Think, Compute, Grow!

A strategy for teaching computing and ICT through sustainable agriculture

PART 6- LESSON PLANS – ADVANCED STAGE

Licence and permissions:

You are free to:

Created by Owen Dobbing, 2026

CONTENTS

Developmental goals

The lessons in this strategy all aim to further children's knowledge and understanding in the **nine development goals**. These goals aim to prepare children to use technology to succeed in their community and in the wider, digital world.

Foundational academic skills	**Goal 1**	**Goal 2**	**Goal 3**
	Foundational academic excellence	**Functional and professional English**	**Geographic information and mapping**
Understanding and using technology	**Goal 4**	**Goal 5**	**Goal 6**
	Introduction to software logic	**Understanding computer systems and networks**	**Digital literacy, E-safety and AI**
Applying technology to agriculture	**Goal 7**	**Goal 8**	**Goal 9**
	Agricultural financial literacy	**Agricultural science**	**Modern marketing and business**

Selecting lesson plans

The lessons plans in this strategy cover all nine developmental goals.

It is recommended that teachers **do not teach all of the lessons sequentially** but instead, select and use lessons to use as they see fit. Teachers may decide to spend two lessons teaching a lesson from this strategy. They may decide to repeat lessons within or in other years.

At the end of each group of lessons, an **objective coverage grid** highlights the skills that the lessons have covered by teaching each development goal.

THREE STAGES

Teaching is split into three stages of development These are:

Stage 1: Foundation stage (ages 7 – 9)

Description: Students become familiar with operating laptops. They enjoy being creative, solving problems and discovering familiar resources for research and learning.

Focus: Core computer problem solving, English language discovery, basic nature observation. Growing understanding of digital literacy.

Tools: Typing.com, E-Paath (grades 1 – 3), Scratch desktop (animation), Google Earth.

Stage 2: Intermediate stage (ages 10 – 12)

Description: Students grow in confidence at solving problems using technology. They make connections to their local environment and the technology they use. Children use digital tools to support their knowledge across subjects.

Focus: Structured document and presentation creation. Independent programming and problem solving using logical structures. Developing digital literacy.

Tools: Typing.com (focus on speed), E-Paath (grades 4 – 6), Scratch, Makecode, (virtual blocks), LibreOffice Writer.

Stage 3: Advanced stage (ages 13 – 16)

Description: Students apply their knowledge of hardware, software and online services to solve increasingly complex problems related to their environment.

Focus: Advanced programming logic (programming with Python), building automation hardware (Microbit), organising finances using spreadsheets, advanced word-processing and presentational skills. Digital and AI literacy.

Tools: LibreOffice Calc (spreadsheets), Python programming (text-based programming), Microbits (programming using Microbit Python).

YEARLY PLANNING

This is an example of how a teacher might select and teach lessons from this strategy across one academic year.

Week	Strategy lesson	Lesson focus
1	**Introduction lesson**	Introduction to the year. Agreeing ICT acceptable use. Lesson routines. Touch typing practice.
2	**Lesson 2**	Formatting a document
3	**Lesson 3**	Touch typing
4	**Lesson 4**	Maths skills
5	**Lesson 32**	Digital health and wellbeing
6	**Lesson 5**	Using Google Translate
7	**Lesson 6**	Google Translate Speech to Text
8	**Lesson 7**	Google Translate Conversation Mode
9	**Lesson 11**	Google Earth terrain measurement
10	**Lesson 12**	Calculating distance and area using Google Earth
11	**Lesson 13**	Introduction to Python, print and input functions
12	**Lesson 31**	Copyright and ownership
13	**Lesson 15**	Programming conditions in Python (basic)
14	**Lesson 17**	Creating while True loops in Python (basic)
15	**Lesson 19**	Functions and parameters in Python (basic)
16	**Lesson 20**	Functions and parameters in Python (intermediate)
17	**Lesson 21**	Functions and parameters in Python (advanced)
18	**Lesson 22**	Lists in Python (basic)
19	**Lesson 23**	Lists in Python (intermediate)
20	**Lesson 24**	Introduction to Microbit Python
21	**Lesson 25**	Using conditions with a Microbit sensor
22	**Lesson 33**	Identifying misinformation

This example teacher has selected lessons from a variety of topics, giving children enough time to explore topics in depth while also covering a variety of content.

E-safety topics are taught throughout the year, both embedded into lessons through frequent discussion and in dedicated E-safety lessons.

ONLINE RESOURCES

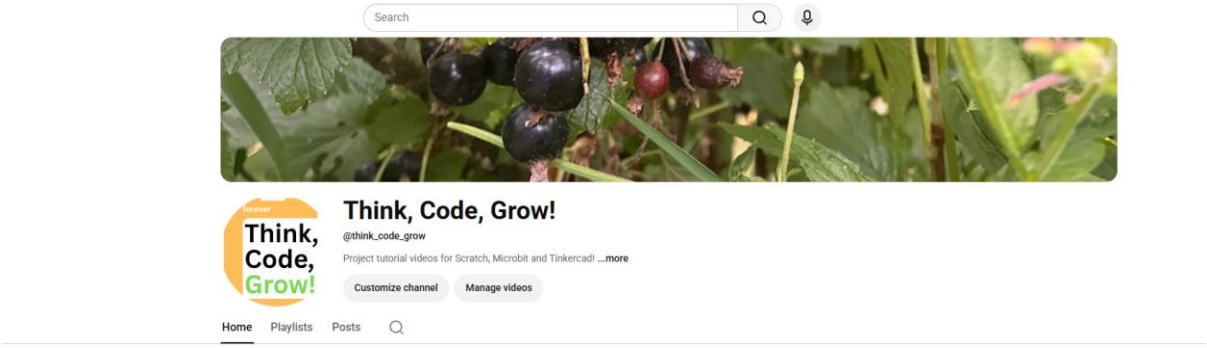
To support teachers and students in delivering and learning with his strategy, a bank of online resources is available.

YouTube videos

The YouTube channel, 'Think, Code, Grow!' features video tutorials for projects created using **Scratch**, **MakeCode** (for Microbit) and **Python** (for Microbit and onscreen using Mu).

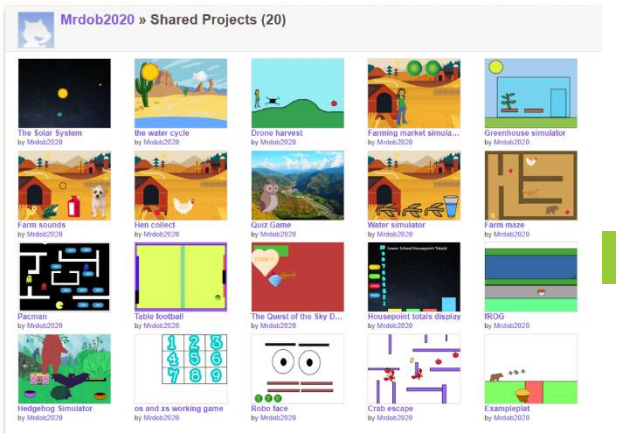
Teachers can use these video tutorials as part of their teaching to the class, playing and pausing videos to discuss them.

Teachers can also make the videos available to students, so that they can learn at their own pace. It is recommended that children watch a video the whole way through once, before using it to help them work on a project.



Scratch project examples

The **Scratch channel**, MrDob2020, contains examples of all of the Scratch projects featured in this strategy. These examples are useful for teachers to show students before they embark on projects.



Python is a text-based programming language. It is commonly taught to children in schools and is also used extensively for software development in industry.

Children are usually introduced to Python when they are familiar with principles of programming taught initially through block-based programming. By the time children start programming using Python, they should be familiar with concepts including **sequence of code**, **repetition** (loops), **selection** (conditions**)**, use of **variables**, **lists** and **functions**.

Effective teaching of Python recognises that children are learning a new language. As such, many of the same principles of teaching children's native, or an additional language apply.

Children should be given lots of opportunities to **read code written in Python.** Just like reading in any language, they should be encouraged to read whole pieces of code, even if they do not understand every section.

The teacher should lead the class in discussions about the **syntax** that Python uses. Early on, children should notice capitalisation, use of commas to separate strings from numbers, layers of indentation, colons after conditional statements and conventions for naming variables.

Children should be given plenty of opportunities to make predictions about what they think code will do. This should include them **making predictions** about where code contains bugs through being presented with code containing errors. As children's understanding develops, they will start to differentiate between **syntax errors** (i.e. a missing bracket) and **logical errors** (correct syntax but incorrect sequence or structure).

Just like when writing in their own language, children can build up their confidence by finishing unfinished pieces of code, rearranging code in the wrong order, or even copying and modifying code that the teacher gives them. At all stages, it is important that children type their own code,

although they should get used to practices like copying and pasting sections of code where they spot repetition.

Example section of code

Let's take the learning objective of teaching children to use a conditional structure to control an output. This is the content of lesson 15 in this strategy.

Here is the standard of code this lesson aims to teach:

```
temp = float(input("Enter today's soil temperature in degrees. "))
if temp < 10:
    print ("Soil temperature is too low! Delay planting! ")
else:
    print ("Soil temperature is ideal. It is safe to plant crops! ")
```

The key aspect of this code is the **branching, conditional statement**, using the **'if'**, **'else'** structure. This would be explored through diagrams prior to showing children the code.

Children would then be encouraged to notice the syntax of a colon after each conditional statement and an indent (using the tab key) after the colon.

The teacher could spend plenty of time covering these things, giving children examples of code containing errors to help build up their familiarity before they wrote their own code.

There are other aspects of this code which could cause confusion, including setting the **'temp' variable** to the **float** of the text inputted after the initial prompt. This is a separate teaching point, but the teacher should be prepared to give a short explanation in this lesson.

Once students have had time to modify the above code, they should be prompted to try to write the code themselves, run it and use the on-screen debugging messages. These tell students which line contains errors and what kinds of errors they are.

Finally, students can attempt to add to the code themselves or create entirely new programs using the taught content. A natural progression could be students asking if they can include **more than two conditions**, which they can be shown how to do using **'elif':**

```
temp = float(input("Enter today's soil temperature in degrees. "))
if temp < 10:
    print ("Soil temperature is too low! Delay planting! ")
elif temp <15:
    print ("Soil temperature is low. Plant and monitor! ")
else:
    print ("Soil temperature is ideal. It is safe to plant crops! ")
```

Effective teaching

Effective teaching of Python makes links to language teaching. It gives students plenty of exposure and celebrates errors as learning points.

Dr Sue Sentance, Chief Learning Officer of the Raspberry Pi Foundation, developed the **'PRIMM' approach**. PRIMM stands for 'Predict', 'Run', 'Investigate', 'Modify, 'Make. This helps teachers to structure he sequence of learning of programming languages. PRIMM is an effective way of structuring one lesson, or a series of lessons where children are learning about concepts in coding.

Above all, children should not be rushed into more and more difficult code and should recognise that success in using Python comes with **repetition** and **familiarity**.

ADVANCED-STAGE ASSESSMENT

Assessment Sheet – Advanced Stage			
Student name: **Date of assessment:** *Best fit assessment – for each of the six sections, choose the closest descriptor*			
	Working towards expectations	**Achieving expectations**	**Exceeding expectations**
ICT Skills			
Understanding and controlling computers	Types using more than one finger, with growing accuracy. Uses bookmarks and files to organise work with growing independence.	Types accurately, at or close to 30 wpm using both hands. Uses file hierarchy and understands file extensions.	Takes on a leadership role in setting up and using computers and peripheral devices. Types accurately at a minimum of 30wpm with two hands.
Creative and effective digital expression	Uses document and spreadsheet software to convey ideas, creating documents that are clear but lacking in detail and design elements. Uses basic formulas in spreadsheets.	Creates multi-page documents and presentations, showing awareness of needs of a reader. Uses a range of formulas effectively in a spreadsheet.	Combines media with data and charts from spreadsheets to create clear and effective presentations. Understands and uses principles of graphics design.
Digital literacy, AI literacy and E-safety			
Digital citizenship and online safety	Uses reliable websites to source information and follow copyright rules. Distinguishes between kind and unkind online behaviour.	Contrasts open-source and copyright material. Uses technology safely and healthily, taking measures to prevent and report cyberbullying.	Leads others in using technology safely and healthily. Takes steps to prevent cyberbullying and demonstrates digital citizenship.
Media literacy, AI literacy and ethics	Recognises uses of AI in society, including in search results. Gives examples of issues associated with AI including reliability, bias and safety.	Gives examples of impact of AI in education and industry. Critically evaluates AI content for bias and misinformation. Explains algorithms behind machine learning.	Explains ways in which AI tools can be used for positive effects, locally and in society. Cross references sources on information.
Programming and computational thinking			
Programming logic and computational thinking	Uses an IDE to write code in Python, with mostly accurate syntax. Uses the print and input functions. Beginning to use loops and conditions.	Types code in Python, with accurate syntax. Uses variables, loops and conditions in Python. Debugs independently using onscreen prompts.	Organises code in Python using functions with parameters. Uses a range of Boolean logic (not, or). Controls sensors and outputs on devices.
Understanding computer systems and networks	Names some of the components of a network and beginning to talk about how they function.	Connects devices to a school network. Describes how data travels over a network.	Explains the benefits of different network topologies in relation to surroundings.

Introduction lessons

INTRODUCTION TO COMPUTING THIS YEAR

Lesson 1 – Introduction, rules and routines	
Goal	**Introduction to computing this year (rules and routines)**
Learning objective	We are learning the routines and rules for computing lessons this year
UN Sustainable Development Goal	 **SDG 4: Quality Education (Target 4.4: …Substantially increase the number of youth who have relevant skills…)** Learning to use computers safely and effectively
Success Criteria	• I can explain the routines for the start of the lesson • I can explain and agree to follow the ICT acceptable use rules • I can set up my folders on the computer for storing information
Introduction	1. Gather the children in the classroom and introduce yourself and the subject of computing. Give children an overview of some of the projects they will do this year. This could include showing them the YouTube channel or Scratch project page. 2. Explain what you would like children to do when they come into computer lessons (sit in assigned places, log on, open the touch-typing website, possible open last piece of work). 3. Read and discuss the ICT acceptable use agreement with children. Ask all children to sign a paper copy of this. Ensure this agreement is displayed clearly in the room. Explain that we have ICT rules to keep use and other people safe. 4. Explain that as this is thew first lesson of the year, children should focus on setting up and naming their folders and writing a short document explaining their aims and interest for the year.

Main lesson	**Step 1: Logging on (5 minutes)** • Instruct the children to log onto their computers. Ask them to double click on the Chrome shortcut. **Step 2: Accessing typing.com (15 minutes)** • Show children how to navigate to Typing.com by typing this URL in. Show children how to save this as a bookmark in Chrome. • Encourage children to practise their typing frequently, including at the start of every computing lesson. **Step 3: Opening and working on a document (15 minutes)** • Show students how to open Word and create a new document. • Show an example of a teacher-created document called **'About Me'**. This should include some basic sentences about interests and an introduction. Children should also write about their aims and interests for the year. • Students create their own documents to introduce themselves. Depending on ability, this can include adding bold effects, titles, changing size and colour of text, and even adding an image form the internet. • Encourage children to format the document clearly, using headings and bullet points. • Students can also copy and paste an image form Pixabay.com to accompany their document. **Step 4: Setting up folders and saving (5 minutes)** • Show students how to name and save their documents and which folder they should save work into. • Students name and save their work and save it into a folder.

	• Students can also create sub folders, i.e. for programming, graphics design.
Plenary	**Time: 10 minutes** 1. Celebrate students' success this lesson and recap rules and procedures. 2. If possible, the teacher can open some of the children's work and offer feedback. 3. Students log off their computers and follow class exit procedure.

Lesson 2 – Formatting a document	
Goal	**Introduction to computing this year (opening work)**
Learning objective	We are learning format a piece of work to create a clear and professional looking document
UN Sustainable Development Goal	 **SDG 4: Quality Education (Target 4.4: …Substantially increase the number of youth who have relevant skills…)** Learning to use computers safely and effectively
Success Criteria	• I can open a document that I saved in a previous lesson • I can add titles to a document • I can add a page of contents • I can change the style of the overall document • I can use bullet points and indentation • I can format an image and use textboxes to add captions
Introduction	1. Show the children an example of a professional-looking document about themselves. Ask the class when we might need to write a document like this – make the connection between documents like a CV. 2. Point out some of the features that make this document professional-looking. Include discussion of headings, contents page, page numbers, formatting, overall page design, borders. Write these things on the board as success criteria for the children's work. 3. Explain that in this lesson, the children will be opening and continuing their **'About Me'** documents from last lesson.

Main lesson	**Step 1: Opening documents (5 minutes)** • Instruct the children to log onto their computers. Ask them to navigate to where they saved their documents form last lesson and to open them. **Step 2: Formatting documents (25 minutes)** • Direct the children to work on their documents, adding the features to them that we discussed at the start of the lesson. • Remind children that they can navigate to Pixabay.com and use any of the non-sponsored images there in their document. **Step 3: Saving a s a Word document and a PDF file (5 minutes)** • Show students how to save their documents as both Word files and as PDF files. • Discuss why PDF files are useful ways to save documents (they are non-editable).
Plenary	**Time: 10 minutes** 1. Ask the students to reflect upon the documents they have created and the word processing skills they have learned. 2. Explain that next lesson, we will be developing our typing skills which should make future word processing even more effective.

Lesson 3 – Typing practice	
Goal	**Introduction to computing this year (touch typing)**
Learning objective	We are learning to type using all our fingers from a resting position on the home row.
UN Sustainable Development Goal	 **SDG 4: Quality Education (Target 4.4: …Substantially increase the number of youth who have relevant skills…)** Learning to use computers safely and effectively
Success Criteria	• I can open the Chrome browser and navigate to the shortcut for Typing.com • I can position my fingers correctly on the F and J keys • I can use the Shift key while I am typing to type special characters • I can assess my own level in Typing.com and choose the appropriate level to improve my accuracy and speed
Introduction	1. Ask the students what they understand by 'touch typing'. Explain that touch typing is being able to type at the speed of thought, without looking down at the keyboard. 2. Guide the students around Typing.com and explain that it can help develop typing at all levels, from beginner to advanced typing practise using paragraphs. 3. Explain that we are aiming to be able to type with all fingers, accurately at 30wpm, including the special keys accessed by using 'shift'.
Main lesson	**Time: 35 minutes** **Step 1: Software launch (5 minutes)**

	• Ask students to open the Typing.com website via the shortcut on the desktop. They could also access this through a bookmark. **Step 2: Resting posture (5 minutes)** • Guide children through the physical setup for their typing. Ask them to rest their left index finger on F and their right index finger on J. Explain that their hands should stay in these positions. Ask them to try reaching for other letters on this row with their other fingers. • Show the children how to use a cloth to cover their hands once they start typing to ensure they learn to touch type. **Step 3: Guided typing practise (15 minutes)** • As this is the first lesson of this unit, the children will spend the whole lesson practising their typing. In future lessons, they might only do 15 minutes typing at the start of the lesson. • Children self-assess their level and chose a section that will enable them to develop their typing skills. • Teacher role – the teacher should watch the children closely for signs that children are using their index fingers for all letters. Remind the children about good posture and looking at the screen at all times. If agreed, the teacher can cover children's hands with cloths.
Plenary	**Time: 10 minutes** Ask the children why they think it is important to be able to touch type. Celebrate the children's success; have them look at their Typing.com dashboards to view their statistics.

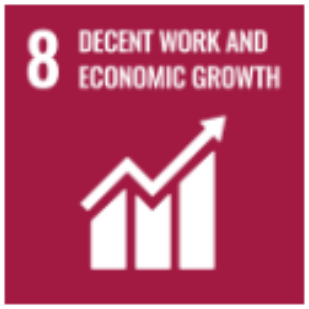
Objectives taught in this unit (highlighted)

Advanced level (age 13 – 16) – teaching objectives	
ICT Skills	**Lesson(s):**
1. Type accurately, at a speed of 30wpm, using both hands correctly	**3**
2. Master advanced file systems/: identify structural extensions (.py, .csv, .pdf, .zip) and manage local desktop vs. cloud directory.	**1**
3. Establish physical hardware connections using both Bluetooth and USB, connect a Microbit to another Microbit via radio	**28**
4. Create multi-page documents including custom page margins, table of contents, text alignment adjustments, page breaks.	**2**
5. Design agricultural data models in a spreadsheet using comparative operators and complex, conditional formulas, e.g. SUM, AVERAGE, COUNTIF.	**37, 38**
6. Deliver a high-impact presentation designed for a target audience using design tools and following conventions.	**9, 10**
7. Use principles of graphics design to create 2D images, i.e. alignment, contrast.	**40, 41, 42**
8. Use sophisticated, interactive online tools to research and critically evaluate information using this to further learning and investigations and create original content	**4, 5, 6, 7, 8, 11, 12, 39**
Digital literacy, AI literacy and E-safety	**Lesson(s):**
1. Explain specific legal frameworks governing technology use. Contrast open-source content with copyright infringement.	**1, 31**
2. Describe and utilise strategies to identify and counter the effects of screen-addition.	**32**
3. Analyse the effects of digital harassment, describing protocols to mitigate and report cyberbullying.	**32**
4. Cross reference primary sources to spot digital manipulation, deepfakes and automated disinformation campaigns.	**33**
5. Evaluate the local and global economic impacts of automation and artificial intelligence on rural labour markets and agricultural supply chains.	**35**
6. Outline the structural mechanics of machine learning, explaining concepts behind algorithms (i.e. decision trees and data pattern recognition).	**34, 35**
7. Critique the risks if modern generative models, covering training bias, data leaks, security risks and cognitive offloading.	**36**
8. Navigate the psychological hazards of anthropomorphised AI, explaining the risk of users forming inappropriate attachments to digital conversational agents.	**36**

Programming and computational thinking	Lesson(s):
1. Open an IDE (i.e. https://python.microbit.org/v/3/) and initialise a new script by importing the core hardware library (from microbit import *)	**13**
2. Use inputs (i.e. button_a.is_pressed()) and display output dynamically using the LED matrix (display.show()) and text scrolling	**24, 25, 26**
3. Use the built in functions in Python, including print(), input() and time.sleep()	**13, 14**
4. Read, process and convert analogue environmental inputs from alligator clip probes and onboard sensors (light, temperature, accelerometer)	**25, 26**
5. Use conditional structures (if, elif, else) and relational operators (<, =, >) to trigger outputs	**15, 16**
6. Use 'or', 'and', 'not' logic in conditional code and construct truth tables to map hardware decision making	**16**
7. Program text-based looping structures (while True: and for I in range()), use timing controls (sleep()) and break commands to manage iteration	**17, 18**
7. Notice examples of variables in text-based code. Define and call variables within a program	**14, 15**
8. Construct and manipulate lists to store sequential datasets (e.g. calculating daily soil temperature)	**22, 23**
9. Use functions to organise code logically, defining and using functions that accept parameters, execute instructions and return processed data values	**19, 20, 21**
10. Use the radio module to transmit and receive data packets between multiple Microbits	**28**
11. Use on-screen information to identify types and locations of bugs	**13 - 28**
12. Convert values between binary and decimal, explaining how sensor readings are stored in digital memory	
13. Apply standard searching (linear) and sorting techniques to process and organise datasets of sensor or spreadsheet data	**27**
14. Explain how information is sent over networks using packets and network topology.	**29, 30**

Goal 1

FOUNDATIONAL ACADEMIC EXCELLENCE

Lesson 4 – Understanding compound interest	
Goal	**Goal 1 – Foundational academic excellence (via E-Paath)**
Learning objective	We are learning to calculate compound interest, loan terms and authorisation for agribusiness start-up capital.
UN Sustainable Development Goal	 **SDG 8: Decent Work and Economic Growth (Target 8.3: Promote development-oriented policies that support productive activities)** Providing students with the skills to manage banking services
Success Criteria	• I can navigate to Class 8/9 mathematics section in E-Paath • I can solve dynamic compound interest equations under different compounding intervals • I can determine long-term cost differences between a high-interest loan and a low interest agricultural fund.
Introduction	**Time: 15 minutes** • Ask the students, "If you borrow 40,000 Rupees to build a greenhouse and the lender charges 10% interest, do you pay back 54,000 Rupees?" • Introduce compound interest. The answer is "No"! You pay back interest on the interest. This means your profits can become consumed by debts. • Introduce the lesson objective: Today in E-Paath, we will be mastering the mathematics of growth and debt so we can grow our business safely.
Main lesson	**Time: 35 minutes** **Step 1: Accessing E-Paath (5 mins)**

	• Ask the children to open E-Paath and navigate to Mathematics Grade 8/9 and open the Interest, Banking and Tax unit **Step 2: Compound Interest Simulator (15 mins)** • Children work through the interactive scenarios. The input different principal amounts, adjust interest rates and see how debt grows over 1, 3 and 5 years. • The teacher walks around the room and talks to the students about the difference between simple interest and compound interest. What happens to the gap between them? **Step 3: Calculating Amortisation (15 mins)** • Ask students to close E-Paath and open Libre Office calc (or Excel) • Ask students to type out a comparison table for a 100,000 NPR loan. • Option A – Cooperative – 8% simple interest, 2-year repayment • Option B – Micro - lender – 12% compound interest, 2-year repayment. • Challenge children to calculate the amounts using basic formulas. (The teacher can provide these)
Plenary	**Time: 15 minutes** 1. Bring the class together. Ask, "Why do commercial micro-finance institutions compound interest monthly instead of yearly?" (To maximise profits and increase the cost of the loan) 2. Reinforce links between this learning and the real world 3. Students save their sheets and log off their computers safely.

Objectives taught in this unit (highlighted)

Advanced level (age 13 – 16) – teaching objectives	
ICT Skills	**Lesson(s):**
1. Type accurately, at a speed of 30wpm, using both hands correctly	**3**
2. Master advanced file systems/: identify structural extensions (.py, .csv, .pdf, .zip) and manage local desktop vs. cloud directory.	**1**
3. Establish physical hardware connections using both Bluetooth and USB, connect a Microbit to another Microbit via radio	**28**
4. Create multi-page documents including custom page margins, table of contents, text alignment adjustments, page breaks.	**2**
5. Design agricultural and business spreadsheets that use comparative operators and conditional formulas, e.g. SUM, AVERAGE, COUNTIF.	**37, 38**
6. Deliver a high-impact presentation designed for a target audience using design tools and following conventions.	**9, 10**
7. Use principles of graphics design to create 2D images, i.e. alignment, contrast.	**40, 41, 42**
8. Use sophisticated, interactive online tools to research and critically evaluate information using this to further learning and investigations and create original content	**4, 5, 6, 7, 8, 11, 12, 39**
Digital literacy, AI literacy and E-safety	**Lesson(s):**
1. Explain specific legal frameworks governing technology use. Contrast open-source content with copyright infringement.	**1, 31**
2. Describe and utilise strategies to identify and counter the effects of screen-addition.	**32**
3. Analyse the effects of digital harassment, describing protocols to mitigate and report cyberbullying.	**32**
4. Cross reference primary sources to spot digital manipulation, deepfakes and automated disinformation campaigns.	**33**
5. Evaluate the local and global economic impacts of automation and artificial intelligence on rural labour markets and agricultural supply chains.	**35**
6. Outline the structural mechanics of machine learning, explaining concepts behind algorithms (i.e. decision trees and data pattern recognition).	**34, 35**
7. Critique the risks if modern generative models, covering training bias, data leaks, security risks and cognitive offloading.	**36**
8. Navigate the psychological hazards of anthropomorphised AI, explaining the risk of users forming inappropriate attachments to digital conversational agents.	**36**

Programming and computational thinking	Lesson(s):
1. Open an IDE (i.e. https://python.microbit.org/v/3/) and initialise a new script by importing the core hardware library (from microbit import *)	**13**
2. Use inputs (i.e. button_a.is_pressed()) and display output dynamically using the LED matrix (display.show()) and text scrolling	**24, 25, 26**
3. Use the built in serial REPL terminal to view live text outputs produced by print() statements for real-time debugging and data tracking.	**13, 14**
4. Read, process and convert analogue environmental inputs from alligator clip probes and onboard sensors (light, temperature, accelerometer)	**25, 26**
5. Use conditional structures (if, elif, else) and relational operators (<, =, >) to trigger outputs	**15, 16**
6. Use 'or', 'and', 'not' logic in conditional code and construct truth tables to map hardware decision making	**16**
7. Program text-based looping structures (while True: and for I in range()), use timing controls (sleep()) and break commands to manage iteration	**17, 18**
7. Notice examples of variables in text-based code. Define and call variables within a program	**14, 15**
8. Construct and manipulate lists to store sequential datasets (e.g. calculating daily soil temperature)	**22, 23**
9. Use functions to organise code logically, defining and using functions that accept parameters, execute instructions and return processed data values	**19, 20, 21**
10. Use the radio module to transmit and receive data packets between multiple Microbits	**28**
11. Use on-screen information to identify types and locations of bugs	**13 - 28**
12. Convert values between binary and decimal, explaining how sensor readings are stored in digital memory	
13. Apply standard searching (linear) and sorting techniques to process and organise datasets of sensor or spreadsheet data	**27**
14. Explain how information is sent over networks using packets and network topology.	**29, 30**

Goal 2

FUNCTIONAL AND PROFESSIONAL ENGLISH

Lesson 5 – Using Google Translate	
Goal	**Goal 3 – Functional and professional English**
Learning objective	We are learning to identify, pronounce and classify key agricultural crops and inputs in English, using digital translation tools to cross-check technical terms
UN Sustainable Development Goal	 **SDG 4: Quality Education (Target 4.4: …Substantially increase the number of youth who have relevant skills…)** Developing skills for using English to communicate
Success Criteria	• I can name at least 8 local agricultural products in English with correct pronunciation • I can use the Google Translate Camera feature to scan labels of farming materials and verify accuracy • I can complete a bilingual "Farm Vocabulary Logbook" using British Council flashcard techniques
Introduction	**Time: 15 minutes** 1. The teacher displays some real items or photos of items of local Nepalese crops (i.e. paddy, ginger, orthodox tea, mustard seeds) 2. In pairs, list local Nepalese names and attempt to translate these into English 3. Discuss why it is important to know the English names of products (English is the international language of trade).
Main lesson	**Time: 35 minutes** **Step 1: British Council resource exploration (10 mins)** • Show students how to navigate to the **British Council Farming website)** website. • Ask students to work though the food vocabulary exercises:

	https://learnenglishteens.britishcouncil.org/vocabulary/a1-a2-vocabulary/fruit https://learnenglishteens.britishcouncil.org/vocabulary/a1-a2-vocabulary/vegetables https://learnenglishteens.britishcouncil.org/vocabulary/a1-a2-vocabulary/food **Step 2: Google Translate scan and verify (10 mins)** • The teacher provides sample packaging labels (seed packets, organic compost bags with English descriptions) • Show students how to use the Google Translate app and use the **Camera/Lens feature** to translate English instructions into Nepali. • Students match core terms *(seeds, fertiliser, soil, harvest, irrigation, crop yield*) with English definitions and images. **Step 3: Google Translate – Scan and Verify (15 mins)** • Teacher provides children with sample packaging labels (e.g. seed packets, organic compost bags with English descriptions) • Students open Google Translate and use the **Camera/Lens tool** to translate English instructions into Nepali. • **Critical thinking** – Students test whether Google Translate translates local crop items accurately. They correct any inaccurate, literal translations. **Step 4: Creating an 'Agricultural word bank' (5 mins)** • Students open Word and create a dictionary with Nepali words and their English translations. Ensure students save their work before the end of the lesson.

Plenary	**Time: 10 minutes** 1. Play a quick game of "pass the ball", where students pass a ball around the room. When the teacher says "stop" the student holding the ball must name one item of vocabulary from today and use it in a sentence. 2. Ensure students save their work and shut down their computers.

Lesson 6 – Google Translate Speech to Text	
Goal	**Goal 3 – Functional and professional English**
Learning objective	We are learning to write a guide using imperative verbs and sequential transition words
UN Sustainable Development Goal	 **SDG 4: Quality Education (Target 4.4: …Substantially increase the number of youth who have relevant skills…)** • Developing skills for using English to communicate
Success Criteria	• I can recognise sequential adverbs (i.e. *First, Next, Then, Finally*) • I can recognise imperative verbs (i.e. *Mix, water, wear, store*) • I can use Google Translate Voice Input to check my English pronunciation • I can draft a farm safety poster using British Council resources
Introduction	1. Ask children about the types of actions that would be used in an agricultural setting. 2. Make a list of these on the board. 3. Explain that these are called imperative verbs.
Main lesson	**Time: 35 minutes** **Step 1: Imperative Simon Says (5 minutes)** • The teacher leads a game of imperative Simon Says, using farm-base imperative verbs, i.e. *"Plant the seed!", "Water the crop!" "Wear protective gloves!" "Store the tools!"* **Step 2: British Council reading model (5 minutes)** • Students read the modified short text form the British Council learn English website explaining how to grow plants.

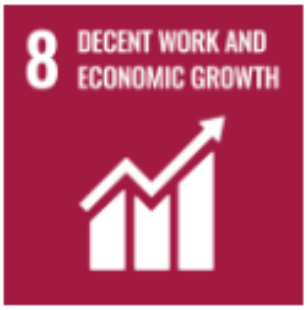
	https://learnenglishkids.britishcouncil.org/read-write/reading-practice/level-1-reading/growing-plant • Students highlight or point out imperative verbs. • Students point out sequential adverbs. **Step 3: Google Translate Speech to text pronunciation check (15 minutes)** • Students dictate English imperative sentences (either their own or form the reading) into Google Translate's microphone (speech to text). • If the app misinterprets their voice, students adjust their stress and pronunciation until the tool transcribes the English correctly. **Step 4: Collaborative task - Farm safety and process poster (10 minutes)** • In groups or pairs, students select a local process *(e.g. How to harvest cardamom, how to store seed grain safely)* • Students write a 5-step guide for this process in English • Groups scan their written sentences into Google Translate to check their grammar and vocabulary.
Plenary	**Time: 10 minutes** • Ask the children to read out their instructions to the class.

Lesson 7 – Google Translate Conversation Tool	
Goal	**Goal 3 – Functional and professional English**
Learning objective	We are learning to use translation tools to help us engage in a negotiation activity
UN Sustainable Development Goal	 **SDG 8: Decent Work and Economic Growth (Target 8.3: Promote development-oriented policies that support productive activities)** Learning not use English for sales and negotiation
Success Criteria	• I can ask and answer questions about price, quantity and quality in English, i.e. *"How much per kilogram?" "Is this organic?" "We offer a bulk discount"* • I can use Google Translate's **Conversation Mode** to bridge a language gap during a conversation practice • I can complete an English order for based on the British Council listening and speaking resources
Introduction	1. Ask how technology can help us in translation situations. 2. Ask what kind of translation situations might arise in the context of agriculture (selling to a foreign buyer)
Main lesson	**Time: 35 minutes** **Step 1: Market vocabulary and number practice (10 minutes)** • The teacher leads the class in fast-paced practice of numbers, counting in multiples of 10 and 100. • Practise pronouncing units of measurements, i.e. grams, kilograms.

	Step 2: Listening to a dialogue (5 minutes) • Students listen to a dialogue in a shop form the **British Council website**. Ask students to note down key phrases. https://learnenglishteens.britishcouncil.org/skills/speaking/a1-speaking/shop **Step 3: Google Translate Conversation mode (15 minutes)** • Students work in pairs, with one playing the part of an international buyer. • Students use Google Translate to conduct a live translation. **Step 4: Recording language (5 minutes)** • Ask students to record key words and phrases in their Microsoft Word vocabulary books form this lesson.
Plenary	**Time: 10 minutes** • Ask for two volunteers to conduct a negotiation exercise in front of the class. One student plays the part of a buyer and the other a seller. They try to negotiate for the best price using phrases from this lesson.

Lesson 8 – Environmental vocabulary	
Goal	**Goal 3 – Functional and professional English**
Learning objective	We are learning to read and summarise and English weather/climate report and explain the effects that climate has on local crops in English
UN Sustainable Development Goal	 **SDG 13: Climate Action (Target 13.3: Improve education, awareness-raising…on climate change mitigation)** Building capacity to read climate reports and adapt agricultural practices
Success Criteria	• I can use key weather and climate vocabulary (*monsoon, drought, landslide, temperature, irrigation, climate resilience)* • I can summarise an English climate article using the British Council reading strategies and looking up words in Google Translate • I can express a cause an effect using structures like *'Due to heavy rain…'* or '*If there is a drought, then…'*
Introduction	• Show some pictures associated with weather and climate on the board. • Ask students to think of as many words as possible to describe weather and climate and the effects of weather and climate
Main lesson	**Time: 35 minutes** **Step 1: British Council materials – "Storm Coming!" (20 minutes)** • Students open the resources from the British Council website (Climate Action in language Education series)

	https://www.teachingenglish.org.uk/teaching-resources/teaching-secondary/lesson-plans/upper-intermediate-b2/storm-coming • Directs students to complete the reading and gap filling worksheet. • Encourage students to use Google Translate to help them with any unfamiliar words. • Discuss answers as a class. **Step 2: Listening activity (10 minutes)** • Guide children to open the listening activity on the same website. • Students listen to news reports in English and take notes as they listen before answering the questions. **Step 3: Guided typing practise (5 minutes)** • In their Word English journals, students complete the following sentences explaining cause and effects: - "If monsoon rains arrive late, farmers must..." - "Excessive rainfall causes..." - "Using organic compost protects the soil because..."
Plenary	**Time: 10 minutes** **Weather presenter mini pitch** - Student volunteers give a 30-second English weather advisory for their local district. - E.g. "Good morning! Expect heavy rain on Wednesday! Make sure you protect any young tree shoots from waterlogging."

Lesson 9 – Business emails	
Goal	**Goal 3 – Functional and professional English**
Learning objective	We are learning to write a professional inquiry email in English to an agricultural supplier or potential customer
UN Sustainable Development Goal	 **SDG 8: Decent Work and Economic Growth (Target 8.3: Promote development-oriented policies that support productive activities)** Learning not use English to communicate professionally with buyers
Success Criteria	• I can use the correct layout for a professional email • I can distinguish between formal and informal English language • I can structure an email with a subject line, formal salutation, body paragraph, call to action and sign off • I can draft an email using Google Translate as an aid, then edit to ensure a professional tone
Introduction	• Ask the students when and why we might send emails? • Try to think of professional reasons and informal, social reasons
Main lesson	**Time: 35 minutes** **Step 1: Formal v informal sorting game (5 minutes)** • Provide the students with phrases to sort into formal and informal. These could be displayed o the screen or cut out for students to sort. • For example: - *"Hi, how are you?" v "Dear Sir/Madam"* - *"Do you have any seeds?" v "I would like to request information regarding seed availability"*

	- *"Speak to you soon" v "Kind regards"* **Step 2: British Council writing model (15 minutes)** • Guide children to navigate to the **British Council activity 'A more formal email':** https://learnenglishteens.britishcouncil.org/skills/writing/b2-writing/more-formal-email • Ask students to complete the preparation activity, answering whether the sentences are true or false. • Students complete the reading and gap filling activity based on the example email. **Step 3: Writing an email (15 minutes)** • Students use Word to write a formal email, requesting information about seed availability. • The teacher provides useful phrases on the board for students to use. - *"Dear Sir/Madam"* - *"I am writing to request information about"* • Encourage children to use Google Translate to help them with phrases and vocabulary, but to be critical of its output.
Plenary	**Time: 10 minutes** - Ask the children to swap emails with a classmate and check: - Is the subject line clear? - Is there a formal opening and closing? - Are polite modals used? (would, could, please)

Lesson 10 – Business presentations	
Goal	**Goal 3 – Functional and professional English**
Learning objective	We are learning to deliver a 3-minute presentation in English describing an innovative agricultural area
UN Sustainable Development Goal	 **SDG 2: Zero Hunger (Target 2.4: Sustainable Food Production)** Researching and presenting innovative solutions for increasing the yield of crops
Success Criteria	• I can structure a presentation with an introduction, main Idea, benefits and a conclusion • I can use language for signposting, *(i.e. "Today I will talk about", "The main benefit is", "Thank you for listening")* • I can use digital translation tools to prepare cue cards and verify complex vocabulary
Introduction	• Introduce this lesson as a chance for the children to use all the English they have learned in this unit of work. • Discuss when and where we might have to give a presentation in the context of agriculture (i.e. to suppliers or customers)
Main lesson	**Time: 35 minutes** **Step 1: British Council model analysis (15 minutes)** • Direct the students to the **British Council Teens oral presentation website:** https://learnenglishteens.britishcouncil.org/exams/speaking-exams/oral-presentation • Ask students to watch the example presentation and complete the vocabulary and phrase activities below.

	Step 2: Choosing presentation topic 10 minutes) • Students will prepare a pitch for a local agricultural improvement project. Introduce three presentation topics for students to choose from: **Idea A: Solar powered ginger dryer** **Idea B: Rainwater collection for greenhouse peppers** Groups outline their presentation using a guided template: 1. Project name and team roles 2. Agricultural problem addressed 3. Proposed solution and tools needed 4. Expected benefits for the village **Step 3: Translation and preparation (10 minutes)** • Students work in groups and prepare a short presentation on their topic. • Students use Google Translate to help them generate vocabulary and phrases for their presentation. **Step 4: Presentation (10 minutes)** • Each group delivers a 5-minute presentation to the rest of the class. • Classmates act as investors, asking questions including "*How much does it cost?"* and *"Where will you build it?"*

Plenary	**Time: 10 minutes**
	
	• Ask the children to fill out an exit card, including answering:
	  • 3 new English words I learned relating to agriculture
	  • 1 new Google Translate feature I find most useful
	  • 1 way English will help me in the future

Objectives taught in this unit (highlighted)

Advanced level (age 13 – 16) – teaching objectives	
ICT Skills	**Lesson(s):**
1. Type accurately, at a speed of 30wpm, using both hands correctly	**3**
2. Master advanced file systems/: identify structural extensions (.py, .csv, .pdf, .zip) and manage local desktop vs. cloud directory.	**1**
3. Establish physical hardware connections using both Bluetooth and USB, connect a Microbit to another Microbit via radio	**28**
4. Create multi-page documents including custom page margins, table of contents, text alignment adjustments, page breaks.	**2**
5. Design agricultural and business spreadsheets that use comparative operators and conditional formulas, e.g. SUM, AVERAGE, COUNTIF.	**37, 38**
6. Deliver a high-impact presentation designed for a target audience using design tools and following conventions.	**9, 10**
7. Use principles of graphics design to create 2D images, i.e. alignment, contrast.	**40, 41, 42**
8. Use sophisticated, interactive online tools to research and critically evaluate information using this to further learning and investigations and create original content	**4, 5, 6, 7, 8, 11, 12, 39**
Digital literacy, AI literacy and E-safety	**Lesson(s):**
1. Explain specific legal frameworks governing technology use. Contrast open-source content with copyright infringement.	**1, 31**
2. Describe and utilise strategies to identify and counter the effects of screen-addition.	**32**
3. Analyse the effects of digital harassment, describing protocols to mitigate and report cyberbullying.	**32**
4. Cross reference primary sources to spot digital manipulation, deepfakes and automated disinformation campaigns.	**33**
5. Evaluate the local and global economic impacts of automation and artificial intelligence on rural labour markets and agricultural supply chains.	**35**
6. Outline the structural mechanics of machine learning, explaining concepts behind algorithms (i.e. decision trees and data pattern recognition).	**34, 35**
7. Critique the risks if modern generative models, covering training bias, data leaks, security risks and cognitive offloading.	**36**
8. Navigate the psychological hazards of anthropomorphised AI, explaining the risk of users forming inappropriate attachments to digital conversational agents.	**36**

Programming and computational thinking	Lesson(s):
1. Open an IDE (i.e. https://python.microbit.org/v/3/) and initialise a new script by importing the core hardware library (from microbit import *)	**13**
2. Use inputs (i.e. button_a.is_pressed()) and display output dynamically using the LED matrix (display.show()) and text scrolling	**24, 25, 26**
3. Use the built-in functions in Python, including print(), input() and time.sleep()	**13, 14**
4. Read, process and convert analogue environmental inputs from alligator clip probes and onboard sensors (light, temperature, accelerometer)	**25, 26**
5. Use conditional structures (if, elif, else) and relational operators (<, =, >) to trigger outputs	**15, 16**
6. Use 'or', 'and', 'not' logic in conditional code and construct truth tables to map hardware decision making	**16**
7. Program text-based looping structures (while True: and for I in range()), use timing controls (sleep()) and break commands to manage iteration	**17, 18**
7. Notice examples of variables in text-based code. Define and call variables within a program	**14, 15**
8. Construct and manipulate lists to store sequential datasets (e.g. calculating daily soil temperature)	**22, 23**
9. Use functions to organise code logically, defining and using functions that accept parameters, execute instructions and return processed data values	**19, 20, 21**
10. Use the radio module to transmit and receive data packets between multiple Microbits	**28**
11. Use on-screen information to identify types and locations of bugs	**13 - 28**
12. Convert values between binary and decimal, explaining how sensor readings are stored in digital memory	
13. Apply standard searching (linear) and sorting techniques to process and organise datasets of sensor or spreadsheet data	**27**
14. Explain how information is sent over networks using packets and network topology.	**29, 30**

Goal 3

GEOGRAPHIC INFORMATION AND MAPPING

Lesson 11 – Interpreting data from Google Earth	
Goal	**Goal 3 – Geographic information and mapping**
Learning objective	We are learning to used advanced measurement tools in Google Earth and interpret this data using a spreadsheet
UN Sustainable Development Goal	 **SDG 11 Sustainable Cities and Communities (Target 11.3: Enhance inclusive and sustainable urbanisation)** Measuring distance and land areas to facilitate urban planning
Success Criteria	• I can trace a slope profile line in Google earth using **Google Earth's Elevation Profile and Path Measurement tool** • I can calculate total slope elevation drop and gradient percentage using =Max() and =Min() formulas in either Libre Office Calc or Excel. • I can generate a custom profile chart to evaluate terrain suitability for terraced farming
Introduction	**Time: 15 minutes** 1. Ask the children how satellite mapping tools can be useful for farming 2. Discuss how Google Earth can generate precise numerical data and remind children how spreadsheets can be used to interpret large amounts of data.
Main lesson	**Time: 35 minutes** **Step 1: Spatial mapping in Google Earth (20 mins)** • Instruct the children to double click on Google earth on the desktop link • Show children how to navigate to a local hilly region

	• Show children how to select the **Measure Tool (Path).** Draw a straight line down a hillside from top to bottom (approx. 500m to 1km long).
	• Enable **Show Elevation Profile.**
	• Collect 10 data points along the path at regular intervals, recording:
	  • Distance from top (metres)
	  • Elevation above sea level (metres)
	
	Step 2: Data entry and formula modelling in Google Sheets (25 mins)
	
	• Show children how to open a new spreadsheet in Excel.
	• Set up columns: **Point (number), Distance (m), Elevation (m)**
	• Input the collected data points from above.
	• Apply the formulas:
	  • **Highest elevation**: =Max(C2:C11)
	  • **Lowest elevation**: = Min(C2:C11)
	  • **Total elevation change**: = Max(C2:C11) - Min(C2:C11)
	  • **Total horizontal distance**: = max(B2:B11) – Min(B2:B11)
	  • **Slope gradient change (%)** = (Total elevation change / Total horizontal distance) * 100
	
	Step 3: Chart creation and agricultural decision making (15 mins)
	
	• Show the children how to highlight the distance and elevation columns.
	• Insert a smooth line chart (X-axis: distance, Y-axis: elevation) to visualise the hillside cross section.
	• Apply the agricultural analysis rule:
	  • Children should make decisions based on:
	  0% - 5% slope – ideal for paddy cultivation
	  5% - 15% slope – Moderate – requires terracing for seasonable vegetables/maize

	>15% slope – Requires agroforestry, fruit trees, or tea plantations to prevent landslides
Plenary	**Time: 10 minutes** 1. Ask students to share a short presentation about their gradient percentage and share their recommendations, i.e. *"Based on our 18% slope calculation, this hillside is too steep for rice paddies without terracing, so we recommend planting fruit trees or tea shoots to hold the topsoil in place."* ***(Provide some sentence examples to support children)***

Lesson 12 – Using Google Earth' Polygon and Historical Imagery tools	
Goal	**Goal 3 – Geographic information and mapping** **This lesson would likely need at least two lessons to complete**
Learning objective	We are learning to use Google Earth's Polygon and Historical Imagery tools to measure farm plots and track change over time
UN Sustainable Development Goal	 **SDG 11 Sustainable Cities and Communities (Target 11.3: Enhance inclusive and sustainable urbanisation)** • Measuring distance and land areas to facilitate urban planning
Success Criteria	• I can draw a precise polygon around a farm plot in Google Earth to measure perimeter and area in metres squared • I can record temporal area changes across different years using the **Historical Imagery slider** • I can use multiplication and conditional formatting formulas in a spreadsheet to estimate required farm inputs (seeds/water).
Introduction	**Time: 15 minutes** 1. Ask the children why the land areas is important for calculating the number of seeds and irrigation that we need. 2. Explain that in this lesson, we will be using measurement tools in Google Earth to help us calculate these things.
Main lesson	**Time: 35 minutes** **Step 1: Polygon measurement using Google Earth (10 mins)**

- Show the children how to find a village farming area using Google Earth
- Draw a polygon around three different fields. Record the perimeter (metres) and area (metres squared).

Step 2: Historical timelapse analysis (10 mins)

- Show children how to activate the **Historical Imagery slider** (using the web-based version of Google Earth)
- Slide the slider back 10-15 years and re-measure the forested area or riverbed boundary near the farm.
- Record the forested area or cultivated area over three different years (i.e. 2010, 2015, 2024)

Step 3: Agricultural Input Calculations using a spreadsheet (10 mins)

- Show the students how to create two tables in a spreadsheet.

Table A: Agricultural input calculations
Column A: Field name
Column B: Area (metres squared)
Column C: Seed need (kg) → Formula: =B2 * 0.005 (assuming 5g seed per metre squared)
Column D:Water requirement (litres/day) → Formula: =B2 * 6 (Assuming 6L water per metre squared of rice)
Total summary row = SUM (B2:B4), =SUM (C2:C4), =SUM(D2:D4)

Table B: L**and change analysis over time**

Year	Forest areas (metres squared)	Cultivated area
2010	12,000	5,000
2018	9,500	7,500
2024	7,300	9,800

	Formulas: Percentage change =((B4-B2) / B2) * 100 **Visualisation**: Insert a stacked column chart showing how forest land transitioned to cultivated land over time. **Step 4: Conditional formatting (10 mins)** • Show students how to highlight data and use conditional formatting to highlight fields consuming over 10,000 litres of water in red, identifying where irrigation is needed.
Plenary	**Time: 10 minutes** • Ask the question: *"If forest cover decreased by 40% over 14 years, how does this affect monsoon water runoff and topsoil for the lower farms?"* Students should justify their answer using her calculated spreadsheet data.

Objectives taught in this unit (highlighted)

Advanced level (age 13 – 16) – teaching objectives	
ICT Skills	**Lesson(s):**
1. Type accurately, at a speed of 30wpm, using both hands correctly	**3**
2. Master advanced file systems/: identify structural extensions (.py, .csv, .pdf, .zip) and manage local desktop vs. cloud directory.	**1**
3. Establish physical hardware connections using both Bluetooth and USB, connect a Microbit to another Microbit via radio	**28**
4. Create multi-page documents including custom page margins, table of contents, text alignment adjustments, page breaks.	**2**
5. Design agricultural and business spreadsheets that use comparative operators and conditional formulas, e.g. SUM, AVERAGE, COUNTIF.	**37, 38**
6. Deliver a high-impact presentation designed for a target audience using design tools and following conventions.	**9, 10**
7. Use principles of graphics design to create 2D images, i.e. alignment, contrast.	**40, 41, 42**
8. Use sophisticated, interactive online tools to research and critically evaluate information using this to further learning and investigations and create original content	**4, 5, 6, 7, 8, 11, 12, 39**
Digital literacy, AI literacy and E-safety	**Lesson(s):**
1. Explain specific legal frameworks governing technology use. Contrast open-source content with copyright infringement.	**1, 31**
2. Describe and utilise strategies to identify and counter the effects of screen-addition.	**32**
3. Analyse the effects of digital harassment, describing protocols to mitigate and report cyberbullying.	**32**
4. Cross reference primary sources to spot digital manipulation, deepfakes and automated disinformation campaigns.	**33**
5. Evaluate the local and global economic impacts of automation and artificial intelligence on rural labour markets and agricultural supply chains.	**35**
6. Outline the structural mechanics of machine learning, explaining concepts behind algorithms (i.e. decision trees and data pattern recognition).	**34, 35**
7. Critique the risks if modern generative models, covering training bias, data leaks, security risks and cognitive offloading.	**36**
8. Navigate the psychological hazards of anthropomorphised AI, explaining the risk of users forming inappropriate attachments to digital conversational agents.	**36**

Programming and computational thinking	Lesson(s):
1. Open an IDE (i.e. https://python.microbit.org/v/3/) and initialise a new script by importing the core hardware library (from microbit import *)	**13**
2. Use inputs (i.e. button_a.is_pressed()) and display output dynamically using the LED matrix (display.show()) and text scrolling	**24, 25, 26**
3. Use the built in functions in Python, including print(), input() and time.sleep()	**13, 14**
4. Read, process and convert analogue environmental inputs from alligator clip probes and onboard sensors (light, temperature, accelerometer)	**25, 26**
5. Use conditional structures (if, elif, else) and relational operators (<, =, >) to trigger outputs	**15, 16**
6. Use 'or', 'and', 'not' logic in conditional code and construct truth tables to map hardware decision making	**16**
7. Program text-based looping structures (while True: and for I in range()), use timing controls (sleep()) and break commands to manage iteration	**17, 18**
7. Notice examples of variables in text-based code. Define and call variables within a program	**14, 15**
8. Construct and manipulate lists to store sequential datasets (e.g. calculating daily soil temperature)	**22, 23**
9. Use functions to organise code logically, defining and using functions that accept parameters, execute instructions and return processed data values	**19, 20, 21**
10. Use the radio module to transmit and receive data packets between multiple Microbits	**28**
11. Use on-screen information to identify types and locations of bugs	**13 - 28**
12. Convert values between binary and decimal, explaining how sensor readings are stored in digital memory	
13. Apply standard searching (linear) and sorting techniques to process and organise datasets of sensor or spreadsheet data	**27**
14. Explain how information is sent over networks using packets and network topology.	**29, 30**

Goal 4

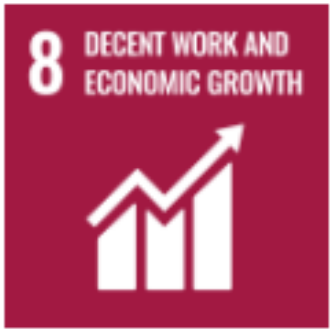
INTRODUCTION TO SOFTWARE LOGIC

Lesson 13 – Using 'print', 'input' functions and variables in Python (basic)	
Goal	**Goal 5 – Introduction to software logic (Python)** This unit of work assumes an understanding of concepts taught first through block-based coding. Students may benefit from doing some of the Foundation or Intermediate coding lessons before the Python lessons. Video tutorial available here
Learning objective	We are learning to use the print() and input() functions in Python with variables
UN Sustainable Development Goal	 **SDG 4: Quality Education (Target 4.4: …Substantially increase the number of youth who have relevant skills…)** Developing skills for using Python to write programs
Success Criteria	• I can open Python through the Mu IDE. https://codewith.mu/ • I can read Python and spot examples of syntax • I can type accurate syntax, including capital letters and tab for indentation • I can use the print and input functions and explain the difference between them • I can create and change variables • I can print variables
Introduction	**Time: 15 minutes** 1. Show the children some code written using Python Ask them if they have ever seen coding like this before. Ask them what they notice. Pick out aspects of syntax (i.e. brackets, capital letters, spaces) and logic.

	2. Explain that over the next few weeks, we are going to be learning to use Python to create programs linked to agriculture. 3. Explain that while Python is a simple language to learn, it is a powerful language that is used in industry.
Main lesson	**Time: 35 minutes** **Step 1: Opening Mu and starting a new project in Python** **(5 mins)** • Show the children how to open Mu and explain that this is an IDE (interactive development environment) for Python. • Ask students to type their first Python program: **print("hello world")** • Students should press 'run; which will prompt them to name and save their program before it runs. • Allow students to experiment with this simple program, (i.e. by copying and pasting other lines of code) • Ask students if they can 'break' their program (i.e. by changing capitalisation). Ask them to notice the debug message at the bottom of the screen which tells them the type of error and where it is. **Step 2: Using the print function (10 mins)** • Look at the following code for the first part of this project: **print("Welcome to the District Agricultural centre")** **print("We help farmers to plant their crops.")**

	• Ask students what they notice about this code, and to predict what it will do. • Guide students in typing this code. • Encourage them to use the debugging messages to fix their code if it doesn't work. **Step 3: Using the input function (10 mins)** • Show children the following code: **farmer_name = input ("What is your name?")** **print ("Namaste", farmer_name)** • Again, ask children to read and make predictions about the code. • Point out the variable farmer_name. Elicit that this is set using the input function. **Step 4: Creating a second variable (10 mins)** • Show children the following code: **farmer_name = input ("Enter farmer name")** **crop_type = input ("What crop are you planting today?")** **print ("---REGISTRTATION CARD---")** **print ("Farmer Name:", farmer_name)** **print ("Registered Crop", crop_type)** **print ("Registration complete!")** • Again, ask children to read and make predictions about the code. • Point out the variable farmer_name. Elicit that this is set using the input function. • Children attempt to type this code themselves. Encourage them to read error messages when the code runs

Plenary	**Time: 10 minutes** 1. Ask the students for their initial thoughts and observations about creating programs using Python. 2. Emphasise the importance of using the debugging tools – these allow students to be independent of the teacher. 3. Ensure students save their work and log off their computers.

Lesson 14 – Using 'print', 'input' functions and variables in Python (advanced)	
Goal	**Goal 5 – Introduction to software logic (Python)** Video tutorial available here
Learning objective	We are learning to use the print() and input() functions in Python with variables
UN Sustainable Development Goal	 **SDG 8: Decent work and Economic Growth (Target 8.2: Achieve higher levels of economic productivity through diversification, technological upgrading and innovation)** Calculating farm yield using digital tools
Success Criteria	• I can open Python through the Mu IDE. https://codewith.mu/ • I can read Python and spot examples of syntax • I can type accurate syntax, including capital letters and tab for indentation • I can use the print and input functions and explain the difference between them • I can create and change variables • I can print variables
Introduction	**Time: 15 minutes** 1. Show the children some code written using Python Ask them if they have ever seen coding like this before. Ask them what they notice. Pick out aspects of syntax (i.e. brackets, capital letters, spaces) and logic. 2. Explain that over the next few weeks, we are going to be learning to use Python to create programs linked to agriculture.

	3. Explain that while Python is a simple language to learn, it is a powerful language that is used in industry.
Main lesson	**Time: 35 minutes** **Step 1: Opening Mu and starting a new project in Python** **(5 mins)** • Show the children how to open Mu and explain that this is an IDE (interactive development environment) for Python. • Ask students to type their first Python program: **print("hello world")** • Students should press 'run; which will prompt them to name and save their program before it runs. • Allow students to experiment with this simple program, (i.e. by copying and pasting other lines of code) • Ask students if they can 'break' their program (i.e. by changing capitalisation). Ask them to notice the debug message at the bottom of the screen which tells them the type of error and where it is. **Step 2: Reading code together (10 mins)** • Look at the finished code for the first part of this project: **farmer_name = input ("Enter farmer name:")** **crop_name = input ("Enter crop name:")** **print ("Yield report for " + farmer_name + "-" + crop_name)** • Ask the students what they predict will happen in this code.

	• Ask them what the two variables are called. Point out the underscore (_) used to separate words when creating variables. • Students try typing this code into Mu and running their programs. • Encourage them to use the debugging messages to fix their code if it doesn't work. **Step 3: Programming variables to change (10 mins)** • Show children the following code: **square_meters = float(input("Enter land size in sq m:"))** **yield_per_sqm = float(input("Enter predicted yield per square metre (kg):"))** **total_yield = square_metres * yield_per_sqm** **print ("Total expected yield:", total_yield, "kg")** • Again, ask children to read and make predictions about the code. • Point out the float()function. Ask for predictions about this. Explain that without the float function, the variable is set to a number entered as text (a string), we want to change this text to a numerical value. • **Extension**: Ask children to multiply total yield by the market price per kilogram to estimate income.
Plenary	**Time: 10 minutes** 1. Ask the students for their initial thoughts and observations about creating programs using Python. 2. Emphasise the importance of using the debugging tools – these allow students to be independent of the teacher. 3. Ensure students save their work and log off their computers.

Lesson 15 – Programming conditions in Python (basic)	
Goal	**Goal 4 – Introduction to software logic (Python)** Video tutorial available here
Learning objective	We are learning to use **if** and **else** statements with make decisions, with the comparator **'=='**
UN Sustainable Development Goal	 **SDG 2: Zero Hunger (Target 2.4: Sustainable Food Production)** Optimising temperature for high-yield nursery crops and protecting delicate seedlings from heat stress, improving crop yields
Success Criteria	• I can start a new Python project in Mu • I can write an if statement that checks a condition • I can use else for the opposite possibility • I can use comparison operators like == to check conditions
Introduction	**Time: 10 minutes** 1. Hold up two signs, 'Rain' and 'Sun'. Ask, *"If it rains, do we open or close the crop drying tarps?"* 2. Explain that computers use if and else to make decisions about changing conditions.
Main lesson	**Time: 35 minutes** **Step 1: Opening Mu and setting up a project (5 mins)** • Ask the children to open a new Python project in Mu. **Step 2: Reading code and making predictions (10 mins)**

	• Show the children the following code and ask them to read it and make predictions: **weather = input ("is it raining today? (yes/no):")** **if weather == "yes":** **print("Cover the drying ginger with tarps!")** **else:** **print("Leave the crops out in the sun to dry")** **Step 3: Shared coding and testing (10mins)** • Guide the children through coding the code above. Encourage them to test their code in stages and fix errors using the debugging messages. **Step 4: Combining input() with decision logic (10mins)** • Show the children the following code and ask them to read it and make predictions: **temp = float(input("Enter today's soil temperature in degrees."))** **if temp < 10:** **print ("Soil temperature is too low! Delay planting")** **else:** **print ("Soil temperature is ideal. Safe to plant.")** • Guide the children through coding the code above. Encourage them to test their code in stages and fix errors using the debugging messages.
Plenary	**Time: 10 minutes** 1. Ask the children to try out the code with one = sign instead of two for the first example. What happens? 2. Explain that Python only uses the '=' sign when comparing numbers

Lesson 16 – Programming conditions in Python (advanced)	
Goal	**Goal 4 – Introduction to software logic (Python)** Video tutorial available here
Learning objective	We are learning to program automated decisions about agriculture using conditional logic in Python
UN Sustainable Development Goal	 **SDG 2: Zero Hunger (Target 2.4: Sustainable Food Production)** Using sensors to help farmers protect their growing crops
Success Criteria	• I can start a new Python project in Mu • I can write an if/else statement to evaluate soil moisture levels or rainfall • I can use an elif to provide 3 different irrigation recommendations • I can combine multiple conditions using logical parameters 'and', 'or'
Introduction	**Time: 15 minutes** 1. Discuss water scarcity during the winter and flooding during Monsoon seasons. *"How can software engineers help a farmer decide whether to open irrigation canals?"* 2. Explain branching logic using flowcharts (if/else) in the context of irrigation channels.

Main lesson	**Time: 35 minutes** **Step 1: Opening Mu and setting up a project (5 mins)** • Ask the children to open a new Python project in Mu. **Step 2: Reading code and making predictions (10 mins)** • Show the children the following code and ask them to read it and make predictions: **rainfall_mm = float (input("Enter today's rainfall in mm:"))** **if rainfall_mm > 50:** **print("Warning! Risk of landslide or flooding! Clear the drainage channels!")** **elif rainfall_mm >=15:** **print("Optimal rainfall. Do not turn on irrigation channels.")** **else:** **print("Low rainfall. Activate drip irrigation system for 2 hours.")** **Step 3: Shared coding and testing (20mins)** • Guide the children through coding the code above. Encourage them to test their code in stages and fix errors using the debugging messages.
Plenary	**Time: 10 minutes** 1. Ask the children to try out each other's code with extreme values to see if the system is working 2. Ensure that children save their code and shutdown their computers carefully.

<table>
<tr><th colspan="2">Lesson 17 – Creating ‘while True’ loops (basic)</th></tr>
<tr><td>Goal</td><td>Goal 4 – Introduction to software logic (Python)

Video tutorial available here</td></tr>
<tr><td>Learning objective</td><td>We are learning to use for loops for repetition and the sleep function to pause.</td></tr>
<tr><td>UN Sustainable Development Goal</td><td> 

SDG 15: Life on Land
(Target 15.5: “Protect and prevent he extinction of threatened species”)
Manage human-wildlife interactions without harming wild animals</td></tr>
<tr><td>Success Criteria</td><td>• I can start a new Python project
• I can create a repeating while True loop
• I can break out of a loop
• I can import the time library
• I can use the time.sleep() function to pause</td></tr>
<tr><td>Introduction</td><td>Time: 15 minutes

1. Ask the children, "If a guard sits at a field all night, do they check for animals just once?"
2. Elicit that this action happens continuously, until the night ends!
3. Discuss how computers also use loops to check something continuously. Ask for examples of continuous checking in loops. (i.e. sensors)</td></tr>
<tr><td>Main lesson</td><td>Time: 35 minutes

Step 1: Reading and making predictions about code (5 mins)

• Show the children the following Python code:</td></tr>
</table>

	while True: **print ("Scarecrow Alert: Wild animal deterrent active!")** • Ask the children what they think 'while True' means. Point out the capital letter in True. **Step 2: Shared programming in Python (15 mins)** • Guide the children through programming the above code in Python. • Encourage them to test and debug using the debugging tool in Mu. **Step 3: Crating a pause (10 mins)** • Show the students the following code: **import time** **while True:** **print ("Scarecrow Alert: Wild animal deterrent active!")** **time.sleep(2)** • Ask the students what is different about this code. (Point out the import of the time library and use of the time.sleep command.) • Take predictions about what this code will do. Try using time.sleep without importing the time library. • Ask students to try adding a time.sleep command to their code.

	Step 3: Breaking out of a loop (10 mins) • Show the children the following Python code: **import time** **counter = 0** **while True:** **counter = counter +1** **print ("Scarecrow Alert: Wild animal deterrent active!")** **print (counter)** **time.sleep(2)** **if counter > 10:** **break** **print ("Wild animal alert inactive")** **1.** Ask children what they think the 'break' command does. When does it break out of the loop? 2. Guide the children through programming the above code in Python. 3. Encourage them to test and debug using the debugging tool in Mu. *4. (**N.B.** the above code could also be written using a 'repeat x times' loop. This could encourage class discussion about which is better).*
Plenary	**Time: 10 minutes** 1. Ask the students how we combined loops and variables today. 2. What kind of loop does while True create? 3. How can we break out of these loops? 4. Ensure children save their work and log off their computer.

Lesson 18 – Creating 'while' and 'for' loops in Python (advanced)	
Goal	**Goal 4 – Introduction to software logic (Scratch)** **Video tutorial available here**
UN Sustainable Development Goal	 **SDG 13: Climate Action (Target 13.3: "Improve education, awareness-raising and institutional capacity on climate change mitigation")** Creating digital systems to respond to changing weather patterns
Learning objective	We are learning to use for loops for fixed repetition and while loops for threshold monitoring
Success Criteria	• I can start a new Python project • I can write a for loop that runs 7 times to collect daily farm data • I can calculate running totals and weekly averages using an accumulator variable • I can use a while loop to trigger an alert when cumulative weekly rain exceeds safety thresholds
Introduction	**Time: 15 minutes** 1. Ask the children, *"Instead of typing the same input() code 7 times for 7 days of the week, how can we write it once and repeat it efficiently?"* 2. Demonstrate a **for i in range ()** loop, and how variables inside loops update over time.
Main lesson	**Time: 35 minutes** **Step 1: Reading and making predictions about code (5 mins)** 1. Show the children the following Python code:

	total_rain= 0 **for day in range (1, 8):** **rain = float(input("Enter rainfall for Day" + str (day) + " (mm): "))** **total_rain = total_rain + rain** **average_rain = total_rain / 7** **print ("Total 7-day Rainfall:", total_rain, mm")** **print ("daily Average:", round (average_rain, 2), "mm")** 2. Ask the children to read and make predictions about what the code will do. Point out the line; **print ("daily Average:", round (average_rain, 2), "mm")** 3. Ask the children what they think the '2' in this line does and explain that it rounds the amount to 2 decimal places. Discuss why this is useful. **Step 2: Shared programming in Python (15 mins)** • Guide the children through programming the above code in Python. • Encourage them to test and debug using the debugging tool in Mu. **Step 3: Adding a while loop (5 mins)** • Show the students how to create a Monsoon flood alert by adding daily rain in a while loop until cumulative rain exceeds 200m, then outputting "Flood Safety Threshold Reached". "
Plenary	**Time: 10 minutes** 1. Ask the students why we have created and used different types of loops in Python today. Which is each type of loop useful for

Lesson 19 – Functions and parameters in Python (basic)	
Goal	**Goal 4 – Introduction to software logic (Python)** Video tutorial available here
Learning objective	We are learning to define as block of code as a function using **def** so that we can reuse sections of code
UN Sustainable Development Goal	 **SDG 4: Quality Education (Target 4.4: …Substantially increase the number of youth who have relevant skills…)** learning how to organise code by defining and calling functions
Success Criteria	• I can start a new project in Python using Mu as the IDE • I can define a basic function • I can place instructions within the function • I can call the functions several times in my program
Introduction	**Time: 15 minutes** 1. Ask the children if they have heard of functions before. 2. Explain that a function is like a program inside a program. We create a block of code and call it a function 3. Parameters are the numerical values that we feed into this function. 4. Explain that the children have used functions before, such as **'print()** and **input(),** the brackets are the parameters where we will feed numerical values today when we create our own functions. 5. Discuss why functions are important – they allow programmers to reuse entire sections of code.

Main lesson	**Time: 35 minutes** **Step 1: Reding and making predictions about Python code (10 mins)** • Show children the following code in Python: **def show header ():** **print (==========)** **print(" Nepal Farm Management ")** **print (==========)** **show header()** • Then make predictions about how the program will work before running the program. • Talk about the empty parentheses after the function. **Step 2: Guided programming (15 mins)** • Guided programming using Python. The teacher guides the children through creating the program above, with children using the debugging tools to check their coding and debug their work. • Ask the children to call the function more than once in their program, i.e.: **show header()** **show header()** **Step 3: Extension (10 mins)** 1. Show the children an example of a function combined with an if/else structure (code below). 2. Ask for predictions about what this code will do.

	def display_alarm(): **print("!Warning!")** **print("Low temperature detected in greenhouse!")** **print("Close all windows immediately!")** **temp = float(input("Enter current greenhouse temperature"))** **if temp < 5:** **display_alarm()** **else:** **print("Temperature is normal.")** 3. Children add the example code to their program. Encourage children to test and debug, using onscreen tools.
Plenary	**Time: 10 minutes** 1. Ask the students to explain the functions they have used in this lesson and suggest ideas for other functions we could create. 2. Point out the empty parenthesis in the functions we created – what could go in here? Explain that we will explore this next lesson. 3. Ensure that children save their work and shut down their computers safely.

Lesson 20 – Functions and parameters in Python (intermediate)	
Goal	**Goal 4 – Introduction to software logic (Python)** This lesson will likely take more than 35 minutes. Feel free to take multiple lessons to explain this difficult and important topic. Video tutorial available here
Learning objective	We are learning to define as block of code as a function using **def** which accepts **arguments**
UN Sustainable Development Goal	 **SDG 4: Quality Education (Target 4.4: …Substantially increase the number of youth who have relevant skills…)** learning how to organise code by defining and calling functions with parameters
Success Criteria	• I can start a new project in Python using Mu as the IDE • I can define a function with a single parameter • I can use the parameter within the function's code block • I can call the function multiple times using pieces of farm data (strings or numbers)
Introduction	**Time: 15 minutes** 1. Show the children a picture of a juice press. 2. Ask what happens if you put apples or pears into the machine (the machine does the same action, but the output changes!) 3. Explain that in this lesson, we will continue making functions, but will add a space to change the output. We call this a **parameter**. This allows the function to perform its task using different pieces of data.

Main lesson	**Time: 35 minutes** **Step 1: Reding and making predictions about Python code (10 mins)** • Show children the following code in Python: **#DEFINE the function with a parameter called 'name'** **def welcome_farmer(name):** **print ("Namaste", name)** **print("Welcome to the village agricultural hub!")** **print("----------")** **#CALL the function with different arguments** **welcome_farmer("Sita")** **welcome_farmer("Bimlesh")** • Then make predictions about how the program will work before running the program. **Step 2: Guided programming (15 mins)** • Guided programming using Python. The teacher guides the children through creating the program above, with children using the debugging tools to check their coding and debug their work. **Step 3: Extension – Calculating with variables (10 mins)**

1. Ask children to read and predict the following code:

```
def calculate_compost(field_size):
        bags_needed = field_size * 2
        print ("for a field size," field_size, "Square metres:")
        print("You will need", bags_needed, "bags of organic compost.")
        print("-----------------------------------------------")

#Call the function for different field sizes
calculate_compost(3)
calculate_compost(10)
```

- Guided programming using Python. The teacher guides the children through creating the program above, with children using the debugging tools to check their coding and debug their work.

Step 4: Extension 2 – User input as a variable (10 minutes)

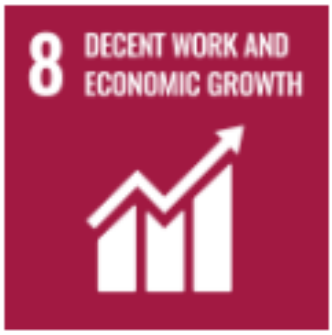
```
#DEFINE the advisory tool

def check_crop_status(crop):
        print("Fetching advisory notes for", crop)
        print("Tip: Ensure soil is well-drained during monsoon season")
        print("----------------------------------------")

#MAIN PROGRAM
user_crop = input("What crop are you planting today?")

#PASS the user's crop input directly into your function parameter

check_crop_staus(user_crop)
```

Plenary	**Time: 10 minutes** 1. Show children the following code. Ask, *"When line 4 runs, what value gets stored inside the parameter, 'item'?"* (answer – the text string, "Potatoes") **def print_price (item):** **print("The market price for", item, "is updated daily")** **print_price("Potatoes")**

Lesson 21 – Functions and parameters in Python (advanced)	
Goal	**Goal 4 – Introduction to software logic (Python)** Video tutorial available here
Learning objective	We are learning to define as block of code as a function using **def**, accepting arguments and returning calculated results
UN Sustainable Development Goal	 **SDG 2: Zero Hunger (Target 2.4: Sustainable Food Production)** Using digital tools to calculate the exact amounts of fertiliser to optimise crop output
Success Criteria	• I can start a new project in Python using Mu as the IDE • I can define a basic function with parameters • I can return a value from the function and print it in the main program • I can create multiple reusable functions for different crops (e.g. Mustard, Apple, Paddy)
Introduction	**Time: 15 minutes** 1. Ask the children if they have heard of functions before. 2. Explain that a function is like a program inside a program. We create a block of code and call it a function 3. Parameters are the numerical values that we feed into this function. 4. Explain that the children have used functions before, such as **'print()** and **input(),** the brackets are the parameters where we will feed numerical values today when we create our own functions.

	5. Discuss why functions are important – they allow programmers to reuse entire sections of code.
Main lesson	**Time: 35 minutes** **Step 1: Reding and making predictions about Python code (10 mins)** • Show children the following code in Python: **def calculate_fertiliser (area_sqm, crop_type):** **if crop_type.lower() =="ginger":** **rate = 5.3 # kg per sqm** **elif crop_type.lower() == "mustard":** **rate = 3.2** **else:** **rate = 4.0** **required_kg = area_sqm * rate** **return required_kg** **#Main Program** **farm_area = float(input("Enter field size (Square metres):"))** **crop = input ("Enter crop (Ginger/Mustard/Other):")** **needed = calculate_fertiliser(farm_area, crop)** **print("Fertiliser required:", needed, "kg")** • This is a fairly complex program. Ask the children to start by highlighting the functions that have been made. • Then make predictions about how the program will work before running the program.

	Step 2: Guided programming (15 mins) • Guided programming using Python. The teacher guides the children through creating the program above, with children using the debugging tools to check their coding and debug heir work. **Step 3: Extension (10 mins)** 1. Ask children to create a second function: soil_health_check(ph_level) that returns "Acidic – add lime", "Alkaline – Add gypsum", or "Optimal", based on pH input.
Plenary	**Time: 10 minutes** 1. Ask the students to explain the functions they have used in this lesson and suggest ideas for other functions we could create. 2. Ensure that children save their work and shut down their computers safely.

Lesson 22 – Using lists in Python (basic)	
Goal	**Goal 4 – Introduction to software logic (Python)** Video tutorial available here
Learning objective	We are learning to create and update a list
UN Sustainable Development Goal	 **SDG 8: Decent work and Economic Growth (Target 8.2: Achieve higher levels of economic productivity through diversification, technological upgrading and innovation)** Providing information about warehouse stock levels
Success Criteria	• I can start a new project in Python using Mu as the IDE • I can create a list with 3 items and display it using print() • I can use .append() to add a new item entered by a user • I can recall an updated list • I can if an if…in… statement to check if an item is in a list
Introduction	**Time: 10 minutes** 1. Ask the children, *"If a farmer harvests multiple vegetables, how do we keep track of everything they have collected, without creating a new variable for every vegetable?"* 2. Explain that in programming we can create something called a list, which holds multiple items under a single name inside square brackets [].

Main lesson	**Time: 35 minutes** **Step 1: Creating a list (10 mins)** • Ask the children to set up a new Python project in Mu as the IDE. • Show children the following code and ask them to read and make predictions about it. • Ask for any questions or observations. **farm_crops = ["Paddy", "Maize", "Ginger"]** **print ("Current warehouse inventory:")** **print (farm_crops)** • Ask students to type the above code and experiment with it. • Why does print(farm_crops) not use speech marks around "farm_crops"? What happens if we do put speech marks around this? • Explain that square brackets create a list. If we use normal brackets, ( ), we create a **tuple**, which is similar to a list but with some key differences.

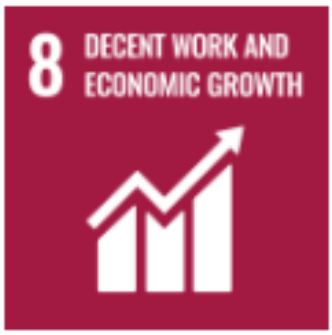
Step 2: Appending a list (15 mins)

- Show children the following code. Ask them to read and make predictions about it.

```
farm_crops = ["Paddy", "Maize", "Ginger"]

#Adding a new harvest item to the end of the list

farm_crops.append ("Mustard")

print ("Updated inventory")
print (farm crops)

#Asking the farmer to input a new crop

new_crop = input ("Enter a new harvested crop to add:")
farm_crops.append(new_crop)

print("Final Inventory after harvest:")
print (farm_crops)
```

- Guide children through typing the above code into Mu and using the debugging tools to check it.
- Encourage children to experiment and expand on the code above.

	Step 3: Checking inside a list (10 mins) • Show children the following code. Ask them to read and make predictions about it. **inventory = ["Paddy", "Potatoes", "Apples", "Cardamom"]** **print ("---Village Storehouse Check---")** **search_crop = input ("Which crop are you looking to buy? ")** **if search_crop in inventory:** **print("YES! We have", search_crop, "available in the storehouse. ")** **else:** **print ("NO! Sorry, ", search_crop, "is currently out of stock. ")** • Guide children through typing the above code into Mu and using the debugging tools to check it. • Encourage children to experiment and expand with the code above. (This coding may need more than one lesson)
Plenary	**Time: 10 minutes** 1. Spot the error: Show children the following code: **crops = ["Rice", "Wheat"]** **crops.append "Buckwheat"** Where is the error? (Missing parenthesis around "Buckwheat")

Lesson 23 – Using lists in Python (intermediate)	
Goal	**Goal 4 – Introduction to software logic (Python)** **Video tutorial available here**
Learning objective	We are learning to store and manipulate information in lists
UN Sustainable Development Goal	 **SDG 8: Decent work and Economic Growth (Target 8.2: Achieve higher levels of economic productivity through diversification, technological upgrading and innovation)** Calculating maximum, minimum and average prices
Success Criteria	• I can start a new project in Python using Mu as the IDE • I can create a list and append user-entered market prices • I can Find the maximum, minimum and average prices using Python built-in functions • I can sort the list and filter out prices below a target threshold.
Introduction	**Time: 10 minutes** 1. Show the children some daily rates from a Fruit and Market development board. 2. Ask the children, *"How can data help farmers decide which market gives the highest profit?"*
Main lesson	**Time: 35 minutes** **Step 1: Reading code and making predictions (10mins)** • Guide children to start a new Python project in Mu.

	• Show children the following code and ask them to read it and predict what it will do: **prices = [80, 95, 75, 110, 85] #Tomato prices in NPR across 5 markets** **print ("All market Prices:", prices)** **print ("Highest Price Available:", max(prices), "NPR")** **print ("Lowest Price Available:", min(prices), "NPR")** **avg_price = sum(prices) / len(prices)** **print ("Average Regional Price:", round(avg_price, 2), "NPR")** • Ask children where they think the list has been created in this program and how it functions. • How do we know it is a list? (It uses square brackets). **Step 2: Guided programming (15 mins)** • Guide children in typing this code into Python, encouraging them to debug their work using the on-screen Python prompts. **Step 3: Extension (10 mins)** 1. Ask children to write a script that prompts the user to input 5 tomato prices dynamically using a loop and **.append()** 2. Print a list of only those prices where selling would yield a profit **(i.e. price > 90 NPR)**
Plenary	**Time: 10 minutes** 1. Summarise how variables, conditions, functions, loops and lists can work together in an agricultural management information system. 2. Ensure children save their work and log off their computers.

Lesson 24 – Introduction to Microbit Python	
Goal	**Goal 4 – Introduction to software logic (Microbit Python)** Video tutorial available here Wide range of Microbit tutorials available here **Microbit Python website:** https://python.microbit.org/v3
Learning objective	We are learning to use Python to program a Microbit device to collect data and display outputs using the LED matrix
UN Sustainable Development Goal	 **SDG 2: Zero Hunger (Target 2.4: Sustainable Food Production)** Optimising temperature for high-yield nursery crops and protecting delicate seedlings from heat stress, improving crop yields
Success Criteria	• I can open a new project in Microbit Python https://python.microbit.org/v3 • I can define a variable to store a string/number and display a scrolling message • I can construct an infinite loop (while True:) and use the sleep() function to prevent display buffer lag. • I can explain how variables hold temporary data in digital memory
Introduction	**Time: 15 minutes** 1. Ask the children how farmers in rural areas can use technology to monitor the status of their crops? *(They use devices that store this data as variables)*

	2. Explain that data is stored as variables and that devices use loops to keep checking conditions over and over again.
Main lesson	**Time: 35 minutes** **Step 1: Open the Microbit Python website and start a new project (5 mins)** • This is students first lesson using Microbit Python to program Microbit devices. Show the students how to navigate to **https://python.microbit.org/v/3** • Model the code that children will need to import the Microbit library. Explain that the * is used to import the whole library. • Model defining a variable and making it scroll across the screen. Show children how to connect their Microbit to the computer using a USB cable and how to initialise their program. Encourage them to experiment until the whole class is at this stage. **from microbit import *** **#Define variables** **crop_type = "Tomato Nursery"** **#Display dynamic text** **display.scroll(crop_type)** **Step 2: Implementing a loop with time controls (10 mins)** • Ask the children to read this program and make predictions about what it will do. Make time for questions and observations.

	from microbit import * **#Define variables** **crop_type = "Tomato Nursery"** **#Display dynamic text** **display.scroll(crop_type)** **while True:** **#Read built-in temperature sensor** **temp = temperature()** **#Display reading dynamically** **display.scroll("Temp:")** **display.scroll(temp)** **#Pause for 2000 milliseconds** **sleep(2000)**
Plenary	**Time: 10 minutes** • Show students the following line of code: **while true:** **#Read built-in temperature sensor** **temp = temperature()** • Ask students to point out where the error *is (it should be while True:)* • Discuss why we needed sleep (2000) inside the loop. (It protects the Microbit battery if it is not constantly checking)

Lesson 25 – Using conditions with a Microbit sensor	
Goal	**Goal 4 – Introduction to software logic (Microbit Python)** Video tutorial available here Wide range of Microbit tutorials available here **Microbit Python website:** https://python.microbit.org/v3
Learning objective	We are learning to read analogue pins and use conditional logic to trigger alerts
UN Sustainable Development Goal	 **SDG 2: Clean Water (Target 6.4 Water-use efficiency)** Automated moisture detection prevents over-watering, conserving scarce water resources in terrace farming systems.
Success Criteria	• I can open a new project in Microbit Python https://python.microbit.org/v3 • I can read the analogue pins using **pin0.read_analogue()** • I can use a conditional statement (if, else) and relational operators (<, >) to evaluate soil moisture • I can use elif to categorise soil conditions into three states: dry, optimal and flooded
Introduction	**Time: 15 minutes** 1. Show the children a Microbit with crocodile clips connected to pins **Pin 0** and **3V** on the Microbit. Connect nails to the ends of the crocodile clips and place them into dry soil. 2. Pour water over the dry soil and ask what might happen to the Microbit (the amount of electricity between the nails would increase).

Main lesson	**Time: 35 minutes** **Step 1: Connecting nails to the Microbit (5mins)** • Show the children how to clip a crocodile clip to the 3V pin on the Microbit. • Show the children how to clip another crocodile clip to pin 0 on the Microbit. • Show them how to attach nails to the wires and push them into soil, about 5cm apart. **Step 2: Programming the Microbit (15 mins)** • Show the children the following code and aske them to read it and make predictions: **from microbit import *** **while True:** **#Read analogue voltage from Pin 0** **moisture = pin0.read_analogue()** **print ("Moisture level:", moisture)** **sleep(500)** • Ask students to enter the code above, testing it and using the on-screen debugging tool to fix errors. • Encourage children to experiment and modify the code until all students are at this stage.

<table>
<tr><td></td><td>

Step 3: Adding conditional logic (15 mins)

- Show the children the following code and ask them to read it and make predictions:

```
from microbit import *

while True:

    moisture = pin0.read_analog()

    if moisture < 300:
        display.show(Image.SAD)

    elif moisture <=700:
        display.show(Image.HAPPY)

    else:
        display.show(Image.SURPRISED)

    sleep (1000)
```

- Ask students to enter the code above, testing it and using the on-screen debugging tool to fix errors.
- Encourage children to experiment and modify the code until all students are at this stage.

</td></tr>
<tr><td>Plenary</td><td>

Time: 10 minutes

1. Ask the students why the <= is used? What would happen if we just used < for the elif condition?
2. Ask students how they could modify this device, i.e. adding a motor to turn on a device.

</td></tr>
</table>

Lesson 26 – To use Boolean logic (or, and, not)	
Goal	**Goal 4 – Introduction to software logic (Microbit Python)** Video tutorial available here Wide range of Microbit tutorials available here **Microbit Python website:** https://python.microbit.org/v3
Learning objective	We are learning to control multiple sensors using the logical operators **and / or**
UN Sustainable Development Goal	 **SDG 2: Responsible Consumption (Target 12.3 Reduce post-harvest food loss)** Prevent heat and humidity spoilage in storage silos, reducing in lower post-harvest grain loss.
Success Criteria	• I can open a new project in Microbit Python https://python.microbit.org/v3 • I can use the conditional logic using and / or operators to make decisions • I can use a range of Boolean operators (and, or) • I can optimise my code based on testing and observation
Introduction	**Time: 15 minutes** 1. Discuss how some of the factors that plants need to grow well. (Temperature, moisture, light) 2. Explain that in programming, we can account for lots of conditions using Boolean logic: **or**, **and**, **not**. 3. Ask students to suggest some algorithms to write on the board using these statements.

<table>
<tr><td></td><td>i.e. "If the Moisture is ok and light is okay, display a happy face."</td></tr>
<tr><td>Main lesson</td><td>
Time: 35 minutes

Step 1: Using ‘or’ logic (25 mins)

• Students open a new project in Microbit Python https://python.microbit.org/v/3
• Show students the following code and give them time to read it, make predictions and ask questions:

from microbit import *

while True:

if button_a.is_pressed():
light_level = display.read_light_level()
temp_level = temperature()

print ("---GRAIN STORE CHECK"---")
print ("Light:", light_level, "Temp:", temp_level)

if light_level > 100 or temp_level > 30:
print ("STATUS: DANGER – Spoilage risk!")

else:
print ("STATUS:SAFE")
display.show(Image.YES)

sleep(2000)
display.clear()

sleep(100)

• Ask students to enter the code above, testing it and using the on-screen debugging tool to fix errors.
</td></tr>
</table>

	• Encourage children to experiment and modify the code until all students are at this stage. **Step 2: Using 'and' logic (15 mins)** • Show students the following code and give them time to read it, make predictions and ask questions: **if light_level > 100 and temp_level > 30:** **print("CRITICAL: Container open AND overheating")** • Ask students to modify their code with the code above and to experiment and modify the code.
Plenary	**Time: 10 minutes** 1. Ask the children how Boolean logic allows for greater possibilities while coding. 2. Can students suggest any other possible uses of Boolean logic?

Lesson 27 – Microbit Data logging and analysis	
Goal	**Goal 4 – Introduction to software logic (Microbit)** Video tutorial available here Wide range of Microbit tutorials available here
Learning objective	We are learning to store sequential readings into Python lists and apply searching/sorting algorithms to analyse crop data
UN Sustainable Development Goal	 **SDG 13: Climate Action (Target 13.1 Resilience to climate hazards)** Tracking soil temperature over time gives mountain farmers local data to adapt planting schedules to shifting climate patterns
Success Criteria	• I can open a new project in Microbit Python https://python.microbit.org/v/3 • I can log data entries using the log module and read logged files • I can implement a linear search algorithm to find the highest recorded temperature in a list
Introduction	**Time: 15 minutes** 1. Show students how data logged on a Microbit can be retrieved as a CSV file or graph when plugged into a computer. 2. Explain that instead of looking at one temperature reading at a time, farmers need to look at historical trends. 3. We use lists in Python to store collections of numbers over time.

Main lesson	Time: 35 minutes **Step 1: Initialise lists and data logging (25 mins)** • Students open a new project in Microbit Python https://python.microbit.org/v/3 • Show students the following code and give them time to read it, make predictions and ask questions: **from microbit import *** **import log** **log.set_labels('temperature', 'light')** **temp_history = []** • Ask students to enter the code above, testing it and using the on-screen debugging tool to fix errors.

Step 2: Collecting and logging data (10 mins)

- Show students the following code and give them time to read it, make predictions and ask questions:

```
from microbit import *
import log

log.set_labels ("soil_temp")
temp_history = []

display.scroll("Logging...")

for i in range(5):
        current_temp = temperature()

        temp_history.append(current_temp)
        log.add({"soil_temp": current_temp})

        print("Recorded Reading", i + 1, ":",
current_temp)

        sleep(2000)

display.show(Image.HAPPY)
```

- Ask students to enter the code above, testing it and using the on-screen debugging tool to fix errors.

Step 3: Processing the list (10 mins)

- Show students the following code and give them time to read it, make predictions and ask questions.
- In this section of code, we use a **custom linear search** to find the maximum temperature in temp_history:

```
max_temp = temp_history[0]

for temp in temp_history:


    if temp > max_temp:
        max_temp = temp

print ("Complete history", temp_history)
print ("Highest Recorded Soil Temp:", max_temp)
```

- Ask students to enter the code above, testing it and using the on-screen debugging tool to fix errors.

Step 4: Sorting the data (10 mins)

- Show students the following code and give them time to read it, make predictions and ask questions.
- In this section of code, we demonstrate Python's built-in sorting method to arrange readings from coldest to warmest:

```
temp_history.sort()
print ("Sorted Temperatures (Coldest to Warmest):", temp_history)
```

	• Ask students to enter the code above, testing it and using the on-screen debugging tool to fix errors.
Plenary	**Time: 10 minutes** 1. Connect the Microbit via USB to a computer and open the '**my data'** file. Look at the graph together. 2. Algorithmic thinking: If a list contains 100 daily soil readings, how many steps does a Linear Search take to find a specific readding in the worst-case scenario? *(100 comparisons)*.

Lesson 28 – Sending Microbit radio data	
Goal	**Goal 4 – Introduction to software logic (Microbit)** Video tutorial available here Wide range of Microbit tutorials available here
Learning objective	We are learning to use modular functions to process data and use the wireless radio module
UN Sustainable Development Goal	 **SDG 9: Innovation and Infrastructure** **(Target 9c: Access to ICT)** Building remote-to-valley wireless communication nodes demonstrates how rural areas can build low-cost ICT infrastructure
Success Criteria	• I can open a new project in Microbit Python https://python.microbit.org/v/3 • I can start my code with import microbit and import radio • I can send a text message when button A is pressed • I can trigger an icon on the screen when a message is detected • I can set my Microbit to match my partner's channel number
Introduction	**Time: 15 minutes** • Ask why it might be important to send information via radio from one device to another *(many farms are in remote locations, far from the crops they are monitoring)* • Explain that Microbits have the capability to send data via radio, if both are on the **same channel**!

Main lesson	**Time: 35 minutes** **Step 1: Programming the field sensor (transmitter) (15 minutes)** • Students open a new project in Microbit Python https://python.microbit.org/v/3 • Show students the following code and give them time to read it, make predictions and ask questions: **from microbit import *** **import radio** **radio.on** **radio.config(group=5)** **while True:** **if button_a.was_pressed():** **radio.send("DRY")** **sleep(1000)** **display.clear()** • Ask students to enter the code above, testing it and using the on-screen debugging tool to fix errors.

	Step 2: Programming the office sensor (receiver) (15 mins) **from microbit import *** **import radio** **radio.on** **radio.config(group=5)** **while True:** **message = radio.receive()** **if message == "DRY":** **display.show(Image.SAD)** **sleep(2000)** **display.clear()** • Ask students to enter the code above, testing it and using the on-screen debugging tool to fix errors.
Plenary	**Time: 10 minutes** 1. How could we stop other groups from interfering with our data? *(Change the channel number!)* 2. How could button B be programmed to send a different message? (This could extend into another lesson)

Objectives taught in this unit (highlighted)

Advanced level (age 13 – 16) – teaching objectives	
ICT Skills	**Lesson(s):**
1. Type accurately, at a speed of 30wpm, using both hands correctly	**3**
2. Master advanced file systems/: identify structural extensions (.py, .csv, .pdf, .zip) and manage local desktop vs. cloud directory.	**1**
3. Establish physical hardware connections using both Bluetooth and USB, connect a Microbit to another Microbit via radio	**28**
4. Create multi-page documents including custom page margins, table of contents, text alignment adjustments, page breaks.	**2**
5. Design agricultural and business spreadsheets that use comparative operators and conditional formulas, e.g. SUM, AVERAGE, COUNTIF.	**37, 38**
6. Deliver a high-impact presentation designed for a target audience using design tools and following conventions.	**9, 10**
7. Use principles of graphics design to create 2D images, i.e. alignment, contrast.	**40, 41, 42**
8. Use sophisticated, interactive online tools to research and critically evaluate information using this to further learning and investigations and create original content	**4, 5, 6, 7, 8, 11, 12, 39**
Digital literacy, AI literacy and E-safety	**Lesson(s):**
1. Explain specific legal frameworks governing technology use. Contrast open-source content with copyright infringement.	**1, 31**
2. Describe and utilise strategies to identify and counter the effects of screen-addition.	**32**
3. Analyse the effects of digital harassment, describing protocols to mitigate and report cyberbullying.	**32**
4. Cross reference primary sources to spot digital manipulation, deepfakes and automated disinformation campaigns.	**33**
5. Evaluate the local and global economic impacts of automation and artificial intelligence on rural labour markets and agricultural supply chains.	**35**
6. Outline the structural mechanics of machine learning, explaining concepts behind algorithms (i.e. decision trees and data pattern recognition).	**34, 35**
7. Critique the risks if modern generative models, covering training bias, data leaks, security risks and cognitive offloading.	**36**
8. Navigate the psychological hazards of anthropomorphised AI, explaining the risk of users forming inappropriate attachments to digital conversational agents.	**36**

Programming and computational thinking	Lesson(s):
1. Open an IDE (i.e. https://python.microbit.org/v/3/) and initialise a new script by importing the core hardware library (from microbit import *)	**13**
2. Use inputs (i.e. button_a.is_pressed()) and display output dynamically using the LED matrix (display.show()) and text scrolling	**24, 25, 26**
3. Use the built in functions in Python, including print(), input() and time.sleep()	**13, 14**
4. Read, process and convert analogue environmental inputs from alligator clip probes and onboard sensors (light, temperature, accelerometer)	**25, 26**
5. Use conditional structures (if, elif, else) and relational operators (<, =, >) to trigger outputs	**15, 16**
6. Use 'or', 'and', 'not' logic in conditional code and construct truth tables yo map hardware decision making	**16**
7. Program text-based looping structures (while True: and for I in range()), use timing controls (sleep()) and break commands to manage iteration	**17, 18**
7. Notice examples of variables in text-based code. Define and call variables within a program	**14, 15**
8. Construct and manipulate lists to store sequential datasets (e.g. calculating daily soil temperature)	**22, 23**
9. Use functions to organise code logically, defining and using functions that accept parameters, execute instructions and return processed data values	**19, 20, 21**
10. Use the radio module to transmit and receive data packets between multiple Microbits	**28**
11. Use on-screen information to identify types and locations of bugs	**13 - 28**
12. Convert values between binary and decimal, explaining how sensor readings are stored in digital memory	
13. Apply standard searching (linear) and sorting techniques to process and organise datasets of sensor or spreadsheet data	**27**
14. Explain how information is sent over networks using packets and network topology.	**29, 30**

Goal 5

UNDERSTANDING COMPUTER SYSTEMS AND NETWORKS

Lesson 29 – Components of a network	
Goal	**Goal 5 – Understanding computer systems and networks**
Learning objective	We are learning to identify the hardware components that are required to build a network and differentiate between Local Area Networks (LANs) and Wide Area networks (WANs).
UN Sustainable Development Goal	 **SDG 9: Industry, Innovation and Infrastructure (Target 9c: Significantly increase access to information communications technology)** Build digital infrastructure to ensure that every school has access to information.
Success Criteria	• I can identify core network hardware: Switch, Routers, Transmission cables) • I can state the difference between Local Area Networks (LANs) and Wide Area networks (WANs). • I can compare Star and Mesh topologies in terms of reliability, cable cost and failure and resilience • I can justify hardware choices based on the location of a school
Introduction	**Time: 15 minutes** 1. Ask the students to imagine a remote location in the mountains. 2. How would information from a website reach this remote location, where it might have to travel hundreds of kilometres across mountainous terrain? 3. Record students' ideas on the board.

Main lesson	**Time: 35 minutes** **Step 1: Introducing key concepts and vocabulary (15 minutes)** **Setup:** • Introduce students to the following terms. This could be through teacher explanation, or students matching the terms to their explanations: - **LAN (Local Area network)** A network over a small areas (i.e. inside your school computer lab or local office) - **WAN (Wide Area Network)** A network over a large geographical area connecting multiple towns and regions. - **Switch**: Connects local devices together inside a LAN using MAC addresses - **Router**: Connects different networks together (e.g. connecting a school LAN to the national internet service provider) - **Transmission media:** Optical fibre (high-speed light cables along highways). Copper (Ethernet cables inside rooms) and Wireless (Radio/Wi-Fi/Satellite) **Step 2: Unplugged activity (20 minutes)** 1. Assign students as roles within the classroom: - **3 students (or more)** Local laptops in the room - **1 student:** central switch - **1 student:** Router - **1 student**: remote server (e.g. national Health Database in Kathmandu) - Make string available to represent network connections between these points.

Help children discover the process of local transmission v wider transmission:

1. LAN mode: **Laptop A** wants to send a file to **Laptop C** inside the classroom. Laptop A hands a piece of paper directly to the **Switch**. The Switch reads the destination device tag and delivers it directly to Laptop C. (The traffic stays on the LAN)

2. LAN mode: Laptop A wants to request a medical report from Kathmandu. The Switch received the paper slip, recognises that it is not local and forwards it to the **Router**, The Router passes it across a mountain bridge tot e.g. **Remote Server.**

Step 3: Exploring Network Topology (10 mins)

Divide students into groups. Present them with this comparison grid and ask them to fill in the missing spaces (remove text from student grids):

Topology	How it Works	Strengths	Weaknesses	Best use
Star Topology	All computers connect directly to one central switch	If one computer's cable breaks, all other PCs keep working	If the main central Switch breaks, the whole room loses connection	School computer labs: Best used for connecting 10 – 20 PCs inside a classroom
Mesh Topology	Every device connects to multiple neighbouring devices	Highly reliable! If one path breaks or loses power, data automatically reroutes	Requires more cables or wireless transmitters	Hilltop villages: Perfect for connecting isolated homes across uneven ridge lines using wireless radio nodes.

Ask students to draw a network map connecting a School, Health Clinic and a local office situated across two hills separated by a river.

	Label where you want **Ethernet Cables**, **Wireless Access Points**, **Switches** and a **Router**.
Plenary	**Time: 10 minutes** 1. Ask the children, if a landslide occurs along the highway and cuts a main underground fibre optic line, will computers inside the school still be able to print to the printer? (Yes, Local traffic (LAN) is unaffected).

Lesson 30 – How the internet works	
Goal	**Goal 5 – Understanding computer systems and networks**
Learning objective	We are learning to explain how data is broken down into small packets, addressed with IP numbers, translated via DNS and routed across the internet
UN Sustainable Development Goal	 **SDG 2: Health and Wellbeing (Target 3.8: Achieve universal health coverage through tele-medicine and digital health systems)** Tele-medicine allows health information to be transmitted to specialists in distant hospitals
Success Criteria	• I can explain what an **IP address** is and explain why computers break large files down into small **packets** • I can explain the role if a **Domain Name System DNS)** in converting human web addresses into computer IP addresses • I can trace the step-by-step journey of a medical scan sent from a village to a specialist hospital in a city
Introduction	**Time: 15 minutes** 1. Present the students with a situation: *"If a local health worked takes an X-ray of an injured worker and there is no specialist doctor in the village, how can the injured worked get a diagnosis?"* (**Tele medicine** allows image files to be sent across a network.) 2. How do large images avoid clogging up networks?

Main lesson	**Time: 35 minutes** **Step 1: Unplugged activity – Packet switching relay (20 mins)** **Concept:** • Computers don't send large files like X-rays or videos as on massive file. They split them into small, numbered pieces called packets, route them along different paths and reassemble them at the destination. **Activity:** 1. Split the class into groups of 4: **Student 1:** Sender (Rural doctor) **Student 2**: Router via path A **Student 3**: Router via path B **Student 4:** Receiver (Hospital doctor in Kathmandu) 2. Step by step procedure: Sender takes a paper message: *"Urgent patient report needed"* and cuts it into 4 separate index cards (Packets). On each card, the sender writes a header: - **Sender IP**: 192.143.1.1 - **Destination IP**: 212.421.1.1 - **Packet number** (1 of 4), (2 of 4) etc. The sender hands packet 1 and 2 to Router Path A and packet 3 and 4 to Router path B. Both routers carry their packets to the receiver. The receiver receives the packets out of order. Using the packet numbers, the receiver rearranges them into the original message.

Activity

2. A student at the desk "The client laptop" hold up a message saying "Requesting rabbit story book"
3. They pass this message to the router in the middle
4. The router walks the message over to the server.
5. The server finds the paper "Rabbit Story book" in the box and the router delivers it back to the requesting student.

Teacher role – explain to the class that this happens every time a link is clicked on a computer!

Step 2: Exploring DNS (15 mins)

Concept:

Humans like words like https://www.un.org/sustainabledevelopment/ , but network routers only understand numbers, e.g. 192.132.13.1. The Domain Name System acts like a digital phonebook, translating names into IP addresses.

Activity:

1. Ask students to type a web address into their browser, i.e. https://www.un.org/sustainabledevelopment/ .
2. Ask them to right click and press ‘inspect’, click ‘network’ and then view ‘remote address’ to see the IP address of a page.
3. Discuss why computers use IP address and humans use DNS names.

Plenary	**Time: 10 minutes** • Show the following steps son the board and ask students to reorder them sequentially: - A. Receiver reassembles packets in numerical order - B. DNS server translates web address into an IP address - C. Filer is split into numbered packets with header info - D. Packets travel independently across different router paths *Correct sequence: B, C, D, A* • Discuss how packet switching and network protocols allow young farmers to build future solutions, such as remote disaster alerts or online farming advisories for their families.

Objectives taught in this unit (highlighted)

Advanced level (age 13 – 16) – teaching objectives	
ICT Skills	**Lesson(s):**
1. Type accurately, at a speed of 30wpm, using both hands correctly	**3**
2. Master advanced file systems/: identify structural extensions (.py, .csv, .pdf, .zip) and manage local desktop vs. cloud directory.	**1**
3. Establish physical hardware connections using both Bluetooth and USB, connect a Microbit to another Microbit via radio	**28**
4. Create multi-page documents including custom page margins, table of contents, text alignment adjustments, page breaks.	**2**
5. Design agricultural and business spreadsheets that use comparative operators and conditional formulas, e.g. SUM, AVERAGE, COUNTIF.	**37, 38**
6. Deliver a high-impact presentation designed for a target audience using design tools and following conventions.	**9, 10**
7. Use principles of graphics design to create 2D images, i.e. alignment, contrast.	**40, 41, 42**
8. Use sophisticated, interactive online tools to research and critically evaluate information using this to further learning and investigations and create original content	**4, 5, 6, 7, 8, 11, 12, 39**
Digital literacy, AI literacy and E-safety	**Lesson(s):**
1. Explain specific legal frameworks governing technology use. Contrast open-source content with copyright infringement.	**1, 31**
2. Describe and utilise strategies to identify and counter the effects of screen-addition.	**32**
3. Analyse the effects of digital harassment, describing protocols to mitigate and report cyberbullying.	**32**
4. Cross reference primary sources to spot digital manipulation, deepfakes and automated disinformation campaigns.	**33**
5. Evaluate the local and global economic impacts of automation and artificial intelligence on rural labour markets and agricultural supply chains.	**35**
6. Outline the structural mechanics of machine learning, explaining concepts behind algorithms (i.e. decision trees and data pattern recognition).	**34, 35**
7. Critique the risks if modern generative models, covering training bias, data leaks, security risks and cognitive offloading.	**36**
8. Navigate the psychological hazards of anthropomorphised AI, explaining the risk of users forming inappropriate attachments to digital conversational agents.	**36**

Programming and computational thinking	Lesson(s):
1. Open an IDE (i.e. https://python.microbit.org/v/3/) and initialise a new script by importing the core hardware library (from microbit import *)	**13**
2. Use inputs (i.e. button_a.is_pressed()) and display output dynamically using the LED matrix (display.show()) and text scrolling	**24, 25, 26**
3. Use the built in functions in Python, including print(), input() and time.sleep()	**13, 14**
4. Read, process and convert analogue environmental inputs from alligator clip probes and onboard sensors (light, temperature, accelerometer)	**25, 26**
5. Use conditional structures (if, elif, else) and relational operators (<, =, >) to trigger outputs	**15, 16**
6. Use 'or', 'and', 'not' logic in conditional code and construct truth tables to map hardware decision making	**16**
7. Program text-based looping structures (while True: and for I in range()), use timing controls (sleep()) and break commands to manage iteration	**17, 18**
7. Notice examples of variables in text-based code. Define and call variables within a program	**14, 15**
8. Construct and manipulate lists to store sequential datasets (e.g. calculating daily soil temperature)	**22, 23**
9. Use functions to organise code logically, defining and using functions that accept parameters, execute instructions and return processed data values	**19, 20, 21**
10. Use the radio module to transmit and receive data packets between multiple Microbits	**28**
11. Use on-screen information to identify types and locations of bugs	**13 - 28**
12. Convert values between binary and decimal, explaining how sensor readings are stored in digital memory	
13. Apply standard searching (linear) and sorting techniques to process and organise datasets of sensor or spreadsheet data	**27**
14. Explain how information is sent over networks using packets and network topology.	**29, 30**

Goal 6

DIGITAL LITERACY, E-SAFETY AND AI

Lesson 31 – Digital ownership	
Goal	**Goal 6 – Digital literacy, E-safety and AI**
Learning objective	We are learning to analyse legal frameworks governing digital content and to contrast copyright law with open-source licensing
UN Sustainable Development Goal	 **SDG 9: Industry, Innovation and Infrastructure** **(Target 9c: Significantly increase access to information communications technology)** •Use open-source technology to allow low-income regions to access digital infrastructure
Success Criteria	• I can define copyright, public domain and open-source licensing • I can explain the legal consequences of copyright infringement v compliant open-source reuse • I can recommend an appropriate Creative Commons or open-source licence for a community tech project
Introduction	• Show students two digital items: A commercial software product/popular song and an open-source tool (e.g. Python or Linux). • Ask the students *"If you modify and share both online, which one would get you sued and why?"* • Discuss ownership of digital products and why creators are protected legally.
Main lesson	**Time: 35 minutes** **Step 1: Legal framework breakdown (10 minutes)** • Introduce key legal terms: **Copyright**, **Fair use, Public Domain** and **Open Source** (e.g. MIT, GNU, Creative Commons).

	• Provide six situation cards (e.g. "Downloading a paid movie onto a USB drive", "Using an MIT licenced Python script in a school project.") • Students categorise them into legal, Infringement or permissible open source) **Step 2: Open-source v Proprietary debate (15 minutes)** • Divide the class into two teams: **Team Copyright** (defending creator profits and intellectual property rights) and **Team Open Source** (defending rapid global innovation and accessible software under SDG 9) • Conduct a structured debate on *"Should all agricultural software code be open-source?"* **Step 3: Licensing a digital project (15 minutes)** • Students inspect a sample code repository / creative work on paper or screen • Students select an appropriate Creative Commons licence (CC-BY, CC-NC, CC-SA) based on project requirements (e.g. a community farming app designed for free local use).
Plenary	**Time: 10 minutes** Ask the students to write down one major difference between copyright infringement and using open-source content and why open-source content helps achieve SDG 9.

Lesson 32 – Digital health and wellbeing	
Goal	**Goal 6 – Digital literacy, safety and AI**
Learning objective	We are learning to understand the causes of screen addiction and digital harassment and to generate strategies for dealing with and reporting both
UN Sustainable Development Goal	 **SDG 2: Good Health and Well-being** **(Target 3.4 Promote mental health and well-being)** • Promote mental health and well-being as essential aspects of a healthy life
Success Criteria	• I can identify the signs and symptoms of screen addiction and spot ways that apps manipulate dopamine to capture attention • I can differentiate between cyberbullying and forms of banter • I can formulate a reporting protocol designed to protect myself and other members of the school community.
Introduction	• Display some statistics about screen usage, including among young people. • Ask students if they are surprised by the statistics. • Ask students about the design features that keep people using apps (notifications, sounds, infinite scroll, streak counters) • Talk about how mental health is as important as physical health and that we must look after ourselves when using technology,
Main lesson	**Time: 35 minutes** **Step 1: Explaining neuroscience behind addiction (10 minutes)**

	• Explain how dopamine is released when beneficial activities take place. Explain the difference between healthy dopamine release caused by activities like walking and sports, and short-term unhealthy release, caused by triggers in software and foods.
	• Ask students to self-identify three pieces of commonly used software. Ask student to write down as many ways that these trigger dopamine release.
	
	Step 2: Managing screen-addiction (15 minutes)
	
	• In pairs, students add notes to the features that they have identified and write down counter measures for them. For example: Notifications – turn off notifications, turn off device.
	
	Step 3: Cyberbullying protocols (15 minutes)
	
	• Show children a list of different forms of cyberbullying. Discuss, or elicit their definitions. Include:
	  • **Harassment** (contacting someone against their will, or sending insulting messages)
	  • **Flaming** (using strong language against someone)
	  • **Outing** (sharing someone's secrets or personal information. Could include embarrassing photos)
	  • **Impersonation** (pretending to be someone else)
	  • **Exclusion** (stopping someone from joining in activities)
	  • **Denigration –** damaging someone's reputation by spreading rumours

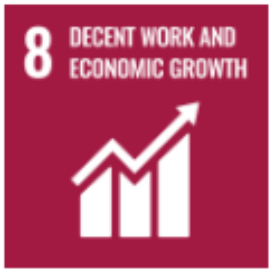
	• Discuss the four-step emergency protocol: • **Stop** (don't reply or retaliate) • **Block** (block the account or app) • **Tell** (tell a trusted adult) • **Report** (use in-app tools to report and/or save evidence) • As a class, discuss a situation where cyberbullying was taking place, but the perpetrator tried to suggest this was just 'banter'. Discuss the difference between 'banter' and cyberbullying. Talk about the mental health consequences of cyberbullying and consequences for the perpetrator. • Guide the class through paired discussion of situation of either cyberbullying or banter. Ask students to decide in pairs which (if any) category of cyberbullying the behaviour gits into and what the victim should do.
Plenary	**Time: 10 minutes** • Ask the children to feedback on their paired discussions and provide opportunity for discussion. • Remind the students that healthy and positive use of technology is in all of our interests.

Lesson 33 – Identifying misinformation	
Goal	**Goal 6 – Digital literacy, safety and AI**
Learning objective	We are learning to verify information and spot 'deepfakes', manipulated media and disinformation
UN Sustainable Development Goal	 **SDG 16: Peace, Justice and Strong Institutions** **(Target 16.10: Ensure public access to information and protect fundamental freedoms)** • Avoiding disinformation which undermines public trust, free elections and social stability
Success Criteria	• I can explain the terms 'deepfake' and 'disinformation' • I can cross reference sources to verify the authenticity of online news or images • I can identify aspects of AI media and explain the threat of disinformation
Introduction	• Display to news reports side by side, one that is genuine and one that is AI generated. • Ask the students to suggest which is genuine and how we can know. • Discuss the problems that AI generated misinformation can cause (on trust, reputation and political outcomes).
Main lesson	**Time: 35 minutes** **Step 1: AI content forensics (10 minutes)** • Guide the students through some of the signs of AI generated media: unnatural lighting/reflections, irregular finger/ear geometry, warped background textures, voice synthesis pitch anomalies • Look at some examples of AI generated news stories, images, and video. Discuss whether we

	can identify these as being AI generated. Discuss how AI technology is getting more sophisticated and these sigs are getting harder to spot. • Model the process of looking at AI generated posts/media and cross referencing these against primary sources. **Step 2: Cross referencing with primary sources (5 minutes)** • Provide the students with examples of news stories/media. • Ask the students to work in groups or pairs and follow the process of cross referencing, by performing searches through reliable websites. **Step 3: Mapping disinformation waves (15 minutes)** • As a class, examine a diagram of how bots can amplify disinformation through repeated posting and coordinate hashtag hijacking. • Students annotate a diagram showing how fake news travels from a single bot account to mainstream social media feeds.
Plenary	**Time: 10 minutes** • Show the class three online comments, ask students to vote upon which is posted by a human and which was posted by an automated bot. Ask students to justify their responses using text patterns.

Lesson 34 – Understanding machine learning	
Goal	**Goal 6 – Digital literacy, safety and AI**
Learning objective	We are learning to explain how machine learning works, using data classification, pattern recognition and decision trees
UN Sustainable Development Goal	 **SDG 2: Life Below Water (Target 14.1: Prevent and significantly reduce marine pollution of all kinds)** • Use technology to help clean oceans and protect marine ecosystems form pollution
Success Criteria	• I can explain the difference between pattern learning and explicit rule-programming • Apply this understanding by training an AI model and describe how decision trees use branching data structures • Explain how human data selection introduces algorithmic bas into machine learning tools
Introduction	• Show the students a decision tree diagram: Is it round? → Yes → is it orange? → yes → It is an orange. • Ask how computers can figure out rules on their own without a human coding every stage. • Introduce the concept of machine learning, in contract to humans writing rules for every stage of the decision process.
Main lesson	**Time: 35 minutes** **Step 1: Code.org AI for Oceans (20 minutes)** • Direct students to access the AI for Oceans activity on code.org https://studio.code.org/courses/oceans/units/1/lessons/1/levels/1

	• In levels 1 – 4, children learn about how they can train an AI model to distinguish between fish and rubbish in the ocean. • Children observe how the more data they provide, the better the model gets, although it still makes mistakes. • In levels 5 – 7, children train an AI model using more subjective terms. They are introduced to the idea that their own human bias and preconceptions can affect an AI model. **Step 2: Application in society (15 minutes)** • Introduce other contexts and uses for machine learning in society. • Introduce the context of AI tools being used to screen CVs in employment recruiting. Ask the students why this can be useful and save time, and how issues of bias could affect company hiring practises.
Plenary	**Time: 10 minutes** • Ask the children how machine learning might affect us in the context of agriculture. Could machine learning help identify issues on farms? How could human bias affect this process?

Lesson 35 – Risks associated with generative AI	
Goal	**Goal 6 – Digital literacy, safety and AI**
Learning objective	We are learning to find, position and rest our fingers on the keyboard's 'home row' to type letters accurately without looking at our hands.
UN Sustainable Development Goal	 **SDG 8: Decent work and Economic Growth (Target 8.2: Achieve higher levels of economic productivity through diversification, technological upgrading and innovation)** • Enabling people to apply their skills and innovation to society
Success Criteria	• I can identify 3 risks associated with generative AI; bias, data privacy leaks and cognitive offloading • I can explain how automation affects the rural workforce • I can propose a regional policy which integrates technology into agriculture while preserving rural communities
Introduction	• Show the students a video clip of an autonomous tractor harvesting crops. • Show the students a clip of a generative AI tool interacting with a human. • Ask the students, *"How are these tools changing the way that we work and what are the implications for rural, farming communities?"*
Main lesson	**Time: 35 minutes** **Step 1: Impact of automation on agricultural labour (15 minutes)** • Provide the students with a case study on how automation is being used in farming. • Ask the students to frame a response in terms of 'pros and cons'.

	• Students might come up with the following pros: - Increased efficiency leading to greater food production - Less use of pesticides with targeted interventions - Saving water with smart watering systems - Allowing agricultural workers more time to develop their business and innovation • Students might come up with the following cons: - Security risk if automated systems are hacked - Expense of agricultural systems - Replacing certain repetitive jobs - Data being leaked - Greater division of wealth between technology users and non-technology users **Step 2: Exploring risks associated with generative AI (15 minutes)** • Rotate students through reading and discussion of three case studies: **Study A** Students read about and discuss a situation where an AI recruitment tool was biased against certain demographics, or when confidential company code was leaked via prompts. **Study B** Students read about the position of Norway, which has restricted children from using generative AI on the grounds of cognitive deskilling and relationships with AI tools

	Study C Students look at example of AI hallucinations and the impact that confident false answers have had on work.
Plenary	**Time: 10 minutes** • Ask the children what they think about cognitive offloading. Is it a threat to education, or is allowing students to become skilled in other areas such as AI literacy?

Lesson 36 – Risks of anthropomorphised AI	
Goal	**Goal 6 – Digital literacy, safety and AI** Teachers should think carefully about how to discuss conversational AI tools with children, without introducing them to tools that they are not old enough to use safely.
Learning objective	We are learning to counter the risks or anthropomorphised conversational AI and how to set boundaries against inappropriate digital attachments
UN Sustainable Development Goal	 **SDG 2: Good Health and Well-being** **(Target 3.4 Promote mental health and well-being)** • Promote mental health and well-being as essential aspects of a healthy life
Success Criteria	• I can explain what anthropomorphism means in the context of technology and give examples • I can explain why humans sometimes form relationships with AI agents and the risks of this • I can draft safety guidelines for teenagers interacting with conversational AI tools
Introduction	• Display a chat transcript on the screen between a human and a conversational AI tool. • Discuss aspects of this chat that are anthropomorphised. • Why do companies build AI tools that sound so human? What kind of problems could this create?
Main lesson	**Time: 35 minutes** **Step 1: Explaining how LLMs work (15 minutes)**

	• Explain that conversations LLMs work by predicting the statistically most likely next word and do not feel emotions. • Discuss design choices that can confuse people, especially young people, including use of words like "I" and "Me", as well as simulated empathetic phrases. **Step 2: Exploring scenarios (15 minutes)** • Guide children through exploration of two scenarios relating to conversational AI. For each scenario, discuss what has happened and what should happen. Emphasise that age limits exist with AI tools for good reasons and that students should seek adult help if they feel uncomfortable or worried when using technology. **Scenario A** Someone spends less time interacting with friends and family members and more time talking with conversational AI tools, particularly in the evening. **Scenario B** Someone shares their personal information with an AI tool, asking for advice on health issues. **Step 3: Drafting boundaries (10 minutes)** • Students work together to produce a set of rules or guidelines for being safe around AI tools.
Plenary	**Time: 10 minutes** • Ask the children to share their guidelines and ask questions about the suggestions of other groups.

Objectives taught in this unit (highlighted)

Advanced level (age 13 – 16) – teaching objectives	
ICT Skills	**Lesson(s):**
1. Type accurately, at a speed of 30wpm, using both hands correctly	**3**
2. Master advanced file systems/: identify structural extensions (.py, .csv, .pdf, .zip) and manage local desktop vs. cloud directory.	**1**
3. Establish physical hardware connections using both Bluetooth and USB, connect a Microbit to another Microbit via radio	**28**
4. Create multi-page documents including custom page margins, table of contents, text alignment adjustments, page breaks.	**2**
5. Design agricultural and business spreadsheets that use comparative operators and conditional formulas, e.g. SUM, AVERAGE, COUNTIF.	**37, 38**
6. Deliver a high-impact presentation designed for a target audience using design tools and following conventions.	**9, 10**
7. Use principles of graphics design to create 2D images, i.e. alignment, contrast.	**40, 41, 42**
8. Use sophisticated, interactive online tools to research and critically evaluate information using this to further learning and investigations and create original content	**4, 5, 6, 7, 8, 11, 12, 39**
Digital literacy, AI literacy and E-safety	**Lesson(s):**
1. Explain specific legal frameworks governing technology use. Contrast open-source content with copyright infringement.	**1, 31**
2. Describe and utilise strategies to identify and counter the effects of screen-addition.	**32**
3. Analyse the effects of digital harassment, describing protocols to mitigate and report cyberbullying.	**32**
4. Cross reference primary sources to spot digital manipulation, deepfakes and automated disinformation campaigns.	**33**
5. Evaluate the local and global economic impacts of automation and artificial intelligence on rural labour markets and agricultural supply chains.	**35**
6. Outline the structural mechanics of machine learning, explaining concepts behind algorithms (i.e. decision trees and data pattern recognition).	**34, 35**
7. Critique the risks if modern generative models, covering training bias, data leaks, security risks and cognitive offloading.	**36**
8. Navigate the psychological hazards of anthropomorphised AI, explaining the risk of users forming inappropriate attachments to digital conversational agents.	**36**

Programming and computational thinking	Lesson(s):
1. Open an IDE (i.e. https://python.microbit.org/v/3/) and initialise a new script by importing the core hardware library (from microbit import *)	**13**
2. Use inputs (i.e. button_a.is_pressed()) and display output dynamically using the LED matrix (display.show()) and text scrolling	**24, 25, 26**
3. Use the built in functions in Python, including print(), input() and time.sleep()	**13, 14**
4. Read, process and convert analogue environmental inputs from alligator clip probes and onboard sensors (light, temperature, accelerometer)	**25, 26**
5. Use conditional structures (if, elif, else) and relational operators (<, =, >) to trigger outputs	**15, 16**
6. Use 'or', 'and', 'not' logic in conditional code and construct truth tables to map hardware decision making	**16**
7. Program text-based looping structures (while True: and for I in range()), use timing controls (sleep()) and break commands to manage iteration	**17, 18**
7. Notice examples of variables in text-based code. Define and call variables within a program	**14, 15**
8. Construct and manipulate lists to store sequential datasets (e.g. calculating daily soil temperature)	**22, 23**
9. Use functions to organise code logically, defining and using functions that accept parameters, execute instructions and return processed data values	**19, 20, 21**
10. Use the radio module to transmit and receive data packets between multiple Microbits	**28**
11. Use on-screen information to identify types and locations of bugs	**13 - 28**
12. Convert values between binary and decimal, explaining how sensor readings are stored in digital memory	
13. Apply standard searching (linear) and sorting techniques to process and organise datasets of sensor or spreadsheet data	**27**
14. Explain how information is sent over networks using packets and network topology.	**29, 30**

Goal 7

AGRICULTURAL FINANCIAL LITERACY

Lesson 37	
Goal	**Goal 7 – Agricultural financial literacy**
Learning objective	We are learning to build a dynamic farm income and expense model using comparative logic and functions in a spreadsheet
UN Sustainable Development Goal	 **SDG 2: Zero Hunger (Target 2.3: Double agricultural productivity…through resources, inputs, knowledge, financial services)** Creating digital systems to evaluate the financial viability of a harvest
Success Criteria	• I can set up a spreadsheet with numerical data, header and units of currency • I can use the SUM and Average formulas to calculate total net profit • I can use comparative operators (>0, <0) within conditional formatting rules to highlight profitable and loss-making crop cycles
Introduction	**Time: 15 minutes** 1. Present the students with a scenario of a farmer who records seasonal yields for Rice, maize and Vegetables. 2. Ask whether if a farmer sells $1,300 worth of maize, but spends $800 on seed and fertiliser, are they making enough money to reinvest for next season? 3. Introduce the terms: **Revenue** (money in), **Expenses** (money out) and **Net Profit** (Revenue minus expenses.)

Main lesson

Time: 35 minutes

Step 1: Building a spreadsheet (5 mins)

- Instruct the children to open a new spreadsheet.
- Guide students to create the following table:

Crop type	Area (Sqm)	Harvest yield (kg)	Total revenue ($)	Input costs ($)	Net profit ($)	Viability check
Rice	5	3000	1,200	980	=D2-E2	
Tomatoes	7	4500	2,400	200	=D3-E3	
Corn	8	2300	3,400	1000	=D4-E4	
Totals			=SUM(D2:D4)	=SUM(E2:E4	=SUM(F2:F4	
Averages		=AVERARAGE(C2:C4)	=AVERARAGE(D2:D4)	=AVERARAGE(E2:E4)	=AVERARAGE(F2:F4)	

Step 2: Using conditional logic (15 mins)

- Discuss with children the logic that could form our viability check. This needs to be, whether the net profit is more than 0, in other words, =IF(F2:>0, "PROFITABLE", "LOSS")
- Help students make the connection between conditions in a spreadsheet and in programming.
- Ask them to write the formulas for the other crops.

Step 3: Conditional formatting (15 mins)

- Show the children how to add conditional formatting in a spreadsheet.
- Model setting up conditional formatting so that loss making cells are highlighted red and profit making cells are highlighted green.

Plenary	**Time: 10 minutes** • Ask the children to calculate the total profit with a formula. • Ask the children if the sot of fertiliser raises by 30% next season, how would we adjust our spreadsheet? • Ensure students save their spreadsheets for use next lesson.

Lesson 38	
Goal	**Goal 7 – Agricultural financial literacy**
Learning objective	We are learning to use the formulas COUNTIF and SUMIF to evaluate crop production
UN Sustainable Development Goal	 **SDG 2: Zero Hunger** **(Target 2.3: Double agricultural productivity…through resources, inputs, knowledge, financial services)** Creating digital systems to evaluate the financial viability of a harvest
Success Criteria	• I can expand the spreadsheet I made last lesson to include additional tags • I can use the **COUNTIF** formula to tally specified harvest conditions and he **SUMIF** formula to aggregate total crop yields • I can use **AVERAGEIF** with comparisons (i.e. >1500) to identify high-performing areas
Introduction	**Time: 15 minutes** • Explain to children that sometimes farmers need to be able to quickly draw conclusions about the status of all of their crops. • To do this, we can use the conditional formulas, COUNTIF and SUMIF to count, only if other conditions are met
Main lesson	**Time: 35 minutes** **Step 1: Modifying last week's spreadsheet (5 mins)** • Instruct the children to open the spreadsheets they made in the last lesson. • Ask students to modify their spreadsheets by adding the following rows and columns:

Last week's spreadsheet:

Crop type	Area (Sqm)	Harvest yield (kg)	Total revenue ($)	Input costs ($)	Net profit ($)	Viability check
Rice	5	3000	1,200	980	=D2-E2	
Tomatoes	7	4500	2,400	200	=D3-E3	
Corn	8	2300	3,400	1000	=D4-E4	
Totals			=SUM(D2:D4)	=SUM(E2:E4	=SUM(F2:F4	
Averages		=AVERARAGE(C2:C4)	=AVERARAGE(D2:D4)	=AVERARAGE(E2:E4)	=AVERARAGE(F2:F4)	

Students add **additional rows** with other vegetables.

Updated spreadsheet:

Crop type	Area (Sqm)	Harvest yield (kg)	Total revenue ($)	Input costs ($)	Net profit ($)	Viability check
Rice	5	3000	1,200	980	=D2-E2	
Tomatoes	7	4500	2,400	200	=D3-E3	
Rice	4	6000	1,200	600	=D4-E4	
Maize	6	3400	1,800	600	=D5-E5	
Ginger	7	5500	1,400	400	=D6-E6	
Corn	8	2300	3,400	1000	= D7-E7	
Totals			=SUM(D2:D7)	=SUM(E2:E7	=SUM(F2:F7	
Averages		=AVERARAGE(C2:C7)	=AVERARAGE(D2:D7)	=AVERARAGE(E2:E7)	=AVERARAGE(F2:F7)	

Step 2: Using COUNTIF and SUMIF formulas (15 mins)

- Show children how to count all the profitable crops, by adding a formula =COUNTIF(f2:f7>0) (Counting all the crops that are profitable).
- Show children how to use the SUMIF formula to find the total of all profitable crops, using the formula =SUMIF(f2:f7>0)

Plenary

Time: 10 minutes

- Ask students to evaluate the useful ness of their spreadsheet for calculating profitability.
- What else would we add to this spreadsheet?

Objectives taught in this unit (highlighted)

Advanced level (age 13 – 16) – teaching objectives	
ICT Skills	**Lesson(s):**
1. Type accurately, at a speed of 30wpm, using both hands correctly	**3**
2. Master advanced file systems/: identify structural extensions (.py, .csv, .pdf, .zip) and manage local desktop vs. cloud directory.	**1**
3. Establish physical hardware connections using both Bluetooth and USB, connect a Microbit to another Microbit via radio	**28**
4. Create multi-page documents including custom page margins, table of contents, text alignment adjustments, page breaks.	**2**
5. Design agricultural and business spreadsheets that use comparative operators and conditional formulas, e.g. SUM, AVERAGE, COUNTIF.	**37, 38**
6. Deliver a high-impact presentation designed for a target audience using design tools and following conventions.	**9, 10**
7. Use principles of graphics design to create 2D images, i.e. alignment, contrast.	**40, 41, 42**
8. Use sophisticated, interactive online tools to research and critically evaluate information using this to further learning and investigations and create original content	**4, 5, 6, 7, 8, 11, 12, 39**
Digital literacy, AI literacy and E-safety	**Lesson(s):**
1. Explain specific legal frameworks governing technology use. Contrast open-source content with copyright infringement.	**1, 31**
2. Describe and utilise strategies to identify and counter the effects of screen-addition.	**32**
3. Analyse the effects of digital harassment, describing protocols to mitigate and report cyberbullying.	**32**
4. Cross reference primary sources to spot digital manipulation, deepfakes and automated disinformation campaigns.	**33**
5. Evaluate the local and global economic impacts of automation and artificial intelligence on rural labour markets and agricultural supply chains.	**35**
6. Outline the structural mechanics of machine learning, explaining concepts behind algorithms (i.e. decision trees and data pattern recognition).	**34, 35**
7. Critique the risks if modern generative models, covering training bias, data leaks, security risks and cognitive offloading.	**36**
8. Navigate the psychological hazards of anthropomorphised AI, explaining the risk of users forming inappropriate attachments to digital conversational agents.	**36**

Programming and computational thinking	Lesson(s):
1. Open an IDE (i.e. https://python.microbit.org/v/3/) and initialise a new script by importing the core hardware library (from microbit import *)	**13**
2. Use inputs (i.e. button_a.is_pressed()) and display output dynamically using the LED matrix (display.show()) and text scrolling	**24, 25, 26**
3. Use the built in functions in Python, including print(), input() and time.sleep()	**13, 14**
4. Read, process and convert analogue environmental inputs from alligator clip probes and onboard sensors (light, temperature, accelerometer)	**25, 26**
5. Use conditional structures (if, elif, else) and relational operators (<, =, >) to trigger outputs	**15, 16**
6. Use 'or', 'and', 'not' logic in conditional code and construct truth tables yo map hardware decision making	**16**
7. Program text-based looping structures (while True: and for I in range()), use timing controls (sleep()) and break commands to manage iteration	**17, 18**
7. Notice examples of variables in text-based code. Define and call variables within a program	**14, 15**
8. Construct and manipulate lists to store sequential datasets (e.g. calculating daily soil temperature)	**22, 23**
9. Use functions to organise code logically, defining and using functions that accept parameters, execute instructions and return processed data values	**19, 20, 21**
10. Use the radio module to transmit and receive data packets between multiple Microbits	**28**
11. Use on-screen information to identify types and locations of bugs	**13 - 28**
12. Convert values between binary and decimal, explaining how sensor readings are stored in digital memory	
13. Apply standard searching (linear) and sorting techniques to process and organise datasets of sensor or spreadsheet data	**27**
14. Explain how information is sent over networks using packets and network topology.	**29, 30**

Goal 8

AGRICULTURAL SCIENCE

Lesson 39	
Goal	**Goal 8 – Agricultural science** (via **E-Pustakalaya or Wikipedia**)
Learning objective	We are learning to identify soil nutrients (Nitrogen, Phosphorus and Potassium) and use electronic tools to research ways to enrich soil
UN Sustainable Development Goal	 **SDG 2: Zero Hunger (Target 2.4: Sustainable Food Production)** • Optimising conditions for high-yield crops and protecting seedlings from stress, improving crop yields
Success Criteria	• I can identify key soil nutrients (Nitrogen, Phosphorus and Potassium) • I can use E-Pustakalaya to research techniques for enriching soil • I can design a crop rotation timetable for a hilly agricultural zone
Introduction	1. Ask students what they know about the conditions that plants need to grow in. 2. How much do factors like soil conditions, light, moisture and other factors affect plant growth? 3. Explain that in this lesson we will use digital resources to research ideal soil conditions and priorities for plant growth.
Main lesson	**Time: 35 minutes** **Step 1: Accessing E-Pustakalaya (5 mins)** • Ask the students to open E-Pustakalaya and search for "soil fertility". • Students should open the ‘Course Materials’, Grades 8/9/10 → Science → Business and Technology Education

	Step 2: Carrying out research (10 mins) • Guide the students through the research task. Ask students to answer two questions: **1) What are the symptoms of nitrogen deficiency in plants?** **2) How are natural fertilisers made using locally available plants?** Ask students to open a Word document and to type the answers to these questions. **Step 3: Evaluating fertilisers (10 mins)** • Ask students to add a table into their Word document. • One column of the table should be 'natural fertilisers' and the other 'chemical fertilisers' • One row should be 'advantages', the other 'disadvantages' • Ask students to use E-Pustakalaya to till in thus table, giving examples of specific crops as well as general answers. • Ask students to share their findings with the rest of the class.
Plenary	**Time: 10 minutes** 1. Bring the class together. Ask how research has enables them to better understand the needs of crops. 2. Save documents and close down computers safely.

Objectives taught in this unit (highlighted)

Advanced level (age 13 – 16) – teaching objectives	
ICT Skills	**Lesson(s):**
1. Type accurately, at a speed of 30wpm, using both hands correctly	**3**
2. Master advanced file systems/: identify structural extensions (.py, .csv, .pdf, .zip) and manage local desktop vs. cloud directory.	**1**
3. Establish physical hardware connections using both Bluetooth and USB, connect a Microbit to another Microbit via radio	**28**
4. Create multi-page documents including custom page margins, table of contents, text alignment adjustments, page breaks.	**2**
5. Design agricultural and business spreadsheets that use comparative operators and conditional formulas, e.g. SUM, AVERAGE, COUNTIF.	**37, 38**
6. Deliver a high-impact presentation designed for a target audience using design tools and following conventions.	**9, 10**
7. Use principles of graphics design to create 2D images, i.e. alignment, contrast.	**40, 41, 42**
8. Use sophisticated, interactive online tools to research and critically evaluate information using this to further learning and investigations and create original content	**4, 5, 6, 7, 8, 11, 12, 39**
Digital literacy, AI literacy and E-safety	**Lesson(s):**
1. Explain specific legal frameworks governing technology use. Contrast open-source content with copyright infringement.	**1, 31**
2. Describe and utilise strategies to identify and counter the effects of screen-addition.	**32**
3. Analyse the effects of digital harassment, describing protocols to mitigate and report cyberbullying.	**32**
4. Cross reference primary sources to spot digital manipulation, deepfakes and automated disinformation campaigns.	**33**
5. Evaluate the local and global economic impacts of automation and artificial intelligence on rural labour markets and agricultural supply chains.	**35**
6. Outline the structural mechanics of machine learning, explaining concepts behind algorithms (i.e. decision trees and data pattern recognition).	**34, 35**
7. Critique the risks if modern generative models, covering training bias, data leaks, security risks and cognitive offloading.	**36**
8. Navigate the psychological hazards of anthropomorphised AI, explaining the risk of users forming inappropriate attachments to digital conversational agents.	**36**

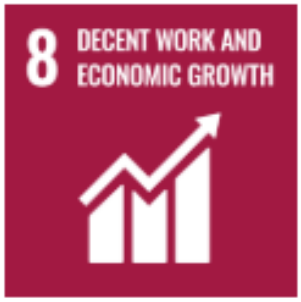
Programming and computational thinking	Lesson(s):
1. Open an IDE (i.e. https://python.microbit.org/v/3/) and initialise a new script by importing the core hardware library (from microbit import *)	**13**
2. Use inputs (i.e. button_a.is_pressed()) and display output dynamically using the LED matrix (display.show()) and text scrolling	**24, 25, 26**
3. Use the built in functions in Python, including print(), input() and time.sleep()	**13, 14**
4. Read, process and convert analogue environmental inputs from alligator clip probes and onboard sensors (light, temperature, accelerometer)	**25, 26**
5. Use conditional structures (if, elif, else) and relational operators (<, =, >) to trigger outputs	**15, 16**
6. Use 'or', 'and', 'not' logic in conditional code and construct truth tables yo map hardware decision making	**16**
7. Program text-based looping structures (while True: and for I in range()), use timing controls (sleep()) and break commands to manage iteration	**17, 18**
7. Notice examples of variables in text-based code. Define and call variables within a program	**14, 15**
8. Construct and manipulate lists to store sequential datasets (e.g. calculating daily soil temperature)	**22, 23**
9. Use functions to organise code logically, defining and using functions that accept parameters, execute instructions and return processed data values	**19, 20, 21**
10. Use the radio module to transmit and receive data packets between multiple Microbits	**28**
11. Use on-screen information to identify types and locations of bugs	**13 - 28**
12. Convert values between binary and decimal, explaining how sensor readings are stored in digital memory	
13. Apply standard searching (linear) and sorting techniques to process and organise datasets of sensor or spreadsheet data	**27**
14. Explain how information is sent over networks using packets and network topology.	**29, 30**

Goal 9

MODERN MARKETING AND BUSINESS

Lesson 40	
Goal	**Goal 9 – Modern marketing and community agribusiness**
Learning objective	We are learning to create interactive data slides using embedded data charts and dynamic visual callouts to communicate with a buyer
UN Sustainable Development Goal	 **SDG 12: Responsible Consumption and Production (Target 12.3: Reduce food losses along production and supply chains)** Provide clear data about yield, harvest windows and storage to buyers to minimise spoilage
Success Criteria	• I can select and use a consistent design layout for my slides • I can present data clearly using a column chart • I can highlight crop quality features using shapes and other highlighting
Introduction	**Time: 15 minutes** 1. Ask the students to imagine that they have a product and a buyer is visiting from a different city or another country. What would be important in securing a deal when we were presenting out product? 2. Show two examples of a slideshow presentation: one which is text-heavy and one which uses grids and charts. Emphasise that the features in the second help to speed up buyer decision making.
Main lesson	**Time: 35 minutes** **Step 1: Set up of slides and grid alignment (10 mins)** • Model opening the presentation software. Navigate to View → Slide master.

	• Model setting up a 3-column layout grid. • Show the children how to decide and choose their brand colours, which should mirror the produce they are selling (i.e. greens, yellows) **Step 2: Visualising data (15 mins)** 1. Model inserting a chart (Insert → Chart → Column) 2. Ask students to insert some sample agricultural data. 3. For columns, input 'Month' 4. For rows, input 'Projected Yield (kg)' vs. 'Available Wholesale Stock (kg)' 5. Format the chart by removing any clutter, i.e. gridline. Contract the colours and bold important data. **Step 2: Image Creation (10 mins)** 1. Insert pictures of the crop being sold and crop them into custom geometric shapes (Crop to Shape → Circle/Rectangle) 2. Add visual callouts (text badges) over images, i.e.: *"100% hand picked and organic certified." "Grown in the foothills of the Himalaya Mountains"*
Plenary	**Time: 10 minutes** 1. Ask students to peer-review each other's slides. 2. Ask the students to look at their partners screen for 5 seconds and then turn their back and recall as much information as possible.

Lesson 41	
Goal	**Goal 9 – Modern marketing and business**
Learning objective	We are learning to build an interactive, non-linear presentation that uses action buttons and internal hyperlinks, embedded audio/video triggers and subtle slide transitions
UN Sustainable Development Goal	 **SDG 8: Decent Work and Economic Growth (Target 8.2: Achieve higher levels of economic productivity through diversification, technological upgrading and innovation)**
Success Criteria	• I can create a dashboard menu which links to other slides • I can Insert a "**Return to Menu**" button on each slide • I can embed audio or video content
Introduction	• Ask the students what the difference between sequential and non-sequential slides are. • What are the advantages and disadvantages of a sequential slideshow presentation? • Emphasise that sometimes a buyer might want to know about something on a later slide, which can make non-sequential presentations useful.
Main lesson	**Time: 35 minutes** **Step 1: Creating an interactive slide menu (15 minutes)** • Show students how to add a second slide titled **"Interactive menu"** • Model adding three buttons: **1) Crop** Specifications, **2) Wholesale pricing, 3) Delivery**

	Step 2: Adding hyperlinks to slides (10 minutes) • Model highlighting the buttons adding to slide 2, **right clicking**, pressing '**link'** then selecting '**slide in this presentation'** • Link '**Wholesaling'** directly to **slide 4.** • On **slide 4,** add a '**home'** button. Link this to **slide 2.** • Test this works in **presenter mode.** **Step 3: Adding media and animation triggers (15 minutes)** • Show students how to **insert** a 15 second video recording showing soil preparation or crop sorting (Pixabay.com can be used to source videos) • Set media options by selecting **'play to click'** on the video • Add smooth/morphing transitions between slides so that movement feels like a mobile application, rather than a presentation
Plenary	**Time: 10 minutes** • Ask the children to keep working on their presentations to make them as professional as possible before their presentations in next week's lesson. • Show students sources of information that they could use to add to their presentations.

Lesson 42	
Goal	**Goal 9 – Modern marketing and business**
Learning objective	We are learning to use **Presenter View** to deliver an effective and engaging presentation
UN Sustainable Development Goal	 **SDG 2: Zero Hunger (Target 2.4: Sustainable Food Production)** Teaching farmers to advocate for sustainable farming techniques
Success Criteria	• I can use **Presenter View** instead of reading form the slides themselves • Combne vocal projection with eye contact to deliver an engaging presentation • Answer questions effectively and convincingly based on information in the presentation
Introduction	• Show the students a video example of an effective presentation. • Ask the students to make notes of some of the thigs that the presenter foes to be effective and engaging (maintains eye contact, does not read from slides) • Emphasise that success in selling to a potential buyer often depends on the effectiveness of a presentation
Main lesson	**Time: 35 minutes** **Step 1: Writing notes for a presentation (10 minutes)** • Ask the students to open the presentations they built in lessons 1 and 2 of this unit of work. • Model accessing the **Presenter Notes** section and writing short notes to prompt information in the presentation.

	Step 2: Setting up Presenter View (10 minutes) • Guide children through the process of connecting a **secondary screen** to their device, or simulating dual screens (**Alt+F5** in PowerPoint, or selecting **'Presenter** View' in Google Slides). • Ask students to practise using their notes to present their slides, including using the timer to help with time spent on each slide. • Encourage students to focus on eye contact and recalling key information – this is why we practise! **Step 3: Pitching to classmates (15 minutes)** • Students work in pairs to pitch their presentation to as many classmates as possible. • Encourage the listening students to interrupt with questions at one point in the presentation. This could take the form of a negotiation. • Provide students with a scoring rubric to score their classmates' presentation on. Categories can include **Content**, **Engagement**, **Delivery**, **Response** (out of 3)
Plenary	**Time: 10 minutes** • Ask the children for constructive feedback on their classmates' presentations, as well as their own presentation. • **Did their presentations improve each time because of practice?** • Emphasise that practice is the key to a successful presentation.

Objectives taught in this unit (highlighted)

Advanced level (age 13 – 16) – teaching objectives	
ICT Skills	**Lesson(s):**
1. Type accurately, at a speed of 30wpm, using both hands correctly	**3**
2. Master advanced file systems/: identify structural extensions (.py, .csv, .pdf, .zip) and manage local desktop vs. cloud directory.	**1**
3. Establish physical hardware connections using both Bluetooth and USB, connect a Microbit to another Microbit via radio	**28**
4. Create multi-page documents including custom page margins, table of contents, text alignment adjustments, page breaks.	**2**
5. Design agricultural and business spreadsheets that use comparative operators and conditional formulas, e.g. SUM, AVERAGE, COUNTIF.	**37, 38**
6. Deliver a high-impact presentation designed for a target audience using design tools and following conventions.	**9, 10**
7. Use principles of graphics design to create 2D images, i.e. alignment, contrast.	**40, 41, 42**
8. Use sophisticated, interactive online tools to research and critically evaluate information using this to further learning and investigations and create original content	**4, 5, 6, 7, 8, 11, 12, 39**
Digital literacy, AI literacy and E-safety	**Lesson(s):**
1. Explain specific legal frameworks governing technology use. Contrast open-source content with copyright infringement.	**1, 31**
2. Describe and utilise strategies to identify and counter the effects of screen-addition.	**32**
3. Analyse the effects of digital harassment, describing protocols to mitigate and report cyberbullying.	**32**
4. Cross reference primary sources to spot digital manipulation, deepfakes and automated disinformation campaigns.	**33**
5. Evaluate the local and global economic impacts of automation and artificial intelligence on rural labour markets and agricultural supply chains.	**35**
6. Outline the structural mechanics of machine learning, explaining concepts behind algorithms (i.e. decision trees and data pattern recognition).	**34, 35**
7. Critique the risks if modern generative models, covering training bias, data leaks, security risks and cognitive offloading.	**36**
8. Navigate the psychological hazards of anthropomorphised AI, explaining the risk of users forming inappropriate attachments to digital conversational agents.	**36**

Programming and computational thinking	Lesson(s):
1. Open an IDE (i.e. https://python.microbit.org/v/3/) and initialise a new script by importing the core hardware library (from microbit import *)	**13**
2. Use inputs (i.e. button_a.is_pressed()) and display output dynamically using the LED matrix (display.show()) and text scrolling	**24, 25, 26**
3. Use the built in functions in Python, including print(), input() and time.sleep()	**13, 14**
4. Read, process and convert analogue environmental inputs from alligator clip probes and onboard sensors (light, temperature, accelerometer)	**25, 26**
5. Use conditional structures (if, elif, else) and relational operators (<, =, >) to trigger outputs	**15, 16**
6. Use 'or', 'and', 'not' logic in conditional code and construct truth tables yo map hardware decision making	**16**
7. Program text-based looping structures (while True: and for I in range()), use timing controls (sleep()) and break commands to manage iteration	**17, 18**
7. Notice examples of variables in text-based code. Define and call variables within a program	**14, 15**
8. Construct and manipulate lists to store sequential datasets (e.g. calculating daily soil temperature)	**22, 23**
9. Use functions to organise code logically, defining and using functions that accept parameters, execute instructions and return processed data values	**19, 20, 21**
10. Use the radio module to transmit and receive data packets between multiple Microbits	**28**
11. Use on-screen information to identify types and locations of bugs	**13 - 28**
12. Convert values between binary and decimal, explaining how sensor readings are stored in digital memory	
13. Apply standard searching (linear) and sorting techniques to process and organise datasets of sensor or spreadsheet data	**27**
14. Explain how information is sent over networks using packets and network topology.	**29, 30**

This scheme of work references third-party interactive educational software, applications, platforms and media resources:

Productivity, Graphics design and desktop software

- **LibreOffice** (The Document Foundation) – Free and open-source office suite used under the Mozilla Public Licence v2.0 (MPLv2).
- **Microsoft Word, PowerPoint, Excel** (Microsoft Corporation) – Referenced for word processing, presentation creation and spreadsheet creation instruction

Coding, AI and Physical Computing

- **Code.org (AI for Oceans)** – Used under Code.org's free educational terms
- **Microsoft MakeCode** (Microsoft Corporation) – Used under Microsoft's free educational terms
  Microsoft makecode screenshots used with permission from Microsoft Corporation. Interface images © Microsoft. Microsoft and makecode are trademarks of the Microsoft group of companies.
- **Micro:bit Python Editor** (Microbit Educational Foundation) – Used for educational coding under the Micro:bit Educational Foundation terms
- BBC Microbit Python Editor screenshots used courtesy if the Micro:bit Educarional Foundation (microbit.org). Micropython is licences under the MIT licence. 'Python' is a registered trademark of the Python Software foundation.
- **Mu** (Code with Mu) – Free, open-source Python IDE used under the GNU General Public Licence v3 (GPLv3).
- **Scratch** (MIT Media lab) – Used under MIT's free educational terms and Creative Commons Attribution-ShareAlike 2.0 licence

Scratch is a project of the Scratch Foundation. It is available for free at http://scratch.mit.edu. Interface screenshots used under Creative Commons Atribution-ShareAlike terms.

Interactive Geography, Learning platforms, Research and Practise tools

- **E-Paath and E-Pustakalaya** (OLE Nepal) – Licenced under CC BY-NC-SA 3.0
- **Google Earth** (Google LLC) – mapping and spatial exploration tool used under Google Earth's standard permissions for educational reference
- **LearnEnglish Kids** (British Council) – Used for educational classroom reference
- **Typing.com** (Teaching.com) – used for educational touch-typing instruction and practice

Information and Media Resources

- **Wikipedia** (Wikimedia Foundation) – Content referenced under the Creative Commons Attribution ShareAlike 4.0 (CC BY-SA 4.0) Licence
- **Pixabay** – Stock images and visual assets used under the Pixabay Content Licence (free for educational and commercial use)
- **United Nations Sustainable Development Goals** – (United Nations) - https://www.un.org/sustainabledevelopment - Used according to the guidelines on the United Nations website, for non-commercial, educational purposes. The content of this publication has not been approved by the United Nations and does not reflect the views of the United Nations or its officials or Member States.
- **YouTube** (Google LLC) – Educational videos embedded or linked under YouTube's standard terms of service