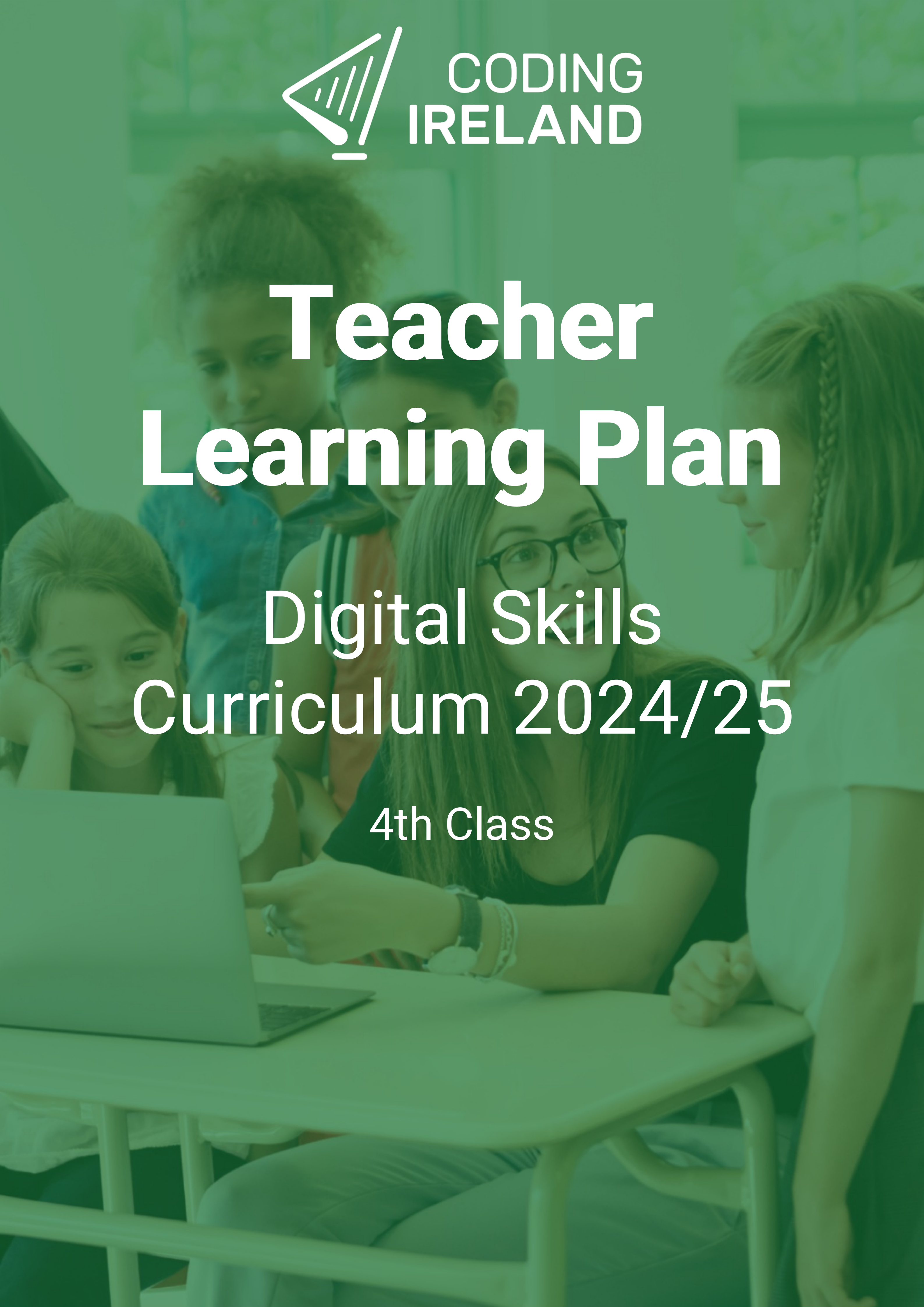


CODING IRELAND

Teacher Learning Plan

Digital Skills Curriculum 2024/25

4th Class

Table of Contents

How to Use This Learning Plan

This learning plan provides an overview of all the modules available for 4th Class, 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.

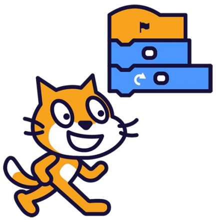
Student Access

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@codingireland.ie or 01 584 9955.

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	System.Threading.Tasks.Task`1[System.String]	☰ 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	System.Threading.Tasks.Task`1[System.String]	☰ 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	System.Threading.Tasks.Task`1[System.String]	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: Advanced Game Development

This module introduces students to game design using MakeCode Arcade, a user-friendly tool for creating arcade games. Teachers should familiarise themselves with MakeCode Arcade and guide students through creating various games, from simple sprite control to complex maze navigation. Encourage creativity, problem-solving, and teamwork throughout The module. The final module allows students to brainstorm and create their own game, fostering a sense of achievement and showcasing their newly acquired skills.

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 design and development. 2. Develop skills in creating and controlling game characters, including movement and interaction. 3. Understand and apply game design concepts such as sprite creation, random object generation, score keeping, and timer functions. 4. Enhance problem-solving abilities and creativity through game modification and design challenges. 5. Collaborate effectively in teams to brainstorm, receive feedback, and develop a game project.	1. Master the use of MakeCode Arcade to design and control a sprite, and add effects. 2. Create an interactive game 'Shark Attack' using MakeCode Arcade, including character creation, interaction, and dynamic gameplay. 3. Develop a game 'Monkey Mayhem', controlling a character to collect objects and adding a countdown timer for challenge. 4. Design a game 'Target Test', creating and moving sprites, generating targets, shooting projectiles, and keeping score. 5. Participate in build battles, demonstrating coding skills and creativity. 6. Construct a maze game 'Prison Break', designing a character, creating a maze, setting a goal, and adding a timer. 7. Create a game 'Car Collector', controlling a character, adding items to collect, and creating an enemy. 8. Brainstorm and develop a unique MakeCode Arcade project in a team, demonstrating creativity and teamwork.

Week 1

Lesson: Introducing MakeCode Arcade

🟢 Beginner	🕒 60 mins	System.Threading.Tasks.Task`1[System.String]	Student Quiz	Student Challenge

This lesson introduces MakeCode Arcade, a tool for creating arcade games. Teachers should familiarise themselves with the MakeCode Arcade interface and its features, including the code editor, simulator, and toolbox. The lesson guides students through creating a new project, designing a sprite, controlling the sprite's movements, keeping the sprite on screen, and adding effects. Teachers should encourage students to experiment with different effects and explore the potential of MakeCode Arcade.

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 functionality and usage of MakeCode Arcade. 2. Develop skills in using the MakeCode Arcade code editor. 3. Create and customise a sprite in MakeCode Arcade. 4. Implement controls to move a sprite and keep it within the screen boundaries. 5. Apply and experiment with different effects in MakeCode Arcade.	1. Understand the basic features and functions of MakeCode Arcade. 2. Utilise the MakeCode Arcade code editor effectively. 3. Create and name a new Arcade project on the MakeCode website. 4. Generate and design a sprite character using code. 5. Implement code to control sprite movement and keep it within the screen boundaries. 6. Add and modify effects to the sprite character. 7. Explore further possibilities in game creation using MakeCode Arcade.

Week 2

Lesson: Shark Attack

🟢 Beginner	🕒 60 mins	System.Threading.Tasks.Task`1[System.String]	☰ Student Quiz	💡 Student Challenge

In this lesson, students will create a game called 'Shark Attack' using MakeCode Arcade. They will learn how to create a new project, design a player sprite, and control its movements. They will also learn how to keep the sprite within the screen boundaries, create enemy sprites, set their positions, and make them chase the player sprite. The lesson will conclude with the students learning how to detect overlaps between sprites and end the game. Encourage students to experiment with their game, adding more enemies or power-ups.

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 platform for game creation. 2. Understand and apply coding concepts to create and control sprites. 3. Implement game logic to create dynamic gameplay, including enemy creation and movement. 4. Apply coding techniques to detect sprite overlaps and trigger game events. 5. Enhance problem-solving and creativity by modifying and expanding game features.	1. Utilise MakeCode Arcade platform to create a new project. 2. Create and customise a player sprite using provided code. 3. Implement sprite movement using joystick or keyboard arrow keys. 4. Apply code to restrict sprite movement within screen boundaries. 5. Create enemy sprites that appear every 5 seconds. 6. Set enemy sprites to appear at a specific position on the screen. 7. Code enemy sprites to follow the player sprite. 8. Implement game over condition when player sprite overlaps with enemy sprite.

Week 3

Lesson: Monkey Mayhem

🟢 Beginner	60 mins	System.Threading.Tasks.Task`1[System.String]	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 a game using MakeCode Arcade. 2. Create and control a player sprite in the game. 3. Generate food sprites at random positions on the screen. 4. Collect food sprites for points and add effects and sounds. 5. Implement a countdown timer to increase game difficulty.	1. Create a new project in MakeCode Arcade. 2. Generate a player sprite and control its movement. 3. Create food sprites at regular intervals. 4. Position food sprites randomly on the screen. 5. Implement sprite overlap detection to collect food sprites. 6. Score points for each collected food sprite. 7. Play a sound effect upon food sprite collection. 8. Implement a countdown timer for game duration.

Week 4

Lesson: Target Test

🟢 Beginner	🕒 60 mins	System.Threading.Tasks.Task`1[System.String]	☑ Student Quiz	💡 Student Challenge

Prepare to guide students through creating a game using MakeCode Arcade. They will learn to create and move sprites, generate targets, shoot projectiles, and keep score. Ensure students understand how to use the MakeCode Arcade website and the basics of coding. Be ready to explain concepts such as sprites, joystick controls, random number generation, and event handling. Encourage creativity and problem-solving as students add their own features to the game.

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 moving sprites using MakeCode Arcade. 2. Understand how to generate targets at random positions within a game environment. 3. Gain proficiency in programming game controls, such as shooting projectiles. 4. Learn to implement a scoring system within a game. 5. Encourage creativity and problem-solving by adding additional features to the game.	1. Create and manipulate player sprite using MakeCode Arcade. 2. Initiate a new project on the MakeCode Arcade platform. 3. Programme player sprite to move left and right using joystick or keyboard arrows. 4. Generate target sprites at random positions at the top of the screen. 5. Programme the A button to fire a projectile from the player sprite. 6. Implement a scoring system that awards a point for each target hit.

Week 5

Lesson: Arcade Build Battles

● Intermediate	🕒 60 mins	System.Threading.Tasks.Task`1[System.String]

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: Prison Break

● Advanced	60 mins	System.Threading.Tasks.Task`1[System.String]	Student Quiz	Student Challenge

Prepare to guide students through the creation of a 'Prison Break' game using MakeCode Arcade. They will design a character, create a maze, set a goal and add a timer for challenge. Ensure familiarity with the MakeCode Arcade interface, sprite creation, and basic coding concepts. Encourage creativity in maze and character design, and emphasise the importance of testing at each stage. Celebrate their accomplishment in creating a complex, interactive game.

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 game creation using MakeCode Arcade. 2. Understand and apply coding concepts to create and control a character sprite. 3. Design and implement a maze using tilemap and wall settings. 4. Integrate game elements such as a goal tile and timer to enhance gameplay. 5. Apply problem-solving skills to navigate through the maze within a set time limit.	1. Create a new Arcade project using MakeCode Arcade. 2. Design and implement a sprite character for the game. 3. Enable character movement using joystick or keyboard arrow keys. 4. Design a maze using the tile map editor and set walls to restrict character movement. 5. Implement camera follow functionality to track character movement. 6. Set a goal tile in the maze and implement game win condition upon reaching the goal. 7. Add a countdown timer to increase game challenge.

Week 7

Lesson: Car Collector

● Advanced	60 mins	System.Threading.Tasks.Task`1[System.String]	Student Quiz	Student Challenge

In this lesson, students will create a game using MakeCode Arcade. They will learn to control a player character, add items to collect, and create an enemy. The lesson will guide them through creating a new project, adding a background, creating a player character, moving the character, changing the sprite image, creating food sprites, detecting overlap with food, creating an enemy, and setting conditions for game over. The lesson will also teach students about X and Y coordinates.

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 character in MakeCode Arcade. 2. Learn to add collectable items within the game environment. 3. Understand how to create an enemy character and implement game over conditions. 4. Gain experience in using code to generate random positions for game elements. 5. Enhance problem-solving and game design skills through coding practice.	1. Develop a game using MakeCode Arcade. 2. Create and control a player character. 3. Add items for the player to collect. 4. Generate an enemy character. 5. Implement game over conditions.

Week 8

Lesson: Game Lab

● Advanced	60 mins	System.Threading.Tasks.Task`1[System.String]

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 Adventures

This module introduces students to the world of microbits, pocket-sized programmable computers. Teachers will guide students through creating projects, writing code, and using various features of the microbit. The module is hands-on, encouraging students to experiment and learn through doing. Teachers should ensure students handle the microbits carefully, particularly when using them as pedometers. The module culminates in a group project, fostering creativity and teamwork.

Duration	Equipment
8 weeks	Students can use any of these devices: • Chromebook/Laptop/PC Required Equipment: • Crocodile clips • Microbit • Some fruit & vegetables
Module Goals	**Module Outcomes**
1. Understand and utilise the basic functions of a microbit, including creating a new project and writing code. 2. Develop skills to use the microphone in a microbit for detecting sound and creating light effects. 3. Apply the accelerometer feature in a microbit to create a step counter. 4. Use the sensors in a microbit to create a compass, thermometer, and sound level detector. 5. Design and execute a unique microbit project, demonstrating creativity, problem-solving, and teamwork.	1. Program a microbit to display messages, react to button presses, and show a happy face. 2. Utilise the microphone in a microbit to detect clapping and turn on lights. 3. Convert a microbit into a pedometer to count and display steps. 4. Transform a microbit into a sound level detector using its built-in microphone. 5. Conduct electricity through the body and fruits to make a microbit play music. 6. Program a microbit to function as a compass and thermometer using its built-in sensors. 7. Create a 'Hot Potato' game using a microbit and its countdown feature. 8. Brainstorm, design, and execute a simple microbit project in a team, demonstrating creativity and teamwork.

Week 1

Lesson: Meet the Microbit

🟢 Beginner	🕓 60 mins	System.Threading.Tasks.Task`1[System.String]	☰ Student Quiz	💡 Student Challenge

Prepare to introduce students to the world of microbit, a pocket-sized programmable computer. Ensure familiarity with the MakeCode for microbit website, where students will create a new project. Guide them through the project editor, including the microbit simulator, toolbox, and code area. Facilitate the creation of their first code, showing numbers and names, and the use of buttons to display icons. Assist in connecting the microbit to the computer for real-world application. Encourage exploration and experimentation with different blocks and functions.

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. Develop skills in creating a new project on the MakeCode for microbit platform. 3. Gain familiarity with the Project Editor interface, including the Microbit Simulator, Toolbox, and Code Area. 4. Acquire the ability to write simple code to display numbers, text, and icons on the microbit. 5. Explore and experiment with different coding blocks to create interactive microbit programs.	1. Identify the key features and functions of a microbit. 2. Create a new project using the MakeCode for microbit website. 3. Understand and navigate the project editor including the microbit simulator, toolbox, and code area. 4. Write and modify code to display numbers and text on the microbit's LED grid. 5. Program the microbit's buttons to display specific messages or icons.

Week 2

Lesson: Microbit Light Clapper

🟢 Beginner	🕒 40 mins	System.Threading.Tasks.Task`1[System.String]	☑ Student Quiz	💡 Student Challenge

Prepare for the 'Microbit Light Clapper' lesson by familiarising yourself with the makecode.microbit.org website. Understand the process of creating a new project, setting up variables, and using sound thresholds. Be ready to guide students in writing code to detect claps and control LED lights. Ensure you can troubleshoot issues and explain how to test the code in the simulator and on the Microbit.

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. Set and utilise sound thresholds for input detection. 3. Implement conditional statements (if-then-else) to control LED light responses. 4. Test and debug code in a simulator environment. 5. Transfer and apply code to a physical Microbit device.	1. Develop a new project using makecode.microbit.org. 2. Create and utilise a variable to control the LED lights on the Microbit. 3. Set a sound threshold for detecting claps using the Microbit's microphone. 4. Implement code to detect a clap based on the set sound threshold. 5. Use an 'if then else' block to control the LED lights based on the clap detection.

Week 3

Lesson: Microbit Step Counter

🟢 Beginner	🕒 60 mins	System.Threading.Tasks.Task`1[System.String]	Student Quiz	Student Challenge

Prepare to guide students through creating a Microbit step counter. They'll start a new project on makecode.microbit.org, create and set up a 'steps' variable, and use the accelerometer to detect steps. They'll write code to display the step count and send it to their Microbit. After connecting a power source, they'll secure the Microbit to their person and start walking. They'll adjust the code to count every step and resend the updated code to their Microbit. Caution them to be careful while walking with the Microbit.

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 programming and project creation. 2. Learn to create and set up variables in Microbit. 3. Understand the use of accelerometer sensor in Microbit for step detection. 4. Gain skills to display data on Microbit using LEDs. 5. Learn to modify and resend code to Microbit for improved functionality.	1. Develop a new Microbit project using the makecode.microbit.org website. 2. Create and set up a 'steps' variable to record the number of steps taken. 3. Utilise the accelerometer sensor in Microbits to detect and record steps. 4. Display the recorded number of steps on the Microbit using its LEDs. 5. Modify the code to accurately count every step taken, and resend the updated code to the Microbit.

Week 4

Lesson: Sound level

● Beginner	40 mins	System.Threading.Tasks.Task`1[System.String]	Student Quiz	Student Challenge

This lesson involves creating a new Microbit project on makecode.com, where students will learn to show the sound level as a graph using the built-in microphone of Microbits (version 2). They will use the 'plot bar graph' block to display the current sound level. The lesson concludes with a challenge to turn the Microbit into a warning device that sounds an alarm when the sound level exceeds a certain limit. Students will need to think about the necessary code and then try to implement it.

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 a new Microbit project using makecode.com. 2. Understand and apply the concept of sound level detection using the built-in microphone in Microbits. 3. Learn to display the detected sound level as a graph on the Microbit. 4. Apply coding skills to manipulate the sound level block and the plot bar graph block. 5. Engage in a challenge to utilise learned skills in a practical scenario, turning the Microbit into a warning device.	1. Create a new Microbit project using the makecode.com website. 2. Understand and utilise the sound level block in Microbits to detect and store sound levels. 3. Display the detected sound level as a graph on the Microbit using the plot bar graph block. 4. Conceptualise and code a warning device that triggers an alarm when the sound level exceeds a certain threshold. 5. Apply the sound level block to monitor sound levels and activate the Microbit speaker as an alarm.

Week 5

Lesson: Microbit Fruit and Veg Piano

● Intermediate	60 mins	System.Threading.Tasks.Task`1[System.String]	Student Quiz	Student Challenge

Prepare to demonstrate the conductivity of the human body and various fruits and vegetables using a Microbit. Gather a Microbit, 4 crocodile clips, and 4 pieces of fruit or vegetables. Familiarise yourself with the Microbit programming interface and the specific code for programming Pins 0, 1, and 2. Ensure you understand how to connect the crocodile clips and test the circuits. Be ready to guide students in connecting the fruit and vegetables to create a musical instrument.

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

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- Crocodile clips
- Some fruit & vegetables

Learning Goals	Learning Outcomes
1. Understand and apply the concept of electrical conductivity using the human body and various fruits and vegetables. 2. Identify and utilise the components of a Microbit, including its pins and GND. 3. Create and modify a Microbit project using the makecode.microbit.org platform. 4. Program Microbit pins to play different musical notes and display different icons. 5. Test and troubleshoot a simple electrical circuit using a Microbit, crocodile clips, and conductive materials.	1. Identify and gather necessary materials for creating an electrical circuit with a Microbit and fruit or vegetables. 2. Create a new project on the makecode.microbit.org website. 3. Program Pins 0, 1, and 2 on the Microbit to play different notes and display different icons when pressed. 4. Connect crocodile clips to Pins 0, 1, 2 and GND on the Microbit and test the circuit. 5. Attach fruit or vegetables to the crocodile clips and demonstrate the ability to play different notes by touching and releasing each piece.

Week 6

Lesson: Microbit Compass and Thermometer

● Intermediate	60 mins	System.Threading.Tasks.Task`1[System.String]	Student Quiz	Student Challenge

Prepare to guide students in creating a Microbit project that utilises the compass and temperature sensor. They will learn to create and set variables, program buttons, and use 'if then else' blocks. The lesson involves coding the Microbit to display cardinal directions based on its orientation and temperature readings. Students will also test their code using a simulator before sending it to their Microbit. Ensure familiarity with the makecode.com platform and basic coding concepts.

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 utilise the compass and temperature sensor features of the Microbit. 2. Develop proficiency in creating and setting variables in a Microbit project. 3. Apply conditional logic to program Microbit buttons for specific functions. 4. Test and debug code using the simulator before transferring to the Microbit. 5. Interpret and display data from the Microbit's sensors in a user-friendly format.	1. Develop a new Microbit project using makecode.com. 2. Create and set a 'direction' variable to store compass readings. 3. Program the A button to display compass direction (N, S, E, W) based on 'direction' variable. 4. Program the B button to display the current temperature reading. 5. Test and debug the code using the simulator and then deploy it to the Microbit.

Week 7

Lesson: Microbit Hot Potatoe

● Advanced	60 mins	System.Threading.Tasks.Task`1[System.String]	Student Quiz	Student Challenge

In this lesson, students will create a 'Microbit Hot Potato' game. They'll start by creating a new project on the Microbit website, then create a variable for a random countdown. They'll code a countdown sequence, add an animation, and a game over signal. Finally, they'll download the code, power their Microbit, and play the game. Ensure students understand the concept of variables, countdowns, and basic coding principles.

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 creating and manipulating variables in Microbit. 2. Apply the concept of random number generation in a practical scenario. 3. Understand and implement countdown functionality using loops. 4. Integrate visual animations into the code to enhance user experience. 5. Understand how to incorporate sound and visual effects to indicate the end of a game.	1. Develop a new project using the Microbit platform. 2. Create and utilise a variable to store and manipulate countdown seconds. 3. Implement a countdown function that decreases the stored seconds until zero. 4. Integrate an animation to display during the countdown process. 5. Design a game over function that displays an icon and plays a sound when the countdown reaches zero. 6. Apply the developed code to a Microbit device and engage in gameplay with peers.

Week 8

Lesson: Microbit Lab

● Advanced	60 mins	System.Threading.Tasks.Task`1[System.String]

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: Dynamic Digital Projects

This module involves teaching students to create interactive games and projects using Scratch. The lessons cover a range of topics, from controlling sprites and implementing game logic, to using variables, creating clones, and adding sound effects. Teachers should familiarise themselves with the Scratch platform and be prepared to guide students through each step of the project creation process. Encourage creativity and problem-solving, and ensure students understand each concept before moving on. Celebrate their final creations and encourage further exploration of Scratch's features.

Duration	Equipment
8 weeks	Students can use any of these devices: • Chromebook/Laptop/PC • iPad/Tablet
Module Goals	**Module Outcomes**
1. Master the creation and programming of interactive games using Scratch, including controlling sprites, implementing game logic, and using conditionals. 2. Understand and apply the concept of variables for scorekeeping and clones for game elements in Scratch. 3. Develop proficiency in coding keyboard controls, modifying sprite size and position, and preventing character passage through maze walls in Scratch. 4. Gain competence in using the Text to Speech feature in Scratch, including making characters speak, altering their voices and accents, and creating dialogues. 5. Learn to create dynamic visual effects in Scratch, such as a trailing effect with clones, colour changes, and adding costumes for different shapes.	1. Design and programme an interactive game using Scratch, controlling sprite movements and implementing game logic. 2. Utilise variables and clones in Scratch to create a scoring system and dynamic game elements. 3. Develop a Maze Game in Scratch, programming character navigation and collision detection. 4. Implement the Text to Speech feature in Scratch, creating dialogues between characters with varied voices and accents. 5. Create an interactive 'Pattern Snake' in Scratch, using clones for trailing effects and costumes for variety. 6. Build a game in Scratch involving loops, conditionals, and variables, enhancing coding skills through practical application. 7. Add sound effects to a Scratch game, enhancing user experience and game dynamics. 8. Animate a sprite in Scratch and participate in build battles, demonstrating creativity and time management skills.

Week 1

Lesson: Diver Game

🟢 Beginner	🕒 60 mins	System.Threading.Tasks.Task`1[System.String]	☑ Student Quiz	💡 Student Challenge

Prepare for a fun and interactive lesson where students will create a 'Diver Game' using Scratch. They will learn to control a diver sprite with their mouse, collect starfish appearing at random positions, and avoid a shark that follows the diver. The lesson will cover creating a new Scratch project, adding and programming sprites, and implementing game mechanics like random positioning and game over conditions. Ensure students understand each step before proceeding to the next.

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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
1. Understand and apply the basic principles of game design using Scratch. 2. Develop skills in programming sprites to interact with mouse movements. 3. Master the use of 'clones' in Scratch for dynamic sprite generation. 4. Gain proficiency in using conditional statements to control game outcomes. 5. Enhance problem-solving and debugging skills in a coding environment.	1. Control the Diver1 sprite using mouse movements in the Scratch environment. 2. Create and manipulate a new Scratch project, including adding and removing sprites. 3. Program the Starfish sprite to appear at random positions and interact with the Diver1 sprite. 4. Introduce the Shark 2 sprite and program it to chase the Diver1 sprite. 5. Implement a game over condition when the Shark 2 sprite catches the Diver1 sprite.

Week 2

Lesson: Star Chaser

🟢 Beginner	🕒 60 mins	System.Threading.Tasks.Task`1[System.String]	Student Quiz	Student Challenge

Prepare to guide students through the creation of a fun game using Scratch, where a character named Ripley chases stars. The lesson will introduce the concept of 'variables' for scorekeeping and 'clones' for game elements. Students will learn to create a new Scratch project, add and manipulate sprites, create a score variable, and program sprite clones. They will also learn to control sprite movement and interactions, and use variables to track game progress. The lesson concludes with a game play and a wrap-up session.

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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
1. Develop understanding and practical application of 'variables' in coding. 2. Gain proficiency in creating and controlling 'clones' within a digital project. 3. Acquire skills to manipulate sprite size and movement using specific code blocks. 4. Learn to create a scoring system using variables in a game environment. 5. Enhance problem-solving and creative thinking skills through game development.	1. Develop a Scratch project by adding and controlling sprites and backdrops. 2. Manipulate sprite size and movement using 'set size to', 'point towards', and 'move' blocks. 3. Create and utilise a variable to keep track of scores in a game. 4. Implement 'hide', 'wait', and 'create clone of' blocks to create dynamic game elements. 5. Program sprite clones to interact with other sprites and respond to events using 'if then', 'start sound', 'change', and 'delete this clone' blocks.

Week 3

Lesson: Maze Game

● Beginner	🕒 60 mins	System.Threading.Tasks.Task`1[System.String]	Student Quiz	Student Challenge

Prepare for an engaging lesson on creating a Maze Game using Scratch. Familiarise yourself with the Scratch platform and the starter project provided. Understand the steps to add and modify the Beetle sprite, and how to code it to move through the maze using arrow keys. Be ready to guide students in programming the beetle to not pass through walls. Encourage creativity as students play and modify their games.

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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
1. Understand and apply basic programming concepts using Scratch. 2. Manipulate sprite properties such as size and position. 3. Program keyboard inputs to control sprite movement. 4. Implement collision detection to prevent sprite from passing through obstacles. 5. Develop problem-solving skills and creativity by modifying and enhancing the game.	1. Program a character to navigate through a maze using Scratch. 2. Modify the size and position of a sprite in Scratch. 3. Code the arrow keys to control the movement of a sprite in Scratch. 4. Implement a collision detection mechanism to prevent a sprite from passing through obstacles in Scratch. 5. Test and play a self-created maze game in Scratch.

Week 4

Lesson: Text to Speech

● Beginner	45 mins	System.Threading.Tasks.Task`1[System.String]	Student Quiz	Student Challenge

For this lesson, teachers should familiarise themselves with the Scratch platform and its Text to Speech extension. They should be prepared to guide students through creating a new Scratch project, adding sprites, and using the Text to Speech blocks to make their sprites speak. Teachers should also be ready to assist students in changing the voice and accent of their sprites, and in creating a conversation between two sprites.

Students will need to use the speakers on their computers for this lesson.

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 the Text to Speech extension in Scratch. 2. Create and modify Scratch projects with speaking characters. 3. Experiment with different voices and accents using the Text to Speech extension. 4. Develop a conversation between two characters using different voices and accents. 5. Apply coding skills to create more complex and interesting conversations in Scratch projects.	1. Utilise the Text to Speech extension in Scratch to make characters speak. 2. Create and modify a Scratch project, including adding and deleting sprites. 3. Alter the speech content of a sprite using the 'speak' block. 4. Change the voice and accent of a sprite using the 'set voice to' and 'set language to' blocks. 5. Programme a conversation between two sprites with different voices and accents.

Week 5

Lesson: Pattern Snake

● Intermediate	60 mins	System.Threading.Tasks.Task`1[System.String]	Student Quiz	Student Challenge

Prepare to guide students through creating a dynamic 'Pattern Snake' on Scratch. They'll learn to draw a sprite, code it to follow the mouse, create clones, and change colours. They'll also add costumes for variety. Ensure students understand each step before moving on. Celebrate their final creations, encouraging further exploration of Scratch's features.

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 clones in Scratch programming. 3. Gain proficiency in using control structures to create movement and interaction in Scratch. 4. Learn to use colour and costume changes to enhance visual effects in Scratch projects. 5. Enhance creativity and problem-solving skills through the design and execution of a unique digital project.	1. Draw and code a sprite to follow the mouse pointer around the screen in Scratch. 2. Create, position, and delete clones of the sprite to create a trailing snake effect. 3. Implement colour changes to the sprite after each clone creation for enhanced visual effect. 4. Add and switch between different sprite costumes using the space bar for varied shapes. 5. Complete a Scratch project, demonstrating the ability to combine various coding elements to create a visually appealing pattern.

Week 6

Lesson: Easter Egg Catch

● Intermediate	🕒 60 mins	System.Threading.Tasks.Task`1[System.String]	☰ Student Quiz	💡 Student Challenge

Prepare to guide students through creating an interactive game using Scratch. The game involves catching falling Easter eggs with a bowl. Students will learn how to create a new project, add a backdrop, add sprites, and use code to control the sprites. They will also learn how to create clones of sprites, detect when sprites touch each other, and keep score using variables. Encourage students to experiment and explore Scratch further after the lesson.

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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
1. Develop understanding and skills in creating and controlling sprites in Scratch. 2. Gain proficiency in using loops and conditionals in Scratch to create game dynamics. 3. Learn to use variables in Scratch for keeping score in a game. 4. Understand the concept of cloning in Scratch and how to use it in game development. 5. Enhance problem-solving and logical thinking skills through coding a game.	1. Develop a basic game using Scratch, incorporating elements such as sprites, backdrops, and clones. 2. Apply coding concepts such as loops, conditionals, and variables within the Scratch environment. 3. Implement user interaction in the game through keyboard inputs. 4. Utilise randomisation to create unpredictability in the game. 5. Design and implement a scoring system within the game.

Week 7

Lesson: Sound effects

● Advanced	60 mins	System.Threading.Tasks.Task`1[System.String]	Student Quiz	Student Challenge

Prepare to guide students through creating a lively baseball game in Scratch, complete with sound effects. They'll learn to add and code sprites, detect hits, and incorporate sound effects for hits, cheers, and misses. Ensure students understand how to use the Scratch interface, including the backdrop and sprite libraries, and how to add and modify code blocks. Encourage creativity and problem-solving as they explore different sound effects.

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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
1. Understand and apply the concept of sound effects in Scratch programming. 2. Create and modify a Scratch project, incorporating different sprites and backdrops. 3. Develop skills in coding for animation and sound effects using Scratch. 4. Apply conditional statements in Scratch to detect interactions between sprites. 5. Enhance creativity and problem-solving skills through the design and improvement of a game.	1. Develop a Scratch project incorporating sound effects. 2. Utilise the 'Baseball 2' backdrop and 'Batter' sprite effectively. 3. Code the 'Batter' sprite to animate a swing when the space bar is pressed. 4. Implement the 'Baseball' sprite and code it to move towards the 'Batter' sprite. 5. Code the 'Baseball' sprite to detect a hit and trigger appropriate sound effects. 6. Code the 'Baseball' sprite to trigger a different sound effect when a hit is missed.

Week 8

Lesson: Build Battles

● Advanced	🕒 60 mins	System.Threading.Tasks.Task`1[System.String]

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.