**Chapter 5**

**IMPLEMENTATION**

**5.1 Built-in functions**

1. **glutPostRedisplay()**

Marks the current window as needing to be redisplayed.

1. **glPushMatrix()** and **glPopMatrix()**

Push and pop the current matrix stack.

1. **glTranslatef() and glRotatef()**

Multiplies current matrix by Translation and Rotation matrix respectively.

1. **glMatrixMode (GL\_PROJECTION)**

Specifies which matrix is the current matrix.

1. **glLoadIdentity()**

Replaces current matrix with identity matrix.

1. **glutSwapBuffers()**

Swaps the buffers of the current window if double buffered.

1. **glutInitDisplayMode (GLUT\_DOUBLE | GLUT\_RGB)**

Sets the initial display mode.

1. **glutInitWindowSize (1000, 1200)** and **glutInitWindowPosition (0, 0)**

Set the initial window size and position respectively.

1. **glutCreateWindow()**

Creates a top level window with the window name as specified.

1. **glutDisplayFunc(display)**

Sets the display callback for the current window.

1. **glutReshapeFunc(reshape)**

Sets the reshape callback for the current window.

1. **glutMainLoop()**

Enters the GLUT event processing loop. This routine should be called at most once in a GLUT program. Once called, this routine will never return. It will call as necessary any callbacks that have been registered.

**5.2 Source Code**

#include <windows.h>

#include <GL\glut.h>

#include<math.h>

#include<string.h>

#include<stdlib.h>

void topview();

void chop1();

void chop2();

void chop3();

void chop4();

void chop5();

float tx,ty,tz;

int to,tooo;

int tchop1;

void Write(double x, double y, double z, double scale, char \* s) {

int i, l = strlen(s);

glRasterPos3f(x, y, z);

for (i = 0; i < l; i++) {

glColor3f(0.0, 0.0, 0.0);

glutBitmapCharacter(GLUT\_BITMAP\_HELVETICA\_18, s[i]);

}

glPopMatrix();

}

void Phil1()

{

glBegin(GL\_LINE\_LOOP);//chair back

glColor3f(0.0f, 0.0f, 0.0f);

glVertex3i(200,410,0);

glVertex3i(200,416,0);

glVertex3i(245,416,0);

glVertex3i(245,410,0);

glEnd();

glBegin(GL\_LINE\_LOOP);//chair back 1

glColor3f(0.8,0,0.1);

glVertex2i(206,410);

glVertex2i(209,410);

glVertex2i(209,374);

glVertex2i(206,374);

glEnd();

glBegin(GL\_LINE\_LOOP);//chair back 1

glColor3f(0.8,0,0.1);

glVertex2i(237,410);

glVertex2i(240,410);

glVertex2i(240,374);

glVertex2i(237,374);

glEnd();

glBegin(GL\_POLYGON);//leg

glColor3f(0.8,0,0.1);

glVertex2i(207,369);

glVertex2i(203,339);

glVertex2i(245,339);

glVertex2i(240,370);

glEnd();

glBegin(GL\_POLYGON);// Phil body 1

glColor3f(0.3,0.7,0.1 );

for(to = 0; to <= 360; to++)

{

tx =22\*sin(to);

ty =15\*cos(to);

tz = 0;

glVertex3f(223+tx,380+ty,tz);

}

glEnd();

glBegin(GL\_POLYGON);// Phil head 1

glColor3f(0,0,0);

for(to = 0; to <= 360; to++)

{

tx =11\*sin(to);

ty =16\*cos(to);

tz = 0;

glVertex3f(223+tx,399+ty,tz);

}

glEnd();

}

void myDisplay(void)

{

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

glShadeModel(GL\_FLAT);

if (flag == 1) {

page1();//spash screen

} else if (flag == 2) {

page2();//menu

} else if (flag == 3) {

page5();//intro

} else if (flag == 4) {

topview();

}

glFlush();

}

int main(int argc, char \*\*argv)

{

glutInit(&argc, argv); // to initialize the toolkit;

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB); // sets the display mode

glutInitWindowSize(1500, 1200); // sets the window size

glutInitWindowPosition(20, 10); // sets the starting position for the window

glutCreateWindow("Dining Philosophers Problem");

glutDisplayFunc(myDisplay);

glutKeyboardFunc(keyboard);

myInit();// additional initializations as necessary

glutMainLoop(); // go into a loop until event occurs

return 0;

}