**Lab Taks-5**

Submission Guidelines-

* Rename the file with your serial number only
* Must submit within the announced time.
* Must include resources for all the section in the table

|  |
| --- |
| **Question-1**  Create an animation using two box that will move in the opposite direction. |
| **Graph Plot (Picture)-** |
| **Code-**  #include <iostream>  #include<GL/gl.h>  #include <GL/glut.h>  using namespace std;  float \_move1 = 0.0f;  float \_move2 = 0.0f;  void drawScene() {  glClear(GL\_COLOR\_BUFFER\_BIT);  glColor3d(1,0,0);  glLoadIdentity(); //Reset the drawing perspective  glMatrixMode(GL\_MODELVIEW);  glPushMatrix();  glTranslatef(\_move1, 0.0f, 0.0f);  glBegin(GL\_QUADS);  glVertex2f(-0.6, 0.8);  glVertex2f(-0.1, 0.8);  glVertex2f(-0.1, 0.2);  glVertex2f(-0.6, 0.2);  glEnd();  glPopMatrix();  glPushMatrix();  glTranslatef(\_move2, 0.0f, 0.0f);  glColor3d(0,1,0);  glBegin(GL\_QUADS);  glVertex2f(-0.6035087719298, -0.1593984962406);  glVertex2f(-0.1035087719298, -0.1593984962406);  glVertex2f(-0.1035087719298, -0.7593984962406);  glVertex2f(-0.6035087719298, -0.7593984962406);  glEnd();  glPopMatrix();  glutSwapBuffers();  }  void update(int value) {  \_move1 += .02;  if(\_move1 > 1.7)  {  \_move1 = -1.0;  }  \_move2 -= .02;  if(\_move2 < -1)  {  \_move2 = 1.0;  }  glutPostRedisplay();  glutTimerFunc(20, update, 0);  }  int main(int argc, char\*\* argv) {  glutInit(&argc, argv);  glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);  glutInitWindowSize(800, 800);  glutCreateWindow("Transformation");  glutDisplayFunc(drawScene);  gluOrtho2D(-1,1,-1,1);  glutTimerFunc(20, update, 0); //Add a timer  glutMainLoop();  return 0;  } |
| **Output Screenshot (Full Screen)-** |

|  |
| --- |
| **Question-2**  Design a car which will have rotating wheels. |
| **Graph Plot (Picture)-** |
| **Code-**  **#include <iostream>**  **#include<GL/gl.h>**  **#include <GL/glut.h>**  **#include <math.h>**  **#include <windows.h>**  **using namespace std;**  **float \_angle1 = 0.0f;**  **void drawScene() {**  **glClear(GL\_COLOR\_BUFFER\_BIT);**  **glClearColor(1,1,1,1);**  **glColor3d(1,0,0);**  **glLoadIdentity(); //Reset the drawing perspective**  **glMatrixMode(GL\_MODELVIEW);**  **glPushMatrix(); //car body**  **glBegin(GL\_POLYGON);**  **glColor3d(0,1,1);**  **glVertex2f(-0.4, 0.4);**  **glVertex2f(-0.3, 0.6);**  **glVertex2f(-0.1, 0.6);**  **glVertex2f(0.1, 0.4);**  **glVertex2f(-0.4, 0.4);**  **glEnd();**  **glBegin(GL\_POLYGON);**  **glColor3d(0,1,1);**  **glVertex2f(-0.6, 0.4);**  **glVertex2f(-0.4, 0.4);**  **glVertex2f(-0.4084507042254, 0.2992481203008);**  **glVertex2f(-0.5246478873239, 0.2992481203008);**  **glVertex2f(-0.6, 0.2);**  **glVertex2f(-0.6, 0.4);**  **glEnd();**  **glBegin(GL\_POLYGON);**  **glColor3d(0,1,1);**  **glVertex2f(-0.4, 0.4);**  **glVertex2f(-0.4084507042254, 0.2992481203008);**  **glVertex2f(-0.3397887323944, 0.209022556391);**  **glVertex2f(0, 0.2);**  **glVertex2f(0.0739436619718, 0.2902255639098);**  **glVertex2f(0.1, 0.4);**  **glVertex2f(-0.4, 0.4);**  **glEnd();**  **glBegin(GL\_POLYGON);**  **glColor3d(0,1,1);**  **glVertex2f(0.1, 0.4);**  **glVertex2f(0.3855633802817, 0.3984962406015);**  **glVertex2f(0.3838028169014, 0.2);**  **glVertex2f(0.2517605633803, 0.2);**  **glVertex2f(0.1848591549296, 0.2902255639098);**  **glVertex2f(0.0739436619718, 0.2902255639098);**  **glVertex2f(0.1, 0.4);**  **glEnd();**  **glPopMatrix();**  **glPushMatrix(); // first tier**  **glTranslatef(-0.47,0.18,0);**  **glRotatef(\_angle1, 0.0f, 0.0f,1.0f);**  **glTranslatef(0.47,-0.18,0);**  **glBegin(GL\_POLYGON);**  **for(int i=0;i<200;i++)**  **{**  **glColor3f(0,0,0);**  **float pi=3.1416;**  **float A=(i\*2\*pi)/200;**  **float r=0.0903 ;**  **float x = r \* cos(A);**  **float y = r \* sin(A);**  **glVertex2f(x-0.4683098591549,y+0.181954887218 );**  **}**  **glEnd();**  **glBegin(GL\_LINES);**  **for(int i=0;i<200;i++)**  **{**  **glColor3f(0,0,0);**  **float pi=3.1416;**  **float A=(i\*2\*pi)/200;**  **float r=0.097 ;**  **float x = r \* cos(A);**  **float y = r \* sin(A);**  **glVertex2f(x-0.4683098591549,y+0.181954887218 );**  **}**  **glEnd();**  **glPopMatrix();**  **glPushMatrix(); //2nd tier**  **glTranslatef(0.13,0.17,0);**  **glRotatef(\_angle1, 0.0f, 0.0f,1.0f);**  **glTranslatef(-0.13,-0.17,0);**  **glBegin(GL\_POLYGON); //tier**  **for(int i=0;i<200;i++)**  **{**  **glColor3f(0,0,0);**  **float pi=3.1416;**  **float A=(i\*2\*pi)/200;**  **float r=0.0903 ;**  **float x = r \* cos(A);**  **float y = r \* sin(A);**  **glVertex2f(x+0.1267605633803,y+0.1729323308271 );**  **}**  **glEnd();**  **glBegin(GL\_LINES); //tier**  **for(int i=0;i<200;i++)**  **{**  **glColor3f(0,0,0);**  **float pi=3.1416;**  **float A=(i\*2\*pi)/200;**  **float r=0.097 ;**  **float x = r \* cos(A);**  **float y = r \* sin(A);**  **glVertex2f(x+0.1267605633803,y+0.1729323308271 );**  **}**  **glEnd();**  **glPopMatrix();**  **glutSwapBuffers();**  **}**  **void update(int value) {**  **\_angle1+=2.0f;**  **if (\_angle1 > 360) {**  **\_angle1 -= 360; // Keep the angle within 0-360 degrees**  **}**  **glutPostRedisplay(); //Notify GLUT that the display has changed**  **glutTimerFunc(20, update, 0); //Notify GLUT to call update again in 25 milliseconds**  **}**  **int main(int argc, char\*\* argv) {**  **glutInit(&argc, argv);**  **glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);**  **glutInitWindowSize(800, 800);**  **glutCreateWindow("Transformation");**  **glutDisplayFunc(drawScene);**  **gluOrtho2D(-1,1,-1,1);**  **glutTimerFunc(20, update, 0); //Add a timer**  **glutMainLoop();**  **return 0;**  **}** |
| **Output Screenshot (Full Screen)-** |

|  |
| --- |
| **Question-3**  Now move your car of question-2 from left to right in a loop. |
| **Graph Plot (Picture)-** |
| **Code-**  **#include <iostream>**  **#include<GL/gl.h>**  **#include <GL/glut.h>**  **#include <math.h>**  **#include <windows.h>**  **using namespace std;**  **float \_angle1 = 0.0f;**  **float \_move = 0.0f;**  **void drawScene() {**  **glClear(GL\_COLOR\_BUFFER\_BIT);**  **glClearColor(1,1,1,1);**  **glColor3d(1,0,0);**  **glLoadIdentity(); //Reset the drawing perspective**  **glMatrixMode(GL\_MODELVIEW);**  **glPushMatrix();//car body**  **glTranslatef(\_move, 0.0f, 0.0f);**  **glBegin(GL\_POLYGON);**  **glColor3d(0,1,1);**  **glVertex2f(-0.4, 0.4);**  **glVertex2f(-0.3, 0.6);**  **glVertex2f(-0.1, 0.6);**  **glVertex2f(0.1, 0.4);**  **glVertex2f(-0.4, 0.4);**  **glEnd();**  **glBegin(GL\_POLYGON);**  **glColor3d(0,1,1);**  **glVertex2f(-0.6, 0.4);**  **glVertex2f(-0.4, 0.4);**  **glVertex2f(-0.4084507042254, 0.2992481203008);**  **glVertex2f(-0.5246478873239, 0.2992481203008);**  **glVertex2f(-0.6, 0.2);**  **glVertex2f(-0.6, 0.4);**  **glEnd();**  **glBegin(GL\_POLYGON);**  **glColor3d(0,1,1);**  **glVertex2f(-0.4, 0.4);**  **glVertex2f(-0.4084507042254, 0.2992481203008);**  **glVertex2f(-0.3397887323944, 0.209022556391);**  **glVertex2f(0, 0.2);**  **glVertex2f(0.0739436619718, 0.2902255639098);**  **glVertex2f(0.1, 0.4);**  **glVertex2f(-0.4, 0.4);**  **glEnd();**  **glBegin(GL\_POLYGON);**  **glColor3d(0,1,1);**  **glVertex2f(0.1, 0.4);**  **glVertex2f(0.3855633802817, 0.3984962406015);**  **glVertex2f(0.3838028169014, 0.2);**  **glVertex2f(0.2517605633803, 0.2);**  **glVertex2f(0.1848591549296, 0.2902255639098);**  **glVertex2f(0.0739436619718, 0.2902255639098);**  **glVertex2f(0.1, 0.4);**  **glEnd();**  **glPopMatrix();**  **glPushMatrix(); // first tier**  **glTranslatef(-0.47,0.18,0);**  **glRotatef(\_angle1, 0.0f, 0.0f,1.0f);**  **glTranslatef(0.47,-0.18,0);**  **glBegin(GL\_POLYGON);**  **for(int i=0;i<200;i++)**  **{**  **glColor3f(0,0,0);**  **float pi=3.1416;**  **float A=(i\*2\*pi)/200;**  **float r=0.0903 ;**  **float x = r \* cos(A);**  **float y = r \* sin(A);**  **glVertex2f(x-0.4683098591549,y+0.181954887218 );**  **}**  **glEnd();**  **glBegin(GL\_LINES);**  **for(int i=0;i<200;i++)**  **{**  **glColor3f(0,0,0);**  **float pi=3.1416;**  **float A=(i\*2\*pi)/200;**  **float r=0.097 ;**  **float x = r \* cos(A);**  **float y = r \* sin(A);**  **glVertex2f(x-0.4683098591549,y+0.181954887218 );**  **}**  **glEnd();**  **glPopMatrix();**  **glPushMatrix(); //2nd tier**  **glTranslatef(0.13,0.17,0);**  **glRotatef(\_angle1, 0.0f, 0.0f,1.0f);**  **glTranslatef(-0.13,-0.17,0);**  **glBegin(GL\_POLYGON); //tier**  **for(int i=0;i<200;i++)**  **{**  **glColor3f(0,0,0);**  **float pi=3.1416;**  **float A=(i\*2\*pi)/200;**  **float r=0.0903 ;**  **float x = r \* cos(A);**  **float y = r \* sin(A);**  **glVertex2f(x+0.1267605633803,y+0.1729323308271 );**  **}**  **glEnd();**  **glBegin(GL\_LINES); //tier**  **for(int i=0;i<200;i++)**  **{**  **glColor3f(0,0,0);**  **float pi=3.1416;**  **float A=(i\*2\*pi)/200;**  **float r=0.097 ;**  **float x = r \* cos(A);**  **float y = r \* sin(A);**  **glVertex2f(x+0.1267605633803,y+0.1729323308271 );**  **}**  **glEnd();**  **glPopMatrix();**  **glutSwapBuffers();**  **}**  **void update(int value) {**  **\_angle1+=2.0f;**  **if (\_angle1 > 360) {**  **\_angle1 -= 360; // Keep the angle within 0-360 degrees**  **}**  **\_move += .02;**  **if(\_move > 1)**  **{**  **\_move = -1.0;**  **}**  **glutPostRedisplay(); //Notify GLUT that the display has changed**  **glutTimerFunc(20, update, 0); //Notify GLUT to call update again in 25 milliseconds**  **}**  **int main(int argc, char\*\* argv) {**  **glutInit(&argc, argv);**  **glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);**  **glutInitWindowSize(800, 800);**  **glutCreateWindow("Transformation");**  **glutDisplayFunc(drawScene);**  **gluOrtho2D(-1,1,-1,1);**  **glutTimerFunc(20, update, 0); //Add a timer**  **glutMainLoop();**  **return 0;**  **}** |
| **Output Screenshot (Full Screen)-** |

|  |
| --- |
| **Question-4**  Design a windmill with rotating blades |
| **Graph Plot (Picture)-** |
| **Code-**  **#include <iostream>**  **#include<GL/gl.h>**  **#include <GL/glut.h>**  **#include <windows.h>**  **#include <math.h>**  **using namespace std;**  **float \_angle1 = 0.0f;**  **void drawScene() {**  **glClear(GL\_COLOR\_BUFFER\_BIT);**  **glColor3d(1,1,1);**  **glLoadIdentity(); //Reset the drawing perspective**  **glMatrixMode(GL\_MODELVIEW);**  **glPushMatrix();**  **glBegin(GL\_QUADS);//stand**  **glVertex2f(-0.4, 0.2);**  **glVertex2f(-0.3, 0.2);**  **glVertex2f(-0.3, -0.6);**  **glVertex2f(-0.4, -0.6);**  **glEnd();**  **glPopMatrix();**  **glPushMatrix();**  **glTranslatef(-0.35,0.2,0);**  **glRotatef(\_angle1, 0.0f, 0.0f,1.0f);**  **glTranslatef(0.35,-0.2,0);**  **glBegin(GL\_POLYGON);// circle**  **for(int i=0;i<200;i++)**  **{**  **glColor3f(1.0,1.0,1.0);**  **float pi=3.1416;**  **float A=(i\*2\*pi)/200;**  **float r=0.0668;**  **float x = r \* cos(A);**  **float y = r \* sin(A);**  **glVertex2f(x-0.35,y+0.2 );**  **}**  **glEnd();**  **glBegin(GL\_QUADS); // left bled**  **glVertex2f(-0.35,0.2);**  **glVertex2f(-0.4824561403509, -0.0255639097744);**  **glVertex2f(-0.6561403508772, -0.1157894736842);**  **glVertex2f(-0.5385964912281, 0.0887218045113);**  **glEnd();**  **glBegin(GL\_QUADS); // middle bled**  **glVertex2f(-0.35,0.2);**  **glVertex2f(-0.3894736842105, 0.4706766917293);**  **glVertex2f(-0.3482456140351, 0.6390977443609);**  **glVertex2f(-0.3087719298246, 0.4676691729323);**  **glEnd();**  **glBegin(GL\_QUADS); // right bled**  **glVertex2f(-0.35,0.2);**  **glVertex2f(-0.1657894736842, 0.0887218045113);**  **glVertex2f(-0.040350877193, -0.0706766917293);**  **glVertex2f(-0.2, 0);**  **glEnd();**  **glPopMatrix();**  **glutSwapBuffers();**  **}**  **void update(int value) {**  **\_angle1+=2.0f;**  **glutPostRedisplay(); //Notify GLUT that the display has changed**  **glutTimerFunc(20, update, 0); //Notify GLUT to call update again in 25 milliseconds**  **}**  **int main(int argc, char\*\* argv) {**  **glutInit(&argc, argv);**  **glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);**  **glutInitWindowSize(800, 800);**  **glutCreateWindow("Transformation");**  **gluOrtho2D(-1,1,-1,1);**  **glutDisplayFunc(drawScene);**  **glutTimerFunc(20, update, 0); //Add a timer**  **glutMainLoop();**  **return 0;**  **}** |
| **Output Screenshot (Full Screen)-** |