LESSON 1

# Introduction to Algorithms

40 minutes | Pattern Play Without Devices

## WHAT THIS LESSON DOES

In this lesson, you'll explore what an algorithm is through a fun, hands-on activity. Follow step-by-step instructions to program a robot in a sandwich-making task, learning the importance of clear, precise directions for successful results.

## WHAT STUDENTS SHOULD BE ABLE TO DO

- Understand the concept of an algorithm as a set of step-by-step instructions.
- Recognise the importance of precision and clarity in creating effective instructions.
- Develop skills in identifying and correcting errors through debugging.
- Apply logical thinking to create a clear and sequential set of instructions.
- Appreciate the impact of well-structured algorithms on achieving successful outcomes.

## BEFORE THE LESSON

- This is an unplugged lesson: no devices needed.
- Have real sandwich-making items ready: a plate, a butter knife, a loaf or slices of bread, butter and jam in a jar. Bring extra bread so a failed first attempt does not leave you short.
- Have a whiteboard or large sheet ready to write the corrected algorithm one step per line.
- Space for pupils to sit in a circle around the demonstration.

# Introduction to Algorithms

## HOW THE LESSON RUNS

- What is an Algorithm?
- Programming the Robot
- The First Attempt
- Debugging the Program
- Writing a Clear Algorithm
- The Second Attempt
- Success!

## TEACHING MOVES

- 1. What is an Algorithm? Anchor the word with an everyday example the class knows, like a recipe or LEGO instructions, then promise them they get to program a robot. Keep it to a minute so the demonstration carries the meaning.
- 2. Programming the Robot. Commit fully to the robot persona: monotone voice, stiff movements, zero common sense. The comedy is the teaching. Tell them plainly you will do exactly what they say and nothing more.
- 3. The First Attempt. Follow every instruction with cheerful literalness. Place the whole loaf on the plate, try to smear an unopened jar. Exaggerate, but never mock the pupil: the instruction failed, not the child.
- 4. Debugging the Program. Ask what went wrong and name it: this is debugging. Steer them away from blaming the idea and toward the wording of the steps. Let them spot the vague instructions themselves.
- 5. Writing a Clear Algorithm. Write each new step on its own line as pupils dictate it. Push for detail: not butter the bread but pick up one slice, open the tub, scoop butter with the knife. The layout itself teaches sequence.
- 6. The Second Attempt. Follow the rewritten steps exactly and let the sandwich come together. Pause visibly at each correct action so pupils feel their precise instructions working.
- 7. Success! Ask what made the second attempt work and guide them to name precise, ordered instructions. Connect it back to the word algorithm so the term sticks.

## WHAT TO WATCH FOR

- Pupils assume the robot will fill in obvious gaps, giving instructions like just make the sandwich or open it, expecting you to know what they mean. Do the literal, unexpected action every time. When they protest that you knew what they meant, remind them a computer only does exactly what it is told.
- Pupils think a step failed because the whole plan was wrong, rather than because one instruction was unclear or out of order. During debugging, keep the good idea and fix only the wording. Show that the same plan works once the steps are precise and in the right order.

**DIFFERENTIATION**

- Emerging: Give a pupil who is unsure a single small step to contribute, such as picking up one slice of bread, so they succeed with a manageable instruction.
- Developing: Ask the pupil to say the next step and then check it before you act: does that tell the robot everything, or would it get confused?
- Proficient: Challenge a finished pupil to spot the vaguest instruction still on the board and rewrite it more precisely, or to invent a step that would trip the robot up.