DigitalSkills.org

Teacher Learning Plan

Digital Skills Curriculum 2024/25

Year 7

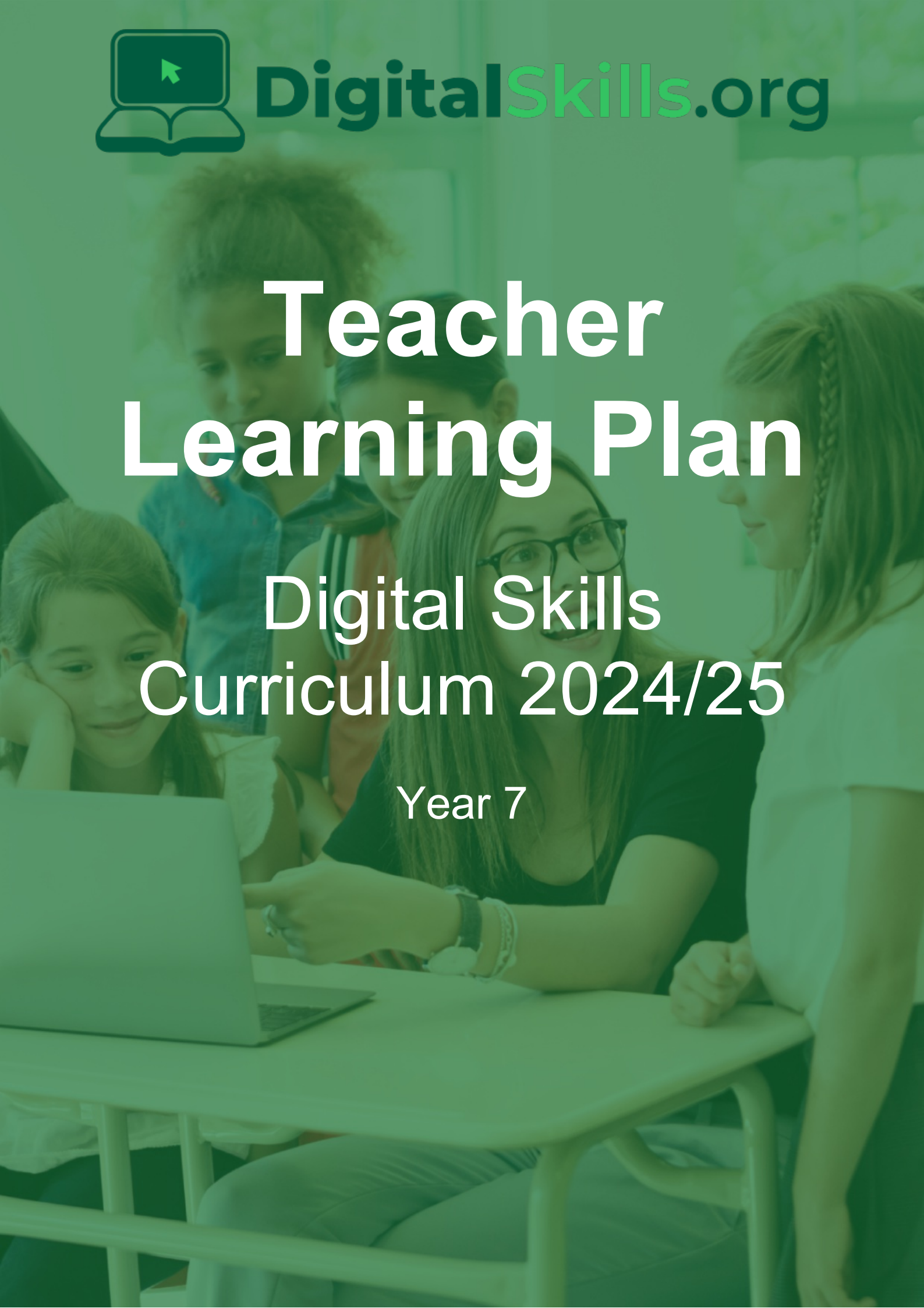


Table of Contents

How to Use This Learning Plan

This learning plan provides an overview of all the modules available for Year 7, including their units, learning goals, and outcomes. Each module is designed to support both new and experienced teachers with easy-to-follow, step-by-step lessons.

Lesson Types

There are two types of lessons in the Digital Skills Curriculum:

- **Teacher-Led Lessons** – The teacher directs and leads students through the lesson, guiding them through the activities and discussions.
- **Teacher/Student-Led Lessons** – Teachers can choose to lead the lesson, or students can follow the step-by-step instructions to work through it independently.

Younger students require a fully guided approach, while older students often benefit from working at their own pace with teacher support as needed.

Flexible Curriculum Approach

Teachers have the flexibility to choose the modules that best fit their class needs. While there are enough lessons to cover a full school year, it is not necessary to complete all the modules. This allows teachers to tailor the learning experience to their students while ensuring they meet their educational goals.

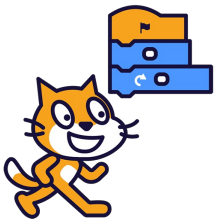
Student Access

Students log into the platform to access their lessons. They can follow the step-by-step instructions independently, or teachers can lead the lesson as needed.

Getting Started

1. **Review the Learning Plan:** Each module includes an overview of its goals, learning outcomes, lesson structure, and required resources. Start by familiarising yourself with the curriculum's scope.
2. **Plan Your Lessons:** Every lesson includes step-by-step guidance, accessible from your teacher dashboard. Adjust the pacing and delivery method based on your students' needs.
3. **Check Required Equipment:** Most lessons only require a laptop, Chromebook, or tablet. Some modules may include additional materials like microbits or LEDs. The required equipment is listed at the start of each module and each individual lesson.
4. **Support Student Learning:** Encourage students to work through the lessons. No prior coding experience is required—teachers can learn alongside their students.
5. **Use Assessments:** Each lesson includes a multiple-choice quiz to help assess student understanding and track progress.
6. **Need Help?:** We're always happy to answer your questions and give advice. You can contact our team at info@digitalskills.org or +44 020 4600 8710.

Module: Introduction to Coding



This module introduces students to the fundamentals of coding, starting with an overview of what coding is and its applications. Teachers should utilise visual aids and interactive discussions. The module progresses to hands-on experience with Scratch, a coding platform for creating games and animations. Teachers should familiarise themselves with Scratch and be prepared to assist students. The module culminates in students creating a Paddle Ball Game, reinforcing their understanding of moving sprites, changing backdrops, and using sensing blocks. Teachers should ensure students understand X and Y coordinates and Scratch coding blocks.

Duration	**Equipment**
2 weeks	Students can use any of these devices: • Chromebook/Laptop/PC • iPad/Tablet
Module Goals	**Module Outcomes**
1. Understand the concept of coding and its potential applications. 2. Gain proficiency in using Scratch for creating projects, including adding sprites and backdrops, and making sprites move. 3. Experiment with different code blocks in Scratch and learn from trial and error. 4. Create a Paddle Ball Game using Scratch, incorporating skills such as moving sprites, changing backdrops, and using sensing blocks. 5. Understand and apply the concepts of X and Y coordinates in the context of Scratch projects.	1. Understand the concept of coding and its applications. 2. Develop basic skills in Scratch, including creating projects, adding sprites and backdrops, and making sprites move. 3. Experiment with different code blocks in Scratch and learn from mistakes. 4. Create a Paddle Ball Game using Scratch, demonstrating the ability to move sprites, change backdrops, and use sensing blocks. 5. Understand and apply the concepts of X and Y coordinates in the context of Scratch coding.

Week 1

Lesson: Introduction to Coding

□ Beginner	□ 10 mins	□ Teacher/Student led	□ Student Quiz

If possible play the video in step 1 on a large screen for all your students to watch together. For steps 2 and 3 you should discuss and demonstrate these with your students.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
• Understand the concept of coding or programming as giving step-by-step instructions to a computer. • Identify examples of household items that contain computers and can be given instructions. • Recognize the importance of precise and correct order of instructions in coding. • Practice giving specific instructions in a sequential order to achieve a desired outcome.	• Define coding as the process of giving step-by-step instructions to a computer. • Identify at least three household items that contain computers and can be given instructions. • Explain the importance of precise and correct order of instructions in coding. • Demonstrate the ability to give specific instructions in the correct order to move from one point to another using a provided image.

Lesson: Scratch Tutorial

□ Beginner	□ 40 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

This lesson introduces students to Scratch, a coding platform for creating games and animations. Teachers should familiarise themselves with the Scratch website and its functionalities. The lesson guides students through creating a project, removing the default sprite, adding a new sprite, making it move, adjusting values, creating a loop, adding a backdrop, and encourages further exploration. Teachers should be prepared to assist with any technical difficulties and encourage experimentation.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
1. Understand and navigate the Scratch coding platform. 2. Manipulate sprites by adding, removing, and controlling their movements. 3. Apply basic coding concepts such as loops and event triggers. 4. Modify code blocks to alter sprite behaviour. 5. Explore and experiment with various Scratch functionalities to create unique projects.	1. Identify Scratch as a coding platform for creating games, animations and projects. 2. Navigate and utilise the Scratch website interface. 3. Remove default sprites and add new ones from the sprite library. 4. Implement basic coding blocks to manipulate sprite movement. 5. Modify values within code blocks to alter sprite behaviour. 6. Create a loop within the code to repeat specific actions. 7. Add a backdrop from the library to enhance the visual aspect of the project. 8. Explore and experiment with various code blocks to diversify sprite actions.

Week 2

Lesson: Paddle Ball Game

□ Beginner	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

Prepare to guide students through creating a Paddle Ball Game using Scratch. They'll learn to move sprites, change backdrops, and use sensing blocks. They'll create a new Scratch project, add a paddle and a football sprite, position the ball, make it bounce, control the paddle, make the ball bounce off the paddle, add a backdrop, add a game over line and program the game over. Ensure students understand X and Y coordinates, and how to use the Scratch coding blocks.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
1. Develop skills in using Scratch to create a simple game. 2. Understand and apply the concept of sprites and backdrops in Scratch. 3. Learn to control sprite movements using mouse input. 4. Implement game logic using conditional statements in Scratch. 5. Understand and apply the concept of X and Y coordinates to position sprites.	1. Manipulate sprites and backdrops in Scratch. 2. Utilise X and Y coordinates to position sprites. 3. Implement code to control sprite movement and interaction. 4. Use sensing blocks to detect sprite collision and mouse position. 5. Create a game over condition using colour detection.

Module: Digital Creation Lab

This module guides students through the exciting world of digital creation using Scratch. From creating a language translator to programming an autonomous car, students will gain a comprehensive understanding of coding principles. Each lesson is interactive and hands-on, fostering creativity and problem-solving skills. Teachers should familiarise themselves with Scratch and be prepared to guide students through each project, encouraging experimentation and reinforcing the importance of practice in mastering coding.

Duration	Equipment
8 weeks	Students can use any of these devices: • Chromebook/Laptop/PC • iPad/Tablet
Module Goals	**Module Outcomes**
1. Develop proficiency in using Scratch to create interactive projects, including games and language translators. 2. Understand and apply coding concepts such as variables, loops, and collision detection in Scratch projects. 3. Gain skills in designing and animating sprites, creating backdrops, and controlling sprite movements in Scratch. 4. Learn to manipulate tools and extensions in Scratch, such as the pen tool, Translate, and Text to Speech. 5. Enhance creativity, problem-solving, and critical thinking skills through hands-on coding challenges and game development.	1. Develop a language translator using Scratch, incorporating Translate and Text to Speech extensions, variables, and interactive elements. 2. Create an engaging 'Shark Swim' game using Scratch, mastering sprite control, animation, collision detection, and game loop establishment. 3. Program an autonomous car using Scratch, understanding autonomous vehicle operation, sprite manipulation, track design, and autonomous navigation. 4. Design unique patterns using Scratch, utilising the pen tool, variables, and pen colour and size manipulation. 5. Construct an interactive game using Scratch, controlling a coloured disc, cloning attacking dots, and detecting dot colours. 6. Develop a rocket landing game using Scratch, programming gravity, rocket movement, animations, and fuel limits. 7. Create a platformer game using Scratch, designing characters, creating platforms, controlling character movements, and adding effects. 8. Engage in build battles, demonstrating problem-solving skills and creativity in tackling code challenges.

Week 1

Lesson: Translate

□ Beginner	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

This lesson involves using Scratch to create a language translator. Students will learn how to add the 'Translate' and 'Text to Speech' extensions, create a new project, and add a sprite. They will also learn how to create variables, upload sprites, and write code to translate text into different languages and make the sprite speak the translation. The lesson concludes with the opportunity to add more languages and test the translator.

Learning Goals	Learning Outcomes
1. Understand how to use Scratch to translate text into different languages. 2. Develop skills in creating and managing a new project in Scratch. 3. Learn to use the Translate and Text to Speech extensions in Scratch. 4. Gain experience in creating and using variables in Scratch. 5. Apply knowledge to add more languages to the translation project.	1. Utilise Scratch to create a new project and add specific extensions. 2. Implement the Translate and Text to Speech extensions in a Scratch project. 3. Create and manipulate variables within Scratch to store language and translation data. 4. Use Scratch to translate text into different languages and vocalise the translation. 5. Add and code multiple language options to a Scratch project.

Week 2

Lesson: Shark Swim

□ Beginner	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

In this lesson, students will create a game using Scratch, where a diver navigates a course without touching the edges or encountering a shark. They will learn how to set up a new Scratch project, create a backdrop, add and position sprites, and write code to control sprite movements. They will also learn how to animate sprites using costumes, detect collisions, and create a simple game loop. The lesson concludes with students testing their game and reflecting on their learning.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
1. Develop skills in creating and managing a Scratch project. 2. Understand and apply the concept of sprite control and animation using costumes. 3. Gain proficiency in using coding blocks for game mechanics such as collision detection and game loop creation. 4. Learn to use the mouse pointer for sprite movement and control. 5. Develop an understanding of game development concepts and apply them in a practical project.	1. Develop a game using Scratch, incorporating elements such as sprites, backdrops, and costumes. 2. Control sprite movements using mouse pointer and code blocks. 3. Implement collision detection between sprites and specific colours. 4. Utilise costumes to create animation effects within the game. 5. Create a simple game loop, demonstrating understanding of game development basics.

Week 3

Lesson: Autonomous Car

□ Intermediate	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

Prepare to guide students through the process of understanding how autonomous cars work. Facilitate the creation of a Scratch project where students will program their own autonomous car, incorporating elements such as car sprites, speed variables, and sensor-driven navigation. Encourage students to experiment with different track designs and speeds, fostering a deeper understanding of autonomous vehicle technology.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
1. Understand the concept and workings of an autonomous car. 2. Develop skills in creating a new Scratch project and manipulating sprites. 3. Learn to use variables and conditional statements in Scratch to control sprite movements. 4. Apply knowledge of sensors in programming an autonomous car to navigate a track. 5. Enhance problem-solving skills by implementing speed control and reverse functions in the autonomous car project.	1. Understand the functioning of an autonomous car and its use of sensors for navigation. 2. Create a new Scratch project and manipulate sprites and backdrops. 3. Program the car to move and navigate using colour detection and conditional statements. 4. Control the speed of the car using variables and keyboard inputs. 5. Implement a reverse function to correct the car's course when it deviates from the track.

Week 4

Lesson: Pattern Creator

□ Intermediate	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

Prepare to guide students through an engaging exploration of pattern creation using Scratch. Familiarise yourself with the Scratch interface and pen tool, as well as the process of creating a new project and adding sprites. Be ready to explain the use of variables, loops, and how to manipulate pen colour and size. Encourage students to experiment with different degrees and pen sizes to create unique patterns. Wrap up by reinforcing the importance of practice and creativity in mastering coding.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
1. Master the use of Scratch for pattern creation. 2. Understand and apply the use of variables in creating complex shapes and patterns. 3. Manipulate the pen tool to draw and create unique patterns. 4. Experiment with different degrees and pen sizes to alter pattern outcomes. 5. Apply creativity in coding to produce vibrant and unique patterns.	1. Code a Scratch project to create basic patterns using the pen tool. 2. Implement the use of variables to manipulate pattern creation. 3. Adjust pen colour and size to enhance pattern design. 4. Utilise loops and conditional statements to control pattern formation. 5. Experiment with different variable values to create unique patterns.

Week 5

Lesson: Attack of the Dots

□ Intermediate	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

Prepare for an interactive lesson where students will create a game using Scratch. They will learn to control a coloured disc, clone attacking dots, and detect the colour of the dots. Ensure students understand how to remix a starter project, make the disc spin, clone the ball, prevent the ball from appearing too close to the disc, make the ball move, detect the colour of the ball, create purple and orange balls, and change the code for the purple and orange balls. Wrap up by congratulating students on their newly acquired skills.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
1. Develop skills in using Scratch to create an interactive game. 2. Understand how to control a coloured disc using keyboard inputs. 3. Learn to clone game elements and set their behaviour. 4. Master the technique of colour detection for game mechanics. 5. Apply problem-solving skills to prevent game elements from spawning too close to the player.	1. Master the use of Scratch to create an interactive game. 2. Control a coloured disc using keyboard inputs. 3. Clone and manipulate game elements, such as coloured dots, using Scratch code. 4. Implement colour detection to trigger game events. 5. Modify and customise game elements to enhance gameplay.

Week 6

Lesson: Rocket Lander

□ Advanced	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

Prepare to guide students through creating a rocket landing game using Scratch. The lesson involves programming gravity, controlling rocket movement, creating animations for rocket thrust and explosion, and adding a fuel limit for an extra challenge. Ensure students understand the concept of variables and conditions in coding. Encourage creativity and problem-solving as they experiment with their game.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
1. Understand the concept of vertical rocket landing and its challenges. 2. Develop a game using Scratch, simulating a rocket landing scenario. 3. Implement gravity and movement controls in the game using code blocks. 4. Create and use costumes to animate rocket thrust and explosion. 5. Introduce and manage a fuel limit for added complexity in the game.	1. Understand and explain the functionality of the Space X Falcon 9 rocket. 2. Create a basic game in Scratch, including setting up a starter project. 3. Program gravity and booster functions for a rocket sprite in Scratch. 4. Design and implement visual effects such as rocket thrust and explosion in Scratch. 5. Implement controls for rocket movement and landing, including fuel limits and landing conditions.

Week 7

Lesson: Scratch Platformer

□ Advanced	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

In this lesson, students will create a platformer game using Scratch. They will design characters, create platforms, and write code to control character movements. The lesson includes creating a new Scratch project, designing sprites, resizing characters, creating variables, applying gravity, enabling character movement and jumping, adding a trailing effect, adding more costumes to the ground sprite, detecting screen edges, receiving messages, and wrapping up. The lesson is hands-on and encourages creativity and problem-solving.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
1. Develop skills in creating and manipulating sprites in Scratch. 2. Understand and apply the concept of variables in game development. 3. Implement control mechanisms for character movement, including gravity and jumping. 4. Utilise broadcasting messages to manage game states and transitions. 5. Enhance game aesthetics through effects like character trailing.	1. Design and create sprites for a platformer game in Scratch. 2. Implement movement controls for a character sprite, including left, right, and jump actions. 3. Apply gravity effect to character sprite using Scratch coding blocks. 4. Create and utilise variables to control game mechanics such as speed, jump height, and gravity. 5. Develop multiple game levels by creating different platform configurations.

Week 8

Lesson: Build Battles

□ Advanced	□ 60 mins	□ Teacher led

Prepare to facilitate a series of build battles using Scratch. Start with an introduction, then guide students through three timed challenges: a 10-minute space-themed project, a 5-minute sports-themed project, and a 1-minute open-themed project. Ensure students understand the time limits and how to submit their projects. Be ready to manage the sharing and judging of projects.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
1. Develop proficiency in using Scratch for quick project creation. 2. Apply creative thinking to design and execute projects under time constraints. 3. Adapt to different themes and incorporate them into coding projects. 4. Improve presentation skills through project sharing and discussion. 5. Enhance competitive spirit and teamwork through build battles.	1. Create a Scratch project with a space theme within a 10-minute timeframe. 2. Present the created project to peers within a 2-minute timeframe. 3. Develop a Scratch project with a sports theme within a 5-minute timeframe. 4. Present the sports-themed project to peers within a 2-minute timeframe. 5. Construct a Scratch project with any theme within a 1-minute timeframe and present it to peers within a 2-minute timeframe.

Module: Advanced Game Development

This module guides students through advanced game development using MakeCode Arcade. Teachers should ensure students understand key concepts such as sprites, coordinates, and coding effects. Encourage creativity and problem-solving as students create a variety of games, including arcade, platform, and space-themed games. The module concludes with a group project, promoting teamwork and creativity. Remember to provide assistance as needed and facilitate constructive feedback during project presentations.

Duration	**Equipment**
8 weeks	Students can use any of these devices: • Chromebook/Laptop/PC • iPad/Tablet
Module Goals	**Module Outcomes**
1. Master the use of MakeCode Arcade for game development, including sprite creation, movement control, and game logic programming. 2. Develop proficiency in designing and implementing various game types such as arcade, platform, and space-themed games. 3. Understand and apply key game development concepts such as object generation, collision detection, scoring systems, and game rules. 4. Enhance problem-solving skills through coding challenges and game modifications. 5. Cultivate creativity and teamwork through collaborative game project brainstorming and development.	1. Create and control game sprites using MakeCode Arcade, including movement, interaction, and visual design. 2. Implement game mechanics such as scoring systems, timers, and life counters. 3. Design and draw game maps, including tile placement and sprite interaction. 4. Apply coding concepts to create interactive games, including object generation, overlap detection, and event handling. 5. Develop teamwork and creativity skills through group brainstorming and project creation.

Week 1

Lesson: First Arcade Project

□ Beginner	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

This lesson guides students through creating their first arcade project using MakeCode Arcade. They will learn about the code editor, how to create a new project, add a sprite, choose a sprite from the gallery, move the sprite, draw a tile map, draw walls, make the camera follow the sprite, add projectiles, set their direction and speed, detect overlap, lose a life, and finally, send the code to a handheld device. The lesson is hands-on and interactive, allowing students to learn by doing.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
1. Understand and utilise MakeCode Arcade for creating games. 2. Manipulate the Code Editor to build and modify game elements. 3. Create and customise sprites for use in a game. 4. Develop a tile map and implement walls for game navigation. 5. Implement game mechanics such as projectiles, sprite movement, and life count.	1. Understand the functions and features of MakeCode Arcade. 2. Use the MakeCode Arcade code editor to create a new project and add a sprite. 3. Manipulate the sprite's movements using the direction buttons in the simulator. 4. Create and edit a tile map, including drawing walls and setting the camera to follow the sprite. 5. Design and implement projectiles, including setting their direction and speed, and programming responses to overlaps with the player's sprite.

Week 2

Lesson: Monkey Mayhem

□ Beginner	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

Prepare to guide students through creating a game using MakeCode Arcade. They will learn to control a character, generate objects at random positions, and collect them for points. They will also add a countdown timer to make the game more challenging. Ensure students understand the concepts of sprites, coordinates, and coding effects. Encourage creativity and problem-solving as they modify the game or create a new one.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
1. Develop skills in creating and controlling a player sprite in MakeCode Arcade. 2. Understand how to generate food sprites at random positions on the game screen. 3. Learn to implement a scoring system based on sprite interaction. 4. Gain knowledge on adding sound effects to enhance game experience. 5. Master the use of a countdown timer to increase game difficulty.	1. Create and control a player sprite in MakeCode Arcade. 2. Generate food sprites at random positions on the game screen. 3. Collect food sprites for points and implement a scoring system. 4. Add sound effects to enhance game play experience. 5. Implement a countdown timer to increase game challenge.

Week 3

Lesson: Space Shooter

□ Beginner	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

Prepare to guide students through creating a space-themed game using MakeCode Arcade. They will design a spaceship sprite, control its movements, set the number of lives, create and program asteroids, fire rockets, destroy asteroids, and lose lives when hit by an asteroid. Ensure students understand the importance of correct code placement and sprite selection. Encourage them to test their game frequently to ensure it functions as expected.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
1. Develop understanding of MakeCode Arcade for game creation. 2. Gain proficiency in creating and controlling game sprites. 3. Learn to implement game mechanics such as scoring and lives. 4. Understand how to detect and respond to sprite interactions. 5. Apply coding skills to create a complete Space Shooter game.	1. Design and create a spaceship sprite in MakeCode Arcade. 2. Control the spaceship sprite using arrow keys and prevent it from going off the screen. 3. Set the number of lives for the spaceship. 4. Create and program asteroids to fly in from the right side of the screen. 5. Fire rockets from the spaceship when the A button is pressed.

Week 4

Lesson: Platform Place

□ Intermediate	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

Prepare to guide students through creating their first platform game using MakeCode Arcade. The lesson involves understanding the basics of platform games, creating a new project, designing a sprite, programming sprite movements, adding gravity, drawing a map with different elements, programming a jump function, testing the game, and adjusting the game's mechanics. Ensure students understand the code snippets and their purpose in the game's functionality. Encourage creativity in sprite and map design.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
1. Understand the basic concept and mechanics of platform games. 2. Create and design a sprite character in a game environment. 3. Implement movement controls for the sprite character. 4. Apply the concept of gravity in a game setting. 5. Design and create a game map with different elements such as ground, danger and goal tiles.	1. Understand the concept of platform games and their mechanics. 2. Create a new project on arcade.makecode.com and design a sprite character. 3. Implement sprite movement controls using code. 4. Apply the concept of gravity to a sprite in a platform game. 5. Design a game map with ground, danger, and goal tiles. 6. Program a sprite to jump and move through the map. 7. Implement game mechanics such as danger tiles and a goal tile.

Week 5

Lesson: Arcade Build Battles

□ Intermediate	□ 60 mins	□ Teacher led

Prepare to facilitate a series of build battles where students create coding projects within set time limits. Ensure students understand the time constraints and how to share their projects. The battles will vary in length and complexity, from a 15-minute arcade project, to a 5-minute themed project, and finally a 1-minute character design task.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
1. Develop and apply coding skills to create an Arcade project within a specified time limit. 2. Design and create a unique character in Arcade within a one-minute timeframe. 3. Enhance project management skills by adhering to strict time constraints during project development. 4. Improve communication skills by sharing and presenting created projects to peers. 5. Cultivate a competitive spirit and teamwork through participation in build battles.	1. Create an Arcade project within a 15-minute time frame. 2. Share the created project within a 2-minute time frame. 3. Develop an Arcade project with any theme within a 5-minute time frame. 4. Design a character in Arcade within a 1-minute time frame. 5. Share the designed character within a 2-minute time frame.

Week 6

Lesson: Galaxy Ghosts

□ Advanced	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

Prepare to guide students through creating a space-themed game using MakeCode Arcade. They will learn to create and control a player sprite, generate enemy sprites, and program interactions between them. The lesson includes creating a new project, coding the player and enemy sprites, setting their positions and movements, and programming the game's scoring system and health bar. Ensure students understand each step and encourage them to experiment with their games.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
1. Develop skills in using MakeCode Arcade to create a space-themed game. 2. Learn to create and control player and enemy sprites, and program interactions between them. 3. Understand how to implement a scoring system and a health bar in the game. 4. Gain knowledge on how to increase game difficulty by increasing enemy speed over time. 5. Develop problem-solving skills by modifying and improving the game.	1. Create and control a player sprite in MakeCode Arcade. 2. Generate enemy sprites and program interactions between them. 3. Implement a scoring system for each enemy sprite hit. 4. Use a health bar to track and display player's health status. 5. Program game over conditions based on player's health status.

Week 7

Lesson: Donut Rush

□ Advanced	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

In this lesson, students will create an interactive game called 'Donut Rush' using MakeCode Arcade. They will learn to write code for creating game sprites, handling events like sprite overlaps, and controlling game logic. The lesson involves setting up the game, creating a new project, and defining variables to track the game's state. Students will also learn to create a function, set up the level, create the donuts, and start the game. They will add code to detect when the player sprite overlaps with a donut sprite and to check if the player has collected the target number of donuts. The lesson concludes with a wrap-up and play session.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
1. Develop an understanding of game creation using MakeCode Arcade. 2. Learn to create and manage variables in a gaming context. 3. Understand the concept and application of functions in game development. 4. Gain skills in handling events such as sprite overlaps and controlling game logic. 5. Apply knowledge to create an interactive game with multiple levels and scoring system.	1. Create a new project in MakeCode Arcade. 2. Set up the game by creating a splash screen, setting up variables, and creating a player sprite. 3. Create a function called 'startLevel' to organise the game's code. 4. Set up the level by adding code to the 'startLevel' function, including setting the background colour, displaying a level message, setting the target number of donuts to collect, and starting a countdown. 5. Create multiple donuts using a loop and place them randomly on the screen. 6. Start the game by calling the 'startLevel' function. 7. Collect donuts by detecting when the player sprite overlaps with a donut sprite, increasing the score, destroying the donut sprite, and playing a smile effect. 8. Complete the level by checking if the player has collected the target number of donuts, increasing the level, playing a 'jump up' sound, and starting a new level. 9. Wrap up the game and play it, aiming to collect as many donuts as possible within the time limit.

Week 8

Lesson: Game Lab

□ Advanced	□ 60 mins	□ Teacher led

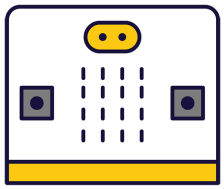
In this lesson, 'Brainstorming Blast', students will brainstorm ideas for their own MakeCode Arcade projects. Start by introducing the lesson and demonstrating a simple MakeCode Arcade project. Divide students into small groups for brainstorming, reminding them of the importance of teamwork. Set a timer for the brainstorming session and encourage students to keep their ideas simple and achievable. After brainstorming, each group will present their project idea and receive feedback from the class. Students will then create their projects in MakeCode Arcade, with the teacher providing assistance as needed. Finally, conduct a 'Show and Tell' session where each group presents their project to the class.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
1. Develop and articulate original ideas for a simple MakeCode Arcade project. 2. Collaborate effectively in small groups to brainstorm and refine project ideas. 3. Present project ideas clearly and constructively, incorporating feedback from peers and teachers. 4. Apply basic MakeCode Arcade blocks to create a simple game or interactive project. 5. Reflect on the process of project creation, identifying learning points and areas for improvement.	1. Brainstorm and develop a simple, achievable idea for a MakeCode Arcade project. 2. Collaborate effectively within a group to discuss and refine project ideas. 3. Present a project idea to the class, explaining the concept, sprites, and tile maps planned for use. 4. Constructively receive and incorporate feedback to improve the project plan. 5. Create a MakeCode Arcade project based on the brainstormed idea, demonstrating basic proficiency in using MakeCode Arcade blocks.

Module: Microbit Innovators



This module introduces students to the world of microbits, pocket-sized programmable computers. Teachers should prepare to guide students through creating projects, writing code, and programming their microbits to perform various tasks. The module encourages critical thinking and creativity, with lessons ranging from creating a 'Bop It' game to a Seismic and Meteorological Station. Teachers should ensure students understand the coding concepts and provide assistance as needed.

Duration	Equipment
8 weeks	Students can use any of these devices: • Chromebook/Laptop/PC Required Equipment: • Microbit
Module Goals	**Module Outcomes**
1. Master the basics of microbits, including creating projects, writing and deleting code, and programming microbits to perform various tasks. 2. Understand and utilise the different sensors in microbits, and interpret their responses to various inputs. 3. Develop the ability to create interactive games using microbits, incorporating elements such as time constraints and randomised actions. 4. Design and implement a microbit voting system, demonstrating an understanding of security considerations in system design. 5. Apply knowledge of microbits to control external applications, such as a Scratch Paddle Ball game. 6. Create a multi-device IoT network using microbits, programming each device to perform specific functions and communicate wirelessly with others. 7. Develop a multiplayer game using microbits, demonstrating an understanding of proximity-based interactions and health value mechanics.	1. Programme a microbit to display messages, react to button presses, show icons, play melodies, and respond to movement. 2. Investigate and utilise the different sensors of a microbit to respond to various inputs. 3. Develop a microbit game that requires precise timing and understanding of the device's capabilities. 4. Create a multi-player game using microbit, incorporating random image display and corresponding actions. 5. Design and implement a microbit voting system, including casting votes, tallying, and resetting the system. 6. Control a Scratch Paddle Ball game using a microbit by tilting it to the left and right. 7. Construct a multi-device IoT network using microbits to monitor and display temperature, light levels, and seismic activity. 8. Develop a multiplayer game that simulates a zombie virus spreading among microbits, requiring multiple people and devices.

Week 1

Lesson: Exploring Microbits

□ Beginner	□ 60 mins	□ Teacher/Student led	□ Student Quiz

Prepare to introduce students to the world of microbits, a pocket-sized programmable computer. The lesson will involve creating a new project on the MakeCode for microbit website, familiarising with the project editor, and writing code to display numbers, names, and icons. Students will also learn to delete code, connect their microbits to their computers, and program their microbits to play music. The lesson concludes with an exploration phase where students can experiment with different blocks from the toolbox.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit

Learning Goals	Learning Outcomes
1. Understand the basic functionality and features of a microbit. 2. Create a new project using the MakeCode for microbit website. 3. Use the Project Editor to write and simulate code. 4. Program the microbit to display numbers and text on its LED grid. 5. Program the microbit to respond to button presses with specific actions.	1. Identify the functions and capabilities of a microbit. 2. Create a new project on the MakeCode for microbit website. 3. Understand the layout and functions of the Project Editor. 4. Write and execute code to display numbers and names on the microbit. 5. Program the microbit to respond to button presses with specific displays. 6. Connect and download code to an actual microbit device. 7. Compose and program a melody to play on the microbit. 8. Explore and experiment with different coding blocks and functions.

Week 2

Lesson: Microbit Sensor Graphs

□ Beginner	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

For the 'Microbit Sensor Graphs' lesson, teachers should familiarise themselves with the makecode.microbit.org website and how to create a new project. They should understand how the light level sensor works and how to display and graph the light level. Teachers should also explore how to graph other sensors such as temperature, compass heading, acceleration, magnetic forces, and sound level. Lastly, they should encourage students to get creative with their graphs and consider how these sensors could be used in other projects.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit

Learning Goals	Learning Outcomes
1. Understand how to create a new project on makecode.microbit.org. 2. Learn to display the light level on the Microbit. 3. Develop skills to graph the light level and observe changes. 4. Gain knowledge on graphing other sensors such as temperature, compass heading, acceleration, magnetic forces and sound level. 5. Apply creativity to graph other possible values and incorporate sensor usage in different projects.	1. Create a new project on makecode.microbit.org. 2. Display the light level on the Microbit. 3. Graph the light level changes on the Microbit. 4. Graph the readings from other sensors on the Microbit, including temperature, compass heading, acceleration, magnetic forces, and sound level. 5. Design and implement a creative application using the Microbit's sensors.

Week 3

Lesson: Exactly 11

□ Intermediate	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

This lesson involves creating a Microbit project where students guess when 11 seconds have passed. They will learn to create and use variables 'starttime', 'taken', and 'difference'. They will also learn to use the 'running time (ms)' block, calculate absolute values, and use conditional statements to display results. The lesson involves coding and understanding the concept of milliseconds.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit

Learning Goals	Learning Outcomes
1. Develop skills in creating and utilising variables in Microbit projects. 2. Understand and apply the concept of running time in programming. 3. Gain proficiency in using mathematical operations to calculate time differences. 4. Learn to use conditional statements to display different outcomes based on user input. 5. Enhance problem-solving skills through the creation of a time-guessing game.	1. Develop a new Microbit project. 2. Create and utilise 'starttime' variable to record the start time. 3. Establish a 'taken' variable to store the time elapsed between two actions. 4. Formulate a 'difference' variable to calculate the difference between 11 seconds and the 'taken' time. 5. Implement code to display the result and provide feedback to the user.

Week 4

Lesson: Microbit Bop It Game

□ Intermediate	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

In this lesson, students will create a 'Bop It' game on Microbit. They will learn to create a new project, set up variables, and use countdown blocks. The game will involve displaying random images on the Microbit's LED display, with students having to perform the correct action corresponding to each image. They will also learn to program the buttons for different actions and handle game over scenarios. A challenge is included to add an extra action and image.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit

Learning Goals	Learning Outcomes
1. Understand and apply the concept of variables in coding. 2. Develop skills in creating and manipulating countdowns in a game context. 3. Implement conditional statements to control game actions. 4. Programme Microbit buttons to interact with the game. 5. Enhance problem-solving skills by adding new features to the game.	1. Create a new Microbit project for the 'Bop It' game. 2. Define and initialise a variable 'action' to store and remember a random action. 3. Implement a start countdown block set to 20000 milliseconds (20 seconds). 4. Assign a random number between 0 and 2 to the 'action' variable. 5. Display an image on the Microbit's LED display based on the value of 'action'. 6. Program the Microbit buttons to respond correctly to the displayed image. 7. Implement a game over check when the countdown has elapsed. 8. Extend the game by adding an additional action and corresponding image.

Week 5

Lesson: Microbit Voting System

□ Intermediate	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

Prepare to guide students in creating a microbit voting system. They'll create two projects on the MakeCode Microbit website, one for voting microbits and another for a central microbit. They'll program the A and B buttons to cast votes, set up the central microbit to receive votes and reset the system, and display the vote results. They'll also enhance the system with a security feature. Ensure students understand the coding involved and the importance of testing their system.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit

Learning Goals	Learning Outcomes
1. Develop a microbit voting system with separate voting and central microbits. 2. Programme the voting microbits to cast a single 'Yes' or 'No' vote. 3. Configure the central microbit to receive votes, count them, and reset the voting system. 4. Implement a reset function on the voting microbits to allow for multiple rounds of voting. 5. Enhance the voting system by adding a security feature to ensure the integrity of the votes.	1. Develop two separate projects on the MakeCode Microbit website for voting and central microbits. 2. Programme the A and B buttons on the microbit to cast a single 'Yes' or 'No' vote. 3. Set up the central microbit to receive votes, count them, and reset the voting system when needed. 4. Configure the individual voting microbits to receive the 'Reset' signal from the central microbit. 5. Display the vote results on the central microbit.

Week 6

Lesson: Microbit Paddle Ball

□ Intermediate	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

In this lesson, students will create a Microbit Paddle Ball game using Scratch. They will learn to create a new project, add and position sprites, and make the ball bounce around the screen. They will also connect a Microbit to control the paddle, make the ball bounce off the paddle, add a backdrop, and create a game over line. The lesson concludes with programming the game over functionality and discussing potential improvements to the game. Teachers should familiarise themselves with Scratch and Microbit prior to the lesson.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit

Learning Goals	Learning Outcomes
1. Develop skills in creating and managing a new Scratch project. 2. Understand and apply the concept of adding and positioning sprites in Scratch. 3. Gain proficiency in coding for sprite movement, interaction, and control using Scratch blocks. 4. Learn to integrate and use a Microbit with Scratch for real-time control of sprites. 5. Enhance critical thinking and problem-solving skills by identifying potential improvements to the game.	1. Develop a new Scratch project and remove the default cat sprite. 2. Add and position the 'Paddle' sprite from the sprite library. 3. Add the 'Soccer Ball' sprite from the sprite library. 4. Set the X and Y coordinates to position the ball at the top center of the screen. 5. Code the ball to move around the screen and bounce off the edges. 6. Connect and configure a Microbit to the Scratch project. 7. Code the paddle to move left and right by tilting the Microbit. 8. Program the ball to bounce off the paddle when it touches it. 9. Add the 'Stars' backdrop from the backdrop library. 10. Draw a red line at the bottom of the screen for the game over line. 11. Code the game to end when the ball touches the red game over line. 12. Propose improvements to the game by adding to or changing the existing code.

Week 7

Lesson: Microbit Zombies

□ Advanced	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

This lesson involves creating a multiplayer game using microbits. Students will learn to create and initialise variables, set radio groups for communication between devices, and use conditional statements to control game logic. The game involves a health system and an infection mechanic, where proximity to an infected microbit reduces health until the player becomes infected. The lesson also introduces the concept of signal strength as a measure of proximity. The game starts with a random infection event and continues until all microbits are infected.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit

Learning Goals	Learning Outcomes
1. Develop a basic understanding of microbit project creation and the use of variables. 2. Understand and implement the use of radio groups in multiplayer games. 3. Learn to display different icons and messages based on variable conditions. 4. Understand and apply the concept of signal strength in determining proximity in the game. 5. Develop skills in creating interactive multiplayer games using randomisation and conditional statements.	1. Construct a new microbit project and understand the concept of health and infection in the game context. 2. Create and utilise two variables 'health' and 'infected' to track game status. 3. Set up a radio group for multiplayer interaction and understand its functionality. 4. Display infection status and health value on the microbit using code. 5. Program the microbit to lose health upon receiving a "zombie" message from an infected microbit. 6. Implement a condition to change the status to infected when health reaches zero. 7. Randomly infect a microbit at the start of the game using a random number generator. 8. Download the project, transfer the .HEX file onto the microbit, and play the game.

Week 8

Lesson: Microbit Lab

□ Advanced	□ 60 mins	□ Teacher led

Prepare to introduce the concept of Microbit projects, demonstrating a simple LED pattern to inspire creativity. Organise students into small groups for brainstorming, emphasising teamwork and achievable project ideas. Facilitate a feedback session after idea presentations, guiding project simplification if necessary. Assist during project creation, encouraging peer support and discovery sharing. Finally, conduct a 'Show and Tell' session, celebrating student effort and creativity, reinforcing learning objectives and the importance of teamwork.

Students can use any of these devices (and can share if necessary):

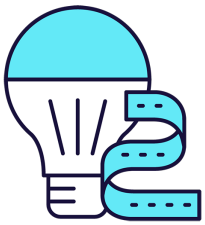
- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit

Learning Goals	Learning Outcomes
1. Develop creative and achievable project ideas using basic Microbit blocks. 2. Collaborate effectively in small groups to brainstorm, plan and execute a Microbit project. 3. Present project ideas clearly and receive feedback constructively. 4. Apply problem-solving skills to create a Microbit project based on the brainstormed idea. 5. Reflect on the project creation process, discussing changes made, challenges faced, and skills learned.	1. Brainstorm and develop a simple Microbit project idea in a group setting. 2. Present the project idea to the class, explaining the planned LED patterns and inputs. 3. Receive, incorporate, and respond to feedback on the project idea. 4. Create a Microbit project based on the brainstormed idea, using basic Microbit blocks. 5. Present the final Microbit project to the class, explaining the coding process and any changes made during the project creation.

Module: Exploring Electronics and Light



This module explores the exciting world of electronics and light, using Microbit and LED strips. Teachers will guide students through creating colourful displays, sound-activated lights, visual thermometers, and even a precision game. The module encourages creativity, problem-solving, and teamwork, with students brainstorming and implementing their own Microbit projects. Teachers should ensure students understand each step and concept before progressing, and provide assistance during the project creation stages.

Duration	Equipment
8 weeks	Students can use any of these devices: • Chromebook/Laptop/PC Required Equipment: • LED Strip with crocodile clips • Microbit
Module Goals	**Module Outcomes**
1. Understand and apply the principles of programming LED strips using Microbit projects. 2. Develop skills to create interactive LED displays that respond to sound and temperature changes. 3. Design and implement a game using LED strip and Microbit programming. 4. Enhance creativity and problem-solving skills through the design of LED flags and stacking effects. 5. Apply teamwork and project management skills in brainstorming and executing a group Microbit project.	1. Programme a strip of LEDs to display colourful patterns using Microbit. 2. Design and implement an LED Strip Clapper that responds to sound, specifically a clap, to turn on and off. 3. Convert an LED strip into a visual thermometer that lights up and changes colour according to the current temperature. 4. Create a voice-activated 'Shooting Stars' display using an LED strip and Microbit. 5. Design and code tricolour flags using LED strips. 6. Create a stacking effect on an LED strip, controlled by Microbit, with the ability to increase and decrease the size of the stack. 7. Develop an LED Strip Precision Game that involves timing and accuracy. 8. Brainstorm, design, and implement a simple Microbit project in a team, demonstrating creativity and teamwork.

Week 1

Lesson: Microbit LED Strip

□ Beginner	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

Prepare to guide students through programming a 30 LED strip using Microbits. Ensure understanding of creating a new Microbit project and adding the neopixel extension. Facilitate the setup of the LED strip and programming it to turn red. Assist with downloading the project onto the Microbit. Encourage creativity when programming the strip to show a rainbow of colours and rotating the rainbow. Finally, encourage exploration of other code blocks in the Neopixel toolbox.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- LED Strip with crocodile clips

Learning Goals	Learning Outcomes
1. Understand and apply the process of programming a strip of LEDs using Microbits. 2. Develop skills in creating a new Microbit project and adding the necessary extensions. 3. Gain proficiency in setting up and programming the LED strip to display various colours. 4. Learn to download and implement the project on Microbits, observing the effects on the LED strip. 5. Explore and experiment with different code blocks in the Neopixel toolbox for creative lighting effects.	1. Program a strip of 30 LEDs to light up in different ways using Microbits. 2. Create a new Microbit project and add the neopixel extension. 3. Set up the LED strip and interact with it using a variable. 4. Program the A button on the Microbit to turn all the LEDs red. 5. Program the LED strip to show a rainbow of colours when the Microbit turns on.

Week 2

Lesson: LED Strip Clapper

□ Beginner	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

In this lesson, students will create an LED Strip Clapper using a Microbit project. They will add the neopixel extension, set up the LED strip, and create an 'on' variable. The lesson will guide them to detect a clap, turning the LED strip on and off accordingly. They will download their code onto their microbit, connect it to the LED strip, and explore further improvements. Familiarity with Microbit and basic coding is beneficial.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- LED Strip with crocodile clips

Learning Goals	Learning Outcomes
1. Develop skills in creating and managing a new Microbit project. 2. Understand and apply the neopixel extension for programming an LED strip. 3. Learn to set up and interact with the LED strip using variables. 4. Gain knowledge on creating and manipulating variables to control the state of the LED strip. 5. Develop the ability to detect sound inputs and use them to trigger changes in the LED strip's state.	1. Develop a new Microbit project using makecode.microbit.org. 2. Integrate the neopixel extension into the project for LED strip programming. 3. Establish a variable for the LED strip and set its value to 30. 4. Create an 'on' variable to control the LED strip's state. 5. Implement a sound detection feature to trigger the LED strip's state change.

Week 3

Lesson: Microbit LED Strip Thermometer

□ Beginner	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

Prepare for this lesson by familiarising yourself with the Microbit project platform and the neopixel extension. Understand how to set up the LED strip and how to program the A button to display temperature. Be ready to guide students in lighting up the LED lights according to temperature readings and downloading their projects onto their Microbits. Ensure you know how to correctly connect the LED strip to the Microbit.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- LED Strip with crocodile clips

Learning Goals	Learning Outcomes
1. Develop skills to create and manage a new Microbit project. 2. Understand and apply the neopixel extension to program the LED strip. 3. Gain knowledge on setting up the LED strip and displaying temperature on the Microbit screen. 4. Learn to light up the LED lights on the strip according to the temperature readings. 5. Acquire practical skills in downloading the project, connecting the LED strip to the Microbit, and testing the functionality.	1. Create a new Microbit project using makecode.microbit.org. 2. Add the neopixel extension to the project for LED strip programming. 3. Set up the LED strip in the project with a value of 30, representing the 30 LEDs on the strip. 4. Program the A button to display the temperature on the Microbit screen. 5. Display the temperature by lighting up the LED lights on the strip, with the number of lights corresponding to the temperature reading. 6. Download the project and transfer it to the Microbit. 7. Connect the LED strip to the Microbit using the specified pin connections and power it using a USB cable.

Week 4

Lesson: Shooting Stars

□ Intermediate	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

Prepare to guide students in creating a Microbit project, adding the neopixel extension, and setting up the LED strip. Facilitate the creation of a 'star' that lights up with a loud sound, and ensure students can test this on their LED strip. Assist students in making the 'star' shoot along the strip and adding random colours. Finally, ensure students can download and test their code, encouraging them to create multiple shooting stars.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- LED Strip with crocodile clips

Learning Goals	Learning Outcomes
1. Develop skills in creating and managing a new Microbit project. 2. Understand and apply the neopixel extension to program the LED strip. 3. Gain proficiency in setting up and programming the LED strip using code blocks. 4. Learn to utilise the microphone in the microbit to detect sound and trigger LED actions. 5. Acquire knowledge on how to test and debug the project on the LED strip. 6. Master the concept of pixel shifting to create the illusion of moving light. 7. Experiment with random colour generation for the LED strip. 8. Learn to download and implement the code onto the microbit for real-world testing.	1. Create and manage a new Microbit project. 2. Integrate the neopixel extension into the project. 3. Set up and programme the LED strip using the provided code. 4. Develop a function to light up the first LED on the strip white when a loud sound is detected. 5. Test the function on the LED strip and ensure it works as expected. 6. Implement a function to make the 'star' shoot along the strip. 7. Enhance the function to display stars in random colours. 8. Download and test the final code on the microbit, ensuring different colour 'stars' shoot along the strip when a loud noise is made.

Week 5

Lesson: LED Flags

□ Intermediate	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

Prepare to guide students in creating LED flags using a Microbit project. They will need to understand how to add the neopixel extension and set up the LED strip. Facilitate as they create bicolor and tricolor flags, using the example of Malta and Ireland respectively. Encourage creativity and problem-solving skills for the challenge of representing the American flag.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- LED Strip with crocodile clips

Learning Goals	Learning Outcomes
1. Understand and apply the concept of bicolor and tricolor flags using LED strips. 2. Create and manage a new Microbit project effectively. 3. Utilise the neopixel extension to program the LED strip. 4. Develop skills to set up and interact with the LED strip using code. 5. Apply coding skills to create complex patterns, such as the American flag, on the LED strip.	1. Construct bicolor and tricolor flags using LED strips. 2. Utilise the neopixel extension to program the LED strip. 3. Set up and interact with the LED strip using a variable. 4. Apply the concept of ranges to light up specific sections of the LED strip. 5. Code the LED strip to represent complex flag designs, such as the American flag.

Week 6

Lesson: LED Stacking

□ Advanced	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

Prepare for this lesson by familiarising yourself with the Microbit project platform and the neopixel extension. Understand how to set up an LED strip and create variables to store the strip and the amount of LEDs. Be ready to guide students in creating a function to show the LED stack, and programming buttons to increase and decrease the stack. Ensure students know how to download their code and connect their LED strip to their microbit.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- LED Strip with crocodile clips

Learning Goals	Learning Outcomes
1. Develop skills in creating and managing a new Microbit project. 2. Understand and apply the neopixel extension for LED programming. 3. Learn to set up and interact with the LED strip using variables. 4. Develop competency in creating and using functions to control LED display. 5. Gain experience in programming button controls to manipulate LED stack size.	1. Create a new Microbit project using makecode.microbit.org. 2. Add the neopixel extension to the project for LED strip programming. 3. Set up the LED strip with a variable storing the strip, set to a value of 30. 4. Create an 'amount' variable to store the number of LEDs in the stack. 5. Develop a 'showStack' function to display the stack of lit LEDs. 6. Create a range of LEDs on the strip to light up, using the 'amount' variable, and call the 'showStack' function from the 'on start' block. 7. Program button A to increase the LED stack by adding 1 to the 'amount' variable and calling the 'showStack' function. 8. Program button B to decrease the LED stack by subtracting 1 from the 'amount' variable and calling the 'showStack' function. 9. Download the code onto a microbit, connect the LED strip using crocodile clips, and test the LED stack's increase and decrease functions with buttons A and B.

Week 7

Lesson: LED Strip Precision Game

□ Advanced	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

Prepare to guide students through creating an interactive LED strip game using a Microbit project. Familiarise yourself with the neopixel extension and the process of setting up the LED strip. Understand the purpose of the four variables: 'target', 'position', 'delay', and 'increment'. Be ready to explain how to set up the level, create a refresh function, and make the blue light move. Prepare to guide students through the steps of going back to the start, hitting the target, and handling a missed target. Finally, ensure you can assist students in downloading their code and playing the game.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- LED Strip with crocodile clips

Learning Goals	Learning Outcomes
1. Understand and apply the concept of LED strip programming using Microbit. 2. Develop skills in creating and manipulating variables in a coding project. 3. Learn to create and use functions for specific tasks within a coding project. 4. Gain proficiency in using conditional statements to control game outcomes. 5. Develop the ability to download and test code on a physical device.	1. Program an LED strip to light up specific LEDs in response to user input. 2. Create and manipulate variables to control game mechanics in a Microbit project. 3. Implement the neopixel extension to interact with an LED strip. 4. Design a function to refresh LED lights based on variable values. 5. Download and test the code on a physical Microbit device.

Week 8

Lesson: Microbit Lab

□ Advanced	□ 60 mins	□ Teacher led

Prepare to introduce the concept of Microbit projects, demonstrating a simple LED pattern to inspire creativity. Organise students into small groups for brainstorming, emphasising teamwork and achievable project ideas. Facilitate a feedback session after idea presentations, guiding project simplification if necessary. Assist during project creation, encouraging peer support and discovery sharing. Finally, conduct a 'Show and Tell' session, celebrating student effort and creativity, reinforcing learning objectives and the importance of teamwork.

Students can use any of these devices (and can share if necessary):

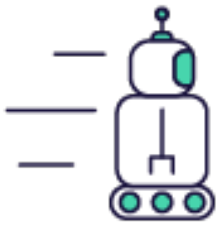
- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit

Learning Goals	Learning Outcomes
1. Develop creative and achievable project ideas using basic Microbit blocks. 2. Collaborate effectively in small groups to brainstorm, plan and execute a Microbit project. 3. Present project ideas clearly and receive feedback constructively. 4. Apply problem-solving skills to create a Microbit project based on the brainstormed idea. 5. Reflect on the project creation process, discussing changes made, challenges faced, and skills learned.	1. Brainstorm and develop a simple Microbit project idea in a group setting. 2. Present the project idea to the class, explaining the planned LED patterns and inputs. 3. Receive, incorporate, and respond to feedback on the project idea. 4. Create a Microbit project based on the brainstormed idea, using basic Microbit blocks. 5. Present the final Microbit project to the class, explaining the coding process and any changes made during the project creation.

Module: Designing and Building for the Future



This module guides students through the process of designing and building future technologies, starting with assembling and programming traffic lights using a Microbit Traffic Lights Kit. Students will then create a traffic light reaction game, a pedestrian crossing simulation, and a Move Motor Sensor Car. They will learn to program the car to follow a line track, use ultrasonic sensors, and be controlled by a Microbit remote. The module concludes with a lesson on coding a set of traffic lights and a robot car to communicate. Teachers should ensure they are familiar with the MakeCode editor, Microbit, and the various extensions used throughout The module.

Duration	**Equipment**
8 weeks	Students can use any of these devices: • Chromebook/Laptop/PC • Microbit Required Equipment: • Microbit • Move Motor Car • Phillips Screwdriver • Traffic Lights Kit
Module Goals	**Module Outcomes**
1. Master the assembly and programming of Microbit Traffic Lights. 2. Develop skills in creating interactive games using Microbit and STOP:bit Traffic Lights. 3. Understand and apply the principles of pedestrian crossing simulations using MakeCode editor and micro:bit. 4. Gain proficiency in building and programming a Move Motor Sensor Car. 5. Learn to code a car to follow a line track and use ultrasonic sensors for object detection and avoidance.	1. Assemble and operate a Microbit Traffic Lights Kit. 2. Program a sequence of traffic lights using on/off and state methods. 3. Create a traffic light reaction game, incorporating variables and reaction time measurements. 4. Construct a pedestrian crossing simulation, incorporating button press detection and traffic light sequencing. 5. Assemble and code a Move Motor Sensor Car, exploring its various sensors and capabilities. 6. Program a Move Motor Car to follow a line track, adjusting code for optimal performance. 7. Utilise ultrasonic sensors to enable a Move Motor Car to follow an object and avoid obstacles. 8. Control a Move Motor Car using a Microbit as a remote controller, based on tilt detection. 9. Code a set of traffic lights and a robot car to communicate and respond to each other's states.

Week 1

Lesson: Build your Traffic Lights

□ Beginner	□ 10 mins	□ Teacher/Student led	□ Student Quiz

Ensure students have all necessary materials, including the Microbit Traffic Lights Kit, a Microbit, and a Phillips head screwdriver. Guide them through opening the package and assembling the stand. Assist them in correctly positioning the Microbit on the traffic lights, ensuring they align the holes correctly. Supervise as they use the screwdriver to secure the Microbit. Celebrate their accomplishment once completed.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- Traffic Lights Kit
- Phillips Screwdriver

Learning Goals	Learning Outcomes
1. Identify and gather necessary components for the Microbit Traffic Lights Kit. 2. Understand and execute the process of unpacking and preparing the kit. 3. Develop skills in assembling the stand for the traffic lights. 4. Apply knowledge of Microbit to correctly align and attach it to the traffic lights. 5. Demonstrate the ability to follow step-by-step instructions to complete a technical task.	1. Identify and gather necessary components for the Microbit Traffic Lights Kit. 2. Unpack and organise the Microbit Traffic Lights package contents. 3. Assemble the stand from the provided parts in the kit. 4. Align and attach the Microbit to the traffic lights using the correct hole configuration. 5. Successfully complete the assembly of the Microbit Traffic Lights.

Lesson: Microbit Traffic Lights

□ Beginner	□ 40 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

In this lesson, students will create a new Microbit project on makecode.com, add the Stopbit extension, and test all the lights. They will learn about sequences in coding and apply this knowledge to program a traffic light sequence using on/off and state methods. Students will need to check the correct display of lights in each sequence.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- Traffic Lights Kit

Learning Goals	Learning Outcomes
1. Understand and apply the process of creating a new Microbit project. 2. Learn to add and utilise the Stopbit extension for programming traffic lights kit. 3. Gain skills in testing and troubleshooting the functionality of the lights. 4. Comprehend the concept of 'sequence' in coding and apply it to program traffic lights. 5. Develop proficiency in programming the sequence of traffic lights using on/off and state methods.	1. Create and manage a new Microbit project on makecode.com. 2. Add and utilise the "stopbit" extension to the Microbit project. 3. Test and troubleshoot the functionality of each light on the Microbit. 4. Understand and apply the concept of 'sequence' in coding to program traffic lights. 5. Program the sequence of traffic lights using on/off and state methods.

Week 2

Lesson: Traffic Light Reaction Game

□ Beginner	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

Prepare to guide students in creating a traffic light reaction game using a micro:bit and STOP:bit Traffic Lights. Ensure students understand how to attach the traffic lights to the micro:bit, create a new project on MakeCode, and add the "stopbit" extension. Explain the purpose of the 'startTime', 'endTime', and 'reactionTime' variables. Walk them through the process of setting up the code for button A and B presses, displaying the reaction time, and testing the game. Encourage students to challenge their peers and improve their reaction times.

Students can use any of these devices (and can share if necessary):

- Microbit

Required equipment for this lesson:

- Microbit
- Traffic Lights Kit
- Phillips Screwdriver

Learning Goals	Learning Outcomes
1. Develop understanding of micro:bit and STOP:bit Traffic Lights for creating a reaction game. 2. Learn to add and utilise the "stopbit" extension in the MakeCode toolbox. 3. Understand and apply the concept of variables to measure reaction time. 4. Develop skills to program micro:bit buttons for specific actions. 5. Learn to display data on the micro:bit's LED matrix and interpret the results.	1. Develop a new project using MakeCode for micro:bit. 2. Add the 'stopbit' extension to the toolbox for programming the traffic lights kit. 3. Create and utilise variables 'startTime', 'endTime', and 'reactionTime' to measure reaction time. 4. Program button A to initiate the traffic light sequence and record the start time. 5. Program button B to record the end time and calculate the reaction time.

Week 3

Lesson: Pedestrian Crossing

□ Beginner	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

Prepare for this interactive lesson by familiarising yourself with the MakeCode editor for micro:bit and the STOP:bit traffic lights. Understand how to add the 'stopbit' extension and create a 'seconds' variable. Be ready to guide students through coding a traffic light sequence, including red light display, button press detection, and the full traffic light sequence. Ensure you know how to download the code to a micro:bit for testing.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- Traffic Lights Kit
- Phillips Screwdriver

Learning Goals	Learning Outcomes
1. Understand and apply the use of MakeCode editor for micro:bit in creating a new project. 2. Learn to add and utilise the "stopbit" extension for programming the STOP:bit traffic lights kit. 3. Develop skills in creating and manipulating variables, specifically the 'seconds' variable in this context. 4. Gain proficiency in coding for specific outcomes such as displaying the red light and detecting button press on the micro:bit. 5. Master the sequence of traffic lights and their corresponding symbols on the micro:bit, and successfully download and test the code.	1. Develop a new project and attach STOP:bit traffic lights to the micro:bit. 2. Add the "stopbit" extension to the MakeCode editor toolbox. 3. Create a variable named 'seconds' for the countdown function. 4. Program the micro:bit to display a red light and an 'X' symbol at the start. 5. Implement button press detection to simulate a pedestrian waiting to cross. 6. Code a traffic light sequence with green, yellow, and red lights, along with appropriate wait times. 7. Download and test the code on the micro:bit.

Week 4

Lesson: Build your Move Motor Sensor Car

□ Intermediate	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

Ensure all materials are ready, including the Microbit and 4 AA batteries. Guide students through the step-by-step instructions provided in the yellow booklet, ensuring they understand each stage of assembly, connection to Makecode, and adding the Move Motor Extension. Facilitate their understanding of coding the motors, using the buzzer, Zip LEDs, line following sensors, and the distance sensor. Encourage exploration and experimentation once the Move Motor Sensor Car is built.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- Move Motor Car

Learning Goals	Learning Outcomes
1. Develop practical skills in assembling a Move Motor Sensor Car. 2. Understand how to connect the Move Motor Sensor Car to Makecode. 3. Acquire coding skills for controlling the motors, buzzer, and LEDs of the Move Motor Sensor Car. 4. Learn to utilise the line following and distance sensors for navigation. 5. Encourage exploration and creativity in coding for different movements and LED usage.	1. Identify and organise components of the Move Motor Sensor Car kit. 2. Assemble the Move Motor Sensor Car following the provided instructions. 3. Connect the assembled car to Makecode and add the Move Motor Extension. 4. Code the motors, buzzer, Zip LEDs, line following sensors, and distance sensor of the car. 5. Apply learned skills to explore and create new movements and LED patterns.

Week 5

Lesson: Line Following Car

□ Intermediate	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

In this lesson, students will program a Move Motor Car to follow a line track using a Microbit. They will create a new project on the MakeCode website, add the kitronik-move-motor extension, and create variables for the left and right line sensors and their difference. Students will then program the car to turn right, left, and move forward based on these sensor readings. After testing their code on a track, students can tweak the code to improve the car's speed and performance on more complex tracks.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- Move Motor Car

Learning Goals	Learning Outcomes
1. Understand and apply the concept of programming a Microbit to control a Move Motor Car. 2. Create and manipulate variables to store sensor values and control the car's movements. 3. Implement conditional logic to guide the car's movements based on sensor readings. 4. Use LEDs for visual feedback and enhance the functionality of the car. 5. Experiment with code modifications to optimise the car's performance on different tracks.	1. Programme the Move Motor Car to follow a line track using a Microbit. 2. Create a new project on the https://makecode.microbit.org website. 3. Add the kitronik-move-motor extension to the project and utilise the custom blocks to program the Move Motor car. 4. Create and utilise variables to store values of the left and right line sensors and their difference. 5. Set up the LEDs on the Move Motor car to light up different colours depending on the car's direction. 6. Programme the car to turn right when the left sensor reads a higher darker value than the right sensor. 7. Programme the car to turn left when the right sensor reads a higher darker value than the left sensor. 8. Programme the car to move forwards when the left and right sensors have similar readings. 9. Test the programmed car on a track and observe its autonomous driving. 10. Tweak the code to improve the car's speed and performance on different tracks.

Week 6

Lesson: Car Distance Sensors

□ Intermediate	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

Prepare to introduce students to the concept of ultrasonic sensors and how they function. Guide them through creating a new project on the MakeCode website, adding the kitronik-move-motor extension. Assist them in programming the sensor to measure distance and display it on the Microbit. Progress to programming the car to maintain a 10cm distance from an object, including reversing. Finally, challenge students to improve the code, adding lights and randomised movement to enhance the car's obstacle avoidance.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- Move Motor Car

Learning Goals	Learning Outcomes
1. Understand the function and operation of ultrasonic sensors. 2. Develop skills in creating a new project using the kitronik-move-motor extension. 3. Acquire the ability to program a sensor to measure distance. 4. Learn to code a car to maintain a specific distance from an object. 5. Enhance problem-solving skills by programming the car to reverse and maintain distance.	1. Understand the function and operation of an ultrasonic sensor. 2. Create a new project on the MakeCode Microbit website and add the necessary extension. 3. Program the sensor to measure and display the distance to an object. 4. Modify the code to make the car maintain a distance of 10cm from an object. 5. Enhance the code to reverse the car until it is exactly 10cm away from an object. 6. Program the car to free roam and avoid objects by stopping, reversing, and turning right when an object is detected within 10cm. 7. Improve the code for better navigation and add lights for visual feedback.

Week 7

Lesson: Tilt Remote Control Car

□ Advanced	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

In this lesson, students will learn to control a Move Motor car using a Microbit as a remote controller. They will create two code projects: one for the remote control and another for the car. The lesson involves programming the Microbit to detect tilts in different directions and send corresponding messages to the car. The students will also add code to stop the car and to light up the LEDs on the car in different colours. Ensure each remote and car set uses a different radio group to avoid crossed signals.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- Move Motor Car

Learning Goals	Learning Outcomes
1. Understand and apply the concept of radio communication between two Microbits. 2. Programme a Microbit to send specific messages based on different gestures. 3. Develop the ability to programme a Move Motor car to respond to different messages received. 4. Test and debug the code to ensure the car responds correctly to the remote control. 5. Extend the project by adding additional features such as LED light changes.	1. Programme a Microbit as a remote control to send directional commands. 2. Programme a Microbit to receive and execute directional commands in a Move Motor car. 3. Test and debug the code to ensure correct functioning of the remote-controlled car. 4. Download and implement the code onto the Microbits. 5. Extend the code to include LED light changes in response to different commands as an additional challenge.

Week 8

Lesson: Traffic Lights and Car Communication

□ Advanced	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

This lesson involves coding a set of traffic lights and a robot car using Microbits. Students will program the traffic lights to display a sequence and broadcast the light being shown. The robot car will receive this broadcast and decide whether to stop or go. The lesson involves creating two code projects, adding a 'stopbit' extension, programming a sequence, broadcasting the state, programming the car, receiving the message, downloading the code, and an additional challenge. Teachers should ensure they have the necessary equipment and familiarise themselves with the coding platforms used.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- Traffic Lights Kit
- Move Motor Car

Learning Goals	Learning Outcomes
1. Understand and apply the concept of radio communication between Microbits. 2. Program a sequence of traffic light signals using code blocks. 3. Develop skills to broadcast and receive specific messages based on traffic light states. 4. Control the movement of a robot car based on received messages. 5. Enhance problem-solving skills by modifying the code to respond based on the proximity of the car to the traffic lights.	1. Code a set of traffic lights to run through a sequence and broadcast the displayed light. 2. Program a robot car to receive the broadcast and decide whether to stop or go based on the traffic light signal. 3. Use the "stopbit" extension to create custom code blocks for programming the traffic lights kit. 4. Program the car to move at different speeds or stop, depending on the received message from the traffic lights. 5. Modify the code to make the car respond to the traffic lights based on its proximity to them.

Module: Discovering Artificial Intelligence

This module explores the fascinating world of artificial intelligence (AI), starting with an introduction to AI models, their types, applications, and limitations. Students will gain hands-on experience creating image and pose models using Google's Teachable Machine, and applying these models in interactive games using Scratch. The module culminates in a project where students conceptualise, plan, and build their own AI Scratch project, applying their newfound knowledge and skills. Teachers should familiarise themselves with the tools and concepts, and be prepared to guide students through each step, encouraging creativity and problem-solving throughout.

Duration	**Equipment**
4 weeks	Students can use any of these devices: • Chromebook/Laptop/PC • iPad/Tablet Required Equipment: • Webcam/camera
Module Goals	**Module Outcomes**
1. Understand the fundamentals of AI models, their types, applications, limitations, and ethical considerations. 2. Develop an image model using Google's Teachable Machine and apply it in a practical project. 3. Create an interactive game using Scratch and Google Teachable Machine, incorporating elements of randomisation and conditionals. 4. Design and develop a pose model using Google's Teachable Machine, and apply it in a space game project. 5. Conceptualise, plan, and execute an original AI Scratch project, demonstrating creativity, problem-solving, and application of AI knowledge.	1. Understand and explain the function, types, applications, and limitations of AI models, including ethical considerations. 2. Create an image model using Google's Teachable Machine for a rock, paper, scissors game. 3. Develop a Rock, Paper, Scissors game using Scratch and Google Teachable Machine, incorporating variables, randomisation, and conditionals. 4. Create a pose model using Google's Teachable Machine for a space game, understanding the importance of testing and adjusting the model. 5. Conceptualise, plan, and build a unique AI Scratch project, demonstrating creativity, problem-solving, and the ability to seek and incorporate feedback.

Week 1

Lesson: An Introduction to AI Models

□ Beginner	□ 20 mins	□ Teacher/Student led	□ Student Quiz

Prepare to introduce students to AI models, explaining their function and various types. Discuss different learning methods such as supervised, unsupervised, and reinforcement learning. Explore the diverse applications of AI models, from speech recognition to autonomous vehicles. Discuss the limitations of AI models, including data quality and computational resources. Finally, delve into the ethics of AI models, discussing responsibility, privacy, transparency, and fairness.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
1. Understand the concept and purpose of AI models. 2. Identify different types of AI models and their learning methods. 3. Recognise various applications of AI models in real-world scenarios. 4. Appreciate the limitations and challenges associated with AI models. 5. Reflect on the ethical considerations in the use of AI models.	1. Identify and describe the different types of AI models: Supervised Learning, Unsupervised Learning, and Reinforcement Learning. 2. Explain the various applications of AI models, including speech recognition, image recognition, natural language processing, recommendation systems, and autonomous vehicles. 3. Discuss the limitations of AI models, focusing on data quality, computational resources, transparency, privacy, and security. 4. Understand the ethical considerations related to AI models, including responsibility, privacy, transparency, and fairness. 5. Demonstrate a basic understanding of how AI models function, their uses, limitations, and ethical implications.

Lesson: Create an Image Model

□ Intermediate	□ 20 mins	□ Teacher/Student led	□ Student Quiz

Familiarise yourself with Google's Teachable Machine tool before the lesson. Ensure students understand the concept of machine learning and how it applies to image recognition. Encourage students to take clear images for their classes and emphasise the importance of quality over quantity. Guide them through the process of training, testing, and exporting their models. Reinforce the practical application of these skills in future projects.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Required equipment for this lesson:

- Webcam/camera

Learning Goals	Learning Outcomes
1. Understand and utilise Google's Teachable Machine to create an image model. 2. Create and define classes within an image model project. 3. Add and manage image samples to each class for effective model training. 4. Train, test, and refine the image model to ensure accurate gesture recognition. 5. Export and save the created image model for future use in projects.	1. Utilise Google's Teachable Machine to create an image model. 2. Create and categorise classes within an image model project. 3. Add and record images to each class using a webcam. 4. Train the image model using the added images and understand the process of machine learning. 5. Test the model's performance, make necessary adjustments, and export the model for future use.

Week 2

Lesson: Scratch AI Rock, Paper, Scissors Game

□ Intermediate	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

Prepare to guide students through creating a Rock, Paper, Scissors game using Scratch and Google Teachable Machine. Ensure they understand the use of variables, randomisation, and conditionals. They'll need to set up Scratch and TM2Scratch, add a sprite, create variables, and load a Teachable Machine Model. They'll also learn to set a confidence threshold, get player choice, determine game outcomes, and add enhancements. Encourage creativity and problem-solving throughout.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Required equipment for this lesson:

- Webcam/camera

Learning Goals	Learning Outcomes
1. Develop a Rock, Paper, Scissors game using Scratch and Google Teachable Machine. 2. Understand and apply the use of variables in Scratch for storing player's choice, computer's choice, and the result of the game. 3. Implement randomisation in Scratch to simulate the computer's choice in the game. 4. Integrate Google Teachable Machine Image models in Scratch projects for gesture recognition. 5. Understand and adjust the confidence threshold for AI model to improve accuracy of gesture recognition.	1. Develop a Rock, Paper, Scissors game using Scratch and Google Teachable Machine. 2. Set up Scratch and TM2Scratch for the game development. 3. Create and utilise variables to store player's choice, computer's choice, and the game result. 4. Implement randomisation for computer's choice in the game. 5. Load and use a Teachable Machine Image model for hand gesture recognition. 6. Set and adjust the confidence threshold for the AI model. 7. Recognise and interpret player's choice through hand gestures. 8. Develop game logic to determine the game result: draw, win, or lose. 9. Improve the game by enhancing the image model, adding new features like sound effects, and improving user interaction.

Week 3

Lesson: Create a Pose Model

□ Intermediate	□ 20 mins	□ Teacher/Student led	□ Student Quiz

Prepare to guide students through creating a pose model using Google's Teachable Machine. Familiarise yourself with the tool and the process of creating classes, adding images, and training the model. Be ready to troubleshoot any issues with webcam permissions or image quality. Ensure students understand the importance of testing their model and making necessary adjustments. Finally, assist them in exporting their model for future use in projects like an AI-powered space game.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Required equipment for this lesson:

- Webcam/camera

Learning Goals	Learning Outcomes
1. Develop an understanding of Google's Teachable Machine and its application in creating pose models. 2. Acquire skills to create and categorise classes within a pose model. 3. Learn to add and manage image samples for each class to train the model. 4. Gain proficiency in training and testing the model for different poses. 5. Master the process of exporting the model for future use in other projects.	1. Operate Google's Teachable Machine to create a pose model. 2. Define and create classes for the pose model. 3. Add and categorise images into the respective classes: Tilt Left, Tilt Right, and No Tilt. 4. Train the pose model using the categorised images and test its performance. 5. Export the created pose model and obtain a shareable link for future use.

Lesson: Scratch AI Pose Space Game

□ Advanced	□ 60 mins	□ Teacher/Student led	□ Student Quiz	□ Student Challenge

Prepare to guide students in creating a Scratch AI Pose Space Game. They'll learn to use Scratch and Google Teachable Machine to control a spaceship with tilt poses. They'll set up Scratch, add a rocketship sprite, load a Teachable Machine Model, display pose labels, set a confidence threshold, and make the spaceship move. They'll also add a star sprite, make stars fall, and face a challenge to improve their game. Ensure students understand each step, and encourage creativity in the challenge.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Required equipment for this lesson:

- Webcam/camera

Learning Goals	Learning Outcomes
1. Develop skills in using Scratch and Google Teachable Machine to create a game. 2. Understand how to control a sprite using pose models. 3. Learn to set up and adjust the confidence threshold for an AI model. 4. Gain knowledge on how to create and manipulate clones of sprites in Scratch. 5. Apply creativity to enhance and personalise the game with additional features.	1. Create a Space game using Scratch and Google Teachable Machine. 2. Set up Scratch and TMPose2Scratch for a new project. 3. Integrate a Teachable Machine Pose model into the Scratch project. 4. Control a sprite's movement using pose labels and confidence thresholds. 5. Enhance the game by adding falling sprites and scoring mechanisms.

Week 4

Lesson: Crafting Your Own AI Project

□ Advanced	□ 60 mins	□ Teacher/Student led

In this lesson, students will utilise their knowledge of AI and Scratch to create their own AI project. They will brainstorm ideas, focusing on real-life routines or challenges that could be enhanced with AI. After selecting their favourite idea, they will create a project proposal, seek feedback, refine their idea, and plan their project. They will then prototype and code their project, before presenting and demonstrating their work. Finally, they will reflect on their learning journey and the process of creating their project.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Required equipment for this lesson:

- Webcam/camera

Learning Goals	Learning Outcomes
1. Develop the ability to conceptualise and plan an AI project. 2. Enhance brainstorming skills and generate creative AI project ideas. 3. Gain proficiency in creating and refining a project proposal. 4. Acquire skills in prototyping and coding an AI project using Scratch and Google Teachable Machine. 5. Improve presentation skills and ability to reflect on the learning process and project outcomes.	1. Generate and evaluate 3-5 AI project ideas, drawing inspiration from daily routines or challenges. 2. Formulate a detailed project proposal, including project name, purpose, required features, and necessary components. 3. Seek and incorporate feedback from peers or teachers to refine the project idea and plan. 4. Code, prototype, and test the core features of the AI project, using problem-solving skills to overcome any issues. 5. Present and demonstrate the final AI project, reflecting on the challenges faced, solutions found, and lessons learned.