Ch 1 :Introduction-The Basics

Introduction
In this chapter you will investigate the task of developing a program using the programming language VB.Net (2005 Express Edition).

VB.Net 2005 Express Edition provides an easy to use environment for developing programs. The Integrated Development Environment (IDE) provides fascilities to create and debug programs. It is useful to outline all the stages involved in producing a working program to put the process of actually using VB.Net in its correct context.

The structure of a Console Application program
A VB.Net program several subroutines but there must always be a sub called Main( ). All program start at the subroutine called Main( ) – you may call other subroutines any name you wish.

```
Sub Main( )
        {brief description of subroutine}

        {variable declaration }

        {program code}
End Sub
```

The convention is to layout your subroutine as shown above and described below:

Brief description of subroutine	A few word of explanation to assist yourself or others at a later date when maintaining the program.
Variable declaration	A variable (aka identifier) is a named storage location (i.e. Age, Name , Address, etc.) items of data are stored in such locations and used throughout the program code as needed.
Program code	All the instructions to perform the task/program. Will use the identifiers (from the Variable section). It may call other subroutines.

Example1.1
As a simple example to illustrate these ideas, Example 1.1 shows a program which calculates the total cost of a purchased item by calculating VAT and adding it to the price of the item. The algorithm on which the program is based is as follows:

1. *Ask the user to enter the price of the item*
2. *Store the price*
3. *Calculate the VA T at 17.5% (i.e.. multiply the price by 0.175)*
4. *Calculate the total price by adding the VAT to the price*
5. *Display the total cost on the screen.*

Note that the line numbers preceding each line in the program are included for explanation only and are not part of the program code.

Example 1.1

```
1     Sub Main( )
2     ' This program calculates the total cost including VAT of a purchased item.
3     Const VAT = 0.175
4     'variables
5     Dim Price As Decimal
6     Dim Tax As Decimal
7     Dim TotalCost As Decimal
8     'program code
9           Console.Write("Enter price of the item :")
10          Price = Console.ReadLine()
11          Tax = Price * VAT
12          TotalCost = Price + VAT
13          Console.WriteLine("The Total Cost is: {0:c}", TotalCost)
14          Console.ReadKey()
15
16    End Sub
```

Code Analysis

1	Start of subroutine Main ( )
2	Brief comment describing the program
3	Const. A constant is a type of variable but unlike variables its value cannot be changed… I.e. it remains constant throughout the program.
4	Variable declaration section
5	DIM (Declare in Memory) a location named Price. It will be allowed to store decimal values
6	DIM (Declare in Memory) a location named Tax. It will be allowed to store decimal values
7	DIM (Declare in Memory) a location named TotalCost. It will be allowed to store decimal values
8	Program code section
9	Write the letters in quotes " " are written to the console screen. Note Write and WriteLine behave differently. WriteLine will move to the next line on screen when continuing the program. Write will pause on the same line.
10	Readline ( ) allowed data to be read from the user. The input is placed into the variable (memory location) named Price.
11	Tax is calculated by multiplying Price by VAT. The result is placed into location named Tax
12	TotalCost is calculated.
13	The letters in quotes are written to the console screen. Note the brackets used {0:c} The 0 (zero) indicates that the value held in TotalCost should appear here. The c indicates the format of the output. The letter c represents Currency (other letters have different meaning) and a pound sign appears before and two decimal places appear after the decimal point.
14	Readkey ( ) serves only to keep the screen visible until the user presses a key.
16	Ends the subroutine. And the program.

Points to Note

1. It is a good idea to include comment lines to describe the purpose of lines or sections of your program. Particularly true for large, complex programs, this is very helpful if it is necessary to change the program at some later date.
2. Using indentation is important. It can really improve the appearance and the clarity of a program, thus making the program easier to read and understand if it has to be modified later.
3. Take great care when writing code. Programming involves meticulous attention to detail; omitting punctuation marks, including them in the wrong place or making spelling mistakes will usually lead to the compiler reporting syntax errors, but sometimes such slips might cause serious errors which are more difficult to detect. So be very careful to form instructions precisely.

Identifiers and data types
The term identifier is a general term used for variables, constants and other programmer-defined names such as procedures and functions. Variables and constants are always associated with a data type. VB.Net requires that variables are given a type such as integer or decimal so that the necessary amount of memory can be reserved for their use.

Variables
A variable, which represents an item of data such as a single number, has a name and a current value. Variables are given names such as Amount, Total or Numb3 and are assigned values by program instructions. These values are stored in the memory of the computer and they are accessed whenever a variable is referenced in a program instruction. So, for example, in the instruction

```
Total = Price + Tax
```

the value associated with the variable **Price** is added to the value of the variable **Tax** and the sum is then assigned to the variable **Total**. If in a previous instruction **Total** had already been assigned a value, that value would be replaced by the new one.

Constants
Constants too are assigned values but only once after the word const preceding the main program. The constant **VAT** in Example 1 is an example. Constants retain their values throughout the execution of a program; VB.Net does not allow you to use a constant in an instruction which tries to change the value of the constant. Thus, if in Example 1, you included an instruction such as

```
VAT = 0.2
```

in the main program, the compiler would report an error.

Special identifiers and reserved words

Certain words in VB are classed as special, or standard, identifiers because they perform the same function as programmer-defined identifiers but they are recognized by the compiler as being pre-defined and they are themfore to be used only in a certain context. Examples of special identifiers are the words Write, WriteLine Read, ReadLine. lf you use any of these words for identifiers, for example by declaring

```
DIM Read  as integer
```

then VB will not necessarily regard this as a mistake, but you will have overridden the standard definition of Read as an input instruction, and you will have to use it as an integer variable; you will not be able then to we read as an input instruction since, in effect, you will have redefined its function. The moral is to avoid using these special identifier names for your own, programmer-defined identifiers.

Reserved words such as begin, end, real and program are words which are actually part of the VB language and are unavailable for use as identifiers. VB's reserved words and special identifiers are shown below.

Reserved words

And	Ansi	As	Auto	Boolean	ByRef	Byte	ByVal	Call
Case	Catch	Char	Class	Const	Date	Decimal	Declare	Default
Dim	Do	Double	Each	Else	ElseIf	End	Enum	Erase
Error	Event	Exit	False	For	Friend	Function	Get	GoTo
Handles	If	InInherits	Integer	Interface	Is	Long	Loop	Me
Mod	Module	MyClass	Namespace	New	Next	Not	Nothing	Object
On	Option	Optional	Or	OrElse	Private	Property	Protected	Public
ReadOnly	Resume	Return	Select	Set	Shared	Short	Single	Static
Step	Stop	String	Structure	Sub	Then	To	True	Try
Until								

A full list of reserved words can be found in the Help section of VB.Net.

Rules for naming identifiers

VB.Net imposes a number of restrictions concerning the formation of names for identifiers:
1 The name must consist only of alphabetic and numeric characters.
2. The name must start with an alphabetic character.
3. The name must not be a special identifier or a reserved word.

Examples of valid identifiers
firstNum NUMBER1 abc31 Counter x
Examples of invalid identifiers
12abc (starts with a numeric character)
first-number (contains a non-alphabetic/numeric character)
var 1 (contains a space)
End (a reserved word)
READ (a special identifier)

Data types
As well as having a name and values, variables are also given a type. Three commonly used types are integer, decimal and string (character). Data types are declared before the main program using a DIM statement. For variables, the type must be shown after the name of the variable, as illustrated on lines 5-7 of Example 1. More examples of type declarations arc shown below.

```
var
DIM  Amount          as Decimal
DIM  CodeLetter      as char
DIM  Name            as String
DIM  NumberOfItems   as Integer
```

Data Type	Description
Decimal	can be used for numbers such as 123.456, 0.22 or-9.93, that is, *signed* numbers that are not whole numbers. The computer holds decimal numbers in floating-point form so that very large, and very small numbers can be stored.
Integer	Signed whole numbers. Some examples of integers are 23,0,-1. 32767 and 559.
Char	stores a single character such as 'a', 'D', '6' or '?'.
String	A string is simply a number of characters which arc collected together and used as a unit. For example, a person's name is a string of alphabetic characters, and a stock number such as 100-234/ABC in a mail order catalogue is a string containing a mixture of alphabetic, numeric and special characters. String variable declarations are illustrated in the examples below, DIM Surname as String(20) DIM Stocknumber as String(12) The number inside the brackets specifics the maximum number of single characters to be handled by the named variable.
Boolean	This type of variable has only one of two possible values, namely true or False. A boolean variable declaration is made as follows: DIM Morevalues as boolean two reserved words true and false which can be used to assign a value to a boolean variable, as in: Morevalues = true

Basic input and output

Practically every program requires that data are provided by some input device such as a keyboard and that results are produced on an output device such as a monitor.

VB provides a number of instructions to simplify these operations. The example program in Example1 used three input-output instructions, namely

Example 1.2

```
1     Sub Main()
2     ' This program calculates ......
3     Const CENTIMETRESPERINCH = 2.54
4     'variables
5     Dim Centimetres As Decimal
6     Dim Inches As Decimal
7     'program code
8          Console.Write("Enter the length in Inches :")
9          Inches = Console.ReadLine()
10         Centimetres = Inches * CENTIMETRESPERINCH
11         Console.WriteLine()
12         Console.Write("A length of {0:f2} inches", Inches)
13         Console.Write(" is equivalent to {0:f2} centimetres.", Centimetres)
14         Console.ReadKey()

15    End Sub
```

Code Analysis

1	Start of subroutine Main ( )
2	Brief comment describing the program
3	Const. A constant is a type of variable but unlike variables its value cannot be changed. i.e. it remains constant throughout the program.
4	Variable declaration section
5	DIM (Declare in Memory) a location named Centimetres. It will be allowed to store decimal values
6	DIM (Declare in Memory) a location named Inches. It will be allowed to store decimal values
7	Program code section
8	Write the letters in quotes " " are written to the console screen. Note Write and WriteLine behave differently. WriteLine will move to the next line on screen when continuing the program. Write will pause on the same line.
9	Readline ( ) allowed data to be read from the user. The input is placed into the variable (memory location) named Inches.
10	Centimetres is calculated by multiplying Inches by CENTIMETRESPERINCH . The result is placed into location named Centimetres
11	A blank line is written to the screen.
12	The letters in quotes are written to the console screen. Note the brackets used {0:f2} The 0 (zero) indicates that the value held in TotalCost should appear here. The f indicates the format of the output. The letter f represents floating point number (other letters have different meaning) and the 2 indicates two decimal places appear after the decimal point.
13	The letters in quotes are written to the console screen. Note the brackets used {0:f2} see line 12 Note also how two Write statements are joined as a single line on screen.
14	Readkey ( ) serves only to keep the screen visible until the user presses a key.
15	End of subroutine.

Example Input & Output

The following program serves to demonstrate simple input & output in VB.Net
Example 1.3

```
Sub Main()
'This program demonstrates simple input, process, output in VB.Net.

        'variables
        Dim Age As Integer
        Dim Balance As Decimal
        Dim Interest As Decimal
        Dim Name As String
        Dim Gender As Char

        'program code
        'Input data
        Console.Write("Name    :")
        Name = Console.ReadLine()
        Console.Write("Age     :")
        Age = Console.ReadLine()
        Console.Write("Gender  :")
        Gender = Console.ReadLine()
        Console.Write("Balance :")
        Balance = Console.ReadLine()
        'Process
        Interest = Balance * 10 / 100
        'Output
        Console.WriteLine("--------------------")
        Console.WriteLine("Name is    :{0}", Name)
        Console.WriteLine("Age is     :{0}", Age)
        Console.WriteLine("Gender is  :{0}", Gender)
        Console.WriteLine("Balance is :{0}", Balance)
        Console.WriteLine("10% interest is :{0:f2}", Interest)

        Console.ReadKey()

End Sub
```

Exercise

1) Amend the above program to display the Interest as currency (i.e. with a £ sign)

2) Write a program to convert a value given in British pounds to US dollars.
Assume that there are 2.5 dollars to a pound.
Hint: example 1.2 is a similar program.

Test data	Input (£)	Expected Result ($)
	2	5
	5	15
	7.5	18.75

2 A program is required to calculate the cost of carpet based on square metres.
Assume that carpet costs £3.50 per square metre.
Input the **length** and **width** of the room then calculate and display the cost of the carpet.

Test Data		Expected Result
Length	Width	Cost of Carpet
3	4	42
8	6	168
12.25	10.5	450.18