Programming Assignment 1

Announcement:	**Session # 1**
Submission Deadline:	**Session # 4**

Description

This OpenGL programming assignment will provide an introduction to OpenGL programming. More specifically, you will learn how to create simple virtual scenes consisting of objects, display them as wireframe meshes by drawing triangles, interact with them, and also how to set up and manipulate the virtual camera to view the scene from different angles and distances.

A **mesh** is a set of polygons that share vertices and edges which describe the shape of a geometric object. In this assignment you have to recreate a simplified 3D version of the Pacman game consisting of the main player mesh and several sphere meshes, a simple grid, and a coordinate axis object.

Implementation Specifications

Develop an OpenGL application with the following functionality and features:

- Creates a set of three lines representing each coordinate axis in virtual world space, centered at the origin.
- Creates a grid of size 21x21 centered around the origin.
- Creates the main player object [Pacman], and several spheres. The objects should be 3D and be located only on the XY-plane i.e. Z=0. The objects should be placed randomly within an area defined by the range [-10, 10] on each dimension.
- Places a camera with the “Pacman” as the point of focus.
- For display and animation:
  - create a GLFW window of size 800x800 with double buffering support.
  - render the objects in the window.
  - The application should use a perspective view to display the objects and use the depth buffer for hidden surface removal.
  - When Pacman is located on the same grid point as a sphere then the sphere should disappear.
- The application should handle the following input:
  - Pressing the spacebar should re-position the main player at a random location on the grid.
  - The user can incrementally size up the objects by pressing ‘U’ for scale-up and ‘J’ for scale-down. Each key press should result in a small size change.
  - The user can control Pacman using keyboard input i.e. A → left, D → right, W → up, S → down. Pacman should also rotate accordingly if the direction is changed.

- The world orientation is changed by using keyboard input i.e. left arrow → Rx, right arrow → R-x, up arrow → Ry, down arrow → R-y. Pressing the "Home" button should reset to the initial world position and orientation.
- The user can change the rendering mode i.e. points, lines, triangles based on keyboard input i.e. key 'P' for points, key 'L' for lines, key 'T' for triangles. The user can pan and tilt the camera as follows:
  - while right button is pressed → use mouse movement in x direction to pan; and
  - while middle button is pressed → use mouse movement in y direction to tilt.
- The user can zoom in and out of the scene - while left button is pressed → use mouse movement to move into/out of the scene.
- Window resize handling: The application should handle window resize events and correctly adjust the aspect ratio accordingly. This means that the meshes should not be distorted in any way.

- The application should use OpenGL 3.0 and onwards, and include brief comments explaining each step.

Submission (electronic submission through Moodle only)

Please create a zip file containing your C/C++ code, vertex shader, fragment shader, a readme text file (.txt). In the readme file document the features and functionality of the application, and anything else you want the grader to know i.e. control keys, keyboard/mouse shortcuts, etc.

Additional Information

- You can use the skeleton code provided during the lab sessions to get started.
- A video demonstrating the functionality is posted on YouTube: **https://www.youtube.com/watch?v=5M2G0V9zj6I**

Extra Credit (3 points each)

(1) Load the provided text file which determines valid paths of where Pacman can move.
(2) Create enemies which run after Pacman. There should be at least three enemies. When an enemy is located at the same grid point as Pacman, the application should restart.
(3) Allow the grid size to be specified by the user at the beginning of the game.

Evaluation Procedure

You MUST demonstrate your solution program to the lab instructor during lab hours. You must run your submitted code, demonstrate its full functionality and answer questions about the OpenGL programming aspects of your solution. Major marking is done on the spot during demonstration. Your code will be further checked for structure, non-plagiarism, etc. However, ONLY demonstrated submissions will receive marks. Other submissions will not be marked.