# CPS511 Assignment 2 Readme

## How to Compile:

This Assignment was compiled and linked using Visual Studio 2013 Professional. The solution file is *“Assignment 2.sln”*. I used freeglut as the glut library. Freeglut files are included in *“./Dependencies/freeglut/”* folder.

To set up the solution in Visual Studio, I took the following steps:

1. In the **Solution Explorer** window, changed **Assignment 1->Properties->Configuration** to *“All Configurations”*. The platform is *“Win32”*.
2. Under **Assignment 1 -> Properties -> Linker -> General**, added *“Dependencies\freeglut;”* to **Additional Library Directories**.
3. Under **Assignment 1 -> Properties -> Linker -> Input**, added *“opengl32.lib;freeglut.lib;”* to **Additional Dependencies**.

To compile and run the solution, I took the following steps:

1. Run **Start Debugging (F5)** or **Build Solution (F7)** once; a system error message will pop-up saying   
   *“The program can’t start because freeglut.dll is missing from your computer. Try reinstalling the program to fix this problem.”*   
   This creates a **Debug** folder (or **Release** folder) under the project solution folder.
2. Paste a copy of ***freeglut.dll*** into the **Debug** folder (or **Release** folder) and rerun the debugger (or build the solution).

## Parameters and Implementation details:

### File Modeller.cpp:

* A fixed size array is used to hold the complex objects. The array is declared in line 52 as “*ComplexObj \*objects[MAXOBJECTS];”*.
* The size of the array is defined in line 19 as “*#define MAXOBJECTS 10”*
* The complex objects array index variable is declared in line 53 as *“int nextobj = 0;”*.
* In line 56, the translation delta (for x-, y-, z-direction) is set to 0.2 units; the scaling increment is set to 0.2x; the rotation angle delta is set to 5.0 degrees.
* Line 24 defines timer frames of 10 – each translate/rotate unit is subdivided into 10 steps of 1/10th size each to create the smooth animation.   
  Line 57 executes each translate/rotate unit at a rate of 1000/20 = 50 ms (i.e. 20 fps).   
  Line 58 sets the timerdelay (between each animation slice), which is 1000/(20\*10) = 5 ms.   
  When *glutTimerFunc()* is called to animate (first-person) translation and rotation, each unit of translation/rotation is subdivided into 0.2/10 units each – *glutPostRedisplay()* is called 10 times.
* “F1” creates the first complex object – mage head  
  “F2” creates the second complex object – tri-rocket   
  “F3” creates the third complex object – jeweled bracelet/ring  
  The objects are created using glu and glut quadric functions. (see file ***ComplexObj.h***) A display list is used to create the mage head and bracelet; a nested display list(s) is used to create the tri-rocket.
* NAVIGATE (first-person) mode is added to the list of Actions (see lines 62, 589-605, 660-669, 716-725, 801-808, 884-891, 895-896).   
  “F5” is used to enter NAVIGATE mode, “F6” is used to exit.

## Features Implemented:

* F1 (create first complex object – mage head and position it on the floor in the middle of the room, i.e. @ (0.0, 0.0, 0.0)) (NOTE: y-coord is set to 0.0 – the bottom of object)
* F2 (create second complex object – tri-rocket and position it on the floor in the middle of the room, i.e. @ (0.0, 0.0, 0.0)) (NOTE: y-coord is set to 0.0 – the bottom of object)
* F3 (create third complex object – jeweled bracelet/ring and position it on the floor in the middle of the room, i.e. @ (0.0, 0.0, 0.0)) (NOTE: y-coord is set to 0.0 – the bottom of object)
* ‘t’ (translate cube in x and z directions; use arrow keys to move)
* ‘s’ (scale cube in x and z directions; use arrow keys to scale)
* ‘r’ (rotate cube around vertical y-axis; use left/right arrow keys to rotate)
* ‘e’ (extrude cube; use up/down arrow keys to increase/decrease height of cube; bottom of cube does not go down below floor, top of cube does not go above ceiling)
* ‘h’ (translate cube up/down; use up/down arrow keys to translate)
* ‘c’ (change selected cube; left/right arrow keys selects the previous/next cube; the previous/next selected cubes wrap around the list)
* ‘+’ (multiple select; left/right arrow keys set previous/next cube to select/include; all translates, rotates, scales, extrudes affect all selected cubes. Collision detection is implemented such that if any one of the cubes will go out of bounds during a transformation, the transformation is stopped.
* ‘-‘ (deselects all selected cubes)
* F5 (activate NAVIGATE (first-person) mode. Activates only if there is at least one object selected. If there are more than one object selected, the first selected object will be used and the rest deselected.)
* F6 (exit NAVIGATE mode. The navigated object remains selected and action is set to TRANSLATE.)
* Mouse left-click + move direction (controls the camera around the hemisphere at the current zoom level)
* Mouse middle wheel scroll (controls the zoom at a fixed polar and azimuthal angle. Scroll forward zooms in; scroll backward zooms out)
* Wall collisions handled
* Smooth animation for translate/rotate in NAVIGATE (first-person) mode is implemented

### Other features:

* ‘q’ (quits application)
* Scaling (in x, z, and y (extrude)) directions are normalized to 1.0 (i.e. divided by the size of the object). I set this scale ratio to accommodate uniform scaling for various sized objects.
* The look from position is set to 1.0 unit directly above the object’s height. The look at position is set to -15° from the horizontal relative to the object’s look from position. This facilitates a more natural viewing angle and allows the object/viewer to see more of the ground in front of it. (see file ***Modeller.cpp***, function *calculateLookat()*).