LESSON 16

Strand 1 challenge build

40 minutes | Strand 1: Introducing computer science | Outcome 1.5 | Outcome 1.6 | Outcome 1.7 | Outcome 1.8 | Outcome 1.9

WHAT THIS LESSON DOES

Combine planning, coding and testing skills into one small build. Choose a project, plan it with pseudo-code or a flow chart, build it in Scratch or MakeCode Arcade, test as you go, then share for feedback and improve it.

CURRICULUM OUTCOMES

Outcome	What the specification asks for
1.5	Develop algorithms using pseudo-code and flow charts.
1.6	Construct code to implement an algorithm.
1.7	Discuss and implement core features of a structured programming language: variables, operators, loops, decisions, assignment and modules.
1.8	Test code to find and manage errors.
1.9	Evaluate a program in groups of two or three.

WHAT STUDENTS SHOULD BE ABLE TO DO

- Plan a small build of your choice with pseudo-code or a flow chart, make it in Scratch or MakeCode Arcade, and test and debug it as you go (LOs 1.5, 1.6, 1.7, 1.8)
- Show your build to a group of two or three and act on one piece of feedback (LO 1.9)

BEFORE THE BELL

- Prepare a board display showing the challenge order, the done looks like criteria, the mini-collector scope exemplar, and the build milestones (plan by min 5, first run by min 15 or cut scope now, save by min 29).
- Optional: print or share a one-page starter scaffold (pre-labelled sprite and score variable) for support students.
- Student devices with Scratch and/or MakeCode Arcade ready; class way of saving projects available

Strand 1 challenge build

HOW THE LESSON RUNS

4 min	introduction	Start
2 min	content	The challenge
2 min	content	What done looks like, with an example
8 min	content	Plan it, then open Scratch
12 min	content	Build, test and save
6 min	content	Hands-on: Show, feedback, improve
3 min	content	Make sense
2 min	content	Reflect
1 min	summary	What you covered

TEACHING MOVES

- Display the order, success criteria, exemplar, and milestones on the board throughout the lesson.
- During the build activity, circulate the room to check that students have written plans before coding.
- Call a hard one-minute freeze after five minutes so plans are complete before blocks continue.
- At minute 15 call the mid-block checkpoint: first run done, or cut scope now to one goal and one decision.
- Before first runs, prompt a quick predict: what do you expect to see?
- Treat plan plus one working path as enough to enter the share round so LO 1.9 stays reachable.
- Default share groups to pairs; use threes only when numbers require it. Cap each run-through at about 60 seconds and protect a fixed 3 to 4 minute improve-and-re-save window on the board clock.
- Model how to give constructive feedback during the group sharing phase.

WHAT TO WATCH FOR

- Students begin coding without a plan: pause them and require the written plan first.
- Overly ambitious project scope: point at the mini-collector exemplar and remind students that one screen or one feature is sufficient. Enforce the minute-15 cut-scope checkpoint.
- Feedback not implemented: ensure each student selects and applies one specific improvement.

DIFFERENTIATION

- Support: Point students at the on-screen plan frame (Goal / Step 1 / Step 2 / Step 3 / Decision), the four starter ideas, and the mini-collector exemplar in the challenge step.
- Support: Offer a one-page starter scaffold with a pre-labelled sprite and score variable so they focus on one decision or feature.
- Extension: Encourage adding a second feature once the first version is working.
- Support: Pair students who need help with planning with a peer who has completed their plan.