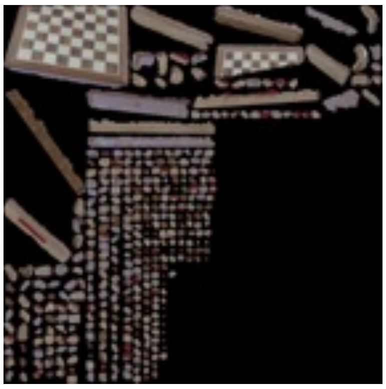
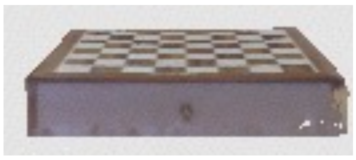
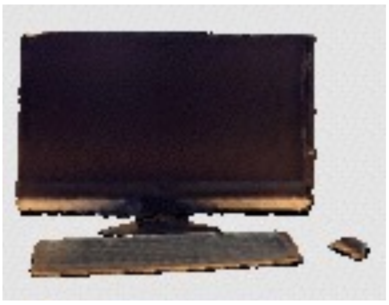
Due Date: April 16th 2015 at 21.00
While completing this practical you should be careful as to where you place your .html and .js files so they cannot be copied by others. You should submit a **single** zip file along with a **single page** PDF report.



Aims and Objectives:
The basic aims and objectives of this assignment are:
- To extend your knowledge of both vertex and fragment shaders
- To understand and apply rotation, translation and scaling transformations to both objects and viewpoints
- To determine suitable transformations for classical orthogonal and perspective model views
- To support basic model interaction
- To load and suitably manage multiple data models captured from real and optionally synthetic sources



The **extension** aims are
•The explore the lighting and shading of 3D models
•To gain exposure to the texture mapping of 3D models
•To explore manipulation of 3D models



The overall objective is to write a WebGl application for processing and rendering multiple 3D .obj models which are read in to your program.

Along with the code you should submit a **2 page report** that should describe what you have achieved and any problems you encountered.

Resources
1. Course Slides and https://studres.cs.st-andrews.ac.uk/CS4102-CG/Lectures/common/
2. 3D models located at https://studres.cs.st-andrews.ac.uk/CS4102-CG/Practicals/P2/data
3. Wavefront OBJ File Format Summary http://www.fileformat.info/format/wavefrontobj/egff.htm
4. Example GLSL code https://studres.cs.st-andrews.ac.uk/CS4102-CG/Practicals/P2/GLSL/gourand.glsl

Learning Outcomes
Deeper understanding of degrees of freedom in computer graphics, clipping and using a frustrum, building matrices for isometric, front elevation and side elevation views. Enhanced learning outcomes include a deeper understanding of smooth vs gouraud shading, texture mapping, blend functions for visual effects.

An answer getting a grade of **Merit** will
- Have a webpage which is a 3D model viewer with basic interactions including the ability to pan, zoom, rotate the view (centered on the origin or the model just loaded)
- Allow the user to select at least 4 predefined views of the model (isometric and front elevation orthographic views and 2 perspective views)
- Allow a user to load into the viewer at least 5 different 3D models from basic .obj files (at the simplest, from a file in the same directory)
- Allow a user to rotate, translate and scale the model just loaded (no need to handle selection) but you need to show a bounding box around the model while it's undergoing manipulation
- Allow a user to load multiple models (eg. first a room model and then a model of objects inside the room)
- Lighting sufficient to be able to see a wireframe, flat filled model and one based on any given vertex colouring

An answer getting a grade of **Excellent** will

- Include everything above
- Handle model data based on inputs from from (.obj .jpg .mtl) examples and texture mapping

An answer getting a grade of **Exceptional** will

- Include everything above
- Allow multiple sources of light (including one spot light)
- Explore smooth and gouraud shading or blend functions for visual effects

Hints:

1. Understanding the format of the data given is the first and key step! (open a small .obj file to get a quick feel for the format)
2. Unless you are planning to release your .obj reader as a general purpose piece of code to the wider community be careful not to over engineer it. However do ensure you handle exceptions well.
3. When reading .obj files you will need to handle the format and reading aspects including vertex data, vertexNormals, textures, index data (you need to study the .obj format noted in resources 3)
4. Wavefront's *.obj has a format which facilitates Vertex Colouring ie. vertex X Y Z then Red Green Blue
5. Some .obj files includes data with vertex colour information (as noted in 4) versus texture data as described from a .mtl file
6. Plan carefully for how you will setup your modelViewMatrix, projectionMatrix etc. and if you will use multiple fragment and vertex shaders
7. The example code shown in resource 4 gives you an example of how aspects a gourand shader is built

If you want to make your approach more general so you take files from across the network please do this.

Aaron Quigley, Mar, 2015