A.A. Local coordinates describe position and direction relative to the origin of a body. Global coordinates describe position and direction relative to the origin of the entire scene.

A.B. Points are transformed from local to global by `isoTransformPoint` (rotation then translation) and from global to local by `isoUntransformPoint` (untranslation then unrotation). For directions the corresponding functions are `isoRotateVector` (rotation only) and `isoUnrotateVector` (unrotation only).

A.C. Recursive ray casts should be done in global coordinates, because they involve interactions among bodies. They could be done in the local coordinates of a particular body, but that code would be less efficient and harder to understand.

A.D. Lighting calculations can be done equally well in local or global coordinates. You just have to pick one or the other and use it consistently.

B.A. Lambertian diffuse reflection involves $\vec{d}_{\text{normal}}$ (the unit outward-pointing normal vector), $\vec{d}_{\text{light}}$ (the unit vector pointing toward the light), $\vec{c}_{\text{diff}}$ (the diffuse surface color), and $\vec{c}_{\text{light}}$ (the light color).

B.B. The diffuse contribution is

$$i_{\text{diff}}\ \vec{c}_{\text{light}} * \vec{c}_{\text{diff}},$$

where $*$ denotes modulation (component-wise multiplication) and

$$i_{\text{diff}} = \max\left(0, \vec{d}_{\text{normal}} \cdot \vec{d}_{\text{light}}\right).$$

B.C. Phong specular reflection involves $\vec{d}_{\text{normal}}$ (the unit outward-pointing normal vector), $\vec{d}_{\text{light}}$ (the unit vector pointing toward the light), $\vec{d}_{\text{camera}}$ (the unit vector pointing toward the camera), $\vec{c}_{\text{spec}}$ (the specular surface color), shininess, and $\vec{c}_{\text{light}}$ (the light color).

B.D. Essentially, the specular contribution is

$$i_{\text{spec}}\ \vec{c}_{\text{light}} * \vec{c}_{\text{spec}},$$

where

$$i_{\text{spec}} = \max\left(0, \vec{d}_{\text{refl}} \cdot \vec{d}_{\text{light}}\right)^{\text{shininess}},$$

and $\vec{d}_{\text{refl}}$ is $\vec{d}_{\text{camera}}$ reflected across $\vec{d}_{\text{normal}}$. However, we treat one case specially: When $i_{\text{diff}} = 0$, we declare that $i_{\text{spec}} = 0$ too, to avoid what one might call "shallow backside reflection".

C. Shader programs were introduced into OpenGL in version 2.0.

D. [I'll omit pictures, although you probably should not.] For the sake of argument, assume that air and glass have indices of refraction of 1.0 and 1.5, respectively. Let $\vec{d}_{\text{normal}}$ be the outward-pointing normal, $\vec{d}_{\text{inc}}$ the incidence direction, and $\vec{d}_{\text{refr}}$ the refraction direction. Let θ_{inc} be the angle between $\vec{d}_{\text{inc}}$ and $\vec{d}_{\text{normal}}$. Let $\vec{d}_{\text{refr}}$ be the angle between $\vec{d}_{\text{refr}}$ and $\vec{d}_{\text{normal}}$.

When a ray enters the glass body, we have

$$\frac{\sin\theta_{\text{refr}}}{\sin\theta_{\text{inc}}} = \frac{1.0}{1.5},$$

because the incident ray is in air and the refracted ray is in glass. Also, θ_{inc} and θ_{refr} are both acute.

When a ray exits the glass body, we have

$$\frac{\sin\theta_{\text{refr}}}{\sin\theta_{\text{inc}}} = \frac{1.5}{1.0},$$

because the incident ray is in glass and the refracted ray is in air. Also, θ_{inc} and θ_{refr} are both obtuse (assuming that total internal reflection isn't happening).

E.A. The final color at a pixel is the sum of ambient lighting, zero or more diffuse-specular contributions from lights, a recursive mirror effect, and a recursive transmission effect.

E.B. The body geometry determines $\vec{x}$ (the position of the point being colored) and $\vec{d}_{\text{normal}}$ (the unit outward-pointing normal vector). It's reasonable to throw the texture coordinates in here too, because typically the texture-mapping scheme depends on the body geometry.

E.C. The "material", out of which the body is made at the point $\vec{x}$, determines five quantities: $\vec{c}_{\text{diff}}$ (the diffuse surface color), $\vec{c}_{\text{spec}}$ (the specular surface color), the shininess, $\vec{c}_{\text{tran}}$ (the transmissive color), and the index of refraction.

E.D. Each light determines $\vec{c}_{\text{light}}$ (the light color at a given world point $\vec{x}$), $\vec{d}_{\text{light}}$ (the unit direction from $\vec{x}$ to the light), and the distance from $\vec{x}$ to the light (which is useful for shadow rays). Indeed, these are exactly the fields of our `lightResponse` data structure.

F.A. [Although this question is dressed up in ray tracing, it actually concerns Day 03 of the course: interpolation.] Notice that $\vec{a}$ corresponds to $(p, q) = (0, 0)$, $\vec{b}$ corresponds to $(1, 0)$, and $\vec{c}$ corresponds to $(0, 1)$. Moreover, the side of the triangle between $\vec{a}$ and $\vec{b}$ is where $0 \le p \le 1$ and $q = 0$. The side of the triangle between $\vec{a}$ and $\vec{c}$ is where $p = 0$ and $0 \le q \le 1$. The side of the triangle between $\vec{b}$ and $\vec{c}$ is where $p + q = 1$. In short, the ray intersects the triangle if and only if all three of these conditions are satisfied: $p \ge 0$, $q \ge 0$, and $p + q \le 1$.

F.B. Compute the 8-dimensional array $\vec{x} = \vec{a} + p(\vec{b} - \vec{a}) + q(\vec{c} - \vec{a})$. Pull out its last three components (indices 5, 6, 7) as a 3-dimensional vector. Normalize this vector to length 1.

F.C. Compute the 8-dimensional array $\vec{x} = \vec{a} + p(\vec{b} - \vec{a}) + q(\vec{c} - \vec{a})$. Pull out its middle two components (indices 3, 4) as a 2-dimensional vector.

F.D. [This question is intended to be the most difficult one on the exam.] Yes, the texture coordinates are perspective-correct. We are interpolating them in local coordinates, which are linearly related to world coordinates. That's what we want. Why did we have to do perspective correction in rasterization? Because we were interpolating in screen coordinates, which are not linearly related to world coordinates when the projection is perspective.