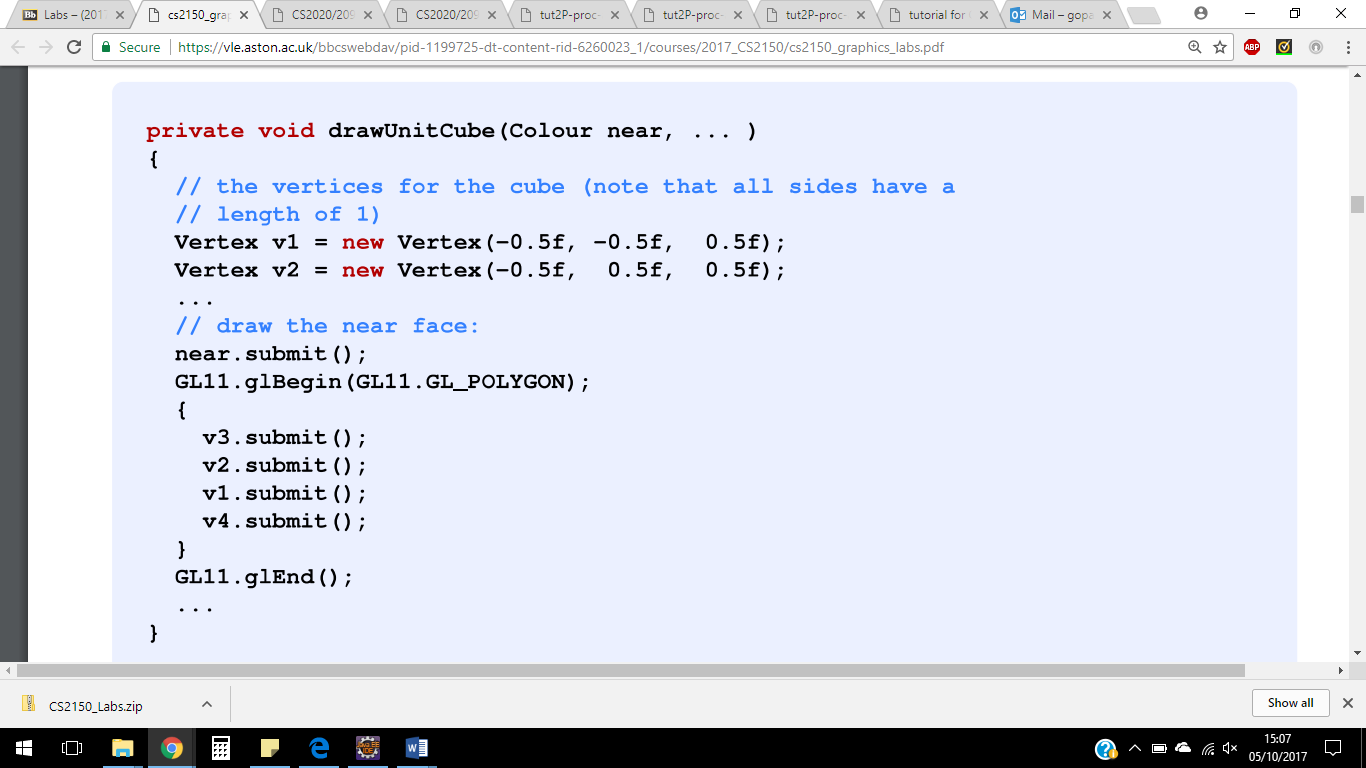
CS2150 Lab Notes

Lab 1:

* Scene graphs:
* Provides and **abstraction of contents** of the scene in a very **compact** form
* A **hierarchal structure** (commonly a **tree**), that represents the **logical relationships** between objects in each virtual world.
* Simplified scene objects consist of 2 element types:
  + Nodes: representing scene objects
  + Arcs: representing spatial relationships between objects
* **Spatial relationships** exist **between parent** and **child** scene objects
  + Includes relative **positioning**, relative **orientation**
* Each **node** in the scene graph **represents a new coordinate system**
  + Scene is a root element which represents the origin of virtual world
* Base class and libraries:
* Tell JVM to import LWJGL library
* New features:
* GL11. – identifies the command as being related to OpenGL v1.1 API.
* .glPushMatrix() - OpenGL command that **stores** the **current value** of composite **model-view matrix** onto a **matrix stack**
* .glPopMatrix() – Recover value from the top of the stack
* .glTranslatef(-.-f, -.-f, -.-f) – Similar to what already **exists in maths** there is a **translation** in the **x,y,z direction** by a factor of “-.-f” which can be +ve or -ve



* stores vertex coordinates
* **.submit() passes vertex** to OpenGL **using** command **glVertex3f(x,y,z),** so it can be drawn
* OpenGL works by **defining objects** to be drawn **between glBegin(---) and glEnd()**
* In glBegin(---) we need to specify what is being drawn
* Vertices must ALWAYS be given in **anti-clockwise** order
* GL\_PROJECTION is used to define lens of the camera (e.g. zoom in and out
* **GL\_MODELVIEW** is used to **position and point** the **camera** to define where it is in space. Use GLU.gluLookAt command to position and point that camera and specify the (x,y,z) coordinates and also for the direction of “up”