Test #1

Due Feb 11 at 11:59pm **Points** 100 **Questions** 40
Available Feb 7 at 12:01am - Feb 11 at 11:59pm **Time Limit** 60 Minutes

Instructions

The test will open Wednesday, February 7, 12:01 AM PST and will close Sunday, February 11, 11:59 PM PST

Test Information:

This is a multiple choice test cast as a Canvas "Quiz".
There are 40 questions, worth 2.5 points each.
You have 60 minutes to complete it. Once you start, you need to finish. Canvas does not allow you to pause, leave, then come back and resume.
The test is open notes and closed friends.
Clearly, I cannot stop you from accessing information on the Internet. *However*, the test has been written against our class notes. If you miss a particular question, any protest of the form "But somethingsomething.com said that..." will be ignored.

Attempt History

	Attempt	Time	Score
LATEST	Attempt 1	46 minutes	100 out of 100

Correct answers are hidden.

Score for this quiz: **100** out of 100
Submitted Feb 9 at 11:45am
This attempt took 46 minutes.

Question 1

2.5 / 2.5 pts

How is noise *Turbulence* created?

- ○ By flipping all noise values more than the midpoint down to the bottom half
- ○ By taking the square of the noise values
- ○ By taking the square root of the noise values
- [✓ By flipping all noise values less than the midpoint up to the top half]

Question 2

2.5 / 2.5 pts

All of these are ways that *cube-mapped refraction* is inferior to actual refraction *except:* (I.e., which one of these is false?)

- ○ There can be no inter-refractions between objects in the scene
- ○ Spatial relationships are baked-in at the moment the photos are taken
- [✓ The Index of Refraction cannot be taken into account]
- ○ There can be no refraction out the back surface

Question 3

2.5 / 2.5 pts

A particular texture image you are using has its s and t values range from 0. to 1.

You now double the pixel resolution of the texture image. Your code must now treat the new image's *s* value as ranging from:

- 0. to .5
- [✓ 0. to 1.]
- 0. to 2.
- -1. to 1.

Question 4

2.5 / 2.5 pts

Why is the *gl_Vertex* variable a vec4, not a vec3? I.e., what is the fourth element for?

- The 4th element is a mask of bit values
- The 4th element is just there to pad it out to 4 floats and is not actually used
- [✓ The 4th element needs to be the homogeneous w]
- The 4th element is a flag to tell the rasterizer what to do

Question 5

2.5 / 2.5 pts

When you use a vertex shader instead of the fixed-function pipeline, the vertex shader takes over all of the following functions *except*:

- [✓ Depth-buffer handling]
- Multiplying by the modelview matrix

- ○ Sending a position to the rasterizer
- ○ Multiplying by the projection matrix

Question 6

2.5 / 2.5 pts

In bump-mapping, if you have two tangent vectors, you compute the surface normal using:

- [✓ The cross product]
- ○ Gram-Schmidt orthogonalization
- ○ The dot product
- ○ The vector inverse

Question 7

2.5 / 2.5 pts

When obtaining an RGBA value from a Cube Map texture, you provide the texture unit number and a:

- [✓ vec3]
- ○ float
- ○ vec4
- ○ vec2

Question 8

2.5 / 2.5 pts

One flaw in Cube-Mapped refraction is:

- [✓ You cannot refract out the back of the object]
- The object you view must have an equation that we know
- The object you view must be a flat plane
- You cannot specify an Index of Refraction

Question 9

2.5 / 2.5 pts

A "Cube Map Texture" is different from our usual image textures because:

- [✓ It has six images inside it instead of one]
- It is no different than the usual image textures
- It has four images inside it instead of one
- It is looked up with a single float instead of a vec2

Question 10

2.5 / 2.5 pts

What does it take to get an A in this course?

- 97%

- [✓ 1080 points]
- 100 points
- 93%

Question 11

2.5 / 2.5 pts

What is the most common purpose of the gl_NormalMatrix?

- [✓ It is used to transform the surface normals so they rotate with the object]
- It is used to transform both the surface normals and the object's coordinates
- It is used to transform the surface normals independently of the object
- It is used to transform both the surface normals and the texture coordinates

Question 12

2.5 / 2.5 pts

For what we have been doing this quarter, where is the GLSL shader compiler located?

- [✓ In the graphics driver]
- In the operating system

- In GPU memory

Question 13

2.5 / 2.5 pts

When bump-mapping, if we have the derivative *dzdy*, we turn that into a tangent vector like this:

- [✓ vec3(0., 1., dzdy)]
- vec3(1., 0., dzdy)
- vec3(0., dzdy, 1.)
- vec3(dzdy, 0., 1.)

Question 14

2.5 / 2.5 pts

In Project #3 (the rock-dropped-in-a-pond), the displacement and bump mapping happened in:

- Both in the fragment shader
- Both in the vertex shader
- Bump in the vertex shader, displacement in the fragment shader
- [✓ Displacement in the vertex shader, bump in the fragment shader]

Question 15

2.5 / 2.5 pts

In the noise equation:

```
vec4 nv = texture( Noise3, A*vMC );

float n = nv.r + nv.g + nv.b + nv.a;

n = n - 2.;

n = B*n;
```

what are the elements of nv?

- ○ The frequency
- [✓ The noise octaves]
- ○ The amplitude
- ○ The noise wavelets

Question 16

2.5 / 2.5 pts

You saw a demonstration of the use of a Displacement Texture. In what shader(s) can a Displacement Texture be used?

- ○ Fragment shader only
- [✓ Vertex shader only]
- ○ Either the vertex or the fragment shader

Question 17

2.5 / 2.5 pts

When you use a fragment shader instead of the fixed-function pipeline, the fragment shader takes over all of the following functions *except*:

- [✓ Controlling the refresh rate of the monitor]
- Color calculations
- Per-fragment lighting
- Texture lookup

Question 18

2.5 / 2.5 pts

We have been using texture functions of the form *texture(texUnit, index)* function. If the index is a *vec3*, what does this function return?

- vec3
- vec2
- [✓ vec4]
- float

Question 19

2.5 / 2.5 pts

In Project #1, an ellipse's boundary is at d = 1. In that project, what function did you call to create a blending factor to make the boundary blurry?

- ○ float t = step(1.+uTol, 1.-uTol, d);
- ○ float t = smoothstep(-uTol, uTol, d);
- ○ float t = step(1., d);
- [✓ float t = smoothstep(1.-uTol, 1.+uTol, d);]

Question 20

2.5 / 2.5 pts

These are all good uses for homogeneous (vec4) coordinates *except* (i.e., which of these is *not* a good use):

- ○ Light sources at infinity
- [✓ Creating colored spot lights in the scene]
- ○ Perspective projection
- ○ XYZ points at infinity

Question 21

2.5 / 2.5 pts

In per-fragment lighting, which of these is interpolated through the rasterizer?

- ○ The RGB color
- ○ The light intensity
- ○ The vector from the eye to the light

[✓ The surface normal]

Question 22

2.5 / 2.5 pts

In the Disco Ball demo, assume that you see 128 lit-up bright spots on the object. How many light sources were used in the shader to get those bright spots?

- 128
- 64
- [✓ 1]
- 2

Question 23

2.5 / 2.5 pts

The user-written *SmoothPulse( )* function, which we used to make stripes, combined two *smoothstep( )* functions. Therefore, the total number of edge locations that we needed to specify was:

- 2
- 3
- 1
- [✓ 4]

Question 24

2.5 / 2.5 pts

In the *mix( )* function, the two quantities being mixed can be of type:

- vec3-only
- vec2-only
- float-only
- [✓ Just about anything]

Question 25

2.5 / 2.5 pts

What is an advantage in using *Bump-mapping* instead of Displacement-mapping?

- It makes it look like the surface is not flat
- It can make it look like lighting is taking place
- It looks more realistic when you fly close to it
- [✓ It uses less polygons]

Question 26

2.5 / 2.5 pts

A particular noise function seeds its random number sequence with a different number every day. Why is this a bad idea?

- Because the noise value would get larger as the day went on
- Because the noise would no longer have the coherency property
- Because the noise value would get smaller as the day went on
- [✓ Because the noise would no longer have the repeatability property]

Question 27

2.5 / 2.5 pts

When using cube mapping for refraction, the *best* way to obtain the refraction direction is

- The built-in reflect( ) function
- [✓ The build-in refract( ) function]
- The built-in transmit( ) function
- A calculus equation that you derive yourself

Question 28

2.5 / 2.5 pts

The ability for the value of noise at one location to be nearly the same as the value of noise at a slightly different location is called:

- [✓ Coherency]
- Repeatability
- Understandability

- ○ Consistency

Question 29

2.5 / 2.5 pts

In practice, we don't use *Positional Noise*. Why?

- [✓ With the ideosyncracies of random numbers, we might accidentally create points with little variation]
- ○ The curves that get created are not coherent
- ○ The curves that get created are not continuous at the points
- ○ The curves that get created are not repeatable

Question 30

2.5 / 2.5 pts

If we see stripes painted on objects, and the stripes appear to live on the object, with the stripes traveling with the object, then in what coordinate system are the stripes being created?

- [✓ Model coordinates]
- ○ Normalized Device Coordinates (NDC)
- ○ Eye/world coordinates
- ○ Screen coordinates

Question 31

2.5 / 2.5 pts

What is an advantage in using Displacement-mapping instead of Bump-mapping?

- ○ It makes it look like the surface is not flat
- [✓ It looks more realistic when you fly close to it]
- ○ It uses less polygons
- ○ It can make it look like lighting is taking place

Question 32

2.5 / 2.5 pts

In the noise equation:

```
vec4 nv = texture( Noise3, A*vMC );
float n = nv.r + nv.g + nv.b + nv.a;
n = n - 2.;
n = B*n;
```

what is B?

- ○ The noise wavelets
- [✓ The amplitude]
- ○ The frequency
- ○ The noise octaves

Question 33

2.5 / 2.5 pts

If we cannot easily vertex-morph a cow to a dinosaur, then why can we easily morph a cow into a sphere?

- ○ Because OBJ files don't allow this
- ○ Because the dino OBJ file is improperly configured
- ○ Because the cow OBJ file is improperly configured
- [✓ Because we know the equation of a sphere without having to read it in]

Question 34

2.5 / 2.5 pts

The ripples bump-map example is very much like the terrain bump map example because:

- ○ Both involve reading heights from a texture file
- ○ Both are emulating a smooth surface
- [✓ Both involve a height sticking up from a flat plane]
- ○ Both are emulating a rough surface

Question 35

2.5 / 2.5 pts

What is the maximum number of Bonus Days you get to use across *all* projects?

- 4
- 3
- [✓ 5]
- 6

Question 36

2.5 / 2.5 pts

The general idea behind the Dome Projection is to use a nonlinear transform to:

- [✓ Create a view where you can see the left side and right side of the scene simultaneously]
- Turn all circles into a series of straight lines
- Turn all straight lines into circles from the left side to the right side of the scene
- Turn all straight lines into ellipses from the left side to the right side of the scene

Question 37

2.5 / 2.5 pts

In Project #3 (the rock-dropped-in-a-pond), the surface being drawn needed to have:

- 64 vertices, 8 in each direction
- Just 4 vertices, because bump-mapping only needs a quad
- 9 vertices, so that you would have 4 quads with a common point in the middle
- [✓ Lots of vertices, so there were enough to displace smoothly]

Question 38

2.5 / 2.5 pts

What is one way you could fix the following piece of incorrect GLSL vertex shader code?

gl_Position = gl_ModelProjectionMatrix * gl_Vertex;

- gl_Position.xyz = gl_ModelViewProjectionMatrix * gl_Vertex.xyz;
- [✓ gl_Position = gl_ModelViewProjectionMatrix * gl_Vertex;]
- gl_Position.xyz = gl_ModelProjectionMatrix * gl_Vertex.xyz;
- gl_Position = gl_ViewProjectionMatrix * gl_Vertex;

Question 39

2.5 / 2.5 pts

Joe Graphics keys off (x,y,z) to make a color pattern on a sphere. Jane Graphics keys off (s,t). The difference will be:

- There will be no difference – a pattern is a pattern

- ○ Jane’s pattern will slice through the sphere, Joe’s will go around the sphere
- [✓ Joe’s pattern will slice through the sphere, Jane’s will go around the sphere]
- ○ Joe’s way is illegal in GLSL

Question 40

2.5 / 2.5 pts

In Projects #1 and #2, you computed a variable called *numins*. What did *numins* tell you?

- ○ What the current fragment’s x location was
- ○ What the current fragment’s s coordinate was
- [✓ How many ellipse boxes the current fragment was over in the s direction]
- ○ How many stripes the current fragment was over in the x direction

Quiz Score: **100** out of 100