**Task 1**

This diamond has 6 vertices which are labelled. The origin is in the centre of the shape. I chose to place it here as the shape is symmetrical meaning all the vertices are an equal distance from the origin.

T

The coordinates of the shape:

T – (0.0, 1.5, 0.0)

1 – (1.0, 0.0, 1.0)

2 – (-1.0, 0.0, 1.0)

3 – (-1.0, 0.0, -1.0)

4 – (1.0, 0.0, -1.0)

B – (0.0, -1.5, 0.0)

4

3

1

2

Bv

This diagram shows each of the triangles in the design. The black numbers refer to triangles in the front of the shape with the red triangles representing the ones behind.

4

3

2

7

1

8

5

6

These triangles represent each of the faces, labelled as above, with the correct vertices and normals (n)

3

0.0, 1.5, 0.0

1.0, 0.0, -1.0

2

0.0, 1.5, 0.0

0.0, 1.5, 0.0

1

n: 0, -0.5547, -0.83205

-1.0, 0.0, -1.0

-1.0, 0.0, -1.0

-1.0, 0.0, 1.0

-1.0, 0.0, 1.0

1.0, 0.0, 1.0

n: 0, -0.5547, 0.83205

n: -0.83205, 0.5547, 0

-1.0, 0.0, -1.0

-1.0, 0.0, 1.0

6

0.0, -1.5, 0.0

1.0, 0.0, 1.0

-1.0, 0.0, 1.0

4

0.0, 1.5, 0.0

1.0, 0.0, 1.0

5

1.0, 0.0, -1.0

n: -0.83205, -0.5547, 0

n: 0, -0.5547, 0.83205

n: 0.83205, 0.5547, 0

0.0, -1.5, 0.0

1.0, 0.0, -1.0

-1.0, 0.0, -1.0

8

0.0, -1.5, 0.0

1.0, 0.0, 1.0

1.0, 0.0, -1.0

0.0, -1.5, 0.0

7

n: 0.83205, -0.5547, 0

n: 0, -0.5547, -0.83205

To calculate the normals stated above, I labelled the three vectors’ as 1, 2 and 3. I then created two vectors (1 - 2) and (1 - 3). I then calculated the cross product of these two vectors to give us a vector of the correct direction. I then finally converted this to the unit vector to get the final normal value.

**Task 2**

glGenVertexArrays()

glBindVertexArray()

glGenBuffers()

glBindBuffer()

#Create 3 arrays to hold each vertex attribute

PositionArray[]

TextureArray[]

NormalArray[]

#Create list of values to hold vertices and each one e.g.

Vertex0 = glm::vec3(0, 1, 1);

PositionArray.push\_back(Vertex0);

Texture0 = glm::vec2(0, 1);

TextureArray.push\_back(Texture0);

Normal0 = glm::vec3(0, 0, 1);

NormalArray.push\_back(Normal0);

#Repeat for all arrays of position, texture and normals

glBufferData(PositionArray)

glBufferData(TextureArray)

glBufferData(NormalArray)

Stride = sizeof(PositionArray) + sizeof(TextureArray) + sizeof(NormalArray)

#Position Array

glEnableVertexAttribArray(0)

glVertexAttribPointer(At position 0)

#Texture Array

glEnableVertexAttribArray(1)

glEnableAttribPointer(At position sizeof(PositionArray))

#Normal Array

glEnableVertexAttribArray(2)

glEnableAttribPointer(At position sizeof(PositionArray) + sizeof(TextureArray))

**Task 5**

When calling perspective() the aspect references to the relative size of the user’s view with the fovy deciding the angle of the view that the user will see. Znear and Zfar refer to the closest and furthest distance from the camera that objects will be rendered at.

**Task 7**

File: diamond.obj:

----------------------------

# (comment lines)

v 0.0 1.5 0.0 Vertex Positions

v 1.0 0.0 1.0

v -1.0 0.0 1.0

v 1.0 0.0 -1.0

v -1.0 0.0 -1.0

v 0.0 -1.5 0.0

.

.

.

vn 0.0 0.5547 0.83205 Vertex Normals

vn 0.0 0.5547 -0.83205

vn -0.83205 0.5547 0.0

vn 0.83205 0.5547 0.0

vn 0.0 0.5547 -0.83205

vn -0.83205 0.5547 0.0

.

.

.

vt 0.0 0.0 Texture coordinates

vt 0.0 1.0

vt 1.0 0.0

.

.

.

f 1/1/1 2/2/1 3/3/1 Faces (indices of v/vt/vn)

f 6/1/2 3/2/2 2/3/2

f 1/1/3 3/2/3 5/3/3

f 6/1/4 5/2/4 3/3/4

f 1/1/2 5/2/2 4/3/2

f 6/1/6 4/2/6 5/3/6

f 1/1/2 5/2/2 4/3/2

f 6/1/2 5/2/2 4/3/2

----------------------------

File: diamond.obj:

-------------------------

magic.png Name of associated texture image

-------------------------