**Text

Description automatically generated with medium confidenceLogo

Description automatically generated with low confidence Scheme of Work: 2022 - 23**

|  |  |  |  |
| --- | --- | --- | --- |
| **Course:** | ***T Level Digital Production, Design and Development*** | **Level:** | ***Level 3*** |
| **Specification Unit(s):** | ***ESP, Core 1 and Core 2*** | **Lecturer(s):** | ***John Glazebrook*** |
| **Curriculum Intent, Implementation and Impact** | | | |
| **Intent of this course** | **Students who complete the T level Digital Production, Design and Development will be able to choose between moving on to further study; for example, a higher or degree level apprenticeship, or higher-level technical study, including higher education, or into a skilled occupation locally or nationally. In addition to specific programming and other IT/Computing knowledge, understanding and practical skill development, learners will also develop other essential work-based behaviours and transferrable skills both through classroom study and through their industry placement. All learners will have the opportunity to develop their skills and practice via workshops, classroom activity and whilst in industry placement.**  **The scheme of work provides a coherent programme of learning and personal development with a high level of pastoral support provided through a comprehensive tutorial system. Learners also have access to a wider programme of academies, study plus, health and well-being and student services provision to enable them to embrace wider aspects of college life, enhance their understanding of future careers, pursue their individual interests and make valuable contributions to the community.Learners will also develop time management skills, the ability to prioritise and enhance research skills alongside embedded delivery of Maths and English.** | | |
| **Intended learner destinations** | **Alongside progression to higher education and apprenticeships, the T level supports progression into entry level job opportunities in Software design and development including:**   * **Junior Software Developer** * **Junior Web Developer** * **Junior Application Developer** * **Automated Test Developer** | | |
| **Topics Covered** | **The core content covers the knowledge, understanding and application of contexts, concepts, theories and principles relating to the following areas:**   1. **Problem solving** 2. **Introduction to Programming** 3. **Emerging issues and impact of digital** 4. **Legislation and regulatory requirements** 5. **Business Context** 6. **Data** 7. **Digital Environments** 8. **Security**   **The Core Project (ESP) covers the knowledge, understanding and practical application of the following areas:**   1. **Planning a Project** 2. **Identifying and fixing defects in existing code** 3. **Designing a solution** 4. **Developing a solution** 5. **Reflective Evaluation** | | |
| **Skills, knowledge, attributes developed through the curriculum to support academic and career progression** | **Embedded within this qualification are academic, professional, employability and soft skills, for example:**  **General English and Maths Competencies:**   * **Convey technical information to different audiences** * **Present information and ideas** * **Create texts for different purposes and audiences** * **Summarise information/ideas** * **Synthesise information** * **Take part in/leading discussions** * **Estimate, calculate and spot errors** * **Work with proportion** * **Use rules and formulae** * **Process data** * **Understand data and risk**   **Leaners will also develop their employability, professional and soft skills through their industry placement, employer led workshops and talks, and dedicated employability skills sessions, careers advice and guidance sessions, CV writing workshops and a full tutorial provision.** | | |
| **Assessment methods** | **External assessment takes place from May in the form of Two core papers (1 and 2) and one employer set project.**  **Formative assessment takes place throughout study with the regular submissions of work for feedback and improvement across all content areas.** | | |
| **Additional elements that make up the curriculum offer** | **The TLevel Diigital production, design and development also includes the following enrichement opportunities:**   * **Guest speakers/trips+visits/industry and career related activities** * **A minimum of 315 hours of Industry placement with relevant local employers.** * **Navigate Career Exploration Award – Completed before February half term - increases awareness of career options and opportunities, to encourage learners start to build career ambitions/ plans. Learners prepare a CV to help them prepare for work experience, placement or part-time work opportunities.** * **Tutorial Programme – the tutorial programme covers core subjects related to learner safety, personal development and growth and to help learners navigate the challenges of becoming an adult. The tutorial programme focuses significantly on preparation for next steps and careers and provides the opportunity for 1:1 guidance and support.** * **Employability – learners complete Navigate Awards to help them hone their career ambitions and develop the skills, knowledge and confidence needed to make employer connections and successfully apply for jobs. Learners have a variety of employer talks to enable them to develop their network of contacts and understand the skills local employers require.** | | |
| **Local and National Skills Needs Addressed/Local Market Information** | * **The** [Final-CIoS-DRAFT-Industrial-Strategy-09.03.20.pdf (cioslep.com)](https://cioslep.com/wp-content/uploads/2021/04/Final-CIoS-DRAFT-Industrial-Strategy-09.03.20.pdf) **states that there are 3915 jobs across 710 businesses within the space and digital sector and 11440 jobs across 2350 businessses within professional business services.** * **Software engineering is currently one of the most in demand skills in the UK – In April 2021, nearly 10,000 roles in software development/engineering were advertised (Ipsos Mori, 2021). The FdEng Software Engineering is in designed to contribute, in its subject area, towards nationally and regionally recognised technical skills shortages and responsive initiatives (IoTs1 (specifically SWIoT) and Higher Technical Qualifications ). The challenges identified within the Cornwall LEP Digital Strategy and Heart of the South West Digital Strategy recommend the following three actions that will be addressed by this programme:** * **There is also a program of higher level study within Truro college such as a new Sotware Engineering degree with first intake in 2023, which has been developed in collaboration with local employers and bridging local and national skills gaps within the digital sector.** | | |
| **Stakeholder Links** (Employers/ 3rd party partners (e.g. universities), industry experts, professionals involved in curriculum content planning and delivery | **An Employer led steering group has been started with local industry employers in Cornwall, the initial meeting took place in February 2022 and subsequent meetings of the group will take place throughout the development of this programme. T&PC is part of the Cornwall Isles of Scilly Digital Skills partnership, has strong links with the developing space sector in Cornwall through the Cornwall Space and Aerospace Technology Training (CSATT) project and hosted the recent Deep Dive event in our new Future Skills Institute.**  **Regular industry professionals/experts delivering presentations, workshops and talks (GCHQ, HiYield, Hi9, NHS, Cornwall Council, Software Cornwall).**  **University encounters, HE open day and UCAS support sessions embedded in tutorial provision to support progression to higher education.** | | |
| **Contact details of key employers/partners/professional links:**  **Andy Hayers – Royal Cornwall Hospital (Treliske) - andy.hayers@nhs.net** | | |

**Course Sequencing (TBC)**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Sep** | **Oct** | **Nov** | **Dec** | **Jan** | **Feb** | **Mar** | **Apr** | **May** | **Jun** |
| Core 1  *John Glazebrook* |  |  |  |  |  |  |  |  |  |  |
| Core 2  *Richard Cotton* |  |  |  |  |  |  |  |  |  |  |
| ESP  *Naomi Johns* |  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Course:** | ***T Level Digital Production, Design and Development*** | **Level:** | ***3*** |
| **Specification Unit(s):** | ***Core 1, ESP*** | **Lecturer(s):** | ***John Glazebrook*** |

| **Week No.**  **and /or**  **w/c date** | **Lesson No.** | **Specification Reference** | **Session Objectives/Content** | **Teaching Methods, Student Activity and how progress is checked** | **Assessment and Feedback**  **(inc Homework)** | **Resources / Support materials** | **Numeracy, Literacy, EDI, Employability, Transferable Skills**  **Wider links** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Wk 1  9th Sept |  | 1.1 Computational thinking | Introduction to course. Ice breakers. Intro to lecturers and who teaches what.  Go over expectations of a student. Ensure all have notebooks and how to take effective notes. |  | Tutor to closely observe to pick up any learners that may need support. | Available in OneNote | Computational thinking.  Communication, research, critical reading skills used. |
| Wk 1  10th Sept |  | 1.1 Computational thinking | Intro to problem solving techniques – in class discussion. Help students to get to know each other, icebreaker activity. Group work. Lecturer to get to know students better.  Intro to coding – what is a variable? Simple maths. | Wolf / goat / cabbage  Worksheet 01 – simple coding challenges. Variables and maths. Maths: add, minus, multiply, divide. | In class  Complete worksheet for HW | Available in OneNote | Computational thinking. |
| Wk 2  14th Sept |  | 1.1 Computational thinking | Polya. Polya 4 step problem solving technique:   * First principle: Understand the problem * Second Principle: Devise a plan * Third principle: Carry out the plan * Fourth principle: Look Back   Discussion. Investigate other problem solving strategies. | Worksheet 3 builds on the previous skills the students have learnt. Polya -- Follow Poylas 4 steps to solve the tasks:   * The power (\*\*) operator * Convert MPG to KM per L   Students to document application of their problem solving skills. | In class  Complete worksheet for HW | Available in OneNote | Computational thinking.  Communicate using mathematics.  Learners will develop wider employability skills, will use numerical concepts and will develop analytical skills. |
| Wk 2 |  | 1.2 Algorithms  2.4 Program structure | Coding -- Selection  Students to get familiar with the python keywords:   * if * else * elfi | Worksheet 04 builds on the previous skills the students have learnt.  Students to create posters on selection and logic – posters then discussed, students to look at each others posters and try to find any bugs (feedback, class discussion, pair programming) | In class  Complete worksheet for HW.  Formative informal feedback provided throughout the session | Available in OneNote | Employers are looking for computational thinking and algorithm understanding |
| Wk 2 |  | 1.2 Algorithms  2.4 Program structure | Problem solving: Top down / bottom up   * Top down – split problem into smaller chunks, then split again. * Bottom up – design the smallest components, then join them together. | Worksheet “05 - top down vs bottom up.docx”. Builds on the previous skills the students have learnt.  Students to research and read about both problem solving techniques. They then write how to approach:   * Making dinner (top down, create the elements - all the bits come together to make the meal at the end) * Creating a new board game (prototyping, testing, feedback. End result is not exactly what you thought)   Lecturer to read and feedback on techniques in lecture.  Students to create posters on problem solving and print them out. | In class  Complete worksheet for HW | Available in OneNote | Employers are looking for computational thinking and algorithm understanding |
| Wk 2 |  | 1.2 Algorithms  2.4 Program structure | While loops.  Students will learn about loops, how loops are represented in flowcharts, how to implement simple loops that count up and down.  Lecturer to demonstrate each on board. Students to copy work into interactive portfolio so they have it for reference in future.  Ensure that students are naming their work correctly and that they can find it for use in worksheet 06  This builds on sequence and selections previously taught. | “06 - Worksheet 06 – loops”. Students to implement a simple menu for the exam marks application. When user enters ‘E’ to exit, the loop stops.  Lecturer to demonstrate on board if students struggle. Group coding exercise. Lecturer to talk through logic and step through code. | In class  Complete worksheet for HW.  Formative informal feedback provided throughout the session.  Peer feedback. | Available in OneNote | In lecture discussion of programming skill needed in employment. |
| Wk 2 |  | 1.2 Algorithms  2.4 Program structure | **For loops**. This builds on the skills learnt in the last lecture, when we looked at **while** loops, that loop a *variable* number of times.  **For** loop and **range** statement fix the number of times a loop executes. Loop over lists.  Range command, start, stop and step parameters. Example: times tables.  Lecturer to demonstrate each on board. Students to copy work into interactive portfolio so they have it for reference in future.  Ensure that students are naming their work correctly and that they can find it for use in worksheet. | Worksheet 07 – “python iteration tasks”.  Create a shopping list. Loop over the list to display the items. Some students will query the list to see if an item exists. | In class  Complete worksheet for HW.  Formative informal feedback provided throughout the session.  Peer feedback. | Available in OneNote | In lecture discussion of programming skill needed in employment |
|  |  |  | Naomi – should we have a lecture on worksheet catchup? One-to-one Feedback and further problems set by lecturer? |  |  |  |  |
| Wk 3 |  | 1.2 Algorithms  2.4 Program structure | **Modulisation**. Break problems (and computer programs) down into modules. Each module encapsulates a single idea.  **Pattern recognition** – has someone solved a similar problem in the past? Heuristics. | Worksheet 08 – “Modulisation decomposition” **research** and write a report on:   * Modulisation * Decomposition * pattern recognition.   Lecturer to monitor and feedback on research. A lot of this is done online and students may become unfocused. Lecturer to check in regularly and ask for **class feedback/discussion** on articles they have found and read. | In class  Complete worksheet for HW.  Formative informal feedback provided throughout the session | Available in OneNote | Employers are looking for computational thinking and understandin. |
| Wk 3 |  | 1.2 Algorithms  2.4 Program structure | Compound if. Nested if.  Selection – which code to run under what conditions.  Introduction to **and,** **or** and **not** Boolean logic. Some students will have researched this in **lecture 4**, these students to discuss how the keywords work.  Lecturer to draw the truth tables for each. Students to copy these into their notebooks.  Lecturer to demonstrate each on board. Students to copy work into interactive portfolio so they have it for reference in future. | Worksheet 09.   * Tropical fish tank temp, display message if too hot or too cold * Students grades (A to F) – input a percentage. Think about possible errors that may occur, e.g. -10, or 110, or “abc” * Nested if statements * Self driving car (are doors shut, are seat belts on?) * Lecturer to give feedback in lecture. Encourage students to look at each others work to see how other people are solving the same problem. | In class  Complete worksheet for HW.  Formative informal feedback provided throughout the session | Available in OneNote | Employers are looking for computational thinking and understanding |
| Wk 3 |  | 1.2 Algorithms | Abstraction.  Simplification of complex data, e.g. London underground map.  How coders abstract away from implementation details in code. Discussion of Python classes and how they relate to modules. Why some methods can be marked as private.  Video of algorithms and how the automation of systems affects us all. | Worksheet 05 abstraction – students to read articles linked and watch videos linked.  Write a short report on:   * Abstraction. * What is an algorithm   “10 - Worksheet 10 - further problems” – small test and further Python problems.  5 tasks taken from real life coding problems. | In class  Complete worksheet for HW.  Formative informal feedback provided throughout the session | Available in OneNote | Writing reports.  Literacy and critical thinking skills. |
| Wk 3 |  | 1.2 Algorithms | Flowcharts. Introduction to flowcharts, what all the shapes mean (pill, arrow, rectangle, parallelogram and diamond)  Builds on: **sequence, selection and iteration**  All students will be able to draw flowcharts that can count to ten and that can make simple decisions.  Some students will be able to create simple functions and call those using a flowchart (double barred rectangle) | Flowchart worksheet + draw.io website  All students to familiarize themselves with the **T Level Specification for flowcharts.**  One-to-one feedback on worksheet 10 (previous lecture) – focusing on **variable names** and **code comments**. | In class  Complete worksheet for HW | **T\_Level\_Specificaion for flowcharts**. | Algorithms – maths, logical thinking. Employers are looking for critical thinking skills. Understanding a spec. |
| Wk 3 |  | 1.2 Algorithms | Pseudocode. Introduction to pseudocode and how it relates to flowcharts and to Python.  When we use it in real projects. | Complete flowchart worksheet with pseudocode as per T-Level spec  **All students to familiarize themselves with the T Level Specification for pseudocode.** | In class  Complete worksheet for HW | **13 - Pseudocode as per T\_Level\_Specification** | Algorithms – maths and logical thinking – employability.  Learners will develop wider employability skills, will use numerical concepts and will develop analytical skills. |
| Wk 4 |  | 1.2 Algorithms | Consolidation.  Recap:   * Python * Selection, sequence and iteration * Truth tables * Code comments   Complete flowchart and pseudocode worksheets with pseudocode as per T-Level spec.  One-to-ones go over student notebooks. | Complete worksheets. Lecturer to go over any points of confusion.  One-to-ones go over student notebooks. Mind maps.  Task students with extension work:   * “13 - worksheet guessing game” * “13 - time” | Worksheet marking and verbal feedback | Available in OneNote | Flowcharts – maths and logical thinking – employability.  Learners will develop wider employability skills, will use numerical concepts and will develop analytical skills. |
| Wk 4 |  | 1.2 Algorithms  2.4 Program structure | Intro to functions. Students can identify:   * Def keyword * Function name * Parameters * Indentation of block * Statements * Return statement   Lecturer to model code and students to follow along.  Lecturer to demonstrate each on board. Students to copy work into interactive portfolio so they have it for reference in future. | Further programming exercises worksheet. Students to practice writing simple functions, worksheet builds on previous problems and increases complexity.  Extension: supermarket discount application | Worksheet marking and verbal feedback.  Formative informal feedback provided throughout the session | Available in OneNote | Functions in code match functions in maths.  Learners will develop wider employability skills, will use numerical concepts and will develop analytical skills. |
| Wk 4 |  | 1.2 Algorithms | Algorithm errors, AI and cyphers. Students will understand how minor biases in algorithms can compound to make big effects.  Discuss the following topics:   * Using an algorithm to generate chaos * Unintended / intended bias in elections * Ethical considerations and fixing (or changing) algorithms (e.g. Twitter shows white faces before coloured faces)   Extension: Caesar cypher worksheet. | Worksheet 15 - Report / discussion. Students to research (links in worksheet) different algorithms, e.g. AI and bias.  Students to discuss different ideas presented during lecture. | In lecture discussions.  Formative informal feedback provided throughout the session | Available in OneNote | Logical thinking. Problem solving – and wider employability.  Present information and ideas  Summarise information/ideas  Synthesise information |
| Wk 4 |  | 2.5 built in functions | Design documentation – cover:   * Abstraction of data * Use of predefined code * Readability of code * Quality of code | Quality of Code Key Terms – worksheet to be filled in. | Formative feedback on worksheet | Available in OneNote | Logical thinking. Problem solving - employability |
| Wk 4 |  | 2.5 built in functions | Data structures - Lists. Demo how to:   * Create * Edit * delete items from a list.   How lists are stored in memory – discussion.  Lecturer to demonstrate each on board. Students to copy work into interactive portfolio so they have it for reference in future. | Worksheet – “Worksheet lists and menus”  Students to research all list methods, including:   * range * item exists? * Append * Delete * Join two lists   Looping over a list to inspect each element.  Remove duplicate entries  Inspect all elements of a list  2D arrays (lists of lists) | Formative feedback on worksheet | Available in OneNote | Logical thinking. Problem solving – employability. |
| Wk 5 |  | 2.5 built in functions | Further Lists   * Pop * Sort * Reverse * Comprehension * Membership   Students to make notes in notebook. This lecture is focused on the worksheet. Ensuring students are happy with lists and how to use them.  This is the first data structure with complex behaviour that they will learn. Lists naturally lead on to more complex structures such as dictionaries.  It is important that all students are comfortable with lists and can complete the worksheet. | Worksheet 19  Coding practice with lists and 2D arrays  Students complete “19 - worksheet more list operations”:   * Indexes, negative indexes * Slices * Sort * Reverse * Pop * Comprehension * Membership tests   2D arrays (lists):   * Create * Access elements and sub-elements * Loop over elements in X axis * Loop over elements in Y axis   Some students to attempt: “19 - worksheet more list operations PART 2”  More complex and real life uses of lists:   * Create a 2D array of Music Band Facts * Using a 2D array – represent pixels on a screen. Draw to this grid. | Formative feedback on worksheet | Available in OneNote | Logical thinking. Problem solving - employability |
| Wk 5 |  | 2.5 built in functions | Python Dictionaries   * Create * Access one element by key * Access all elements by key * Edit an element * Delete an element * How many elements * Does an element exist * Looping   Lecturer to over more complex examples on the board, class discussion.  Lecturer to demonstrate each on board. Students to copy work into interactive portfolio so they have it for reference in future. | Worksheet 20  Recap previous lecture.  Students complete worksheet: Dictionary Worksheet.docx  Complete all the dictionary operations:   * add items * understand keys and values and their associations * months lookup table * fizzy sports drinks lookup table * Dictionary of lists (building on last lecture) * Merge two dictionaries * Loop over dictionary printing either key or value * Dictionary of dictionaries * Build a simple “RPG” game   Complete the tasks in worksheet. | Formative feedback on worksheet | Available in OneNote | Logical thinking. Problem solving - employability |
| Wk 5 |  | 2.5 built in functions | Dictionaries lecture 2 - finish off worksheet.  Consolidation lecture. One-to-one feedback on dictionary worksheet.  Darts problem takes about an hour to code. Lots of modelling without giving the complete answer. | Worksheet 20  Complete the worksheet from the previous lecture.  All students to complete problem 3 – darts.  Some students will complete further problems. | Formative feedback on worksheet/code | Available in OneNote | Logical thinking. Problem solving - employability |
| Wk 5 |  | 2.5 built in functions | Consolidation – complete lists and dictionaries worksheets.  Extension: student to use functions to organise their code.  Naomi – insert fancy pants reasons for this lecture | All worksheets to be completed. Students to attempt the extension / harder challenges. | Formative feedback on outstanding worksheets and any code the students wish to share | Available in OneNote | Logical thinking. Problem solving - employability |
| Wk 6 |  | 2.7 Maintainable Code  2.8 Testing | Consolidation of skills project - RPG game/text adventure.  Reason: so far students have learnt how to code, and how lists and dictionaries work. They have **not** attempted a large body of code. This project will give them time to concentrate on one single task. They will learn:   * How to manage their time * Manage a large codebase * The complexity of how code interacts * User interfaces * Real world use of data structures * Testing * Iterative coding technique * Learn to read large chunks of code (e.g. in exam)   Lecture: RPG Introduction   * Generating game ideas * Mapping the game locations   Location and location descriptions held in a dictionary. Items held in a list.  Worksheets and lecture notes from previous lectures used as reference. | Discussion with students – how to attempt this project. How to break it down into manageable tasks.  Students to create a task list.  How to keep the game simple enough to complete, but with an eye on how to expand it if time allows.  Students to create the game map. Using dictionaries and lists. | Formative feedback on code | Available in OneNote | Following instructions and completing a program to a spec – employability. |
| Wk 6 |  | 2.7 Maintainable Code  2.8 Testing | Consolidation of skills project - RPG game/text adventure (lecture 2).  Implementing the:   * Input * Move * Output   Loop | Implementation of a simple menu system that allows the user to navigate around the map.  Lecturer to check progress:   * Direction input (N,E,S,W) * Exit the game * Location descriptions being displayed | Formative feedback on code | Available in OneNote | Following instructions and completing a program to a spec - employability |
| Wk 6 |  | 2.7 Maintainable Code  2.8 Testing | Consolidation of skills project - RPG game/text adventure (lecture 3).  Allow the user to move around the map and see locations.  Check that the user can not move in a direction that is not allowed.  Check that the students map and their game match.  Implement the players inventory (one item) | Class discussion of how to implement the player inventory.  Lecturer to check progress:   * Increased number of locations * Player inventory - one item | Formative feedback on code | Available in OneNote | Following instructions and completing a program to a spec – employability. |
| Wk 6 |  | 2.7 Maintainable Code  2.8 Testing | Consolidation of skills project - RPG game/text adventure (lecture 4).  Students will:   * Add items to locations * allow the user to interact with them * E.g. the user should pick up a key to open a door.   **Some** students will implement the inventory as a list. | Lecturer to check progress:   * Player inventory – add, remove (drop) items * Locations that have items – e.g. a key * Locations that require items – e.g. a door needs a key * “use key” * New description of room “with an open door to the west” * New exit available – through door, west | Formative feedback on code | Available in OneNote | Following instructions and completing a program to a spec - employability |
| Wk 6 |  | 2.7 Maintainable Code  2.8 Testing | RPG game – Project / Consolidation  Separation of concerns, ensure menu, IO and logic all in separate functions.   * Move all UI code into functions * Move all game logic into functions * Function names should be descriptive   Extension tasks:   * Implement combat. * Enemies * NPCs that move * Wield weapons | Lecturer to check progress:   * UI/UX code to be separate from game logic. | Formative feedback on code | Available in OneNote | Scoring systems, dice, probability - maths |
| Wk 6 |  | 2.7 Maintainable Code  2.8 Testing | Complete RPG games. Show off games! Let students play and help debug the games.  Verbal feedback on games. Lecturer to tie this to final year project, how feedback is part of the assessment. | Students to give feedback. In class discussion of good feedback, constructive criticism. | Formative feedback on code | Available in OneNote | Scoring systems, dice, probability - maths |
| Wk 7 |  | 2.5 built in functions | Coding data structures. Tuples and sets.  Students will understand what a tuple is and how is differs from a list. What operations can be performed on it.  Students to understand what a set is and how it differs from a list. What operations can be performed on it. | Worksheet – “tuples and sets worksheet”  Tasks to create tuples. Students to answer questions on tuples and lists.  Implement 2 sets, add items to one or the other. The show the **union**, **intersection** and **difference**.  Student investigation: what is a **frozenset**? | Formative feedback on code | Available in OneNote | Sets and group theory - maths |
| Wk 7 |  | 2.5 built in functions | Importing modules.  What is a module and why do we use them. Discussion.  Reminder of flowcharts and functions. Talk about modules and how they fit in.  Lecturer to demonstrate a simple module with three functions in it.  Students to do some research on how to pull modules into their own code.  Complete worksheet. | Worksheets to be completed. Extension sections to be attempted.  Students to email lecturer a link to the module they created.  The lecturer will pull that module into the class code.  Lecturer will call each of the three functions in that module and then read out the results.  Students to then complete further tasks in the worksheet:   * Datetime module * Calendar module * Math module * Random module   Students to research the PATH variable and how Python finds modules and what order they are pulled into the code. | Formative feedback on worksheet | Available in OneNote | Reusing libraries and modules. Efficiency - employability |
| Wk 7 |  | 2.7 Maintainable Code | Menu workshop – command line menus, separation of concerns.  Some students will have attempted separation of concerns in the text adventure.  This is often a tricky concept and requires a lecture on consolidating this skill.  It also consolidates skills:   * Functions * User interface * Modules | In lecture tasks – modelling on digital whiteboard. Students to copy and extend demo code.  Then students will   * Chose a previous program they have written * Copy it * Apply separation of concerns to it * Show lecturer for feedback | Formative feedback on code | Available in OneNote | Making programs useable and helpful - employability |
|  |  |  | **HALF TERM** |  |  |  |  |
| Wk 8 |  |  | **Project week** – Chada BMX coding project | Project to be completed. | Marking and one-to-one feedback on project | Available in OneNote | Following and understanding a specification. Writing code to satisfy requirement. |
| Wk 9 |  | 2.5 built in functions | TODO – fix this? Strings – string manipulation and investigate regex.  Explain regex   * string find and replace – case insensitive * Parts of email address, replace domain name * Phone number extension | 25 - strings worksheet v1.2  7 tasks, each building on the previous. Checking character is in string, string length, and other methods. | Formative feedback on worksheet | Available in OneNote | Reusing libraries and modules. Efficiency - employability |
|  |  |  | Naomi - REGEX – do we need a lecture on this, I get students to do some research and learn it after they complete the above lecture. I was going to put a new lecture in this year – do I need to? Has it ever come up in an exam? I need to check the spec  Its not specifically mentioned in the spec. |  |  |  |  |
| Wk 9 |  |  | **Revision** – revise all content for Assessment 1 | Questioning. | Assessment | Available in OneNote |  |
| Wk 9 |  |  | **Revision** | Questioning. | Assessment | Available in OneNote |  |
| Wk 9 |  |  | **Assessment 1** |  | Feedback in lecture. Go over all answers. In class discussions. | Available in OneNote |  |
| Wk 9 |  | 2.3 File handling | Files – open and read and close. Lecturer to model each operation. Students to follow along. Files:   * Films.txt * Films.csv   Students to read in data, parse data into structures (e.g. a list)  Students will now be expected to complete larger works of code, often taking multiple lectures / workshops to complete. | In lecture. Check on progress. Answer common questions. Model answers if needed.  Produce the following programs   * Read the file films.txt * Filter the output by genre, or date, or rating or length | Formative assessment of code and comments produced. Answer any questions. | Available in OneNote | Reusing libraries and modules. Efficiency - employability |
| Wk 10 |  | 2.3 File handling | Files 2 – write data. Building on previous lecture.  CRUD – weather application.  5 functions, C.R.U.D and menu.   * C Create * R Read * U Update * D Delete   Note: This usually takes two lectures to complete the “write files worksheet” | Students to write application to save weather/temp data in a file. Use the ‘append’ flag to write data.  Extension: provide complete CRUD, create, read, update and delete for each record of data. | Formative assessment of code and comments produced. Answer any questions. | Available in OneNote | Reusing libraries and modules. Efficiency - employability |
| Wk 10 |  | 2.3 File handling | Consolidation: Files  Ensure application works and investigate the different file modes, “r”, “w”, “a”, “rw”  Students to complete weather application. | Students to write application to save weather/temp data in a file. Use the ‘append’ flag to write data.  Extension: provide complete CRUD, create, read, update and delete for each record of data. | Formative assessment of code and comments produced. Answer any questions. | Available in OneNote | Reusing libraries and modules. Efficiency - employability |
| Wk 10 |  | 2.3 File handling | Files 3 – File formats  This usually requires **two lectures**.  Students to complete a fun task of reading unstructured data. This takes around 30 minutes, Lecturer to model solution on board.  Finite state machines – discussion.  Data formats and problems they have – discussion.  Introduction to JSON and pickle. | Worksheet 30 – exporting data in different file formats, csv, json and pickle. | Formative feedback on worksheet | Available in OneNote | Reusing libraries and modules. Efficiency - employability |
| Wk 10 |  | 2.3 File handling | Files 3 – File formats (workshop/lecture 2)  Complete all tasks from worksheet 30.  In later lectures we will cover javascript, and in year two the students will learn javascript in websites. JSON is a good introduction to both of these topics. It is important that the students are comfortable with this technology. | Worksheet 30 – exporting data in different file formats, csv, json and pickle.  Implement CRUD, allow user to select file output format. E.g. “Save as” | Formative feedback on worksheet | Available in OneNote | Reusing libraries and modules. Efficiency - employability |
|  |  |  | Exceptions. How errors can be caught, different types of errors. Teach the following new keywords:   * Try * Except * Else * Finally   A small introduction to classes. How exceptions bubble up through the code. |  |  |  |  |
| Wk 10 |  | 2.4.6 searching and sorting algorithms | Bubble sort  Min, max (first / last item) | Worksheet  Workshop – modelling as students find this hard.  Ensure swap, iteration, loops etc. | Formative feedback on worksheet | Available in OneNote | Understanding the trade offs in algorithm design and choice – employability. |
| Wk 11 |  | 2.4.6 Searching and sorting algorithms | TODO – where is this lecture?  Bubble sort  Ascending, descending, advanced coding techniques, lambdas, measure speed, investigate other sorting algorithms.  Algorithm is built on the whiteboard. Starting with the inner loop, the students are asked how to “semi sort” a set of numbers, then how to keep doing that until they are completely sorted.  Questioning to gauge student understanding. | Workshop and investigation.  Model:   * Calling function * Assigning function to variable/parameter * Lambda / anonymous functions | Formative feedback on code, offer advice on how different algorithms work, speed vs memory consumption | Available in OneNote | Understanding the trade offs in algorithm design and choice – employability. |
| Wk 11 |  | 2.4.6 Searching and sorting algorithms | TODO – where is this lecture?  Investigate other sorting algorithms.  Look at Big-O notation for a couple.  Do a presentation on how bubble sort works and how a second sorting algorithm works. | Students to do presentation. Class discussion. | Verbal feedback. Questioning to ascertain knowledge | Available in OneNote | Presentations, public speaking - employability |
| Wk 11 |  | 6.2 Data formats  6.1 Javascript | Minecraft – Introduction to javascript.  Reason: So far we can concentrated on the basics of coding. Now we introduce another coding language, javascript. This has the more classic ‘c’ style of code. The exam and the workplacements will require the students to understand multiple languages. Using Minecraft is a fun way to learn, **compare** and **contrast** Python, Javascript and Block Code. Thinking of space as 3D, voxels, X, Y, Z. Will require a greater understanding of loops, maths and geometry. Students are familiar with Minecraft which makes getting the lectures started extremely fast. Visualising the output of an algorithm as the robot moves and places blocks gives a deep insight into what the code is doing, rather than just the completed output a student would normally see. Naomi justification – looks good!  Get familiar with coding in Minecraft education.  Lecturer to model agent, moving agent, interacting with user.  Students to start the tutorial. | Student to create code so that the robot creates a “line” of blocks.  Then add a length argument. Test and evidence the code and results.  Ensure all code is commented and well formatted. | Formative feedback. Modelling and questioning. | Available in OneNote | Student led learning, investigation - employability |
| Wk 11 |  | 6.1 Javascript  2.6 Validation  2.8 Testing | Minecraft – hour of code. Students to complete the hour of code and become familiar with the three different ways of coding:   * Visual * Python * Js | All students will complete the hour of code.  Lecturer to spend time with students who are unfamiliar with ‘c’ style syntax. | Formative feedback. Modelling and questioning. | Available in OneNote | Student led learning, investigation - employability |
| Wk 12 |  | 6.1 Javascript  2.6 Validation  2.8 Testing | Minecraft: using the robot, make a wall.  Reason: This sequence of lectures introduces javascript (used later in the year) and covers flowcharts. The students are to automate a robot to build a structure in a 3D world. Once completed they will write a test plan and complete a test log.  Lecturer to model and discuss:   * Robot * Movement * Strategies to make it simpler * If statements * Anonymous functions / lambdas | All students to use a loop to create a sequence of blocks.  Some student will complete 4 walls and make the first layer of a house. | Formative feedback. Questioning and student to demonstrate progress. | Available in OneNote | 3D co-ordinates - maths |
|  |  | 1.2 Algorithms | Fun snowflake challenge (last lecture before Christmas) | Create a snowflake using Python turtle, functions and recursion. Students to show snowflakes at end of lecture. | Formative. Students to feedback on each others work. | Available in OneNote |  |
|  |  |  | CHRISTMAS ❄☃ |  |  |  |  |
| Jan 4th |  | 6.1 Javascript  2.6 Validation  2.8 Testing | Project: Uther Pendragon needs a castle  Lecture:   * Design documentation * User brief * Requirements spec * Project proposal   Model:  Minecraft – using 4 walls create a basic box. Allow user to select box size. | Lecturer to monitor progress  Students to evidence code and results (screenshots)  Students produce:   * Design documentation * User brief * Requirements spec * Project proposal | Formative feedback. Questioning and student to demonstrate progress. | Available in OneNote |  |
|  |  | 6.1 Javascript  2.6 Validation  2.8 Testing | Minecraft – crenelations. Class discussion of loops, finite state machines. Students to implement:   * Crenelations * Windows * Doors. | Students to complete the **Pitch Document**  Lecturer to monitor progress  Students to continue with project, evidence code and results (screenshots) | Formative feedback. Questioning and student to demonstrate progress. | Available in OneNote |  |
|  |  | 6.1 Javascript  2.6 Validation  2.8 Testing | Students present their pitch. Feedback given by lecturer and other students.  Minecraft – students to complete the castle accents:   * Crenelations * Windows * Doors.   Some students will implement:   * Moat | Lecturer to give feedback on pitches.  Students to continue with project, evidence code and results (screenshots) | Formative feedback. Questioning and student to demonstrate progress. | Available in OneNote |  |
|  |  | 2.8 Testing | Introduction to testing.  Why do we test software.  How do we document it.  Test plan.  Test log.  Complete testing document. Students to write a series of tests with expected outcomes for their castle application. | “mc code evidence” worksheet.  Students to write a test plan / test spec. Evidence via screenshots. Students to log any bug fixes and code changes. | Formative feedback. Questioning and student to demonstrate progress. | Available in OneNote | Jordan and BlueFruit – a Truro College apprentice who works as a software tester |
| 2022-01-07 |  | 2.8 Testing | Workshop  Students to carry out tests and evidence the actual outcomes and any differences with expected outcomes.  Fix bugs if necessary. | Students to complete a test log (referencing the test plan) | Formative feedback. Questioning and student to demonstrate progress. | Available in OneNote |  |
|  |  | 2.8 Testing | Workshop.  Students to write up the test specification and test log. Students to complete the testing document and hand in.  2022 – hand in on 7th Jan | Complete the testing document and hand in.  Students to try each others castle making applications. Question each other on how the algorithms work and decisions they made.   * What is the smallest and the largest castle that can be created? * How are errors trapped? * Is the user interface easy to use? | Documentation feedback – one-to-ones. | Available in OneNote |  |
| 2022-01 |  | 6.1 C#  6.2 UI / UX | Introduction to C#, demo visual studio. Students to follow along with hello world app, then a discussion of variables and data types.  Cover key difference between Python and C# syntax.  Lecturer to model Visual Studio – create new application.  Reason: The students have learnt Python and Javascript. C# has the same syntax as javascript, but it also allows them to build complete applications with a graphical user interface. | Students will complete a “Hello world app”.  Then a basic application that demonstrates maths. | Questioning. Observation of progress. | Available in OneNote | Employers and customers expect programs that are easy to use and well tested. |
| 2022-01 |  | 6.1 C#  6.2 UI / UX | C# selection. The ‘if’ statement, if, else, else if. Student to implement:   * number to word task * days of week task   Make use of **case** statement | Students to show lecturer a demo of the tools:   * Demo an app * Debug * Inspect variables * If and else statement | Formative. Discussion and feedback from other students. | Available in OneNote | Employers and customers expect programs that are easy to use and well tested. |
| 2022-01 |  | 6.1 C#  6.2 UI / UX | C# loops and arrays. Discussion of different data types in C#. Comparison to Python.  How to create an array of integers and how to loop over it. | User to enter 4 values, store these in an array. Loop over the array and calculate the total. |  | Available in OneNote | Employers and customers expect programs that are easy to use and well tested. |
| 2022-01 |  | 6.1 C#  6.2 UI / UX | C# classes and class member variables. Lecturer to model class, class methods and class member variables.  Discussion of why these are used. | Students to copy code from worksheet. Create an incrementor class.  Extend these ideas and implement a dice rolling application (discuss D&D and other dice board games) |  | Available in OneNote | Employers and customers expect programs that are easy to use and well tested. |
| 2022-01 |  | 6.1 C#  6.2 UI / UX | C# Dice application.  This sequence of lectures is designed to push students to write longer programs, get experience with large code base, to carry on a project over a number of lectures. To answer the following:   * How to structure the code * How to name functions and methods so the code is readable * How to test (white box and black box) sections * How to test user interface and user experience * How to go “above and beyond” customer expectations.   (2022 – students did cookie clicker rather than Dice) | Workshop to work on the dice application.  Extension: add a second dice. Show when a double has been rolled. | Formative feedback during lecture/workshop. | Available in OneNote | Creating a windows application that uses a GUI |
| 2022-01 |  | 6.1 C#  6.2 UI / UX | C# Dice application. Start to think about user interface, images, fonts, colours. | Workshop to work on the dice application.  Extension:   * add a second dice. Show when a double has been rolled. * Implement other sided dice, e.g. an 8 sided die. | Formative feedback during lecture/workshop. | Available in OneNote | Creating a windows application that uses a GUI |
| 2022-01 |  | 6.1 C#  6.2 UI / UX | C# Dice application. Complete the application. Demo application to lecturer, discussion of code implementation. Talk lecturer through algorithms used. | Complete application and test. | Lecturer to demo dice app | Available in OneNote | Creating a windows application that uses a GUI |
| 2022-01 |  | 6.1 C#  6.2 UI / UX | C# CSV files – testing documentation  Refer back to Python CSV file manipulation.  Lecturer to go over the worksheet, the research links in it. Discussion of how Python implemented .csv files. | Students to complete worksheet. Create a ‘record browser’ to view records saved in a .csv file. | Lecturer to monitor record browsing application progress. | Available in OneNote | Problem solving and coding skills. Meeting a clients requirement spec. |
| 2022-01 |  | 6.1 C#  6.2 UI / UX  2.3 File handling | CSV file workshop x 4 (1)  NAOMI – remove all this and make cookie clicker ? Lets talk about this one again. | Students to complete record browsing application. | Lecturer to monitor record browsing application progress. | Available in OneNote | Problem solving and coding skills. Meeting a clients requirement spec. |
| 2022-01 |  | 6.1 C#  6.2 UI / UX  2.3 File handling | CSV file workshop x 4 (2) | Students to complete record browsing application. | Lecturer to monitor record browsing application progress. | Available in OneNote | Problem solving and coding skills. Meeting a clients requirement spec. |
| 2022-01 |  | 6.1 C#  6.2 UI / UX  2.3 File handling | CSV file workshop x 4 (3) | Students to complete record browsing application. | Lecturer to monitor record browsing application progress. | Available in OneNote | Problem solving and coding skills. Meeting a clients requirement spec. |
| 2022-01 |  | 6.1 C#  6.2 UI / UX  2.3 File handling | CSV file workshop x 4 (4)  Students to write test plan and carry out tests. Evidence should be captured using screenshots. | Students to complete record browsing application. | Test document – write test plan and test log | Available in OneNote | Problem solving and coding skills. Meeting a clients requirement spec. |
| 2022-01 |  | 2.8 Testing – five whys | Problem solving and root cause analysis.   * Logic errors * Five whys * Data errors * Root cause analysis   Lecturer to tie bug fixing back to C# application | Apply root cause analysis to bugs found in C# application.  Implement “add a new record”  Students to write a report on:   * Logic errors * Five whys * Data errors * Root cause analysis | Finish off Test document | Available in OneNote | Logical thinking. |
| 2022-01 |  | 3.2 Emerging trends and technologies | Emerging issues. Students to research and write a presentation on emerging IT issues. E.g. covid, working from home, test and trace.  Class discussion of what they know, what they are concerned about.  Stop class at around 2/3rds lecture and get an update on what sort of things they are looking at. Give feedback. Stop anyone who is doing non-IT issues (e.g. ocean pollution) | Students to do research, keep notes and then create a presentation.  Cut the presentation down so it is simple to follow.  Lecturer to show how to have a “notes” section in the presentation. | In class discussion | Available in OneNote | Presenting in front of a group |
|  |  | 3.2 Emerging trends and technologies | Workshop: students to complete presentations, then present them. Approx. 7-8 mins per presentation (5 slides) | Students to present their research.   * Feedback given by lecturer * Class discussion of each technology * Class feedback on presentation | In class discussion |  |  |
|  |  | 2.8 Testing | Testing – the bigger picture 1.  Go over:   * Root cause analysis * Polya * Data errors * Bug – five whys   Again to remind the students. Cover each topic in more depth. Students to take notes. | Previously have notes on how to apply the five whys to bugs you had in C# application.  Expand notes, what we covered today:   * Boundary testing * Unit tests * Other approaches to Root Cause Analysis | One-to-one with students | Available in OneNote |  |
|  |  | 2.8 Testing | Testing – the bigger picture 2 . Quality assurance.   * Concept * Boundary * Unit testing * integration | TODO – dream up a better task:  Implement a unit testing class to C# application.  Lecturer to model how.  Class / library re-use.  ---- maybe move this up into Python modules / libraries, then pull that code in here into the student notes? Expand the code to do boundary testing?  Naomi? Mini project? | Formative – lecturer to check unit tests are working | Available in OneNote | Boundary testing - maths |
|  |  | 2.8 Testing | Testing – the bigger picture 3   * Functional testing * Non-functional testing * Hardware * Data testing | You have been asked by your company to write training materials explaining what testing is and why it is important to new staff at the company.  Create some training materials for teaching new hires in an engaging way. |  | Available in OneNote |  |
|  |  | 1.2 Algorithms | Binary numbers.  Teach the students binary numbers, how to count, how to do simple maths. | Students to complete the fun binary tasks (online game)  Lecturer to model the game, then student compete to get high scores, who can get to 100 fastest, etc. |  | Available in OneNote | Number systems - maths |
|  |  |  | Number systems:   * Binary * Denary * HEX * Oct | Students to complete the “Binary and HEX worksheet”. |  |  |  |
|  |  | 3.1 Moral and ethical issues | Moral and ethical issues. Students will choose two (or three) of the following topics. Then write a presentation:   * acceptable use * autonomous operation * changes in societal norms and the behaviour of individuals * changes in the culture within an organisation * environmental issues * Globalisation * inclusion and diversity * monitoring of employees * open source and Creative Commons * the collection and use of data * unequal access to technology and/or digital services.   Lecturer will ensure at least two students cover the same topic so there are two points of view of each. | Students to write presentations. |  | Available in OneNote | Understanding, judge, evaluate – English  Learners will develop the following competencies:  Convey technical information to different audiences.  Present information and ideas  Summarise information/ideas |
|  |  | 3.1 Moral and ethical issues | Presentations – each student to present research. | Presentations. In class discussions. Feedback on presentation. |  | Available in OneNote |  |
|  |  | 3.1 Moral and ethical issues | Moral and ethical issues 2. Lecture that covers the above points. Ensure no gaps in knowledge. | In class discussion. Cover any points missed in student presentations. | Formative assessment takes place throughout the session. | Available in OneNote |  |
|  |  | 1.2 Algorithms | Searching algorithms.   * Linear * Binary   What the benefits of both are, drawbacks. When to use them.  How to implement a binary search. How this ties into sorting algorithms (binary search only works on sorted data)  Discussion on databases and how they find data quickly. Why this is important. | Students to create poster on liner vs binary search.  Lecturer will check posters. If students agree, the posters will be printed out and put on the wall as a revision aid. | Formative assessment of posters. | Available in OneNote | Learners will develop their research skills, and present their findings in a clear, logical and cohesive way. |
|  |  | 1.2 Algorithms | Sorting algorithms. Refer back to the Python bubble sort.   * Relational * NoSQL * Graph | Mini research task:   * Relational * NoSQL * Graph   Students to each talk about findings. In class discussion. Students and lecturer to probe learning. | Formative. Discussion. | Available in OneNote | Learners will develop their research skills, and present their findings in a clear, logical and cohesive way. |
|  |  | 1.2 Algorithms | Sorting algorithms – further research. | “53 - Searching and Sorting Worksheet”  Students to implement all three sorting algorithms in Python  1. Source code for Bubble Sort  2. Source code for Insertion Sort  3. Source code for Merge Sort | Formative. Discussion.  Swapped “This is an extremely hard worksheet.”  For  Learners are set a worksheet including some complex tasks, designed to stretch and challenge. | Available in OneNote | Learners will develop effective employable skills such as, self management, perseverance when working with complex task, troublehooting, and working independently. |
|  |  | 1.2 Algorithms | Workshop:  Carry on with the searching and sorting worksheet. | “53 - Searching and Sorting Worksheet”  Students to implement all three sorting algorithms in Python |  | Available in OneNote |  |
|  |  |  | Workshop:  Mock exam revision. |  |  |  |  |
|  |  |  | Mock exam |  |  |  |  |
|  |  |  | Workshop. Cover exam, go over questions and answers. Answer any queries the students have. Show good exam technique and how to answer 1, 2 and 3 point questions.  1 – define  2 – define, explain (expand)  3 – define, expand, example/argue/analyse |  |  |  |  |
|  |  | 4.2 Guidelines and codes of conduct | RICHARD?  Guidelines from Professional Organisations  Internal Policy Documents  Whistle Blower  Company Culture |  |  | Available in OneNote |  |
|  |  | 4.2 Guidelines and codes of conduct | Complete presentations on ethics. Guidelines from professional bodies. BCS. Web accessibility. ICO. UK Government legislation. Policy documents.  Whistle blowers.  Company culture. | Create a presentation on:   * Guidelines from Professional Organisations * Internal Policy Documents * Whistle Blower * Company Culture   Students to present next lecture. |  | Available in OneNote |  |
|  |  | 4.2 Guidelines and codes of conduct | Student presentations on ethics. | Students to present research from last lecture. |  |  |  |
|  |  | 4.1 Legislation | 2021 – Naomi and Richard did this  Are you just recapping or do you want to take it out?  Legislation and regulatory requirements. Computers can be used in unlawful ways.  Risk assessment.  GDPR | Investigate a Data Protection breach  List the issues and then give some examples of things they could have done to have not breached  What happened  Could it have been avoided?  Could it have been mitigated?  What needs to be done now to fix?  What changes should be made in the future? |  |  |  |
|  |  | 4.1 Legislation | 2021 – Naomi and Richard did this  Are you just recapping or do you want to take it out?  Computer misuse act. Hacking, viruses, data theft, phone hacking.  Cover the three offences.  Section 1: Unauthorised access to computer material Section 2: Unauthorised access with intent to commit or facilitate commission of further offences Section 3: Unauthorised Acts with intent to impair, or with recklessness as to impairing the operation of a computer | What are the three sections of the computer misuse act? |  |  |  |
|  |  |  | TODO –  move C# here  OR  start HTML and Python OOP  OR  Advance Python Techniques. |  |  |  |  |
| 14th march |  |  | Advanced Python Libraries – Pandas  Data analysis. Who uses it and why. Cover basics:   * Series * Dataframe * Row / Column   Ensure all students have the padas library correctly installed and can access the tutorials. | Students to complete the W3SCHOOLS pandas tutorial. Student led investigation. |  |  |  |
|  |  |  | Workshop: Pandas continued (1 of 3 workshops) | Students to complete the W3SCHOOLS pandas tutorial. Student led investigation. |  |  |  |
|  |  |  | Workshop: Pandas continued (2 of 3 workshops)  Some students will have finished the W3SCHOOLS tutorial. They will start the “weather data task” worksheet. | Students to complete the W3SCHOOLS pandas tutorial. Student led investigation.  “weather data task” worksheet |  |  |  |
|  |  |  | Workshop: Pandas continued (3 of 3 workshops)  All students to work on the weather data worksheet. | “weather data task” worksheet |  |  |  |
| 23rd march |  |  | Advanced Python Libraries – Tkinter  Tkinter - 1 x lectures, 3 x workshops  Cover:   * What is TCL/TK * Cross platform * Events * Window loop * 3 x Geometry managers   In class discussion of how this is different to the C# layout the students did after Christmas.  Either calculator or cookie clicker | Students to complete in lecture tasks   1. Geeksforgeeks tutorial 2. Calculator 3. Weight conversion calculator |  |  |  |
|  |  |  | Workshop: Tkinter continued (1 of 3 workshops) | 1. Geeksforgeeks tutorial 2. Calculator 3. Weight conversion calculator |  |  |  |
|  |  |  | Workshop: Tkinter continued (2 of 3 workshops) | 1. Geeksforgeeks tutorial 2. Calculator 3. Weight conversion calculator |  |  |  |
|  |  |  | Workshop: Tkinter continued (3 of 3 workshops) | 1. Geeksforgeeks tutorial 2. Calculator 3. Weight conversion calculator |  |  |  |
|  |  |  | Advanced Python Libraries – NumPy and matplotlib  Covers:   * What is NumPy and how it relates to pandas * What is matplotlib – displays graphs and charts * How to use NumPy to display the data on a graph. * Labels, axis, colours.   NumPy - 1 x lectures, 2 x workshops | Create a scatter plot (in lecture task)  W3SCHOOLS numpy tutorial |  |  |  |
|  |  |  | Workshop: NumPy continued (1 of 2 workshops) | W3SCHOOLS numpy tutorial |  |  |  |
|  |  |  | Workshop: NumPy continued (2 of 2 workshops) | W3SCHOOLS numpy tutorial |  |  |  |
|  |  |  | Advanced Python Libraries – TensorFlow  Approx. 1 lecture, 2 workshops.  Covers:   * How the human brain works * Neurons * Mathematical model of a neuron * Activation shapes * Coding the model * Updating the weights and biases * Installing the libraries   Tried to get the smiley sorter working – too hard.  Maybe next year just generate 1,00 smileys and frowneys per students? | 1. TensoFlow website – quick start tutorial 2. TensorFlow tutorial – the keras tasks 3. Udemy deep learning with NumPy tutorials |  |  |  |
|  |  |  | Workshop: TensorFlow continued (1 of 2 workshops) | 1. TensoFlow website – quick start tutorial 2. TensorFlow tutorial – the keras tasks   Udemy deep learning with NumPy tutorials |  |  |  |
|  |  |  | Workshop: TensorFlow continued (1 of 2 workshops) | 1. TensoFlow website – quick start tutorial 2. TensorFlow tutorial – the keras tasks   Udemy deep learning with NumPy tutorials |  |  |  |
| 5th april |  |  | Go through the revision Mock 0 test with students. Cover answers. |  |  |  |  |
| 6th April |  |  | Make Revision Notes on:  Legislation |  |  |  |  |
|  |  |  | Make Revision Notes on:  Testing |  |  |  |  |
|  |  |  | EASTER |  |  |  |  |
|  |  |  | Revision |  |  |  |  |
| 28th |  |  | Mock 1 |  |  |  |  |
| 3rd May |  |  | Intro to HTML |  |  |  |  |
|  |  |  | Set up github  Move projects into github |  |  |  |  |
|  |  |  | Revision or HTML |  |  |  |  |
| Thur 5th may |  |  | Mock 2 |  |  |  |  |
| 9th may |  |  | revision |  |  |  |  |
| 12th may |  |  | 2021 Nov exam used as a mock |  |  |  |  |
| 16th may |  |  | GCHQ day. Cyber security |  |  |  |  |
| 18th may |  |  | Peer mark the mock exam. Discussion of answers, students to make notes on answers. |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |