CERTLABZ CERTIFICATION BOARD

Certified Game Developer (CGMD)

Ten Sample Questions

These ten questions are written at the difficulty of the live exam. Six carry a figure, a data table, a configuration or a code listing; four are plain scenario questions, as on the live paper. Every one is a scenario a working game developer would meet, and none of them appears in the live question bank.

The live exam is multistage adaptive: how you score on one stage selects the difficulty of the next, so no two candidates sit the same paper. It is a performance-based exam: each form can carry up to 5 to 12 performance-based questions alongside the multiple-choice questions, which means you build, sequence or complete something rather than only choosing from a list. Answers and full explanations follow each question here; the live exam does not disclose item-level answers.

Question 1

Domain 1: Game Development Foundations

DATA

Sarah Miller compares common code patterns, with one benefit missing.

Pattern	Main benefit
Game loop	Steady frame updates
Component	(to choose)
State machine	Clear mode changes
Object pool	Reuse allocations

Table 1.1 Patterns and benefits.

What benefit does the component pattern give?

A. Locks the frame rate
B. Removes all state
C. Flexible entity mix
D. Bans reuse of data

Correct answer: C

The component pattern composes behaviour from small parts, giving a flexible mix of traits per entity without deep inheritance. The game loop drives updates, the state machine handles modes, and object pools reuse allocations.

Question 2 Domain 2: Game Design and Mechanics DATA

James Carter reviews difficulty tools, with one description missing.

Tool	What it does
Fixed curve	Same ramp for all
Rubber banding	Aids trailing player
Dynamic scaling	(to choose)
Checkpoints	Save safe points

Table 2.1 Difficulty tools.

What does dynamic scaling do?

A Locks one set level
B Deletes save points
C Hides the score bar
D Adapts to player skill

Correct answer: D
Dynamic scaling adapts challenge to measured player skill during play, unlike a fixed curve that treats everyone the same. Rubber banding helps a trailing player in races, and checkpoints store safe points to resume from.

Question 3 Domain 3: Graphics and Rendering DATA

Emma Wilson weighs two rendering paths for a scene with many lights.

Path	Best when
Forward	Few lights, MSAA
Deferred	(to choose)
Forward plus	Tiled light lists
Baked	Static lighting

Table 3.1 Rendering paths.

When does deferred shading fit best?

A One light, no depth
B Many dynamic lights
C No geometry buffer
D Text only rendering

Correct answer: B
Deferred shading writes a geometry buffer first and shades once per pixel, so it scales well with many dynamic lights. Forward suits few lights with MSAA, forward plus tiles lights, and baked lighting fits static scenes.

Question 4 Domain 4: Physics and Animation CONFIG

David Brown reviews the update loop that drives physics and drawing.

```
while running:
  input()
  accumulator += frameTime
  while accumulator >= dt:
    step_physics(dt)
    accumulator -= dt
  render(accumulator / dt)
```

Listing 4.1 The main loop.

Why step physics with a fixed dt?

A To skip rendering
B To disable input
C Stable, repeatable sim
D To drop all frames

Correct answer: C
A fixed dt keeps the physics step stable and repeatable regardless of frame rate, so the simulation behaves the same on fast and slow machines. The render call interpolates with the leftover accumulator to keep motion smooth.

Question 5 Domain 5: Gameplay Programming CONFIG

Olivia Davis reviews a shader snippet that tints a sprite by team.

```
uniform vec4 teamColor;
in vec2 uv;
out vec4 frag;
void main() {
  vec4 tex = texture(atlas, uv);
  frag = tex * teamColor;
}
```

Listing 5.1 Fragment shader.

What decides each pixel output colour?

A Vertex count alone
B Texture times tint

C The frame number

D Screen size only

Correct answer: B
The fragment output multiplies the sampled texture by the team colour uniform, so each pixel is the texture value tinted by team. Vertex count, frame number, and screen size play no part in this per-pixel result.

Question 6

Domain 6: Optimization, Testing and Release

CONFIG

Liam Anderson reviews the release build settings for a shipped title.

```
config: release
strip_symbols: true
texture_compression: on
log_level: warn
asserts: off
```

Listing 6.1 Release settings.

Why turn asserts off for release?

A To add more logs

B To grow the binary

C To slow the frame

D Cut runtime overhead

Correct answer: D
Asserts add runtime checks useful in development, so turning them off for release cuts overhead and speeds the shipped build. Stripping symbols and compressing textures further shrink size, while warn level logging keeps output lean.

Question 7

Domain 1: Game Development Foundations

A team hard codes every enemy behaviour into one giant class, and adding a new enemy type means editing many unrelated methods. What design best fixes this?

A One bigger class

B Copy the class

C Compose components

D Remove all classes

Correct answer: C
Composing behaviour from reusable components lets a new enemy pick the parts it needs without touching a monolithic class. Making the class bigger or copying it spreads the same coupling, and removing classes does not address the structure.

Question 8 Domain 2: Game Design and Mechanics

Playtests show new players quit at the first level because the tutorial dumps every mechanic at once. What design change best helps retention?

A. Add more text
B. Pace the teaching
C. Hide the controls
D. Raise difficulty

Correct answer: B
Pacing the teaching introduces mechanics gradually as they become relevant, so players learn without being overwhelmed at the start. Adding more text, hiding controls, or raising difficulty would deepen the confusion that drives early quits.

Question 9 Domain 4: Physics and Animation

At low frame rates fast objects pass through thin walls without a collision being detected. What technique best prevents this tunneling?

A. Bigger time steps
B. Disable gravity
C. Swept collision
D. Remove the walls

Correct answer: C
Swept, or continuous, collision tests the path an object travels between frames, so a fast mover cannot skip past a thin wall. Larger time steps make tunneling worse, and disabling gravity or removing walls does not solve detection.

Question 10 Domain 6: Optimization, Testing and Release

A game runs slowly, and the team starts rewriting random systems by guesswork with no measurement. What is the sound first step to optimize?

A. Rewrite everything
B. Profile the hot paths
C. Guess the cause
D. Buy new hardware

Correct answer: B
Profiling the hot paths shows where time is actually spent, so effort goes to the real bottleneck rather than guesswork. Rewriting everything or guessing wastes effort, and new hardware hides a problem players on modest machines still face.

CGMD-001. https://certlabz.com/certifications/certified-game-developer.html