**Lab Taks-5**

|  |
| --- |
| **Question-1**  Create an animation using two box that will move in the opposite direction. |
| **Graph Plot (Picture)-** |
| **Code-**  #include <GL/glut.h> float box1\_x = -0.9f; float box2\_x = 0.9f; float speed1 = 0.01f; float speed2 = -0.01f; void drawBox1() {     glColor3f(0.0f, 0.0f, 0.0f);     glBegin(GL\_QUADS);         glVertex2f(box1\_x, 0.2f);         glVertex2f(box1\_x + 0.2f, 0.2f);         glVertex2f(box1\_x + 0.2f, 0.4f);         glVertex2f(box1\_x, 0.4f);     glEnd(); } void drawBox2() {     glColor3f(0.0f, 0.0f, 0.0f);     glBegin(GL\_QUADS);         glVertex2f(box2\_x, -0.4f);         glVertex2f(box2\_x - 0.2f, -0.4f);         glVertex2f(box2\_x - 0.2f, -0.2f);         glVertex2f(box2\_x, -0.2f);     glEnd(); } void update (int value) {     box1\_x += speed1;     box2\_x += speed2;     if (box1\_x > 1.0f) box1\_x = -1.0f;     if (box2\_x < -1.0f) box2\_x = 1.0f;     glutPostRedisplay ();     glutTimerFunc (16, update, 0); } void display () {     glClear(GL\_COLOR\_BUFFER\_BIT);     drawBox1();     drawBox2();     glutSwapBuffers(); } void init() {     glClearColor(0.9f, 0.9f, 0.9f, 1.0f); } int main (int argc, char\*\* argv) {     glutInit(&argc, argv);     glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);     glutInitWindowSize(600, 400);     glutCreateWindow("Two Boxes");     init();     glutDisplayFunc(display);     glutTimerFunc(0, update, 0);     glutMainLoop();     return 0; } |
| **Output Screenshot (Full Screen)-** |

|  |
| --- |
| **Question-2**  Design a car which will have rotating wheels. |
| **Graph Plot (Picture)-** |
| **Code-**  #include <GL/glut.h> #include <cmath>  float wheelAngle = 0.0f;  void drawwheel(float x, float y, float radius) {     glPushMatrix();     glTranslatef(x, y, 0.0f);     glRotatef(wheelAngle, 0.0f, 0.0f, 1.0f);     glColor3f(0.1f, 0.1f, 0.1f);     glBegin(GL\_POLYGON);     for (int i = 0; i < 360; i++) {         float rad = i \* 3.14159 / 180;         glVertex2f(cos(rad) \* radius, sin(rad) \* radius);     }     glEnd();     glColor3f(1.0f, 1.0f, 1.0f);     glBegin(GL\_LINES);     for (int i = 0; i < 6; i++) {         float rad = i \* 60 \* 3.14159 / 180;         glVertex2f(0.0f, 0.0f);         glVertex2f(cos(rad) \* radius, sin(rad) \* radius);     }     glEnd();     glPopMatrix(); }  void drawcarBody() {     // Changed car body color to green     glColor3f(0.0f, 1.0f, 0.0f);  // green     glBegin(GL\_POLYGON);     glVertex2f(-0.6f, -0.2f);     glVertex2f(0.6f, -0.2f);     glVertex2f(0.6f, 0.0f);     glVertex2f(-0.6f, 0.0f);     glEnd();      glBegin(GL\_POLYGON);     glVertex2f(-0.4f, 0.0f);     glVertex2f(0.4f, 0.0f);     glVertex2f(0.2f, 0.2f);     glVertex2f(-0.2f, 0.2f);     glEnd(); }  void window() {     glColor3f(0.5f, 0.8f, 1.0f);     glBegin(GL\_POLYGON);     glVertex2f(-0.18f, 0.05f);     glVertex2f(-0.05f, 0.05f);     glVertex2f(-0.05f, 0.15f);     glVertex2f(-0.18f, 0.15f);     glEnd();     glBegin(GL\_POLYGON);     glVertex2f(0.05f, 0.05f);     glVertex2f(0.18f, 0.05f);     glVertex2f(0.18f, 0.15f);     glVertex2f(0.05f, 0.15f);     glEnd(); }  void display() {     glClear(GL\_COLOR\_BUFFER\_BIT);     drawcarBody();     drawwheel(-0.4f, -0.2f, 0.1f);     drawwheel(0.4f, -0.2f, 0.1f);     window();      glutSwapBuffers(); }  void update(int value) {     wheelAngle += 5.0f;     if (wheelAngle > 360) wheelAngle -= 360;     glutPostRedisplay();     glutTimerFunc(30, update, 0); }  void init() {     glClearColor(1.0f, 1.0f, 1.0f, 1.0f);     glMatrixMode(GL\_PROJECTION);     glLoadIdentity();     gluOrtho2D(-1.0f, 1.0f, -0.5f, 0.5f); }  int main(int argc, char\*\* argv) {     glutInit(&argc, argv);     glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);     glutInitWindowSize(800, 400);     glutCreateWindow("Rotating Wheels Car");     init();     glutDisplayFunc(display);     glutTimerFunc(0, update, 0);     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 <GL/glut.h> #include <cmath>  float wheelAngle = 0.0f; float carPosX = -1.0f;      // Starting position on the left float carSpeed = 0.01f;     // Speed of the car movement  void drawwheel(float x, float y, float radius) {     glPushMatrix();     glTranslatef(x, y, 0.0f);     glRotatef(wheelAngle, 0.0f, 0.0f, 1.0f);     glColor3f(0.1f, 0.1f, 0.1f);     glBegin(GL\_POLYGON);     for (int i = 0; i < 360; i++) {         float rad = i \* 3.14159 / 180;         glVertex2f(cos(rad) \* radius, sin(rad) \* radius);     }     glEnd();     glColor3f(1.0f, 1.0f, 1.0f);     glBegin(GL\_LINES);     for (int i = 0; i < 6; i++) {         float rad = i \* 60 \* 3.14159 / 180;         glVertex2f(0.0f, 0.0f);         glVertex2f(cos(rad) \* radius, sin(rad) \* radius);     }     glEnd();     glPopMatrix(); }  void drawcarBody() {     glColor3f(0.0f, 1.0f, 0.0f);  // green car body     glBegin(GL\_POLYGON);     glVertex2f(-0.6f, -0.2f);     glVertex2f(0.6f, -0.2f);     glVertex2f(0.6f, 0.0f);     glVertex2f(-0.6f, 0.0f);     glEnd();      glBegin(GL\_POLYGON);     glVertex2f(-0.4f, 0.0f);     glVertex2f(0.4f, 0.0f);     glVertex2f(0.2f, 0.2f);     glVertex2f(-0.2f, 0.2f);     glEnd(); }  void window() {     glColor3f(0.5f, 0.8f, 1.0f);     glBegin(GL\_POLYGON);     glVertex2f(-0.18f, 0.05f);     glVertex2f(-0.05f, 0.05f);     glVertex2f(-0.05f, 0.15f);     glVertex2f(-0.18f, 0.15f);     glEnd();     glBegin(GL\_POLYGON);     glVertex2f(0.05f, 0.05f);     glVertex2f(0.18f, 0.05f);     glVertex2f(0.18f, 0.15f);     glVertex2f(0.05f, 0.15f);     glEnd(); }  void display() {     glClear(GL\_COLOR\_BUFFER\_BIT);      glPushMatrix();     glTranslatef(carPosX, 0.0f, 0.0f);  // Move car horizontally      drawcarBody();     drawwheel(-0.4f, -0.2f, 0.1f);     drawwheel(0.4f, -0.2f, 0.1f);     window();      glPopMatrix();      glutSwapBuffers(); }  void update(int value) {     wheelAngle += 5.0f;     if (wheelAngle > 360) wheelAngle -= 360;      carPosX += carSpeed;  // Update car horizontal position     if (carPosX > 1.2f) { // Reset position if off screen to the right         carPosX = -1.2f;     }      glutPostRedisplay();     glutTimerFunc(30, update, 0); }  void init() {     glClearColor(1.0f, 1.0f, 1.0f, 1.0f);     glMatrixMode(GL\_PROJECTION);     glLoadIdentity();     gluOrtho2D(-1.0f, 1.0f, -0.5f, 0.5f); }  int main(int argc, char\*\* argv) {     glutInit(&argc, argv);     glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);     glutInitWindowSize(800, 400);     glutCreateWindow("Moving Rotating Wheels Car");     init();     glutDisplayFunc(display);     glutTimerFunc(0, update, 0);     glutMainLoop();     return 0; } |
| **Output Screenshot (Full Screen)-** |

|  |
| --- |
| **Question-4**  Design a windmill with rotating blades |
| **Graph Plot (Picture)-** |
| **Code-**  **#include <GL/glut.h>**  **#include <cmath>**  **float bladeAngle = 0.0f;**  **void drawBase() {**  **glColor3f(0.4f, 0.2f, 0.0f);**  **glBegin(GL\_POLYGON);**  **glVertex2f(-0.1f, -0.4f);**  **glVertex2f(0.1f, -0.4f);**  **glVertex2f(0.04f, 0.2f);**  **glVertex2f(-0.04f, 0.2f);**  **glEnd();**  **}**  **void drawNacelle() {**  **glColor3f(0.6f, 0.6f, 0.6f);**  **glBegin(GL\_POLYGON);**  **glVertex2f(-0.05f, 0.2f);**  **glVertex2f(0.05f, 0.2f);**  **glVertex2f(0.05f, 0.25f);**  **glVertex2f(-0.05f, 0.25f);**  **glEnd();**  **}**  **void drawBlade(float angle) {**  **glPushMatrix();**  **glRotatef(angle, 0.0f, 0.0f, 1.0f);**  **glColor3f(0.9f, 0.1f, 0.1f);**  **glBegin(GL\_POLYGON);**  **glVertex2f(0.0f, 0.0f);**  **glVertex2f(0.04f, 0.5f);**  **glVertex2f(-0.04f, 0.5f);**  **glEnd();**  **glPopMatrix();**  **}**  **void drawBlades() {**  **glPushMatrix();**  **glTranslatef(0.0f, 0.25f, 0.0f);**  **glRotatef(bladeAngle, 0.0f, 0.0f, 1.0f);**  **drawBlade(0.0f);**  **drawBlade(120.0f);**  **drawBlade(240.0f);**  **glPopMatrix();**  **}**  **void update(int value) {**  **bladeAngle += 2.0f;**  **if (bladeAngle > 360.0f)**  **bladeAngle -= 360.0f;**  **glutPostRedisplay();**  **glutTimerFunc(16, update, 0);**  **}**  **void display() {**  **glClear(GL\_COLOR\_BUFFER\_BIT);**  **drawBase();**  **drawNacelle();**  **drawBlades();**  **glutSwapBuffers();**  **}**  **void init() {**  **glClearColor(0.8f, 0.9f, 1.0f, 1.0f);**  **glMatrixMode(GL\_PROJECTION);**  **glLoadIdentity();**  **gluOrtho2D(-1.0f, 1.0f, -1.0f, 1.0f);**  **}**  **int main(int argc, char\*\* argv) {**  **glutInit(&argc, argv);**  **glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);**  **glutInitWindowSize(800, 800);**  **glutCreateWindow("Windmill with rotating Blades");**  **init();**  **glutDisplayFunc(display);**  **glutTimerFunc(0, update, 0);**  **glutMainLoop();**  **return 0;**  **}** |
| **Output Screenshot (Full Screen)-** |