Visual Basic

Visual Basic is used to create applications for Microsoft Windows. It includes tools that allow a programmer to create an application that has features similar to other Windows applications without having to write many lines of code.

Visual Basic is based on the BASIC programming language developed in the 1960s by John Kemeny and Thomas Kurtz at Dartmouth University. BASIC stands for Beginner's All-Purpose Symbolic Instruction Code and was used by Kemeny and Kurtz to teach programming to their students.

In 1975, Bill Gates and Paul Allen developed a version of BASIC especially for the Altair personal computer. With the success of this new version of BASIC, Gates and Allen founded Microsoft Corporation. BASIC then evolved to QuickBasic, a structured language that made programming easier for the rapidly growing number of personal computer users. In 1985, the Windows GUI (Graphical User Interface) was introduced. In 1992, Microsoft used QuickBasic and a program called Ruby to develop Visual Basic, an *object-oriented programming* environment for creating Windows programs.

Visual Basic programs are **event-driven**. An *event* is a way in which the user can interact with an object, such as using the mouse to click on a button in a dialog box. An *event-driven program* waits for an event to occur before executing any code, then only code for the current event is executed.

Getting Started

1. **Start Visual Basic** by clicking on the **Microsoft Visual Basic Express Edition on the Program menu or** [shortcut icon] **Shortcut**.

2. The next page to show up is the Start Page.
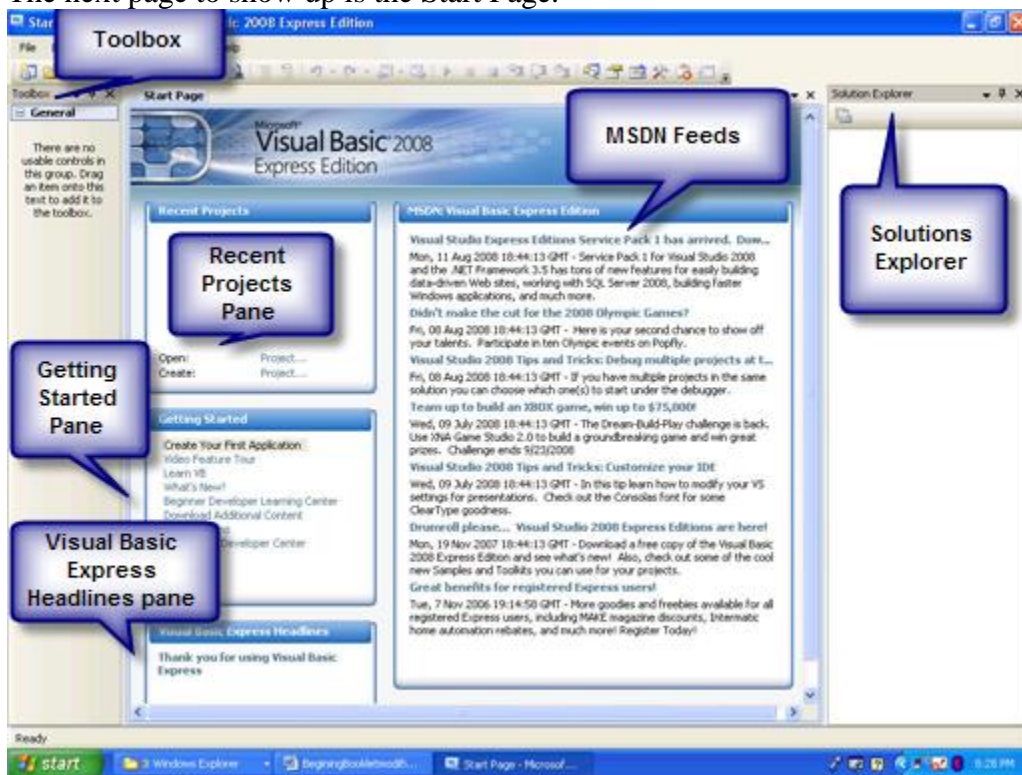
The start page contains the following information:

Toolbox: When expanded the toolbox contains all the controls for creating the applications.

Recent Projects Pane: Lists all the recent projects that have been opened.

Getting Started Pane: Allows the user to get help with projects by hyper linking to programmers or seeing a list of how to articles.

Visual Basic Express Headlines Pane: This is where you receive up to date information about VB from Microsoft if you have a live internet connection.

**MSDN Feeds:** By default this feed will come from the MSDN VB 2008 Express RSS feeds. This feed can be changed through the tools and options page.

Solutions Explorer: This features lists the files and components in your project.

3. Once you click on Create new project the following templates screen will appear. We will be using the Windows Form Applications for our projects.
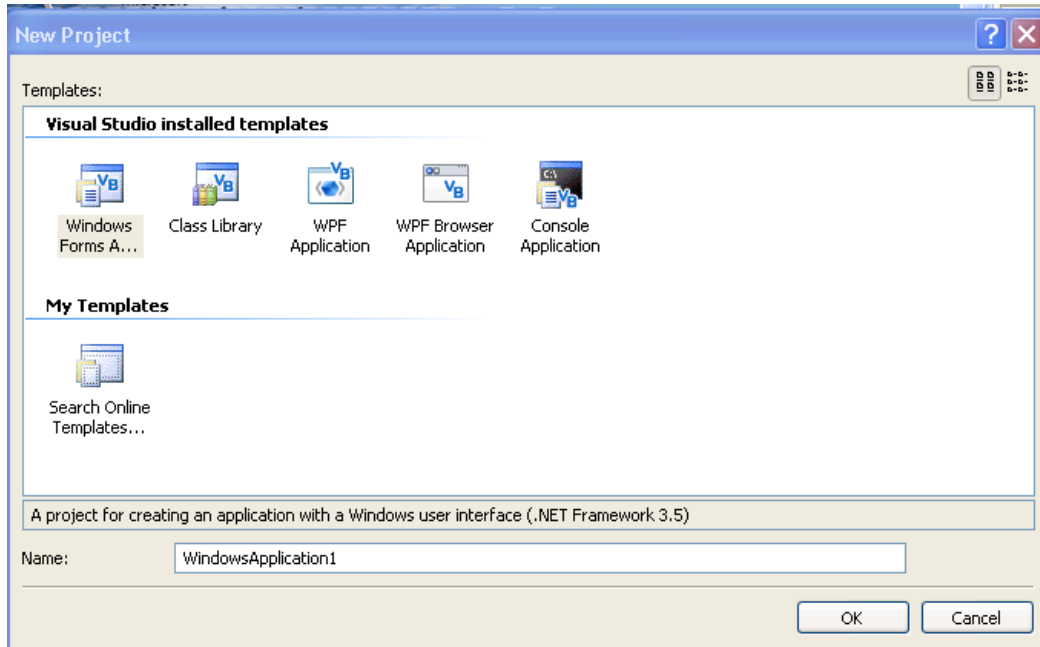
4. Your screen should look similar to the picture below. Examine the various parts. If any of them are missing, they can be restored by clicking on View—Then selecting the appropriate part. If the Form Windows does not show, double click on the Form name in the Project Window. Although these Windows can be moved around the screen, it is advisable to leave them in the configuration shown below.
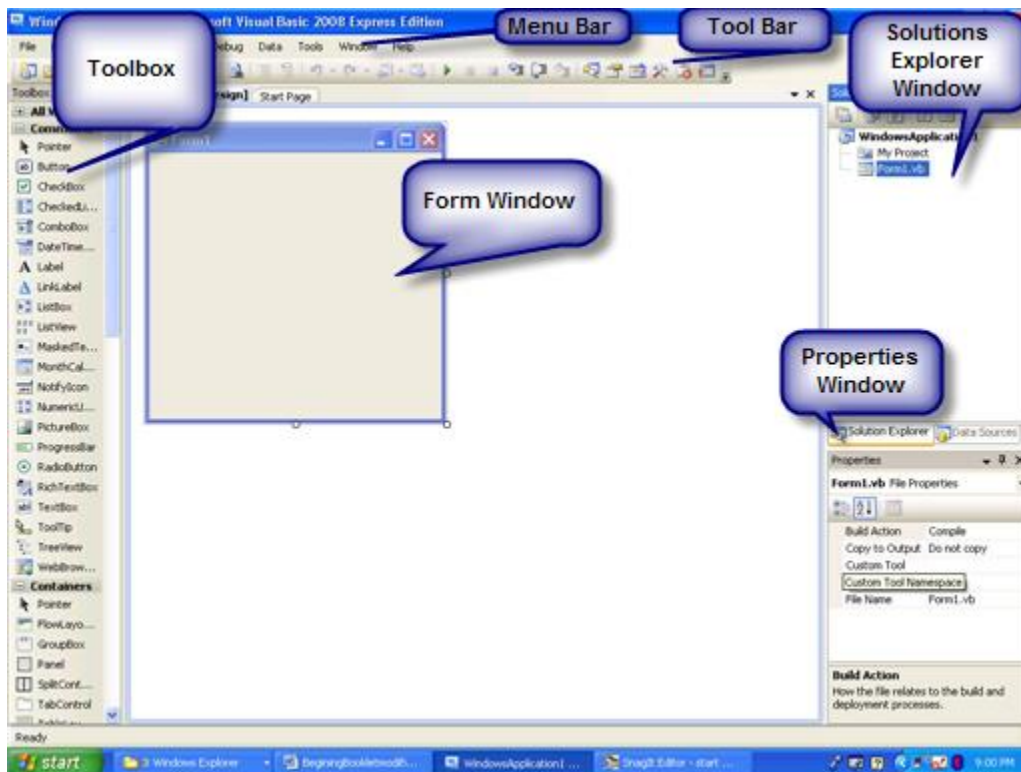

The Form Window

This is where you design the Windows for your program. You can add additional forms by clicking on the New Form icon (or selecting New Form from the File menu).

The Solutions Explorer Windows (The Project Window VB6)
Forms are saved as projects. If you need only one form in your program, then that project will contain only a single form. The project window lists the form names used in your project.

The Toolbox As you can see there are many different controls that can be used in a VB project. The most commonly used one are boxed in red.
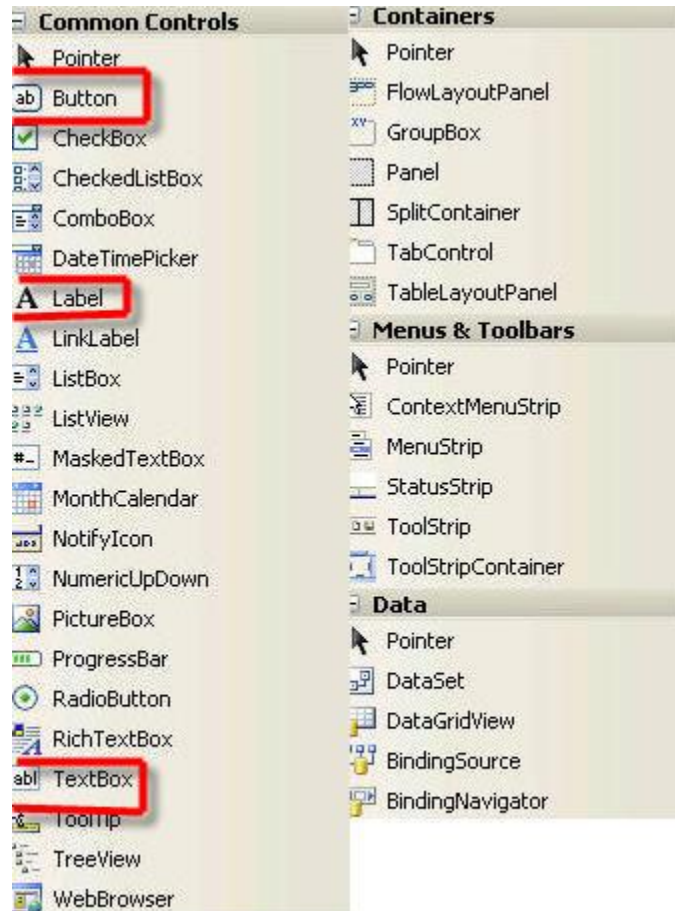
You can select various tools and put them on the form. This can be done by clicking once on the object and then clicking and dragging across a section on the form, or by double clicking on the tool and then moving it to the point you wish. By clicking on the object on the form, you will get handles which indicate that that tool is the active one and you can then resize it, move it, or set properties on it.

The Properties Window Here is an example of two different objects with some of their properties. Some are the same while others are totally different.

Properties — Label1 System.Windows.Forms.Labe		Properties — TextBox1 System.Windows.Forms.T	
⊞ (ApplicationSetti		⊞ (ApplicationSetti	
⊞ (DataBindings)		⊞ (DataBindings)	
(Name)	**Label1**	(Name)	**TextBox1**
AccessibleDescri		AcceptsReturn	False
AccessibleName		AcceptsTab	False
AccessibleRole	Default	AccessibleDescri	
AllowDrop	False	AccessibleName	
Anchor	Top, Left	AccessibleRole	Default
AutoEllipsis	False	AllowDrop	False
AutoSize	**True**	Anchor	Top, Left
BackColor	☐ Control	AutoCompleteCt	**(Collection)**
BorderStyle	None	AutoCompleteMc	None
CausesValidatior	True	AutoCompleteSc	None
ContextMenuStr	(none)	BackColor	☐ Window
Cursor	Default	BorderStyle	Fixed3D
Dock	None	CausesValidatior	True
Enabled	True	CharacterCasing	Normal
FlatStyle	Standard	ContextMenuStr	(none)
⊞ Font	Microsoft Sans Se	Cursor	IBeam
ForeColor	■ ControlText	Dock	None
GenerateMembe	True	Enabled	True
Image	☐ (none)	⊞ Font	Microsoft Sans Se
ImageAlign	MiddleCenter	ForeColor	■ WindowText
ImageIndex	☐ (none)	GenerateMembe	True
(ApplicationSettings)		**(ApplicationSettings)**	
Maps property settings to an application configuration file.		Maps property settings to an application configuration file.	

One of the characteristics of an object is that it has a "state." This state is made up of a number of properties. Visual Basic lists the properties of objects in a Properties window. At any one time, the window shows the properties of only one object. First click on the object to select it, then go to the properties window and the properties for that object are displayed. If no object is selected, then the properties for the form are displayed.

Placing the Objects

Textboxes—These are boxes that contain text. A textbox can display numbers, letters, or a mixture of both. You use textboxes to accept INPUT from people running your program.

Labels—You use a label object to tell something to people running your program—in other words, to print something or give directions.

Buttons – The button controls are buttons that users click as they are running your program. When a user clicks on one of these buttons, something (an action) happens. The caption of a button, such as "Exit" or "Quit," explains what that action will be.

Naming Conventions

The way you name the controls placed on a form is important because the names are used to refer to the controls in the program code. Meaningful names make understanding the program code easier.

Textbox names start with the prefix "txt". The prefix for labels is "lbl", for Command (Formerly cmd) Buttons, "btn", and for forms "frm". The prefix is followed by one or more words describing the function of the control. No spaces are used. Each new word is capitalized. Here are some examples:
> txtLastName
> lblBirthPlace
> txtUnpaidBalance
> btnDisplayPicture
> frmDisplay

Here is the list of prefixes:

Prefix Abbreviations for Control Names

Prefix	Control
cbo	Combo Box
chk	Check Box
btn	Button
dir	Directory list box
drv	Drive list box
fil	File list box
fra	Frame
frm	Form
grd	Grid
hsb	Horizontal scrollbar

img	Image
lbl	Label
lin	Line
lst (lower case L)	List box
mnu	Menu
ole	OLE client
opt	Option button
pic	Picture Box
shp	Shape
tmr	Timer
txt	Text box
vsb	Vertical scrollbar

Setting Properties for Forms,Textboxes, Labels, and Commands

Labels
Text—This is the text that the label will display
Name—The name of the label—eg. lblBirthplace—The Name is used to identify the label in code.

Textboxes
Name—The name of the Textbox—eg. txtLastName—used in code
Text—The text that you wish to see displayed, if any. Often this is just deleted as the user will fill in the textbox when the program is run.

Commands / Buttons
Text—Words you wish to see displayed on the command button.
Name—the name of the button—eg btnExit—used in code

Forms
Name—the name of the form—eg. frmTitle—used in code. This name will appear in the Project Window. It will also be used as the filename when you save the form.
Text—This text will appear in the Title Bar of the form.

Steps in Creating a Program

This is a **three-step process**.
1. Select the object and arrange on the form.
2. Set the properties of the form and the objects.
3. Write the code.

To Write the Code—Attaching Actions to Objects

Double click on the object to get to the Code Window. Notice that Sub and End Sub are already there for you. Insert your code in between. For example, the code for the Exit button would be **Me.Close**. This command should be indented to make your code easier to read.

Double clicking on the form name in the Project Window will return you to the form. Then just double click on the next object you wish to attach code to.

Our First Program

Let's write a program to display "**Play Ball**" in large red letters when we hit a command button called **OK**.
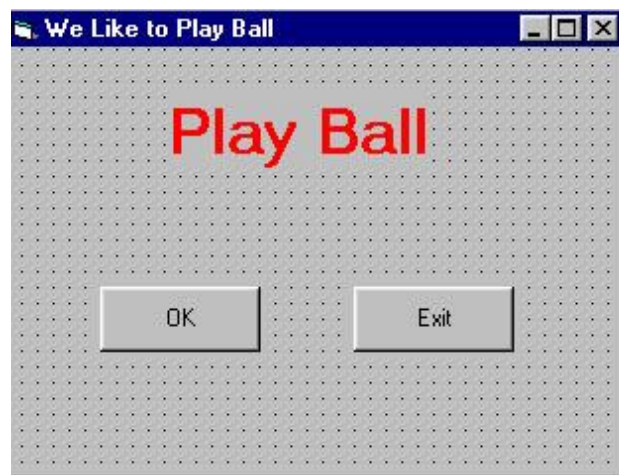
Step 1—Place a **label** on the form, as shown, large enough to hold the words **Play Ball**. Place 2 **buttons** on the form as shown.

Step 2--Set the properties for the form and the two objects as follows:

Form   Set **text** to **We Like to Play Ball**
                  Set **Name** to **frmPlay**

Label   Set **text** to **Play Ball**
                  Set **name** to **lblPlayBall**
                  Set the **font** to **MS Sans Serif Bold 24 pt**
                  Set **forecolor** to **red**
                  Set **visible** to **false**
                  Set **Autosize** to **true**

1st Button	Set **name** to **btnOK**
	Set **text** to **OK**
2nd Button	Set **name** to **btnExit**
	Set **text** to **Exit**

Step 3—Write the code. **Double click** on the **Exit** command. Tab to indent, and insert this code between the sub and end sub commands:

> **Me.close**

Close the code window to return to the form, and **double click** on the **OK** command. Enter this line of code between the sub and end sub commands.

> **lblPlayBall.Visible = True**

<u>Run your program.</u> Test the OK and Exit buttons.

<u>Saving a Project</u>—Select **File, <u>Save Project As</u>**

<u>Select your own folder on the fileserver with your login name on it. For each Visual Program you wish to save, create a new folder within your folder in which to store all the files associated with your project. This is necessary because most programs consist of several files and it is important to keep them together.</u>

You will first be asked to save the form—Yes—The default filename should be frmPlay.—Make sure to put it in a new folder within **your folder**.
Then you will save the project—Yes—Name it **PlayBall** and put it in **the same folder**.

Using Variables and Making Calculations

<u>Using Variables</u>
Visual Basic programs are designed to deal with a large amount of data stored in the computer. This data is not entered by you the programmer. Instead, it comes from:
- Users entering information

- Data gathered as the program runs

A good variable name should reflect something about the information it represents. There are some rules to follow as you create variable names. These names may be up to forty characters (letters or numbers) long. You should also follow the Intercap method to name variables, using capital letters to show how words or portions of words are put together.

Eg.   *SpeedOfTrain*
LastName
DateOfBirth

Variable Declaration

This means you will have to declare any variables before you use them in the appropriate sub program. When you declare a variable you are telling the program <u>the name of the variable</u> and <u>the type of data</u> it will hold.

Eg.   *Dim LastName as String*
Dim Amount as Currency
Dim Quantity as Integer
Dim Degrees as Single, Frequency as Double

Types of Data

In the beginning, we will use three types of data. If you declare a variable incorrectly, you will get an error message when you try to run the program.

String   This type of variable is used to store words. More correctly, it could be defined as anything inside quotes.
Eg. "Miramichi" or "******" are both strings.

Single   This type of variable is used to store numbers, including decimal places, to an accuracy of about 8 digits.

Integer   This type is used to store whole numbers only, in the range –32,768 to +32,767

The Other Visual Basic Data Types

Datatype	Description and Range
Boolean	A Datatype that takes on one of two values only: True or False
Byte	Positive numeric values without decimals that range from 0 to 255.
Currency	Data that holds dollar amounts from -$922,337,203,685,477.5808 to 922,377,203,685,477.5807. The four decimal places ensure that proper rounding can occur.
Date	Holds date and time values. The date can range from January 1, 100, to December 31, 9999.

Decimal	A new Datatype not yet supported in Visual Basic except in a few advanced situations. The **Decimal** Datatype represents numbers with 28 decimal places of accuracy.
Double	Numeric values that range from -1.79769313486232E+308 to 1.7979769313486232E+308. The Double Datatype is often known as double-precision.
Integer	Numeric values with no decimal point or fraction that range from -32, 768 to 32,767.
Long	Integer values with a range beyond that of Integer data values. Long data values range from -2,147,483,648 to 2,147,483,647. Long data values consume more memory storage than integer values, and they are less efficient.
Object	A special Datatype that holds and references objects such as controls and forms.
Single	Numeric values that range from -3.402823E+38 to 3.402823E+38. The Single Datatype is often called single-precision.
String	Data that consists of 0 to 65, 400 characters of alphanumeric data. Alphanumeric means that the data can be both alphanumeric and numeric. String data values may also contain special characters such as ^, %, and @. Both fixed length strings and variable – length strings exist.
Variant	Data of any Datatype, used for control and other values for which the Datatype is unknown.

Making Calculations

Assignment Statements

A value is assigned to a variable. Only the variable being assigned the value can appear on the left side of =. The calculations (expressions) appear on the right side.

Eg.

Interest = Principal * Rate * Time
AvgSpeedPerHour = DistanceTravelled / Hours
TestAverage = (Test1 + Test2 + Test3)/3
Area = 3.14 * Radius ^ 2

Order of Operations

In writing assignment statements, it is important to understand the order of operations (the order in which calculations will be performed).

1.	Whatever is inside brackets	( )	
2.	Exponents	5 ^2 would be 5 squared	
3.	Multiplication and Division	*	/
4	Addition and Subtraction	+	-

Go to **http://msdn.microsoft.com/en-us/library/fw84t893(VS.80).aspx**
for the complete list of operators.

A Program with Calculations

This Program will Calculate the TotalHours you have slept in a Lifetime—assuming 8 hours per day

First, design a form similar to the one below. You should have 3 labels (for the words written and to display the answer), 1 textbox (to enter your age), and 2 buttons.

Next, Set the Properties as shown in the table below:

Form	Name	frmAge
	Text	Determine Hours Slept in Lifetime
Label1	Text	Enter Your Age
	Font	18
Textbox	Name	txtAge
	Text	(delete Text1)
Label2	Text	Hours spent Sleeping
	Font	14
Label3	Name	lblTotalHours
	Text	(delete the Text)
	Border Style	1—Fixed Single
Button1	Name	btnCalculate
	Text	Calculate
Button2	Name	btnExit
	Text	Exit

Next, enter the code for the Buttons—Remember, double click on the button to take you to the code section.

Private Sub btnCalculate_Click()

```
Dim Age As Single, HoursPerDay As Single
Dim TotalHours As Single, DaysPerYear As Single
Age = Val(txtAge.Text)
HoursPerDay = 8
DaysPerYear = 365
TotalHours = Age * HoursPerDay * DaysPerYear
lblTotalHours.Text = TotalHours
txtAge.Focus()
```

End Sub

Private Sub btnExit_Click()
Me.close
End Sub

Run the Program
You should now be able to enter your age (or someone else's). If you then click on the Calculate button, it should display the total numbers of hours spent sleeping.

Add Another Button to the Form
Add a third command button to the form whose purpose is to clear the textbox and label so that you can enter someone else's age. Name this command **btnClear** and set the caption to **Clear**. The code for the command is

```
txtAge.Clear()
lblTotalHours.text =String.Empty
```

Save Your Program
Select **Save Project As**. Within your folder, create a new folder called **Age**. First save the form—it should be called **frmAge**. Then save the project—call it **Sleep**.

Explanation of Code

- The Dim statements on the first two lines declare the required variables as numeric.

- *Age = Val(txtAge.Text)* takes the Text property from the textbox, txtAge, and assigns it to the variable *Age*. *Val* converts the text property (a text property is a string value) to a numeric value. Think of Val for Value.

- The next two lines just assign values to the variables.

- *TotalHours = Age * HoursPerDay * DaysPerYear* is an assignment statement which performs the calculations on the right side of the equal sign and then assigns that value to the variable *TotalHours*

- *lblTotalHours.Caption = TotalHours* sets the Text property of *lblTotalHours* to the value of the variable *TotalHours*

Visual Basic Tips

To clear a text box:
TextBoxName.Clear()
TextBoxName.Text = String.Empty
TextBoxName.Text = ""

E.g. txtArea.clear()
txtArea.text = string.empty
txtArea.text = ""

To clear a label:
LabelName.text = String.Empty
LabelName.text = ""

lblFinalCost.text =String.Empty
lblFinalCost.text = ""

Math functions such as Square Root can be called through the Math functions.
Math.sqrt(number) – this will take the sqrt of the number in the brackets.

PROGRAMMING PROBLEMS

1. **The Material Problem**
   Create a program to calculate the cost of buying material for a dress. Use a textbox to enter the number of meters of material and another textbox to enter the cost per meters. You will also need labels so the user understands what to enter in the textboxes. Use another label to display the final cost. Include Calculate, Clear, and Exit commands.

2. **The Pet Problem**
   Create a program to calculate and display the cost of buying a pet. Use textboxes to enter values for the following:
   a)   Purchase price of pet
   b)   Veterinary fee for checking pet
   c)   Accessories
   Calculate and display the sum in a label as the total cost.

3. **The Test Average Problem**
   Create a program to calculate and display the average of three test scores. Use textboxes to enter values for the three test scores. Calculate the sum of the scores and divide by three to calculate the average. Display the average in a Label.

4.   **Area of a Circle**
        Create a program to enable the user to enter the radius of a circle and then have the computer calculate and display the area. $A=\pi r^2$

5.   **Length of Hypoteneuse**
        Create a program to enable the user to enter two numbers to represent the sides of a right-angled triangle. The computer will calculate and display the length of the hypoteneuse. $c^2 = a^2 + b^2$

6.   **Square and Square Root**
        Create a program to enable the user to enter one number. The computer will calculate and display both the square and the square root of that number.

7.   **Area of a Trapezoid**
        The formula for the area of a trapezoid is A = ½(sum of the parallel sides)*height. Create a program to let you enter the height, and both parallel sides. Then calculate and display the Area.

8.   **Cost of Carpeting**
        Create a program that does all of the following: (a) allows you to enter the length and width of a room in feet and will calculate the area of the floor in square yards. 3 ft = 1 yd. (b) allows you to enter the cost of carpeting in square yards. The computer will then give you the total cost of the carpeting, tax included.

9.   **Calculate Number of Miles**
        Create a program to allow you to enter any number of kilometres and have the computer calculate and display the equivalent number of miles.
                1 km = .6213 miles

10.   **The Drag Race Problem**
        The distance a racecar undergoing a constant acceleration travels over a given period of time is given by the formula:
                $s=1/2at^2$
        where s is the distance traveled, a is the acceleration, and t is the elapsed time. Create a program that prompts the user to enter the elapsed time in seconds. Assuming the distance traveled is one-quarter mile (1320 feet), calculate and display the acceleration in feet per second squared. Start by solving the equation for a.

11.   **Concrete**
        Create a program to calculate the cost of concrete for a sidewalk. Use textboxes to enter the length and width of the sidewalk in feet. Assume the concrete is poured 4 inches thick. Calculate the volume of the sidewalk by multiplying the length times the width times the depth. Make sure the units agree. Display the volume in both cubic feet and cubic yards. Assume the cost of concrete is 60 dollars per cubic yard. Display the cost of the concrete.