Year 3 – Programming A - Sequencing sounds

Unit introduction

This unit explores the concept of sequencing in programming through Scratch. It begins with an introduction to the programming environment, which will be new to most learners. They will be introduced to a selection of motion, sound, and event blocks which they will use to create their own programs, featuring sequences. The final project is to make a representation of a piano. The unit is paced to focus on all aspects of sequences, and make sure that knowledge is built in a structured manner. Learners also apply stages of program design through this unit.

If learners are using the online version of Scratch, be aware this allows them to share and comment on projects. A simplified version of the Scratch's community guidelines is included in lesson one, and can also be found at the end of this unit guide. For the full guidelines, see the Scratch website.

Software and Hardware requirements

Learners will need to have access to Scratch for this unit. The online version of Scratch runs via a web browser and can be accessed on desktops, laptops and tablets. You may want to consider setting up a teacher account, to create logins for learners to save and access their projects. If internet connectivity is an issue in school, Scratch can be accessed offline via the Scratch app.

If you've adapted this unit to better suit your school, please share your adapted resources with fellow teachers in the STEM community. Alternatively, if this unit isn't quite right for your school, why not see if an adapted version which better suits has already been shared?

Overview of lessons

Lesson	Brief overview	Learning objectives

1. Introduction to Scratch	This lesson introduces learners to a new programming environment: Scratch. Learners will begin by comparing Scratch to other programming environments they may have experienced, before familiarising themselves with the basic layout of the screen.	To explore a new programming environment • I can identify the objects in a Scratch project (sprites, backdrops) • I can explain that objects in Scratch have attributes (linked to) • I can recognise that commands in Scratch are represented as blocks
2. Programming sprites	In this lesson, learners will create movement for more than one sprite. In doing this, they will design and implement their code, and then will create code to replicate a given outcome. Finally, they will experiment with new motion blocks.	To identify that commands have an outcome • I can create a program following a design and understand that each sprite is controlled by the commands I choose • I can predict the coding blocks used to move a sprite • I can match coding blocks to their actions
3. Sequences	In this lesson, learners will be introduced to the concept of sequences by joining blocks of code together. They will also learn how event blocks can be used to start a project in a variety of different ways. In doing this, they will apply principles of design to plan and create a project.	To explain that a program has a start • I can start a program in different ways • I can create a sequence of connected commands • I can explain that the objects in my project will respond exactly to the code
4. Ordering commands	This lesson explores sequences, and how they are implemented in a simple program. Learners have the opportunity to experiment with sequences where order is and is not important. They will create their own sequences from given designs.	To recognise that a sequence of commands can have an order • I can explain what a sequence is • I can combine sound commands • I can order notes into a sequence

5. Looking good	This lesson develops learners' understanding of sequences by giving them the opportunity to combine motion and sounds in one sequence. They will also learn how to use costumes to change the appearance of a sprite, and backdrops to change the appearance of the stage. They will apply the skills in Activity 1 and 2 to design and create their own project, including sequences, sprites with costumes, and multiple backdrops.	To change the appearance of my project • I can build a sequence of commands • I can decide the actions for each sprite in a program • I can make design choices for my artwork
6. Making an instrument	In this lesson, learners will create a musical instrument in Scratch. They will apply the concept of design to help develop programs and use programming blocks — which they have been introduced to throughout the unit. They will learn that code can be copied from one sprite to another, and that projects should be tested to see if they perform as expected.	To create a project from a task description • I can identify and name the objects I will need for a project • I can relate a task description to a design • I can implement my algorithm as code

Subject knowledge and CPD opportunities

Throughout this unit, there are opportunities to model skills within Scratch, where concepts have been demonstrated through an embedded screen recording. Pedagogically, it is more beneficial to model the concepts to the learners, which allows for easier questioning and understanding. We recommend that you use the videos for your own subject knowledge, to see what needs to be modelled, and then give a live demonstration within the lesson. However, the videos are provided on the slides if you wish to use them instead.

This unit focuses on developing learners' understanding of sequences in a new programming language. It highlights that the order of sequences is important. This unit also develops learners' understanding of design in programming, using the approach outlined below.

When programming, there are four levels which can help describe a project (known as levels of abstraction). Research suggests that this structure can support learners in understanding how to create a program and how it works:

- Task - what is needed
- Design - what it should do
- Code - how it is done

- Running the code - what it does

Spending time at the task and design levels before engaging in code-writing can aid learners in assessing the 'do-ability' of their programs. It also reduces a learner's cognitive load during programming. Learners will move between the different levels throughout the unit and this is highlighted within each lesson plan.

Continual Professional Development
Enhance your subject knowledge to teach this unit through the following free CPD:

- Getting started in Year 3 – short course
- Introduction to primary computing remote or face to face
- Introduction to Programming with Scratch
- Teaching programming using Scratch and Scratch Jr

Teach primary computing certificate
To further enhance your subject knowledge, enrol on the teach primary computing certificate. This will support you to develop your knowledge and skills in primary computing and gain the confidence to teach great lessons, all whilst earning a nationally recognised certificate!

Progression

This unit assumes that learners will have some prior experience of programming; via the KS1 NCCE units. They will have experienced programming via floor robots; Moving A Robot Year 1 and Robot algorithms Year 2, alongside the use of ScratchJr through Programming animations Year 1 and Programming quizzes Year 2. ScratchJr uses a similar programming environment to Scratch, which is highlighted in lesson 1 of this unit.

Common Misconceptions

This unit introduces Scratch to learners. When using the software, learners should be aware that the colour of the coding blocks is important. The programming palette is organised into nine colour-code categories, and understanding this will help learners navigate the platform. For all projects to run, they must use an event command (e.g. when green flag is clicked) to start their program; learners often forget this, and as such their code will not start.

When working with multiple sprites on Scratch, learners may code the wrong Sprite. To avoid this, learners must click on the sprite to bring up its individual code tab, before starting.

Learners will need to understand that an algorithm is a precise set of ordered instructions, and that this is different from a program. They will also need to be aware that the sequence of their coding blocks is important, with the computer carrying out the commands in that order.

Curriculum links

Computing

- 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
- Select, use and combine a variety of software (including internet services) on a range of digital devices to design and create a range of programs, systems and content that accomplish given goals, including collecting, analysing, evaluating and presenting data and information

Assessment

Formative assessment

Assessment opportunities are detailed in each lesson plan. The learning objectives and success criteria are introduced in the slide decks at the beginning of each lesson and then reviewed at the end. Learners are invited to assess how well they feel they have met the learning objective using thumbs up, thumbs sideways, or thumbs down.

Summative assessment

Please see the assessment rubric document for this unit. The rubric can be used to assess student's work from lesson 6.

We recommend the use of teacher and learner accounts in Scratch to help with assessment throughout this unit. For guidance on setting up teacher accounts, visit scratch.mit.edu/educators/faq. A teacher account enables you to manage learners' accounts and organise projects into studios. If you are unable to use teacher and learner accounts, work can be saved offline to local devices.

Scratch guidelines

- **Stay Safe Online**: Don't share personal info like your full name, address, or phone number. Also, don't share details about where you go to school or your social media accounts.

- **Be Kind and Helpful**: When you comment on someone's project, say something nice about it and offer suggestions in a friendly way. Don't be mean or spammy.

- **Share and Collaborate**: You can use other people's stuff on Scratch to make your own cool projects but remember to give credit. And when you share your work, others can use it too, as long as they give credit and make changes.

- **Be Honest**: Always tell the truth and be yourself when you're on Scratch. Don't pretend to be someone else.

- **Keep Scratch Friendly**: Make sure your creations and chats are friendly for everyone. If you see something mean or inappropriate, you can click the link that says "report" on any project, comment, discussion post, studio, or profile page. If you're unsure or it's a bit complicated, you can ask your teacher or a trusted adult to get in touch with us. The Scratch team will take care of it.