>
<td>Command</td>
<td>Name</td>
<td>cmdAdd</td>
</tr>
<tr>
<td></td>
<td>Caption</td>
<td>Add the numbers</td>
</tr>
<tr>
<td></td>
<td>Font</td>
<td>14</td>
</tr>
</tbody>
</table>
3 Now add the following line of code to the button’s Click event:

```
LblResult.Caption = Str(Val(txtFirst.Text) + Val(txtSecond.Text))
```
The code will take the values typed into the two text boxes, add them together and then output them in the Result label.

4 Run your application and test it by inputting various numbers in the two text boxes and clicking on the button to get the result.

**Essential Elements**

Some essential terms to understand in Visual Basic are as follows:

- **Statement**
  This is an instruction that performs an action and is also known as a command. For example, Beep sounds the computer’s speaker; FileCopy copies one disk file to another.

- **Functions**
  These are instructions that return a value. For example, the function Now returns the current date and time.

- **Variables**
  These are words that store a value. For example, the line
  ```
  Myvar = “Visual Basic”
  ```
  stores the string of characters in a variable called myvar.

- **Operators**
  These are the arithmetical operators like “+”, “-” and “=”.
  Multiplication uses the symbol “*” and division uses the symbol “/”.

- **Objects**
  Objects can be visible like forms or buttons or invisible like the timer.

- **Properties**
  These are the characteristics of an object.

- **Methods**
  These are actions that an object can perform. For example, forms have a hide object which makes them invisible.

**Variables**

A variable is a storage word whose contents can change (vary). One of the important characteristics of a variable is its scope. The scope determines which parts of a program can inspect or change the value of the variable.

To declare a variable is to tell Visual Basic something about it – usually its name or identifier and its data type. Variables can be declared within a subroutine and can then
only be used there. If they are declared in the Declarations section of a form using Dim or Private, then any routine within the form can use it. If they are declared in the Declarations section using the keyword Public, then they can be used anywhere in the project. Variables such as these are called public or global variables.

A good rule is always to declare a variable before using it. If you don’t, then your code is more likely to contain errors that are hard to spot. This can be done in two ways:

1. Write Option Explicit before declaring your variables eg
   Option explicit
   Dim Number as Integer
   Dim ……………………

2. Select Tools/Options and click the check box Require Variable Declaration and then Visual Basic will automatically put Option Explicit at the head of the Code window.

Another important characteristic of a variable is its type. This determines what kind of information it can hold. A string variable holds a series of characters and is useful for names and addresses but is no use for calculations. If you want to store numbers and perform calculations you need an integer variable. A variable declared as a variant type can hold data of any type, and Visual Basic will assume this type if no other type is specified.

To declare a string variable, you would use the code:
   Dim MyVar As String

Although using variants is easier, using specific data types helps to prevent errors and allows the computer to work more efficiently.

Available data types are:

- **String** – a sequence of up to 2 billion characters
- **Integer** – a whole number from −32,768 to +32,767
- **Long** – a whole number form −2,147,483,648 to +2,147,583,647
- **Single** – holds positive and negative numbers, including fractional numbers, in a special format known as floating point which allows for decimal places.
- **Double** – as single, but up to twice the size
- **Currency** – this is known as a scaled integer as it can hold fixed point numbers of up 15 digits to the left of the point and 4 to the right.
- **Date** – holds dates from January 1 0100 to December 31 9999 as well as time information.
- **Boolean** – has two possible values – true and false.
**Program 11**

1. Open a new project and call it Calculator.
2. On the blank form, place two labels and text boxes for inputting two numbers; four buttons for the mathematical operations and a label for the answer (called lblResult). The layout should be similar to this:

   ![Calculator Form](image)

3. Double click on the add button and enter the following code:
   ```vb
   lblResult.Caption = "Numbers add to: " + Str(Val(Text1.Text) + Val(Text2.Text))
   ```
4. Repeat, with appropriate changes, for the other buttons.
5. Run the program to try it out.

**If Statements**

We use the equivalent of If statements all the time. For example, we might say “If it is Saturday then I can have a lazy day otherwise I have to go to work”. This can be translated into Visual Basic code as:

```vb
If (Weekday(Now)) = vbSaturday Or (Weekday(Now)) = vbSunday Then
  lblMessage.Caption = "Relax – it’s the weekend"
Else
  lblMessage.Caption = "Today is a working day"
End If
```

Enter this code into the form load event. Place one label on the form. Run it and see it working.

The `Now` function contains the current date and time, while the command `Weekday` converts it into a number representing the day of the week. `vbSaturday` and `vbSunday` are constants so you do not have to know how the weekdays are numbered.
Program 12

1. Open a new project and bring up the Load event code template for the form. Enter the following code:

   Dim Mark as Integer
   Mark = InputBox(“Enter an exam mark from 0 to 100”)
   If Mark >= 60 Then
     MsgBox “Merit”
   ElseIf Mark >= 40 Then
     MsgBox “Pass”
   Else
     MsgBox “A mark of “ & Mark “ is a Fail”
   End If

In the above code, InputBox and MsgBox provide the appropriate windows for you, so you do not have to use labels and text boxes etc.

ElseIf provides a multiple If statement for use when there is more than one condition to be tested in the If statement.

Testing multiple conditions

Sometimes you need to test two things to select the items you need and to do this you need to use the logical operators ‘AND’ and ‘OR’.

An AND condition

When you test a condition, it returns a value True or False. When using the AND condition, both conditions must be true for the whole condition to be true. If just one of them is false, then the overall condition is false.

Therefore, the code to select all females over 18 for entry in a night club would be:

   Dim Age As Integer
   Dim Gender As String
   Age = txtAge.Text
   Gender = txtGender.Text
   If (Age >= 18) And (Gender = “F”) then
     MsgBox “Allow into nightclub”
   Else
     MsgBox “Do not allow into nightclub”
   End If