

Outline scheme of learning

Edexcel GCE in Applied ICT A2 - Unit 12: Customising Applications

Topic	Learning activities/outcomes	Learning Objectives &/or Key question	Resources	Homework and Assignments	Syllabus extract(s)
1.	Induction: unit specification <ul style="list-style-type: none"> • Discuss unit requirements (e.g. assessment evidence) • Discuss client requirements • Discuss where automation is useful • Discuss the scenario – is it realistic and worth automating? • Briefly discuss the development stages 	Students will be able to: <ul style="list-style-type: none"> • Understand the role of client requirements and the need for detailed clarification • Begin to appreciate the need for development stages • Understand the scenario 	Chapter 12.1 Unit 12 Specification	Exercise on mileage claim request (see Teacher's Notes)	Unit 12 <ul style="list-style-type: none"> • Introduction • Standard ways of working • Assessment Evidence • Assessment Criteria 12.2 The need to code
2.	Event-driven programming <ul style="list-style-type: none"> • Create code using the Macro Recorder: Activity 1 • Get students to check and mark each others code • Discuss importance of naming conventions • Investigate error messages (additional activity: run macro from chart and discuss error message options, e.g., End, debug, help) • Investigate Objects, Methods, Properties, etc.: Activity 2 (additional exercise: see Teacher's Notes) • Modify code: Activity 3 • Introduce Literals (see Teacher's Notes) • Introduce commenting (additional exercise: see Teacher's Notes) • Use the Object Browser: Activity 4 • List events: Activity 5 • Use an event to run code: Activity 6 	Students will be able to: <ul style="list-style-type: none"> • Use the macro recorder to generate small amounts of code • Understand the purpose of naming conventions • Understand error message options • Understand the meaning of concepts such as object, methods, properties • Begin to modify code • Use the object browser • Understand the meaning and purpose of events • Use an event to run code 	Chapter 12.2 Spreadsheet software package with event-driven programming language enabled Projector to demonstrate macro recorder and code.	Activities 1 to 6 Identify objects, methods and properties in Word and PowerPoint (homework?)	12.3 Objects, control properties and events

Topic	Learning activities/outcomes	Learning Objectives &/or Key question	Resources	Homework and Assignments	Syllabus extract(s)
3	<p>Program structures and skills</p> <ul style="list-style-type: none"> • Brief explanation of sequence • Create code to demonstrate selection: Activity 1. Also compare VBA IF with the worksheet function IF • Introduce the importance of testing: Activity 1 • Create code to demonstrate logical and relational operators: Activity 2. Also, compare these with use in IF worksheet function. • Create code to demonstrate select case and nested IFs: Activity 3 and see nested if example in Teacher's Notes. • Discuss module sheets – what are they? • Create code to demonstrate repetition and conditional repetition : Activities 4 and 5 • Investigate arithmetic operators using worksheet formulae • Discuss subroutines and calls • Introduce variables, constants and Option Explicit • Modify code in ActiveBook data file to demonstrate variable declaration and role of Option Explicit: Activity 6 • Modify code in ActiveBook data file to demonstrate use of constants: Activity 7 • Modify code in ActiveBook data file to demonstrate global and local variables and scope: Activity 8 • Create code using correct data types and naming conventions: Activity 9 • Use Arrays: Activity 10 • Create code that uses parameter passing: Activity 11 and investigate further: Activity 12 • Create code to validate input values: Activities 13 and 14 (solutions in Teacher's Notes) • Create code that uses error trapping: Activity 15 	<p>Students will be able to:</p> <ul style="list-style-type: none"> • Use correct program structures • Begin testing code • Use correct operators • Understand the need for robust programming: e.g. declarations, strong data typing, commenting, naming, modular programming, error trapping • Understand the difference between global and local declaration. 	<p>Chapter 12.3</p> <ul style="list-style-type: none"> • ActiveBook data files <p>Spreadsheet software package with event-driven programming language enabled</p>	<p>Activities 1 to 15</p> <p>Additional exercise in Activity 3 on background colour (see the ActiveBook version). A solution is in the Teacher's Notes.</p> <p>Investigate arithmetic operators using worksheet formulae</p> <p>Activity 14 exercises on validation (solutions are in the Teacher's Notes)</p>	<p>12.5 Programming structures</p> <p>12.8 Programming and the spreadsheet</p> <p>12.9 Testing</p> <p>12.12 Programming skills</p>
4	<p>Specify, design and document</p> <ul style="list-style-type: none"> • Review the software life cycle • Produce a functional specification for the scenario: Activities 1 & 2 (use additional exercise specified in Teacher's Notes) • Discuss the following aspects of design: evolutionary delivery; and top-down design • Create a structure chart: Activity 3 • Create a skeleton program solution: Activity 4 • Read document in ActiveBook on testing and discuss • Create test data table: Activity 5 • Read documents in ActiveBook on documentation and discuss • Create draft technical and user guides for the scenario (see Activity 1 next chapter) 	<p>Students will be able to:</p> <ul style="list-style-type: none"> • Understand the processes involved in producing a functional specification • Produce functional specification checklists • Design a solution to a project • Test a solution • Understand the processes involved in producing documentation 	<p>Chapter 12.4</p> <ul style="list-style-type: none"> • ActiveBook data files <p>Spreadsheet software package with event-driven programming language enabled</p> <p>Word processing package</p> <p>Organisation chart application (e.g. Word, Visio)</p>	<p>Activities 1 to 5</p> <p>Additional exercise on functional specification (see Teacher's Notes)</p> <p>Read document in ActiveBook on testing and documentation</p> <p>Create draft technical and user guides for the scenario (or leave to Activity 1 next chapter)</p>	<p>12.1 Functional Specification</p> <p>12.4 Designing Routines</p> <p>12.9 Testing</p> <p>12.10 Program Documentation</p>

Topic	Learning activities/outcomes	Learning Objectives &/or Key question	Resources	Homework and Assignments	Syllabus extract(s)
5	<p>Advanced Programming Techniques</p> <p><i>Spreadsheet solutions</i></p> <ul style="list-style-type: none"> Investigate using named ranges in code: ActiveBook data file Investigate using the Control toolbox (ActiveBook data file) and complete the exercises specified in this file (ActiveBook data file) Investigate searching a folder for workbooks Complete the code for the scenario (see Teacher's Notes for solution): Activity 1 Complete the documentation, etc., for the scenario: Activity 1 Investigate code to identify data duplication in a spreadsheet (see data file in Teacher's Notes) <p><i>Database solutions</i></p> <ul style="list-style-type: none"> Investigate modules and macros (see Teacher's Notes for additional exercises using DoCmd) Explain problems of opening database files from a CD (see Teacher's Notes which includes additional exercise) Investigate database form objects: Activity 2 Investigate database recordsets: Activity 3 Investigate duplicate control: Activity 3 Create database form object: Activity 3 (use alternative exercise in Teacher's Notes if Access 2003 not available) Link two applications via code: Activity 4 Note: solution to importing data into Access via VBA is on the Teachers CD Note: code on displaying reports in Access is on the Teacher's CD 	<p>Students will be able to:</p> <ul style="list-style-type: none"> Complete the solution to the scenario Customise a database application Customise a spreadsheet application 	<p>Chapter 12.5</p> <ul style="list-style-type: none"> ActiveBook data files <p>Spreadsheet software package with event-driven programming language enabled</p> <p>Database software package with event-driven programming language enabled</p> <p>Teachers CD: solution for importing data from Excel into Access and displaying reports in Access</p>	<p>Active workbook data files</p> <p>Activities 1 to 4</p> <p>Database exercises using DoCmd (Teacher's Notes)</p> <p>Database exercise to display database reports (Teacher's Notes)</p>	<p>12.8 Programming and the spreadsheet</p> <p>12.7 Programming and the database</p> <p>12.3 Objects, control properties and events</p> <p>12.2 The need to code (facilitate data sharing between applications)</p>
6	<p>Tackling the Unit 12 Assessment</p> <ul style="list-style-type: none"> Discuss checklist for eportfolio Discuss considerations when project has been chosen Re-read the assessment evidence Recap on documentation Recap on version control for prototypes and need to keep as evidence Discuss what makes a good or poor user interface, e.g., when designing forms, reports (see Teacher's Notes for code) navigation methods (HCI issues). Get students to provide examples of these in the non-computing world. 	<p>Students will be able to:</p> <ul style="list-style-type: none"> Begin the unit 12 assessment 	<p>Chapter 12.6</p>	<p>Investigate what makes a good or poor user interface</p>	<p>Unit spec Assessment evidence and criteria</p> <p>12.6 Human Computer Interface</p>