THIS PAPER IS NOT TO BE REMOVED FROM THE EXAMINATION HALLS

UNIVERSITY OF LONDON **CO3355 ZA**

BSc Examination

COMPUTING AND INFORMATION SYSTEMS AND CREATIVE COMPUTING

Advanced Graphics and Animation

Date and Time: Thursday 19 May 2016, 10.00 – 12.15

Duration: 2 hours 15 minutes

There are FIVE questions on this paper. Candidates should answer **THREE** questions. All questions carry equal marks and full marks can be obtained for complete answers to **THREE** questions. The marks for each part of a question are indicated at the end of the part in [.] brackets.

Only your first **THREE** answers, in the order that they appear in your answer book, will be marked.

There are 75 marks available on this paper.

A hand held calculator may be used when answering questions on this paper but it must not be pre-programmed or able to display graphics, text or algebraic equations. The make and type of machine must be stated clearly on the front cover of the answer book.

UL16/0095

1.

(a) What are the six major elements of a graphics system? Name and describe each one of them in a sentence.

[6]

(b) Assume a vector $[x, y, z]^T$ in 3D space. Based on matrix operations, and working on homogeneous coordinates, produce a single matrix that can be used to perform scaling followed by translation so that the transformed vector becomes: $[s_x x+t_x, s_y y+t_y, s_z z+t_z]^T$.

[5]

(c)

(i) Construct a transformation matrix to rotate a 3D vector 30° about the y-axis.

[4]

(ii) Compute the coordinates of $[2, 3, 2]^T$ after rotation by 30° about the y-axis.

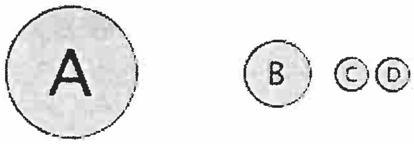
[2]

(d)

(i) Briefly describe what is meant by the term *transformation stack* and explain how it can assist in constructing hierarchical models of objects.

[3]

(ii) Consider the example of a simple two-dimensional solar system, consisting of four planets, named A, B, C and D respectively, as depicted below:



Suppose that you want to create an animation where A remains static at the center of the frame, B rotates around A, while both C and D rotate around B, but in opposite directions. Provide pseudocode for an iterative function –such as Processing's `draw()` – that draws the planets and animates them. Briefly explain how you make use of the transformation stack.

[5]

2.

(a) Name two ways of generating 3D geometric models. What is meant by the term *wireframe* in computer graphics?

[5]

(b) Describe what is meant by

(i) *Viewpoint Coordinate System*

(ii) *Screen Coordinate System.*

[4]

(c) Explain what the term *culling* refers to and describe how it works, considering the example of a cube.

[6]

(d)

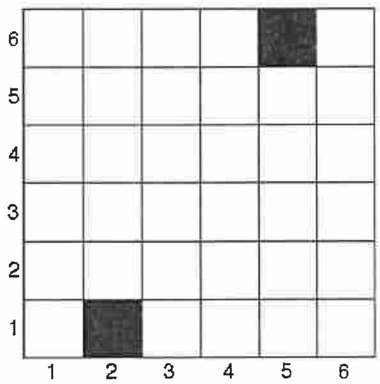
(i) Describe the steps of the Bresenham line drawing algorithm for the case of a line with a positive slope of less than 45 degrees.

[4]

(ii) How would the former be modified in order to treat a line with a slope between 45 and 90 degrees?

[2]

(e) Using the Bresenham line drawing algorithm, calculate the pixels to draw a line between points (2,1) and (5,6), as per the illustration below. Show your working.



[4]

3.

(a) Describe two characteristics of triangles that make them more adequate for graphics calculations than other polygons.

[4]

(b) Create a simple diagram naming the stages involved in a typical graphics pipeline, going from geometric models to images. Where in this pipeline are vertex and fragment shader programs positioned?

[4]

(c) Name the types of light considered by the Phong reflection model.

[2]

(d) What is image-based lighting and how does it compare to environment mapping? Explain how the process is related to global illumination and high dynamic range imaging and describe the steps involved.

[8]

(e) The following listing is the source code of a GLSL shader. Identify the type of shader and explain what it does and how it does it, line by line.

```
varying vec4 vertColor;
varying vec3 transformedNormal;
varying vec3 lightDir;
void main() {
	vec3 direction = normalize(lightDir);
	vec3 normal = normalize(transformedNormal);
	float light = max(0.0, dot(direction, normal));
	gl_FragColor = vec4(light, light, light, 1) * vertColor;
}
```

[7]

4.

(a) How is texturing an improvement over plain shading?

[4]

(b) What is a *texel* and how is it different from a pixel?

[3]

(c)

(i) Describe the technique of *bump mapping*, explaining how it affects the object geometry and the number of polygons.

[4]

(ii) Describe the heightfield method for bump representation.

[3]

(iii) How is bump mapping different from *displacement mapping*? Taking into account the complexity of each technique and the quality of representation it achieves, state which of the two you would choose in order to

- add realism to a textured representation of an orange.
- animate waves of water.

Justify your answer.

[6]

(d) What is described with the term *environment mapping*? Explain the steps of the process.

[5]

5.

(a) Describe what *collision shapes* are and how they differ from *graphics shapes*.

[4]

(b) Explain the functionality of the following types of collision shapes:

(i) *simple (primitive) shapes*
(ii) *compound shapes*
(iii) *meshes*

[6]

(c) What is restitution and how does it affect the behaviour of objects after a collision?

[6]

(d) What does the following code do? Explain how it does this line by line.

[9]

```
public void setup() {
  size(1280, 720, P3D);
  physics = new BPhysics();
  physics.world.setGravity(new Vector3f(0, 500, 0));
  BSphere s = new BSphere(this, 2, 0, 0, 0, 20);
  s.rigidBody.setDamping(0, 0);
  Vector3f pos = new Vector3f(640, 0, 0);
  BObject r = new BObject(this, 100, s, pos, true);
  physics.addBody(r);
}

public void draw() {
  lights();
  physics.update();
  physics.display();
}
```

END OF PAPER