Lab 03: GL Transformations (Chapter 7,9)

I. GL is only 3D. To fake 2D do transforms with a fixed z component.

II. \textit{Basic Geometric Transformations}

A. \textit{Translation} - \texttt{glTranslate3} (tx, ty, tz)
   \texttt{glTranslate3f} (tx, ty, tz)

B. \textit{Rotation} - \texttt{glRotate3f} (theta, vx, vy, vz) - angle-axis formulation. Theta in \textit{degrees} not \textit{radians}.

C. \textit{Scale} - \texttt{glScale3f} (sx, sy, sz)

D. GL stores composite transforms so the following:
   \texttt{glTranslatex(0, 0, 1);}  \hspace{1cm} \texttt{glScale(2, 3, 4);}  \hspace{1cm} \texttt{glDrawVertices();}
   \hspace{1cm} \text{is stored as } M = \texttt{TR} \text{S}

  \text{*Ask downsides/effects*} - need to be able to reverse operations

III. GL matrix operations

A. Matrix mode - There are 4 different matrices used by GL sources
   \texttt{GL_MODELVIEW} - world coordinates + camera
   \texttt{GL_PROJECTION} - projection transformations
   \texttt{GL_COLOR} - color modifications \textit{not as useful}
   \texttt{GL_TEXTURE} - texture modifications

   \texttt{glMatrixMode(int)} - set current matrix

B. Reset transform - \texttt{glLoadIdentity()}

C. Set/Get matrix - avoid using (unless you know better, e.g. need shear)
   \texttt{glLoadIdentity();} - pass by 16 element array pointer
   \texttt{glGetDoublev(GL_MODELVIEW_MATRIX, matrix);} get 4x4 matrix
   \texttt{glGetDoublev(GL_MODELVIEW_MATRIX, matrix);} set 4x4 matrix
   \texttt{glMatrixMode(int)} - pass by 16 element array pointer

D. Manually multiply matrices - avoid using (unless you know better, e.g. shear)
   \texttt{glMultMatrixd(\ast \ast)}

E. Each matrix is also on a stack to allow easy reversal of computations
   \texttt{glPushMatrix()}
   \texttt{glPopMatrix()}

   so for each model:
   \texttt{glPushMatrix()}
   \texttt{// model transforms specific}
   \texttt{// Draw model}
   \texttt{glPopMatrix()}
