ICS 186A: Computer Graphics
Spring 2002
Gopi Meenakshisundaram

Programming Assignment 1
Assigned : Wednesday, April 3, 2002
Due: Friday, April 12, 2002, 9am.
Estimated Time: 4 hrs

Implement a stack of matrices and the following functions to operate on the stack of matrices.
void my_glLoadIdentity(void)
void my_glPushMatrix(void)
void my_glPopMatrix(void)

void my_glLoadMatrixf(const GLfloat *m)
void my_glLoadMatrixd(const GLdouble *m)

void my_glTranslated(GLdouble x, GLdouble y, GLdouble z)
void my_glTranslatef(GLfloat x, GLfloat y, GLfloat z);

void my_glRotated(Gldouble angle, GLdouble x, GLdouble y, GLdouble z)
void my_glRotatef(Glfloat angle, GLfloat x, GLfloat y, GLfloat z);

void my_glScaled(GLdouble x, GLdouble y, GLdouble z)
void my_glScalef(GLfloat x, GLfloat y, GLfloat z);

void my_glGetMatrixf(const GLfloat *m)
void my_glGetMatrixd(const Gldouble *m)

1. Allow a maximum of 16 matrices to be pushed.
2. Report error if the stack is empty when a my_glPopMatrix function is called and continue.
3. Report error if the stack is full (16 elements) when a my_glPushMatrix is called and continue.
4. *m* is a pointer to the array of 16 consecutive values (linear array) of the matrix (4x4 matrix) in *column major order*.
5. Implement a "static" array of matrices so that consecutive calls to your matrix manipulation routines will be accumulated.
6. Include gl.h to make use of the data types GLfloat* and Gldouble*.
7. The meaning of each of the above functions (except my_glGetMatrix*) takes the same semantic meaning as the functions in OpenGL library (without "my_"). Use man pages available on Sun machines to find the semantics of these functions.
8. my_glGetMatrix* function returns through *m* the matrix you have on the top of the stack.
9. All internal computation of composition of matrices should use GLdouble.
10. Michael Shafae (the TA) will send you the details of the program that would interface with your implementation of the functions, and the details on how to electronically turn-in the assignments.