Level 3: Advanced (instructions only, no code)

Goal

Build Breakout by splitting responsibilities into 5 parts:

1. Pins/constants
2. Input (debounce + joystick repeat)
3. Game logic (state updates)
4. Render (drawing + score digits)
5. `.ino` glue (setup/loop scheduling)

Step-by-step instructions

1. Create a sketch folder and add:
    - `Pins.h` constants
    - `Input.*` for debounced inputs and joystick repeat
    - `Game.*` for state updates and rules
    - `Render.*` for drawing
    - `.ino` for setup/loop glue
2. In `Pins.h`:
    - Put every pin define (LED=2, buttons 3/4/9/10, joystick 5/6/7/8)
    - Put grid size (10x20), paddle settings, timing settings (frame, ball step, brick drop), repeat tuning.
3. In `Input`:
    - Implement a debounced button struct with edge detection.
    - Provide:
        - init for all pins
        - update once per frame
        - latch once per frame
        - function returning “serve pressed this frame”
        - function returning paddle step (-1/0/+1) with repeat while held.
4. In `Game`:
    - Store all game state (bricks array, ball, paddle, score, timers).
    - Provide:
        - `reset()` that fills first 8 brick rows and resets everything
        - `serve()` that unsticks ball
        - `movePaddle(dx)` that clamps and drags ball if stuck
        - `brickDropTick()` that shifts down and creates a new top row; if bottom occupied -> game over

        - `stepBallOnce()` that does bounces, brick hits, paddle hits, miss -> game over.

5. In `Render`:
    - Own the strip/pixelGrid/lcdPanel objects and initialise them.
    - Provide:
        - `updateScoreDigits(score)`
        - `wheelColor(pos)` for brick rows
        - `renderFrame()` that draws background, bricks, paddle, ball, or game-over screen.
6. In `Breakout.ino`:
    - In `setup()` call: render init, input init, game reset.
    - In `loop()`:
        - limit frame rate (every 16ms)
        - update inputs
        - if game over and serve pressed -> reset
        - else if serve pressed -> serve
        - every 15s do brick drop tick
        - apply joystick repeat steps to paddle
        - every ballStepMs do one ball step
        - render frame
        - latch inputs

After it works, extend it

- Add lives (don't end on first miss)
- Add multiple balls
- Add angled bounces (more than -1/0/+1)
- Add brick types (strong bricks, power-ups)
- Add levels (clear all bricks to speed up or change paddle size)