

AQA

A LEVEL COMPUTER SCIENCE

PAPER 1 & PAPER 2

PROGRAMMING HANDBOOK

VB.NET

NAME:

CLASS:

Published by:
LMC Education Ltd
Queens House
Paragon Street
Hull HU1 3NQ

© Copyright 2015 by LMC Education Ltd

All rights reserved. No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopy, recording, or any information storage and retrieval system, without permission in writing from the publisher or under licence from the Copyright Licensing Agency Limited. Details of such licences may be obtained from the Copyright Licensing Agency Limited, Saffron House, 6-10 Kirby Street, London EC1N 8TS.

No patent liability is assumed with respect to the use of the information contained herein. While every precaution has been taken in the preparation of this document, the publisher assumes no responsibility for damages or legal claims resulting from the use of the information herein. The content of this document is believed to be correct at the time of publication.

All company names and personal names used in this document are fictitious and do not represent any existing company or person. Any resemblance, to an actual company name or personal name, is coincidental and unintentional.

CONTENTS

INTRODUCTION	5
SUPPORTING MATERIALS	8
HOW TO USE THIS HANDBOOK.....	9
Handbook sections.....	10
Self-assessment.....	12
BACKGROUND KNOWLEDGE.....	13
History and features of VB.Net.....	14
Hello, World! In VB.Net	15
Using the Visual Studio IDE	15
Exercise 1	16
Principles of designing programs and writing algorithms.....	18
Exercise 2	19
Assemblers, compilers and interpreters	21
Exercise 3	22
Exercise 4	23
Debugging.....	24
Exercise 5	24
PAPER 1 PROGRAMMING THEORY.....	29
Section summary.....	30
4.1.1.1: Data types	30
Exercise 6	31
4.1.1.2: Programming concepts.....	32
Exercise 7	33
4.1.1.3–5: Programming operations	35
Exercise 8	35
4.1.1.6: Constants and variables	37
Exercise 9	37
4.1.1.7: String handling	38
Exercise 10	39
4.1.1.8: Random numbers.....	41
Exercise 11	42
4.1.1.9: Exception handling	43
Exercise 12.....	43
4.1.1.10-14: Subroutines	44
Exercise 13.....	45
4.1.1.16: Recursion	46
Exercise 14	46
4.1.2.2: Procedural-oriented programming.....	49
Exercise 15.....	49
4.1.2.3: Object-oriented programming.....	52
4.2.1.2: Arrays.....	55
Exercise 17	56
4.2.1.3: Fields, records and files.....	58
Exercise 18.....	59
4.2.1.4: Abstract data types.....	60
Exercise 19.....	61

4.2.2.1: Queues.....	61
Exercise 20.....	62
4.2.3.1: Stacks.....	65
Exercise 21.....	65
4.2.4.1: Graphs.....	67
Exercise 22.....	67
4.2.5.1: Trees.....	70
4.2.6.1: Hash tables.....	71
Exercise 24.....	71
4.2.7.1: Dictionaries.....	73
Exercise 25.....	73
4.3.4: Searching algorithms.....	74
Exercise 26.....	74
4.3.5: Sorting algorithms.....	75
Exercise 27.....	75
4.3.6: Optimisation algorithms.....	77
Exercise 28.....	77
PAPER 2 PROGRAMMING EXERCISES	80
Section summary.....	81
4.4.2.3: Regular expressions.....	81
Exercise 29.....	82
4.5.1: Number systems.....	83
Exercise 30.....	83
4.5.4: Binary.....	87
Exercise 31.....	87
Exercise 32.....	89
Exercise 33.....	91
4.5.5: Information coding systems.....	93
Exercise 34.....	93
4.5.6: Representing images, sound and other data.....	98
Exercise 35.....	98
4.9.1: Communication.....	101
Exercise 36.....	101
EXTENSION WORK.....	108
Section summary.....	109
Task 1: Mobile Phone Expenses.....	110
Task 2: Buy-To-Let Register.....	116
Task 3: Weekend Planner.....	122
Task 4: Did I Sleep Well?.....	126
Task 5: Air Travel Planner.....	132



INTRODUCTION

Introduction

The ability to write rigorous and computationally correct programs is an essential skill that you must become proficient in to be able to complete the AQA Computer Science A level course. Writing in pseudo-code is examined in both Paper 1 and Paper 2 of the course and you will also be expected to write code in VB.Net as this is chosen by your exam centre.

This handbook assumes that you have access to Microsoft Visual Studio or an open source development environment that enables you to write programming code.

At A level, there is an expectation that you will be able to write programs and algorithms to solve complex problems. You should expect that the problems will involve using a mathematical model, e.g. the use of simultaneous equations or matrix algebra. You should also expect that the problems will involve simulation of complex real-world problems, e.g. queuing and optimisation situations.

Objectives of writing programs in VB.Net

The ability to write efficient algorithms is an important skill to obtain. Algorithms need to be efficient both time-wise and space-wise. In Computer Science, the following process is used to write code:

- Pseudo-code is used to write algorithms. Writing an algorithm enables you to see the overview of how to solve the problem and so makes the coding process easier as you have already thought about the objective of the code.
- Pseudo-code and algorithms are universally understood. This enables a full and proper discussion between groups of programmers. This discussion should provide ways in which the algorithm can be improved in terms of time-efficiency as well as space-efficiency.
- When the algorithm has been proved to be correct, the most appropriate High Level Language (HLL) in which it can be implemented is chosen and it is converted to that language.

Writing an algorithm first in pseudo-code or using a diagramming technique has several advantages:

- It enables programmers involved to have a good understanding of the overall aims of the code.
- It allows for a change in the HLL at a later stage in the development of the code.

VB.Net is often chosen as the HLL language because it is a general purpose programming language that is well-supported by Microsoft and has a large

system library by way of the .NET framework. The language also supports object-oriented and functional programming styles.

Learning how to program in VB.Net teaches you to:

- Use structured programming techniques. The word “structured” means that the code is divided and sub-divided so that the subdivision accomplishes one task only. Structuring your code also involves the use of good programming style, such as indentation of related blocks and use of meaningful identifiers for variables.
- Write readable code. VB.Net is well-known for the ease of reading code if the programmer uses comments and meaningful identifier names.
- Test your code for bugs using the Visual Studio IDE
- Prove that a problem is solvable computationally by deploying the code amongst the users

Remember the following:

- At A level, the code you develop may involve the use of mathematical formulas. You therefore need to make sure that you are able to use and manipulate algebraic equations and understand how to model real-world events mathematically.
- Your code will involve complex data structures such as lists, tables and queues.
- You will be expected to use iteration as well as recursion when producing logic-solving routines.

Overview of this handbook

Programming is a skill. Skills increase with practise. This handbook is designed to increase your programming skills by:

- Providing new situations, exercises and scenarios for your to apply your knowledge of programming.
- Helping you to understand the background to programming and how to use programming language environments (IDEs) to write code.
- Providing an opportunity for you to understand the programming theory in your syllabus through exercises and practical work.
- Providing practice exercises and extension exercises to A level standard.

Supporting materials

This workbook is one of a series produced by LMC Education Ltd for the AQA syllabus in Computer Science. Other workbooks, practice papers and programming handbooks available are:

Workbooks & Practice Papers for AQA AS & A-Level

Workbooks & Practice Papers	AS		A-Level		
	Paper 1	Paper 2	Paper 1	Paper 2	NEA
Theory Workbooks	✓	✓	✓	✓	-
Practical Workbooks	✓	-	✓	-	-
Project Workbooks	-	-	-	-	✓
Exam Revision Workbooks	✓	✓	✓	✓	-
Exam Practice Papers	✓	✓	✓	✓	-

Programming Handbooks for AQA AS & A-Level

Programming Handbooks	AS	A-Level
C#	✓	✓
Delphi	✓	✓
Java	✓	✓
Pascal	✓	✓
Pseudo-code	✓	✓
Python	✓	✓
VB.Net	✓	✓



**HOW TO USE THIS
HANDBOOK**

How to use this handbook

This handbook is designed for use with the syllabus for the AQA A level Paper 1 and Paper 2 in Computer Science.

Handbook sections

The following is a summary of how to use each of the sections in the handbook.

- **Background knowledge**

This section shows you how programs are designed and written using the VB.Net programming language. The section also covers the use of compilers and interpreters and how to remove errors from the code that you write.

- **Paper 1 programming theory**

This is an important section of the handbook and focuses on the main aspects of programming theory that can be implemented in VB.Net:

- 4.1.1.1: Data types
- 4.1.1.2: Programming concepts
- 4.1.1.3–5: Programming operations
- 4.1.1.6: Constants and variables
- 4.1.1.7: String handling
- 4.1.1.8: Random numbers
- 4.1.1.9: Exception handling
- 4.1.1.10-14: Subroutines
- 4.1.1.16: Recursion
- 4.1.2.2: Procedural-oriented programming
- 4.1.2.3: Object-oriented programming
- 4.2.1.2: Arrays
- 4.2.1.3: Fields, records and files
- 4.2.1.4: Abstract data types
- 4.2.2.1: Queues
- 4.2.3.1: Stacks
- 4.2.4.1: Graphs
- 4.2.5.1: Trees
- 4.2.6.1: Hash tables
- 4.2.7.1: Dictionaries
- 4.3.4: Searching algorithms
- 4.3.5: Sorting algorithms
- 4.3.6: Optimisation algorithms

Programming theory is explained by following the syllabus sections. You should therefore expect to read-ahead for specific sections and perform your own research on newly introduced topics.

- **Paper 2 programming exercises**

These exercises help you to apply the programming knowledge that you have gained in the theory section to topics from the syllabus that will be examined in Paper 2. These topics are:

- 4.4.2.3: Regular expressions
- 4.5.1: Number systems
- 4.5.4: Binary
- 4.5.5: Information coding systems
- 4.5.6: Representing images, sound and other data
- 4.9.1: Communication

- **Extension work**

The final part of the handbook has some further tasks to test your skills at applying your programming knowledge. The extension tasks provided are: [here]

- Task 1: Mobile Phone Expenses

In this task, you are working as a freelance programmer for a telecoms company under the direction of Wesley, who is your line manager. You have to create a computational model of different tariffs for different contracts from various competitor companies. This model will be used to create an app that allows customers to compare tariffs and choose the best one for their anticipated usage.

- Task 2: Buy-To-Let Register

Commercial loans are often used by property buyers who want to invest in domestic property and obtain a future return. This task requires you to create a register of properties for a developer to determine the annual profitability of a portfolio.

- Task 3: Weekend Planner

This task requires you to create a program to record the activities performed at the weekend. The program will be used by a housemaster of a boarding school to ensure that boarders have a full programme of activities.

- Task 4: Did I Sleep Well?

Rhythmic sleep is an important aspect of mammalian physiology. It is needed to build various systems in the body including the immune and nervous systems. In this task, you are developing an app that analyses sleep data and provide recommendations to the user regarding the amount of REM

sleep and their sleep efficiency. Logging of the data is to be performed by the accelerometer in the mobile phone.

- Task 5: Air Travel Planner

The final extension task requires you to write an app for reps in a holiday company so that they are able to meet holiday makers as they arrive at airports in the US. The task requires you to perform research of airline flights and display the relevant information for the rep at the specific hotel.

Self-assessment

After completing a particular section, you should:

- Submit your work to your teacher for marking.
- You should try to improve your work from the feedback in the Teacher's Comments on all the work you have done before moving further.
- Complete the Notes section by summarising how you can improve on your own work.



**BACKGROUND
KNOWLEDGE**

Background knowledge

This section provides the background knowledge to some important programming concepts and is divided into the following topics:

- History and features of VB.Net
- Principles of designing programs and writing algorithms
- Compilers and interpreters
- Debugging

History and features of VB.Net

Visual Basic.Net (VB.Net) is a compiled programming language that implements Microsoft's .NET framework. The programming language was originally launched in 1991 as Visual Basic. This was derived from the BASIC (Beginners All-purpose Symbolic Instruction Code) programming language. Visual Basic enabled the development of GUI-based applications through the use of forms that allowed controls such as combo-boxes to be placed on them.

In 2002, Microsoft launched VB.Net as a successor to Visual Basic. The programming language was included in the Visual Studio package, which is the Integrated Development Environment (IDE) preferred for VB.Net development. Since 2002, there have been many further versions of VB.Net developed and the latest is Visual Studio 2015.

VB.Net includes the following programming features:

- It is a readable language using English keywords.
- Variables are defined using the **Dim** keyword where necessary.
- It includes all of the standard programming constructs and statements, e.g. **If** and **Select Case** statements, **While** and **Do... Until** loops.
- Comments are shown by a single quote. Comments can be made either on the same line of code or on a separate line
- It includes a full set of mathematical operators to enable the development of programs that can model complex mathematical systems.
- It is not case sensitive.

Hello, World! In VB.Net

“Hello, World!” is the name given to the most basic program in a computer language. The program shows you how to write and structure a simple program that displays “Hello, World!”.

The following shows how VB.Net code is written:

```
Imports System

Public Module modMain
    Sub Main()
        'Display the output
        Console.WriteLine ("Hello, World!")
        Console.ReadKey()
    End Sub
End Module
```

Note the following:

- The **Imports System** line tells the compiler to use the **system** namespace.
- The **Public Module modMain** line defines the application’s code module container. This container is terminated by a corresponding **End Module** line. Modules must contain one or more procedure. It is usual for a procedure to be either a **Sub** or **Function**.
- The **Sub Main()** statement is used to define the start of the code section. The code is terminated by a corresponding **End Sub** line.
- The **Console.WriteLine** statement is used to display the output.
- The **Console.ReadKey()** command is used to pause the screen display.

Using the Visual Studio IDE

An IDE enables you to work more efficiently than a text editor because of the following features:

- Syntax highlighting
- Line numbers
- Auto-completion of brackets and indentation

An IDE also enables you to perform debugging of your code.

To compile your code using the Visual Studio IDE:

- Choose **File, New, Project** and then click on Visual Basic.
- Choose Console Application and specify the name and location of the project.
- Click OK and the new project appears in the Solution Explorer.
- Use the Code Editor to write the code.
- Click the Run button to execute the code.

Exercise 1

Describe how the following is performed in VB.Net.

Whitespace indentation:.....

.....

If... then... else construct:.....

.....

.....

.....

.....

Constant and variable declaration:

.....

.....

.....

Assignment statements:

.....

.....

.....

.....

Defining a procedure:

.....
.....
.....

Defining a function:

.....
.....
.....
.....

Performing addition, subtraction, multiplication and division:

.....
.....
.....
.....

Performing integer division:

.....
.....
.....
.....

(10 marks)

Principles of designing programs and writing algorithms

There are three combining principles of writing algorithms and VB.Net code. These are:

- **Sequence**

Sequence consists of a series of programming instructions that are executed one after another. A sequence of instructions can include selection and iteration instructions. A flowchart is often used to indicate the flow of the program instructions.

- **Selection**

Selection consists of instructions that change the sequence. They are also known as choice instructions. Selection parts of a program contain sequence statements.

- **Iteration**

Iteration occurs when the sequence of instructions is repeated. Iterative parts of a program may contain sequence and/or selection statements and a condition that has to be met so that the iteration can finish.

Selection and iteration structures can be nested, or placed within each other. This allows more complex algorithms to be created where an action can be performed on a set of data many times.

Designing algorithms consists of the following stages:

- (a) Produce an abstract model of the computation in the problem that removes complexities by breaking them down into smaller sub-tasks. This is known as code abstraction. This process may involve several stages until the model is sufficiently simplified.
- (b) Ensure that the data is modelled correctly by removing unnecessary and irrelevant details. This is known as data abstraction. The data model is usually implemented using a file-based system. For complex data models involving large quantities of data and relationships between tables, a DBMS is used.
- (c) Write the model into pseudo-code or English-like statements, ensuring that the code follows good style conventions.
- (d) Test the model for correctness and efficiency.

Designing programs consists of the following stages:

- (a) Convert the algorithm model into VB.Net code.
- (b) Convert the data model into files or implement it using a database system.
- (c) Debug and test the code.
- (d) Deploy the program.
- (e) Review the efficiency of the program in solving the problem.

Program modules such as functions and subroutines can be re-used by the program or by other programs. The technique of recursion is used in this way.

Algorithms rely on the designer to choose the correct data structure to model the real-world data. Examples of these include:

- Stacks
- Queues
- Trees
- Hash tables
- Dictionaries

Exercise 2

Explain what is meant by the following programming terms.

(a) Procedural paradigm:

.....

.....

.....

.....

(b) Structured approach to program design:

.....
.....
.....
.....

(c) Hierarchy charts:

.....
.....
.....

(d) Polymorphism:

.....
.....
.....
.....
.....

(e) Class diagram:

.....
.....
.....
.....
.....

(10 marks)

Assemblers, compilers and interpreters

Explanation

Assembly language is an alternative method of writing programs compared to High Level Languages (HLLs). In VB.Net, you learn how to write code using high level language constructs. In reality, these are translated by the compiler into machine code instructions that are understood by the CPU. Assembly language is an intermediate stage between machine code and the VB.Net programming language. Writing in assembly language enables any inefficiencies of the interpreter to be removed by the programmer.

Here is an example of assembly language code for the “Hello, World!” program:

```
;-----  
; helloworld.asm  
;-----  
  
global  _main  
extern  _printf  
section .text  
  
_main:  
    push    message  
    call    _printf  
    add     esp, 4  
    ret  
  
message:  
    db     'Hello, World!', 10, 0
```

Although it is not possible to write and mix assembly language within VB.Net code, it is useful to understand the syntax of the available commands.

Syntax

The following syntax is used for common assembly language instructions:

Mnemonic	Instruction
ADD	Add
SUB	Subtract
STA	Store into accumulator

Mnemonic	Instruction
LDA	Load into accumulator
BRA	Branch always
BRZ	Branch if flag is zero
BRP	Branch if flag is positive
INP	Input
OUT	Output
HLT	End program

Exercise 3

(a) Define the terms compiler, interpreter and intermediate code.

Definition of a compiler:

.....

.....

.....

Definition of an interpreter:

.....

.....

.....

.....

Definition of intermediate code:

.....

.....

.....

.....

(3 marks)

(b) VB.Net is primarily a compiler. Explain why some programmers use interpreters compared to compilers.

.....

.....

.....

.....

.....

.....

.....

.....

(3 marks)

Exercise 4

When writing code in assembly language, it is important to know the addressing mode being used by the instruction. List ten assembly language instructions by dividing them into opcodes and operands. Explain their purpose and addressing mode.

#	Instruction Opcode & Operand	Purpose	Addressing mode
1			
2			
3			
4			
5			
6			

#	Instruction Opcode & Operand	Purpose	Addressing mode
7			
8			
9			
10			

(10 marks)

Debugging

The process of finding errors in an algorithm or a program is called debugging. Debugging involves tracing through a program or algorithm in a line by line way to check that it operates as intended. Debugging can be performed manually or by using a computer.

- Debugging is performed manually using a trace table as shown in this handbook.
- When using a programming environment, debugging is completed using a watching function to show how variables change during the execution of the program. Break points can also be included in the code to only test up to a certain stage of the program.

Exercise 5

The following is the pseudo-code for a recursive binary search algorithm:

```
function binarySearch(values, target, start, end)
{
    /** check if value exists **/
    if (start > end){
        return -1;
    }

    /** get the midpoint of start and end values **/
    var middle = floor((start + end) / 2);

    /** get the new value at the midpoint **/
    var value = values[middle];
}
```




```
/** check if the value is greater than target
/** if it is then recursively call to the left subtree */
if (value > target) {
    return binarySearch(values, target, start, middle-1);
}

/** check if the value is less than target
/** if it is the recursively call the right subtree */
if (value < target) {
    return binarySearch(values, target, middle+1, end);
}

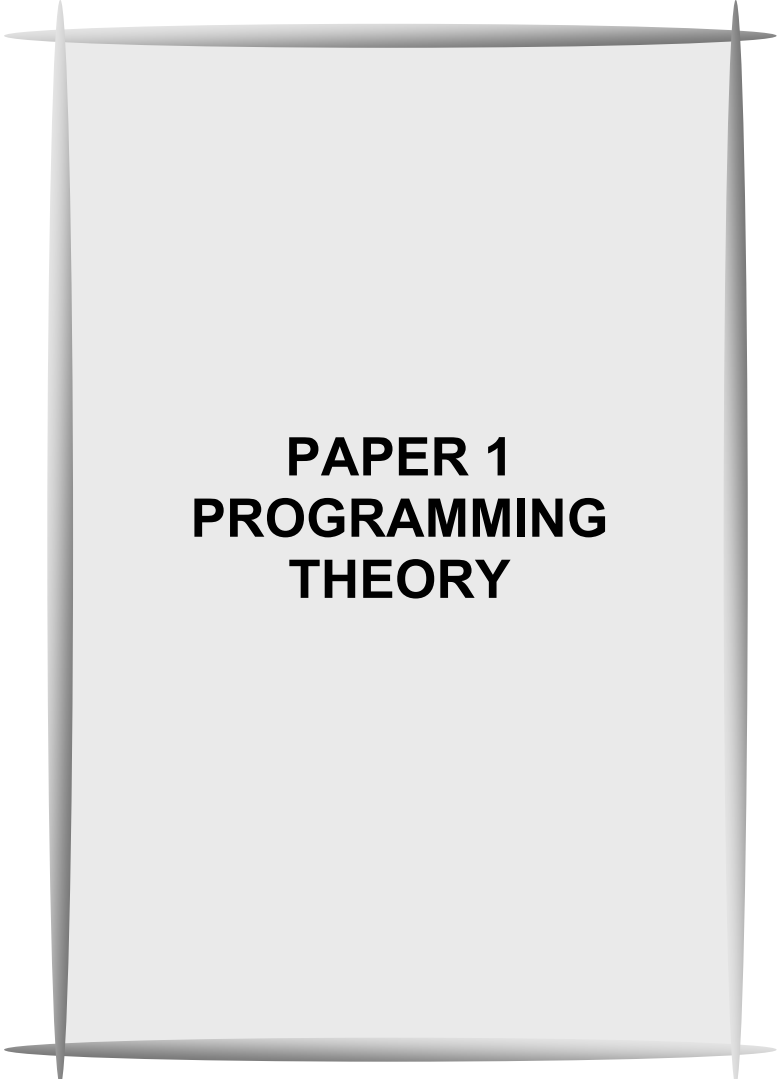
/** target found */
return middle;
}
```

- (a) Convert this algorithm to VB.Net code. Remember to create a program in your IDE. Save your file regularly to prevent data loss.
- (b) Execute the code by running the debugging feature to check for errors.
- (c) Print your code and save for future reference.

(10 marks)

 Teacher's Comments	Marks Awarded: <u> 46 </u>
<p>What you did well:</p> <p>What you need to work on:</p>	

Student's Comments
<p>What can I do to improve?</p>



**PAPER 1
PROGRAMMING
THEORY**

Paper 1 Programming theory

This section provides the programming theory for Paper 1 of the syllabus.

Section summary

The section is divided into the following topics:

- 4.1.1.1: Data types
- 4.1.1.2: Programming concepts
- 4.1.1.3–5: Programming operations
- 4.1.1.6: Constants and variables
- 4.1.1.7: String handling
- 4.1.1.8: Random numbers
- 4.1.1.9: Exception handling
- 4.1.1.10-14: Subroutines
- 4.1.1.16: Recursion
- 4.1.2.2: Procedural-oriented programming
- 4.1.2.3: Object-oriented programming
- 4.2.1.2: Arrays
- 4.2.1.3: Fields, records and files
- 4.2.1.4: Abstract data types
- 4.2.2.1: Queues
- 4.2.3.1: Stacks
- 4.2.4.1: Graphs
- 4.2.5.1: Trees
- 4.2.6.1: Hash tables
- 4.2.7.1: Dictionaries
- 4.3.4: Searching algorithms
- 4.3.5: Sorting algorithms
- 4.3.6: Optimisation algorithms

Note that sections 4.1.1.1 to 4.1.1.14 and 4.2.1.2 to 4.2.1.3 are common to the AQA AS syllabus.

4.1.1.1: Data types

Explanation

Data types consist of integers, reals, Boolean, character, string, date/time, records and arrays. Data types are assigned to variables and constants so that they can store a data value that matches the intended type. Doing so reduces errors that can occur with mismatched data, e.g. attempting to store a string value in an integer variable.

The process of assigning a data type to a variable is called declaration of a variable.

Syntax

In VB.Net, a variable is declared by the **Dim** statement.

```
Dim Firstname As String = "Steve"
Dim noTeaPacketsBought As Integer = 5
Dim userid(5) As String
userid(1) = "1431"
```

Exercise 6

(a) Write VB.Net statements to use variables with the following data types:

Integer:

Real/float:

Character:

Boolean:

String:

Pointer/reference:

.....

Record:

.....

.....

Array:

.....

(8 marks)

(b) Explain how a user-defined data type can be created in VB.Net.

.....

.....

.....

(2 marks)

4.1.1.2: Programming concepts

Explanation

You have already used the main constructs to create code. Remember these are sequence, selection, and iteration. Programmers also use recursion as a fourth construct.

In keeping with good programming style, try to apply the following points when using programming constructs:

- All constants and variables must be declared as a data type. This is considered to be good programming style.
- It is good practice to make sure that all constants and variables have values assigned to them, even if these are blank or null values.
- Use functions to group together instructions that perform a specific task. Indent your code to show that it forms part of a function.
- The processes of abstraction and decomposition are used to produce sub-tasks.

Some characters are reserved and not allowed to be used in variable names. Examples of these are:

- #, { }, *
- “
- File path separators

Syntax

The following is the syntax used for selection and iteration:

- **Selection**

This is achieved using **if... then... else** statements.

```
If type="z" Then
    Console.WriteLine ("Type Z selected")
Else
    Console.WriteLine ("Try again")
End If
```

- **Count controlled iteration**

This is achieved as follows:

```
For j=0 To 10
    Console.WriteLine (j + j*2)
Next
```

- **Condition controlled iteration**

```
While j <> 89
    'carry out actions
End While
```

Exercise 7

(a) An estate agent earns commission on the sale of a house at a rate of 7.5%. Write a VB.Net function to calculate the amount of commission to be paid on these transactions: £167000, £445000, £87000.

Your answer:

.....

.....

.....

.....

.....

.....

(3 marks)

(b) Stamp duty is paid on the purchase of a house. The rates are as follows:

Band	Normal rate
less than £125k	0%
£125k to £250k	2%
£250k to £925k	5%
£925k to £1.5m	10%
rest over £1.5m	12%

4.1.1.3–5: Programming operations

Explanation

Program code consists of a number of operations that can be performed on data. These operations can be divided into the following categories:

- **Arithmetic operations**

These include mathematical operations such as addition, subtraction, division, exponentiation (raising a value to a power) and truncation.

- **Relational operations**

These are used to compare two values and variables.
Examples are: >=, <=

- **Boolean operations**

Examples are: AND, OR, NOT and XOR.

Syntax

The following is an example of the use of the “not equal to” operator.

```
If (var1 <> var2) And (var1 < 20000) Then
    breakamt = var1 * 0.05
End If
' remainder of code
```

Exercise 8

Write VB.Net statements for the following operations, stating their results.

Operation	VB.Net statement	Explanation of results
Rounding		
Equal to		

Operation	VB.Net statement	Explanation of results
Less than or equal to		
Greater than		
Greater than or equal to		
XOR		
Truncation		
Exponentiation		

(10 marks)

4.1.1.6: Constants and variables

Explanation

Constants and variables are used to store data which can be used while the program is executing.

- Constants are used to store values that do not change while the program is executing.
- Variables store values that can change during program execution.
- Both constants and variables are usually declared to be of a specific data type.
- Both constants and variables can be local or global.

Exercise 9

(a) Explain the meaning of the following terms:

Term	Meaning
Local	
Global	

(b) Give examples of VB.Net code to declare the following constants and variables.

Local constant studentAge:

.....

.....

Local variable parentSurname:

.....

.....

Global constant tutorGroupSize:

.....

.....

Global variable tutorGroupTutorname:

.....

.....

(5 marks)

4.1.1.7: String handling

Explanation

Strings are used to store a sequence of more than one text or non-numeric characters. The text characters are taken from the ASCII character set or the Unicode character set.

Several operations can be performed using strings:

- The position of a character in the string, or the length of a string, can be determined.
- A substring can be extracted.
- Two strings can be combined together. This is known as concatenation.

Syntax

To obtain the length of a string:

```
Dim lenString As Integer = Len(stringname)
```

To obtain a substring:

```
Dim substring As String =  
    strname.Substring(<start>, <numberOfChar>)
```

Exercise 10

Write the VB.Net code for a procedure as follows:

- Input a string from the user.
- Display each word in the string.

Your answer:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

(8 marks)

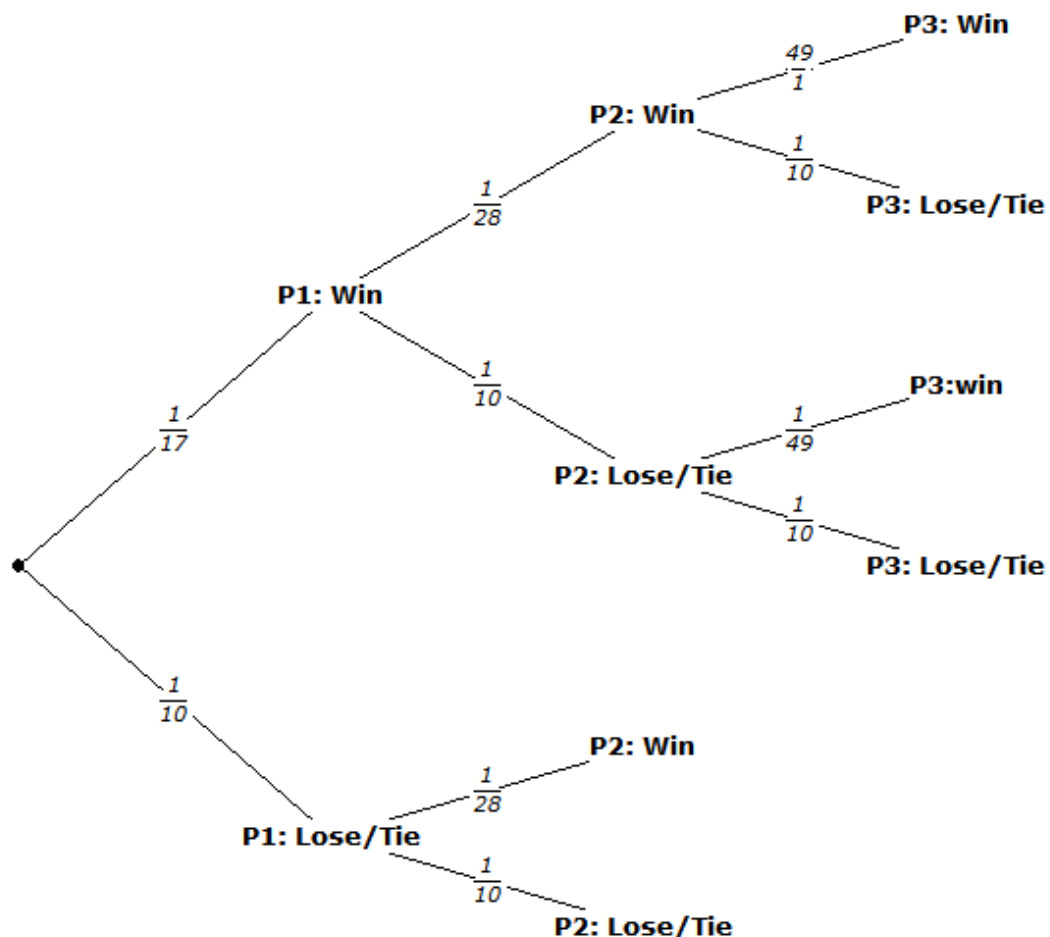
4.1.1.8: Random numbers

Explanation

Random numbers are used for mathematical and simulation purposes.

An example is the Monte Carlo simulation method. This is a method for iteratively evaluating a mathematical model using sets of random numbers. This method is often used when the model is complex, nonlinear, or involves more than just a couple of uncertain parameters.

An example is the use of probability trees to indicate the combined probabilities of a sequence of random events.



Syntax

To obtain a random number between 1 and 100:

Randomize()

```
Dim randomnumber As Integer = CInt(Int((rnd()*100)) + 1)
```

Exercise 11

A probability tree is often used to show a sequence of random events with the associated probabilities attached. Write a VB.Net function to return the probabilities of all events in the probability tree shown on the previous page.

Your answer:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

(5 marks)

4.1.1.9: Exception handling

Explanation

When creating programs, it is usual for errors to occur during the execution of the program itself. These errors are usually logical or user-generated errors and can be handled by the program so that they do not cause it to halt execution. Such errors are called exceptions. Special code statements can be created to handle such exceptions.

Examples of exceptions are:

- Opening a non-existent file.
- Searching a list for a non-existent value.
- Referencing a non-existent variable.
- Calling a non-existent procedure or function (method).

Syntax

To trap an exception:

```

Try
    <statements>
Catch ex As Exception when <var> = <value>
    <error handling messages>
Exit Sub
End Try
    
```

Exercise 12

Revisit the code that you have written in the previous exercise and add exception handling to it.

Your answer:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

(8 marks)

4.1.1.10-14: Subroutines

Explanation

Subroutines consist of functions and procedures. These are used to subdivide tasks into logical sections.

Subroutines use variables to process their own data or to obtain data from outside the subroutine as follows:

- **Parameters**

Data passed to the subroutine to process is known as a parameter.

- **Global variables**

Data accessed by the subroutine from other subroutines or the main program is known as global variables.

- **Local variables**

Data used by the subroutine and then discarded is known as local variables.

Syntax

A procedure is declared as:

```
Sub procname(ByVal parameter As <type>)  
    <statements>  
    <calculation>  
End Sub
```

A function is declared as:

```
Function funcname(ByVal parameter As <type>)  
    As <returntype>  
    <statements>  
    funcname = result  
End Function
```

Exercise 13

The following is a table of conversion rates for different currencies.

Note: USD = US Dollar, EUR = Euro, GBP = British Pound.

Auto-refresh 14x 0 : 34		 USD	 EUR	 GBP
	1 USD Inverse:	1.00000 1.00000	0.91792 1.08942	0.70850 1.41143
	1 EUR Inverse:	1.08942 0.91792	1.00000 1.00000	0.77185 1.29558

Write a VB.Net function that will perform the following conversions:

- (a) USD100 to Euros.
- (b) EUR100 to British Pounds.
- (c) GBP100 to US Dollars.

Your answer:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....
.....
(10 marks)

4.1.1.16: Recursion

Explanation

Recursion is a technique that is used to improve algorithm efficiency by reusing code. A base case is first implemented and then a terminating condition for the recursive call is decided and coded.

Syntax

An example of a recursive function is:

```
Function funcname(ByVal p1 As <type>) As <returntype>
    <statements>
    funcname(variables)
    funcname = result
End Function
```

Exercise 14

A Fibonacci sequence is a series of numbers where the next number is found by adding up the two numbers before it.

An example is:

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, ...

The sequence can be recursively defined as follows:

$$F_n = \begin{cases} 1, \\ F_{n-1} + F_{n-2} \end{cases}$$

Where

- The first case is where $n = 0$ or 1 .
- The second case is where $n \geq 2$.

.....

.....

.....

.....

.....

(5 marks)

(b) The Golden Ratio is the number 1.618034... Extend your function to prove the relationship between the Golden Ratio and the Fibonacci sequence.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....
.....
.....
.....
.....

(5 marks)

4.1.2.2: Procedural-oriented programming

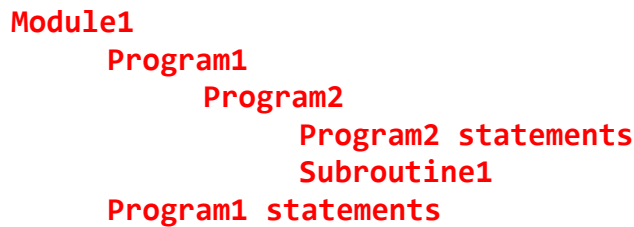
Explanation

Procedural-oriented programming involves using the structured approach to solve programming problems. This is achieved using techniques such as splitting programs into smaller subroutines until each subroutine can be solved.

Hierarchy charts are also used when designing programs to show how the sub-tasks fit together.

Syntax

A procedural program can be divided into smaller programs as follows:



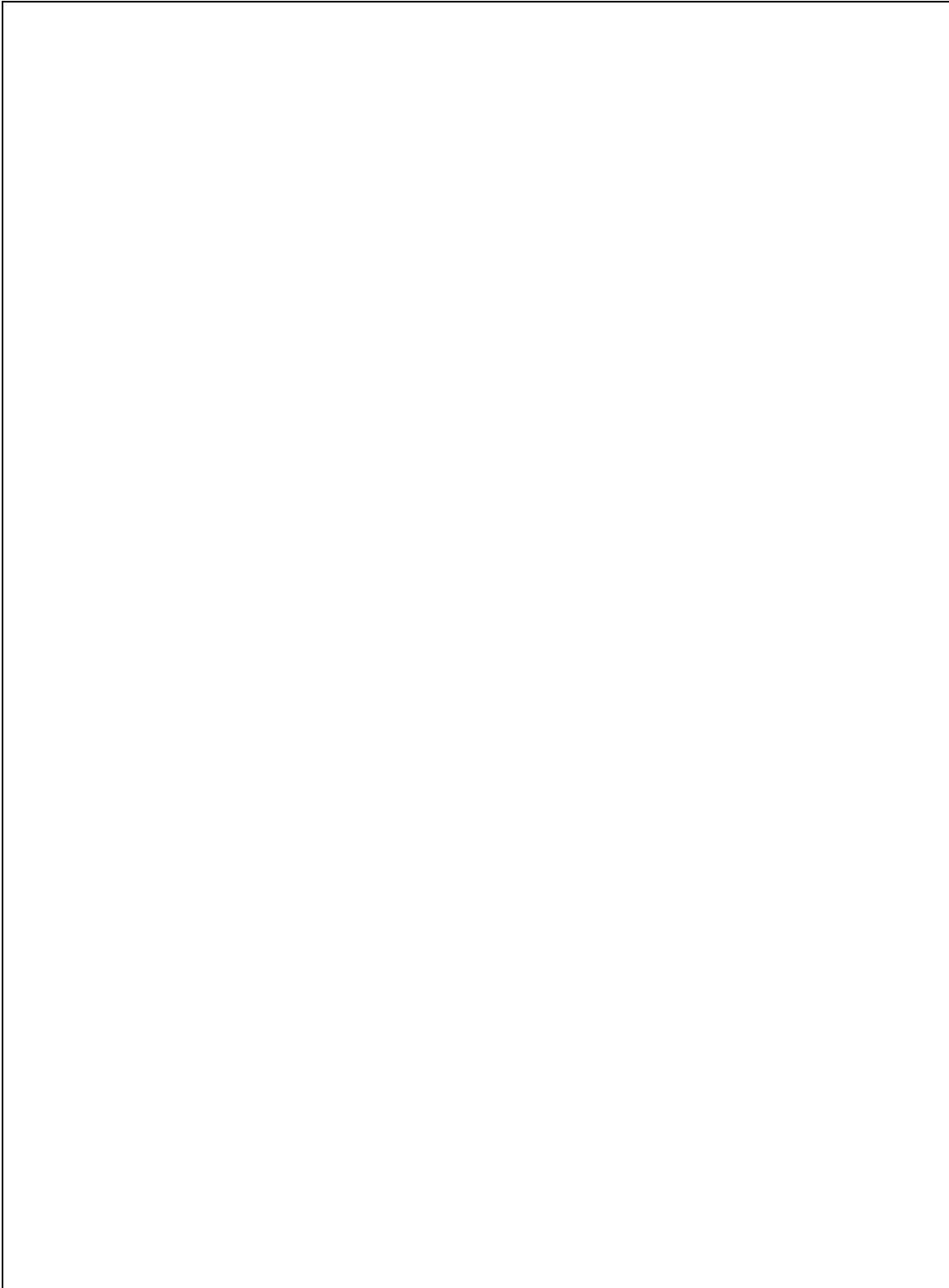
Exercise 15

(a) Write the outline using VB.Net statements for a program for a picking list in a warehouse. The program should enable the user to add, delete, edit and save the picking list.

Your answer:
.....
.....
.....
.....

(b) Draw a hierarchy chart for app showing how the code combines together to form the main program.

Your answer:



(5 marks)

4.1.2.3: Object-oriented programming

Explanation

In this style of programming, data and its properties is separated from the actions that can take place on it.

Several important concepts are used in object-oriented programming:

- **Class and methods**

A class defines the methods and properties for the object.

- **Encapsulation and inheritance**

Objects can be defined from other objects and so obtain the methods and properties of that objects. Encapsulation is applied to variables in the data.

Syntax

A class definition starts with the keyword **Class** followed by the class name and the class body, ended by the **End Class** statement.

The following is used to define vehicles in an object-oriented style:

```
Class Vehicle  
    Public Sub alignWheels(ByVal wheeltype As Integer)  
        'code  
    End Sub  
End Class
```

Exercise 16

(a) Write object-oriented VB.Net statements for a building simulation game. Assume the following:

- The base class called building, and subclasses are residential, commercial and utility.
- All buildings have a location and a value.
- All buildings pay building tax.
- Three types of buildings exist:
 - Residential buildings provide living space.
 - Commercial buildings provide working space.
 - Utility buildings provide services.

(b) Draw a class diagram for your program.



(5 marks)

4.2.1.2: Arrays

Explanation

An array stores homogenous data items, i.e. data of the same data type. The advantage of an array is that data is stored in RAM, so access to the data is relatively quicker than data stored on secondary storage. For this reason, data is often transferred from a file on disk into an array in RAM. The number of items that can be stored in an array is called the number of elements, and cannot be changed whilst the program is executing.

A **For... Next** loop is usually used to manipulate the items in an array. Brackets are used to indicate the item number in the array being declared or accessed.

Syntax

To declare an array:

```
Dim <arrayname>(noOfItems) As <type>
```

Where some common values for type are:

```
Integer = signed integer  
Double = floating point number
```

To assign data into an array:

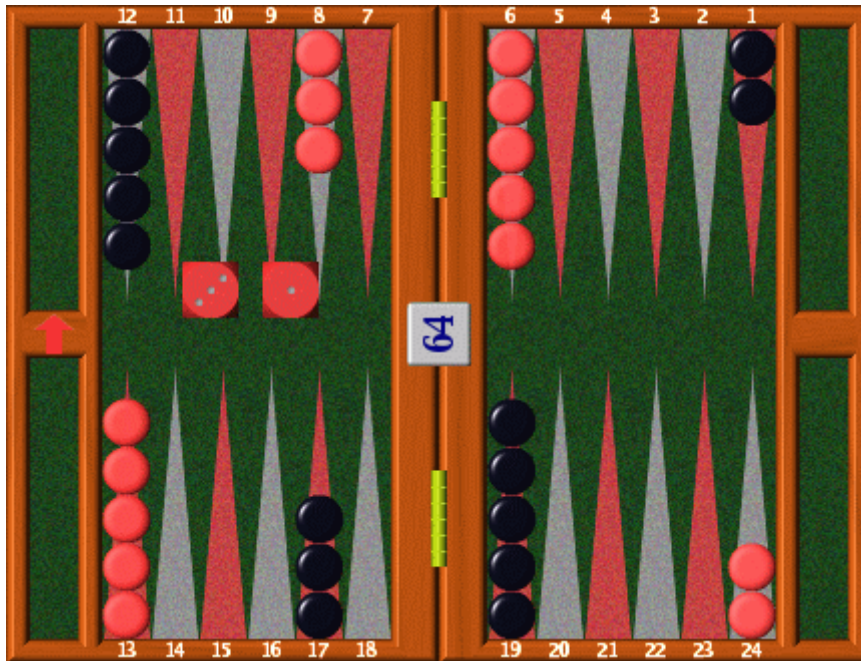
```
arrayname(0) = <item0>  
arrayname(1) = <item1>  
arrayname(2) = <item2>
```

To output the items in an array:

```
Console.WriteLine (arrayname(item))
```

Exercise 17

Backgammon is a two player game where playing pieces are moved according to the roll of dice, and a player wins by removing all of his pieces from the board before his opponent. A Backgammon board is shown below:



- (a) Research the rules of Backgammon and how it is played.
- (b) Create a VB.Net program to simulate the game of Backgammon.

Your answer:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

4.2.1.3: Fields, records and files

Explanation

A file is a data structure that is designed for storing data primarily on secondary storage. Once the data has been saved in the file, it can be retrieved at a later date for further processing. Each row (or record) in the file is sub-divided into columns (or fields). A good analogy for a file is a spreadsheet table where data is divided into columns and rows.

When programming, it is usual for a separator is used to divide the fields in the record. This is known as the file format. In this example, a comma is used to separate the fields:

```
Record1 = smith,Jonas,07/09/97,23 Swan Street,Bristol,BS1 3VZ
```

To access the data in each record, the file format needs to be known in advance by the programmer.

To use the file handling library in VB.Net, add this line to your code:

```
Imports System.IO
```

Syntax

To open a file for reading and then close the file:

```
Using iFile As StreamReader =  
        New StreamReader("filename")  
        <statements>  
End Using
```

To read a record from a file:

```
Dim line = iFile.ReadLine()
```

To determine the end of the file:

```
While (line <> Nothing)  
        Console.WriteLine(line)  
        line = iFile.ReadLine()  
End While
```

Exercise 18

Write a VB.Net program to enter the medical details from a user as follows:

- Height
- Weight
- Blood group
- Medical conditions
- Dr’s name and address
- Allergies and reactions
- Medications

Save these to a file then retrieve the saved data and display to the user.

Your answer:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

(10 marks)

4.2.1.4: Abstract data types

Explanation

Abstract data types are used to represent data that cannot be modelled using the primitive data types provided in programming languages. They are data types that exist to model real world data that is dynamic or static in nature. Static data structures do not change their size during the execution of the program and are useful for pre-known data. Dynamic data structures can change during run-time and are useful for user-generated data.

Each data type can be used in different situations and is particularly suited to the data model.

When designing programs, it is important to choose the correct abstract data type. The following is a list of the common abstract data types:

- **Graphs**

Graphs are used to represent complex relationships with interconnecting nodes.

- **Trees**

A tree is a non-linear, hierarchical arrangement of data that is a special example of a graph. Trees are connected undirected graphs with no cycles.

A binary tree is a special example where each node has only two children.

- **Hash tables**

A hash table consist of a series of keys that map to data stored in memory locations indicated by the keys. To generate the memory location, a hash function is needed to map the data to its location in the hash table.

Hashing can produce collisions where two data items produce the same key. In this situation, the hashing algorithm needs to be recalculated or an alternative method of key generation obtained.

Exercise 19

Write VB.Net statements to create a hash table.

Your answer:

.....
.....
.....
.....
.....

(5 marks)

4.2.2.1: Queues

Explanation

Queues are First-In, First-Out (FIFO) data structures. There are three types of queues that are most used by programmers to simulate real world queuing systems. These are:

- **Linear queues**

Entries enter a queue until it is full. Entries are removed on a first-come, first-served basis.

- **Circular queues**

Entries can wrap around and go to the back of the queue until it is full if they cannot exit the queue.

- **Priority queues**

Data is not removed on a FIFO basis, but on a priority basis.

Syntax

Examples of queues are:

- To define a queue

```
Dim Q As Queue(Of Integer) = New Queue(Of Integer)()
```

- To add an item to a queue

```
Q.Enqueue(<data item>)
```

Exercise 20

(a) Write VB.Net statements to create linear, circular and priority queues.

Linear queue answer:

.....

.....

.....

.....

Circular queue answer:

.....

.....

.....

.....

.....

Priority queue answer:

.....

.....

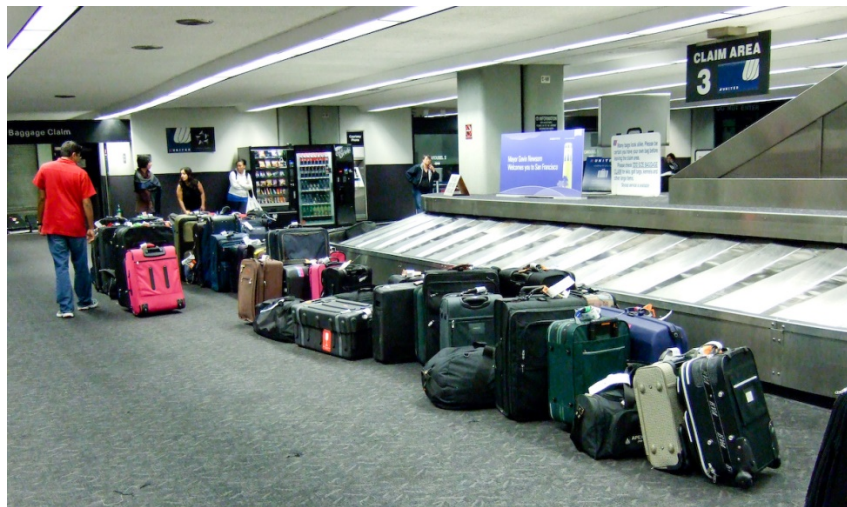
.....

.....

.....

(5 marks)

(b) A luggage carousel machine with a capacity of 50 items operates at an airport as shown below:



Items are physically added to the carousel by baggage handlers and removed by passengers. Write a VB.Net program to simulate the following carousel processes:

- Add a baggage to a queue by the baggage handler
- Remove an item from a queue by a passenger
- Check if the carousel is full

Your answer:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

4.2.3.1: Stacks

Explanation

A stack is a Last-In, First-Out (LIFO) data structure. The common operations used with a stack include:

- Adding an item to the top of the stack (push)
- Removing an item from the top of the stack (pop)
- Looking at the top element (peek)
- Testing the stack

Syntax

Below are examples of syntax for a stack:

- To define a stack
Dim S As New Stack
- To add an item to a stack
s.Push(item)
- To remove an item from a stack
item=s.Pop()

Exercise 21

Write VB.Net statements to implement the following stack operations:

- Testing for an empty stack
- Testing for a full stack
- Performing a peek at the top element
- Removing an item from the stack

Your answer:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

(10 marks)

4.2.4.1: Graphs

Explanation

Graphs are used to represent complex relationships with interconnecting nodes.

Syntax

A graph is represented using the syntax of arrays or lists.

Exercise 22

(a) Research how electrical power is generated using the grid system. Write VB.Net code to represent a graph of an electrical power grid. Your answer should include local and national substations.

Your answer:

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

(10 marks)

(b) The Oracle of Bacon is a web site that uses graph theory to connect other actors and actresses to Kevin Bacon. Write the VB.Net code to traverse the graph created in this database to find an actor and provide the Bacon number.

Your answer:

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

.....

.....

.....

.....

.....

.....

.....

(10 marks)

4.2.5.1: Trees

Explanation

A tree is a non-linear, hierarchical arrangement of data that is a special example of a graph. Trees are connected undirected graphs with no cycles.

A binary tree is a special example where each node has only two children.

Syntax

A tree is usually represented using an array or a linked list.

Exercise 23

Elvis Presley wrote many songs that have been covered by other artists. A programmer wishes to create an “Elvis” app. One of the parts of the app is to provide a search of songs that have been covered by other artists. Write VB.Net code to initialise a tree made from ten songs and the artists that have covered each song.

Your answer:

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

(10 marks)

4.2.6.1: Hash tables

Explanation

A hash table consist of a series of keys that map to data stored in memory locations indicated by the keys. To generate the memory location, a hash function is needed to map the data to its location in the hash table.

Hashing can produce collisions where two data items produce the same key. In this situation, the hashing algorithm needs to be recalculated or an alternative method of key generation obtained.

Exercise 24

(a) Write a VB.Net function that generates a hash value for a key using the prime number of 31.

Your answer:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

(10 marks)

(b) Extend your code so that a record of hash values is kept and collisions are noted.

Your answer:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

(10 marks)

4.2.7.1: Dictionaries

Explanation

A dictionary is a container that stores key-element pairs in a similar way to a priority queue. Dictionaries allow find, insertion & removal operations. The data can be either ordered or unordered.

Syntax

An array is used to store the elements and a hash algorithm is used to map the keys to the array.

Exercise 25

Write VB.Net statements to simulate a log file in a computer system. The code should contain procedures to insert, and find items in the log file.

Your answer:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

(10 marks)

4.3.4: Searching algorithms

Explanation

Searching algorithms include:

- Linear searching, where a list is searched from the start to the end.
- Binary searching, where a list is divided into half successively.
- Binary tree search, where a binary tree is searched successively.

Syntax

All searches use arrays or trees to perform the searching. The syntax of the statements used should follow the convention for these data structures.

Exercise 26

Write VB.Net statements for a linear search of 10 items.

Your answer:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

(10 marks)

4.3.5: Sorting algorithms

Explanation

Sorting is used to produce a list of data items that are in a specific order, e.g. alphabetical or chronological.

Two common sorting methods are:

- **A bubble sort**

This method is the most inefficient time-wise. It requires a minimum of one iteration to perform the sort. Large values are usually sorted first.

- **A merge sort**

This method is a recursive algorithm that splits the unsorted list in half recursively. Once the halves are sorted, a merge is performed to combine them into a single sorted list.

Syntax

Both sorting algorithms use arrays to perform the operations. The syntax of this data structure should be used.

Exercise 27

Write VB.Net code for a merge sort. Assume the following:

- N = the number of data items in the unsorted list
- A = an array containing the unsorted data

Your answer:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

(10 marks)

4.3.6: Optimisation algorithms

Explanation

Optimisation involves finding the shortest path between points. Dijkstra’s algorithm is often used to determine the shortest path. The algorithm finds the shortest path between that node and every other node and can be used for route planning on road networks.

Syntax

The algorithm uses arrays to store vertices, best estimates of length and predecessors of the vertex. Iteration is accomplished using a while loop. The syntax of the statements used is the same as those for manipulation of arrays.

Exercise 28

Write VB.Net code for Dijkstra’s algorithm. Assume the following:

- D = array of best estimates
- P = array of predecessors for each vertex
- S = array of shortest paths
- V = array of remaining vertices

Your answer:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....


.....

.....

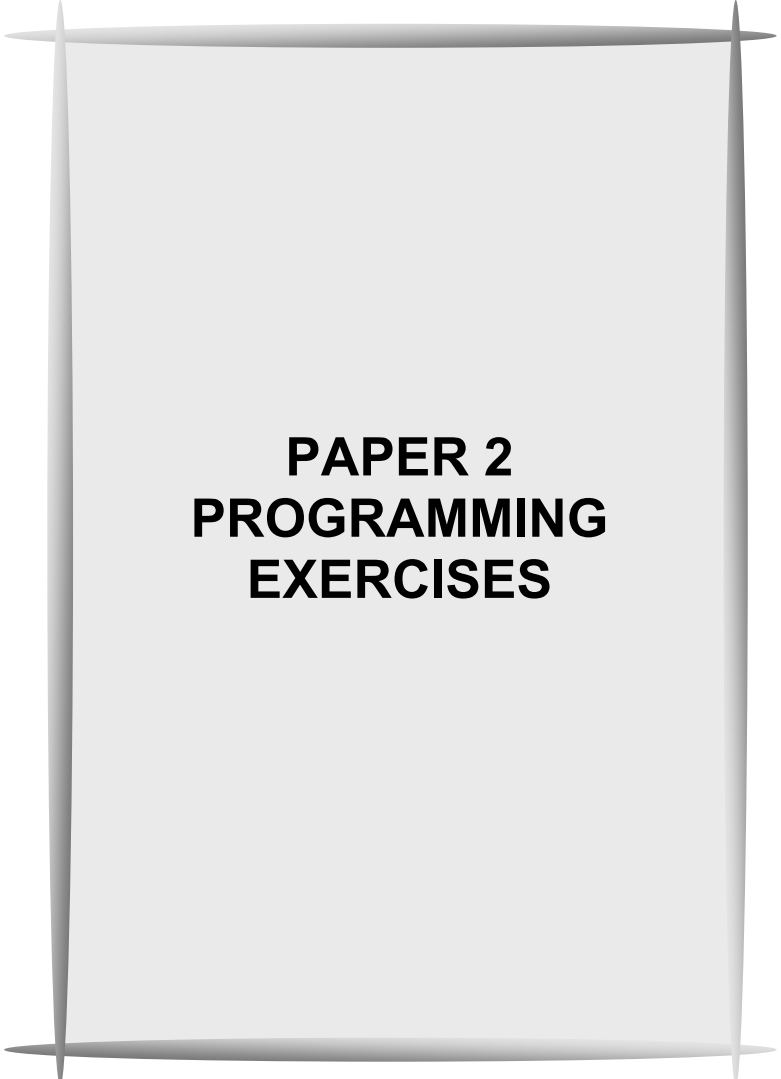
.....

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

(10 marks)

 Teacher's Comments	Marks Awarded: <u> </u> 229
<p>What you did well:</p> <p>What you need to work on:</p>	

Student's Comments
<p>What can I do to improve?</p>



**PAPER 2
PROGRAMMING
EXERCISES**

Paper 2 Programming exercises

The following is a selection of programming exercises from Paper 2 of the syllabus.

Section summary

The section is divided into the following topics:

- 4.4.2.3: Regular expressions
- 4.5.1: Number systems
- 4.5.4: Binary
- 4.5.5: Information coding systems
- 4.5.6: Representing images, sound and other data
- 4.9.1: Communication

4.4.2.3: Regular expressions

Explanation

A regular expression is a method of describing a set of values or objects. Regular expressions are used to describe languages in a shorthand notation and for string matching and manipulation.

When using regular expressions to search for characters in a string, the following apply:

- Anchors are used to specify the position of the pattern in relation to a line of text, e.g. the symbols “**^**” and “**\$**”.
- Character sets match one or more characters in a single position, e.g. the symbol “**#**”.
- Modifiers specify how many times the previous character set is repeated, e.g. the symbol “*****”.

Example

- **^A** matches all text that has the character “**A**” at the beginning of the line.
- **A\$** matches all text that has the character “**A**” at the end of the line.

Exercise 29

Write a VB.Net function that reads the file **mail.dat** and displays all lines that begin with the text **"From:"**.

Your answer

Line	VB.Net code
01.	
02.	
03.	
04.	
05.	
06.	
07.	
08.	
09.	
10.	
11.	
12.	
13.	
14.	
15.	
16.	
17.	
18.	
19.	
20.	
21.	
22.	
23.	
24.	
25.	
26.	
27.	

(5 marks)

4.5.1: Number systems

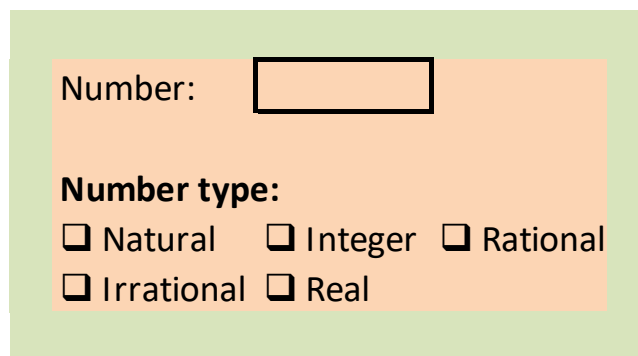
Explanation

A number system is a method of representing the different types of numbers that can exist. These numbers can then be used in a computer system for the application that they are intended for. The number systems that you are expected to be familiar with are:

- Natural numbers, e.g. 0, 1, 2, 3, 4, ...
- Integer numbers, e.g. -4, -3, -2, -1, 0, 1, 2, 3, 4
- Fractions or rational numbers
- Numbers that cannot be written as fractions
- Real numbers, which includes all possible numbers
- Ordinal numbers, which are used to indicate a position in an ordered set

Exercise 30

(a) Write a VB.Net program for a program that accepts a number from the user and displays the type of number that it is. Here is a screenshot of the user interface:



The screenshot shows a user interface with a light green background. It contains a label 'Number:' followed by a text input box. Below this is a label 'Number type:' followed by five radio button options: 'Natural', 'Integer', 'Rational', 'Irrational', and 'Real'.

Your answer

Line	VB.Net code
01.	
02.	
03.	
04.	
05.	

Line	VB.Net code
06.	
07.	
08.	
09.	
10.	
11.	
12.	
13.	
14.	
15.	
16.	
17.	
18.	
19.	
20.	
21.	
22.	
23.	
24.	
25.	
26.	
27.	
28.	
29.	
30.	
31.	
32.	
33.	
34.	
35.	

(8 marks)

(b) Extend your code so that the program displays a confirmation if the number is an ordinal number.

Your answer

Line	VB.Net code
01.	
02.	
03.	
04.	
05.	
06.	
07.	
08.	
09.	
10.	
11.	
12.	
13.	
14.	
15.	
16.	
17.	
18.	
19.	
20.	
21.	
22.	
23.	
24.	
25.	
26.	
27.	
28.	
29.	

Line	VB.Net code
30.	
31.	
32.	
33.	
34.	
35.	
36.	
37.	
38.	
39.	
40.	
41.	
42.	
43.	
44.	
45.	
46.	
47.	
48.	
49.	
50.	

(10 marks)

4.5.4: Binary

Explanation

The binary number system is a number counting system that uses zeros and ones to represent values. Owing to the nature of computers, zeros and ones are suited to the way in which data is stored in a computer system and so binary is the best suited method of counting.

There are several types of binary numbers:

- Unsigned binary (positive numbers only).
- Signed binary (positive and negative numbers).
- Two's complement (positive and negative numbers).
- Floating point (positive and negative real numbers).

Exercise 31

Write the VB.Net code for a program that performs the following:

- Enter the binary number $1001\ 0110_2$ and convert it to decimal, and hexadecimal.
- Enter a positive decimal number and convert it to its equivalent in binary and hexadecimal.

Your answer

Line	VB.Net code
01.	
02.	
03.	
04.	
05.	
06.	
07.	
08.	
09.	
10.	
11.	
12.	

Line	VB.Net code
13.	
14.	
15.	
16.	
17.	
18.	
19.	
20.	
21.	
22.	
23.	
24.	
25.	
26.	
27.	
28.	
29.	
30.	
31.	
32.	
33.	
34.	
35.	
36.	
37.	
38.	
39.	
40.	

(5 marks)

Exercise 32

Extend your program so that it performs calculations for negative numbers.

Your answer

Line	VB.Net code
01.	
02.	
03.	
04.	
05.	
06.	
07.	
08.	
09.	
10.	
11.	
12.	
13.	
14.	
15.	
16.	
17.	
18.	
19.	
20.	
21.	
22.	
23.	
24.	
25.	
26.	
27.	
28.	
29.	

Line	VB.Net code
30.	
31.	
32.	
33.	
34.	
35.	
36.	
37.	
38.	
39.	
40.	
41.	
42.	
43.	
44.	
45.	
46.	
47.	
48.	
49.	
50.	
51.	
52.	
53.	
54.	
55.	

(5 marks)

Exercise 33

Change your program so that it performs subtraction, multiplication and division of positive and negative decimal, binary and hexadecimal numbers.

Your answer

Line	VB.Net code
01.	
02.	
03.	
04.	
05.	
06.	
07.	
08.	
09.	
10.	
11.	
12.	
13.	
14.	
15.	
16.	
17.	
18.	
19.	
20.	
21.	
22.	
23.	
24.	
25.	
26.	
27.	
28.	

Line	VB.Net code
29.	
30.	
31.	
32.	
33.	
34.	
35.	
36.	
37.	
38.	
39.	
40.	
41.	
42.	
43.	
44.	
45.	
46.	
47.	
48.	
49.	
50.	
51.	
52.	
53.	
54.	
55.	

(5 marks)

4.5.5: Information coding systems

Explanation

Character sets are used to display bytes in files. The most common character set is the American Standard Code for Information Interchange (ASCII) character set. The character set consists of a binary number followed by the equivalent character that the number represents.

In the example shown here:

- The binary number 65 represents the letter “A”.
- The binary number 74 represents the letter “J”.

So, the following bytes:

67 81 77 80 85 84 69 82

represents the text:

C O M P U T E R

A limitation with the ASCII code is that it has a limited number of characters that it can represent. To overcome this limitation, Unicode was introduced. Unicode allows different languages and pictograms to be included in a computer’s character set. Unicode represents numbers from the range 0_{16} to $10FFFF_{16}$, which allows for 1,114,112 characters or code points. Owing to the large number of characters possible, scripting languages such as Arabic are able to be represented using Unicode.

The transmission of bytes has to be checked for accuracy, especially as the bytes represent text that will be interpreted by human readers. To enable this to take place, several types of error checking methods have been created, e.g. parity checks, majority voting and check digits.

Exercise 34

- (a) Research how majority voting is used to perform error checking.
- (b) Write the VB.Net code for a program that performs error checking using majority voting.

Your answer

Line	VB.Net code
01.	
02.	
03.	

Line	VB.Net code
04.	
05.	
06.	
07.	
08.	
09.	
10.	
11.	
12.	
13.	
14.	
15.	
16.	
17.	
18.	
19.	
20.	
21.	
22.	
23.	
24.	
25.	
26.	
27.	
28.	
29.	
30.	

(5 marks)

(b) The International Standard Book Number (ISBN) is a system used to uniquely identify books and publications. The ISBN-10 number system allocates the first 9 characters following which a check digit is calculated.

The method of calculating the check digit is as follows:

- Multiply each digit with its position in the number starting from the right hand side.
- Sum the products.
- Take the sum and divide by 11.
- Take the remainder and perform a modulus-11. If the result is 10, use X as the check digit.

Example

A book is allocated the ISBN = 010242630

$$\begin{aligned} \text{Sum of products} &= (10 * 0) + (9 * 1) + (8 * 0) + (7 * 2) + (6 * 4) + \\ &\quad (5 * 2) + (4 * 6) + (3 * 3) + (2 * 0) \end{aligned}$$

$$\text{Sum of products} = 90$$

$$\frac{90}{11} = 8 \text{ remainder } 2$$

$$\text{Modulus} - 11 = 11 - 2 = 9$$

$$\therefore \text{Check digit} = 9$$

The ISBN number for the book is 0102426309.

Required

- Write the VB.Net code for a program that generates the check digit for a 9-digit ISBN number.
- Extend your program to accept a 10-digit ISBN number and determine if the number is valid.

Your answer

Line	VB.Net code
01.	
02.	
03.	
04.	
05.	
06.	
07.	
08.	
09.	
10.	
11.	
12.	
13.	
14.	
15.	
16.	
17.	
18.	
19.	
20.	
21.	
22.	
23.	
24.	
25.	
26.	
27.	
28.	
29.	
30.	
31.	

Line	VB.Net code
32.	
33.	
34.	
35.	
36.	
37.	
38.	
39.	
40.	
41.	
42.	
43.	
44.	
45.	
46.	
47.	
48.	
49.	
50.	
51.	
52.	
53.	
54.	
55.	
56.	
57.	
58.	
59.	
60.	

(10 marks)

4.5.6: Representing images, sound and other data

Explanation

Bitmaps are used to represent images. Learn these formulas:

$$\begin{aligned} \text{Resolution (pixels)} &= \text{Width} \times \text{height} \\ \text{Storage size} &= \text{Resolution} \times \text{Colour depth} \end{aligned}$$

The resolution refers to the level of detail that a bitmap image holds. The higher resolution, the more pixels are in the image and the more detail can be stored.

The storage sizes of files can be reduced using various data compression techniques. One method is known as Run Length Encoding (RLE). In this method, the physical size of a repeating string of bytes is reduced by representing it with two bytes. The first byte represents the number of characters and is called the run count. The second byte is the value of the character in the run, which is in the range of 0 to 255, and is called the run value.

Compression techniques such as RLE vary in the amount of compression that they can achieve.

Example

The 10-byte string **ADCCCCGGF** would be represented by the 10-byte string **1A1D5C2G1F**. The amount of compression is 0%.

The 20-byte string **ADCCCCGGFFFFFFFFF** would be represented by the 11-byte string **1A1D5C2G10F**. The amount of compression is 45%.

Owing to the repeated bytes in the second example, the level of compression is greater.

Exercise 35

An alternative to using the RLE algorithm for compression is to use dictionary-based methods. Write a program that implements a dictionary-method of compression.

Your answer

Line	VB.Net code
01.	
02.	
03.	
04.	

Line	VB.Net code
05.	
06.	
07.	
08.	
09.	
10.	
11.	
12.	
13.	
14.	
15.	
16.	
17.	
18.	
19.	
20.	
21.	
22.	
23.	
24.	
25.	
26.	
27.	
28.	
29.	
30.	
31.	
32.	
33.	
34.	
35.	
36.	
37.	

Line	VB.Net code
38.	
39.	
40.	
41.	
42.	
43.	
44.	
45.	
46.	
47.	
48.	
49.	
50.	
51.	
52.	
53.	
54.	
55.	
56.	
57.	
58.	
59.	
60.	
61.	
62.	
63.	
64.	
65.	

(10 marks)

4.9.1: Communication

Explanation

Communication between different devices is the key to the proliferation of computers in today's society. The communication takes place using networks of computers that are arranged in specific ways, e.g. a bus network, a star network or a ring network.

The communication itself can take place using parallel or serial transmission. In serial transmission, the bits in a byte are transferred one at a time. In parallel transmission, whole bytes or blocks are transferred.

When data is transferred, two methods can be used. In synchronous transmission, a clock is used to ensure that the bits are in the correct sequence. In asynchronous transmission, start and stop bits are added to the data.

Exercise 36

(a) Research the difference between synchronous and asynchronous data transmission.

Your answer:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

(5 marks)

(b) Write a program that calculates the bandwidth, latency and bit rate for a communication medium. Use your program to determine the relationship between bit rate and bandwidth.


Your answer

Line	VB.Net code
01.	
02.	
03.	
04.	
05.	
06.	
07.	
08.	
09.	
10.	
11.	
12.	
13.	
14.	
15.	
16.	
17.	
18.	
19.	
20.	
21.	
22.	
23.	
24.	
25.	
26.	
27.	
28.	
29.	

Line	VB.Net code
30.	
31.	
32.	
33.	
34.	
35.	
36.	
37.	
38.	
39.	
40.	
41.	
42.	
43.	
44.	
45.	
46.	
47.	
48.	
49.	
50.	
51.	
52.	
53.	
54.	
55.	
56.	
57.	
58.	
59.	
60.	
61.	
62.	

Line	VB.Net code
63.	
64.	
65.	
66.	
67.	
68.	
69.	
70.	
71.	
72.	
73.	
74.	
75.	
76.	
77.	
78.	
79.	
80.	
81.	
82.	
83.	
84.	
85.	
86.	
87.	
88.	
89.	
90.	

(10 marks)

 Teacher's Comments	Marks Awarded: <u>78</u>
<p>What you did well:</p> <p>What you need to work on:</p>	

Student's Comments
<p>What can I do to improve?</p>

Notes

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....



EXTENSION WORK

Extension work

To be skilled in programming, you need to practise actually writing code and using the Visual Studio IDE. The following are more challenging programming exercises and problems for you to attempt.

You should complete these exercises as follows:

- (a) Write pseudo-code algorithms for each of these exercises. You may need to make several attempts at producing the code that meets the requirements of the problem.
- (b) Check that the pseudo-code is correct in terms of syntax required by the exam board.
- (c) Convert the algorithm to VB.Net code and type it using a Visual Studio.
- (d) Check that the code is correct by debugging it using a dry run on paper. To do this, complete the dry run table with the names of the variables and the sequence of instructions.
- (e) Perform a check using the debugging tools in your IDE.
- (f) Ask your teacher to confirm that your code is correct by checking your dry run results.

Section summary

The section is divided into the following computational problems:

- Mobile Phone Expenses
- Buy-To-Let Register
- Weekend Planner
- Did I Sleep Well?
- Air Travel Planner

Task 1: Mobile Phone Expenses

Scenario

Mobile phones are charged using either a contract or a system of credits called Pay As You Go (PAYG). The key differences between these two methods are:

- When a user has a contract, they are allocated a number of usage units per month which automatically rolls-over into the following month. Any additional usage during the month beyond the allocated amount is added to the monthly bill. The monthly bill is usually paid automatically by Direct Debit.
- For a PAYG user, the allocation is initially purchased at the start of the monthly period. Any additional usage during the month outside the allocated amount has to be purchased when it occurs.

You are working as a freelance programmer for a telecoms company. Wesley is your direct line manager and has just purchased a new mobile phone. He would like to produce a model to determine if it is cheaper for him to use a contract or PAYG. Eventually, the model will be used as the back-end processing for an app that notifies customers of the best tariff available to them.

The following is Wesley's average usage for the last 12 months:

- Data usage = 750MB per month
- Text usage = 176 per month
- Voice usage = 89 minutes per month

Wesley has researched the deals available to him from various telecoms companies. Below is a table showing the monthly contract deals available.

Mobile Operator	Contract Name	Data Allowance	Text Allowance	Voice Allowance	Cost (£)
PurpleTelecom	Gold	2GB	300	150mins	18.99
	Silver	1GB	200	100mins	16.99
	Bronze	500MB	100	50mins	14.99
8T	Mega100	3GB	100	100mins	12.99
	Super75	1GB	75	75mins	9.99
YickettyYak	Con#1	4GB	100	50mins	19.99
	Con#2	2GB	100	50mins	15.99
	Con#3	1GB	50	25mins	9.99

Below is a table showing the monthly PAYG deals available.

Mobile Operator	PAYG Name	Data Allowance	Text Allowance	Voice Allowance	Cost (£)
PurpleTelecom	Mars	3GB	200	100mins	18.99
	Mercury	1GB	100	100mins	15.99
	Pluto	750MB	75	50mins	12.99
8T	Bundle1	2GB	75	60mins	11.99
	Bundle2	750MB	75	75mins	8.99
YickettyYak	Pay1	2GB	100	50mins	15.99
	Pay2	1GB	100	50mins	12.99
	Pay2	500MB	50	25mins	8.99

Wesley would also like to the model to have a method of keeping track of his monthly expenses and actual usage so he can determine if the anticipated expenditure suggested by the model is correct.

Requirements

Write a program to:

- (a) Enter the monthly contract and PAYG deals available and save these in a file.
- (b) Calculate the annual cost of each deal.
- (c) Display the results.
- (d) Provide Wesley with a method of recording his actual usage.
- (e) Perform a dry run of your code to check that it works.

Your answer

Line	VB.Net code listing
01.	
02.	
03.	
04.	
05.	
06.	
07.	
08.	
09.	
10.	
11.	
12.	
13.	
14.	
15.	
16.	
17.	
18.	
19.	
20.	
21.	
22.	
23.	
24.	
25.	
26.	
27.	
28.	
29.	
30.	
31.	

Line	VB.Net code listing
32.	
33.	
34.	
35.	
36.	
37.	
38.	
39.	
40.	
41.	
42.	
43.	
44.	
45.	
46.	
47.	
48.	
49.	
50.	
51.	
52.	
53.	
54.	
55.	
56.	
57.	
58.	
59.	
60.	

(10 marks)

Dry-run trace table

Line	List variable & constant names from code									
	Write values for each named variable at each line as code executes									

Line	List variable & constant names from code									
Write values for each named variable at each line as code executes										

(10 marks)

Task 2: Buy-To-Let Register

Scenario

A Buy-To-Let loan is the name given to a commercial loan. The loan is given to the buyer of a domestic property who wants to rent out the property. The purchaser must supply a minimum of 25% of the value of the property as a deposit and the loan lasts for a maximum of 25 years. Interest rates will vary during the lifetime of the loan.

Sandy has purchased a number of properties in the North West on a Buy-To-Let scheme. She hopes that she can rent out the properties and obtain an income in future years once the commercial loan has been fully paid.

Alan is Sandy's accountant. He uses the following table to calculate the profitability of each property:

Property details	Purchase Price (£)	Deposit (£)	Monthly Rental Income (£)	Monthly Loan payment (£)	Maximum Monthly profit (£)
12 The Spires	65,000	16,250	500	325	175
276 Snow Road	80,000	21,000	480	395	85
2 Meadow Court	82,000	21,000	525	401	124
15 Salt Ln	75,000	21,000	505	345	160
Total					544

Alan also advises Sandy the following:

- Build-up reserves to set aside 2 months payments for each property in case of non-occupancy.
- Build-up reserves to set aside £1,000 for each property for repairs.

Requirements

Write a program to allow Sandy to keep a register of her properties. The program should:

- (a) Enter the details of Sandy's properties.
- (b) Calculate the annual profitability of the portfolio with reserves taken into account.
- (c) The 2 Meadow Court property is a student house. Display the annual profitability assuming the property is only rented for 9 months of each year.
- (d) Perform a dry run of your code to check that it works with the data provided.

Your answer

Line	VB.Net code listing
01.	
02.	
03.	
04.	
05.	
06.	
07.	
08.	
09.	
10.	
11.	
12.	
13.	
14.	
15.	
16.	
17.	
18.	
19.	
20.	

Line	VB.Net code listing
21.	
22.	
23.	
24.	
25.	
26.	
27.	
28.	
29.	
30.	
31.	
32.	
33.	
34.	
35.	
36.	
37.	
38.	
39.	
40.	
41.	
42.	
43.	
44.	
45.	
46.	
47.	
48.	
49.	
50.	
51.	
52.	
53.	

Line	VB.Net code listing
54.	
55.	
56.	
57.	
58.	
59.	
60.	
61.	
62.	
63.	
64.	
65.	
66.	
67.	
68.	
69.	
70.	
71.	
72.	
73.	
74.	
75.	
76.	
77.	
78.	
79.	
80.	

(10 marks)

Dry-run trace table

Line	List variable & constant names from code									
	Write values for each named variable at each line as code executes									

Line	List variable & constant names from code									
	Write values for each named variable at each line as code executes									

(10 marks)

Task 3: Weekend Planner

Scenario

A housemaster in a boarding school plans the weekends for boarders so that they have a full programme.

The activities that are required to be completed are:

- Homework
- Games
- Shopping
- Room cleaning
- Laundry
- Ironing
- Relaxation time

The housemaster requires a program to keep a record of which activities have been performed for each weekend.

Requirements

Write VB.Net code to:

- (a) Enter the activities planned for a specific weekend in a file.
- (b) Record when these activities are completed and the name of the person who completed them.
- (c) Display the results showing the activities.
- (d) Perform a dry run of your code.

Your answer

Line	VB.Net code listing
01.	
02.	
03.	
04.	
05.	
06.	
07.	
08.	
09.	
10.	
11.	
12.	
13.	
14.	
15.	
16.	
17.	
18.	
19.	
20.	
21.	
22.	
23.	
24.	
25.	
26.	
27.	
28.	
29.	
30.	

(10 marks)

Dry-run trace table

Line	List variable & constant names from code									
	Write values for each named variable at each line as code executes									

Line	List variable & constant names from code									
	Write values for each named variable at each line as code executes									

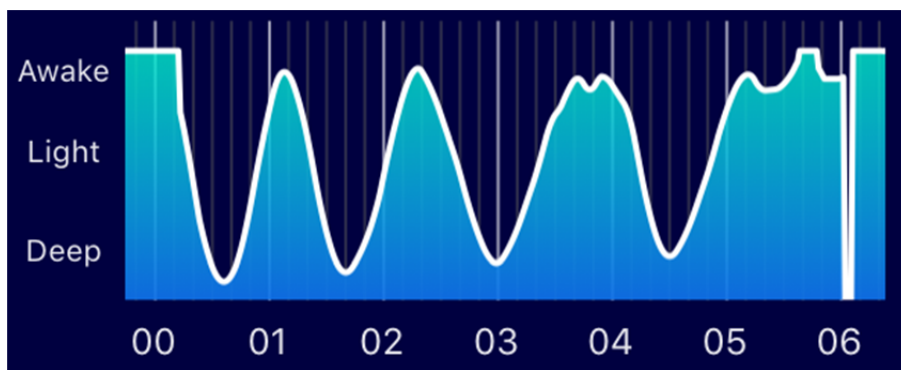
(10 marks)

Task 4: Did I Sleep Well?

Scenario

Most medical authorities recommend at least 8 hours sleep for adults and more for people under 18 years old. This is needed to build up the immune, nervous, skeletal, and muscular systems.

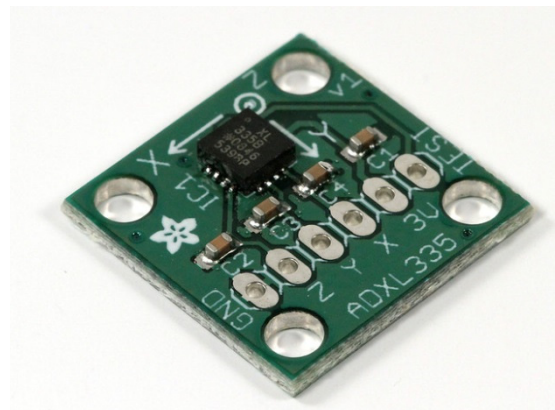
Sleep in humans is rhythmic. The body alternates between periods of Rapid Eye Movement (REM) sleep and non-REM sleep. In general, REM sleep occurs at 2.5hrs, 5hrs, 7.5hrs and 9hrs in sleep. The following graph shows a typical sleep cycle of an adult:



A programmer wishes to develop an app that allows users to enter data about their sleep and store this data. The program should analyse the data and provide recommendations about:

- The amount of REM sleep obtained.
- The sleep efficiency (defined as the amount of time asleep / amount of time awake).

The data is logged by an accelerometer (shown here) placed on the mattress of the person sleeping. This logs the amount of time the person has been asleep and when there has been no movement.



The following additional analysis should be performed by the program using these formulas:

Assume:

$$\begin{aligned} \text{Bed time} &= A \\ \text{Sleep onset time} &= B \\ \text{Waking up time} &= C \end{aligned}$$

Calculations are:

$$\begin{aligned} \text{Time in Bed} &= D = C - A \\ \text{Sleep Duration} &= E = C - B \\ \text{Awake Duration} &= D - E \\ \text{Sleep Efficiency} &= \frac{E}{D} * 100\% \end{aligned}$$

The accelerometer records the Sleep onset time and displays this to the user. The user manually records the Bed time and the Waking up time.

Requirements

Write a program to achieve the following objectives:

- (a) Allow the sleeper to enter the Sleep onset time, Bed time and Waking up time. Store this data for each night slept in a file.
- (b) Calculate, display and store the following for the night slept:
 - a. Time in Bed
 - b. Sleep Duration
 - c. Awake Duration
 - d. Sleep Efficiency
- (c) Display the sleep efficiency for a number of days slept.
- (d) Perform a dry run of your code with the following data:

Date	Bed time	Sleep onset time	Waking up time
12 March	10:07pm	10:32pm	6:30am
13 March	11:08pm	11:45pm	7:00am
14 March	10:00pm	10:32pm	6:00am
15 March	09:30pm	09:54pm	6:00am

Your answer

Line	VB.Net code listing
01.	
02.	
03.	
04.	
05.	
06.	
07.	
08.	
09.	
10.	
11.	
12.	
13.	
14.	
15.	
16.	
17.	
18.	
19.	
20.	
21.	
22.	
23.	
24.	
25.	
26.	
27.	
28.	
29.	
30.	
31.	

Line	VB.Net code listing
32.	
33.	
34.	
35.	
36.	
37.	
38.	
39.	
40.	
41.	
42.	
43.	
44.	
45.	
46.	
47.	
48.	
49.	
50.	
51.	
52.	
53.	
54.	
55.	
56.	
57.	
58.	
59.	
60.	

(10 marks)

Dry-run trace table

Line	List variable & constant names from code									
	Write values for each named variable at each line as code executes									

Line	List variable & constant names from code									
Write values for each named variable at each line as code executes										

(10 marks)

Task 5: Air Travel Planner

Scenario

AB Air Tours is a travel company that wants to produce a program to be used by its representatives in the cities where they operate. The aim of the program is to display the arrival and departure of airline flights so that the reps are able to meet the holiday makers as they arrive at the airport.

The travel company uses the following hotels in the US:

- New York: New York Hilton
- Boston: Mandarin Oriental
- Chicago: Mark Twain Hotel
- Los Angeles: Millennium Biltmore Hotel
- San Francisco: Phoenix Hotel

The program should display the following:

- Flight No
- Operator
- Destination
- Arrival time
- Passenger name and hotel

Requirements

1. Research the details of airline flights to the following US destinations from airports in the UK:

- New York
- Boston
- Chicago
- Los Angeles
- San Francisco

2. Write a VB.Net program to achieve the following objectives:

- (a) Input the details of the flights, the passengers and the hotel they are staying at in each city.
- (b) Store the data in a file.
- (c) Display the passenger name, flight no, operator, and arrival time according to the hotel.
- (d) Perform a dry run of your code with your own sample data.

Your answer

Line	VB.Net code listing
01.	
02.	
03.	
04.	
05.	
06.	
07.	
08.	
09.	
10.	
11.	
12.	
13.	
14.	
15.	
16.	
17.	
18.	
19.	
20.	
21.	
22.	
23.	
24.	
25.	
26.	
27.	
28.	
29.	
30.	
31.	

Line	VB.Net code listing
32.	
33.	
34.	
35.	
36.	
37.	
38.	
39.	
40.	
41.	
42.	
43.	
44.	
45.	
46.	
47.	
48.	
49.	
50.	
51.	
52.	
53.	
54.	
55.	
56.	
57.	
58.	
59.	
60.	


(10 marks)

Dry-run trace table

Line	List variable & constant names from code									
	Write values for each named variable at each line as code executes									

Line	List variable & constant names from code									
	Write values for each named variable at each line as code executes									

(10 marks)

 Teacher's Comments	Marks Awarded: <u> </u> 100
<p>What you did well:</p> <p>What you need to work on:</p>	

Student's Comments
<p>What can I do to improve?</p>

Notes

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

LMC Education Ltd

© Copyright LMC Education Ltd 2015