Aubrey Church

Due: Nov 14

Input format: Num(1,1), Num(1,2), … Num (1,9)

Num(2,1), Num(2,2), … Num (2,9)

Num(K,1), Num(K,2), … Num(K,9)

**Part A**

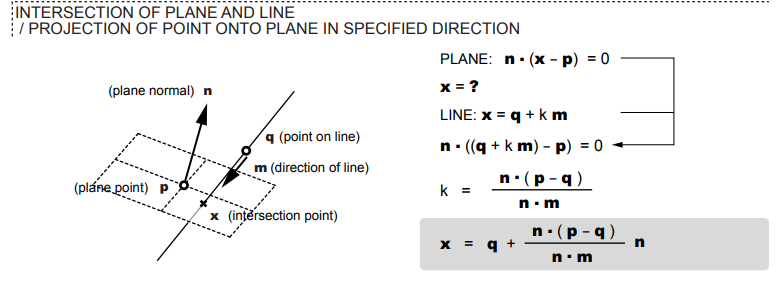
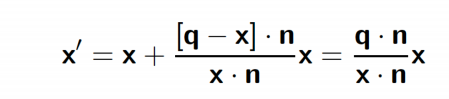
The first line of the input corresponds to the eye location point (Num11, Num12, and Num13() and the light direction (Num14, Num15, and Num16) while ignoring the last three numbers, while the rest of the lines correspond to the three points of the vertices of the triangular planar facets. Use the notation and the orientation in Chapter 8 and Chapter 8.6 where each of three numbers define p, q, r, respectively, in that order.

19.00 45.00 91.00 = x1

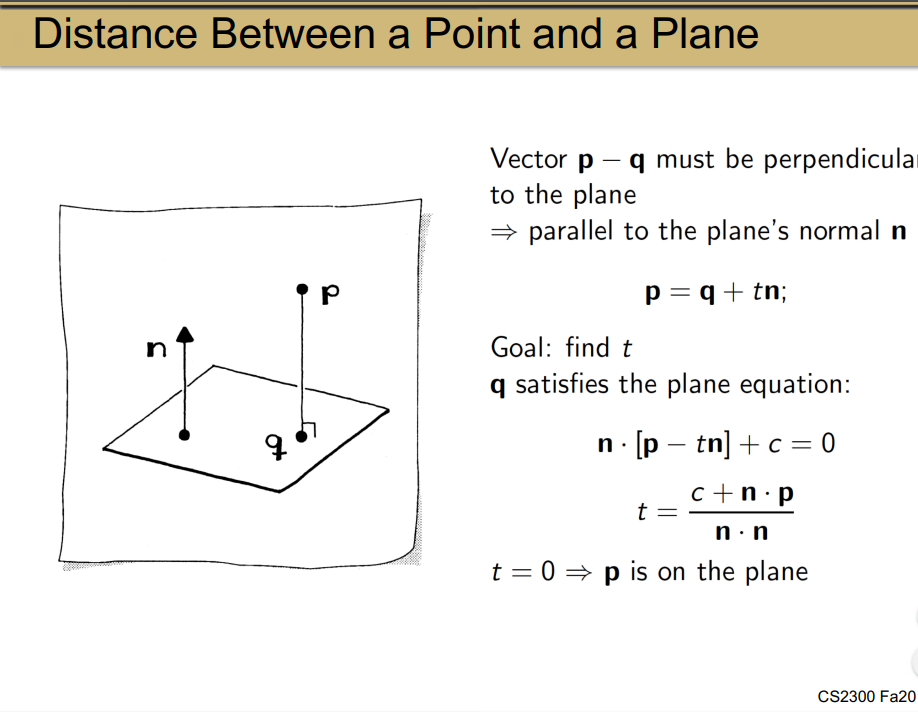
* Line 1: Culling
  + Identify weather the triangle planar facet is front-facing vs back facing
    - Need vertex normal
    - v=p-q w=q-r
    - normal n = v cross w / ||v cross w||
    - n Dot (eye - centroid)
    - centroid : p+q+r/3
  + Output will be binary num, 1 if front facing and 0 if back facing
* Line 2: Lighting intensity
  + Compute the light intensity where it normalized by the maximum intensity when the light arrives at a perpendicular angle (i.e., 1 if perpendicular) and proportional to the cos(x) where x is the angle of incidence if cos(x) is non-negative. The light intensity is non-negative, i.e., if cos(x) is negative, then output 0.
    - 14 15 and 16 is i
    - Use n from above
* Line 3: Culling-then-intensity
  + write a program building on the previous two which performs culling first and then, if front-facing from the eye's perspective, computes the lighting intensity (culling generally makes the rendering and shading efficient, especially if the shading is more sophisticated than the simple shading used in this problem).
    - Do same as part 2 but only if its front facing

**Part B**

The first line of the input defines the plane and the projection direction (if it is parallel projection). Num11, Num12, and Num13(P is q in second part) corresponds to the point on the plane and Num14, Num15, and Num16 (N Still same ein part 2) defines the normal to the plane before normalization. If it is a parallel projection, Num17, Num18, Num19(M) defines the projection direction. The rest of the input lines define points where there are three points per line. Q (or x in part 2) is the other inputs ((Num 21, 22, 23) then (Num 24, 25, 26) … 31, 32, 33)

* Parallel projection:
  + for each point, project it along the projection direction to into the plane
  + 
  + <http://atlv.org/education/geometry/geometry_seminar4handout.pdf>
* Perspective projection:
  + for each point (let's say x), have the projection direction depend on the point coordinates and be that of the vector from the point to the origin (o-x = -x) and project the point into the plane
  + 
* Output two separate files for the two sub-parts where each line includes **three projected image points**, similarly to the input file format.

**Part C**

* Compute the distance between the point and the plane and output K numbers for K lines on the input. Generate an output file for this sub-part.
  + Each line of the input defines a plane (first six numbers) (**first 3 are x1, x2, x3 (1,2,3), next three are n(4,5,6))** and a point (the last three numbers **last three is p(7,8,9)**). For the J-th line on the input, NumJ1, NumJ2, and NumJ3 corresponds to the point on the plane and NumJ4, NumJ5, and NumJ6 defines the normal to the plane before normalization. NumJ7, NumJ8, NumJ9 are the point coordinates.
    - Plane = (||n||x)+c
      * It looks like n1(p1)+n2(p2)+n3(p3) + c = 0
      * c = - n1(p1) + n2(p2) + n3(p3)
  + 
* If the input had K lines, there are K-1 triangles for testing. For each triangle, if it intersects with the line, find the point of intersection. If it does not intersect, output "Does not intersect." Generate a file for your output including K-1 lines for one triangle per line.
  + The first line of the input defines a line defined by the two points, where Num11, Num12, and Num13 yields one point (**X)**  and Num14, Num15, and Num16 provides the other point coordinates **(Y)**  **These two points make a line**. The rest of the lines on the input defines the three vertex points of a triangle plane (a bounded plane) **V=p-q**
    - Exclude last three in input
    - ANS IN DOES NOT INTERSECT
      * n = v cross w
        + v=q-p w = r-p (p = 21 22 23) (q = 24 25 26) (r = 27 28 29)
        + Cross V and W gets A B and C components

A\*p1

B\*p2

C\*pc

Add them all together, multiply by -1 to get D

THAT IS THE PLANE EQUATION

* + - * 21,22,23 is one point, 24,25,26 is another, 27,28,29 is another
    - LINE EQUATION (parametric) X = X+(Y-X)T
      * x = 84+4t
      * y =36 + 59t
      * z = 55 -24t
      * PLUG INTO PLANE EQUATION, SOLVE FOR T
    - **T IS NOT INTERSECTION POINT**
      * **PLUG IN T TO x y z TO GET INTERSECTION POINTS**
      * **SEE IF POINTS ARE IN TRIANGLE RANGE WITH THE THREE VERTICES WE WERE GIVEN IN THE DOCUMENT**
        + **Plug new x y z into plane equation, if equals zero then points are in plane, if not then points are not in plane**