**Coding Curriculum and Progression**

**Key Concepts, Skills and Approaches to Programming**

|  |  |  |
| --- | --- | --- |
| **Computational Thinking Skills For Every Lesson**  Each lesson from the Everyone Can Code Teacher Guides has an ‘unplugged’ activity which develops these thinking skills in a real life problem. The second activity applies these thinking skills to coding skills through the use of Scratch Junior and Scratch 4.0. | | |
| **LOGICAL REASONING**  **Predicting and analysing**  If you set up two computers in the same way, give them the same instructions (the [program](http://barefootcas.org.uk/barefoot-primary-computing-resources/concepts/programming/)) and the same [input](http://barefootcas.org.uk/programme-of-study/work-various-forms-input/inputs/), you can pretty much guarantee the same [output](http://barefootcas.org.uk/programme-of-study/work-various-forms-output/outputs/). This means that they are predictable. Because of this we can use logical reasoning to work out why something happens. Part of logical reasoning is the ability to use existing knowledge to make reliable predictions about future behaviour of a system. | **PATTERN SPOTTING**  **Spotting and using similarities**  Patterns are everywhere, for example, we use weather patterns to create weather forecasts.  By identifying patterns we can make predictions, create rules and solve more general problems.  Children need to be able to identify repeating patterns in a list of commands to understand how this could be made more efficient using a repeat loop. | **DECOMPOSITION**  **Breaking down into parts**  The process of breaking down a problem into smaller manageable parts is known as decomposition. Decomposition helps us solve complex problems and manage large projects. |
| **DEBUGGING**  **Finding and fixing errors**  Errors in [algorithms](http://barefootcas.org.uk/sample-resources/algorithms/) and code are called ‘bugs’, and the process of finding and fixing these is called ‘debugging’. Getting pupils to take responsibility for thinking through their algorithms and code, to identify and fix errors is an important part of learning to think and work like a programmer.   1. Predict what should happen. 2. Test -find out -exactly what happens when a program is run 3. Work out where something has gone wrong. 4. Fix it. | **EVALUATING**  **Making judgements**  Evaluation is about making judgements, in an objective and systematic way where possible.  Children need to evaluate the effectiveness of their programs in solving a specific task. Pupils should be encouraged to reflect on the quality of the work that they produce – is it fit for purpose? | **TINKERING**  **Playing with coding skills**  We want to develop in children a willingness to experiment and explore a new app or new software. Children should be encouraged to ‘play’ with a new piece of software, sharing what they discover about it to one another, rather than always coming to the teacher for the answers. Pupils can explore how to use others’ code as a starting point for their own programming projects. Tinkering should help develop independence and perseverance when working with technology. |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Key Stage 1** | | **LKS2** | | **UKS2** | | | | | |
| **Unplugged Plans:**  **Scratch Jr Printed Blocks**  [**https://www.scratchjr.org/pdfs/blocks.pdf**](https://www.scratchjr.org/pdfs/blocks.pdf)  **Patterns Unplugged:** [**https://www.barefootcomputing.org/resources/patterns-unplugged-activity**](https://www.barefootcomputing.org/resources/patterns-unplugged-activity)  **Decomposition Unplugged:**  [**https://www.barefootcomputing.org/resources/decomposition-unplugged-activity-ks1**](https://www.barefootcomputing.org/resources/decomposition-unplugged-activity-ks1)  **Unplugged KS1 Pages 6-12:**  [**http://www.code-it.co.uk/wp-content/uploads/2015/05/StaffordUnpluggedComputing1.pdf**](http://www.code-it.co.uk/wp-content/uploads/2015/05/StaffordUnpluggedComputing1.pdf) | | **Unplugged Resources:**  **Scratch Blocks to Print:**  [**https://www.stem.org.uk/elibrary/resource/35230**](https://www.stem.org.uk/elibrary/resource/35230)  **Patterns Unplugged:**  [**https://www.barefootcomputing.org/resources/patterns-unplugged-activity**](https://www.barefootcomputing.org/resources/patterns-unplugged-activity)  **Decomposition Unplugged:**  [**https://www.barefootcomputing.org/resources/decomposition-unplugged-activity-ks2**](https://www.barefootcomputing.org/resources/decomposition-unplugged-activity-ks2)  **Unplugged LKS2 Pages 13-20**  [**http://www.code-it.co.uk/wp-content/uploads/2015/05/StaffordUnpluggedComputing1.pdf**](http://www.code-it.co.uk/wp-content/uploads/2015/05/StaffordUnpluggedComputing1.pdf)  **CS Unplugged:**  [**https://csunplugged.org/en/topics/**](https://csunplugged.org/en/topics/) | | **Unplugged Resources**  **:**  **Scratch Blocks to Print:**  [**https://www.stem.org.uk/elibrary/resource/35230**](https://www.stem.org.uk/elibrary/resource/35230)  **Patterns Unplugged:**  [**https://www.barefootcomputing.org/resources/patterns-unplugged-reusing-recipes**](https://www.barefootcomputing.org/resources/patterns-unplugged-reusing-recipes)  **Variables unplugged:** [**https://www.barefootcomputing.org/resources/variables-unplugged-activity**](https://www.barefootcomputing.org/resources/variables-unplugged-activity)  **Logical Reasoning unplugged:** [**https://www.barefootcomputing.org/resources/logical-reasoning-unplugged-activity**](https://www.barefootcomputing.org/resources/logical-reasoning-unplugged-activity)  **CS Unplugged:**  [**https://csunplugged.org/en/topics/**](https://csunplugged.org/en/topics/) | | | | | |
| **Lesson Plans:**  **Scratch Jr Playground**  [**https://www.scratchjr.org/teach/curricula/playground**](https://www.scratchjr.org/teach/curricula/playground)  **Scratch Jr Activities**  [**https://www.scratchjr.org/teach/activities**](https://www.scratchjr.org/teach/activities)  [**ilearn2**](https://www.ilearn2.co.uk/teachers-html/) | | **Lesson Plans:**  [**http://scratched.gse.harvard.edu/guide/curriculum.html**](http://scratched.gse.harvard.edu/guide/curriculum.html)  Unit 1 2 3  [**ilearn2**](https://www.ilearn2.co.uk/teachers-html/) | | **Lesson Plans:**  [**http://scratched.gse.harvard.edu/guide/curriculum.html**](http://scratched.gse.harvard.edu/guide/curriculum.html)  **Unit 4 5 6**  [**ilearn2**](https://www.ilearn2.co.uk/teachers-html/) | | | | | |
|  | | | See the source image | | See the source image | |  |  | |
|  |  | |  | |  |  | | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **FS** | | **Year 1** | | **Year 2** | **Year 3** | | **Year 4** | | **Year 5** | | **Year 6** | |
| **SEQUENCING SKILLS** | Sequence forwards and turns e.g. with Beebot  [IMG_2795](https://enablingenvironments.files.wordpress.com/2015/02/img_2795.png)  **Predict** the outcome of a set of instructions and test the results.  Use symbols to represent an instruction e.g. 🡩🡪 for forward and turn.  Know how to clear the code  **Decompose** by breaking the code down into chunks (1 step at a time)  1) (clear)  2)  (clear)  3)  (clear)  4) (clear | | **Commands** make a sprite move.  **Sequence** commands of forwards, back, left, right using arrow blocks.  Understand that a sequence of instructions needs to be clear, precise and unambiguous. | | **Sequence** commands including forwards, back and turns more efficiently using blocks.  Understand that some steps in a sequence can be reordered but still achieve the same outcome (flexible sequence).  Understand that the order in which instructions are given will make a difference to the outcome. | Understand that a sequence of instructions in computing is called an **Algorithm**.  Use **decomposition** to break the sequence in to manageable steps.  Understand how to approach **debugging** a program or **algorithm**. | | Sequence commands in Scratch Code blocks  Use **abstraction** as a way of making it easier to think about problems.  (Explanation of abstraction https://www.theschoolrun.com/what-is-abstraction-in-computing)  Understand how **functions** help us think more efficiently. | | Describe what **commands**, **functions, debugging** and **sequences** are.  (Explanation of functions and procedures https://www.theschoolrun.com/what-are-procedures-and-functions)  To read code in Scratch Code blocks   * Repeat loops * Event handling * Selection   Be able to assess success of given instructions and identify and correct any errors that occur. | | To sequence an algorithm using Scratch code blocks  To read Scratch code using:   * Repeat loops * Functions * Event handling * Selection * Variables   Be able to evaluate the effectiveness of an algorithm written by their peers in class. | |
| **UNPLUGED RESOURCES** |  | | **Stafford Unplugged Computing**  Human Robots | | **Stafford Unplugged Computing**  Human Robots | **Scratched**  Unit 1 - Time to dance  Unit 2 - Performing Scripts | | **Scratched**  Unit 1 - Time to dance  Unit 2 - Performing Scripts  Unit 2 - Animations | | **Scratched**  Unit 4 – Dream Game List | | **Scratched**  Unit 4 – Dream Game List | |
| **PLUGED RESOURCES** |  | | **Scratch Jr Playground**  **Lesson 1: Movement Blocks and the Reset Button**  Lesson 2: Backgrounds and Start on Green Flag Triggering Block  Lesson 3: New Characters and Start on Tap Triggering Block  Lesson 4: Recording Sound, and Using the Wait Block and the Speed Block  **Scratch Jr Activities**  Lesson 1 – Drive a car  Lesson 2 – Run a race  Lesson 3 - Sunset  Lesson 4 – sun and moon  Lesson 5 – Spooky Forest  Lesson 6 – Basket ball  Lesson 7 – Dance Party (adapted to contain a loop) | | **Scratch Jr Playground**  **Lesson 1: Movement Blocks and the Reset Button**  Lesson 2: Backgrounds and Start on Green Flag Triggering Block  Lesson 3: New Characters and Start on Tap Triggering Block  Lesson 4: Recording Sound, and Using the Wait Block and the Speed Block  Lesson 5: Simple Character Interaction using Start on Bump  Lesson 6: More Character Interaction using Message Trigger and Stop Block  Lesson 7: New Pages  Lesson 8: The Paint Editor  **Scratch Jr Activities**  Lesson 1 – Drive a car  Lesson 2 – Run a race  Lesson 3 - Sunset  Lesson 4 – sun and moon  Lesson 5 – Spooky Forest  Lesson 6 – Basket ball  Lesson 7 – Dance Party (adapted to contain a loop) | **Scratched**  Unit 1 – Step by Step  Unit 1 – Debug it  Unit 1 – 10 blocks  Unit 1 - About me  Unit 2 – Build a band  Unit 2 – Orange square purple circle  Unit 2 – It’s alive  Unit 2 – Debug it  Unit 2 – Music video  Unit 3 - Debugging | | **Scratched**  Unit 1 – Step by Step  Unit 1 – Debug it  Unit 1 – 10 blocks  Unit 1 - About me  Unit 2 – Build a band  Unit 2 – Orange square purple circle  Unit 2 – It’s alive  Unit 2 – Debug it  Unit 2 – Music video  Unit 3 - Debugging | | **Scratched**  Unit 4 – starter games  Unit 4 – Score  Unit 4 – Extensions  Unit 4 – Interactions  Unit 4 – Debug it  Unit 5 – Advanced Concepts  Overall Unit 6 – Hackathon (series of 8 lessons) | | **Scratched**  Unit 4 – starter games  Unit 4 – Score  Unit 4 – Extensions  Unit 4 – Interactions  Unit 4 – Debug it  Unit 5 – Advanced Concepts  Overall Unit 6 – Hackathon (series of 8 lessons) | |
| **REPEAT LOOPS**  **(iteration)** |  | | Loop a set of commands by a given amount.  Use a number to specify movement rather than repeated commands (e.g. in Scratch JR enter 🡩4 rather than 🡩🡩🡩🡩) | | Loop a set of commands by a given amount. | Understand what simple **loops** are and how they can make a program more efficient.    Identify repeat loops in everyday life | | Understand what **loops** are and how they can make a program more efficient.  **Pattern spotting** - be able to identify which commands need to be repeated and how many times to achieve a desired end. | | Describe what **for loops** are.  Use the instruction **repeat until …**    Read, write and debug **nested loops** (loops within a loop)  See the source image | | To read and write **loops** in Scratch code. | |
| **UNPLUGED RESOURCES** |  | |  | |  |  | |  | |  | |  | |
| **PLUGED RESOURCES** |  | | Lesson 6 – Basket ball  Lesson 7 – Dance Party (adapted to contain a loop) | | Lesson 6 – Basket ball  Lesson 7 – Dance Party (adapted to contain a loop) | Unit 1 – 10 blocks  Unit 1 - About me  Unit 2 – Build a band  Unit 2 – Orange square purple circle  Unit 2 – It’s alive  Unit 2 – Music video | | Unit 1 – 10 blocks  Unit 1 - About me  Unit 2 – Build a band  Unit 2 – Orange square purple circle  Unit 2 – It’s alive  Unit 2 – Music video  Unit 2 - Animations | | Unit 4 – starter games  Unit 4 – Score  Unit 4 – Interactions  Unit 4 – Debug it  Unit 5 – Advanced Concepts  Overall Unit 6 – Hackathon (series of 8 lessons) | | Unit 4 – starter games  Unit 4 – Score  Unit 4 – Interactions  Unit 4 – Debug it  Unit 5 – Advanced Concepts  Overall Unit 6 – Hackathon (series of 8 lessons) | |
| **EVENT HANDLING SKILLS** | Know that pressing Go will make the robot move.  [IMG_2795](https://enablingenvironments.files.wordpress.com/2015/02/img_2795.png) | Use the yellow green flag block means that clicking the green flag runs the program. | | Two events can be sequenced at the same time.  The green flag will run the program, clicking the Sprite runs a second event. | | | Use yellow blocks to sequence commands and run events.    Be able to create an animation or game using an existing template or scaffold | | Be able to create an animation or game  **Parallelism** – Allow more than one event to happen at the same time e.g. having more than one set of blocks or instructions running at the same time.  See the source image | | See Sequencing Strand | | See Sequencing Strand |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **UNPLUGED RESOURCES** |  |  | |  | |  | |  | |  | | |  | |
| **PLUGED RESOURCES** |  | **Scratch Jr Playground**  Lesson 2: Backgrounds and Start on Green Flag Triggering Block  Lesson 3: New Characters and Start on Tap Triggering Block  Lesson 4: Recording Sound, and Using the Wait Block and the Speed Block  Lesson 5: Simple Character Interaction using Start on Bump  Lesson 6: More Character Interaction using Message Trigger and Stop Block  Lesson 7: New Pages  Lesson 8: The Paint Editor  **Scratch Jr Activities**  Lesson 1 – Drive a car  Lesson 2 – Run a race  Lesson 3 - Sunset  Lesson 4 – sun and moon  Lesson 5 – Spooky Forest  Lesson 6 – Basket ball  Lesson 7 – Dance Party (adapted to contain a loop) | | **Scratch Jr Playground**  Lesson 3: New Characters and Start on Tap Triggering Block  Lesson 4: Recording Sound, and Using the Wait Block and the Speed Block  Lesson 5: Simple Character Interaction using Start on Bump  Lesson 6: More Character Interaction using Message Trigger and Stop Block  Lesson 7: New Pages  Lesson 8: The Paint Editor  **Scratch Jr Activities**  Lesson 1 – Drive a car  Lesson 2 – Run a race  Lesson 4 – sun and moon  Lesson 5 – Spooky Forest  Lesson 6 – Basket ball  Lesson 7 – Dance Party (adapted to contain a loop) | | **Scratched**  Unit 1 – Step by Step  Unit 1 – 10 blocks  Unit 1 - About me  Unit 2 – Build a band  Unit 2 – Orange square purple circle  Unit 2 – It’s alive  Unit 2 – Music video | | **Scratched**  Unit 1 – Step by Step  Unit 1 – 10 blocks  Unit 1 - About me  Unit 2 – Build a band  Unit 2 – Orange square purple circle  Unit 2 – It’s alive  Unit 2 – Music video  Unit 3 – Characters  Unit 3 – Conversations  Unit 3 - Scenes | |  | | |  | |
| **CONDITIONAL STATEMENTS SKILLS** |  | | | | | Understand that we can make actions occur only ​ under certain conditions.  Use IF statements in everyday life and in coding  See the source image | | Understand **conditional statements** as a way of ​ handling different situations (using If, Then, Else commands)  See the source image | | Describe what **Conditionals** are.  Read **conditional statements** as Scratch code. | | | Describe what **Conditionals** are.  Read **conditional statements** as Scratch code. | |
| **PLUGED RESOURCES** |  | | | | | Unit 3 Conversations | | Unit 3 Conversations | | Unit 4 – Extensions  Unit 5 – Advanced Concepts  Overall Unit 6 – Hackathon (series of 8 lessons) | | | Unit 4 – Extensions  Unit 5 – Advanced Concepts  Overall Unit 6 – Hackathon (series of 8 lessons) | |
| **VARIABLES SKILLS** |  | | | | | | | | | | Understand **variables** as a way of working with changing values.  Image result for Scratch Variable Blocks | | Describe what **variables** are and how to use them in Scratch code. | |
| **PLUGED RESOURCES** |  | | | | | | | | | | Unit 4 – Score  Unit 5 – Advanced Concepts  Overall Unit 6 – Hackathon (series of 8 lessons) | | Unit 4 – Score  Unit 5 – Advanced Concepts  Overall Unit 6 – Hackathon (series of 8 lessons) | |
| **TINKERING OPPORTUNITIES** |  | |  | |  | |  | |  | | |  | |  |