Module 3.3 – Programming I

All programming languages perform the same function: they allow you to give **instructions** to the computer. When you run a program the computer **executes** these instructions.

Like any foreign language you must learn vocabulary - the commands used by the language - and grammar - the way to structure the instructions.

In this module you will start to learn the techniques involved in solving problems by writing computer programs. You will learn to program using Visual Basic .Net, the latest version of Microsoft's popular programming language.

Task 1 - Translator Programs

Computers have tremendous calculating power and can store huge amounts of information. But without instructions the processor will sit idle, unable to do anything. Computer programmers write these instructions to allow the computer to do useful work.

The computer doesn't understand the instructions directly - they only understand binary **machine code** instructions.

It is possible to write programs in machine code but it is difficult and time consuming to code and debug. Instead programmers prefer to use **high level languages** that use English-like commands.

There are many high level programming languages but they all have one thing in common, namely, the need to translate the HLL to machine code.

Compilers

A compiler is one example of a translator program. Compilers take the high level language code that you enter, **source code**, and translate it to binary **object code** in a single operation.

This object code is saved so, when you run the program again, you do not have to translate it again. The disadvantage of a compiler is that any errors are not identified until the source code is compiled.

Interpreters

An interpreter is another example of a translator program. Interpreters take the source code one instruction at a time then translate and execute each instruction individually. Each time the program is run it must be reinterpreted. The advantage of an interpreter is that errors are identified as they are entered.

Software Portability

A software program is said to be **portable** if it can easily be adapted and run on computer systems other than the one it was written on.

Answer the following questions in your jotter.

1. Would you use a compiler or interpreter when testing a program? Explain your answer.

2. Would you use a compiler or interpreter when preparing a program for final release? Explain your answer.

3. Why could a portable program be more commercially successful?

4. Give **three** examples of programs that you think are portable.

5. Name **five** high level languages?

6. For one of these languages give an example of the commands or instructions used in that language that indicates that it is a high level language.

Task 2 - Special Purpose Languages

There are literally hundreds of different programming languages all used by different people for lots of different tasks.

Some programs are very versatile and can be used to solve a wide range of problems. These are called **general purpose languages.**

Other languages are designed for specialist purposes. These are called **special purpose languages**.

Language	Use	Features
Fortran	Mathematical and Scientific	Built in functions and data types suitable for mathematicians, engineers and scientists
COBOL	Commercial Data Processing	Ability to handle huge numbers of files
Prolog and LISP	Artificial Intelligence	Ability to "learn" and "think" by analysing a knowledge base
Java	Writing Applets	Creates small programs to run on a wide variety of computer systems.
SQL	Database query language	Uses keywords to allow very specific searches in database files.
ADA	Real time applications	Used for applications that must be continually updated.

- One of the most common programs that are written when trying to learn a new language is the "Hello World" program. This program, when executed, displays a message on screen saying "Hello World".

- For at least five different programming languages complete the following table. One example has been completed for you.

Programming Language	Hello World Code
Pascal	**Program** Hello_World; **Begin** Writeln('Hello Word'); **End**.

Task 3 – Variables and Data Types

All programs handle **data.** Usually, data will be **input** from the keyboard to the program. Some calculations or **processing** will be carried out before results are **outputted** to the screen.

This data is stored in **VARIABLES.**

Variables are locations in memory that are given names and can store data. This allows us to store and retrieve data.

Variables must only contain data of a certain type. For example, if you create a certain variable for storing whole numbers then we can't store text in it.

Some variables take up more memory than others; you should try and reduce the memory required by specifying the smallest data types possible.

The three main categories of data types in Visual Basic are:

- Numeric (Whole numbers and real numbers)
- Character
- Miscellaneous

For whole numbers use one of the following:

Type	Bytes	Range
Byte	1	0 to 255
Integer	4	-2,147,483,648 to 2,147,483,647
Long	8	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
Short	2	-32,768 to 32,767

Any of the following types can be used for **floating point** numbers i.e. numbers with decimal fractions.

Type	Bytes	Range
Decimal	12	±79,228,162,514,264,337,593,543,950,335 to ±0.0000000000000000000000000001
Double	8	±1.79769313486231 x 10^{308} to ±4.94065645841247 x 10^{-324}
Single	4	±3.402823 x 10^{38}

For text you must use one of two data types:

Type	Bytes	Range
Char	2	Used for storing a single character
String	10 bytes + 2 bytes per character	Used for storing a series of characters

The other types you might use include:

Type	Bytes	Range
Boolean	2	Set to true or false
Object	4	Can store any type of data but this loose typing is bad programming practice

In Visual Basic you declare a variable in the following way:

```
        Dim input1 As Integer
```

Another example:

```
        Dim forename As String
```

For the following items of data identify the most suitable data type for storing this data.

1. 1
2. a
3. Xavier
4. 3.14
5. 1000000
6. 10000000000000
7. X3
8. nine
9. &
10. 3.1415932653589
11. True
12. 65000
13. 66000
14. -32000
15. -40000
16. False

Task 4 - Visual Basic .Net

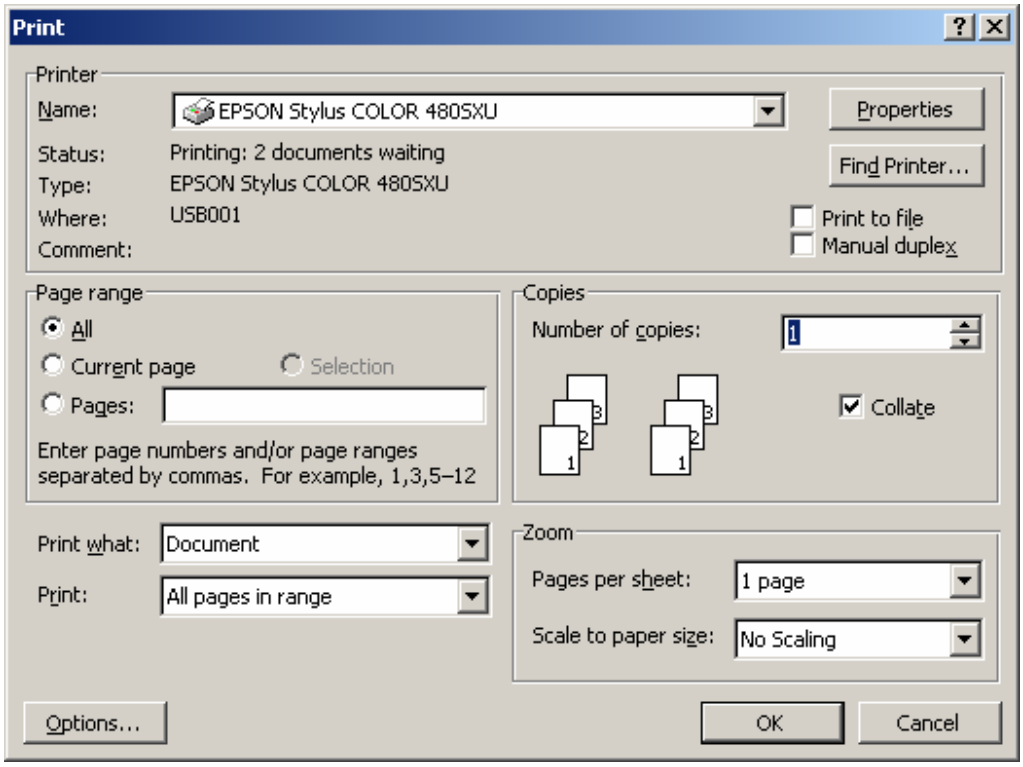
VB is a **programming language** - it allows you to issue instructions to the computer.

VB produces **Windows applications** - programs produced look like your normal Windows programs.

VB is **event driven** - things happen in the program when you click on or choose items on the screen.

VB is **object oriented** - the various parts of a window - buttons, menus, check boxes etc. - are the objects that make up an application.

Look at the following dialog box and answer the following questions.



1. What objects are in this dialog box? E.g. 3 check boxes.

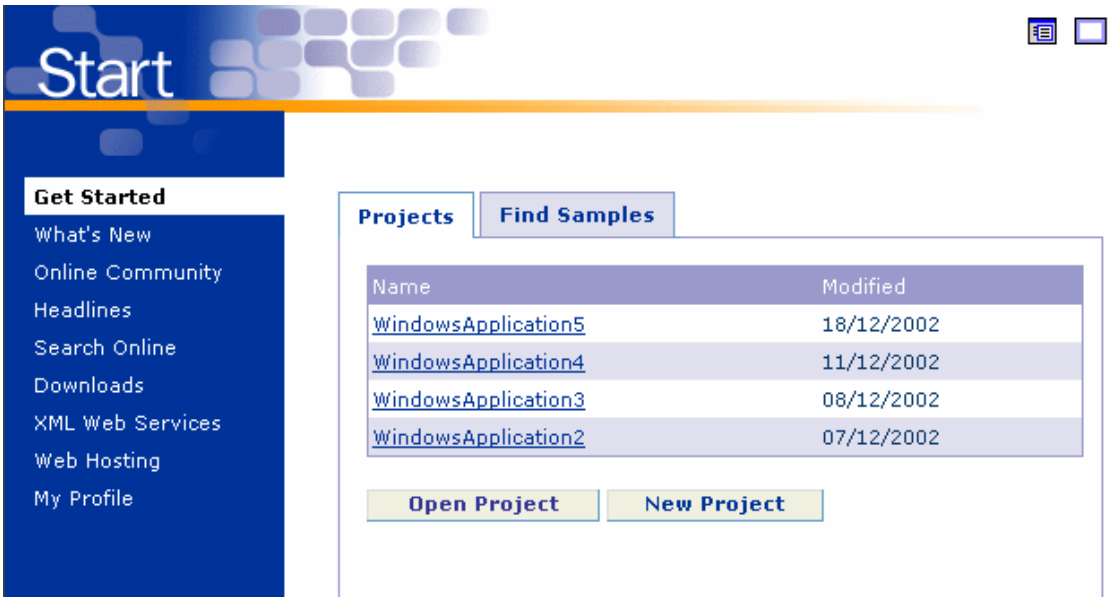
2. What events can occur with these objects? E.g. you can select none, one, two or three check boxes.

3. Which object(s) must contain instructions to close the dialog box?

4. Which object would apply the choices made in the dialog box?

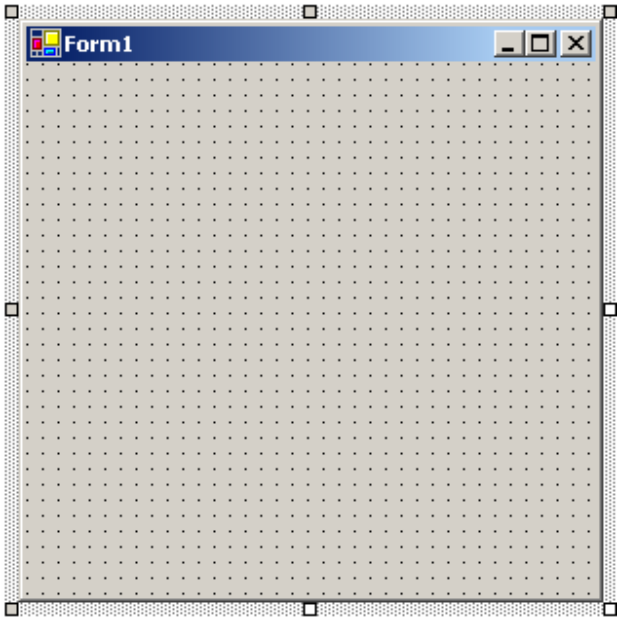
Task 5 - Creating A VB Application

When you launch Visual Studio you will see the following screen appear.



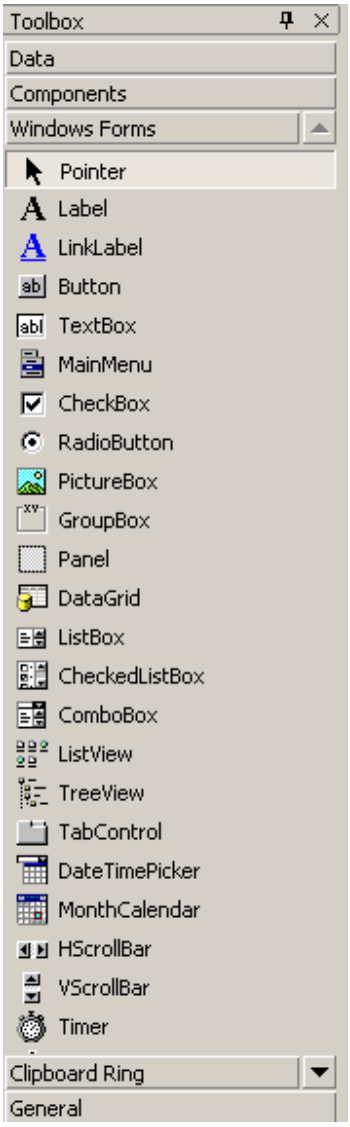
When you choose to create a new project you must ensure that you choose a Windows Application, name the task and that the project is created in your own network folder.

Then you will have a screen crowded with many different tool boxes. The most important parts of the window are the following:

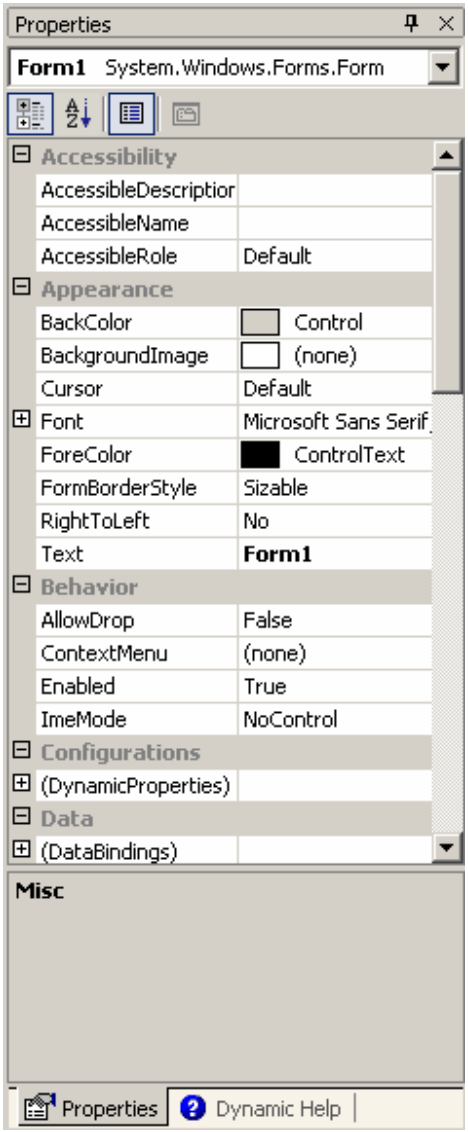


The form window - the form is the window upon which you will build your application. From this window you can also view the VB code.

The Components Toolbox - from here you will select the various objects to place on the form in order to create your application.



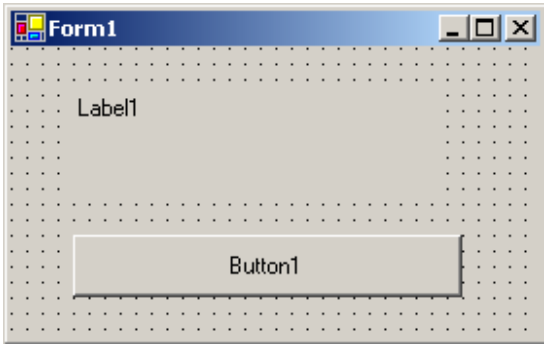
The Properties Toolbox - here you can change the many attributes of all the objects in your application.



In each of the following tasks you will add objects to the form, change their properties and add code to the objects.

Do the following:

- From the Start Menu choose Visual Studio .Net
- Choose to create a New Application called Task 5, ensuring it is created in your own folder.
- Use 1 label and 1 button to create the form shown below.



- Use the property toolbox to change the name of label1 to 'output'
- Save this application.

Task 6 - Outputting Text

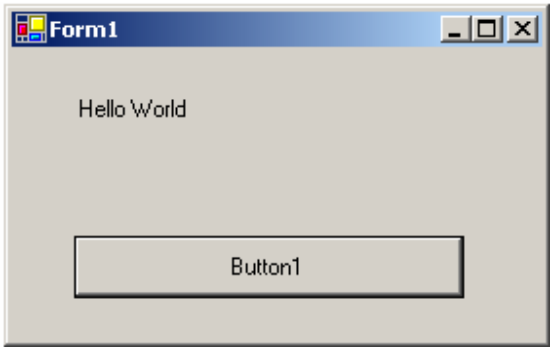
In the previous task you created a form with two objects on it. However this form had no purpose - it didn't do anything because you had not instructed it to do anything. To actually create a useful form you must write instructions attached to the form or the button or any object in your application.

When you begin to enter Visual Basic code it is important to use the correct syntax. That means using the correct commands in the correct way.

- Open your Task 5 application.
- You want something to happen when the button is pressed so double click on the button and enter the following code:

```
        output.text="Hello World"
```

- Choose Build -> Build Solution
- To run your application you must minimise Visual Studio, open My Computer, open your folder then the folder containing your application. In this folder, open the Bin folder and choose the application icon.
- When you run this application, pressing the button causes the label caption to change.



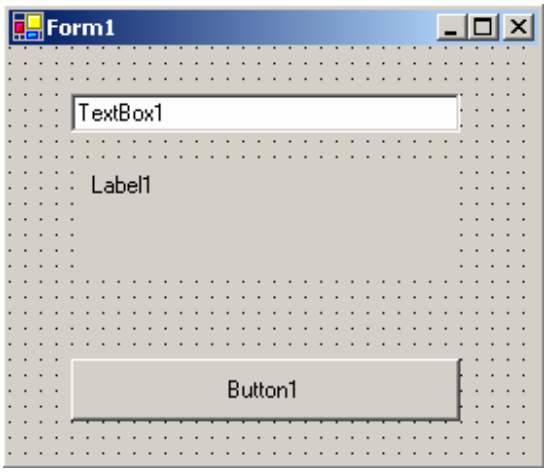
- Change your button's code to display a different message.

Task 7 - Inputting & Outputting Values

Your previous program only outputs text; you have no way of inputting values to the program. In this task you will use a text box to allow you to enter values.

The caption in a text box or label can be altered by using VB instructions to alter the text property of the object.

- Create a new application named Task 7.
- Create a form to look like the following:



- Change the names of the objects to:

  Label1 = output and Textbox1 = forename

- Double click on Button1 and enter the following code.

```
Dim Name As String  'Declares a variable Name of type string
Name = Forename.text 'transfers contents of text box to the variable
output.text = Name  'Writes the variable to the label
```

- Build and run this application. Enter a name in the text box and press the button. The contents of the text box will be displayed on the label.
- Alter the code as follows

```
Dim Name As String
Name = Forename.text
output.text = "Hello" & Name 'Writes text and variables to the label
```

- Build and run this application.

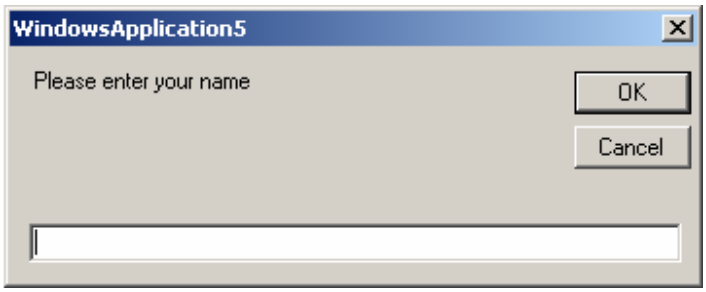
Task 8 – Input Boxes

An alternative way to input values to a program is to use an input box. These allow you to enter a value that is then stored in a variable.

- Open your task 7 application.
- Double click on the button to edit your code.
- Enter the following code.

```
Dim Name As String
Name = InputBox("Please enter your name")
output.text = "Hello" & Name 'Writes text and variables to the label
```

- Now when you build and run this application the following box will appear, prompting you to enter a value.

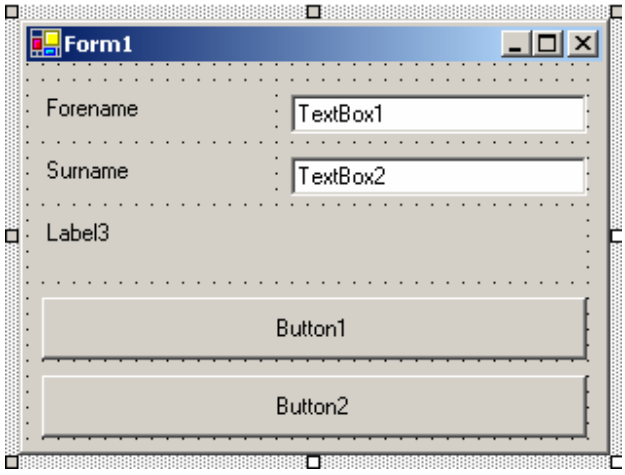


- Once you have entered your name and pressed OK the program will say hello in the same way as the previous task.

Task 9 – Text Boxes and Input Boxes

You will now practice working with these objects and extending the programs that you have already written.

- Create a new application named Task 9.
- Create a form to look like the following.

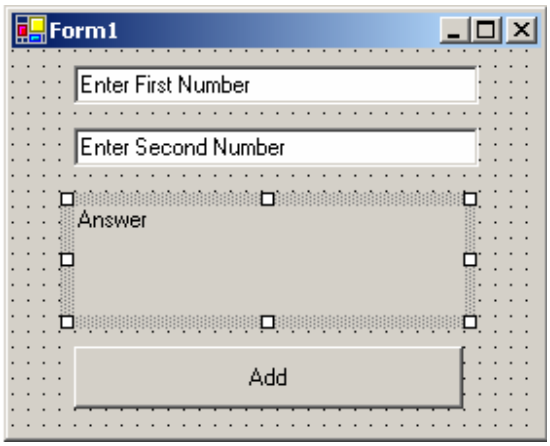


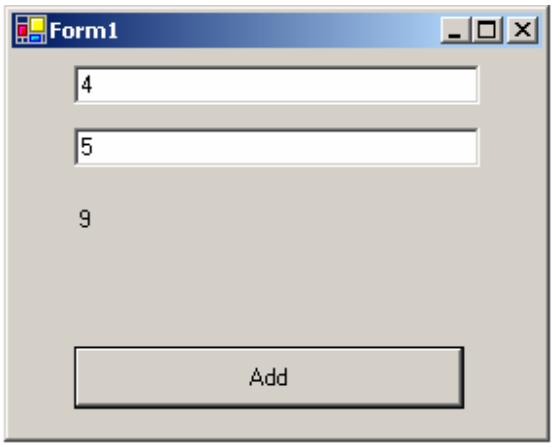
- Button 1 should be coded to take the forename and surname from the two text boxes and say 'Hello *forename surname*' in Label3.
- Button 2 should be coded to prompt for the forename and surname to be entered in two input boxes and say 'Hello *forename surname*' in Label3.

Task 10 – Calculations

So far all your programs have dealt with **strings** i.e. a string of characters. In the following tasks we will start to deal with whole numbers or **integers** and perform calculations on those values.

- Create a new application called Task 10.
- Create the following form.

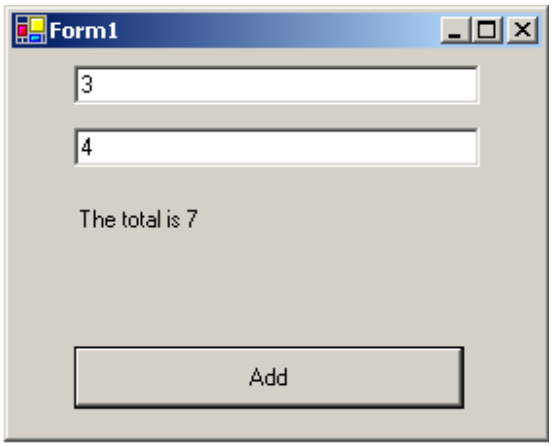




- Name the text boxes *number1* and *number2* and name the label *answer*.
- Enter the following code.

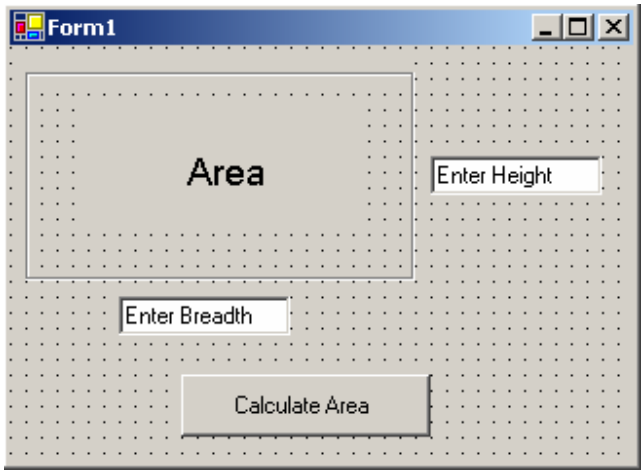
```
Dim input1 As Integer, input2 As Integer, total As Integer
input1 = number1.text
input2 = number2.text
total = input1 + input2
answer.text = total
```

Change your program so that it outputs the following.

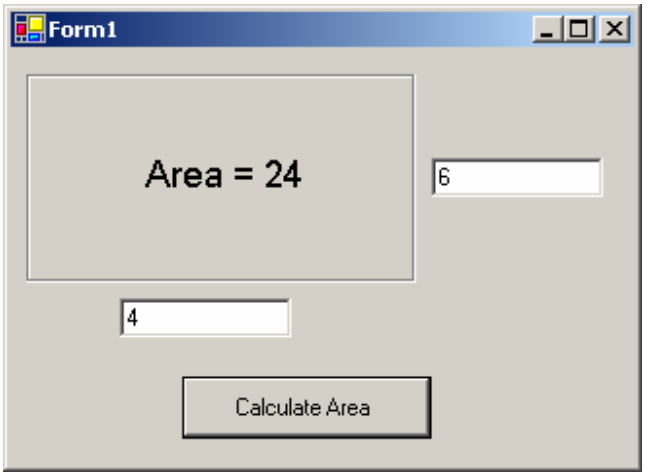


Task 11 – More Calculations

- Create a new application named Task 11.
- Create the following form.
- Code the following buttons to calculate the area of the rectangle.



- Build and run this application.

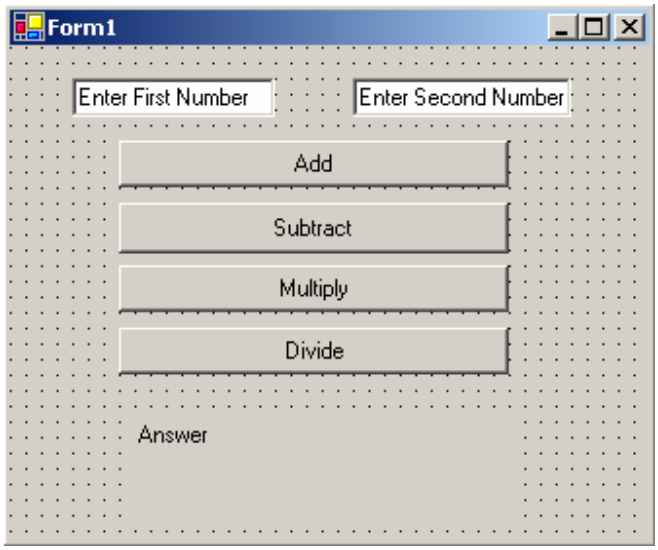


Task 12 – Further Calculations

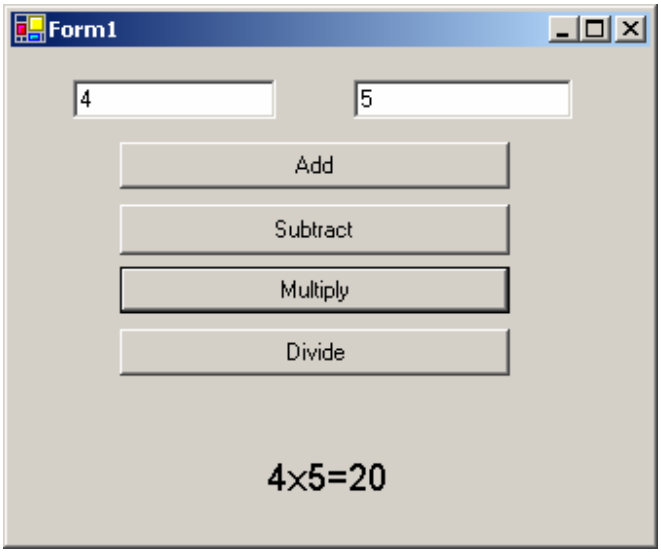
In Visual Basic there are a series of arithmetic operations that can be carried out.

+ Add	- Subtract	* Multiply	/ Floating Point Division
\ Integer Division	^ Exponent	mod Remainder	
+= Add 1	-= Subtract 1		

- Create a new application called Task 12
- Create a form to look like the following.



- Code each button to produce output like the following.



- Build and run this application.

Task 13 - The Software Development Process

People write programs to solve problems. In this module you will learn a formal approach to developing a software solution to a program. This approach can be applied to any programming language.

The solution to a problem can be found by following seven steps:

- Analysis
- Design
- Implementation
- Testing
- Documentation
- Evaluation
- Maintenance

Answer the following questions in your jotter.

1. Why is it not possible to simply write a program without all the associated steps?

2. The software development process is followed all the time in the commercial software industry. Why is it important that you use the same process for producing your small programs?

3. A mnemonic is an aid to memory, e.g. Richard Of York Gave Battle In Vain to remember the initials of the colours of the rainbow. Try and make a mnemonic for the steps of the software development process.

Task 14 – Analysis

At this stage you must look at the problem to be solved and state, very clearly, what the program will do.

You must produce a **requirements specification**, a document stating very clearly what the problem is and what the program must do to solve it.

Then you should state the inputs to your program, the calculations you will perform on those values and the outputs you expect to receive.

Example

Problem – Write a program that, given the radius of a circle as input, will calculate and output the circumference of that circle.

Inputs – Radius of circle - Integer

Process – Circumference = 2 x pi x radius

Outputs – Circumference – Single

In your jotter, analyse the following problem.

A program is required that, given the radius of a circle as input, will calculate and output the diameter, circumference and area of that circle.

Task 15 – Design

Before you begin to write your program you must plan your solution. There are two aspects of your program to design:

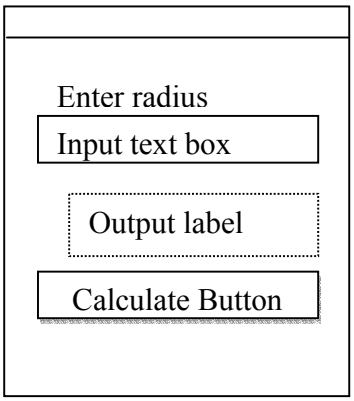
- Design of the user interface
- Design of software logic

When you design your user interface you must plan what objects you intend to place on your form. Will you use text boxes or input boxes? Will you use buttons or menus? What other objects will you use on the form? You should sketch out the various screens that you will draw.

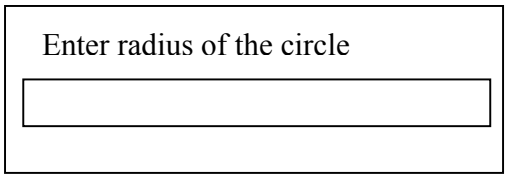
Example

Problem – Write a program that, given the radius of a circle as input, will calculate and output the circumference of that circle.

Form



Input Box



A program is required that, given the radius of a circle as input, will calculate and output the diameter, circumference and area of that circle.

- In your jotter, draw the form that you will create to create the program described above. Label each object. If you plan to use any input boxes, draw these also.

Design of Software Logic

The second stage of the design stage is designing a solution. We can do this in two ways.

You can write an **algorithm** i.e. break the problem down into a series of steps that can be followed to solve the problem, writing the steps in pseudocode, part English and part programming language.

Example

Problem – Write a program that, given the radius of a circle as input, will calculate and output the circumference of that circle.

1. Get radius
2. Calculate circumference
3. Output circumference

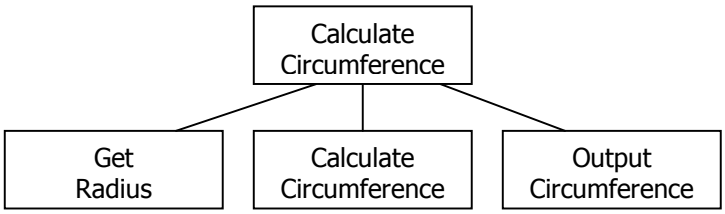
These steps can be refined:

1.1 radius = InputBox("Please Enter Radius")
2.1 circumference = 2 * 3.14 * radius
3.1 output.text = radius

Alternatively you can break the solution down into smaller steps and represent this in a **structure diagram**.

Example

Problem – Write a program that, given the radius of a circle as input, will calculate and output the circumference of that circle.



A program is required that, given the radius of a circle as input, will calculate and output the diameter, circumference and area of that circle.

- In your jotter, design a solution using an algorithm and structure diagram.

Task 16 – Implementation

At the implementation stage you will actually use a **programming language environment** to produce a working program that solves the problem.

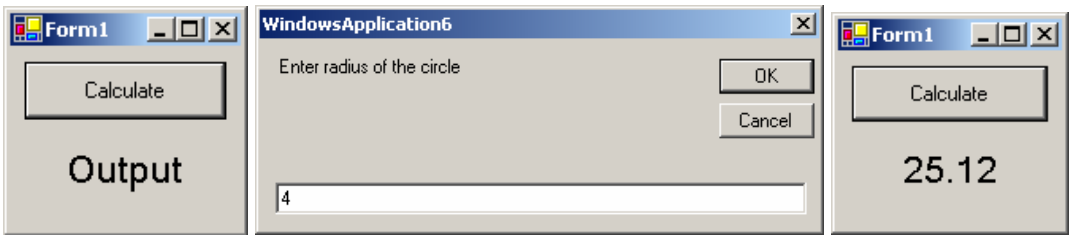
The code that you write should be **readable.** This will help when debugging the program, i.e. finding and fixing errors in the program.

You can make your code readable by including internal commentary. These are comments that describe what different sections of code are for.

Another way to enhance the readability of a program is to make sure that all variables are given meaningful names, i.e. give them names that relate to the data they store. The inputs and outputs from your analysis will become your variables.

Example

Problem – Write a program that, given the radius of a circle as input, will calculate and output the circumference of that circle.




```
Dim radius As Short 'Input Variable
Dim circumference As Single 'Output variable
radius = InputBox("Enter radius of the circle") 'Input radius
circumference = radius * 3.14 * 2 'Calculate circumference
output.Text = circumference 'Output circumference to label
```

A program is required that, given the radius of a circle as input, will calculate and output the diameter, circumference and area of that circle.

- On your computer, using your analysis and design, implement a solution to the program above. Include internal commentary as shown above.

Task 17 – Testing

At the testing stage you must check that your program is **fit for purpose** i.e. that it does what is was meant to do as instructed in the requirements specification, and that the program is reliable i.e. it produces correct output valups for normal inputs.

To test the reliability of a program you can input a series of test cases.

Normal test data is the everyday data that should always give correct output.

Extreme test data is data close to - but not outwith - the limits of the program. Your program should still be expected to produce accurate results.

Exceptional test data is data beyond what would normally be used. You should not expect correct answers. A program that can take exceptional data without crashing is said to be **robust**.

There are several types of error that can be identified at the testing stage.

Syntax errors are errors in the programming language e.g. integar instead of integer.

Logic errors are errors in the design of the software e.g. trying to add two numbers before you have inputted them.

Example

Problem – Write a program that, given the radius of a circle as input, will calculate and output the circumference of that circle.

	Expected	
	Radius	**Circumference**
Normal +ve	4	25.12
Normal –ve	-4	-25.12
Zero	0	0
Extreme +ve	5000	31400
Extreme -ve	-5000	-31400
Exceptional +ve	6000	37680
Exceptional -ve	+6000	-37680

Actual	
Radius	**Circumference**
4	25.12
-4	-25.12
0	0
5000	31400
-5000	-31400
6000	37680
+6000	-37680

A program is required that, given the radius of a circle as input, will calculate and output the diameter, circumference and area of that circle.

- Test your program fully, recording the expected and actual results.

Task 18 – Documentation

Throughout the software development process there are various forms of documentation.

At the analysis stage you produce a requirements specification. At the design stage you produce an algorithm or structure diagram showing the logic of the program along with sketches of the user interface.

At the implementation stage you can add **internal commentary**, small comments that describe how individual procedures or particularly complex sections of code work.

Testing produces test reports and error reports. A structured listing is a formatted printout of the source code. A dry run is way of

Of great importance at the Documentation stage is the production of a **user guide** and a **technical guide**.

A user guide usually contains a quick start guide, a description of the main functions of the program including its menus, toolbars etc.

A technical guide will detail the minimum and preferred hardware and software specifications for installing and running the software. It will include installation information and, perhaps, a troubleshooting guide.

Example

Problem – Write a program that, given the radius of a circle as input, will calculate and output the circumference of that circle.

User Guide
This program calculates the circumference of a circle when you enter the radius. Launch the program and press the calculate button. In the box that appears, enter the radius and press OK. The circumference will be calculated and displayed automatically.
Technical Guide
The program will run on all Windows operating systems. It has minimal memory, processor and hard disk requirements. It is guaranteed to correctly calculate circumferences for values of radius up to 5000. Entering values in excess of this can cause the program to crash.

A program is required that, given the radius of a circle as input, will calculate and output the diameter, circumference and area of that circle.

- In your jotter, write a brief user guide and technical guide.

Task 19 – Evaluation

You must evaluate your finished program by answering a series of questions.

- Does your program solve the problem described in the requirements specification?
- Is your program reliable?
- Are there any limitations to your program?
- Could your program be improved in any way?
- Could your program be adapted to other uses?
- Is your program robust?

Example

Problem – Write a program that, given the radius of a circle as input, will calculate and output the circumference of that circle.

1. The program prompts the user for the radius of a circle and accurately calculates and outputs the circumference of that circle. The value used for pi is 3.14 and this is less accurate than 3.1415926 etc.
2. The program is reliable for all normal and extreme inputs.
3. The program is limited by the range of the data types Short and Single. This is not a major problem unless you wish to calculate extremely large values.
4. The program could be improved by outputting in a more user friendly way e.g. display 'The Circumference is 15' instead of just '15'.
5. The program could be adapted to work with other shapes and calculations, e.g. area of circles, perimeters of rectangles, volume of spheres etc.
6. The program is not very robust since very large values can cause the program to freeze or crash.

A program is required that, given the radius of a circle as input, will calculate and output the diameter, circumference and area of that circle.

- In your jotter, write an evaluation of your program.

Task 20 – Maintenance

The development of a software program does not end immediately after the program is released. **Maintenance** is the release of bug fixes, patches and updates.

There are three types of maintenance:

Corrective maintenance is the correction of bugs missed during the testing process that surface during prolonged use of the program.

Adaptive maintenance are the changes necessitated by hardware or software changes in the operating environment. For example, programs may be updated as a result of a new operating system or new hardware devices.

Perfective maintenance is adding new features or enhancing existing tools in order to create a more desirable program.

Example

Problem – Write a program that, given the radius of a circle as input, will calculate and output the circumference of that circle.

For such a small program there is little need for either type of maintenance. Some of the suggestions you made in your evaluation could be implemented. Any bugs found during testing could be fixed. Any enhancements to the user interface could be implemented. Any changes to the data types could be made.

A program is required that, given the radius of a circle as input, will calculate and output the diameter, circumference and area of that circle.

- In your program carry out any maintenance activities suggested in your evaluation.
- In your jotter document the maintenance activities you have undertaken.

Task 21 – Programming Assessable Task

In this task you will use the steps of the software development process to produce a program to solve a problem.

Problem
A program is required that decides upon class sizes for the new first year pupils in St Aloysius' College. Practical sets should have a maximum of 20 pupils, other classes may have up to 25 pupils. The program must prompt for the total number of pupils in the new year group and then calculate and display how many practical and non-practical sets are required.

Produce and submit the following documentation:

Analysis
A requirements specification stating clearly what the program must do and a list of all inputs and outputs with their data types.
Design
A sketch of the forms, dialog boxes, input boxes, message boxes etc. that your program will display. An algorithm or structure diagram showing the logic of the program.
Implementation
A printout of your VB code and forms.
Testing
An expected and actual test table with normal, extreme and exceptional test data.
Documentation
A brief user guide and technical guide.
Maintenance
A maintenance document detailing any corrective, perfective or adaptive maintenance you have undertaken.
Evaluation
A response to the evaluation questions to evaluate each stage of the software development process.

Programming I – Support Tasks

These tasks should be used as further practice in the fundamental aspects of programming. Implement these programs in Visual Basic.

1. A program is required that will calculate the area of a triangle if you enter its base and height.

2. A program is required that has two buttons – Show and Hide. Clicking the Show button should display a label with the message "Hello". Clicking the Hide button should remove this message.

3. Write a program that asks for the ages of two children and outputs their average age.

Programming I – Extension Tasks

These tasks should be used as further practice in more complicated tasks that can be used to advance your skills in implementing programs to solve a variety of problems. In each problem take the opportunity to use a variety of objects from the toolbox.

1. Write a program so that the words typed in a text box will change colour when a button or menu choice is made. E.g. change to green text when the Green button is pressed or Green menu option is selected.

2. Workers must work a basic week of 40 hours. They are paid £10 per hour and £15 for any overtime hours worked. Create a program to calculate the total salary for a worker when their total hours worked are entered.

3. Write a program that allows you to enter a number of seconds, press a button and this value will be displayed in minutes and seconds. Alternatively you could enter two values, for minutes and seconds, pressing a button converts this value to seconds. For example 80 seconds <-> 1 minute 20 seconds.

Homework Exercise 1

1.	Why must high level programming languages be translated?	(1)
2.	What is another characteristic of high level languages?	(1)
3.	Why are computers sometimes called *very fast idiots*?	(1)
4.	Why would interpreters be useful when debugging a new program?	(1)
5.	What is the main advantage of a compiler?	(1)
6.	What is the computer's fetch-execute cycle?	(1)
7.	How can software portability improve the commercial success of a software program?	(1)
8.	Why are there so many different programming languages?	(1)
9.	Which type of program should be written in machine code?	(1)
10.	Why is it necessary to have special purpose languages?	(1)
	Total	(10)

Homework Exercise 2

1. What is a variable? (1)

2. What data types are available to hold whole numbers? (1)

3. Which data type would hold a single letter of the alphabet? (1)

4. Which data type would hold a series of letters? (1)

5. Which data type would hold decimal fractions? (1)

6. Which data type can be set to two values, true or false? (1)

7. If the *object* data type can store all of the above, why should we not use the object data type every time? (1)

8. Name four objects in Visual Basic. (2)

9. Why is Visual Basic called an **event driven** programming language? (1)

Total (10)

Homework Exercise 3

1. Describe the effect of the following code samples:

   (a) label1.text = "Xavier"

   (b) label1.text = number1

   (c) label1.text = "The answer is" & total

   (d) number1 = label1.text

   (e) number1 = InputBox("Enter the length")

   (f) number1 = input1 + input2

   (g) label1.text = "The answer is" & length * breadth

   (h) number1 = number1 +1

   (i) number1 = number1 * 2

   (j) input = label1.text
   label1.text = input & "new text"

Total (10)

Homework Exercise 4

1. The following variables contain the values below:

   input1 = 5 input2 = 6 input3 = 10

   What would be the value stored in output after the following instructions have been executed:

   (a) output1 = input2 (1)

   (b) output = input1 * input2 (1)

   (c) output = input3 / 2 (1)

   (d) output = input 3 \ input1 (1)

   (e) output = input 3 mod input2 (1)

   (f) input1 input 3 / input2 (1)

   (g) input3 = input3 * input3
   output = input3 + input1 (2)

   (h) input1 = input1 + 1
   input1 = input1 + 1
   output = input1 +1 (2)

   Total (10)

Homework Exercise 5

1.	Why is it important that the requirements specification is accurate?	(1)
2.	At the analysis stage you define your inputs and outputs. What do these values become at the implementation stage?	(1)
3.	Why is it important, particularly when using Visual Basic, to design the User Interface?	(1)
4.	Describe the difference between pseudocode and structure diagrams.	(2)
5.	Describe the two tables you complete during the testing stage.	(2)
6.	Why are errors in programs described as 'bugs'?	(1)
7.	Why are there so many different programming languages available?	(1)
8.	Why is it beneficial to follow the software development process when designing a software solution to a problem?	(1)
	Total	(10)

Homework Exercise 6

1. Microsoft is the biggest and richest software company in the world. Why does it release programs that contain bugs? (1)

2. When Microsoft released Windows XP many hardware and software companies were forced to release updates for their programs. Which type of maintenance is this? (1)

3. Championship Manager was released several years ago. There have been several new versions since then. Which type of maintenance is this? (1)

4. Windows XP searches for updates when it is connected to the Internet. Which type of maintenance is this? (1)

5. A scanner requires new driver software to run with Windows XP. Which type of maintenance is this? (1)

6. Which type of maintenance will cost software companies the most money? (1)

7. Which type of maintenance will make software companies the most money? (1)

8. Why have user guides increased in size in recent years? (1)

9. Why do many companies supply user guides on CD instead of on paper? (1)

10. Do software users benefit from internal commentary? (1)

Total (10)