24783 Advanced Engineering Computation: Problem Set 8

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

START EARLY!

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 ps8 subdirectory from course_files to your directory. The directory structure must look like:

```
Your User Directory
 ├── (Other files and directories)
 └── 24783
      ├── course_files
      └── yourAndrewID
           └── ps8
```

3 Make a CMake Project

Write CMakeLists.txt for ps8. It is a graphical application, therefore make sure to use MACOSX_BUNDLE. The project name must be ps8. Case sensitive.

Executable ps8 uses all souurce files in the directory, ps8.cpp, mesh.cpp, mesh.h, lattice.h, makeslice.cpp, makeslice.h, glutil.cpp, and glutil.h.

In this assignment, you do not have to make sub-directories. You need only one CMakeLists.txt. Make sure to include public libraries. ps8 must link ysclass and fssimplewindow libraries.

4 Render a slice of the mesh (ps8.cpp)

The basecode ps8.cpp is almost same as stlviewer.cpp we did in class. In PS8, you make changes to the program so that your program can render a slice of the mesh.

When the user picks a polygon, render a white line loop is an intersection between the mesh and a plane that passes through the center of the picked polygon and perpendicular to vector $(1, 1, 1)/\sqrt{3}$. The program must start from the picked polygon and traverse to a neighboring polygon that shares an edge intersecting with the plane. Repeat traversal until it comes back to the original polygon or can no longer find the next polygon.

For this purpose, implement MakeSlice function in makeslice.cpp. The function prototype is already written in the base code. The function takes a reference to the mesh, a polygon handle where the traversal should start, and a point that the cutting plane passes through, and the normal vector of the plane.

While traversing, you need to make sure you do not step back to the polygon that has already been visited. You need to keep track of which edge has been already crossed, and prevent same edge to be crossed twice. If one of the vertex is on the plane, the process easily becomes unstable, so if you detect, one vertex is on the plane, stop traversal.

In the main function, when pickedPlHd is not a null-polygon, calculate the center of the polygon and then call MakeSlice function to get a slice. Then populate sliceVtx and sliceCol for rendering.

In the rendering part of the main function, render the slice using sliceVtx and sliceCol.

5 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.

6 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.

6.1 Add Files to git's control

In this case, you want to add all the files under ps8 subdirectory. To do so, type:

```
git add ~/24783/yourAndrewID/ps8
```

This command will add ps8 directory and all files under the subdirectories.

6.2 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 8 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.

7 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.