Experiment 10

Aim : Construct the following 3d Shapes: Cube and Sphere

a) CUBE CODE:

```
#include <GL/glut.h>

GLfloat xRotated, yRotated, zRotated;

void init(void)

{ glClearColor(0,0,0,0);

}

void DrawCube(void) {

glMatrixMode(GL_MODELVIEW);

// clear the drawing buffer. glClear(GL_COLOR_BUFFER_BIT);

glLoadIdentity(); glTranslatef(0.0,0.0,-10.5);

glRotatef(xRotated,1.0,0.0,0.0); // rotation about Y axis glRotatef(yRotated,0.0,1.0,0.0);
// rotation about Z axis glRotatef(zRotated,0.0,0.0,1.0);

glBegin(GL_QUADS); // Draw The Cube Using quads glColor3f(0.0f,1.0f,0.0f); // Color
Blue

glVertex3f( 1.0f, 1.0f,-1.0f);

glVertex3f(-1.0f, 1.0f,-1.0f);

glVertex3f(-1.0f, 1.0f, 1.0f);

glVertex3f( 1.0f, 1.0f, 1.0f);

glColor3f(1.0f,0.5f,0.0f); // Color Orange

glVertex3f( 1.0f,-1.0f, 1.0f); // Top Right Of The Quad (Bottom)

// Top Right Of The Quad (Top)
```

// Top Left Of The Quad (Top)

// Bottom Left Of The Quad (Top)

// Bottom Right Of The Quad (Top)

glVertex3f(-1.0f,-1.0f, 1.0f); glVertex3f(-1.0f,-1.0f,-1.0f); glVertex3f(1.0f,-1.0f,-1.0f);
glColor3f(1.0f,0.0f,0.0f); // Color Red

glVertex3f(1.0f, 1.0f, 1.0f); glVertex3f(-1.0f, 1.0f, 1.0f); glVertex3f(-1.0f,-1.0f, 1.0f);

glVertex3f(1.0f,-1.0f, 1.0f); glColor3f(1.0f,1.0f,0.0f); // Color Yellow

glVertex3f(1.0f,-1.0f,-1.0f); glVertex3f(-1.0f,-1.0f,-1.0f); glVertex3f(-1.0f, 1.0f,-1.0f);

glVertex3f(1.0f, 1.0f,-1.0f); glColor3f(0.0f,0.0f,1.0f); // Color Blue

glVertex3f(-1.0f, 1.0f, 1.0f); glVertex3f(-1.0f, 1.0f,-1.0f); glVertex3f(-1.0f,-1.0f,-1.0f);
glVertex3f(-1.0f,-1.0f, 1.0f); glColor3f(1.0f,0.0f,1.0f); // Color Violet

glVertex3f(1.0f, 1.0f,-1.0f); glVertex3f(1.0f, 1.0f, 1.0f); glVertex3f(1.0f,-1.0f, 1.0f);
glVertex3f(1.0f,-1.0f,-1.0f);

// Top Right Of The Quad (Right)

// Top Left Of The Quad (Right)

// Bottom Left Of The Quad (Right)

// Top Left Of The Quad (Bottom)

// Bottom Left Of The Quad (Bottom) // Bottom Right Of The Quad (Bottom)

// Top Right Of The Quad (Front)

// Top Left Of The Quad (Front)

// Bottom Left Of The Quad (Front) // Bottom Right Of The Quad (Front)

// Top Right Of The Quad (Back)

// Top Left Of The Quad (Back)

// Bottom Left Of The Quad (Back)

// Bottom Right Of The Quad (Back)

```
// Top Right Of The Quad (Left)

// Top Left Of The Quad (Left)

// Bottom Left Of The Quad (Left) // Bottom Right Of The Quad (Left)

// Bottom Right Of The Quad (Right) glEnd(); // End Drawing The Cube

glFlush(); }

void animation(void) {

yRotated += 0.01;

xRotated += 0.02; DrawCube();

}

void reshape(int x, int y) {

if (y == 0 || x == 0) return; //Nothing is visible then, so return //Set a new projection
matrix glMatrixMode(GL_PROJECTION);

glLoadIdentity();

//Angle of view:40 degrees

//Near clipping plane distance: 0.5 //Far clipping plane distance: 20.0

gluPerspective(40.0,(GLdouble)x/(GLdouble)y,0.5,20.0);
glMatrixMode(GL_MODELVIEW);

glViewport(0,0,x,y); //Use the whole window for rendering

}

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

glutInit(&argc, argv);

//we initizlilze the glut. functions glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
glutInitWindowPosition(100, 100); glutCreateWindow(argv[0]);

init();

glutDisplayFunc(DrawCube); glutReshapeFunc(reshape);
```

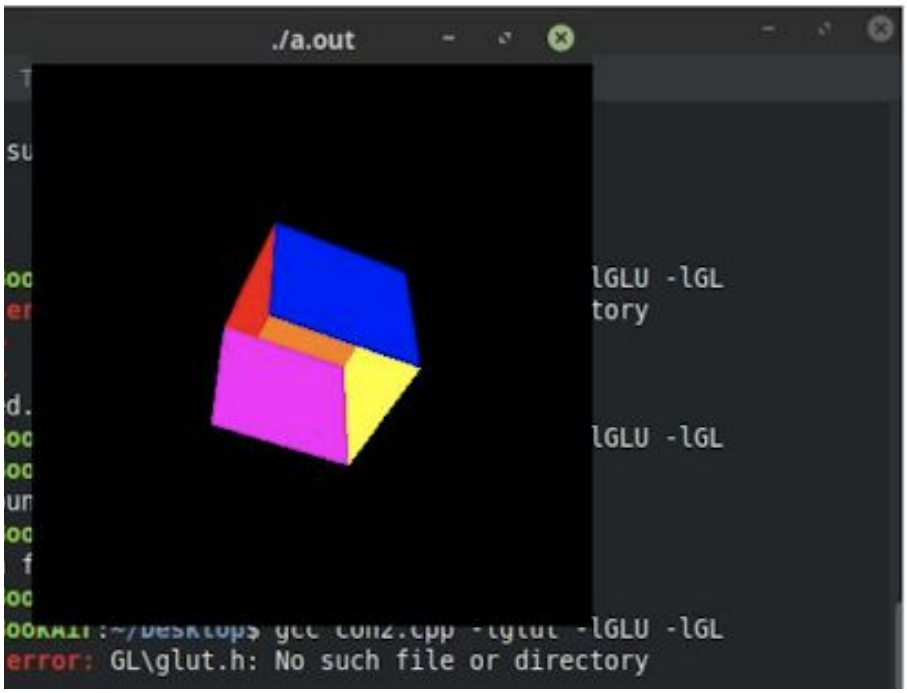
//Set the function for the animation. glutIdleFunc(animation);

glutMainLoop();

return 0;

}

Output :



b) SPHERE CODE:

#include <GL/glut.h>

GLfloat xRotated, yRotated, zRotated; GLdouble radius=1;

void redisplayFunc(void) {

glMatrixMode(GL_MODELVIEW);

// clear the drawing buffer. glClear(GL_COLOR_BUFFER_BIT); // clear the identity matrix. glLoadIdentity();

```
// traslate the draw by z = -4.0

// Note this when you decrease z like -8.0 the drawing will looks far , or smaller.
glTranslatef(0.0,0.0,-4.5);

// Red color used to draw.

glColor3f(0.8, 0.2, 0.1);

// changing in transformation matrix.

// rotation about X axis

glRotatef(xRotated,1.0,0.0,0.0);

// rotation about Y axis glRotatef(yRotated,0.0,1.0,0.0);

// rotation about Z axis glRotatef(zRotated,0.0,0.0,1.0);

// scaling transfomation

glScalef(1.0,1.0,1.0);

// built-in (glut library) function , draw you a sphere. glutSolidSphere(radius,20,20);

// Flush buffers to screen

glFlush();

// sawp buffers called because we are using double buffering // glutSwapBuffers();

}

void reshapeFunc(int x, int y) {

if (y == 0 || x == 0) return; //Nothing is visible then, so return //Set a new projection
matrix glMatrixMode(GL_PROJECTION);

glLoadIdentity();

//Angle of view:40 degrees

//Near clipping plane distance: 0.5 //Far clipping plane distance: 20.0

gluPerspective(40.0,(GLdouble)x/(GLdouble)y,0.5,20.0);
```

```
glMatrixMode(GL_MODELVIEW);

glViewport(0,0,x,y); //Use the whole window for rendering }

void idleFunc(void) {

yRotated += 0.01;

redisplayFunc(); }

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

//Initialize GLUT

glutInit(&argc, argv);

//double buffering used to avoid flickering problem in animation
glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);

// window size

glutInitWindowSize(400,350);

// create the window

glutCreateWindow("Sphere Rotating Animation");
glPolygonMode(GL_FRONT_AND_BACK,GL_LINE);

xRotated = yRotated = zRotated = 30.0;

xRotated=33;

yRotated=40; glClearColor(0.0,0.0,0.0,0.0); //Assign the function used in events
glutDisplayFunc(redisplayFunc);

glutReshapeFunc(reshapeFunc); glutIdleFunc(idleFunc);

//Let start glut loop glutMainLoop();

return 0;

}
```

OUTPUT: