hi-impact

Teaching Computer Science with Code Studio

Code Studio is designed with the idea that the teacher will also act as the lead learner. As the lead learner, the teacher's role begins with being the exponent of knowledge. The lead learner's mantra is: "I may not know the answer, but I know that together we can figure it out."

As the teacher, you'll start by introducing the Computer Science concept to your students, presenting it as an abstract idea. To help them understand it better, you'll relate it to their everyday experiences and show how it fits into different contexts. Using simple and concrete examples, you'll make sure they grasp the concept easily, and video resources will be there to support this process.

Afterward, you'll guide your students in organising the knowledge back into its abstract form. This might involve creating algorithms, which will later be translated into code for the computer to execute.

For each lesson, the support material will include essential vocabulary and suggestions for a teaching sequence. Additionally, a series of slides will be available to help you lead the class smoothly through the learning journey. Your guidance and creativity will play a crucial role in making the subject engaging and accessible to the students.

In the KS1 courses, computational knowledge and skills development is best addressed through work on spatial skills and supported by physical computing. This is reflected in the course content, using directional arrows and cardinal points in early programming blocks.

Strategies that teachers may demonstrate through Code Studio include:

- Parson's problems: Complete prewritten programs or algorithms, where lines or blocks are mixed up and need correct arrangement. These problems may include distractors.

- Subgoal labelling or TIPP and SEE - Textual labels or hints are provided for component steps of a problem-solving process in worked examples.

- PRIMM methods where pupils take a staged approach to programming through states of: predict, run, investigate, modify and make.

- In Code Studio, every lesson includes important concepts and important words for students to learn. Students review these concepts multiple times across different lessons to help them truly understand and remember them.

The activities in Code Studio are designed to help students solve problems in a more abstract way. We focus on the main ideas first and avoid getting bogged down by unnecessary details until students have a solid grasp of the core concepts. Once they have a good understanding, they are challenged with more complex tasks that require them to apply the knowledge they've gained from earlier lessons. This way, they learn how to use the key information effectively to solve tough problems.

The teacher regularly intervenes in the lessons to ensure any misconceptions are addressed either as a whole class or with groups of students. The teacher dashboard will highlight this in real-time.

In the lessons for Key Stage 1 (KS1) and Key Stage 2 (KS2), students have clear and focused instruction on the essential knowledge of computing science. The topics we cover are outlined as follows:

Year 1	Year 2	Year 3	Year 4	Year 5	Year 6
Sequencing (4 lessons)	Sequencing (4 lessons)	Sequencing (5 lessons)	Sequencing (3 lessons)	Sprites (3 lessons)	Sprites (4 lessons)
Loops (4 lessons)	Loops (4 lessons)	Loops (4 lessons)	Events (4 lessons)	Game Design (5 lessons)	Variables (6 lessons)
Events (2 lessons)	Impacts of Computing (1 lesson)	Events (3 lessons)	Loops (3 lessons)	Functions (3 lessons)	Data & Simulations (4 lessons)
	Events (2 lessons)	Data (2 lessons)	Conditionals (5 lessons)	Conditionals (3 lessons)	

The end-of-course project and other mini-projects within the course, allow pupils to apply the component knowledge and skills learnt in a wider context.

PRIMM Methods

Predict, Run, Investigate, Modify, Make (PRIMM) is a structured sequence of activities for teaching programming concepts to learners. It starts with the presentation of a program that students are asked to read. Then, working in pairs, they make a written prediction of what the program does. When everyone has made their predictions, the code is run online so students can quickly check their predictions and possibly discuss them with the class.

To support reflection on the predictions and the actual results from running the program, the teacher provides a set of activities (e.g. code tracing, guided explorations, answering questions, and debugging) to help students investigate the structure of the program. To continue the exploration of the proposed program, the students are challenged to modify it, changing its functionality at progressive levels of complexity. During this phase, when moving from using somebody else's program to modifying it, there is a transfer of ownership from "not mine" to "partly mine".

Once the students are confident in modifying the program, they can make their own versions, reusing the same/modified structures in a new program. The PRIMM approach builds upon and combines results from several research studies into supporting students in aspects of learning to program: reading and tracing code, adapting existing code, and moving between different levels of abstraction. The PRIMM portal (https://primmportal. com) provides a set of lesson plans to scaffold lower secondary students learning to program in Python.

https://publications.jrc.ec.europa.eu/repository/handle/JRC128347

Subgoal labelling or TIPP and SEE

TIPP and SEE is a structured scaffolding strategy to help students learn programming by using and modifying existing Scratch projects to create their own. Scratch projects are organised around sprites, each having scripts that are executed in response to events. The teacher proposes a project that uses a particular programming concept that students investigate so they can understand how it works. They do this by running the project, looking inside to examine sprite scripts, and exploring what happens after modifying the scripts.
This strategy is a specialisation of the Use-Modify-Create approach tailored to the Scratch programming environment, which features a platform for users to publish and share their projects. Each Scratch project has a title and instructions on how to use it, so the first part of the strategy guides the students to get a TIPP from the project web page, namely: Does the Title tell you something about the project?; What do the Instructions tell you to do?; What is the Purpose of this activity?; Play with the project, run the code and observe what happens.

In the second part, the learner is guided to SEE inside the project to read and comprehend the code. The suggested roadmap (SEE) is to start by clicking on the Sprite to examine, then look at the Events blocks and the associated scripts (the teacher provides the students with worksheets that set questions and tasks to perform). Finally, the worksheet has suggestions to Explore by modifying the scripts. TIPP and SEE is described on the developers' website (https://www.canonlab.org/) along with the Scratch Encore Curriculum (https://www.canonlab.org/scratchencore) for introducing Computer Science to primary school students.