Lab 2 – CPE 471 – Software Rasterizer: single triangle
Due one week from when assigned (aim to finish the lab in one day)
This project is to be worked on individually

Objective: Introduction to rasterization – single triangle

Overview:
For your first project, you will be writing a program to render (draw) an indexed face set (aka polygonal mesh of triangles) as an image via software rasterization.

Goals for Today:
For today, we will add onto your first lab and only rasterize a single triangle. Your program today is very simple. It must:
• read in three vertices which represent a triangle (from standard in)
• read in (or assign) different color values to each vertex
• compute the bounding box of the triangle using your prior lab’s code
• compute the barycentric coordinates for every pixel in the bounding box
• for any pixel that falls in the triangle, color that pixel, the interpolated color (using the barycentric coordinates to weight the vertex colors)
• write out those pixels as a tga image (base code provided)

Details:
1. Use your code from lab 1 to prompt the user for triangle vertices (in window coordinates).
2. Read-in or assign three different red, green, blue values to the three different vertices (consider differing colors that make a blended color you can reason about such as red and blue or yellow and blue – look up any RGB color chart to pick good colors).
3. Using your prior code, compute the bounding box of the triangle.
4. Now for every pixel in the bounding box, we will rasterize the triangle by computing the barycentric coordinates of each pixel. (see: http://facultyfp.salisbury.edu/despickler/personal/Resources/Graphics/Resources/barycentric.pdf for a reference but watch out for typos). Any pixel that falls within the triangle will have an alpha, beta and gamma between 0 and 1.
5. For any pixel that will be drawn, its color should be a blend of the colors specified at the triangle vertices. Blend the colors using the weights alpha, beta, and gamma that were computed in the barycentric coordinate computation. See the example figure below for an idea of what your rasterized triangle should look like.
6. Write out the pixels into an image similar to in lab 1
8. For lab credit you will need to demonstrate your working code on several vertex tuples specified by myself or the TA. Be sure to test many various shaped triangles!)
For example, rasterization should work as follows for a triangle with red, blue and black vertices