Foundations of Computer Graphics
Final Exam

Instructions:

- Write your **name** here :
- Questions are not arranged by difficulty.
- The points for the questions may not be proportional to the their difficulty or the time required to answer them.
- This is an open book exam but you are not allowed to discuss with others.
- Please submit your solutions on NYU classes.

All the best!

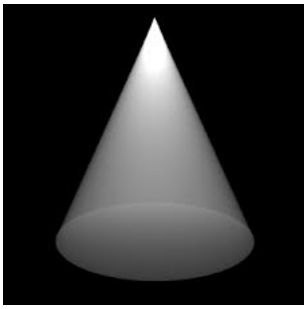
Problem 1 (35 points)**.**
Give short answers to the following questions.

1. What is the 4×4 homogeneous transformation matrix for translation by $(1, 2, 3)$ followed by counterclockwise rotation by 90 degrees about the x-axis?
   Please write the exact matrix and **not** *an expression.*

2. What is the equation of the quadratic Bézier curve through the control points $\mathbf{p}_0 = (0, 0)$, $\mathbf{p}_1 = (1, 0)$ and $\mathbf{p}_2 = (1, 1)$?

3. Recall that in WebGL, we use a depth buffer to do hidden surface removal. Consider the following alternative for hidden surface removal: we draw the triangles in a mesh in the decreasing order of the distances of their centroids from the camera location. Note that no depth buffer is used. Whenever a triangle is drawn, it overwrites all pixels it affects. Give an example where this alternative algorithm fails.

4. Suppose that we want to implement a "spot light" instead of a "point light" source that we have considered in class. A spot light located at a point p is similar to a "point light" source located at p except that it only emits light within a cone with apex at p as shown in the picture below. How would you modify the Phong lighting equations for such light sources? *You can assume that the apex, the axis and the central angle of the cone are given.*



5. Recall that in projection normalization, we map the viewing volume to the canonical cube $[-1, 1]^3$. It is tempting to make the viewing volume very large, for instance by arbitrarily increasing the distance to the far plane. Why is this **not** a good idea?

6. Consider a plane h defined by the equation $3x + 4y - z = 2$ in three dimensions. Write down a matrix Q s.t. for any point $p = [x, y, z, 1]^T$ in homogeneous coordinates, the square of the distance of p from h is $p^T\ Q\ p$. *Recall mesh simplification using Quadric Error Metric (QEM).*

7. How are bounding volumes hierarchies used to speed up ray tracing?

Problem 2 (15 points)**.**
An axis parallel cuboid is a cuboid whose sides are parallel to the coordinate axes. Such a cuboid can be expressed as $[x_1, x_2] \times [y_1, y_2] \times [z_1, z_2]$ which represents the set $\{(x, y, z) : x \in [x_1, x_2], y \in [y_1, y_2]$ and $z \in [z_1, z_2]\}$. Let $R(e, \vec{d})$ represent the ray $e + t\vec{d}$, $t \geq 0$ emanating from the point e and going in the direction $\vec{d}$.

Please give brief but precise answers to the following questions ***not*** *just the broad idea.*

1. Describe a procedure to compute the intersection of the ray $R(e, \vec{d})$ with the cuboid $C = [-1, 1] \times [-1, 1] \times [-1, 1]$. Your procedure should return an interval $[t_1, t_2]$ where $t_1, t_2 \geq 0$ s.t. the point $e + t\vec{d}$ lies in C if and only if $t \in [t_1, t_2]$.

   Explain clearly how t_1 and t_2 are computed using e and $\vec{d}$ as parameters. If the ray does not intersect the cuboid C, you can set $t_2 < t_1$ for that the range $[t_1, t_2]$ is empty.

2. Given a procedure for the previous, how would you use it to compute the intersection of a ray $R(e, \vec{d})$ with an arbitrary cuboid $C = [x_1, x_2] \times [y_1, y_2] \times [z_1, z_2]$?

3. Given a procedure for the above, how would you use it to find the intersection of a ray $R(\mathbf{e}, \vec{d})$ with the $C_1 \setminus C_2$ where C_1 and C_2 are two axis parallel cuboids and $C_1 \setminus C_2$ represents the set of points which lies in C_1 but not C_2.

Problem 3 (10 points)**.**
We would like to animate a square as shown in the attached file `Square.gif` (open the file in a browser). We can assume that initially, i.e., at time $t = 0$, the square has the left bottom corner at $(0, 0)$ and the top right corner at $(1, 1)$. At time $t = 1$, the bottom left corner of the square is at $(3, 0)$ and the top right corner is at $(4, 1)$. In between, it moves as shown in `Square.gif`. Let $S(t)$ denote the square at time t. Construct a 3×3 homogeneous transformation matrix $M(t)$ such that $S(t) = M(t)S(0)$.

Optional. If it helps you, you can experiment with the code in the folder "RollingSquare". You only need to change the matrix M in the vertex shader. You don't need to change (or even read) the rest of the code.