

<b>Subject</b>	Computing
<b>Term</b>	Cycle 1
<b>Duration (approx.)</b>	6+ lessons
<b>Module</b>	To be completed

## **Building Retention: What prior learning must be built upon/revisited and how will it be assessed?**

Pupils will have varying experiences of programming and of computer games. The purpose of this unit is to introduce some key programming concepts and link to an understanding of how games work. The purpose of an algorithm can be linked to everyday activities.

## **Skills and concepts to be developed and assessed (linking to identified AOs)**

- Planning a computer game.
- Creating algorithm (step by step instructions) for how the code in the game will work.
- Kodu game programming skills, for example:
  - *Creating worlds*
  - *Adding behaviours (sequencing)*
  - *Using clones and creatables*
  - *Using pages or IF tests (selection)*
  - *Using scoring, timers, health, power-ups*
- Fixing code bugs to make game work.

## **Factual knowledge to be taught and assessed (including subject specific vocabulary).**

- What algorithms are and why they are important in computer programs.
- What *sequencing* means in computer programming.
- What *selection* means in computer programming.

## **Formative Assessment/key piece of work prior to end of unit:**

- Creation of game plan and pseudo-code to explain how code will work.

## **Summative Assessment**

- The final plan.
- Finished Kodu game.

## **Spelling-Punctuation-Grammar How will you promote high standards within this module?**

Encourage use of spell checker in software as well as proof-reading. Encourage this as part of the process of bug fixing.

## **Link forward: where next for the learning?**

Programming terminology will be revisited in the Small Basic and Microbit units; links will be made between the same terms but their use to create a different type of product.

<b>Subject</b>	Computing
<b>Term</b>	Cycle 2
<b>Duration (approx.)</b>	6+ lessons
<b>Module</b>	Kodu Game Programming

## **Building Retention: What prior learning must be built upon/revisited and how will it be assessed?**

Pupils will have varying experiences of programming and of computer games. The purpose of this unit is to introduce some key programming concepts and link to an understanding of how games work. The purpose of an algorithm can be linked to everyday activities.

## **Skills and concepts to be developed and assessed (linking to identified AOs)**

- Planning a computer game.
- Creating algorithm (step by step instructions) for how the code in the game will work.
- Kodu game programming skills, for example:
  - *Creating worlds*
  - *Adding behaviours (sequencing)*
  - *Using clones and creatables*
  - *Using pages or IF tests (selection)*
  - *Using scoring, timers, health, power-ups*
- Fixing code bugs to make game work.

## **Factual knowledge to be taught and assessed (including subject specific vocabulary).**

- What algorithms are and why they are important in computer programs.
- What *sequencing* means in computer programming.
- What *selection* means in computer programming.

## **Formative Assessment/key piece of work prior to end of unit:**

Creation of game plan and pseudo-code to explain how code will work.

## **Summative Assessment.**

- The final plan.
- Finished Kodu game.

## **Spelling-Punctuation-Grammar How will you promote high standards within this module?**

Encourage use of spell checker in software as well as proof-reading. Encourage this as part of the process of bug fixing.

## **Link forward: where next for the learning?**

Programming terminology will be revisited in the Scratch unit; links will be made between the same terms but their use in a different type of coding environment.

<b>Subject</b>	Computing
<b>Term</b>	Cycle 2
<b>Duration (approx.)</b>	6 lessons
<b>Module</b>	Computing Basics

**Building Retention: What prior learning must be built upon/revisited and how will it be assessed?**

Pupils will all have experiences of using computers, but will have a varying understanding of the terminology of the equipment they use and its purpose. This unit will assume little prior knowledge.

**Skills and concepts to be developed and assessed (linking to identified AOs)**

- Converting binary to decimal and decimal to binary.
- Using software to present research.

**Spelling-Punctuation-Grammar  
How will you promote high standards within this module?**

Encourage use of spell checker software as well as proof reading. Use guides and displays to help with spelling of key terminology.

**Factual knowledge to be taught and assessed (including subject specific vocabulary).**

Technical knowledge of computing architecture to be covered:

- Input and output devices
- Main parts of a computer system
- Understanding binary code
- Connecting to the internet
- What networks are and how they work
- Possible health and safety impacts of using computers.

**Link forward: where next for the learning?**

Future links can be made back to the essentials of how a computer operates and binary code in all future units through relating all technology processes to inputs, processes and output.

**Formative Assessment/key piece of work prior to end of unit:**

Binary and ASCII task marking.

**Summative Assessment.**

- Completed research tasks.
- Final Assessment quiz to check understanding of the key theory learnt.

<b>Subject</b>	Computing
<b>Term</b>	Cycle 3
<b>Duration (approx.)</b>	6 lessons
<b>Module</b>	To be completed

**Building Retention: What prior learning must be built upon/revisited and how will it be assessed?**

It is unlikely pupils will have used spreadsheets before. Links can be made, however, with the first two units (Kodu Game programming and Small Basic) and the importance of testing formulae.

**Skills and concepts to be developed and assessed (linking to identified AOs)**

- How to enter data into a spreadsheet.
- How to use formatting tools in a spreadsheet to make it easier to read.
- How to use formulae and functions to carry out calculations.
- How to test formulae are working accurately.
- How to use data in a spreadsheet to create charts.

**Factual knowledge to be taught and assessed (including subject specific vocabulary).**

- What spreadsheets are used for and why they are useful.
- What spreadsheet modelling is and how this can be used to solve problems.

**Formative Assessment/key piece of work prior to end of unit:**

Development of spreadsheet to contain data and appropriate formatting.

**Summative Assessment.**

- The final spreadsheet.
- Final modelling task to check pupils' understanding of how to use spreadsheets to answer questions.

**Spelling-Punctuation-Grammar  
How will you promote high standards within this module?**

Encourage use of spell checker in software as well as proof-reading. Encourage this as part of the process of testing the accuracy of the spreadsheet.

**Link forward: where next for the learning?**

Links can be made to the final unit in the importance of breaking instructions down into small tasks and testing thoroughly.

<b>Subject</b>	Computing
<b>Term</b>	Cycle 3
<b>Duration (approx.)</b>	6+ lessons
<b>Module</b>	To be completed

## **Building Retention: What prior learning must be built upon/revisited and how will it be assessed?**

All pupils have completed Unit 7.1 Kodu Game Programming and Unit 7.2 Small Basic. Links can be made to these units to explain the code created and the algorithms needed to break a coding problem down into steps.

## **Skills and concepts to be developed and assessed (linking to identified AOs)**

- How to use Microsoft block programming to create code for the Microbit.
- How to write and test code in the editor.
- How to compile, download and run code on the Microbit.
- How to use a range of programming techniques.
- Using algorithms to represent code as 'pseudocode'.

## **Factual knowledge to be taught and assessed (including subject specific vocabulary).**

Understanding key programming terminology—sequence, selection, iteration, variables.

## **Formative Assessment/key piece of work prior to end of unit:**

Code created using lesson tutorials.

## **Summative Assessment.**

- Development of algorithm to give step by step instructions for a piece of code.
- Final range of code created using development of skills.

## **Spelling-Punctuation-Grammar How will you promote high standards within this module?**

Encourage use of spell checker in software for written work. Introduce testing code (including spelling errors) as an important part of the programming process.

## **Link forward: where next for the learning?**

Further coding units in Years 8 and 9 can be linked back to the experiences of using the Microbit. When pupils advance their writing of text based code in Year 8, they will see similar patterns to what they have created using the block editor.