24783 Advanced Engineering Computation: Problem Set 9

(*) In the following instruction (and in all of the course materials), substitute your Andrew ID for where you see *yourAndrewId*.

START EARLY!

Please submit Faculty Course Evaluation

Faculty Course Evaluation (FCE) is open. Please submit FCE.

FCE is a very important part of education and I am making changes to the course contents based on the feedback.

If you submit FCE, I honor your contribution and credit you 1% toward the final grade.

To take advantage of this credit, please take a screenshot of your FCE confirmation, either immediately after submitting the FCE, or a list of FCEs you have completed, and submit to Canvas.

1 Check Out or Update Base Code and Libraries

Please make sure you have up-to-date libraries and course files before starting an assignment.

If you have not done working-directory set up as described in the first assignment (like in case you need to work from a different computer), please see Problem Set 1 and set up the working directory.

I assume you created the working directory called *24783* under you home directory and you checked out your Git repository in there.

Home directory is typically *C:\Users\username* in Windows, */Users/username* in macOS, and */home/username* in Linux, where *username* is the user name in your local computer.

First, open command-line (Developer PowerShell or Terminal), and move to your working directory by typing:

```
cd ~/24783
```

You need to check out (or clone) Git repositories once. If you have not checked out yet, do the following:

```
git clone https://yourAndrewId@ramennoodle.me.cmu.edu/Bonobo.Git.Server/course_files.git
git clone https://yourAndrewId@ramennoodle.me.cmu.edu/Bonobo.Git.Server/yourAndrewId.git
```

You need to replace "yourAndrewId" with your Andrew ID. You'll be asked to type in credentials.

Also we are going to use two additional repositories:

```
git clone https://github.com/captainys/MMLPlayer.git
git clone https://github.com/captainys/public.git
```

If you are successful, you should have the following directory structure under your home directory.

```
Your User Directory
 |__(Other files and directories)
 |__24783
     |__course_files
     |__yourAndrewID
```

If you already have checked out these repositories (most likely you did for Problem Set 1), you need to update (or git pull) in those repositories. By change directory to the location where you checked out repositries and then type:

```
git pull
```

To update all four repositories, you can type the following commands in a sequence:

```
cd ~/24783/course_files
git pull
cd ~/24783/yourAndrewID
git pull
cd ~/24783/public
git pull
cd ~/24783/MMLPlayer
git pull
```

2 Copy Base Code and Add to Git's Control

Copy ps9 subdirectory from course_files to your directory. The directory structure must look like:

```
Your User Directory
 |__(Other files and directories)
 |__24783
     |__course_files
     |__yourAndrewID
         |__ps9
```

3 Using GLSL Shader Program

See the sample code below. The sample draws clover leaves on the window. The goal of this problem set is to do it with a GLSL program.

Write CMakeLists.txt for ps9. The executable project needs to be named as ps9, and it needs to link ysgl, ysclass, and fssimplewindow libraries from my public repository.

In the base code, source code of vertex and fragment shaders are already given. You need to:

1. Reserve identifiers for vertex and fragment shaders, and a shader program.

2. Assign shader source code for vertex and fragment shaders.
3. Compile shaders.
4. Link shaders together to make a shader program.
5. Cache an identifier for the vertex attribute called ”angle”.

at the beginning of the main function (immediately after FsOpenWindow).

Then, complete MakeVertexArray function so that it makes a vertex array. In this GLSL program, a vertex is a one-dimensional value, just an angle. The vertex shader calculates x and y from the input angle. Therefore, the array created by this function must be an array of one-dimensional values.

Then, render clover leaves using the GLSL program. You need to:

1. Enable the GLSL program.
2. Enable a vertex-attribute array.
3. Set vertex-attribute pointer.
4. Draw Arrays.
5. Disable vertex-attribute array.

If you are successful, your window will look like Fig. 1.

```
#include <stdio.h>
#include <math.h>
#include "fssimplewindow.h"

const double YsPi=3.1415927;

class Vec2
{
public:
    double v[2];
};

Vec2 Clover(double angle)
{
    double r=fabs(sin(angle*2));
    double c=cos(angle);
    double s=sin(angle);

    Vec2 v;
    v.v[0]=c*sqrt(r);
    v.v[1]=s*sqrt(r);
    return v;
}

int main(void)
{
    FsOpenWindow(0,0,800,600,1);
    for(;;)
    {
```

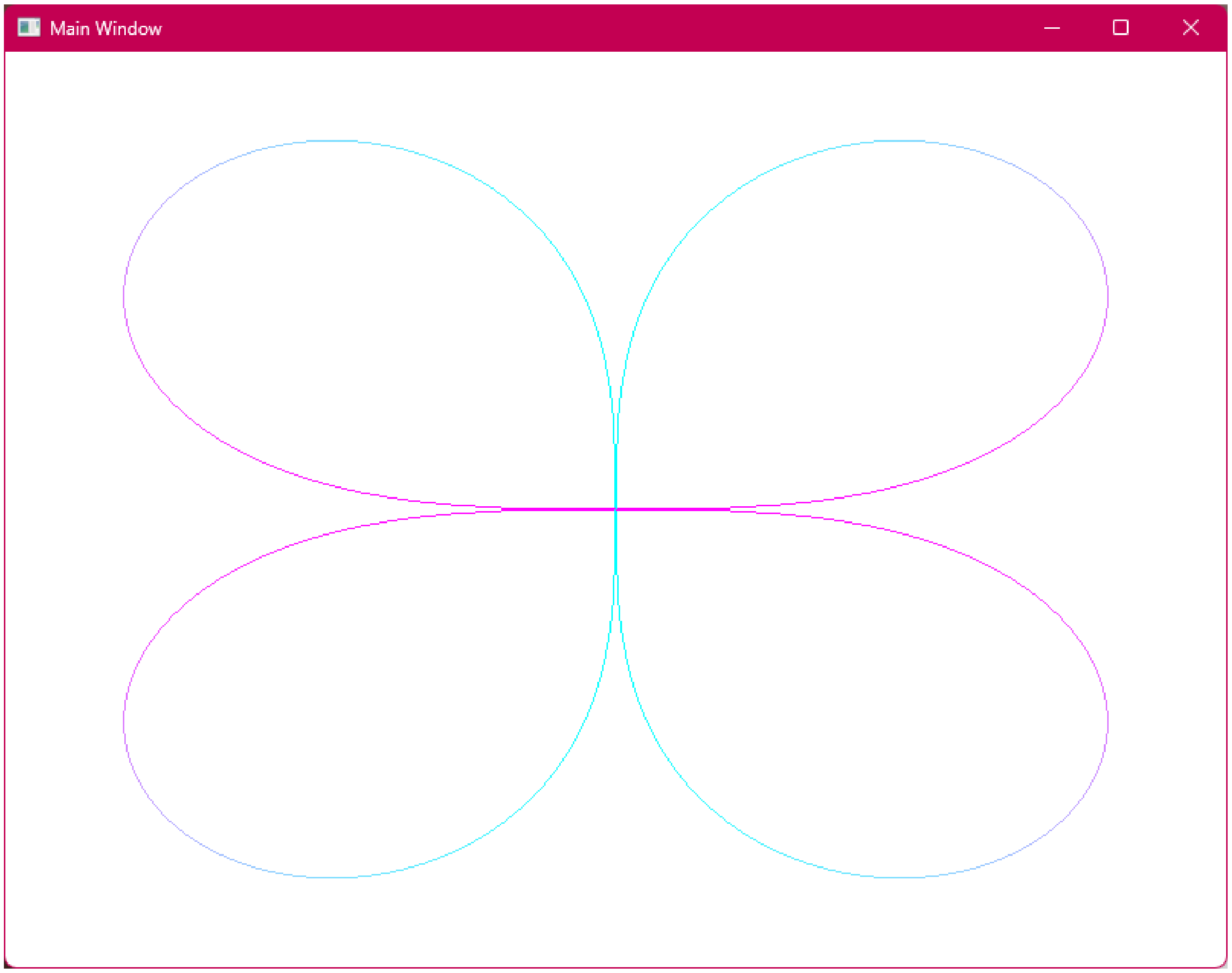

Fig. 1: Screenshot

```
        FsPollDevice();
        if(FSKEY_ESC==FsInkey())
        {
            break;
        }

        glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        glMatrixMode(GL_MODELVIEW);
        glLoadIdentity();

        glBegin(GL_LINE_LOOP);
        for(int i=0; i<360; ++i)
        {
            double angle=double(i)*YsPi/180.0;
            glColor3d(fabs(cos(angle)),fabs(sin(angle)),1);
            glVertex2dv(Clover(angle).v);
        }
        glEnd();
        FsSwapBuffers();
    }

    return 0;
}
```

4 Test Your Code on the Compiler Server

Test your source files (.cpp and .h files) on the compiler server. Some assignment may not require .h files. You do not have to test files that you don't make modifications. The files you need to test are the ones you write or modify.

We have four compiler servers:

- http://freefood1.lan.local.cmu.edu
- http://freefood2.lan.local.cmu.edu
- http://freefood3.lan.local.cmu.edu
- http://freefood4.lan.local.cmu.edu

Make sure you don't see red lines when you select your files and hit "Compile Test" button on the server.

We have multiple servers to make it less likely that all of them need to shut down for maintenance. If do not have to test on all of the servers. You need to make sure that your code passes on one of the servers.

5 Submit

Lastly, you need to submit using git. What you need to do are two things: (1) add files to git's control, and then (2) send to the git server.

5.1 Add Files to git's control

In this case, you want to add all the files under ps7 subdirectory. To do so, type:

```
git add ~/24783/yourAndrewID/ps7
```

This command will add ps7 directory and all files under the subdirectories.

5.2 Send to the Git Server

In Git, sending files to the server is a two-step process. The first step is local commit. You can do it by:

```
git commit -m "Problem Set 7 solution"
```

The message can be anything, but it is recommended to type something meaningful, at least you can see what changes you made to your repository.

Local commit is just local. Git server does not know about any local commit unless the commit is sent (or pushed) to the server. To do so, type:

```
git push
```

Make sure to do it in the CMU network. If you are working from home (probably most likely), use VPN to connect to the CMU network.

You can re-submit (commit and push) your solution as many times as you want with no penalty before the submission due.

6 Verification

It is recommended to clone your repository to a different location and make sure that all of your files have been sent to the Git server.

You can do the following:

```
cd ~
mkdir 24783Verify
cd 24783Verify
git clone https://yourAndrewID@ramennoodle.me.cmu.edu/Bonobo.Git.Server/yourAndrewId.git
```

Once you made sure all the files have been submitted, you can delete files and directories under 24783Verify directory.