Year 7 — Programming I

Unit introduction

This unit is the first programming unit of KS3. The aim of this unit and the following unit ('programming 2') is to build learners' confidence and knowledge of the key programming constructs. Importantly, this unit does not assume any previous programming experience, but it does offer learners the opportunity to expand on their knowledge throughout the unit.

The main programming concepts covered in this unit are sequencing, variables, selection, and count-controlled iteration. All of the examples and activities for this unit use Scratch 3.

Overview of lessons

Lesson	Brief overview	Learning objectives
Lesson 1: Introduction to programming and sequencing	Learners will be introduced to the unit and will take part in an activity to help them understand the precise nature of instructions that computers need to execute. Learners will be taught the song *Frère Jacques* before working in pairs to place blocks of code into the appropriate subroutines so that their program will play the song correctly.	• Compare how humans and computers understand instructions (understand and carry out) • Define a sequence as instructions performed in order, with each executed in turn

		• Predict the outcome of a simple sequence • Modify a sequence
Lesson 2: Sequence and variables	In this lesson the learners will be introduced to variables as well as the opportunity to get more confident with sequences. The lesson will start with a story that includes variables; learners have to replace the variable names with the values they refer to when they reach the relevant places in the story. Learners will then be given a Scratch program where they will work in pairs to predict, run, investigate, and modify. The lesson will conclude with an activity requiring the learners to trace the value of a variable in an algorithm.	• Define a variable as a name that refers to data being stored by the computer • Recognise that computers follow the control flow of input/process/output • Predict the outcome of a simple sequence that includes variables • Trace the values of variables within a sequence • Make a sequence that includes a variable
Lesson 3: Selection	The focus of this lesson is to introduce learners to the concept of selection statements and how they can be used to control the flow of a program. The lesson starts with activities that allow the learners to understand expressions that evaluate to 'true' or 'false'. This will be followed by a PRIMM activity using another version of the 'Chat with Big Ed' program from the last lesson, this time using selection (**If** statements). The lesson will finish with a Parson's Problem that requires the learners to rearrange code to form a working program.	• Define a condition as an expression that will be evaluated as either true or false • Identify that selection uses conditions to control the flow of a sequence • Identify where selection statements can be used in a program • Modify a program to include selection

Lesson 4: Operators	This lesson will build on the previous lesson by introducing the use of logical and comparison operators to use in selection statements. The learners will start by following Scratch code and working out what the program will output given different inputs. They will be introduced to logical and comparison operators before taking part in an activity where they are given a playing card and have to decode if it evaluates to 'true' or 'false' using various different expressions. The learners will then build a 'Brain game' Scratch program by adding new questions to subroutines.	• Create conditions that use comparison operators (>,<,=) • Create conditions that use logic operators (and/or/not) • Identify where selection statements can be used in a program that include comparison and logical operators
Lesson 5: Count-controlled iteration	In this lesson learners will be introduced to the concept of iteration, the examples will be specifically focused on count-controlled iteration. The learners will be given an inefficient program and be asked to spot patterns and repetition. They will be taken through a live coding demonstration of taking their inefficient program and adding iteration to make it more efficient. Learners will then use pair programming to create a Scratch version of the nursery rhyme *Ten Green Bottles* using count-controlled iteration. Finally learners will be introduced to the concept of debugging and they will be given a program to debug by tracing the value of the variables.	• Define iteration as a group of instructions that are repeatedly executed • Describe the need for iteration • Identify where count-controlled iteration can be used in a program • Implement count-controlled iteration in a program • Detect and correct errors in a program (debugging)
Lesson 6: Problem-solving	This is the final lesson of the first unit of programming in Year 7. The lesson starts with a game of 'Beat the teacher' where the learners are required to write down as many as the key words relating to this unit that they can. After a minute, the teacher will also play and the learners will see if they'd written down more words than the teacher. The main activity for the lesson will be learners' main summative assessment task where they are required to independently work	• Independently design and apply programming constructs to solve a problem (subroutine, selection, count-controlled iteration, operators, and variables)

	through tasks to complete a dance move game. The plenary for the lesson allows the learners an opportunity to reflect and assess the skills that they have developed throughout the unit.	

Progression

Please see the learning graph for this unit for more information about progression.

Note: this also covers objectives for Year 7 — Programming II

Curriculum links

National curriculum links

- Use two or more programming languages, at least one of which is textual, to solve a variety of computational problems; make appropriate use of data structures (e.g. lists, tables, or arrays); design and develop modular programs that use procedures or functions
- Understand several key algorithms that reflect computational thinking; use logical reasoning to compare the utility of alternative algorithms for the same problem
- Understand simple Boolean logic (e.g. and, or, and not)
- Create, reuse, revise, and repurpose digital artefacts for a given audience, with attention to trustworthiness, design, and usability

Assessment

Summative assessment

The final lesson of the unit requires learners to complete a set of tasks using a Scratch program. Please see the assessment rubric document for this unit, and the Scratch Program Moves like Jim (ncce.io/moveJim).

Multiple choice questions

This unit contains two homework activities (in Lesson 2 and Lesson 5) that ask a set of multiple choice questions. The intention of the questions is to help you pick up on which (if any) misconceptions have been picked up by the learners so that they can be addressed before moving on. Alternatively, you could ask the full set of questions at the end of the unit to help inform planning ahead of the next unit.

Subject knowledge

This unit focuses on the development of the following key techniques:

- Sequencing
- Variables
- Selection
- Operators
- Count-controlled iteration

Scratch is used throughout the unit so it is important that you are comfortable with the language. In order to get a feel for the level of skill required, why not try the activities in the unit yourself before using them with your learners?

Enhance your subject knowledge to teach this unit through the following training opportunities:

Online training courses

- Secondary Programming Pedagogy: Inspiring computing teaching (ncce.io/secondarypedagogy)
- Programming 101 (ncce.io/prog101)
- Programming 102 (ncce.io/progr102)

Face-to-face courses

- Key Stage Three computing for non-specialists (ncce.io/f2fks3)