Solutions

2.1

- $20 + 5 - 1 = 24$ cycles ✓.

2.2

- A read-after-write hazard on R1 ✓.

2.3

- Routes an ALU result straight to the next ALU's inputs ✓. It fails when the producer is a load (data exists only after MEM): one stall ✓.

3.1

- Loads: 600; load-use stalls: $600 \times 0.4 = 240$ ✓.
- Bubbles: $2000 \times 0.10 = 200$ ✓.
- Total cycles $= 2000 + 240 + 200 + 4 = 2444$; CPI $= 2444/2000 = 1.222$ ✓.
- Unpipelined: 2000×900 ps $= 1.8$ ms; pipelined: 2444×250 ps $= 611$ µs – speedup $\approx 2.94\times$ ✓.
- As n grows the fill fades and the speedup tends to $900/250 = 3.6\times$ minus the stall tax ✓.

3.2

- `lw R1` feeding `add R3` is a load-use pair: forwarding cannot remove it – one stall ✓.
- `add R3` feeding `or R8` (ALU to ALU) forwards free; `sub R5` to `or` likewise; `lw`/`sub` are independent ✓.
- Extra cycles: 1 (the single load-use bubble) ✓.
- An independent instruction between them fills the bubble: zero extra cycles – scheduling is the cheaper fix ✓.