

Computing Science Glossary

Sequence One thing following another in which the order may or may not be important.

Parsons Code is provided but not connected for pupils to connect

PRIMM A strategy that promotes Predicting, Running, Investigating and Modifying code before Making something.

IPRIMM Stages of PRIMM but in a different order

Algorithm Part of planning stage before programming written for another human to read.

Code Written for a digital device

USE MODIFY MAKE A strategy that promotes using and modifying code before creation

VIEW DESIGN USE MODIFY MAKE A modification of use modify create where pupils view the design first.

Loop An event or pattern of events repeated

Helicopter Game

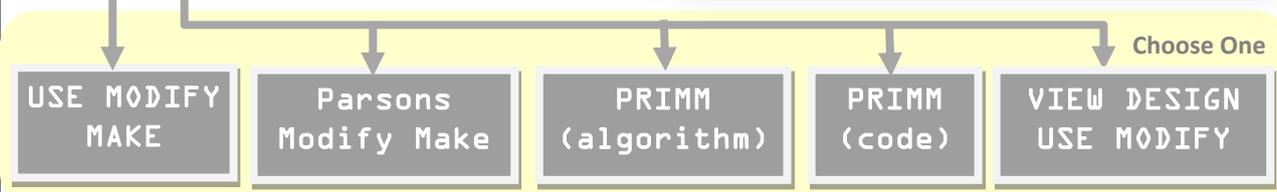
Questions & Answers

How does this fit in with other game programming? See overview document
What age is this for? KS2 (7-11 year olds). **How hard is this to teach?** Very easy as all the code instructions are in the booklets. **How do we Assess learning?** Pupils use answers provided to mark their own work. **Is it in line with NC?** Yes see next page. **Why is there a choice?** All of the methods chosen have good research behind them but we don't know which are best or even if there is a best for all pupils. We do know that it helps pupils to encounter a variety of different types of method so they are continually challenged. If you work your way through all modules I recommend you switch strategies each time to keep the challenge high. **How creative is this?** This combines the best knowledge we have about how to learn something new with the opportunity at the end for pupils to create something that they want to create that uses their new knowledge.

Role-play & write Continuous Loops

Download Concepts before coding PDF from <http://code-it.co.uk/wp-content/uploads/2019/04/conceptbeforecoding.pdf> Follow the links in the menu to continuous loops. Use slides 34-39 to roleplay and write simple continuous loops. Make the point that you are using loop always but the Scratch code will use forever. You do this so that pupils develop an idea of continuous loops away from code making it easier to transfer the idea between different programming languages.

Role-play & Write Continuous Loops



Booklet Choices

You choose from one of the options above. Each option has its own booklet which guides pupils through the stages, making them think deeply about either the code or the algorithm before modifying it and having a choice of things to make. Pupils are instructed when to work in pairs and when to work alone. Cut up the answer sheet (last pages of the booklet) into sections as pupils will need to mark their work as part of their learning process. If you are not sure which to choose download the booklets and look at the differences.

National Curriculum Programs of Study

(bold text is covered in this module)

Pupils should be taught to:

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

Before the module

Read the planning and download the PDFs for everyday loops and role-play and write loops. Decide which booklet variation you are going to use and download and print it out one per pupil. Remove the answer sheets for pupils and trim them into sections for pupils to access when they need to. Download the code for your version of Scratch 2 or 3 and place it on your network where pupils can access it, or note where it is on the Scratch website if using Scratch online

Formative assessment support

Asking pupils to read code or algorithm out loud also helps if pupils are struggling to predict, investigate or modify.

If pupils are struggling to work together in a meaningful way then encouraging and rewarding positive attitudes to working collaboratively using the communicates stickers shown at the end helps.

Lots of misconceptions can be solved by reading the code or algorithm slowly and out loud to their partner.

If pupils are struggling to plan a new game it helps to see what a good version might look like. The first page of the view design worksheet has a planning example.

Classroom Organisation

In some sections pupils are asked to work with a partner of similar programming ability. If you are not sure what programming ability they are go with Maths skills as a starting place. Move partners around between modules so that pupils benefit from different interactions.

Research Help

Get pupils to mark their booklets, collect in all the marks by sections. Collect the marks from the modify section and calculate a mean average for the whole class. Email or Tweet this to phil.bagge@code-it.me or @baggiepr stating clearly what age, module and version your class did. For example

Resources

Role-play & Write Continuous Loops

[PDF Download](#) Slides 34-39
(10 mins)

Parsons Modify Make

Pupil booklets
Scratch 2 & 3 Code to download
Scratch 3 code on Scratch website

PRIMM Algorithm & Code

Pupil booklets
Scratch 2 & 3 Code to download
Scratch 3 code on Scratch website

USE MODIFY CREATE

Pupil booklets
Scratch 2 & 3 Code to download
Scratch 3 code on Scratch website

VIEW DESIGN USE MODIFY CREATE

Pupil booklets
Scratch 2 & 3 Code to download
Scratch 3 code on Scratch website

All Resources at

<http://code-it.co.uk/goldgame/>

Further Research Reading



Use Modify Create

Irene Lee et al Computational thinking for Youth in practice (2011)

PRIMM Sentence

<https://blogs.kcl.ac.uk/cser/2017/09/01/primm-a-structured-approach-to-teaching-programming/>

Four levels of abstraction

This article includes an example of the four levels of abstraction and sign posts the work of Waite and Armoni

<http://code-it.co.uk/algprogdiff/>

Y4 8-9 Years old Helicopter PRIMM Code Mean average 5.2/9 for 32 pupils.



Inspired by Behaviour Rubric created with @MarkDorling and linked at <http://code-it.co.uk/attitudes/>

@baggiepr