# CPSC 314
# Assignment 4 (7%)

Due Friday Nov 2, 2018

Answer the questions in the spaces provided on the question sheets. If you run out of space for an answer, use separate pages and staple them to your assignment.

Name: _____

Student Number: _____

| | |
|---|---|
| Question 1 | / 14 |
| Question 2 | / 17 |
| Question 3 | / 6 |
| TOTAL | / 37 |

**DO NOT HAND IN THIS VERSION. IT WILL NOT BE GRADED.**

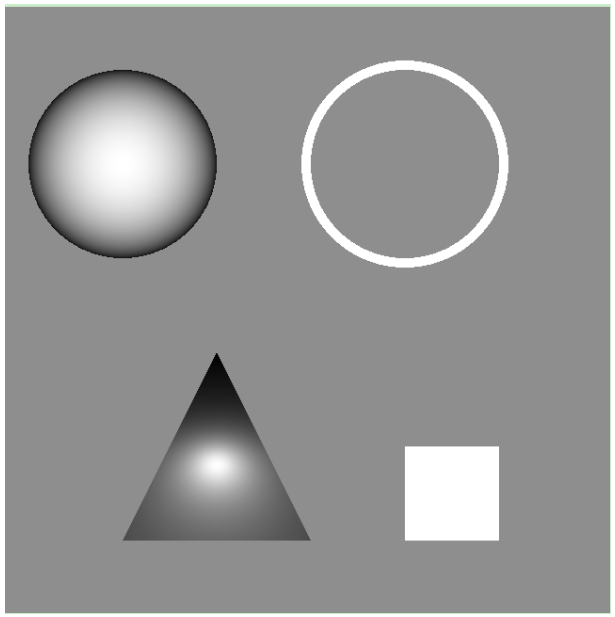
**PLEASE COLLECT A UNIQUELY-NUMBERED COPY IN CLASS, WHICH WE NEED FOR GRADING.**

1. Scan Conversion using Implicit Functions

The learning goals for this question are: (a) to be able to define and implement implicit functions, as applied to scan conversion; (b) to gain additional experience writing shaders.

Download, install, and run the template code for A4, available on the course website. You will be making edits only to the fragment shader included as part of `a4.html`.

**Important**: Because the shaders are compiled when `a4.html` is loaded by your browser, you will need to look at the console window for compile errors. The compile errors can sometimes be cryptic, and thus it is important that you develop your shader code very incrementally. A common error is something as simple as missing a semicolon. We suggest to always keep a copy of your last working code on hand. The final image should look something like the image shown below.



(a) (3 points) Complete the code for `insideSquare(x,y)` by defining implicit functions for all four edges, such that a positive result for each means that a point is 'inside' and should be rendered.

(b) (3 points) Complete the code for `insideCircle(x,y)` by defining two implicit functions, such that a positive result for each means that a point is 'inside' and should be rendered.

(c) (3 points) Complete the code for `insideSphere(x,y)` by defining an implicit functions for being inside the circle centered at `(xc,yc)` and with radius `rad`. In addition, we'll give the circle the appearance of a sphere. Compute a surface normal that is in the direction `(dx,dy,dz)`, where `dz` can be computed knowing the radius and `dx, dy`. This can then be normalized using `N = normalize(N);` Lastly, compute the intensity `i` using the dot product of N and L, and use this for the r,g,b components of the assigned pixel color.

(d) (2 points) Lastly, complete the code for `insideTriangle(x,y)`. First, complete the code for the function `scaledImplicitABC()`. Also take the time to understand the function `getBarycentric()`.

(e) (3 points) Render the triangle in white using the computed barycentric coordinates. Then change the code to render the triangle in a way that visualizes the alpha parameter. Now comment that out, and change the code to render the triangle using interpolated normals and a simple diffuse lighting model. To do this, first compute an interpolated normal using barycentric interpolation of the normals at the vertices. Note that the GLSL language supports scalar-vector multiplication and vector addition, i.e., `P = 0.2*P1 + 0.8*P2;` is a valid computation, where P,P1,P2 are of type `vec3`, for example. Then compute the intensity as the dot product of N and L, and use this to render the r,g,b components of the fragment.

Submit your code using:
`handin cs314 a4`.

2. Barycentric coordinates and interpolation

   A triangle has device coordinates $P_1(10, 20)$, $P_2(10, 60)$, $P_3(50, 60)$. We wish to be able to interpolate a value $v$ for an arbitrary point $P(x, y)$ in the triangle, given the values at the vertices.

   (a) (1 point) Sketch the triangle and the point $P(30, 50)$.

   (b) (3 points) Write an explicit line equation for each of the three edges, i.e., $y = f(x)$ or $x = f(y)$.

   (c) (3 points) Rearrange the terms of each of these equations to trivially transform these into implicit line equations where $F(x, y) = 0$ for points on the line. Label these with the vertices that they pass through, i.e., $F_{12}(x, y)$, $F_{23}(x, y)$, and $F_{13}(x, y)$.

   (d) (3 points) Give scaled implicit line equations, e.g., $\hat{F_{12}}(x, y)$, for each edge such that $\hat{F}(x, y) = 1$ at the third vertex, i.e., the vertex not on the line segment. Relate these expressions to the barycentric coordinates $(\alpha, \beta, \gamma)$ of a point $P(x, y)$, where $P = \alpha P_1 + \beta P_2 + \gamma P_3$.

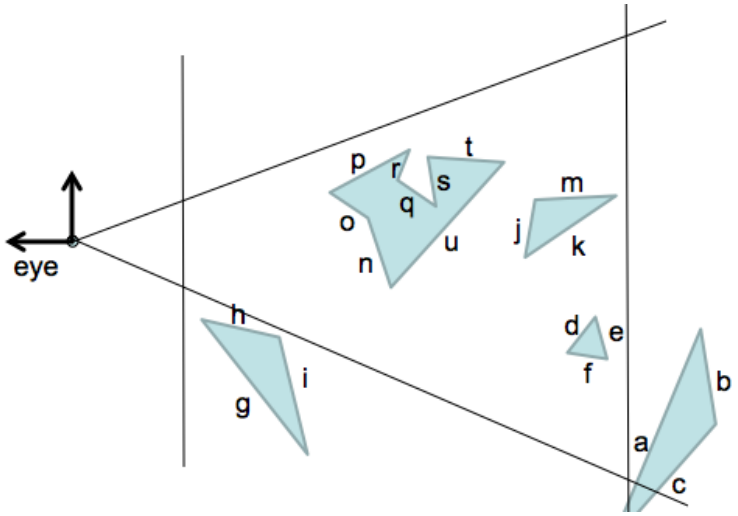
(e) (1 point) Verify that $\alpha + \beta + \gamma = 1$ holds true for any point $P(x, y)$ by using your expressions.

(f) (2 points) On your diagram above, label each of the vertices with their $(\alpha, \beta, \gamma)$ values.

(g) (2 points) Sketch and label lines corresponding to $\alpha = 0, \alpha = 0.5, \alpha = 1$ in the above diagram.

(h) (2 points) Compute the barycentric coordinates for $P(30, 50)$. Then use them to interpolate $v$ for that point, given the following known values for $v$ at the vertices: $v_1 = 10, v_2 = 20, v_3 = 60$.

3. Visibility and Culling
Consider the scene below, shown as a side-view of VCS, together with view frustum. Assume that all the objects shown are solid, and that the labelled lines represent polygonal faces of the objects.




(a) (2 points) List, in alphabetical order, the polygons that would be culled by view frustum culling.

(b) (2 points) List, in alphabetical order, the polygons that would be culled by back-face culling. Note: consider both types of culling independently of each other.

(c) (2 points) After view-frustum culling and back-face culling, list in alphabetical order the remaining faces that would be completely removed by z-buffer tests.