# CS 450: Assignment 03

## Programming Assignments (95%)

- Copy src/app/Assign02.cpp and name it **src/app/Assign03.cpp**
    - Similar to before, make sure the shaders are loaded from the **shaders/Assign03** folder (instead of shaders/Assign02)
- Make a copy of the shaders/Assign02 folder and name it **shaders/Assign03**
- Modify **CMakeLists.txt** by adding the following lines to the end of the file:

```
add_executable(Assign03 ${GENERAL_SOURCES} "./src/app/Assign03.cpp")
target_link_libraries(Assign03 ${ALL_LIBRARIES})
install(TARGETS Assign03 RUNTIME DESTINATION bin/Assign03)
install(DIRECTORY shaders/Assign03 DESTINATION bin/Assign03/shaders)
```

- Make sure the sample configures, compiles, and runs as-is
- Change the window title to "Assign03: " + your SITNET ID
    - E.g., "Assign03: realemj"
- **Add the following includes**:
    - #include <assimp/Importer.hpp>
    - #include <assimp/scene.h>
    - #include <assimp/postprocess.h>
- Create a new function: **void extractMeshData(aiMesh *mesh, Mesh &m)**
    - The overall purpose of this function to grab the vertex positions and shape indices from the given aiMesh and store this information in the provided Mesh struct.
        - This is similar to what createSimpleQuad does, except the latter has a hard-coded mesh.
    - Clear out the Mesh's vertices and indices.
    - Loop through all vertices in the aiMesh (mesh->mNumVertices):
        - Create a Vertex.
        - Grab the vertex position information from mesh->mVertices[i] and store it in the Vertex's position.
            - Note: mVertices is an array of aiVector3D structs, so you will have to somehow convert to glm::vec3 structs
            - Set the color of the Vertex to **any color other than (0,0,0,1) or the background.**
                - You may use different colors per vertex using any reasonable scheme.
                - However, alpha values should always be 1.0f.
        - Add the Vertex to the Mesh's vertices list.

- Loop through all faces in the aiMesh (mesh->mNumFaces):
  - Grab the aiFace *face* from mesh->mFaces[i].
  - Loop through the number of indices for this face (face.mNumIndices):
    - Add each index for the face (face.mIndices[k]) to the Mesh's list of indices.

- **In the main function:**
  - **The model to load will be provided on the command line:**
    - **Use "sampleModels/sphere.obj" as your default model path.**
    - If argc >= 2, grab argv[1] as the model path to load and convert to a string.
    - ***NOTE: Do NOT use cin!! You MUST use the command line arguments!***
  - **Load the model (given by the model path) using Assimp to get an aiScene**
    - *Use the following options (OR'ed together):*
      - aiProcess_Triangulate
      - aiProcess_FlipUVs
      - aiProcess_GenNormals
      - aiProcess_JoinIdenticalVertices
  - **Check to make sure the model loaded correctly**
    - *Print error and exit if ANY of these are true:*
      - The scene object is null
      - scene->mFlags & AI_SCENE_FLAGS_INCOMPLETE
      - scene->mRootNode is null
  - **Comment out the previous Mesh object, createSimplePentagon, createMeshGL, and MeshGL object code.**
  - **Create a C++ vector of MeshGL's**
    - *Add item:* myVector.push_back(thing);
    - *Number of items:* myVector.size()
    - *Get item:* myVector.at(i) OR myVector[i]
  - **Before your drawing loop BUT after the GLFW/GLEW setup:**
    - For each **mesh** in the scene (mMeshes with mNumMeshes):
      - Create a **Mesh** object and **MeshGL** object inside the loop
      - Call **extractMeshData** to get a **Mesh** from each scene->mMeshes[i]
      - Call **createMeshGL** to get a **MeshGL** from that **Mesh**
      - Add the **MeshGL** to your **vector of MeshGL's**
  - **In your drawing loop:**
    - Comment out the drawing call for the original MeshGL
    - Loop through all of your MeshGL objects and call drawMesh() on each one
  - **After your drawing loop:**
    - Comment out previous cleanupMesh call
    - Loop through all of your MeshGL objects and call cleanupMesh()
    - Clear out your list of MeshGL objects

## Screenshot (5%)

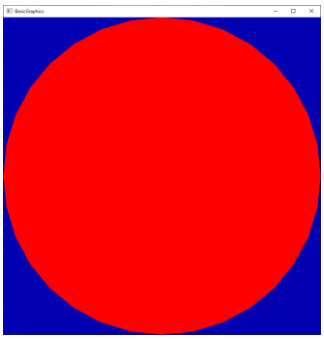
Sample models have been provided for this assignment. To pass in command line arguments:

- Go to .vscode/launch.json
- Change “args” entry to [ "./sampleModels/teapot.obj" ],

For this part of the assignment, **upload TWO screenshots** of the application window (note that window title should be "Assign03: yoursitnetid"):

- **One displaying sphere.obj**
- **One displaying teapot.obj**

*sphere.obj (color need not match this image):*



*teapot.obj (color need not match this image):*

## Grading

Your OVERALL assignment grade is weighted as follows:

- 95% - Programming
- 5% - Screenshot