Lab Taks-5

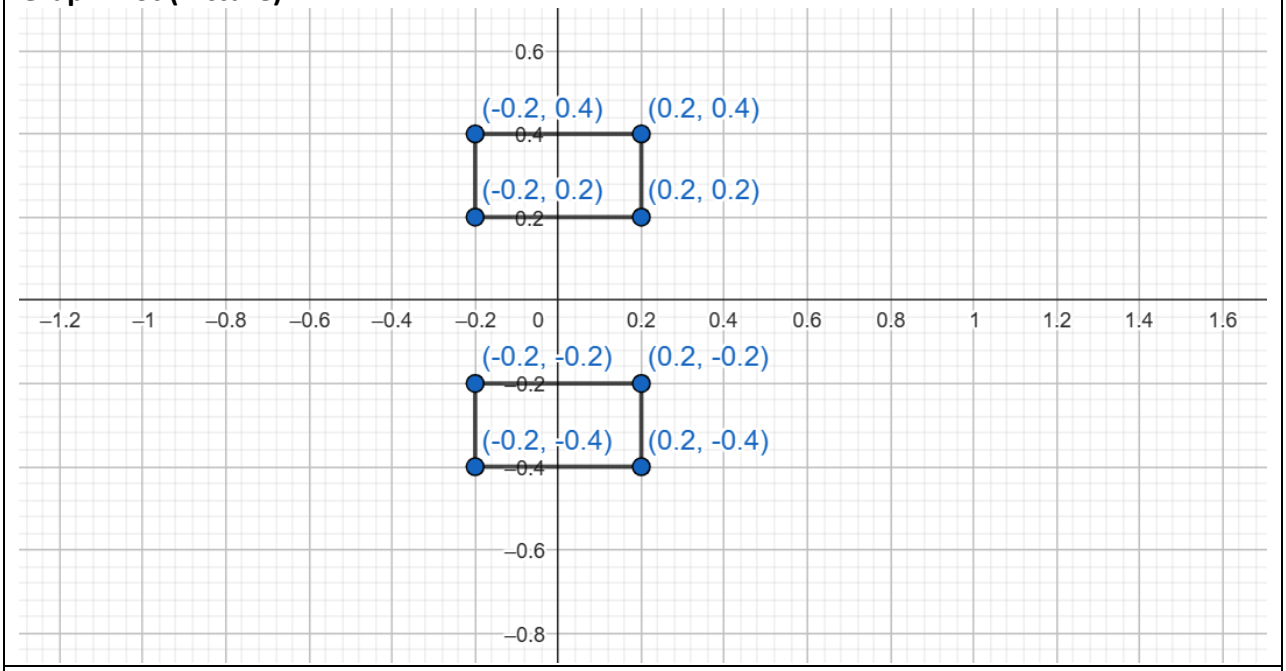
Submission Guidelines-

- Rename the file to your id only. If your id is 18-XXXXX-1, then the file name must be 18-XXXXX-1.docx.
- 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)-



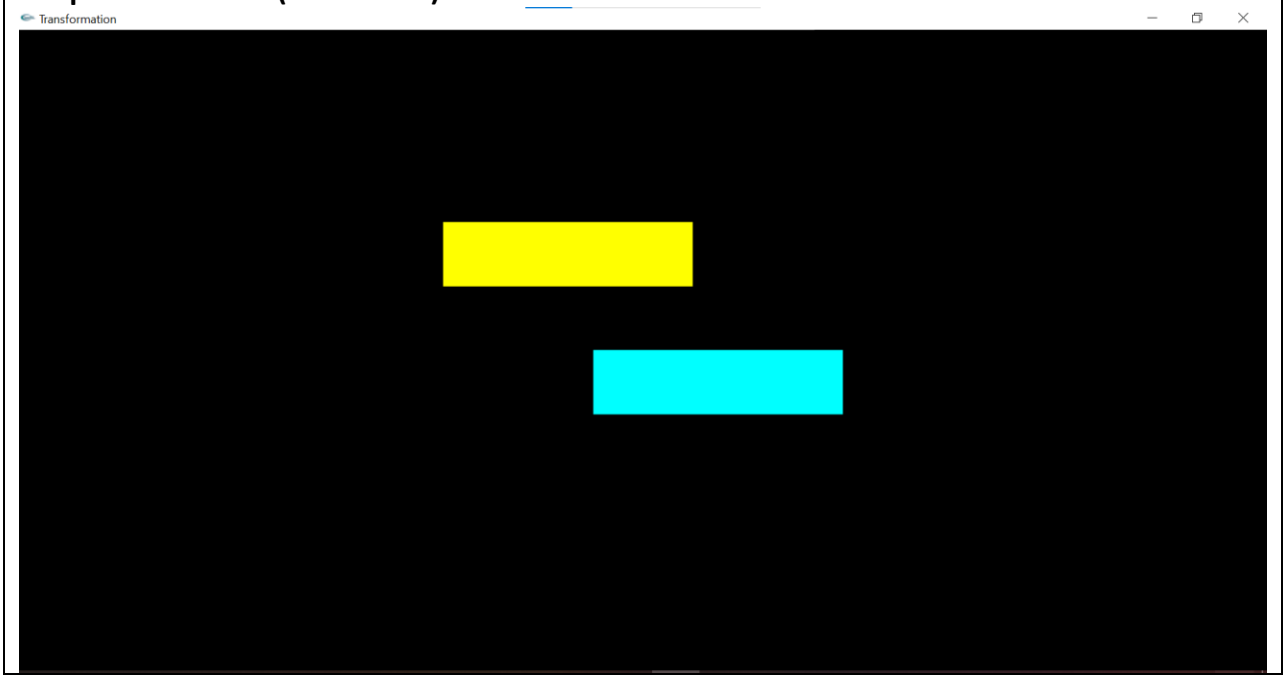
Code-

```
#include <iostream>
#include<GL/gl.h>
#include <GL/glut.h>
using namespace std;

float _move = 0.0f;

void Box(int x,int y,int z){

    glColor3d(x, y, z);
    glBegin(GL_QUADS);
```

```
	glVertex2f(-0.2f, 0.2f);
	glVertex2f(0.2f, 0.20f);
	glVertex2f(0.2f, 0.4f);
	glVertex2f(-0.2f, 0.4f);
	glEnd();
}

void drawScene() {
	glClear(GL_COLOR_BUFFER_BIT);

	glLoadIdentity();
	glMatrixMode(GL_MODELVIEW);

	glPushMatrix();
	glTranslatef(_move, 0.0f, 0.0f);
	Box(1,1,0);
	glPopMatrix();
	glTranslatef(-_move, 0.0f, 0.0f);
	glTranslatef(0.0f, -0.40f, 0.0f);
	Box(0,1,1);
	glPopMatrix();
	glutSwapBuffers();
}



void update(int value) {



	 _move += .02;
	if(_move > 1.3)
	{
	_move = -1.3;
	}
	glutPostRedisplay();
	glutTimerFunc(30, update, 0);
	}



int main(int argc, char** argv) {
	glutInit(&argc, argv);
	glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB);
```

```
    glutInitWindowSize(800, 800);
    glutCreateWindow("Transformation");
    glutDisplayFunc(drawScene);
    glutTimerFunc(30, update, 0); //Add a timer
    glutMainLoop();
    return 0;
    }
```

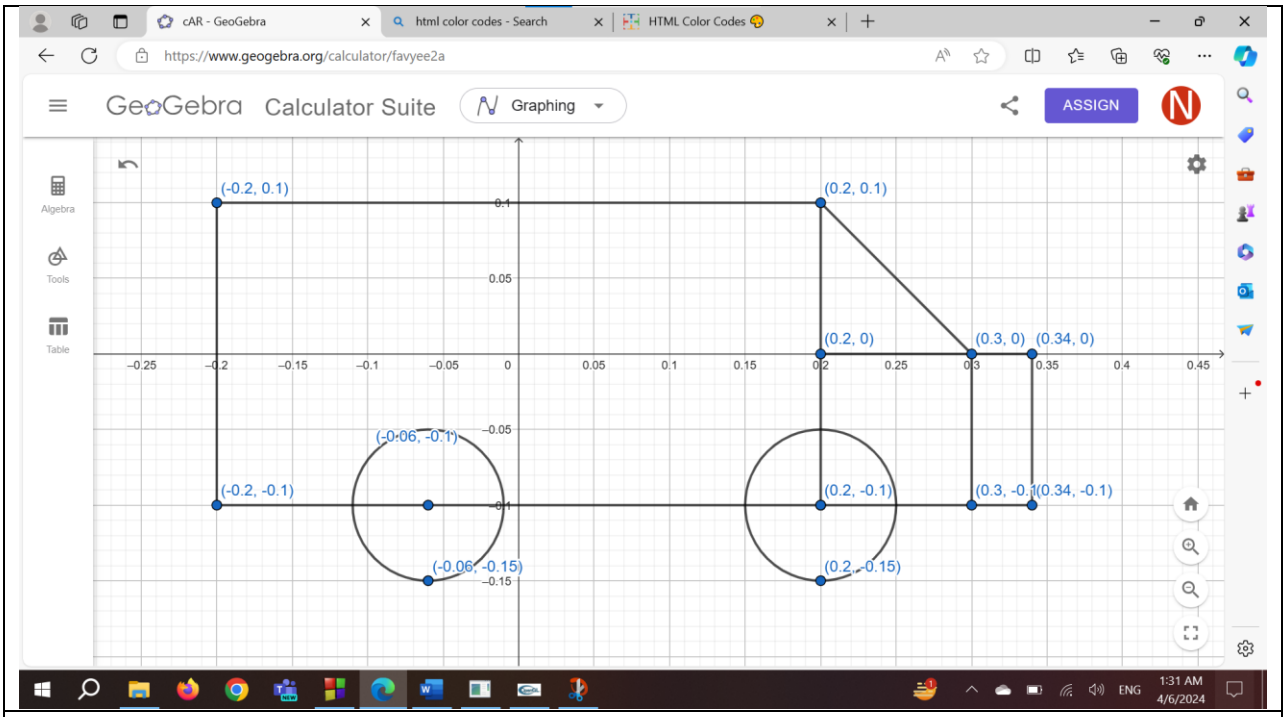
Output Screenshot (Full Screen)-



Question-2

Design a car which will have rotating wheels.

Graph Plot (Picture)-



Code-

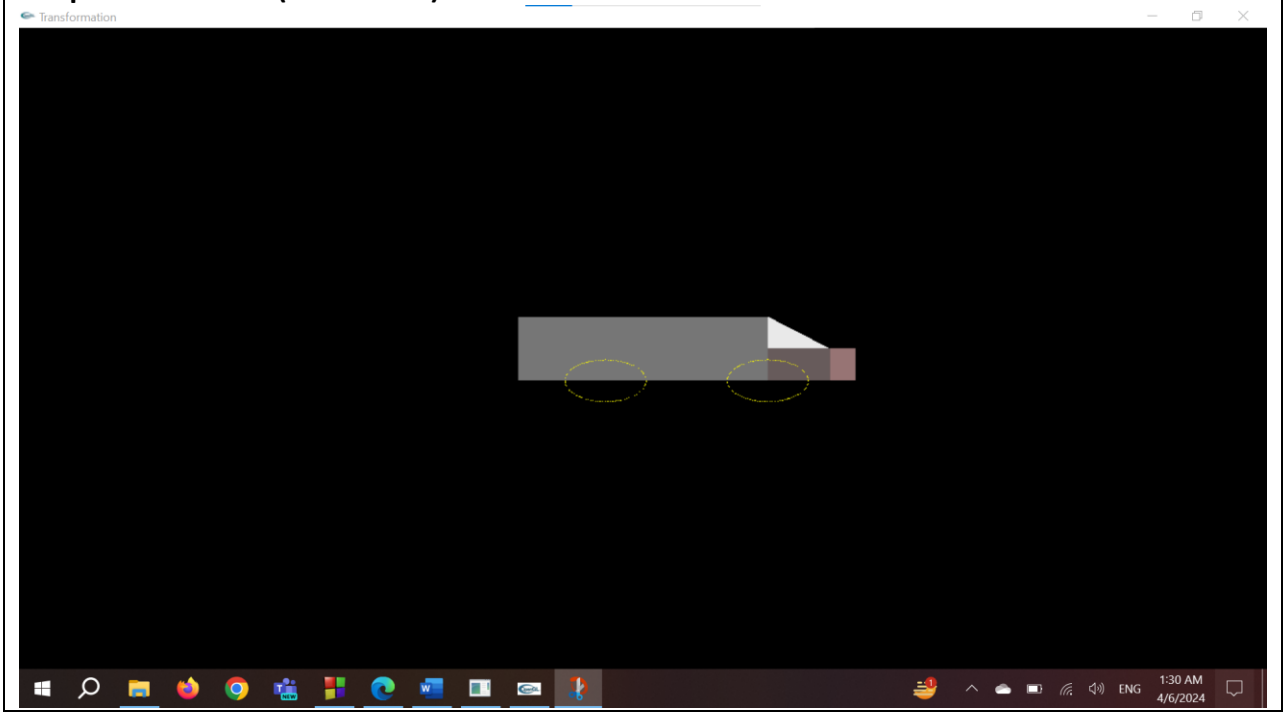
```
#include <iostream>
#include<GL/gl.h>
#include <GL/glut.h>
#include <math.h>
using namespace std;
float _angle1=0.0f;

void wheel()
   {
   glLoadIdentity();
   glMatrixMode(GL_MODELVIEW);
   glPushMatrix();
   glTranslatef(-0.06,-0.10,0);
   glRotatef(_angle1, 0.0f, 0.0f,1.0f);
   glBegin(GL_LINES);
   for(int i=0;i<200;i++)
   {
   glColor3f(1.0,1.0,0.0);
   float pi=3.1416;
   float A=(i*2*pi)/200;
   float r=0.065;
   float x = r * cos(A);
   float y = r * sin(A);
   glVertex2f(x,y );
   }
```

```
    glEnd();
    glPopMatrix();



    glLoadIdentity();
    glMatrixMode(GL_MODELVIEW);
    glPushMatrix();
    glTranslatef(0.2,-0.10,0);
    glRotatef(_angle1, 0.0f, 0.0f,1.0f);
    glBegin(GL_LINES);
    for(int i=0;i<200;i++)
    {
    glColor3f(1.0,1.0,0.0);
    float pi=3.1416;
    float A=(i*2*pi)/200;
    float r=0.065;
    float x = r * cos(A);
    float y = r * sin(A);
    glVertex2f(x,y );
    }
    glEnd();
    glPopMatrix();
    }

void Car(){
    glColor3ub(118,118,118);
    glBegin(GL_QUADS);
    glVertex2f(-0.2f, -0.10f);
    glVertex2f(0.2f, -0.10f);
    glVertex2f(0.2f, 0.10f);
    glVertex2f(-0.2f, 0.10f);
    glEnd();

    glColor3ub(233,233,233);
    glBegin(GL_TRIANGLES);
    glVertex2f(0.2f, 0.0f);
    glVertex2f(0.3f, 0.0f);
    glVertex2f(0.2f, 0.10f);
    glEnd();

    glColor3ub(103,91,91);
    glBegin(GL_QUADS);
    glVertex2f(0.2f, -0.10f);
```

```
   glVertex2f(0.3f, -0.10f);
   glVertex2f(0.3f, 0.0f);
   glVertex2f(0.2f, 0.0f);
   glEnd();

   glColor3ub(151,116,116);
   glBegin(GL_QUADS);
   glVertex2f(0.3f, -0.10f);
   glVertex2f(0.34f, -0.10f);
   glVertex2f(0.34f, 0.0f);
   glVertex2f(0.3f, 0.0f);
   glEnd();
}

void drawScene() {
   glClear(GL_COLOR_BUFFER_BIT);
   glColor3d(1,0,0);
   Car();
   wheel();
   glutSwapBuffers();
   }


void update1(int value) {
   _angle1+=2.0f;
   if(_angle1 > 360.0)
   {
   _angle1-=360;
   }
   glutPostRedisplay();

   glutTimerFunc(20, update1, 0);
   }



int main(int argc, char** argv) {
   glutInit(&argc, argv);
   glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB);
   glutInitWindowSize(800, 800);
   glutCreateWindow("Transformation");
   glutDisplayFunc(drawScene);
   glutTimerFunc(20, update1, 0);
   glutMainLoop();
```

```
return 0;
}
```

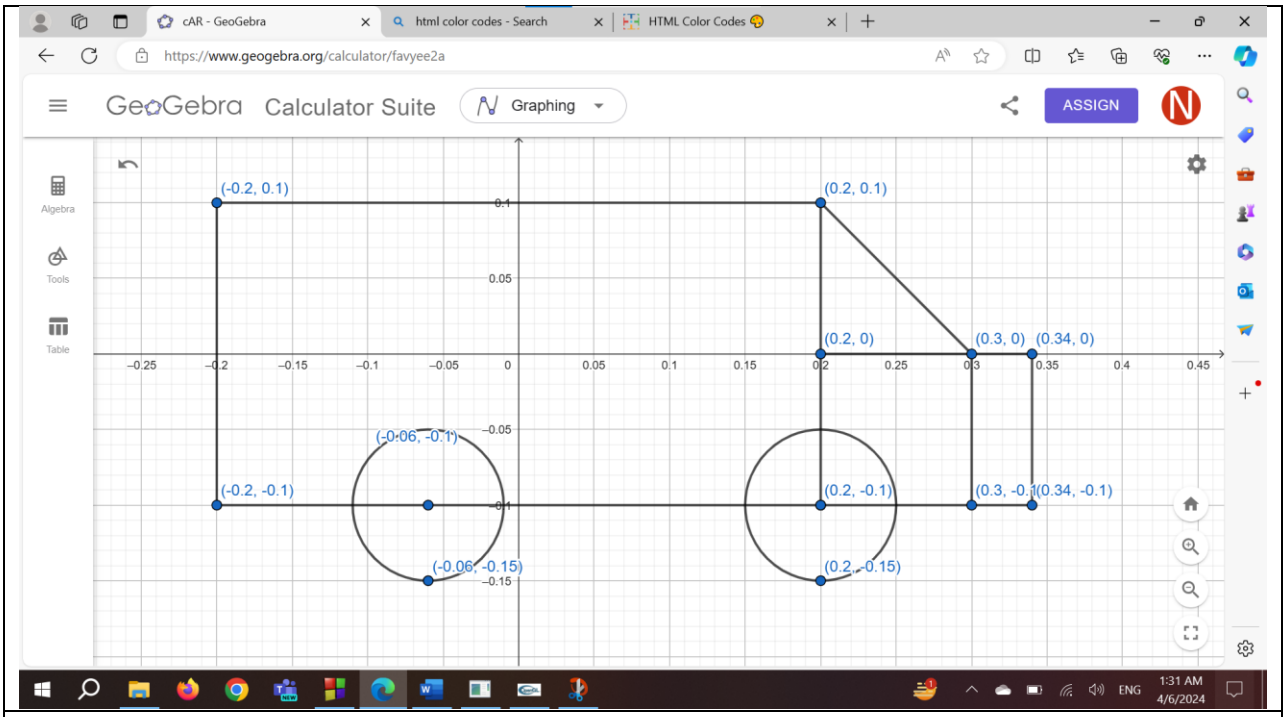
Output Screenshot (Full Screen)-



Question-3

Now move your car of question-2 from left to right in a loop.

Graph Plot (Picture)-



Code-

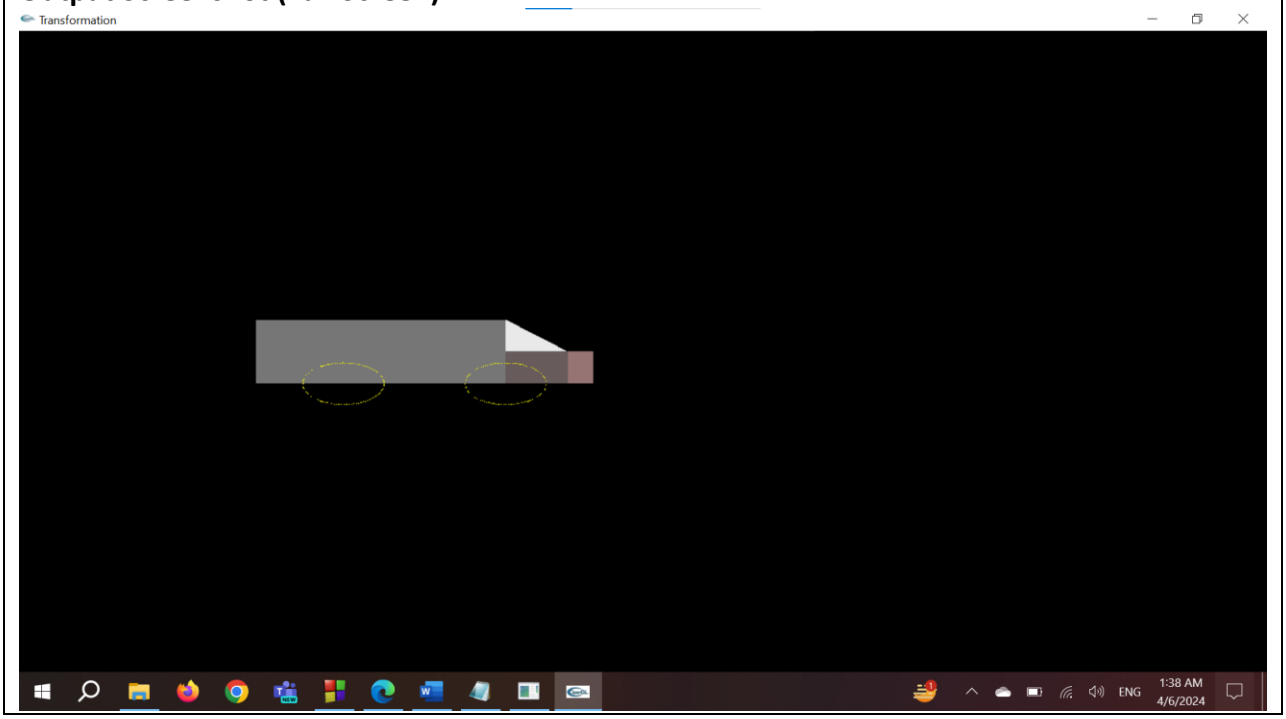
```
#include <iostream>
#include<GL/gl.h>
#include <GL/glut.h>
#include <math.h>
using namespace std;


float _move = 0.0f;
float _angle1=0.0f;

void wheel()
   {
   glLoadIdentity();
   glMatrixMode(GL_MODELVIEW);
   glPushMatrix();
   glTranslatef(_move, 0.0f, 0.0f);
   glTranslatef(-0.06,-0.10,0);
   glRotatef(_angle1, 0.0f, 0.0f,1.0f);
   glBegin(GL_LINES);
   for(int i=0;i<200;i++)
   {
   glColor3f(1.0,1.0,0.0);
   float pi=3.1416;
   float A=(i*2*pi)/200;
   float r=0.065;
```

```
    float x = r * cos(A);
    float y = r * sin(A);
    glVertex2f(x,y );
    }
    glEnd();
    glPopMatrix();



    glLoadIdentity();
    glMatrixMode(GL_MODELVIEW);
    glPushMatrix();
    glTranslatef(_move, 0.0f, 0.0f);
    glTranslatef(0.2,-0.10,0);
    glRotatef(_angle1, 0.0f, 0.0f,1.0f);
    glBegin(GL_LINES);
    for(int i=0;i<200;i++)
    {
    glColor3f(1.0,1.0,0.0);
    float pi=3.1416;
    float A=(i*2*pi)/200;
    float r=0.065;
    float x = r * cos(A);
    float y = r * sin(A);
    glVertex2f(x,y );
    }
    glEnd();
    glPopMatrix();
    }

void Car(){

    glColor3ub(118,118,118);
    glBegin(GL_QUADS);
    glVertex2f(-0.2f, -0.10f);
    glVertex2f(0.2f, -0.10f);
    glVertex2f(0.2f, 0.10f);
    glVertex2f(-0.2f, 0.10f);
    glEnd();

    glColor3ub(233,233,233);
    glBegin(GL_TRIANGLES);
    glVertex2f(0.2f, 0.0f);
    glVertex2f(0.3f, 0.0f);
```

```
   glVertex2f(0.2f, 0.10f);
   glEnd();

   glColor3ub(103,91,91);
   glBegin(GL_QUADS);
   glVertex2f(0.2f, -0.10f);
   glVertex2f(0.3f, -0.10f);
   glVertex2f(0.3f, 0.0f);
   glVertex2f(0.2f, 0.0f);
   glEnd();

   glColor3ub(151,116,116);
   glBegin(GL_QUADS);
   glVertex2f(0.3f, -0.10f);
   glVertex2f(0.34f, -0.10f);
   glVertex2f(0.34f, 0.0f);
   glVertex2f(0.3f, 0.0f);
   glEnd();
}

void drawScene() {
   glClear(GL_COLOR_BUFFER_BIT);
   glColor3d(1,0,0);
   glLoadIdentity(); //Reset the drawing perspective
   glMatrixMode(GL_MODELVIEW);
   glPushMatrix();
   glTranslatef(_move, 0.0f, 0.0f);
   Car();
   wheel();
   glPopMatrix();
   glutSwapBuffers();
   }


void update(int value) {
   _move += .02;
   if(_move > 1.3)
   {
   _move = -1.0;
   }
   glutPostRedisplay();
   glutTimerFunc(20, update, 0);
   }
```

```
void update1(int value) {
   _angle1+=2.0f;
   if(_angle1 > 360.0)
   {
   _angle1-=360;
   }
   glutPostRedisplay();

   glutTimerFunc(20, update1, 0);
   }



int main(int argc, char** argv) {
   glutInit(&argc, argv);
   glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB);
   glutInitWindowSize(800, 800);
   glutCreateWindow("Transformation");
   glutDisplayFunc(drawScene);
   glutTimerFunc(20, update, 0);
   glutTimerFunc(20, update1, 0);
   glutMainLoop();
   return 0;
   }
```

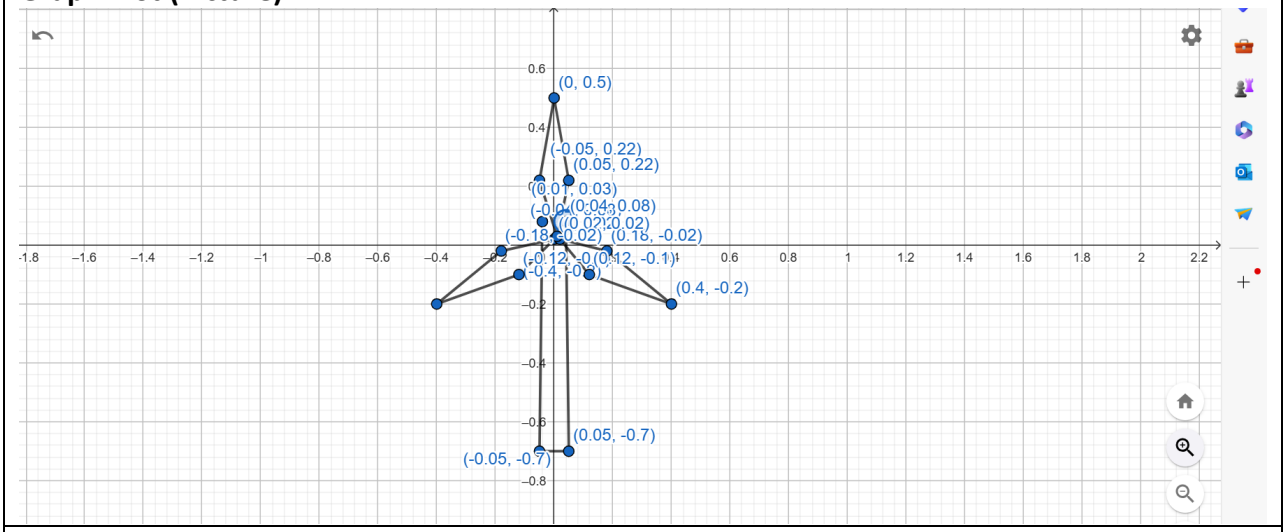
Output Screenshot (Full Screen)-



Question-4

Design a windmill with rotating blades

Graph Plot (Picture)-

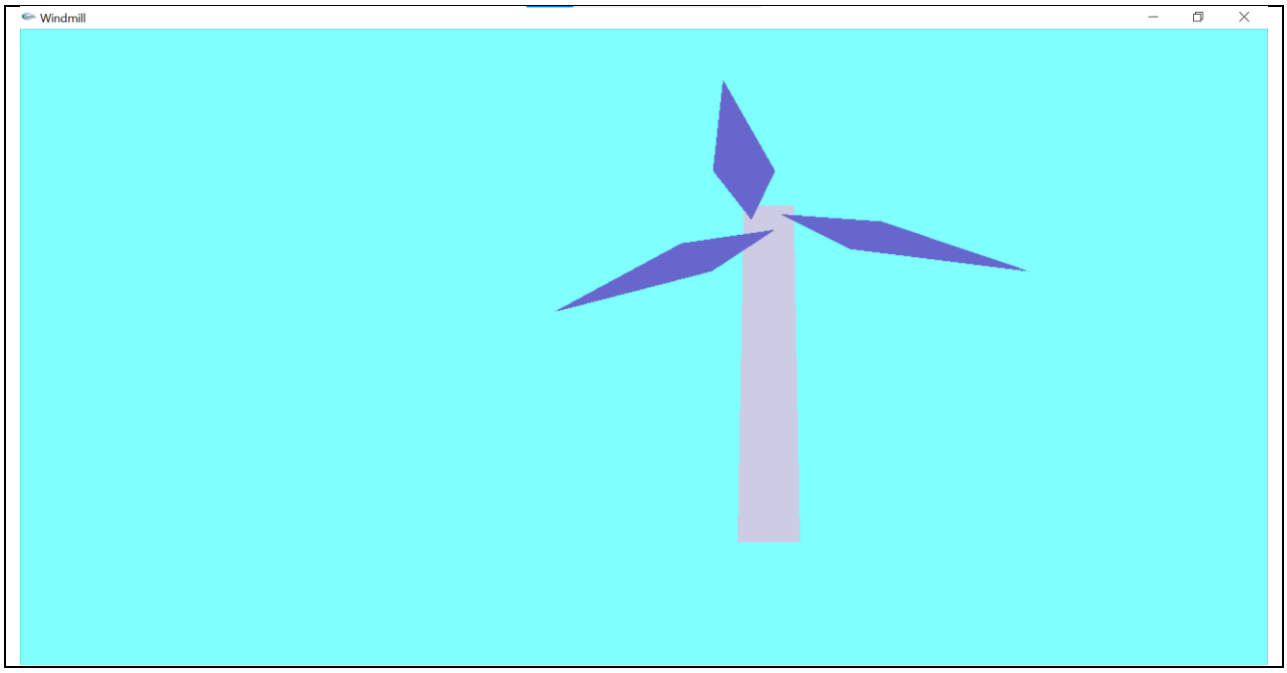


Code-

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

int frameNumber = 0;

void drawWindmill() {
   int i;
    glColor3f(0.8f, 0.8f, 0.9f);
   glBegin(GL_POLYGON);
   glVertex2f(-0.05f, -0.7);
   glVertex2f(0.05f, -0.7);
   glVertex2f(0.04f, 0.35);
   glVertex2f(-0.04f, 0.35);
   glEnd();
   glTranslatef(0, 0.3, 0);
   glRotated(frameNumber * (180.0 / 30), 0, 0, 1);
   glColor3f(0.4f, 0.4f, 0.8f);
   for (i = 0; i < 3; i++) {
      glRotated(120, 0, 0, 0.1);
      glBegin(GL_POLYGON);
      glVertex2f(0.02, 0.02);
      glVertex2f(0.12f, -0.1f);
      glVertex2f(0.4f, -0.2);
      glVertex2f(0.18f, -0.02f);
      glEnd();
```

```
    }
}

void display() {
    glClear(GL_COLOR_BUFFER_BIT);
    glClearColor(0.5f, 1, 1, 1);
    glLoadIdentity();
    glPushMatrix();
    glTranslated(0.2, 0.1, 0);
    drawWindmill();
    glPopMatrix();
    glutSwapBuffers();
}

void doFrame(int v) {
    frameNumber++;
    glutPostRedisplay();
    glutTimerFunc(30, doFrame, 0);
}

int main(int argc, char** argv) {
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_DOUBLE);
    glutInitWindowSize(700, 500);
    glutInitWindowPosition(100, 100);
    glutCreateWindow("Windmill");
    glutDisplayFunc(display);
    glutTimerFunc(200, doFrame, 0);
    glutMainLoop();
    return 0;
}
```

Output Screenshot (Full Screen)-


Windmill