



Declarative and Procedural Knowledge

Year 5

Contents

Introduction	3
Introduction to Purple Mash	4
Quizzing.....	5
Databases.....	6
Game Creator	7
Spreadsheets.....	8
Coding.....	9
Coding Crash Course.....	11
Word Processing	15
Concept Maps.....	17
Coding: External Devices	18
micro:bits.....	19

Introduction

The Declarative and Procedural Knowledge documents are designed to support teachers in understanding the intended learning outcomes of each unit. They outline the specific knowledge and skills that children should acquire and demonstrate by the end of their learning.

- Declarative Knowledge sets out what children will **know**. This includes facts, concepts, definitions, and key ideas that form the foundation of the unit.
- Procedural Knowledge sets out what children will **be able to do**. This focuses on the skills and processes children should develop and apply when using technology.

These documents are used to:

- Provide teachers with a clear overview of learning expectations for each unit.
- Ensure consistency of teaching and progression of knowledge and skills across year groups.
- Support planning, teaching, and assessment by highlighting the essential outcomes to focus on.
- Reinforce the balance between understanding (knowing) and application (doing) in computing.

This document aims to help teachers see the bigger picture of what children will learn, how they will apply it, and how these elements connect across the computing curriculum.

Introduction to Purple Mash

National Curriculum Links	Dominant objectives for this unit: Select, use and combine a variety of software (including internet services) on a range of digital devices to design and create a range of programs, systems and content that accomplish given goals, including collecting, analysing, evaluating and presenting data and information.
---	---

Declarative - By the end of the unit, the children will know that:	Procedural – By the end of the unit, the children will know how to:
It is important to log in to a site, the importance of keeping passwords safe and the need to log out at the end of a session.	- Access Purple Mash from home and school. - Log out of Purple Mash. - Give reasons why it is important to keep a password safe and not share it with other people.
An avatar is a virtual representation of a person suitable for use online.	Make and edit their own avatar.
The 2Do system is used to set work for children within Purple Mash.	- Open 2Dos. - Save 2Dos. - Hand in 2Dos and communicate with their teacher via the 2Do.
Online sites have a main page called the homepage.	Access the Purple Mash homepage when on the site.
Online sites often use an alert system to communicate with the user.	Access alerts within Purple Mash.
To move to a different activity in Purple Mash, you must close the current activity.	Close activities in Purple Mash.
Many online sites, including Purple Mash, have an area for an individual's work that is accessible only to the individual (and in Purple Mash to their teacher as well).	- Access their work area. - Save work in their work area. - Locate and open work they have done previously in their work folder.
To access Purple Mash programs, you use the Tools area.	Open a specified tool.
To access activities related to a specific topic, you can use the Topics area.	Find activities on a specified topic.
You can access non-visible parts of a screen using scrolling.	Scroll up and down and from side to side where applicable.
Purple Mash includes collaborative tools.	Recommend a tool to use for collaborative group or class work.

Need more support? Contact us:

Call: 020 8203 1781 | Email: support@2simple.com | Visit: help.2simple.com

Powered by  2simple

Quizzing

National Curriculum Links	Dominant objectives for this unit: Select, use and combine a variety of software (including internet services) on a range of digital devices to design and create a range of programs, systems and content that accomplish given goals, including collecting, analysing, evaluating and presenting data and information.
----------------------------------	---

Declarative - By the end of the unit, the children will know that:	Procedural – By the end of the unit, the children will know how to:
There are different types of quiz, suitable for a variety of different purposes.	Select the best type of quiz to make for any given task.
Within 2Quiz, there are different types of quiz questions, suitable for different purposes.	Select the most suitable type of quiz question based on a chosen topic and the question being posed.
- The success of a quiz can be determined by considering how difficult it is, whether it peaks interests and the capability of the audience in relation to the subject matter. - It is also important to consider the enjoyability, visual variety and feedback to the end user when creating a quiz.	Design successful quizzes that meet a specific brief by making use of the full functionality within 2Quiz.
Additional features and enhancements, including title screens and content screens can maximise the success of a quiz.	Explore the additional features, selecting which to use and how to make best use of these.
Settings, including quiz style, time limits, feedback settings and feedback sounds change the look and feel of a quiz.	Explore the settings, selecting which to use to create the best outcome for a quiz.

Databases

National Curriculum Links	Dominant objectives for this unit: • Select, use and combine a variety of software (including internet services) on a range of digital devices to design and create a range of programs, systems and content that accomplish given goals, including collecting, analysing, evaluating and presenting data and information.
---	--

Declarative - By the end of the unit, the children will know that:	Procedural – By the end of the unit, the children will know how to:
• A database contains a set of data that can be searched and sorted to retrieve information.	• Use the functionality of the 2Investigate database tool to work with data.
• A table-based database contains records and fields.	• Identify, create and edit records and fields of a database.
• Database tools support interpreting data using functionality including sorting, filtering, grouping and searching.	• Sort, group and arrange information in a database. • Search for information in a database. • Answer questions involving the interrogation of a database.
• Collaborative databases allow multiple people to work on one set of data at the same time.	• Add records to a collaborative database. Interrogate the data of a collaborative database.
• A well-designed database has built-in validation to ensure correct data formatting.	• Design a database with validation included.
• Queries can be built using the database tools to answer specific questions about the data.	• Use the query builder to specify conditions for the data returned. • Use conditional operators correctly when building a database query.
• Some table-based databases have multiple tables of related data. These can be linked on common fields to build complex queries across multiple tables.	• Build complex queries across multiple tables to answer questions about the data.
• Databases support the production of automated reports using fields within the data.	• Produce automated reports using fields within the data.

Game Creator

National Curriculum Links	Dominant objectives for this unit: • Select, use and combine a variety of software (including internet services) on a range of digital devices to design and create a range of programs, systems and content that accomplish given goals, including collecting, analysing, evaluating and presenting data and information.
---	--

Declarative - By the end of the unit, the children will know that:	Procedural – By the end of the unit, the children will know how to:
• There are criteria which determine the playability of a video game.	• Evaluate the playability of a video game.
• 3D games can be made using the 2DIY 3D tool.	• Make a 3D game using the key functions of the 2DIY 3D game creator tool.
• To maximise playability, it is important to plan and design a game before making it.	• Use a design document to plan the theme and aim, characters and game world, game features, timing, character movement and interaction for a 3D game.
• The visual properties of a game should fit with the theme and add a feeling of immersion to the experience.	• Choose and design appropriate textures for the game world walls, floors, lighting, hazards and scenery.
• The audio properties of a game should fit with the theme and add a feeling of immersion to the experience.	• Choose or compose appropriate sound effects and music for the game.
• The design of baddie sprites and collectable quest items is a key aspect of game creation.	• Design the collectable quest items and add movement, sound effects and actions. • Consider where to place the collectable quest items so it is possible to finish the game. • Place sprites in the game in such a way as to provide challenge but not make it impossible to play. • Select appropriate penalties for encountering an enemy.
• It is important to give the player good instructions to enable them to enjoy playing the game.	• Create useful instruction screens for players.
• Evaluation is important so a game can be improved.	• Evaluate games made by their peers using given criteria. • Read evaluations of their game from others. • Make appropriate improvements to their game.

Spreadsheets

National Curriculum Links	Dominant objectives for this unit: Select, use and combine a variety of software (including internet services) on a range of digital devices to design and create a range of programs, systems and content that accomplish given goals, including collecting, analysing, evaluating and presenting data and information.
----------------------------------	---

Declarative - By the end of the unit, the children will know that:	Procedural – By the end of the unit, the children will know how to:
A spreadsheet is a digital tool for storing, organising and analysing data.	Enter data accurately into cells, using rows, columns and headings appropriately.
Cells, rows, columns and cell references provide a structure for organising information.	Select, navigate and reference cells to build formulas and structure spreadsheets clearly.
Formulas allow spreadsheets to perform calculations automatically, including addition, subtraction, multiplication, division.	Create and edit formulas (e.g. =A2*B2, =A3+B3) and apply them across cells to update calculations efficiently.
Conversions rely on known mathematical relationships (e.g. cm–m, mm–cm, g–kg).	Apply conversion formulas within spreadsheets to automate unit changes.
Line graphs are used to represent continuous data and show trends over time.	Create line graphs in 2Calculate by selecting relevant data, choosing a suitable scale, and labelling axes clearly.
Continuous data changes smoothly over time, while discrete data represents separate, countable values.	Sort data into continuous or discrete categories and choose the correct type of graph or representation for each.
Weather data contains measurable patterns that can be compared and interpreted.	Input weather datasets and analyse them to identify trends, make predictions or determine risks.
Budgets are used to plan money, compare costs and make financial decisions.	Use totals and calculations to compare event costs and stay within a set budget.
Spreadsheets help solve real-world problems by presenting information clearly and accurately.	Use spreadsheets to plan an event, analyse weather, or cost a holiday, making justified decisions based on evidence.
A good spreadsheet is clear, well-labelled, accurate and structured logically.	Format spreadsheets with headings, consistent layout, neat tables and correctly labelled graphs to improve readability.

Coding

National Curriculum Links	Dominant objectives for this unit: • Design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts. • Use sequence, selection, and repetition in programs; work with variables and various forms of input and output. • Use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs.
----------------------------------	---

Declarative - By the end of the unit, the children will know that:	Procedural – By the end of the unit, the children will know how to:
• The way that code is written and structured can improve the ability to debug it, simplify the code and run it more efficiently.	• Use tags to write code for several objects in one block of commands facilitated by the computer-generated variables for tagged objects. • Use tabs to organise code. • Use functions to reduce the need to repeat blocks of code and make the program easier to read.
• A simulation is a model that represents a real or imaginary situation.	• Explain what a simulation is and provide examples of physical systems that could be made into a simulation. • Plan an algorithm for a physical system such as traffic light sequences using 2Chart. • Use the algorithm to write the code. • Test and evaluate the program. • Use the program to model changes to the physical system for instance improvements if there are traffic jams at the lights.
• Abstraction is a way to think about a situation without unnecessary details, enabling a focus on the function of a program.	• Recognise what abstraction is and why it is important. • When planning a program, use abstraction to focus thinking on the task itself.
• Decomposition is a method of breaking down a task into manageable components. This makes coding easier as the components can then be coded separately and then brought back together in the program. Each part is known as a sub-routine.	• During planning, use decomposition to break down the plan into the key parts that are required to get the program functioning. • Use decomposition as a way to organise the code. For example, using tabs in 2Code. • Use 2Chart to create a flowchart that contains sub-routines.

Need more support? Contact us:

Call: 020 8203 1781 | Email: support@2simple.com | Visit: help.2simple.com

Powered by  2simple

• A function is a named group of commands that a program can run when called by name; reducing the need to rewrite code repeatedly.	• Create functions in 2Code and call these functions appropriately instead of repeating code. • Refine existing code to include functions by recognising where code is repeated within the program.
• A datatype defines the type of data, for example, a number or text, and what operations can be performed with it.	• Make sensible choices of datatype when creating variables. • Anticipate the effect of operators on the variable values given their datatype. • Use the Variable Watch window to monitor the changing value of the variables in a program.
• It is important to initialize variable values to avoid unanticipated bugs in the program.	• Initialize all variables appropriately. • When debugging, look at the variable data type and initialization as possible sources of bugs.
• Concatenation is the name given to the action of linking things together in a series. For example, in programs we might want to link words together to form random phrases that are seen by a user of a program.	• Recognise where concatenation can be useful in programs. • Use the print to screen command in combination with random words that are joined together to demonstrate concatenation. • Demonstrate concatenation in other programs created such as linking a variable (score) with text (well done) for a game. • Debug programs that involve concatenation errors in strings.

Coding Crash Course

National Curriculum Links	Dominant objectives for this unit: • Design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts. • Use sequence, selection, and repetition in programs; work with variables and various forms of input and output. • Use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs.
---	--

Declarative - By the end of the unit, the children will know that:	Procedural – By the end of the unit, the children will know how to:
• In computing, a set of instructions is known as an algorithm. Steps in an algorithm must be followed to achieve the intended outcome.	• Interpret what a piece of code means. • Follow a written algorithm. • Identify required events, objects and actions from a program algorithm. • Create instructions in the form of simple algorithms with attention to the order and the level of detail.
• Flowcharts are a type of diagram that use specifically shaped labelled boxes and arrows to represent an algorithm as a diagram.	• Identify features of a flowchart including the starts, inputs, outputs, delays such as a timer, actions and events. • Follow the flow of the chart and interpret what it is representing. • Use a flowchart as part of a planning process for designing a program.
• Programs follow a sequence of instructions (commands) in order.	• Run code and see the sequential effect that occurs.
• In 2Code, code is created using coloured code blocks that can connect to form commands.	• Make commands using the code blocks. • Run these commands.
• Commands consist of a combination of different block types; these are actions, events, objects and outputs	• Decide which type of block is needed and which combinations of block types will be needed to complete a command.
• An object is an item in a program that can be given instructions to change in some way by adding an action block.	• Use objects within commands when coding.
• An action is the way that an object changes when programmed to do so. For example, move.	• Connect object and action blocks to form commands.

Need more support? Contact us:

Call: 020 8203 1781 | Email: support@2simple.com | Visit: help.2simple.com

Powered by  2simple

An event is something that makes a block of code run in response to an action such as a user pressing a key or clicking a screen.	- Use the 'When Clicked' and 'Collision Detection' event code blocks. - Position a command inside an event.
The way that code blocks are positioned within each other determines the order in which the code runs.	Create code that is nested for example putting an object and action command within an event command.
Selection in a program means that it has a decision command that will run different procedures dependent on whether a condition is met or not.	Interpret flowcharts depicting selection and explain what happens if a condition is or isn't met within it.
'If' statements and 'if/else' statements are used to create selection in 2Code.	- Identify 'if' and 'if/else' control blocks. - Read code that includes selection to determine what the program will do. - Create selection within 2Code using 'if' and 'if/else' statement blocks. - Debug programs that use selection. - Nest an if/else block inside an if/else block to give a user different validation options.
Co-ordinates are used in computer programming to determine the position of a point, shape or object and that these change according to where they are positioned on the screen.	- Identify co-ordinates in Design view. - Change the co-ordinates of objects within the code.
Repeat is a control block allowing blocks of commands to repeat a specified number of times.	- Identify situations when a repeat command can make the code more efficient. - Use a repeat command to make other commands repeat.
- Timers can control when a block of commands run. - Timer commands can be run after a timed delay or at regular intervals. - In 2Code there are two timer options, timer every or timer after. These can be altered by changing the number of seconds/quarter seconds.	- Code the features of timer after command including the interval, action(s), correct nesting of commands. - Distinguish between the use of the timer every and timer after commands. - Use a timer every command to make an event happen repeatedly.
- Variables are a place in computer memory that remains for the length of the program. - Variables have a name and a value. - The values can be changed in the code. - Variables are used in programming to keep track of things.	- Create number variables and initialise their values. - Change the value of variables within the program by using the 'Change Variable' command. - Use the value of variables within programs.

	Use the variable watch to monitor how a variable changes as the program executes code.
A datatype defines the type of data, for example, a number or text, and what operations can be performed with it.	- Make sensible choices of datatype when creating variables. - Anticipate the effect of operators on the variable values given their datatype. - Use the Variable Watch window to monitor the changing value of the variables in a program.
Concatenation is the name given to the action of linking things together in a series. For example, in programs we might want to link words together to form random phrases that are seen by a user of a program.	- Recognise where concatenation can be useful in programs. - Use the print to screen command in combination with random words that are joined together to demonstrate concatenation. - Demonstrate concatenation in other programs created such as linking a variable (score) with text (well done) for a game. - Debug programs that involve concatenation errors in strings.
- Testing and debugging are an important part of the process of making computer programs. - Understanding what nesting is and the effect it has on a program can help when trying to debug a program.	- Read the code of example programs with nesting to determine the outcome of running the code. - Test what happens when changing how a program is nested. Examples could include moving a timer so that it is nested within a timer that is currently nested within a when clicked command and comparing how it executes. - Use the knowledge of nesting to help debug a program that isn't working as intended.
Decomposition is a method of breaking down a task into manageable components. This makes coding easier as the components can then be coded separately and then brought back together in the program. Each part is known as a sub-routine.	- During planning, use decomposition to break down the plan into the key parts that are required to get the program functioning. - Use decomposition as a way to organise the code. For example, using tabs in 2Code. - Use 2Chart to create a flowchart that contains sub-routines.
A function is a named group of commands that a program can run when called by name; reducing the need to rewrite code repeatedly.	Create functions in 2Code and call these functions appropriately instead of repeating code.

	• Refine existing code to include functions by recognising where code is repeated within the program.
--	---

Word Processing

National Curriculum Links	Dominant objectives for this unit: Select, use and combine a variety of software (including internet services) on a range of digital devices to design and create a range of programs, systems and content that accomplish given goals, including collecting, analysing, evaluating and presenting data and information.
---	--

Declarative - By the end of the unit the children will know that:	Procedural – By the end of the unit the children will know how to:
A word processing tool can be used to create a range of documents.	- Recognise different documents that can be produced using a word processor. - Open new blank documents and existing documents. - Identify a range of templates that can be used to create documents. - Navigate around the main toolbar including sub-toolbars for individual objects. - Expand toolbar categories to show additional options. - Save a document and rename it. - Select specific words to format. - Type sentences and then format them accordingly.
The layout of a document can be altered.	- Switch between portrait and landscape mode. - Change the margins.
Images can be inserted and edited in a word processor.	- Search for images online or using stock images. - Download images to a document. - Edit an image using the image handles. - Crop and resize an image. - Wrap text around an image. - Use the various options to change the appearance of an image.
The look and placement of text within a document can be modified.	- Use appropriate keyboard or touchscreen shortcuts for copy, paste and cut. - Use the text formatting toolbar to change the text within the document. - Add headings and subheadings to a document. - Use a range of bullet points including numbered lists.

	• Use bold, underline, italic formatting options purposefully. • Use a range of font choices to suit a document's purpose.
• Tables can be used to present information within a document.	• Insert a table. • Add in columns and rows to a table. • Distribute rows and columns in a table. • Choose table borders. • Change the background colour of a cell in a table.

Concept Maps

National Curriculum Links	Dominant objectives for this unit: Select, use and combine a variety of software (including internet services) on a range of digital devices to design and create a range of programs, systems and content that accomplish given goals, including collecting, analysing, evaluating and presenting data and information.
----------------------------------	---

Declarative - By the end of the unit the children will know that:	Procedural – By the end of the unit the children will know how to:
A concept map is used to organise ideas and show how they are linked.	Create a concept map using 2Connect.
Concept maps usually begin with a main idea and branch into categories and sub-ideas.	- Add nodes and create linked ideas. - Structure ideas into categories and sub-categories.
Concept maps can be used for different purposes such as planning, organising and presenting information.	Move, connect and organise nodes logically.
Different layouts and structures help make information clearer for an audience.	Adjust their layout, spacing and formatting to improve the clarity of their concept map.
Concept maps can be adapted depending on the purpose or topic.	Choose an appropriate map structure for a task.
Clear organisation helps others understand information more easily.	Refine and improve a concept map to ensure clarity.
Concept maps can be used to support presentations.	Explain the thoughts behind a concept map to an audience.
Presentations should be structured and clearly organised.	Present ideas clearly.
Feedback can help improve organisation and presentation.	Review and improve their work based on feedback.

Coding: External Devices

National Curriculum Links	Dominant objectives for this unit: • Design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts. • Use sequence, selection, and repetition in programs; work with variables and various forms of input and output. • Use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs.
---	--

Declarative - By the end of the unit, the children will know that:	Procedural – By the end of the unit, the children will know how to:
• External devices can be used to control a coded program.	• Write if something is an output or an input. • Define what an external device is. • Start coding using external device code blocks. • Use a QR code to connect an external device. • Play their simple code.
• An external device can be used as a game controller.	• Talk about the features of a game controller. • Complete a partially made game. • Debug their code if it isn't working.
• Text can be outputted to an external device.	• Find blocks relating to graphics and text. • Choose blocks to output text to the external device. • Write a program to code text to a device.
• An external device can be used to model real life situations.	• Recognise when an external device can use sensors and how these can be used in code. • Find blocks relating to sensors in the external device (namely shake, tilt and sound detected). • Plan and code a program using sensors.
• A program can be written for the external device to meet a specific design brief.	• Plan out a program before they start to code. • Code a program written for a specific purpose. • Debug their code if it isn't working. • Share a program with others. • Evaluate a program and make improvements where needed.

micro:bits

National Curriculum Links	Dominant objectives for this unit: • Design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts. • Use sequence, selection, and repetition in programs; work with variables and various forms of input and output. • Use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs.
----------------------------------	---

Declarative - By the end of the unit, the children will know that:	Procedural – By the end of the unit, the children will know how to:
• Inputs from an accelerometer, button, pin or sound can be used to initiate a program.	• Identify event blocks. • Code using an event block to initiate a program. • Nest code that is to be run inside an event block.
• Inputs from the micro:bit can be used by the program to control the output from the micro:bit. This can simulate control systems.	• Write code that takes actions dependent upon what is being sensed as an input e.g. the temperature. • Identify input code blocks and use them. • Identify output code blocks and use them. • Create a simple piece of code that simulates a control system. • Use 'if' statements to introduce selection. • Combine events, output and variable code blocks to produce code that makes a micro:bit change its output based on environmental changes.
• External equipment, such as crocodile clips and electronic circuitry can be attached to the micro:bit to initiate actions in the program that can produce outputs.	• Connect crocodile clips to the micro:bit pins anticipating what action will initiate input to the micro:bit. • Write code that responds to input on the pins.
• Variables can be used to track changes in a program and help with logic.	• Create a number variable. • Initialise a variable value. • Change a variable value when an event is detected. • Use If statements to check the current value of a variable. • Produce outputs based on the variable value.