| N Scheme of Work 2020-21<br>Subject: Computing Unit 1 Principles of Computer Science | | | | | | |
|---|---|---|---|---|---|---|
| **Year Group: Year 12**<br>**Specification: BTEC Computing – Unit 1 – this is to run co-currently as Unit 2 (students have 5 hours a week 2 hours for Unit 1 and 2 hours for Unit 2 and 1 hour self-directed learning)** | | | | | | |
| **Lesson No** | **Topic & Objectives** | **Big Question – What will students learn?** | **Key Activities & Specialist Terminology (Do Now Task / Starter/Tasks/Plenary** | **Planned Assessment** | **Homework or flipped learning resources**<br><br>**DODDLE resources** | **Lit Num SMSC Codes** |
| 1 & 2 | **A1 Decomposition** | **What is decomposition?** | • First lesson a baseline test should be completed.<br>• **Lead-in:** Introduce the concept of computational thinking (CT) and stages of CT. Explain that CT will be applied throughout this and other units.<br>• **Small group/paired activity:** Give learners a scenario in which they are asked to reverse-engineer a 'clone' of a simple computer game (eg Space Invaders or Tetris®). Learners must identify the distinct steps of the problem.<br>**Link for resources:**<br>There are different scenarios for you to pick for the students to do in the following link<br>https://teachinglondoncomputing.org/resources/inspiring-unplugged-classroom-activities/<br>• **Tutor-led discussion:** Learners feed their thoughts back to the whole group.<br>• **Plenary:** Using what they have learned from the discussion, learners complete work on the scenario activity. | Baseline test | **Expansion – independent learning activity:** Students to use the following link to look for articles that illustrate decomposition – then they are to write a small paragraph detailing the main points https://teachinglondoncomputing.org/resources/developing-computational-thinking/decomposition/<br>Exercises on problem solving (CT) from PG Online resources | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 3 & 4 | **A2 Pattern recognition** | **What is the pattern? Read and understanding code** | • **Lead-in:** Recap the concept of decomposition with a Q&A session. Explain that in this lesson they will continue to explore CT.<br>• **Small group/paired activity**: In the same groups as the previous lesson, learners expand their analysis of the example game (eg identify patterns and common elements or features). Introduce some programming terms and ask learners to identify variables in the example game (eg 'lives').<br>• **Tutor-led discussion:** Ask learners to contribute their thoughts to the discussion.<br>• **Plenary:** Using what they have learned from the discussion, learners complete work on the scenario activity in the following link https://teachinglondoncomputing.org/resources/inspiring-unplugged-classroom-activities/<br>**(Teacher's note:** the above link contains lots of different activities for you to choose the ones suitable for your learners) | Exam questions on Problem Solving (CT) – this to be complete on their own – use peer marking<br>(pg 39 PG Online Book) | Teachers to pick an activity from the following link that gets students to identify elements of code https://teachinglondoncomputing.org/resources/inspiring-unplugged-classroom-activities/<br><br>Computational Thinking - Problem Solving Worksheet 1 + Homework sheet 1 | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |
| 5 & 6 | **A3 Pattern generalisation and abstraction** | **How do you identify the variables?** | • **Small group/paired activity:** In the same groups as the previous lesson, learners expand their analysis of the example game (eg identify patterns and common elements or features). Introduce some programming terms and ask learners to identify variables in the example game (eg 'lives').<br>• **Tutor-led discussion:** Ask learners to contribute their thoughts to the discussion. | Exam questions – PG online – Constants and Variables (pg6-7) | Activities from BTEC revision guide pg 1-6<br><br>Programming – Worksheet and Homework sheet 1 | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | • **Plenary:** Using what they have learned from the discussion, learners complete work on the scenario activity. | | | |
| 7 & 8 | **A4 Algorithm design** | **How to write an algorithm – flowchart introduction.** | • **Lead-in:** Recap the concepts of 'decomposition', 'pattern recognition' and 'pattern generalisation and abstraction' through Q&A session. Explain that over the next two lessons learners will explore the first stages of algorithm design.<br>• **Small group or paired activity:** Using the same groups as previous lessons, learners consider the 'algorithms' for different parts of the game (eg controlling the ship, firing or invader movement). Highlight that in the early design stages of a software solution the 'algorithms' only need to be step-by-step instructions, using everyday language, and do not need to be presented in pseudocode or full programming code.<br>• **Group activity (peer review and feedback):** At different stages during the two lessons, ask groups to share their work with other small groups. Each group should give feedback about the algorithms produced. Feedback should focus on completeness and accuracy.<br>• **Plenaries:** Give groups different-coloured pens to help them annotate work when giving feedback, and so they can annotate changes they have made as a result of feedback.<br>• **Extension – independent learning activity:** If time and resources allow, give learners a chance to implement some of their algorithms into a working | Deep Assessment in Purple books on topics learnt over the last couple of weeks – with exam style question (taken from Sample Papers – BTEC online) | Exercise on Algorithm Design BTEC Revision Guide Pg 8<br><br>Structured Programming – Worksheet and Homework sheet 2 | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | system, eg the 'invader movement' algorithm using Scratch. Learners could use pre-made sprites and apply their algorithm. They should check their algorithm by running the 'code' to see if it performs as they would expect. | | | |
| 9 & 10 | **B1 Structured English (pseudocode)**<br>• Interpreting pseudocode.<br>• Producing pseudocode.<br>**B2 Flowcharts using standard symbols** | **How do you read and write pseudocode?** | • **Tutor presentation:** Introduce the concept of pseudocode when designing algorithms and planning computer programs. Explain why pseudocode is used as a planning and design tool (ie to identify and plan the correct sequence of tasks/processes for a computer program). Introduce the 'basic operations' listed in the specification. Explain that operations, key commands and common programming functions are typically written in upper case within the pseudocode. See the following video link: Pseudocode Tutorial (6.5 minutes) www.youtube.com/watch?v=rSz7549WSjY<br>• **Independent learning activity:** Give learners simple scenarios and flow charts or components of a computer program that they must present as pseudocode. Focus on the basic operations and avoid, if possible, decisions and repetition. However, depending on learners' ability and their programming experience, you may decide to introduce concepts from later lessons.<br>*Note: try to introduce a different scenario from previous lessons. It is important that learners can apply knowledge and demonstrate understanding in a range of contexts.* | Clearly written pseudocode that contains a clear result | BTEC Revision Guide pg 9<br>Pg 12 Exercises in PG Online Book | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | *They should be able to work from scenarios and/or flow charts.*<br>Sequence-based scenarios could include:<br>inputting a number, squaring it and outputting the results<br>temperature conversion between Fahrenheit and Celsius<br>calculating the area of a circle.<br>• **Plenary:** Ask learners to discuss/compare their solutions in small groups. Ask one group to share their solutions. As a class, discuss the merits of each solution (try to identify a group that has two varying but still valid solutions). Highlight that, depending on the scenarios, there may be different ways of presenting solutions which are still valid, just as there are when creating a programmed solution. | | | |
| 11 & 12 | **B1 Structured English (pseudocode)**<br>• Interpreting pseudocode.<br>• Producing pseudocode.<br>**B2 Flowcharts using standard symbols** | **How to read and write pseudocode to solve problems in previous coded work.** | • **Lead-in:** Recap learners' understanding by asking them to produce an algorithm for a simple scenario using 'basic operations'.<br>• **Tutor presentation:** Introduce the 'decisions' listed in the specification. Reiterate that operations, key commands and common programming functions are typically written in upper case within the pseudocode. Use Q&A with some example logical problems to establish learners' understanding. Explain that indentation can be used to show hierarchy of tasks.<br>• **Independent learning activity:** Give learners some simple scenarios, and where appropriate supporting flow charts, or components of a computer program that require the use of | Deep Assessment on the computer of what code they have learnt over the last couple of weeks. Students to screen shot their code and print for marking and grading – Exam questions from Unit 1 Papers used to assess students fully. | Exercises Pg 42-50 PG online book | Lit<br>Social<br>So8<br>C3<br>Sp2<br>Sp5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | 'decisions'. They should write pseudocode to represent the solutions.<br>Selection-based scenarios could include:<br>• cinema tickets (Adult, Child, Senior and Student rates)<br>• calculating the cost of an invoice, with free delivery for orders of £100.<br>• **Tutor-led class discussion:** Go through the solutions as a class. Select examples that highlight correct solutions and also some that highlight common misconceptions.<br>• **Plenary:** Give learners time to correct or develop their pseudocode algorithms. Or ask them to annotate their pseudocode to explain its function. | | | |
| 13 & 14 | **B1 Structured English (pseudocode)**<br>• Interpreting pseudocode.<br>• Developing pseudocode.<br>Producing pseudocode. | **What is nested logic and repetition?** | • **Lead-in:** Recap learners' understanding by asking them to produce an algorithm for a simple scenario using 'decisions'.<br>• **Tutor presentation:** Introduce the concepts of 'nested logic' and repetition. Explain that in computer programs 'decisions' are often not stand-alone and conditions are regularly chained or combined with repetition to reach an outcome.<br>• **Independent learning activity:** Give learners a series of varying scenarios and/or problems that require 'nested logic' and/or repetition. They should write pseudocode to represent the solutions.<br>*Note: you could give a wider range of possible scenarios by introducing some* | Written pseudocode that clearly represents a solution | Exercises on Pg 10 – 11 BTEC Revision Guide | Lit<br>Social<br>So8<br>C3<br>Sp2<br>Sp5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | *of the arithmetic operations listed in topic C2.*<br><br>Iteration-based scenarios could include the following.<br>• Guessing game - the computer randomly generates a number between 1 and 30 and the player has to guess the number. For each guess the computer responds with higher or lower and the player has a limited number of guesses.<br>• Palindrome problem - inputting a word, reversing it, comparing the results and declaring whether or not it was a palindrome.<br>• **Plenary – paired activity:** Ask learners to compare and contrast their solutions. Support them to identify any errors in the logic and allow time for them to correct their solutions. | | | |
| **October Half Term** | | | | | | |
| 15 & 16 | **B1 Structured English (pseudocode)**<br>• Interpreting pseudocode.<br>• Producing pseudocode.<br>**B2 Flowcharts using standard symbols** | **What improvements can be made to the written pseudocode?** | • **Lead-in:** Organise learners so they can complete assessment tasks independently.<br>**Individual activity:** Give learners a series of scenarios and flow charts. They should produce pseudocode that demonstrates their learning over the last few lessons. The activities should increase in demand as the assessment progresses. Give learners at least one scenario where they have to evaluate pseudocode and suggest improvements or write an improved version. *NB: try to structure and present activities in a similar way to the sample assessment material supporting the unit.*<br><br>Problems could include: | Written pseudocode with improvements | Writing and interpreting algorithms worksheet 3. | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | • password problem (maximum three attempts)<br>• setting up a password (must be a certain length and contain an upper case letter, a number and a symbol)<br>• weights and measures convertor.<br>Discuss the command verbs and how learners should structure their answers when presented with them. Definitions are in the specification. | | | |
| 17 & 18 | **C1 Handling data within a program**<br>Defining and declaring constants and variables. | **Identifying what is the best data type to use in different cases.** | • **Lead-in:** Explain that over the next few lessons learners will explore the use of data and variables in computer programs.<br>• **Tutor presentation:** Give learners a list of the data types in the specification. Identify the properties/characteristics of each data type and explain the difference between the primitive data types and the composite data types.<br>• **Paired activity:** Give learners a series of examples of pseudocode containing a range of variables and example data. In pairs, learners should discuss the examples and identify which data types they would use in each case.<br>• Float examples - currency problems such as conversion.<br>• String examples - counting the occurrence of a letter in a string or reversing a string.<br>• Integer examples - problems using counters (for example displaying a bar chart).<br>• **Class discussion:** Discuss the decisions made during the activity. | Deep Assessment in Purple books on topics learnt over the last couple of weeks – with exam style question - Question 1a & b from Exam paper June 2017 | Pg 12 & 13 BTEC Revision book | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | • **Individual activity:** Give learners an additional example scenario and specification for a computer program that would use different data types (eg a program that could handle personal information for employees of a company). Try to give learners data that could possibly be stored using different data types and ask them to justify their choices. | | | |
| 19 & 20 | **C1 Handling data within a program**<br>Managing variables. | **How computers handle data and understand what is the role of local and global variables?** | • **Lead-in:** Explain that learners will extend their work on how computers handle data and look at the role of local and global variables.<br>• **Tutor presentation:** Introduce the concept of local and global variables. Explain how using them may affect the way a program works and the data within the program. Learners should think of global variables as being used by a number of different functions/subroutines (passing values will be explored later). See video link: Scope of variables (7.5 minutes) in the following link www.youtube.com/watch?v=RlQQK-mYmZs<br>• **Individual activity:** Give learners pseudocode examples containing local and global variables that are used in different ways. Ask them to show what happens when different scenarios and data are passed in to the code.<br>Validation routines in particular as the original copy of the data should be held back and only a copy worked on inside a function.<br>• **Plenary:** Discussion regarding the validation used, discuss how relevant they were or not. | Give learners a written exam-style question to help summarise their learning. – Use Sample Learner paper | Pg 13 & 14 BTEC Revision book | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 21 & 22 | **C1 Handling data within a program**<br>Managing variables. | **How to name conventions?** | • **Lead-in:** Explain that learners will extend their work on local and global variables and also look at naming conventions.<br>• **Tutor presentation:** Recap the last lesson and explain how naming conventions might work. Then present some examples using simple programming code (Python may be a good starting point). Give examples (in pseudocode and Python) that show inconsistent/confusing approaches to variable names. Explain how in pseudocode they can still be understood but when executed they may cause different outcomes.<br>• **Paired activity:** Give examples of pseudocode that require debugging to each pair of learners. Ask learners to identify problems within the code (either limited to naming conventions or extended to global and local variables). Learners should rewrite parts of the given code to correct problems and improve efficiency.<br>• **Plenary:** Re-organise learners so that each pair joins another group. Ask each group to compare and contrast their solutions. | Re-written code that corrects the errors | Pg 16 & 17 BTEC Revision book | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |
| 23 & 24 | **C1 Handling data within a program**<br>Managing variables. | **Enforcing the concepts learnt in the previous sessions.** | • **Lead-in:** Reinforce the concepts explored over the last two lessons and explain that learners will continue to develop this work.<br>• **Individual activity:** Give learners examples of pseudocode with a number of different problems and contexts. Ask them to complete tasks including explaining the functions, describing how data is processed by the code, debugging errors and re- | Clear explanations of the functions used, in the programming and description of how the data is processed by the code.<br><br>AP testing – mock exam – Jan 2018 – Paper 1 & 2 | Pg 18 & 19 BTEC Revision book | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | writing sections to improve efficiency. At this stage learners can still be working in pseudocode and do not need to create functioning programs. | | | |
| 25 & 26 | **C2 Arithmetic operations**<br>**C3 Built-in functions**<br>Arithmetic functions. | **What are high-level programming languages?** | • **Lead-in:** Explain that learners will look at common built-in functions that are available in many high-level programming languages.<br>• **Tutor presentation:** Introduce the concepts of arithmetic functions. Explain the purpose and use of the functions and how they may be used to add functionality to a program. At this stage, just discuss and use these functions at pseudocode level. Learners will explore the syntax and use of these within specific languages later. However, you may wish to give examples of the functions working within a program, without going into the code, in order to give context.<br>• **Individual learning activity:** Give learners a series of problems. Use pseudocode that contains the arithmetic operations from the specification. Ensure problems include a range of contexts in which learners must apply understanding.<br>In particular these problems should include the need to process using BODMAS - they should be multi-stage calculations including nested brackets.<br>• **Small group activity:** Organise learners into groups to improve their solutions. | Successful pseudocode code that makes sense and contains arithmetic.<br>Solving problems that contain BODMAS –<br>Learners must show clear understanding of al the contexts used. | Pg 20 & 21 BTEC Revision book | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 27 & 28 | **C2 Arithmetic operations**<br>**C3 Built-in functions**<br>• Arithmetic functions.<br>• String handling functions.<br>• General functions. | **What functions are available in high-level programming languages?** | • **Lead-in:** Explain that learners will continue to look at common built-in functions available in many high-level programming languages.<br>• **Tutor presentation:** Introduce the string handling functions. Explain their purpose and use, and how they may be used to add functionality to a program. Explain how general functions are used as a matter of course when developing solutions. Again, at this stage these functions only need to be discussed and used at pseudocode level. Learners will explore the syntax and use of these within specific languages later.<br>• **Individual learning activity:** Give learners a series of problems. Use pseudocode that contains the string handling functions from the specification. Ensure problems include a wide range of contexts in which learners must apply understanding. Allow time to discuss solutions with individuals or as small groups.<br>Problems could include: A palindrome phrase checker, for example 'A car, a man, a maraca.' – see link in the following link www.palindromelist.net/A-car-a-man-a-maraca/ The spaces and punctuation in this phrase need to be stripped and the spaces closed up before reversing and comparing the strings. | Give learners a problem that requires the development of pseudocode. Try to give them a problem that requires application of arithmetic functions and string handling functions. The task should be completed in exam conditions and marked by the tutor to gauge progress. Adjust the support given depending on the learners' progress. | Research what 'Hungarian Notation' means in the context of naming conventions. What are the advantages and disadvantages of using this approach to naming? | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |
| 29 & 30 | **C2 Arithmetic operations**<br>**C4 Validating data** | **Understanding variables in a database.** | • **Lead-in:** Start with a practical activity. Give learners a spreadsheet or simple flat file database containing a series of different fields. Ensure some fields have validation and others do not. | Deep Assessment in Purple books on topics learnt over the last couple of weeks – with exam style | Validation exercises pg 160 – 162 BTEC Revision | Lit<br><br>Social<br><br>So8<br>C3 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | • Validation check techniques.<br>• Post-check actions. | | Only some validation should have appropriate error messages. Learners should enter data from the test plan or list, and record what happens.<br>• **Class discussion:** Discuss what happened when the data was entered. Explore why only some fields behaved as expected.<br>**Tutor presentation:** Show learners the validation settings of the different fields. Show the different ways that entry is checked/restricted (give enough examples to cover the techniques listed in topic C4). Explain that although learners are currently looking at validation in a spreadsheet/database, the basic principles and logic are always the same. Explore the importance of validation within computer programs and systems that handle data. See video link: Validating User Input in Java (7 minutes) www.youtube.com/watch?v=PWez5mVXACc<br>• **Individual task:** Give learners a series of common scenarios that might require data to be held. In each case, give them some fields for which they should develop validation rules. Learners do not need to implement the rules in a database or program; they could just give a clear description of what the validation would be.<br>Fields should be checked for:<br>• presence (ie no empty cells)<br>• age should be validated – no one can be less than 0 and older than 120 would be unlikely | question – Use June 2017 | | Sp2<br>Sp5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | • format, eg UK National Insurance numbers have a specific format - LL 99 99 99 L.<br>• **Plenary:** Learners should work in pairs to compare and contrast their solutions. | | | |
| 31 & 32 | **C2 Arithmetic operations**<br>**C4 Validating data**<br>• Validation check techniques.<br>• Post-check actions. | **Knowing when to apply validation for a series of scenarios.** | • **Lead-in:** Recap the last lesson.<br>• **Individual task:** Give learners a series of scenarios which require validation techniques to be applied. Learners should write/develop/improve pseudocode that could be applied to produce sections of programming code. For some learners, introduce aspects of topic C5 as they explore post-check actions, although this is covered more in the next lesson.<br>• **Class discussion:** Discuss the solutions. Consider the different ways in which problems could have been solved. Explore the solutions to see if they give robust validation.<br>• **Plenary:** After the discussion, allow learners to revisit solutions and refine their work. | Improved pseudocode for a particular section of programming – pseudocode must contain clear validation | Validation exercises pg 163 – 164 BTEC Revision | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |
| 33 & 34 | **C2 Arithmetic operations**<br>**C4 Validating data**<br>• Validation check techniques.<br>• Post-check actions.<br>**C5 Control structures**<br>• Loops.<br>Branches. | **Knowing when to apply validation for a series of scenarios.** | • **Lead-in:** Recap the last lesson. Explain that learners will now be expected to develop more complicated solutions that require different levels of logic.<br>• **Individual task:** Give learners a series of scenarios which require validation techniques to be applied. To continue progression, scenarios should require nested logic, loops and branches to be used. Again, learners should write/develop/improve pseudocode that could be applied to | Ask learners to annotate their solutions to help explain their logic and function. Use this task to assess progress and give feedback to learners. | Validation exercises pg 165 – 166 BTEC Revision | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | produce sections of programming code.<br>• Limit checks (upper limit only).<br>• Checking a full name has been entered rather than just an initial.<br>• Check that inputs are correct months of the year or days of the week.<br>• Checking that a check-in date in a hotel is not after the check-out date (or a travel ticket scenario). | | | |
| 35 & 36 | **C5 Control structures**<br>Function calls. | **Why is modularisation used in programming?** | • **Lead-in:** Explain that in this lesson learners will be considering the use of functions/subroutines within programming.<br>• **Tutor presentation:** Introduce the concept of modularisation and why it is used in programming. Refer back to some of the ideas explored during the local and global variables topic, including values used within single functions and those used by more than one. Remind learners about the principles of naming conventions for variables and functions.<br>• **Small group activity:** This task requires a scenario that allows learners to look at a small area of a problem and devise a number of subroutines for it. Organise the class into groups of three and assign each group a section of the problem to work on. Give learners a program 'specification' that they can use to help identify what their part of the program must do. Help learners by using a similar problem to previous lessons (eg a simple computer game). When assigning tasks, ensure each group has the same task as at least one other so they can | **2nd AP testing – Jan 2020**<br><br>Re-organise learners so each group is now working with another group that was assigned the same problem. Ask the groups to discuss their solutions, considering why they broke their problem down the way they did. As a larger group, they should try to come up with a final list of subroutines | Validation exercises pg 167 BTEC Revision | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | discuss their solutions later. Each group should deconstruct their problem, and identify the different 'functions' that are required for the solution and the 'variables' that are needed. Identify the variables that would be used by only one function (local) and ones that may need to be accessed by others (global). Extend the task by asking learners to write a function using pseudocode.<br>Learners should understand that functions are used for two main reasons - a) to break down a program down so that it can be built by a team of developers who each create one or more functions that are they joined together in the main program and b) where code is identified as needing to be reusable.<br>Problems could include:<br>• a system to manage different aspects of a house such as switching lights on and off, heating, cookers, closing curtains etc.<br>• **Plenary**: Assessment Activity | | | |
| 37 & 38 | **C5 Control structures**<br>Function calls. | **Why is modularisation used?** | • **Lead-in:** Recap the last lesson. Through Q&A, gauge the learners' understanding and cover why modularisation is important, both in this case and in general.<br>• **Individual task:** Give learners an example program (written in pseudocode) for a stated problem. The example should contain some subroutines but also some inefficiencies. Learners should identify how the code could be improved and rewrite the code in a more efficient | Deep Assessment in Purple books on topics learnt over the last couple of weeks – with exam style question | BTEC Revision Book pg 168 | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | way. A good example of this would be to give learners a pseudocode where an integer number is input by the user several times in a program. Each time the number is input it has to be checked to confirm it is an integer within a specific range. The range can vary between inputs, but the process of checking is exactly the same. So the code to check the number is repeatedly written into the program.<br>Learners should then create a function that will pass in the number to be checked, plus the upper and lower limits and will pass back a Boolean value of true or false (is valid or is not valid). This means coding the validation routine just once and reusing as often as necessary.<br>• **Class discussion:** Discuss the learners' work and consider their solutions.<br>• **Plenary:** Further to discussion, allow learners to revisit solutions and refine their work. | | | |
| 39 & 40 | **C6 Data structures**<br>• Lists.<br>• Arrays. | **What are lists, and arrays; what is their purpose?** | • **Lead-in:** Give a brief introduction to the concepts of data structures. Explain how the lesson will expand on previous lessons, which have focused on single value variables.<br>• **Independent learning activity:** Learners do independent research into 'lists' and 'arrays'. They should research the characteristics of, and similarities and differences between, the two data structures.<br>• **Class discussion:** Use Q&A to check learners' understanding of lists and arrays. | Explanation of the purposes of lists and arrays. | BTEC Revision Book pg 147 | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | • **Independent task:** Give learners a series of scenarios/programming problems that require the use of data structures. They should identify in each case if a list or an array should be used, and explain why.<br>Problems could include the following.<br>  • A program that counts the number of spaces and different punctuation in an input string.<br>  • A program that accepts an input string and then encodes it by alternately taking characters from each end and creating a new list of characters. Hello world would become hdellrloow.<br>• **Small group task:** Ask learners to form small groups to discuss their answers. | | | |
| 41 - 42 | **C6 Data structures**<br>• Records.<br>• Sets. | **How are records and sets used?** | • **Lead-in:** Recap the last lesson. Use Q&A to check understanding of lists and arrays. Explain that in this lesson learners will explore other data structures used in programming (listed in topic C6).<br>• **Independent learning activity**: Learners do independent research into the characteristics of 'records' and 'sets'. See video link: Data Structures - arrays and records (5 minutes) in the following link www.youtube.com/watch?v=OirFldMTJWE<br>• **Independent task:** Learners should produce a written piece of work examining records, lists, sets and arrays. They should identify the characteristics of each of the structures, compare how they are | Written piece of work that identifies the characteristics of each of the structures. The work should compare how they are used, and discuss the benefits and drawbacks of each – Completed as a Deep Assessment task (Purple Books) | BTEC Revision Book pg 24 - 25 | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | used, and discuss the benefits and drawbacks of each. | | | |
| 43 - 44 | **C7 Common/ standard algorithms**<br>Sorting. | **What is the standard algorithm – sorting?** | • **Lead-in:** Explain that learners will now start to look at standard algorithms that are used in programming, starting with sorting algorithms.<br>• **Tutor presentation**: Explain the concepts of sorting algorithms and how the sorting algorithms in topic C7 work.<br>• **Small group activity:** Organise the learners into groups of two or three. Give them examples of the sorting algorithms (in pseudocode) and example data. Learners should use the data to produce a set of descriptions of how each of the sorts work. Their descriptions should show how the data is affected/stored at different stages of the sorting process.<br>• The following website has animations of sorting techniques and is a useful tool to show learners conceptually how each sort type works:<br>Sorting Algorithm Animations<br>www.sorting-algorithms.com<br>• **Plenary:** Through a class discussion and Q&A, explore learners' understanding of sorting algorithms. | Clear descriptions showing how the data is affected/stored at different stages of the sorting process. | BTEC Revision Pg 26 – 27 | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |
| 45 – 46 | **C7 Common/ standard algorithms**<br>Searching. | **How to search in algorithms?** | • **Lead-in:** Explain that in this lesson learners will look at searching algorithms.<br>• **Tutor presentation:** Explain the concepts of searching algorithms and how the searching algorithms in topic C7 work.<br>• **Small group activity:** Organise the learners into groups of two or three. | Results from activities should show clear descriptions of how each of the searches work. | BTEC Revision book Pg 28 – 29 | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | Give them examples of the searching algorithms (in pseudocode) and example data. Learners should use the data to produce a set of descriptions of how each of the searches work. Their descriptions should show how the data is used and outputted.<br><br>Algomation has a series of computing animations - use the following link and search on "searching" to find animations for a range of techniques<br><br>www.algomation.com/<br><br>• **Plenary:** Through a class discussion and Q&A, explore learners' understanding of searching algorithms.<br>• **Lead-in:** Explain that in this lesson learners will look at count occurrences and validation algorithms.<br>• **Tutor presentation:** Explain the concepts of the algorithms in topic C7 and how the count occurrences algorithm might work. Refer learners back to lessons 15–17 on validation when discussing validation algorithms.<br>• **Small group activity:** Organise the learners into groups of two or three. Give them example scenarios and, where appropriate, example data. Learners should, using pseudocode, produce example count and validation algorithms for assigned tasks.<br><br>Tasks could include those such as counting the frequency of ages in a dataset or the instances of eye colour or height in a group of people - this should show learners how data is analysed by spreadsheets and database functions.<br>• **Plenary:** In turn, ask each group to explain their solutions to the rest of | | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | the class. If time allows, facilitate discussion between learners as to the efficiency of the proposed solutions. | | | |
| 47 - 48 | **C7 Common/ standard algorithms**<br>Using stacks and queues to implement sorting and searching. | **How stacks and queues are used by computers when searching and sorting** | • **Lead-in:** Explain that in this lesson learners will look at how stacks and queues are used by computers when searching and sorting.<br>• **Tutor presentation:** Explain the concepts of stacks and queues in topic C7.<br>• **Small group activity:** Organise the learners into groups of two or three. Give them example algorithms (in pseudocode) and example data. The data should be simple such as numbers or characters. Learners should use the data to produce a set of descriptions of how stacks and queues work, both in writing and as diagrams.<br>• **Individual activity/** | Deep Assessment (Purple Books)<br>Give learners a series of tasks to assess their understanding of the content of topic C7. Assessment tasks should be varied in their approach and could include explaining how a specified algorithm works, showing how data is processed by a particular algorithm, writing a simple algorithm in pseudocode, and identifying errors/improving efficiency of a given piece of pseudocode. | Revision questions set to, explore learners' understanding of stacks and queues | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |
| 49 – 50 | **D1 Procedural programming** | **Learning the concepts of procedural programming by using Python 3.4.** | Learners should spend these lessons learning the concepts of procedural programming by using Python 3.4.<br>• **Tutor presentations:** Explain key syntax, commands and procedures as they are introduced and learners' understanding of the programming language progresses. The concepts and functions could be taught using a similar progression to topic C (lessons 9–26). | Written code that solves problems (without errors in the written code) | Long exam question to be set – students to develop their answers | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | • **Individual learning activities:** Give learners opportunities to write small amounts of code to solve problems and explore features of the programming language. They should also be given larger code examples that they must debug and improve.<br>Problems have been supplied in lessons 5, 6 and 7.<br>• **Examination preparation:** Learners will not be required to write large sections of code from scratch in the exam but working with the code will aid understanding. Develop exam techniques through starter, plenary and homework activities. | | | |
| 48 - 66 | **D2 Object-orientated programming** | Learning the concepts of procedural programming by using a C family language | Learners should spend these lessons learning the concepts of procedural programming by using a C family language. See video link: Intro to Object Oriented Programming (C++) (5 minutes) in the following link www.youtube.com/watch?v=MTfWr7o7Q4U<br>• **Tutor presentations:** Explain key syntax, commands and procedures as they are introduced and understanding of the programming language progresses. The concepts and functions could be taught using a similar progression to topic C (lessons 9–26).<br>• **Individual learning activities:** Give learners opportunities to write small amounts of code to solve problems and explore features of the programming language. Learners should also be given larger code examples that they must debug and improve. | Code that has been de-bugged by the learners | Long exam question to be set – students to develop their answers | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | Examples could include the players in a game or athletes at an event, capturing performance data for later analysis.<br>**Examination preparation:** Learners will not be required to write large sections of code from scratch in the exam but working with the code will aid understanding. Develop exam techniques through starter, plenary and homework activities. | | | |
| 67 | **D3 Event driven programming** | Explanation of the features and characteristics of the event driven programming paradigm | • **Lead-in:** Explain to learners that they will now look at event driven programming.<br>• **Tutor presentation:** Give an overview of the features and characteristics of the event driven programming paradigm. Introduce the structure of event driven programming. See video link: What are event-driven programs? in the following link www.youtube.com/watch?v=qrA7eD18CZo<br>• **Individual task:** Learners should research and make notes on the features, uses, benefits and drawbacks of event driven programming.<br>• **Small group task:** Re-organise learners into small groups to discuss their findings. Allow them to add to and expand their notes based on the discussions. | Clear explanations given on the features and characteristics of the event driven programming paradigm | | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |
| 68–86 | **D3 Event driven programming** | **What are the concepts of event driven programming by using Visual Basic®** | Learners should spend these lessons learning the concepts of event driven programming by using Visual Basic®. | Clear explanations of key syntax, commands and procedures of the programming/mark-up language Visual Basic® | Develop exam techniques through starter, plenary and homework activities. | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | • **Tutor presentations:** Explain key syntax, commands and procedures as they are introduced and understanding of the programming language progresses. The concepts and functions could be taught using a similar progression to topic C (lessons 9–26).<br>• **Individual learning activities:** Give learners opportunities to write small amounts of code to solve problems and explore features of the programming language. Learners should also be given larger code examples that they must debug and improve.<br>Problems could include:<br>○ maths program for children, for example testing simple additions and subtractions.<br>• **Examination preparation:** Learners will not be required to write large sections of code from scratch in the exam but working with the code will aid understanding. Develop exam techniques through starter, plenary and homework activities. | | | |
| 87 | **D4 Coding for the web**<br>• The uses, applications and implications of client side processing and scripting.<br>The uses, applications and implications of server side | **Gain an understanding of coding for the web by exploring the principles of server side and client-side processing.** | • **Lead-in:** Explain that learners will expand their understanding of coding for the web by exploring the principles of server side and client-side processing.<br>**Tutor presentation:** Give an overview of the concepts of server and client processing. See video links: Web technology tutorial: Front end design (3.75 minutes) and Web technology tutorial – Server-Side Scripting (3.5 minutes) in the following links Web technology tutorial: Front end design (3.75 minutes) | Learners should share their findings and give expansions and feedback on other learners' work. | | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | processing and scripting. | | www.youtube.com/watch?v=-qI6Xns-w90<br>Web technology tutorial - Server Side Scripting (3.5 minutes)<br>www.youtube.com/watch?v=JnCLmLO9LhA<br>• **Individual task:** Learners should conduct research and make notes on server and client side processing, giving an analysis of how each is used and implemented, and the benefits and drawbacks of each.<br>• **Class discussion:** Through discussion and Q&A, establish learners' understanding. They should share their findings and give expansions and feedback on other learners' work.<br>• **Plenary:** Give learners time to expand their notes based on class discussion. | | | |
| 88 - 106 | **D4 Coding for the web** | **Learning the concepts of coding for the web by using HTML5.** | Learners should spend these lessons learning the concepts of coding for the web by using HTML5.<br>• **Tutor presentations:** Explain key syntax, commands and procedures as they are introduced and understanding of the programming/mark-up language progresses. The concepts and functions could be taught using a similar progression to topic C (lessons 9–26).<br>• **Individual learning activities:** Give learners a chance to write small amounts of code to solve problems and explore features of the programming/mark-up language. They should also be given larger code examples that they must debug and improve, or evaluate in terms of issues and implications. | Clear explanations of key syntax, commands and procedures of the programming/mark-up language HTML5 | BTEC Revision workbook<br>Assessment 1 | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | Problems could include:<br>• a site for extreme cycling locations and routes that needs new pages for competition events.<br>• **Examination preparation:** Learners will not be required to write large sections of code from scratch in the exam but working with the code will aid understanding. Develop exam techniques through starter, plenary and homework activities. | | | |
| 107-108 | **D5 Translation** | **How to provide a full analysis and recommendations of a situation.** | • **Lead-in:** Explain that learners will apply what they have learned over the last few lessons regarding translation of code. Organise the learners into pairs.<br>• **Small group activity:** Give each pair a unique scenario and some example code from any of the studied languages. The pairs should prepare a business-style presentation (that they will give to the rest of the group) which analyses the situation using the D5 topic and makes a supported recommendation as to whether the code should be translated in this instance.<br>• **Plenary:** Each group should present their analysis and recommendations to the rest of the group. Give the other learners opportunities to ask questions and give feedback. | A comprehensive presentation that consists of a full analysis and recommendations of a situation provided by the teacher | BTEC Revision workbook Assessment 1 | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |
| 109 - 110 | **Whole specification** | **What needs improvement?** | • **Lead-in:** Introduce the purpose of the lessons: to look at how to improve performance on the extended writing questions. | Improved responses from extended written answers | Revise areas of weakness (This will be different for each learner, depending on their needs) | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | • **Tutor presentation:** Reiterate the meaning of different command words. For each of the 'extended writing' command words, look at the structure of the level-based mark schemes and what the descriptors mean.<br>• **Individual tasks:** Give learners example extended questions from either the SAMs, past papers or ones that the tutor has prepared. Allow them to respond to these in exam conditions. Tutors could give example responses that learners could critique.<br>• **Small group tasks:** Learners should discuss/share their interpretations of the questions. They could do a peer marking activity. | | | Sp5 |
| 111 - 112 | **Whole specification** | **What needs improvement?** | • **Lead-in:** Introduce the purpose of the lessons: to allow learners to work on areas of the specification they find challenging. Ask them to look through the specification (and their work from the last few lessons) and identify where they think they need additional focus.<br>• **Tasks:** Set up 'stations' in different areas of the room that learners can use, based on their areas of weakness. | Improved responses from extended written answers | Revise areas of weakness (This will be different for each learner, depending on their needs) | Lit<br><br>Social<br><br>So8<br>C3<br>Sp2<br>Sp5 |