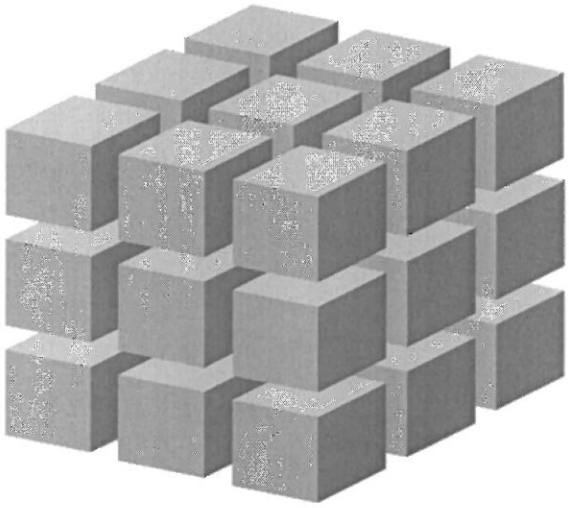
VB .NET 2005/2008 *EE*

Introduction to Console Mode

Practical Programming for AQA Computing
(2008 Specification)

DAVID BETHUNE



computing@zigzageducation.co.uk
zigzageducation.co.uk

F38 3365

POD 5001

Contents

Topic 1 – Getting Started with VB Console Programming

1.1 Introduction

Welcome to AS Level Computer Programming, using Visual BASIC (console mode).

The first thing to understand is that this is a course in **programming**, not a specialised course in Visual BASIC. Visual BASIC is a rather unusual language, with many useful features. Many of these features are very specific to Visual BASIC. In these notes, we will use VB.Net in Console Mode to focus on the standard constructs which you would expect to find in almost any high level programming language. There are (at least) three good reasons for doing this:

- It means that what you learn will fairly easily translate into other languages which you might learn later
- It means that we will focus on the standard constructs that are assessed in the AS Level Computing exam
- It means that we can focus on the principles of software development, rather than the details of Visual BASIC

If you already know some Visual BASIC, there will be times when you think, 'There is an easier way to do this in Visual BASIC'. You might be right, but the reasons for doing things the way chosen in these notes are as described above.

Another set of course notes are available from ZigZag Education which cover AS Level Computer Programming using VB.Net Windows Applications. You may wish to progress to VB.Net Windows Applications once you have completed this course, or you may prefer to study it in parallel. Your teacher or tutor will be able to advise you.

You will see the following icons throughout these notes:

Computer-based practical task – you will need access to a computer with Visual BASIC.Net installed for this task

Questions for you to answer – you can check your own answers against the sample answers given at the end of each topic

Activity (not computer-based) – this will usually require some written work

You should ask your teacher, tutor or lecturer to check your work whenever you complete a computer-based practical task or a non-computer-based activity.

1.2 Learning Outcomes

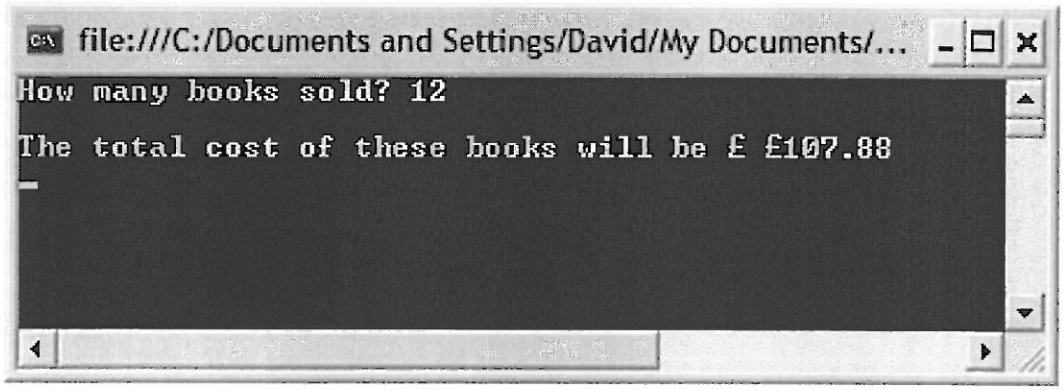
After completing this topic, you should be able to:

- Create a simple VB program
- Save and print a VB program
- Input data using Readline()
- Display messages and data using Console.Write()

1.3 Visual BASIC terminology

A program is simply a list of instructions, written in a programming language, which tells the computer what you want it to do.

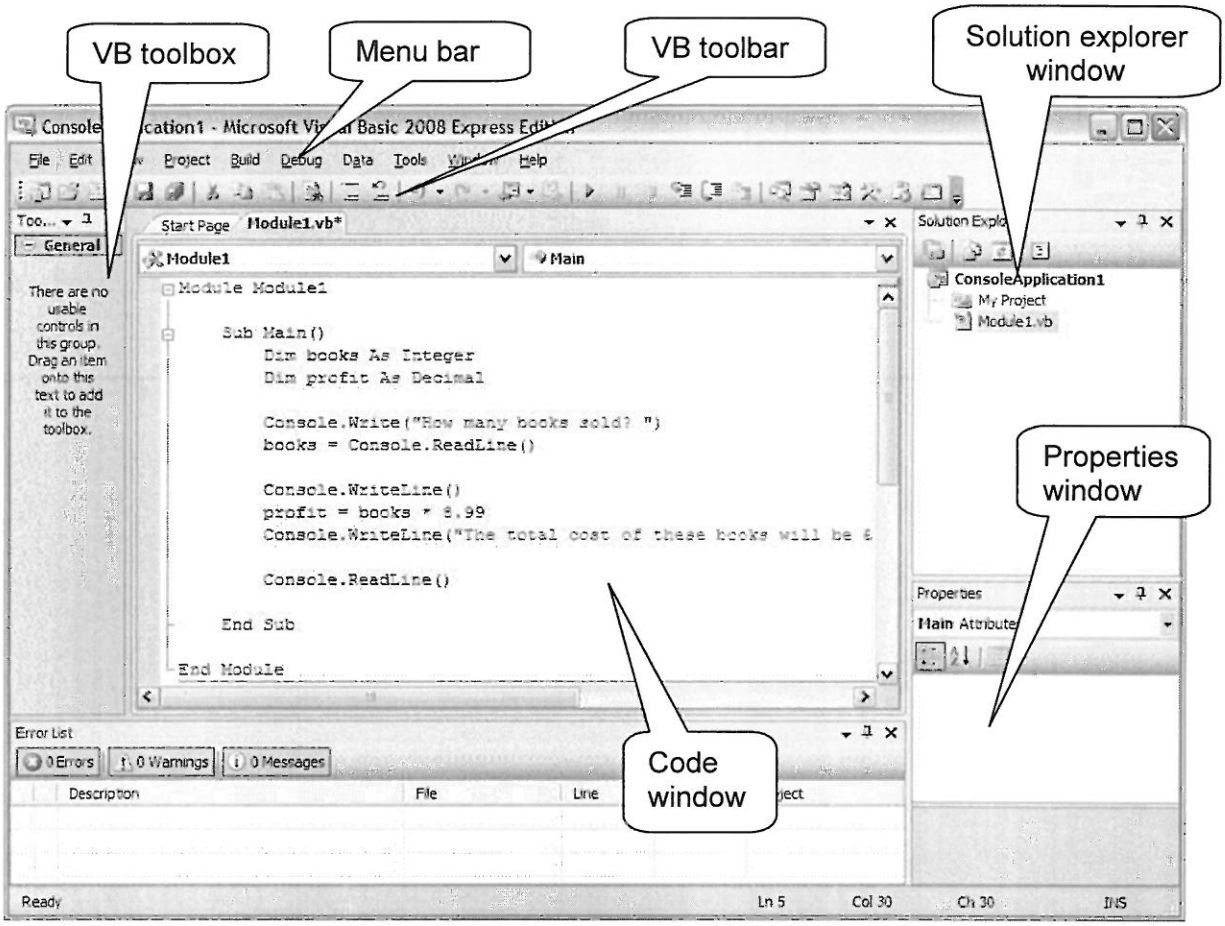
Here is what a simple Visual BASIC console program might look like:



When executed, the program appears on screen in a **console window**, like the one above.

To develop a program in Visual BASIC, you need to work in the VB development environment.

Here is the same program while it was being developed, showing some of the main features of the VB development environment:



At the top is the **menu bar**, with the usual menus (File, Edit ...) and some specialised VB menus.

Below this is the VB **toolbar**. The item you will use most is the ▶ icon, which you use to run (execute) a program.

Down the left side is the **toolbox**; you won't need this for console programming. The program instructions (code) appear in the **code window** in the middle of the screen.

On the right side of the screen you will see the **solution explorer window** and the **properties window**. These will be explained as you work through the examples.

Important advice!
If any of the windows disappear unexpectedly, you can get them back by selecting them in the **View** menu.

1.4 Input and Output – example

Almost every program that has ever been written follows a pattern called I-P-O. This stands for input – process – output. Most programs are designed to take in some data, to process it in some way, then to output some data.

Example 1

To keep things simple for your first VB program, we are going to develop one which misses out the middle step! It will simply take in some information, and give it out again. It's not very useful, but it will teach you some of the basics of VB programming.

We start with the **program specification**:

> *Design, implement and test a program which will prompt the user to enter his or her name. The program should then display the name and a welcome message.*

Stage 1 – Analysis

Start by thinking about what data goes in and what data comes out of the program while it is running. A **data flow diagram** is a good way of analysing this.

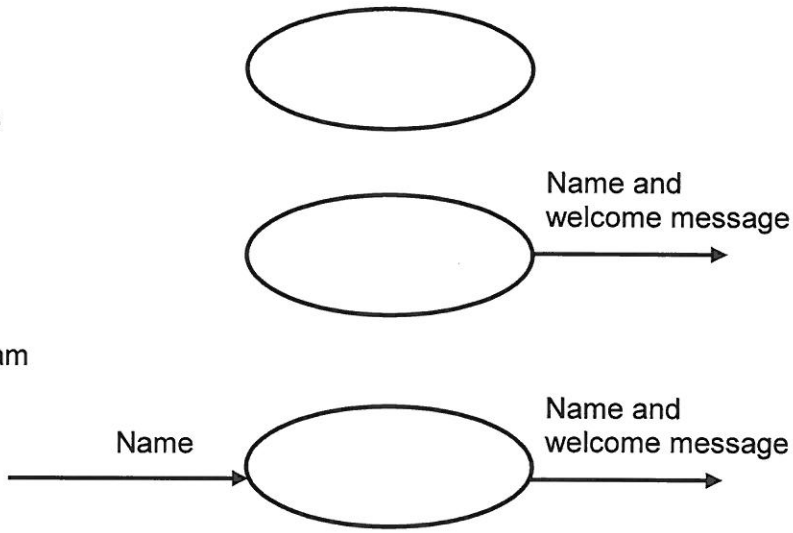
Start by representing the program as a 'blob'...

Think... what information comes out of the program?

Show this as an arrow coming out of the blob...

Now think – what information must the user put into the program to give this output?

Show this as an arrow going into the blob...



Copy this diagram *– you have drawn your first* ***data flow diagram****.*

Stage 2 – Design: Pseudocode

Pseudocode is just a fancy name for a list of steps that the program should carry out every time you run it. You write it in a sort of cross between English and computer language, known as pseudcode. It lets you think about the steps carefully without getting bogged down in the actual coding. Another advantage of pseudocode is that it can then be easily converted into almost any high level language you want – in our case, Visual BASIC.

Here is a list of steps for our program (pseudocode):

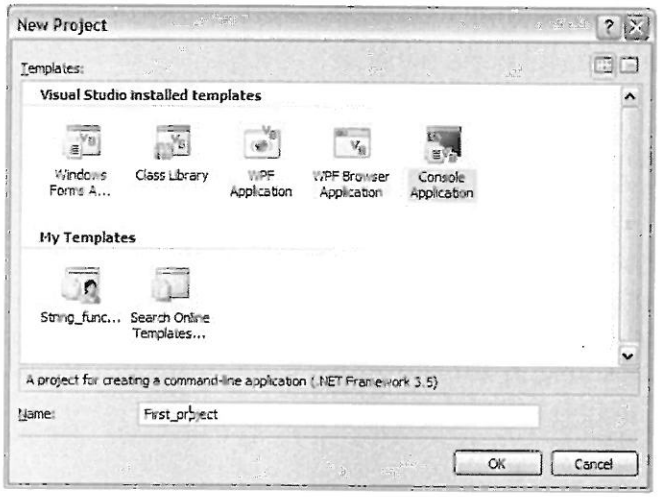
1. prompt the user to enter the name
2. store the name
3. display the message

That's the design complete!

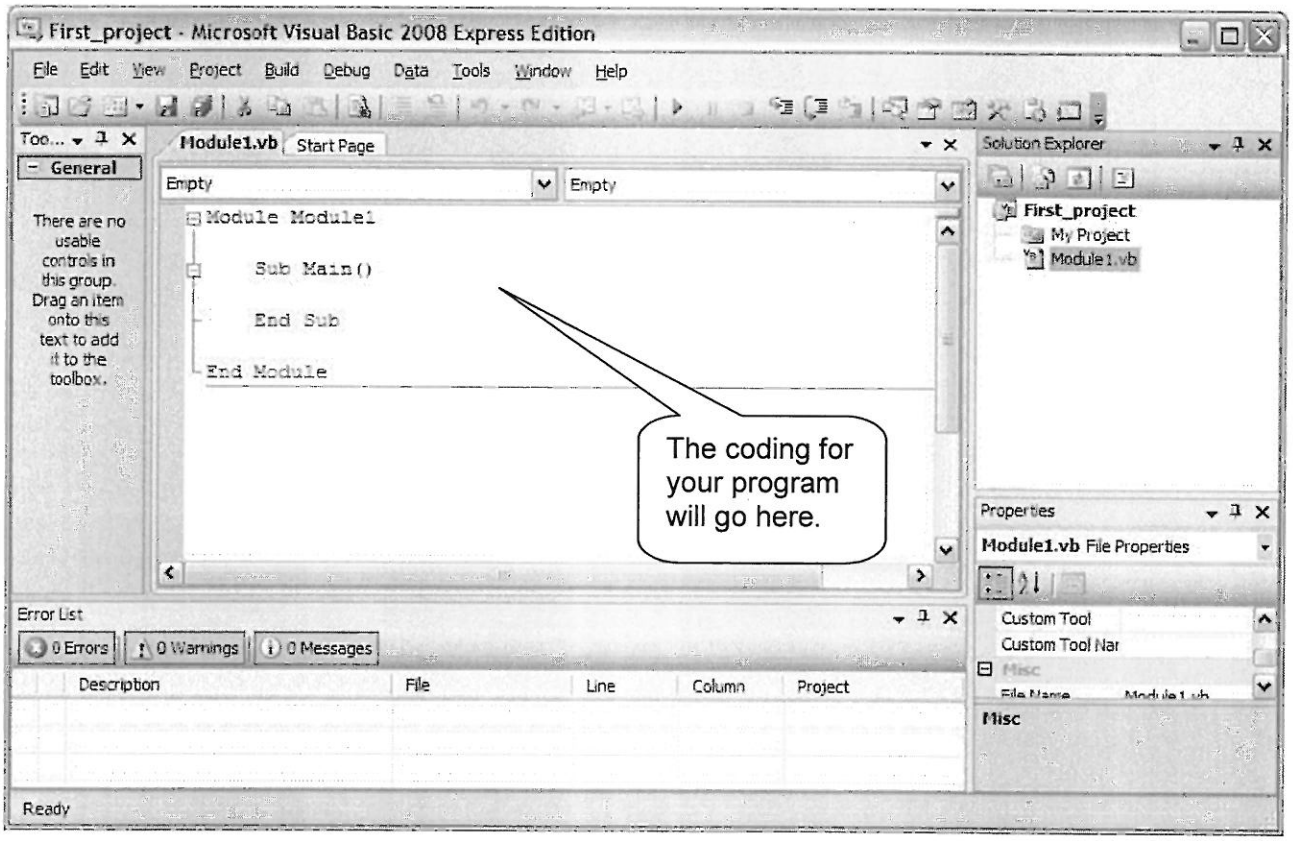
Stage 3 – Implementation

Now we are ready to start coding.

- Start up Visual BASIC on your computer
- Choose File / New Project
- Select Console Application
- Name the project 'First_Project'



Your screen should now display the VB development environment:



It's a good idea at this stage to **save your work**:

- Select **Save all** from the **File** menu
- Browse to where you want to save your projects
- Check the name is First_project, then click Save

Now enter the lines of code which are highlighted below in bold. Read the explanatory notes as you go along so that you understand what the code does.

The first 2 lines of code are as follows:

'code for input – output program
'by A. Programmer on 04/01/10

(These two lines start with a single apostrophe ('), which means that they are comment lines – they are ignored by the computer but make the program more readable – it is always good practice to start any new section of coding with some comment lines like these.)

The next line is:

Dim name As String

This line tells the computer to set aside a storage location in memory which will need to hold a string of characters, and to call it 'name' – we know this from the data flow diagram – remember, the program will input a name, then output the name, so it has to be able to store it in between.

Now we need to turn the pseudocode we wrote at the design stage into Visual BASIC.

Pseudocode		Visual BASIC
Prompt the user for the name	→	Console.Write("Enter your name")
Store the name	→	Name = Console.ReadLine()
Display the message	→	Console.Write("welcome to VB," & name)

These lines need a little explaining:

Console.Write("Enter your name")

(This line instructs the computer to display the text "Enter your name" in the console window.)

name = Console.ReadLine()

*(This line is best read from right to left – it tells the computer to take any **text** entered by the user into the console window, and store it in the memory location called **name**.)*

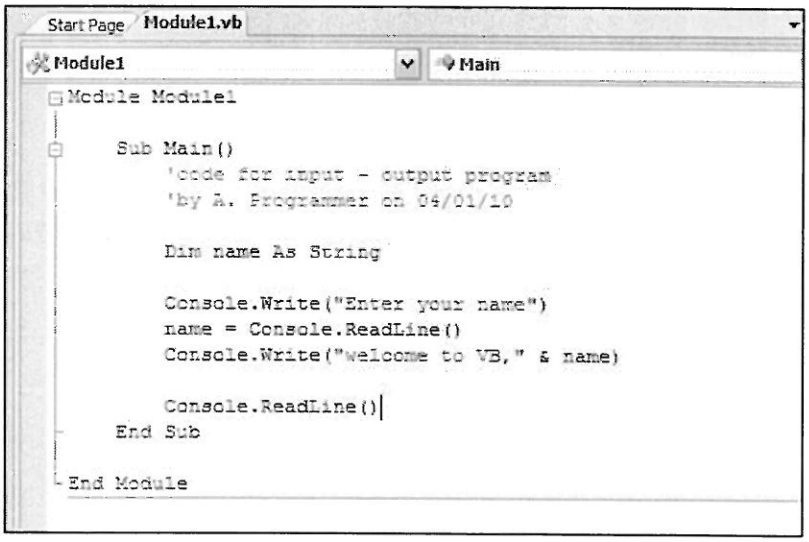
Console.Write("welcome to VB," & name)

*(The final line tells the computer to display the words **Welcome to VB**, followed by whatever data is stored in the location called **name**, on a new line in the console window.)*

The commas and speech marks must all be there in the correct order!

Add a final line
Console.ReadLine()
To prevent the message disappearing until the user presses the Enter key.

If you have entered all that correctly, it should look like this:



(Notice the extra blank lines – they don't do anything, but they make the program more **readable** – just like gaps between paragraphs in an essay or report.)

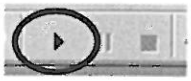
Notice also that VB uses colours to make the program more readable:

- Comment lines show in green
- Ordinary code in black
- VB keywords in blue
- Text in red

Now **save** the program again (just click **Save all**).

Stage 4 – Testing

The next stage is to make sure the program works correctly.

Click on the **start icon** on the menu bar ⟶ 

- A console window should appear, prompting you to enter your name
- Enter your name and press return
- An appropriate message should appear

If not, click on the **end icon**, go back to the coding and check it carefully. Correct any errors. Run it again.

If it does work, well done!

Would you buy a piece of software that had only been tested once? Probably not!

One test is not enough! We need to test the program **systematically**.

So... run some more tests – some 'normal' tests, like the one above – but also try some more 'extreme' testing. For example, what happens if you enter a number instead of a name, or if you press <Enter> before entering a name?

Documentation

At this stage, you might want some hard copy evidence of your program – it's your first VB program, so you may feel justifiably proud of it!

Printing your Program

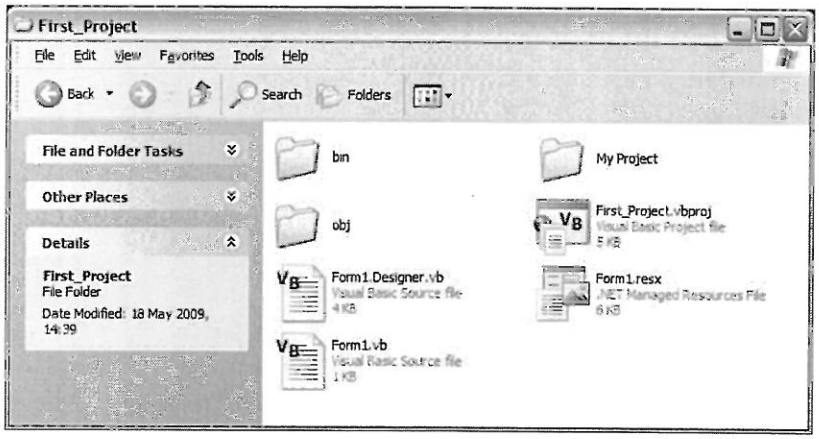
Select **Print** from the **File** menu to print your program coding.

Saving your Program

Choose **Save all** from the **File** menu to save any changes you have made.

Visual BASIC will have created several files, storing your project in a folder called 'First_Project'.

Do not delete or rename any of these files!



Stage 5 – Evaluation

The evaluation of your program should answer the following questions:

1. Is the program **effective**? (Does it do what is required by the specification?)
2. Is the program **usable**? (Is it clear what the user has to do?)
3. Is the program **maintainable**? (Could another programmer understand how it works?)

Your answers might look like this:

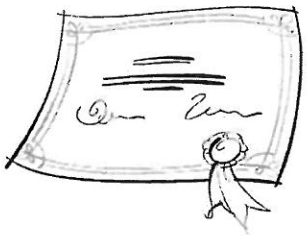
Evaluation

The program is effective. It fulfils the specification. If you enter a name, it responds with an appropriate message.

The user interface is easy to use – the program prompts for the required input.

The coding has comment lines and uses sensible variable names to make it maintainable by another programmer.

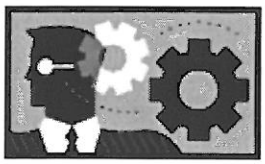
Topic 1 Summary



Well done! You have completed Topic 1. In this topic, you have learned how to:

- ☑ Create a simple VB console program
- ☑ Save and print a VB program
- ☑ Input data using Console.ReadLine()
- ☑ Display messages and data using Console.Write()
- ☑ Carry out testing
- ☑ Evaluate software in terms of effectiveness, usability and maintainability

Check all the items on this list. If you are not sure, look back through this section to remind yourself.



When you are sure you understand all of these items you are ready to continue with the next topic.

