LESSON 4

Flowcharts and Pseudocode

80 minutes (2 periods) | Foundations of Computer Science | Outcome 1.6 | Outcome 1.7 | Outcome 2.5 | Outcome 2.6 | Outcome 2.7 | Outcome 1.22 | Outcome 3.13

WHAT THIS LESSON DOES

In this lesson, you'll explore algorithms by learning to create flowcharts and pseudocode. Follow step-by-step tasks to design visual and text-based plans, then apply them to code a simple program using MakeCode for micro:bit.

CURRICULUM OUTCOMES

Outcome	What the specification asks for
1.6	explain the operation of a variety of algorithms
1.7	develop algorithms to implement chosen solutions
2.5	use pseudo code to outline the functionality of an algorithm
2.6	construct algorithms using appropriate sequences, selections/conditionals, loops and operators to solve a range of problems, to fulfil a specific requirement
2.7	implement algorithms using a programming language to solve a range of problems
1.22	read, write, test, and modify computer programs
3.13	develop a program that utilises digital and analogue inputs

WHAT STUDENTS SHOULD BE ABLE TO DO

- Understand the purpose and structure of flowcharts as visual tools for representing algorithms.
- Grasp the concept of pseudocode as a bridge between planning and coding.
- Develop skills in creating flowcharts to map out logical processes and decisions.
- Acquire the ability to write clear and structured pseudocode based on algorithmic designs.
- Apply flowchart and pseudocode planning to translate ideas into functional code.

BEFORE THE LESSON

- Open draw.io (www.drawio.com) yourself and confirm it loads through the school filter, students will need it twice.
- Check makecode.microbit.org loads and that the simulator runs, the temperature task works on the on-screen micro:bit even without physical boards.
- If using real micro:bits, have them cabled and confirm the temperature sensor reads in the simulator so students know what to expect.

Flowcharts and Pseudocode

HOW THE LESSON RUNS

- Introduction
- Recapping Flowcharts
- Practical Task: Create a Shopping Cart Flowchart
- Introduction to Pseudocode
- Practical Task: Write some Pseudocode
- Practical Task: Code a Temperature Reaction
- Practice Activity: Your Own Algorithm
- Wrap Up

TEACHING MOVES

- 2. Recapping Flowcharts. Ask students to name the four core shapes and what each does before you show them. Stress precise over vague using the flour example: a step must be actionable, not "add some".
- 3. Practical Task: Create a Shopping Cart Flowchart. Point out the decision diamonds explicitly: "add to cart?" and "continue shopping or checkout?" both need two labelled exit arrows. That is where students forget the yes/no branches.
- 4. Introduction to Pseudocode. Frame pseudocode as the bridge: readable English structured like code but with no syntax rules to get wrong. Show one IF-THEN-ELSE so they see the indentation before they write their own.
- 5. Practical Task: Write some Pseudocode. Have them work directly from their own flowchart, line by line. Each diamond becomes an IF, each loop-back arrow becomes a REPEAT. Catch anyone inventing new steps not on their chart.
- 6. Practical Task: Code a Temperature Reaction. Map the given pseudocode onto the MakeCode blocks aloud: forever loop, temperature read, if/else, icons. Let the simulator warm and cool the sensor so students see the icon switch.
- 7. Practice Activity: Your Own Algorithm. Insist their chosen process has a clear start, at least one decision and an end before they open draw.io. Approve the idea verbally first, it saves rework when the flowchart has no decision to draw.

WHAT TO WATCH FOR

- Students draw a decision diamond but only give it one exit arrow, so the flowchart has no real branch. Check each diamond has two labelled arrows before they move to pseudocode. Ask "what happens if the answer is no?".
- Students think pseudocode must follow strict rules like a real language and get stuck on exact wording. Remind them there is no syntax to fail. Show two different but equally valid pseudocode versions of the same step.
- In MakeCode, students expect the temperature to change on its own and think their code is broken when the icon stays the same. Show the simulator's temperature slider so they can drive the value above and below 20 themselves.

DIFFERENTIATION

- Emerging: Give a partly drawn shopping cart flowchart and ask them only to add the missing decision branches. Pair them with a student who has the MakeCode blocks placed so they focus on the logic, not hunting the toolbox.
- Developing: Ask them to explain out loud how each flowchart arrow became a pseudocode line before they code it. Have them predict what the micro:bit shows at 18 degrees and at 22 before running the simulator.
- Proficient: Challenge them to add a third temperature band, warm, cool and cold, extending both flowchart and pseudocode consistently. Ask them to describe how planning first with a flowchart would strengthen a Coursework Project write-up.