

Sensory classroom

**Lessons:** 4

**Programming languages:** MakeCode

**Target age:** 11-14 yrs

**Subjects & topics:**

* Computational thinking: Algorithms, Abstraction
* Programming: Iteration
* Computer systems: Input/output

# Unit of work summary

In this series of four lessons students consider how a sensory classroom can be used to meet the needs of learners who are sensitive to sensory stimulus. They evaluate sensory aids to learn how they meet the needs of their users and use this understanding to plan and devise a classroom sensory aid using the micro:bit.

Ideally, this unit should be taught after [Computing fundamentals](https://microbit.org/teach/lessons/computing-fundamentals-unit-of-work/) and assumes students have experience of writing algorithms using pseudocode and have used the MakeCode editor.

## Overall key learning

* can understand and apply the fundamental principles and concepts of computer science
* have repeated practical experience of writing computer programs in order to solve problems
* can evaluate and apply information technology
* are responsible, competent, confident and creative users of information and communication technology

## Additional skills

Creative thinking, problem-solving, collaboration, evaluation

## Lesson 1: Exploring learning environments

Students consider the effects a standard classroom environment can have on learners who are sensitive to sensory stimuli, learn about sensory rooms and evaluate sensory aids.

**Key learning:**

* To understand that some learners are sensitive to sensory stimulus
* To know some benefits of sensory environments
* To evaluate sensory aids

## Lesson 2: Light patterns

Students create a light pattern for a sensory aid, writing and following an algorithm using pseudocode and iteration to program their micro:bit.

**Key learning:**

* To use pseudocode to write an algorithm for a light pattern
* To use iteration in algorithms and programs to create a repeating light pattern
* To evaluate an algorithm and program to ensure they meet criteria

## Lesson 3: Developing pattern algorithms

Students plan how to create a sensory aid to provide a visual sensory pattern using the BBC micro:bit. They write an algorithm using inputs, outputs, iteration and selection.

**Key learning:**

* To create a sensory aid for a classroom that meets given criteria.
* To use pseudocode to write an algorithm using inputs, outputs, iteration and selection

## Lesson 4: Building sensory aids

Students follow their algorithms to program their BBC micro:bit to act as a classroom sensory aid.

**Key learning:**

* To follow a design plan to create a classroom sensory aid that meets given criteria
* To follow an algorithm to create a program using inputs, outputs, iteration and selection
* To test and debug code and develop solutions to problems that may arise
* To evaluate the classroom sensory aid effectively

# Curriculum links

## England National Curriculum

#### KS3 computing curriculum

Curriculum aims:

* can understand and apply the fundamental principles and concepts of computer science, including abstraction, logic, algorithms and data representation
* can analyse problems in computational terms, and have repeated practical experience of writing computer programs in order to solve such problems
* can evaluate and apply information technology, including new or unfamiliar technologies, analytically to solve problems
* are responsible, competent, confident and creative users of information and communication technology.

Students should be taught to:

* design, use and evaluate computational abstractions that model the state and behaviour of real-world problems and physical systems
* use logical reasoning to compare the utility of alternative algorithms for the same problem
* use 2 or more programming languages, at least one of which is textual, to solve a variety of computational problems;
* develop their capability, creativity and knowledge in computer science, digital media and information technology
* develop and apply their analytic, problem-solving, design, and computational thinking skills

[Read the full KS3 computing curriculum](https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/239067/SECONDARY_national_curriculum_-_Computing.pdf)

## Scotland Curriculum for Excellence

#### Technologies

* I can describe in detail the processes used in real world solutions, compare these processes against alternative solutions and justify which is the most appropriate. (TCH 4-13a)
* I can informally compare algorithms for correctness and efficiency. (TCH 4-13b)

[Read the full Curriculum for Excellence: technologies](https://education.gov.scot/Documents/Technologies-es-os.pdf).

## Northern Ireland Curriculum

#### Science and technology - technology and design - statutory requirements, KS3

* Design – identifying problems; investigating, generating, developing, modelling and evaluating design proposals; giving consideration to form, function and safety;
* Explore technical inventions and designs that have met a social need cost-effectively.
* Design cost effective and appropriate solutions to meet the specific needs of diverse local and global groups.
* show deeper understanding by thinking critically and flexibly, solving problems and making informed decisions, using Mathematics and ICT where appropriate;
* demonstrate creativity and initiative when developing ideas and following them through;

[Read the full technology and design statutory requirements](https://ccea.org.uk/downloads/docs/ccea-asset/General/Statutory%20Requirements%20for%20Technology%20and%20Design%20at%20Key%20Stage%203.pdf)

#### Primary using ICT - desirable features - computational thinking and coding

**Level 5**

Pupils should:

* create more sophisticated coding projects using a broad range of commands and more than one platform; and
* solve a more complex problem using commands in a programming environment.

**Programmable devices (such as Parrot Drone, MicroBit or Sphero)**

* as a class look at and talk about examples of coding projects, including using multiple ‘if...then’ and ‘if...then...else’ commands, variables, sensors, events, operators and comparators;
* recognise how they can decompose these projects;
* in small groups, plan their own coding project, demonstrating a clear sense of purpose and audience, showing understanding of abstraction by deciding what details they need to include and what they can leave out, working out what different parts of the program must do and using logical reasoning to discuss and compare the commands that are required for their algorithm and predicting the outcome;
* use a range of commands to create a project, including variables, operators and control statements such as ‘if... then...’ alongside the use of ’if...then...else’ and comparators;
* test and debug at regular intervals and collaborate with others to solve problems as they arise;

**Finally**

* share their work (possibly using digital tools), respond to feedback, and comment on the work of others evaluating process and outcome; and
* organise files and publish work online (if available) so that others can view it.

[Read all Primary using ICT desirable features](https://ccea.org.uk/downloads/docs/ccea-asset/Curriculum/Primary%20Using%20ICT%20Desirable%20Features%20Update%202019.pdf)

## Curriculum for Wales

#### Science and technology

Progression step 4 - computation is the foundation for our digital world:

* I can decompose given problems and select appropriate constructs to express solutions in a variety of environments.
* I can plan and implement test strategies to identify errors in programs.

Progression step 5 - computation is the foundation for our digital world:

* I can identify, define and decompose problems, choose appropriate constructs and express solutions in a variety of environments.
* I can test, evaluate and improve a solution in software.

[Read the full science and technology curriculum](https://hwb.gov.wales/curriculum-for-wales/science-and-technology/descriptions-of-learning/)

#### Digital competence framework

Progression step 4 - data and computational thinking - problem-solving and modelling:

* I can create a simple model or self-contained algorithm.
* I can identify the different parts of an algorithm to determine their purpose.
* I can detect and correct errors in algorithms.

Progression step 5 - data and computational thinking - problem-solving and modelling:

* I can independently create and design models, and explain how they represent real-world problems, e.g. selecting and correctly using an appropriate method for illustrating a problem, such as a flowchart or spreadsheet.
* I can develop logical solutions to determine the input, outputs and processes of a program, e.g. following pseudocode or a flowchart to come to an outcome, developing a written sequence of steps that could be followed.

[Read the digital competence framework](https://hwb.gov.wales/curriculum-for-wales/cross-curricular-skills-frameworks/digital-competence-framework)

## Code.org

#### CS Discoveries

Unit 1

Concepts included:

* problem solving
* inputs and outputs
* storing and processing information

Unit 4

Concepts included:

* social impact of computing
* understanding the needs of others when designing a solution
* team project
* testing and acting on feedback
* iteration

[Read the full Code.org CS Discoveries curriculum](https://studio.code.org/courses/csd-2021)

## USA CSTA Standards

#### Grades 6-8

* 2-AP-10 - Use flowcharts and/or pseudocode to address complex problems as algorithms.
* 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.
* 2-AP-17 - Systematically test and refine programs using a range of test cases.

[Read the CSTA Standards in full.](https://csteachers.org/k12standards/ )

This content is published under a [Creative Commons Attribution-ShareAlike 4.0 International (CC BY-SA 4.0)](https://creativecommons.org/licenses/by-sa/4.0/) licence.