

Year 4 – Repetition in games

Unit introduction

This unit explores the concept of repetition in programming using the Scratch environment. It begins with a Scratch activity similar to that carried out Code.org, Course D, where learners can discover similarities between two environments. Learners look at the difference between count-controlled and infinite loops, and use their knowledge to modify existing animations and games using repetition. Their final project is to design and create a game which uses repetition, applying stages of programming design throughout.

This unit complements the online coding course at Code.Org and reinforces skills and concepts acquired there whilst giving learners the opportunity to experience another block based, visual programming language.

Throughout this unit, there are opportunities to model within Scratch or to demonstrate a concept through a video. Pedagogically, it is more beneficial to model the concepts to the learners, which allows for easier questioning and understanding. We recommend that the videos are used by the teacher to see what needs to be modelled, but give a live demonstration within the lesson. However, the videos are provided on the slides if you wish to use them instead.

Overview of lessons

Lesson	Brief overview	Learning objectives
1. Using loops to create shapes	In the first lesson, learners look at real life examples of repetition, and identify which parts of instructions are repeated. Learners then use Scratch, a block-based programming environment, to create shapes using count-controlled loops. They consider what the different values in the loop signify, use existing code to modify and create new code, and work on reading code and predicting what the output will be once the code is run.	To develop the use of count-controlled loops in a different programming environment <ul style="list-style-type: none">• I can list an everyday task as a set of instructions including repetition• I can predict the outcome of a snippet of code• I can modify a snippet of code to create a given outcome

2. Different loops	In this lesson, learners look at different types of loops - both infinite loops and count-controlled loops. They practise using these within Scratch and think about which might be more suitable for different purposes.	To explain that in programming there are infinite loops and count controlled loops <ul style="list-style-type: none"> ● I can modify loops to produce a given outcome ● I can choose when to use a count-controlled and an infinite loop ● I can recognise that some programming languages enable more than one process to be run at once
3. Animate your name	In this lesson, learners create designs for an animation of the letters in their names. The animation uses repetition to change the costume (looks) of the sprite, and the letter sprites will all animate together when the event block (green flag) is clicked. Once they have designed their animations, they will program them in Scratch. After programming, learners then evaluate their work, considering how effective their use of repetition was.	To develop a design which includes two or more loops which run at the same time <ul style="list-style-type: none"> ● I can choose which action will be repeated for each object ● I can explain what the outcome of the repeated action should be ● I can evaluate the effectiveness of the repeated sequences used in my program
4. Modifying a game	In this lesson, learners look at an existing game and match parts of the game with the design. They make changes to a sprite in the existing game to match the design. They then look at a completed design, and implement the remaining changes in the Scratch game. They add a sprite, and re-use and modify code blocks within loops, and explain the changes made.	To modify an infinite loop in a given program <ul style="list-style-type: none"> ● I can identify which parts of a loop can be changed ● I can explain the effect of my changes ● I can re-use existing code snippets on new sprites
5. Designing a game	In this lesson, learners look at a model project using repetition. They then design their own game based on the model project, producing a design and algorithm for sprites in the game. They share these designs with a partner and have time to make any changes to their design as required.	To design a project that includes repetition <ul style="list-style-type: none"> ● I can evaluate the use of repetition in a project ● I can select key parts of a given project to use in my own design

		<ul style="list-style-type: none"> I can develop my own design explaining what my project will do
6. Creating our games	In this lesson, learners build their games, using the designs they created in lesson 5. They follow their algorithms, fix mistakes and refine designs in their work as they build. They evaluate their work once it is completed, and showcase games at the end.	<p>To create a project that includes repetition</p> <ul style="list-style-type: none"> I can refine the algorithm in my design I can build a program that follows my design I can evaluate the steps I followed when building my project

Progression

This unit assumes that learners will have some prior experience of programming; the KS1 NCCE units cover floor robots and ScratchJr, and Scratch is introduced in the Year 3 programming units. However, experience of other languages or environments may also be useful.

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

Curriculum links

National curriculum links

- 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

Assessment

Formative assessment

Assessment opportunities are detailed in each lesson plan. The learning objectives and success criteria are introduced in the slide deck 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.

We recommend the use of teacher accounts in Scratch to help with assessment throughout this unit. For guidance on setting up teacher accounts, please visit [the Scratch website](https://scratch.mit.edu/educators/faq). (<https://scratch.mit.edu/educators/faq>)

Subject knowledge

This unit focuses on developing learners' understanding of repetition within the Scratch programming environment. Repetition is where actions or commands in programming are repeated. The repeating commands can also be referred to as a loop. Loops can be repeated indefinitely and are known as infinite loops,, or for a set number of times, known as 'count controlled loops'.

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.

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

Online training courses

- [Raspberry Pi Foundation online training courses](#)

Face-to-face courses

- [National Centre for Computing Education face-to-face training courses](#)

Resources are updated regularly — please check that you are using the latest version.

This resource is licensed under the Open Government Licence, version 3. For more information on this licence, see [nccpe.io/ogl](https://www.nccpe.io/ogl).