ETGG1803 Final Exam Review Questions

If your section is…	your final is on:
02 (MW 9:00am – 11:15am)	Wednesday May 4, 8:00am – 9:50am
01 (MW 12:00 – 2:15pm)	Monday May 2, 12:00 – 1:50pm
03 (TR 3:00 – 5:15pm)	Thursday May 5, 2:00 – 2:50pm

I should be around for last-minute questions:

- Monday (5/2/2016) from ~ 8:15am – 11am
- Tuesday (5/3/2016) from ~8:15am – 11am

Some advice:

- Do:
  - Get plenty of sleep and eat before the test.
  - Break up your studying (don't just cram the night before)
  - Make a **cheat sheet** (one sheet of 8.5" x 11" paper [typed or hand-written], single-sided)
    - It's OK if you want to share copies of your cheat-sheet with others.
- Don't:
  - Just study the answers to the quizzes and practice final.
  - Panic. Whatever happens, there are solutions…talk to me after the final (in person / email) and we can figure something out.

Class outline

- Section01 (Coordinate Spaces and Vectors)
  - 2d polar <-> Euclidean conversions
  - Vector notation and terminology (head, tail, magnitude)
  - Connection between vectors and points.
  - Python OOP basics and "class hooks": __init__, __str__, __len__, __getitem__, __setitem__, __eq__
  - hasattr and isinstance functions
  - Raising exceptions
- Section02 (Vector Operations)
  - [for each operation, a symbolic, numeric, and *graphical* interpretation]
  - Negation
  - Vector-scalar multiplication
  - Magnitude
  - Addition / Subtraction
  - Normalization
  - Additional "class hooks": __add__, __sub__, __mul__, __rmul__, __truediv__, __rtruediv__, __neg__, etc.
- Section03 (Basic Vector Physics)
  - Integrating position from velocity.
  - Newton-Euler-1 integration (the kind we used in the lab)
    - The connection between acceleration, velocity, and dt (w/ variable frame rate)
  - Gravity Accleration
  - Rotating and pointing in pygame.
  - Basic inheritance
  - Basic friction (both methods: opposing force and time-based velocity-scaling)

- Section04 (Dot and Cross Product)
    - The two numeric methods and why they are equivalent.
    - V-dot-v == magnitudeSquared
    - Finding the angle between two vectors
    - Shotgun problem (including the linear interpolation of damage
    - Acute-Right-Obtuse classification
    - Projections
    - Link-beamos problem
    - Cross Product
        - Main application
        - Magnitude of the result (and connection to area)
- Section05 (RayTracer Part I)
    - For each: know how we define them and the "main" formula used with them
        - Ray
        - Plane
        - Sphere
    - I (usually) don't ask you to completely replicate any hit detection algorithms, but I probably will ask you for parts of it…
    - Main idea / algorithm
    - Constructing camera coordinate system (camX, camY, camZ, and camPos)
    - Calculating view plane width & height
    - Calculating view plane origin
    - Calculating the position of a pixel on the view plane (based on a pygame pixel)
    - Finding the closest object hit.
- Section06 (Ray Tracer II)
    - material / light / perceived color connection
    - be able to describe the 3 "passes" of lighting (ambient, diffuse, specular)
    - be able to show how to calculate the diffuse, ambient, specular color of an object
    - be able to describe the ray-tracer shadow algorithm.
- Section07 (Matrices)
    - Matrix terminology
        - dimension
        - notation for indicating a single element or a row or column (0-based)
        - square-ness
    - Identity matrix
    - Matrix-Vector connection
        - Right and Left-handed difference
    - Transposing
    - Matrix * Matrix
    - Matrix * Vector and Vector * Matrix
        - The connection to __mul__ and __rmul__ methods in python
        - The connection with left and right-handed vectors.
    - Connection between matrices and systems of linear equations
- Section08 (Transformations)
    - Terminology

        - Basis vectors
        - Using matrices to transform vectors
        - local vs world basis vectors
    - Analyzing a mystery matrix
        - Point-method (pick some points, observe how the mesh is transformed)
        - By analyzing rows (in a LHS) – the basis vectors
    - Derivation and matrix for
        - Mirror / Reflection
        - Rotation around cardinal axes (worldX, worldY, worldZ)
        - Scale
        - Shear (in 1 or 2 dimensions)
        - Translation
    - Homogeneous space
        - As used in translation
        - Specifying points & vectors
    - Concatenating transforms
        - Using a "pipeline"
        - By matrix multiplying
        - Why method 2 is better than method 1 (in terms of number of adds / multiplies)
    - Scene Graphs
        - Purpose (esp. hierarchical transforms)
        - Specifying (e.g. I might give you a hierarchy and ask you to sketch out the tree)
        - Implementing matrix concatenation / tree-traversal.
- Section 09 ("Finished Rasterizer")
    - Whatever we get to in lecture☺