QUESTION 1

• Main of purpose of Graphics Rendering Pipeline
  • Generates (or renders) a 2D image given a 3D scene
  • 3D scene $\rightarrow$ 2D image
QUESTION 2

• GPU = Graphics Processing Unit
  • Basically your video card
QUESTION 3

• Three stages of the Graphics Rendering Pipeline
  • 1) Application
  • 2) Geometry
  • 3) Rasterizer
QUESTION 4

• What does each stage output?
  • 1) Application $\rightarrow$ outputs geometry (points, lines, triangles, etc.)
  • 2) Geometry $\rightarrow$ outputs transformed and projected vertices
  • 3) Rasterizer $\rightarrow$ outputs final 2D image
QUESTION 5

• Measure rendering speed in:
  • **FPS** (frames per second)
  • **Hertz** (1/second)
QUESTION 6

• 3 most basic geometric primitives:
  • Points (1 vertex)
  • Lines (2 vertices)
  • Triangles (3 vertices)
QUESTION 7

• Why triangles?
  • Simple
  • Fits in single plane
  • Can define any polygon in terms of triangles
QUESTION 8

• Model coordinates (relative to model) → MODEL TRANSFORM → World coordinates
QUESTION 9

• After **view transform** is applied, camera is pointing in the direction of the **NEGATIVE Z axis** (OpenGL)
QUESTION 10

• Two programmable parts of the pipeline:
  • Vertex shading
  • Pixel/fragment shading
QUESTION 11

- After the *projection transform* is applied, the extents of the view volume are \((-1, -1, -1)\) to \((1, 1, 1)\)
  - ...in OpenGL
QUESTION 12

- Two common kinds of projections:
  - Orthographic (parallel)
  - Perspective
QUESTION 13

• Z coordinates are **NOT** changed during *screen mapping*
QUESTION 14

• **Fragment** = data necessary to shade/color a pixel due to a primitive covering or partially covering that pixel
  • Data can include color, depth, texture coordinates, normal, etc.
  • Values are interpolated from primitive’s vertices
  • Can have multiple fragments per pixel
  • Final pixel color will either be one of the fragments (i.e., z-buffer chooses nearest one) or combination of fragments (e.g., alpha blending)

• Since I didn’t very clearly define this term before the quiz, this question is now free.
QUESTION 15

• Major problem with the Z-buffer algorithm ➔ transparent objects
QUESTION 16

• *Computer-graphics application programming interfaces (CG API)*
  → interface between programming language and hardware
QUESTION 17 (BONUS)

• View volume for *perspective projection* = view frustum