Student Number: **Name:**

Write down answers in-between questions. Please answer using short sentences.

The back of each page can be used for practice, but DO NOT write down the answer on the back.

Be sure to write your student number and name on each page.

1. (6 pts) What mathematical conditions must a square matrix satisfy to be a "rotation matrix"?

2. (4 pts) A 3D position **p** is rotated by Rotation A and then by Rotation B, w.r.t. body frame. Rotation A is represented by $\mathbf{R}_a$ (in rotation matrix) or $\mathbf{q}_a$ (in unit quaternion). Rotation B is represented by $\mathbf{R}_b$ (in rotation matrix) or $\mathbf{q}_b$ (in unit quaternion). Write down the rotated position **p'** 1) using the rotation matrices and 2) using the unit quaternions. For an answer for 2), just use **p** as if it were a pure imaginary quaternion representing a 3D position.

3. (5 pts) Why is back-face culling performed in NDC (Normalized Device Coordinates) space rather than view space?

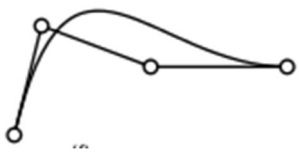
4. (7 pts) Below are four curves and their "control points/polygon." Some of the control points are the Bezier control points for the curve drawn with it (i.e., the curve in this case is a Bezier curve); the others are not. Assume that none of the control points overlap or are repeated.

   1) Indicate which cases illustrate Bezier control points / curves and which are not.

   2) For non-Bezier cases you choose, explain why they are not Bezier control points / curves.

a. 

b. 

c. 

d. 

Student Number: **Name:**

5. (6 pts) The following code is for loading a texture image and creating a texture object. When executing this code, instead of the object that should originally be drawn, only a black window was displayed. Modify only one line or add one line in the following code to ensure that the object is properly drawn. Assume that i) there are no issues with opening the image, ii) except for the code provided below, the rest of the program code all works correctly, iii) there are no calls to texture-related functions in the remaining parts of the program code.

```
def main():
    ...
    texture1 = glGenTextures(1)
    glBindTexture(GL_TEXTURE_2D, texture1)
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER,
GL_NEAREST_MIPMAP_LINEAR)
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER,
GL_LINEAR)

    try:
        img = Image.open('./image.jpg')
        img = img.transpose(Image.FLIP_TOP_BOTTOM)
        glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, img.width,
img.height, 0, GL_RGB, GL_UNSIGNED_BYTE, img.tobytes())
        img.close()
```

6. (6 pts) The following figure shows the directions that influence lighting at a point on the surface of an object. Using the symbols described below, express the diffuse and specular colors $\mathbf{I}_d$ and $\mathbf{I}_s$ at this point, determined by the diffuse and specular component of the Phong illumination model. Note that **L, N, V,** and **R** are unit vectors. Use * for element-wise multiplication and · for inner product (dot product).

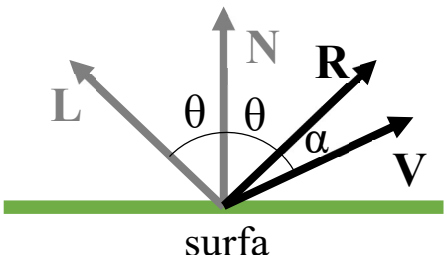
$\mathbf{l}_d$: light diffuse color, $\mathbf{m}_d$: material diffuse color,

$\mathbf{l}_s$: light specular color, $\mathbf{m}_s$: material specular color,

L : light direction, **N** : surface normal,

V : view direction, **R** : reflection direction of light

n : shininess coefficient



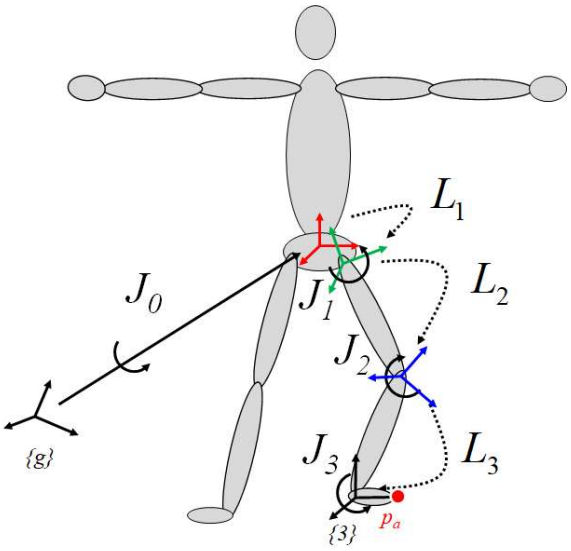
Diffuse color $\mathbf{I}_d$ =

Specular color $\mathbf{I}_s$ =

7. (6 pts) You want to render a spherical object using the Phong illumination model. Assume the shape and normal vectors of the sphere, and the light and view directions are fixed. 1) What factor affects the size of the specular highlight area on this object? 2) Explain how the size of the highlight area changes as that factor changes.

Student Number: **Name:**

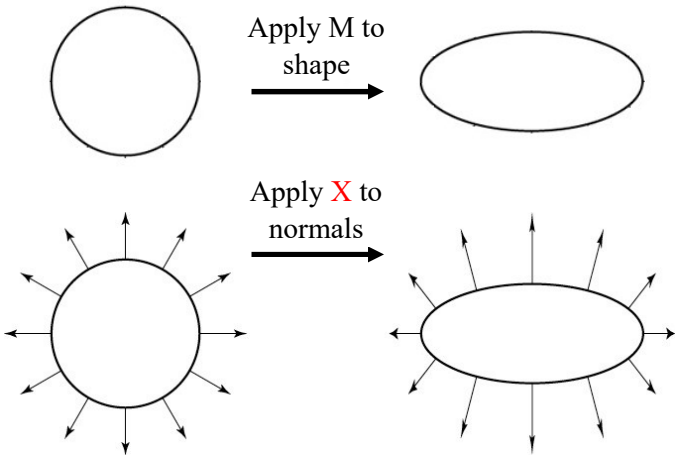
8. (4 pts) What is the position of the point $\mathbf{p}_a$ w.r.t. global frame {g}, $\mathbf{p}_a^{\{g\}}$? Use the joint and link transformations shown in the right figure and the position $\mathbf{p}_a^{\{3\}}$ which is the position of the point $\mathbf{p}_a$ w.r.t. the frame {3}.



9. (8 pts) Fill the following table with yes or no to indicate which type of spline has which properties.

	typically has C2 continuity?	allows local control?	interpolates all or some of the control points?
Bezier spline			
Catmull-Rom spline			
Natural cubic spline			
B-spline			

10. (7 pts) If a set of points on a surface is transformed by an affine transformation **M**, 1) what transformation **X** should be applied to the normals at those points? 2) Describe the process of calculating this **X**. Refer to the diagram below to answer.



11. (5 pts) Choose **ALL** techniques that are used to simulate high-frequency surface by creating the "illusion" of depth without modifying the actual geometry of a 3D model.

1) Normal mapping 2) Bump mapping 3) Displacement mapping 4) Texture mapping

Student Number: **Name:**

12. (6 pts) Choose **ALL false(incorrect)** statement about glBufferData() and glBufferSubData().

    1) glBufferData() allocate GPU memory for and copy vertex data to the currently bound VBO.

    2) By passing 0 as the third argument of glBufferData(), you can only copy vertex data to the currently bound VBO without allocating memory for it.

    3) glBufferSubData() only allocates memory for the currently bound VBO and does not copy vertex data to it.

    4) glBufferSubData() is used to bind a buffer object to a specific buffer target.

13. (5 pts) Choose **ALL false(incorrect)** about local and global illumination models.

    1) Local illumination models only consider light that comes directly from light sources, while global illumination models simulate indirect light interactions between object surfaces.

    2) Local illumination models simulate all light interactions within a limited area, while global illumination models consider the entire scene.

    3) Local illumination models are usually faster to compute than global illumination models.

    4) Local illumination models only consider diffuse reflection, while global illumination models consider both diffuse and specular reflection.

14. (6 pts) Below is the pseudo code for z-buffer (depth buffer) algorithm. Fill in the blanks (a), (b), and (c). You have to use functions already used in the code and variables already defined in the code. Additionally, you can use a function call `read_depth_buffer(x,y)` to get the current recorded depth value at (x,y).

```
allocate depth_buffer;

for each pixel (x,y)
   write_frame_buffer(x,y,backgrnd_color);
   write_depth_buffer(x,y,farPlane_depth);

   for each polygon
      for each pixel (x,y) in polygon
         new_color = polygon's color at (x,y);
         new_depth = polygon's z-value at (x,y);
         if (___________(a)_____________)
             ___________(b)_____________;
             ___________(c)_____________;
```

15. (3 pts) Match each term about the visibility problem on the left with the corresponding description on the right.

    1) Clipping

    2) Hidden surface removal

    3) Back-face culling

    a) Removing back-facing primitives

    b) Removing primitives outside the viewing frustum

    c) Removing primitives occluded by other objects closer to the camera

Student Number: **Name:**

16. (6 pts) There are two objects, object1 and object2, and vertex array objects vao1 and vao2 to draw them. vao1 refers to a vertex buffer object that stores vertex positions and vertex colors, while vao2 refers to a vertex buffer object that stores vertex positions and vertex normals. Fill in the blanks (a)~(f) using the shader source code variable names **shader1**, **shader2**, **shader3**, **shader4**, and the shader program variable names **program1**, **program2** below to enable proper rendering using vao1 and vao2. Note that the load_shader() function takes the vertex shader code as the first argument and the fragment shader code as the second argument and returns a shader program object. '...' indicates omitted code, assuming that the relevant parts are appropriately written.

```
shader1 = '''
#version 330 core
layout (location = 0) in vec3
vin_pos;
layout (location = 1) in vec3
vin_normal;
out vec3 vout_surface_pos;
out vec3 vout_normal;
...
void main()
{
    gl_Position = ...
    vout_surface_pos = ...
    vout_normal = ...
}
'''

shader2 = '''
#version 330 core
in vec4 vout_color;
out vec4 FragColor;
void main()
{
    FragColor = vout_color;
}
'''
```

```
shader3 = '''
#version 330 core
in vec3 vout_surface_pos;
in vec3 vout_normal;
out vec4 FragColor;
...
void main()
{
    FragColor = ...
}
'''

shader4 = '''
#version 330 core
layout (location = 0) in vec3
vin_pos;
layout (location = 1) in vec3
vin_color;
out vec4 vout_color;
...
void main()
{
    gl_Position = ...
    vout_color = ...
}
'''
```

```
def main():
    ...
    program1 = load_shader(__(a)__, __(b)__)
    program2 = load_shader(__(c)__, __(d)__)
    ...
    while not glfwWindowShouldClose(window):
        ...
        glUseProgram(__(e)__)
        glBindVertexArray(vao1)
        glDrawArrays(...)

        glUseProgram(__(f)__)
        glBindVertexArray(vao2)
        glDrawArrays(...)
```

Student Number: **Name:**

17. (10 pts) The following text is the hierarchy section of a bvh file which has 6 joints. If each joint's position w.r.t. global frame at the rest pose (when all motion values are zeros) is given as the following table, fill in the blanks (a), (b), (c), (d), and (e).

joint	global position
J0	(0.0, 0.0, 0.0)
J1	(0.5, 0.5, 0.0)
J2	(0.75, 1.0, 0.0)
J3	(0.25, 1.0, 0.0)
J4	(-0.5, 0.5, 0.0)
J5	(-0.5, 1.0, 0.0)

```
HIERARCHY
ROOT J0
{
	OFFSET 0.00   0.00   0.00
	CHANNELS 6 Xposition Yposition Zposition Zrotation Xrotation
Yrotation
	JOINT J1
	{
		OFFSET  ____(a)____
		CHANNELS 3 Zrotation Xrotation Yrotation
		JOINT J2
		{
			OFFSET  ____(b)____
			CHANNELS 3 Zrotation Xrotation Yrotation
			End Site
			{
				OFFSET  0.00   0.50   0.00
			}
		}
		JOINT J3
		{
			OFFSET  ____(c)____
			CHANNELS 3 Zrotation Xrotation Yrotation
			End Site
			{
				OFFSET  0.00   0.50   0.00
			}
		}
	}
	JOINT J4
	{
		OFFSET  ____(d)____
		CHANNELS 3 Zrotation Xrotation Yrotation
		JOINT J5
		{
			OFFSET  ____(e)____
			CHANNELS 3 Zrotation Xrotation Yrotation
			End Site
			{
				OFFSET  0.00   0.50   0.00
			}
		}
	}
}
```