LESSON 43

Build a number-guessing game on the micro:bit, then read it in Python

40 minutes | Strand 3: Coding at the next level | Outcome 3.1 | Outcome 3.4

WHAT THIS LESSON DOES

Build a number-guessing game on the micro:bit using blocks, with variables for a secret number and your guess. Button A changes the guess on the LED screen and button B checks it, showing a happy face for a correct guess or an arrow pointing the way to go. Then switch the same project to its Python view and read your own code, matching each line back to the block you placed.

CURRICULUM OUTCOMES

Outcome	What the specification asks for
3.1	creatively design and write code for short programming tasks to demonstrate the use of operators for assignment, arithmetic, comparison, and Boolean combinations
3.4	describe program flow control, e.g. parallel or sequential flow of control (language dependent)

WHAT STUDENTS SHOULD BE ABLE TO DO

- Build a number-guessing game on the micro:bit using a variable, button events and a comparison (LO 3.1)
- Flip the same project to its Python view and recognise the variable and comparison you placed as blocks, with only the way they are written changing (LO 3.1, LO 3.4)

BEFORE THE BELL

- micro:bits paired at devices, or the on-screen simulator ready if hardware is short
- MakeCode open on the board at makecode.microbit.org so you can model the first variable and the button handlers
- Know where the Python view is before the lesson: students flip the same project to it near the end and read their own code back

Build a number-guessing game on the micro:bit, then read it in Python

HOW THE LESSON RUNS

4 min	introduction	Start: what we're building
3 min	content	Create a new micro:bit project
3 min	content	Make the secret number
3 min	content	Show your guess on the screen
3 min	content	Button A changes the guess
3 min	content	Wrap the guess back round to 1
3 min	content	Button B checks for a win
4 min	content	Tell the player higher or lower
3 min	content	Test your program
2 min	content	Read it in Python
3 min	activity	Your turn
3 min	content	Make sense: what worked, what broke
2 min	content	Reflect
1 min	summary	What you covered

TEACHING MOVES

- Make the first variable on the board and name it aloud as you go. Students who miss where variables live lose the rest of the build
- The wrap back to 1 is the step where most hands go up. Model it once with the number climbing past 9 so they see why it is needed
- When the project flips to Python, resist reading it top to bottom. Ask students to point at the block each line came from; recognising the comparison they placed is the outcome, not fluent Python
- Test after each button is wired rather than at the end, so a wrong comparison shows up while it is still one block to fix

WHAT TO WATCH FOR

- The button A stack dropped inside on start, where it never fires, so pressing A looks like it does nothing
- Higher and lower reversed, because the comparison reads the guess against the secret in the wrong order: walk one concrete pair of numbers through it
- The two set blocks placed after show number inside on start, so the screen shows 0 instead of the first guess
- Forgetting the wrap, so the guess climbs past 9 and off the display

DIFFERENTIATION

- Support: fix the secret number to a set value first so the game is testable, then swap in the random pick once the buttons work
- Extension: count the guesses and show the total when the player wins
- Extension: change the range to 1 to 20 and work out everything else that has to change with it