# PG612 - Mandatory Assignment 2: Shadow Mapping and Shaders

In this assignment, you will learn how to implement real-time shadows using shadow maps, and the use of GLSL shaders for different effects.

## Important information:

* **Read all parts of this text, and do as described.**
* **All subtasks are possible to implement independently. Do not get stuck on a single subtask. It is better to get 90% right of all than 100% right on one.**
* **You have to write all of the code yourself. Base your solution on the lecture notes and books in the course.**

### Subtask 1: Implement shadows using the shadow maps technique.

* Base your implementation on the skeleton code.
* Start by implementing render to an FBO depth texture
* Toggle showing the FBO depth texture as a texture on screen when pressing “t”

**Hints:**

* Use the alpha value to make the on-screen FBO texture semi-transparent.
* Do not show the debug FBO texture on the whole screen, but in the lower left corner.
* Remember that the FBO texture will be shown differently if you set GL\_TEXTURE\_COMPARE\_MODE or GL\_TEXTURE\_COMPARE\_FUNC
* See lecture 5 for details on implementation.

### Subtask 2: Implement wireframe and hidden line rendering

* Implement wireframe and hidden line rendering
* Pressing “1” gives regular phong, “2” gives wireframe, “3” gives hidden line

**Hints:**

* Implement as different shaders for each mode
* See lecture 6 for details and hints

Subtask 3: Cube mapping for diffuse lighting

* Use the included cubemap images to create a diffuse cubemap
* Use the normal vector in the fragment shader to access look up into the cube map
* Use the texture value as the diffuse component of the lighting equation

**Hints:**

* A diffuse cube map replaces the diffuse light in the phong lighting equations with a single texture lookup.
* See lecture 6 for details and hints.

## Requirements

* Code must extend the skeleton from itsl
* Code must compile and run out-of-the-box using
  + Visual Studio 2010, SDL, GLM, Assimp, OpenGL 3.3+
* Short (5-25 lines, 80 columns) text (.txt) README-file
* Doxygen compliant source code comments (javadoc)
* No temporary files (Visual Studio, svn, etc.).

## Grading

The following criteria are used for grading this assignment:

* 40% Doing the assignment. For example:
  + Have you completed all subtasks?
  + Have you written every line of code yourself?
* 30% Code quality. For example:
  + Is your solution correct, simple, and elegant?
  + OpenGL efficiency,
  + Use of deprecated OpenGL functionality,
  + Useful comments,
  + Compilation errors, warnings, etc.
* 20% Visual quality and natural feel. For example:
  + Does your solution “feel” natural, does it look “right”, etc.
* 10% Overall rating. For example:
  + Anything I feel is not covered by the above points which deserves extra credit