CS770/870 Assignment 1

Due date: Friday, September 9th, 2022, before midnight.

Lateness Penalty: Sat/Sun/Mon: 5% Tue: 10% Wed: 20% Thu: 50% Fri: 100%

Goals

In this assignment, you will install a `C++` compiler and linker, install the `GLFW` windowing and `GLM` math libraries, gain familiarity with `C++`, and practice drawing graphics primitives with OpenGL. You will also gain familiarity with the `GLFW` event model, and with 2D graphics.

Tasks

1. [35 points] (Basic drawing with OpenGL and the GLFW library): Write a C++ program that draws a scene with two parts:

   - Your family name, drawn using `GL_LINESTRIP`, and
   - The first initial of your name, drawn using `GL_TRIANGLE_FAN`.

   I am providing an example program that uses these two graphics primitives to draw my name. The program also draws a grid, to help you locate the vertices.

Here is my solution:

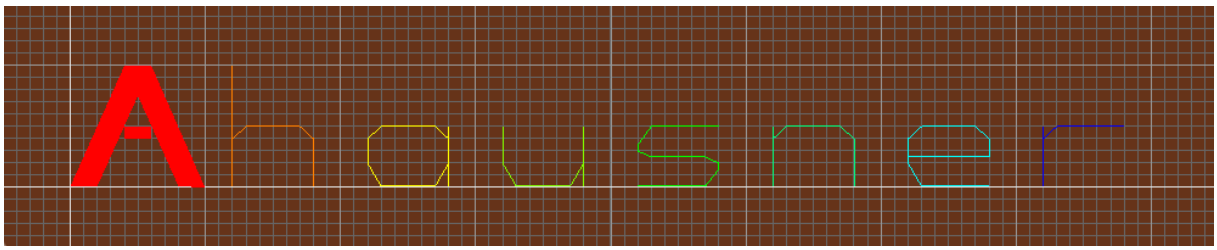

Figure 1: .

Specifications

Here are some requirements:

- Each letter of your name should take up *one* unit of space, left-to-right.
- Each letter should have a different color.

- Leave margin of 0.5 unit on all four sides.
- Use any letter height you want, as long as the text looks readable. The first initial should also take up one unit of space. Thus the name `Dunn` should take up 5 units left-to-right, including margin.
- The world rectangle (and the viewport rectangle) should fit the text, with a small margin all around. The aspect ratio (width/height) of both rectangles should be the same, to prevent distortion.

---

2. [65 points] (Interaction): Write a program using OpenGL which draws a graph. Each graph is defined in a data file. For example, to draw the graph in the file `graph-complex.txt`, you would run your program thus:

   ```
   uncross graph-complex.txt
   ```

   Which should produce this image:

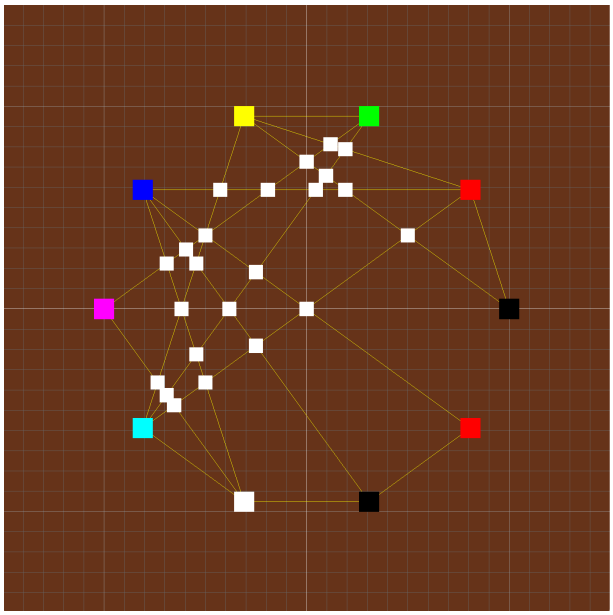

Figure 2: .

Your program lets the user drag the vertices until (hopefully) the edges of the graph don't cross (which is possible only if the graph is planar):

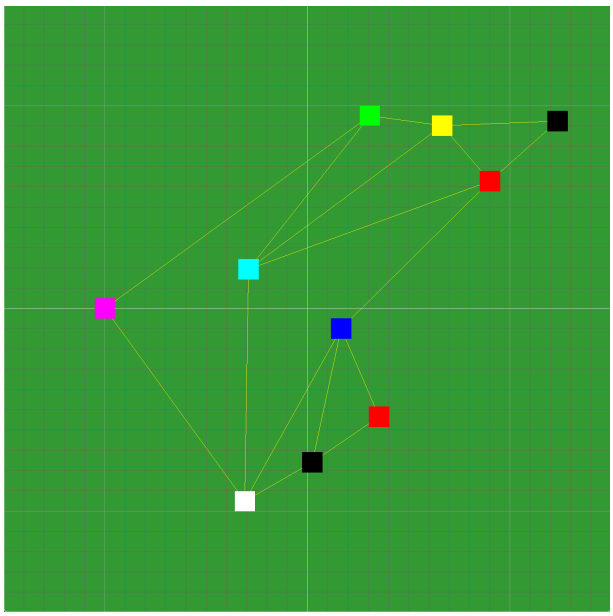

Figure 3: .

For 45 points, your program should:

First read the graph's description from a file (the file's name is the program's command-line argument), and give each vertex an initial position on a circle of radius 1, centered at the origin. Then the program should react to user inputs, so that:

- the user can drag a vertex, and
- the graph is re-drawn in the new position.

You may find it easiest to define your task as a finite-state machine, powered by user events. Take care to handle ALL (mouse up, mouse down, mouse drag) events:

Event	Action
mouse down on vertex	remember vertex
mouse moved	move active vertex
mouse up	forget vertex

For the final 20 points, add these features:

- Wherever any edges intersect, draw small white squares to mark the intersections.
- If there are no intersections (the graph has been shown to be planar), fill the background

color in green.

I am providing skeleton code.

Specifications

Here are some requirements:

- graph vertices should be squares 0.1 x 0.1.
- initially, the vertices should be evenly spaced on a radius 1 circle with center at the origin 0,0.
- graph edges should be yellow (r,g,b = 1,1,0)
- the color of the `i`-th vertex should be derived from its index `i`. The `rgb` color components correspond to the bits of `i`. For example, vertex `0` should be black, and vertex `6` should be yellow (r,g,b = 1,1,0). This may help you with debugging.
- at each intersection, place a small *white* square 0.05 x 0.05.

Turning in Your Work

You should modify the two starting files that I'm supplying:

- `name.cpp`
- `uncross.cpp`

Submit those files before the due date. If you created any other files, submit them as well. If you had to modify the `Makefile` to mention those new files, submit it too.

Please **DO NOT** modify the other files in the skeleton code supplied.

DO NOT UPLOAD ANYTHING ELSE. Above all, *DO NOT* upload a `.zip` file containing a multi-megabyte Visual Studio project!