Computing at School

Powered by bcs The Chartered Institute for IT

Primary Pedagogy

A guide for Primary Teachers on five pedagogical approaches

Primary Pedagogy

A guide for Primary Teachers on five pedagogical approaches

What you'll find in this guide

- PRIMM (KS1)
- Unplugged
- Parson's problems
- Physical computing
- Culturally relevant approaches

This guide has been designed so that you can use it as a digital document or easily print it as an A5 booklet.

Why not keep it as a quick reference in your classroom or staff room.

With thanks to Neil Rickus for content development.

Introduction

Impactful learning relies on teacher knowledge in three key areas:

1. **Knowledge of the subject:** What the teacher knows about the lesson content they are delivering.
2. **Knowledge of teaching methods:** The approaches and strategies the teacher plans to use to deliver the lesson content.
3. **Knowledge of the pupils:** What the teacher knows about the class, including prior knowledge, required adaptations, misconceptions, and interests.

This is commonly referred to as pedagogical content knowledge (PCK), a concept introduced by Shulman in 1986.

This guide aims to introduce some of the most popular approaches to delivering the content of the computing curriculum to primary-aged learners. While many teaching approaches used in other areas of the curriculum can be successfully utilised when teaching primary computing content (such as modelling, scaffolded examples, and prediction), the technical aspect of computing lessons means that some teachers may have had little experience with the most effective approaches in this subject.

In this guide, you will find an overview of these approaches and suggestions on how they can be used in primary classrooms. It also signposts where further information and support on each approach can be found.

PRIMM

What is PRIMM?

PRIMM stands for **Predict**, **Run**, **Investigate**, **Modify**, **Make**, which represent stages of a programming lesson. Each stage enables pupils to undertake specific tasks when learning to program.

How does it help students?

Pupils can read sections of code before writing it. This encourages discussion and helps reduce cognitive load when developing their own programs. Pupils gradually take ownership of their programs as they progress through the stages. PRIMM also provides a familiar structure for pupils to follow in lessons.

Predict	Pupils discuss an existing program and predict what it might do.
Run	Pupils run the program to check their prediction.
Investigate	Activities are provided to investigate the code, such as explaining what different parts do or labelling the functionality of certain blocks.
Modify	Pupils make modifications to the program by undertaking tasks that become progressively more challenging.
Make	Pupils apply their knowledge from the previous stages by making their own program.

What can I try in lessons?

PRIMM could be used to develop KS1 pupils' understanding of the movement blocks in ScratchJr. If you are new to Scratch Jr check out the website's Teach Section here: www.scratchjr.org/teach/activities.

Predict

Show pupils the following blocks on the screen. Ask pupils to discuss what they think the program does, including explaining their reasoning.

Run

Ask pupils to use ScratchJr to run the program to see if they were correct. Discuss any incorrect predictions and clarify any misconceptions.

Investigate

Ask pupils to investigate the program by answering questions related to the functionality of each block. For example, pupils might be asked, "what does this block do?" and "how many times does the instruction take place?".

Modify

Ask pupils to change the program by completing a number of tasks. For example, pupils might be asked to make the move right block take place 10 times, or add a jump block to the end of the program.

Make

Ask pupils to create their own program based on the one they have been exploring. It should follow the same structure as the original program. Before starting the "Make" section, use the resources from Scratch Jr's "Teach" section to allow pupils to rehearse adding sprites and backgrounds.

Unplugged Computing

What is Unplugged Computing?

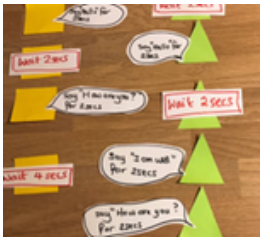
Computing unplugged activities allow children to learn about computing concepts without the use of technology. Pupils use familiar contexts and their existing knowledge to develop their knowledge and understanding.

How does it help pupils?

Pupils can focus on the concept, rather than how to use the technology. To ensure children understand the relevance of the unplugged activity, it is important to link the activity back to the concept on the computer.

How can I use an unplugged resource?

Programming a conversation is a good way to help pupils develop their understanding of sequence, as they need to think about what actions are carried out by the sprite who is currently not talking. Before starting the coding, pupils can act out and write their own conversation algorithms, allowing them to consider the need for the non-speaking character to wait while the other is speaking. This understanding is then transferred when they learn how to code a conversation in Scratch, as shown on the next page:



A conversation algorithm for pupils to act out

The conversation algorithm implemented as a program using Scratch

What else can I try in lessons?

KS1 - Pupils could develop their understanding of precise instructions by following algorithms to move around a floor map and then exploring different ways to write a direction-based algorithm for others to follow (using symbols, words, or abbreviations). This understanding can then be transferred when using floor robots such as the BeeBot.

KS2 - Pupils can plan and carry out their own dance sequences, identifying where actions are repeated and exploring how to use repetition to make their algorithms more succinct. They can then use repetition in Scratch to program a sprite to dance, as shown in this example: scratch.mit.edu/projects/10128067/

Parson's Problem

What is a Parson's Problem?

A Parson's Problem helps children solve a problem by providing them with all the blocks required. However, the blocks have to be rearranged to complete the task.

How does it help students?

Having the required blocks to solve a problem is less daunting than a blank screen and reduces cognitive load. Children can also discuss how to solve the problem and collaborate on solutions.

Can you give an example?

Pupils could be asked to rearrange the blocks on the right so that: when the green flag is clicked, the sprite says "Wow!" for 2 seconds, then play the sound "Alert" until done, then move 10 steps.

What can I try in lessons?

KS1 - Give pupils the required instructions to move Bee-Bot around a mat. Ask them to put the instructions in the right order and to explain how they know the order is correct.

KS2 - When programming two sprites having a conversation in Scratch, display the required blocks on screen. Ask children to discuss the correct order of the blocks.

Physical Computing

What is Physical Computing?

Physical computing devices enable pupils to interact with the real world. Children can program outputs, such as lights, buzzers and motors, along with using a range of inputs, such as buttons and sensors.

What are the benefits?

Physical computing devices provide further opportunities for pupils to develop their knowledge and understanding of programming. It also helps them appreciate how the technological world around them functions.

What devices are available?

In KS1 pupils commonly use **BeeBots** (www.tts-group.co.uk/bee-bot-programmable-floor-robot/1015268.html) and **EaRL** (www.hope-education.co.uk/earl). While in KS2 pupils typically use the **micro:bit** (microbit.org) or the **Crumble Controller** (redfernelectronics.co.uk/crumble/). Both devices can be programmed using laptops or tablet devices.

What can I try in lessons?

KS1 - Pupils could program a BeeBot or EaRL robot to move around a floor map and reach specific locations

KS2 - Pupils could use the Crumble to program lights and a motor for a fairground ride they have made in DT.

Culturally relevant approaches

What are culturally relevant approaches?

Culturally relevant approaches involve recognising pupils' own knowledge and cultural experiences. This ensures the curriculum, teaching approaches and learning materials are relevant, engaging and accessible.

What are the benefits?

Pupils recognise the value of computing, its relevance to themselves and the world around them. Pupils' interest and attitude towards the subject are also improved.

What can I try in lessons?

Use contexts and images in lessons that show diversity and represent the experiences and cultures of pupils.

KS1 - When developing pupils' use of graphics software, allow them to develop art that reflects their cultural heritage (e.g. Islamic art).

KS2 - When pupils are creating digital content (presentations, podcasts, animations, etc.), allowing them to draw on their own experiences. Creating a digital poster about a celebration rather than a specific celebration.

Useful links

More information on PRIMM from Barefoot:
www.barefootcomputing.org/my-barefoot/research-zone/primm

Barefoot PRIMM resources:
Crystal Flowers PRIMM 1 -
www.barefootcomputing.org/resources/crystal-flowers-primm-1

Crystal Flowers PRIMM 2 -
www.barefootcomputing.org/resources/crystal-flowers-primm-2

More information on Parson's Problem from Teach Computing:
teachcomputing.org/blog/quick-read-improving-program-comprehension-throughparsons-problems/

More information on Cultural relevant approaches from Raspberry Pi:
www.raspberrypi.org/culturally-responsive-pedagogy-for-computing-education/

CAS Primary

Scan the QR code to find all CAS has to offer for Primary and EYFS teachers - resources, events, blogs and our discussion forum. And why not join the CAS Primary Community?

SCAN HERE

Computing at School

Powered by bcs