Solutions

2.1

- $C = \max(2, 5) + 1 = 6$ ✓.

2.2

- Yes – at different processes; equal clocks imply nothing about order ✓.

2.3

- Executing it more than once has the same effect as once ✓.

3.1

- m_1 sent at $C = 4$ (P0: $3 \to 4$) ✓.
- P1 receives: $C = \max(7, 4) + 1 = 8$ ✓.
- m_2 sent at $C = 9$ (P1: $8 \to 9$) ✓.
- P0 receives: $C = \max(4, 9) + 1 = 10$; m_3 sent at 11 ✓.
- Ordered: send(m_1) before receive(m_1), before send(m_2), before receive(m_2), before send(m_3) – a causal chain ✓.
- Concurrent: none in this trace – every event lies on one chain; concurrent events would share no arrow path ✓.
- Wall clocks skew without bound across machines: a later event can carry an earlier timestamp, so no guaranteed order ✓.

3.2

- Risk: the request lands, the reply is lost, the client retries – two deductions fire ✓.
- Design: idempotency key `id` plus a ledger – the first execution records `id` as done; the retry looks it up and returns the original result without re-executing ✓.
- A timeout proves nothing about execution – the client retries, and safely, because the operation is idempotent ✓.
- Rule: every operation a client may retry (any network call) must be idempotent ✓.