Mastering the Code: A Computing Challenge

1. An algorithm is a set of step-by-step instructions to complete a task. The order is important because computers follow instructions exactly; if the sequence is wrong, the final result will be incorrect. (2 marks)

Working: Definition of algorithm + explanation of sequence importance.

1 mark for definition, 1 mark for explaining that computers follow sequence strictly.

2. If sensor detects wall, then stop moving. (3 marks)

Look for: Correct use of 'If... then...' logic.

Good: If it hits a wall, then stop.

Better: If the sensor detects a wall, then the robot must stop moving.

Best: If sensor == wall, then stop robot movement.

What moves an answer up: Moving from simple language to precise logical syntax improves the answer.

3. Debugging is the process of finding and fixing errors in code. If the robot fails to turn left, I would check the code for the turn command, test the sensor, or add a print statement to see where the program stops. (4 marks)

Look for: Definition of debugging and a logical troubleshooting step.

Good: Debugging is fixing mistakes. I would check my code to see what is wrong.

Better: Debugging is finding and fixing errors. I would check if I wrote 'turn left' correctly in the code.

Best: Debugging is the process of identifying and removing errors. I would isolate the turn command and test it individually to see if the robot receives the signal.

What moves an answer up: Adding specific troubleshooting methods demonstrates a deeper understanding of the process.

4. A loop allows you to repeat a set of instructions a specific number of times. Instead of writing the movement code four times, you put the code inside a 'Repeat 4 times' loop, which makes the program shorter and easier to edit. (5 marks)

Look for: Explanation of efficiency and repetition.

Good: A loop repeats things so you don't have to write it again.

Better: A loop makes the code shorter because you can tell the robot to repeat the square path four times.

Best: A loop is more efficient because it reduces the number of lines of code. By nesting the movement instructions in a 'Repeat 4' loop, the program is cleaner and easier to debug.

What moves an answer up: Explaining the benefit of 'cleaner' and 'easier to edit' code shows higher-level thinking.

5. 1. Start. 2. Check sensor. 3. If hedgehog detected, then turn on light. 4. If no hedgehog, keep light off. 5. End. (6 marks)

Look for: Logical sequence showing input (sensor) and output (light).

Good: If the sensor sees a hedgehog, turn on the light.

Better: Start -> Check sensor -> If hedgehog detected, turn on light -> End.

Best: A flowchart starting with 'Input: Sensor', followed by a decision diamond 'Hedgehog detected?', with 'Yes' leading to 'Output: Flash light' and 'No' leading to 'Wait'.

What moves an answer up: Using formal flowchart terminology or clear step-by-step logic provides a professional answer.

6. A variable acts like a container to store a value. It is useful because the robot can add 1 to the 'books_collected' variable every time it picks up a book, allowing it to remember the total count. (5 marks)

Look for: Understanding that a variable stores changing data.

Good: It helps the robot count the books.

Better: A variable stores the number of books so the robot knows how many it has.

Best: A variable is a storage location for data. It is useful because it can be updated (incremented) each time a book is collected, allowing the robot to track the total count dynamically.

What moves an answer up: Using technical terms like 'incremented' and 'dynamically' shows mastery of the concept.

What a secure answer looks like

A secure answer demonstrates an understanding that code is a logical sequence. Students should use precise terminology like 'input', 'output', and 'loop' to explain their reasoning.

If your child is stuck

Ask your child to imagine they are the robot. What is the very first thing they need to do? What happens if they hit an obstacle? Writing it out as a list of 'human' instructions first often helps.

Common mistakes to watch for

Students often forget that computers are literal and need explicit instructions for every movement. Another common error is confusing a 'variable' with a fixed number; remind them that variables change during the program.