A Practical Activity Report Submitted for
Computer Graphics (UCS505)

On

3D Car Project

Jadi Akhilesh Prasad - 101803188

Arshnoor Batra - 101803189

Aaryaman Singla - 101803683

Submitted to:

Mr. Ashwini Kumar

Computer Science And Engineering Department

Thapar Institute Of Engineering & Technology,

(A Deemed To Be University), Patiala, Punjab India

Jan-May 2021

TABLE OF CONTENTS

Introduction Of The Project

In this era of computing, computer graphics has become one of the most powerful and interesting facts of computing. Graphics today is used in many different areas. Graphics provides one of the most natural means of communicating within a computer. Interactive computer graphics is the most important means of producing pictures since the invention of photography and television.

3D Computer Graphics or three dimensional computer graphics are graphics that use three dimensional representation of geometric data that is stored on the computer for the purpose of viewing images in real time.

OpenGL is an application program interface (API) offering various functions to implement primitives, models and images. This offers functions to create and manipulate render lighting, coloring, viewing the models. OpenGL offers different coordinate systems and frames. It also offers translation, rotation and scaling of objects. Functions in the main GL library have names that begin with gl and are stored in a library usually referred to as GL.

Our project of **3D Car Animation** has the following objectives :

- It deals with the movement of the car in the X, Y and Z axis.
- It provides car custom size selection.
- It provides camera view settings.
- It also provides special key symbols for forward and backward movement of the car.

System Design

- **Program Structure :** The program consists of many functions such as display function, reshape function, text display function, call back function and the main function.

- **Start Screen :** As you execute the program, the first thing that comes is the start screen where you will see the details of the project. The start screen has the name of the college and the name of the project. It also has the user interaction instructions about the use of keyboard functionality. There is an option of going to the main screen, by pressing the 'space' key.

- **Main Screen:** After you get into the main screen from the start screen you will see the 3D Car drawn in blue colour.

- **User Interaction:** The user interaction is one of the most important parts of any program. This 3D OpenGL program has user interaction using a keyboard.

Computer Graphics Concepts Used

- Fundamentals of Computer Graphics: During the process of making this project, we learnt about the importance of computer graphics in the field of computing. We also learnt about the input output devices as well as the raster scan displays.

- Graphics Primitives : We also came across the algorithms for drawing various output primitives like line, circle, ellipse, arcs & sectors.

- 2-D & 3-D Geometrical Transformations: Our project deals with the concepts of Translation, Rotation, Scaling, Shearing, Reflection, Homogeneous coordinate system and Composite transformations.

User Defined Functions created in the project

GLvoid Transform(GLfloat Width, GLfloat Height) - Carries simple transformation routine. It takes two parameters - Width and Height.
Following operations are carried out:

- Set the viewport
- Select the projection matrix
- Reset The Projection Matrix
- Calculate The Aspect Ratio Of The Window
- Switch back to the modelview matrix

GLvoid InitGL(GLfloat Width, GLfloat Height) - It is a general OpenGL initialization function. It sets all of the initial parameters. It takes two parameters - Width and Height.
Following operations are carried out:

- Add line width
- Perform the transformation
- Create light components
- Assign created components to GL_LIGHT0

GLvoid ReSizeGLScene(GLint Width, GLint Height) - The function called when our window is resized. It takes two parameters - Width and Height.

void display_string(int x, int y, char const* string, int font) - This is the main drawing function. Here, we put all the OpenGL and calls to routines which manipulate the OpenGL state and environment. This is the function which will be called when a "redisplay" is requested.

void display1(void) - It displays the text on the current menu.

GLvoid DrawGLScene() - It is the function that takes no argument. Here, we perform basic concepts of computer graphics to give a shape to the 3D car.
Following operations are carried out to give shape to the car:

- Front Body
- Middle Body
- Enter Window
- Ignition System
- Wheel

void NormalKey(GLubyte key, GLint x, GLint y) - The function called whenever a "normal" key is pressed. This function uses switch-case.
Following are the cases that the user can access:

- X, x - Rotate the car in 'x' direction
- Y, y - Rotate the car in 'y' direction
- Z, z - Rotate the car in 'z' direction
- A, a - Increase the size of car in ‘x' direction
- S, s - Increase the size of car in 'y' direction
- Q, q- Increase the size of car in 'z' direction
- U, u- Camera top view
- F, f- Camera side view
- esc - Exit from the program
- spacebar - Enter the main screen from the start screen.

static void SpecialKeyFunc(int Key, int x, int y) - The function called whenever a "special" key is pressed. This function uses switch-case.
Following are the cases that the user can access:

- left arrow key - Move car in forward direction
- right arrow key- Move car in backward direction

Code Pasted As Text

```
#include <stdio.h>
#include <stdlib.h>
#include <glut.h>
#include <math.h>
#include <string.h>

/* ASCII code for the escape key. */
#define ESCAPE 27

GLint window;
GLint window2;
GLint Xsize = 1000;
GLint Ysize = 800;
float i, theta;
GLint nml = 0, day = 1;

char name3[] = "PROJECT:  3D CAR  ANIMATION";

GLfloat xt = 0.0, yt = 0.0, zt = 0.0, xw = 0.0;   /* x,y,z translation */
GLfloat tx = 295, ty = 62;
GLfloat xs = 1.0, ys = 1.0, zs = 1.0;

GLfloat xangle = 0.0, yangle = 0.0, zangle = 0.0, angle = 0.0;   /* axis angles */

GLfloat r = 0, g = 0, b = 1;
GLint light = 1;
int count = 1, flg = 1;
int view = 0;
int flag1 = 0, aflag = 1;            //to switch car driving mode
int wheelflag = 0;   //to switch fog effect
GLUquadricObj* t;

static void SpecialKeyFunc(int Key, int x, int y);

/* Simple  transformation routine */
GLvoid Transform(GLfloat Width, GLfloat Height)
{
        glViewport(0, 0, Width, Height);             /* Set the viewport */
```

```
	glMatrixMode(GL_PROJECTION);		/* Select the projection matrix */
	glLoadIdentity();			/* Reset The Projection Matrix */
	gluPerspective(45.0, Width / Height, 0.1, 100.0);  /* Calculate The Aspect Ratio Of The
Window */
	glMatrixMode(GL_MODELVIEW);		/* Switch back to the modelview matrix
*/
}


/* A general OpenGL initialization function.  Sets  all of the initial parameters. */
GLvoid InitGL(GLfloat Width, GLfloat Height)
{

	glClearColor(1.0, 1.0, 1.0, 1.0);
	glLineWidth(2.0);		/* Add line width,   ditto */
	Transform(Width, Height); /* Perform the transformation */
	//newly added
	t = gluNewQuadric();
	gluQuadricDrawStyle(t, GLU_FILL);

	glEnable(GL_LIGHTING);

	glEnable(GL_LIGHT0);

	// Create light components
	GLfloat ambientLight[] = { 0.2f, 0.2f, 0.2f, 1.0f };
	GLfloat diffuseLight[] = { 0.8f, 0.8f, 0.8, 1.0f };
	GLfloat specularLight[] = { 0.5f, 0.5f, 0.5f, 1.0f };
	GLfloat position[] = { 1.5f, 1.0f, 4.0f, 1.0f };

	// Assign created components to GL_LIGHT0
	glLightfv(GL_LIGHT0, GL_AMBIENT, ambientLight);
	glLightfv(GL_LIGHT0, GL_DIFFUSE, diffuseLight);
	glLightfv(GL_LIGHT0, GL_SPECULAR, specularLight);
	glLightfv(GL_LIGHT0, GL_POSITION, position);

}

/* The function called when our window is resized  */
GLvoid ReSizeGLScene(GLint Width, GLint Height)
```

```
{
        if (Height == 0)     Height = 1;                /* Sanity checks */
        if (Width == 0)      Width = 1;
        Transform(Width, Height);                /* Perform the transformation */
}

void init()
{
        glClearColor(0, 0, 0, 0);
        glPointSize(5.0);
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        glOrtho(0.0, 900.0, 0.0, 600.0, 50.0, -50.0);
        glutPostRedisplay();              // request redisplay
}


/* The main drawing function

  In here we put all the OpenGL and calls to routines which manipulate
  the OpenGL state and environment.

  This is the function which will be called when a "redisplay" is requested.
*/

void display_string(int x, int y, char const* string, int font)
{
        int len, i;
        glColor3f(0.8, 0.52, 1.0);
        glRasterPos2f(x, y);
        len = (int)strlen(string);
        for (i = 0; i < len; i++) {
                if (font == 1)
                        glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24, string[i]);
                if (font == 2)
                        glutBitmapCharacter(GLUT_BITMAP_HELVETICA_18, string[i]);
                if (font == 3)
                        glutBitmapCharacter(GLUT_BITMAP_HELVETICA_12, string[i]);
                if (font == 4)
                        glutBitmapCharacter(GLUT_BITMAP_HELVETICA_10, string[i]);
```

```
        }

}

void display1(void)
{

        glClearColor(1.0, 1.0, 0.1, 1.0);
        display_string(180, 540, "THAPAR INSTITUTE OF ENGINEERING AND
TECHNOLOGY", 1); //correct cordinate according to name
        display_string(215, 500, name3, 1);
        display_string(390, 470, "HELP", 2);
        display_string(10, 450, "MOUSE", 2);
        display_string(10, 410, "PRESS RIGHT BUTTON FOR MENU", 3);
        display_string(10, 370, "KEYBOARD", 2);
        display_string(10, 340, "X-Y-Z KEYS FOR CORRESPONDING ROTATION", 3);
        display_string(10, 310, "A-S-Q CAR CUSTOM SIZE SELECTION", 3);
        display_string(10, 280, "U-F FOR CAMERA VIEW SETTINGS", 3);
        display_string(10, 250, "USE LEFT ARROW(<-) AND RIGHT  ARROW(->) TO MOVE
CAR", 3);
        display_string(10, 220, "ESCAPE TO EXIT", 3);
        display_string(250, 150, "PRESS SPACE BAR TO ENTER", 2);
        glutPostRedisplay();
        glutSwapBuffers();

}

GLvoid DrawGLScene()
{


        glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);        /* Clear The
Screen And The Depth Buffer */
        if (view == 0)
        {
                init();
                display1();
        }
        else
        {
```

```
if (count == 1)
	InitGL(Xsize, Ysize);
if (aflag == 1)/* Initialize our window. */
	glClearColor(1, 1, 1, 1);
else
	glClearColor(0.1, 0.1, 0.1, 0);
glPushMatrix();
glLoadIdentity();
glTranslatef(-1.0, 0.0, -3.5);
glRotatef(xangle, 1.0, 0.0, 0.0);
glRotatef(yangle, 0.0, 1.0, 0.0);
glRotatef(zangle, 0.0, 0.0, 1.0);
glTranslatef(xt, yt, zt);
glScalef(xs, ys, zs);
glEnable(GL_COLOR_MATERIAL);
glColorMaterial(GL_FRONT, GL_AMBIENT_AND_DIFFUSE);

if (!aflag) {
	glBegin(GL_POINTS);
	glColor3f(1, 1, 1);
	glPointSize(200.0);
	int ccount = 0;
	float x = 10, y = 10;
	while (ccount < 20)
	{
		glVertex2f(x, y);
		x += 10;
		y += 10;
		if (y > Ysize) y -= 10;
		if (x > Xsize) x -= 10;
		ccount++;
	}
	glEnd();
}

glColor3f(1.0, .75, 0.0);
glPointSize(30.0);
glBegin(GL_POINTS);
glVertex3f(0.2, 0.3, 0.3);
glVertex3f(0.2, 0.3, 0.5);
```

```
		glEnd();
		glPointSize(200.0);



		glBegin(GL_QUADS);		/* OBJECT MODULE*/

		/* top of cube*/
		//************************FRONT
BODY**************************************
		glColor3f(r, g, b);
		glVertex3f(0.2, 0.4, 0.6);
		glVertex3f(0.6, 0.5, 0.6);
		glVertex3f(0.6, 0.5, 0.2);
		glVertex3f(0.2, 0.4, 0.2);

		/* bottom of cube*/
		glVertex3f(0.2, 0.2, 0.6);
		glVertex3f(0.6, 0.2, 0.6);
		glVertex3f(0.6, 0.2, 0.2);
		glVertex3f(0.2, 0.2, 0.2);

		/* front of cube*/
		glVertex3f(0.2, 0.2, 0.6);
		glVertex3f(0.2, 0.4, 0.6);
		glVertex3f(0.2, 0.4, 0.2);
		glVertex3f(0.2, 0.2, 0.2);

		/* back of cube.*/
		glVertex3f(0.6, 0.2, 0.6);
		glVertex3f(0.6, 0.5, 0.6);
		glVertex3f(0.6, 0.5, 0.2);
		glVertex3f(0.6, 0.2, 0.2);

		/* left of cube*/
		glVertex3f(0.2, 0.2, 0.6);
		glVertex3f(0.6, 0.2, 0.6);
		glVertex3f(0.6, 0.5, 0.6);
		glVertex3f(0.2, 0.4, 0.6);
```

```
                /* Right of cube */
                glVertex3f(0.2, 0.2, 0.2);
                glVertex3f(0.6, 0.2, 0.2);
                glVertex3f(0.6, 0.5, 0.2);
                glVertex3f(0.2, 0.4, 0.2);

//*************************************************************************
                glVertex3f(0.7, 0.65, 0.6);
                glVertex3f(0.7, 0.65, 0.2);
                glVertex3f(1.7, 0.65, 0.2);       //top cover
                glVertex3f(1.7, 0.65, 0.6);
                //**************************back
guard*****************************
                glColor3f(r, g, b);          /* Set The Color To Blue*/
                glVertex3f(1.8, 0.5, 0.6);
                glVertex3f(1.8, 0.5, 0.2);
                glVertex3f(2.1, 0.4, 0.2);
                glVertex3f(2.1, 0.4, 0.6);

                /* bottom of cube*/
                glVertex3f(2.1, 0.2, 0.6);
                glVertex3f(2.1, 0.2, 0.2);
                glVertex3f(1.8, 0.2, 0.6);
                glVertex3f(1.8, 0.2, 0.6);

                /* back of cube.*/
                glVertex3f(2.1, 0.4, 0.6);
                glVertex3f(2.1, 0.4, 0.2);
                glVertex3f(2.1, 0.2, 0.2);
                glVertex3f(2.1, 0.2, 0.6);

                /* left of cube*/
                glVertex3f(1.8, 0.2, 0.2);
                glVertex3f(1.8, 0.5, 0.2);
                glVertex3f(2.1, 0.4, 0.2);
                glVertex3f(2.1, 0.2, 0.2);

                /* Right of cube */
                glVertex3f(1.8, 0.2, 0.6);
                glVertex3f(1.8, 0.5, 0.6);
```

```
                glVertex3f(2.1, 0.4, 0.6);
                glVertex3f(2.1, 0.2, 0.6);
                //******************MIDDLE
BODY**********************************
                glVertex3f(0.6, 0.5, 0.6);
                glVertex3f(0.6, 0.2, 0.6);
                glVertex3f(1.8, 0.2, 0.6);
                glVertex3f(1.8, 0.5, 0.6);

                /* bottom of cube*/
                glVertex3f(0.6, 0.2, 0.6);
                glVertex3f(0.6, 0.2, 0.2);
                glVertex3f(1.8, 0.2, 0.2);
                glVertex3f(1.8, 0.2, 0.6);

                /* back of cube.*/
                glVertex3f(0.6, 0.5, 0.2);
                glVertex3f(0.6, 0.2, 0.2);
                glVertex3f(1.8, 0.2, 0.2);
                glVertex3f(1.8, 0.5, 0.2);
                //*********************ENTER
WINDOW**********************************
                glColor3f(0.3, 0.3, 0.3);
                glVertex3f(0.77, 0.63, 0.2);
                glVertex3f(0.75, 0.5, 0.2);       //quad front window
                glVertex3f(1.2, 0.5, 0.2);
                glVertex3f(1.22, 0.63, 0.2);

                glVertex3f(1.27, 0.63, .2);
                glVertex3f(1.25, 0.5, 0.2);       //quad back window
                glVertex3f(1.65, 0.5, 0.2);
                glVertex3f(1.67, 0.63, 0.2);

                glColor3f(r, g, b);
                glVertex3f(0.7, 0.65, 0.2);
                glVertex3f(0.7, 0.5, .2);      //first separation
                glVertex3f(0.75, 0.5, 0.2);
                glVertex3f(0.77, 0.65, 0.2);

                glVertex3f(1.2, 0.65, 0.2);
```

```
glVertex3f(1.2, 0.5, .2);       //second separation
glVertex3f(1.25, 0.5, 0.2);
glVertex3f(1.27, 0.65, 0.2);

glVertex3f(1.65, 0.65, 0.2);
glVertex3f(1.65, 0.5, .2);     //3d separation
glVertex3f(1.7, 0.5, 0.2);
glVertex3f(1.7, 0.65, 0.2);

glVertex3f(0.75, 0.65, 0.2);
glVertex3f(0.75, 0.63, 0.2);       //line strip
glVertex3f(1.7, 0.63, 0.2);
glVertex3f(1.7, 0.65, 0.2);

glVertex3f(0.75, 0.65, 0.6);
glVertex3f(0.75, 0.63, 0.6);       //line strip
glVertex3f(1.7, 0.63, 0.6);
glVertex3f(1.7, 0.65, 0.6);

glColor3f(0.3, 0.3, 0.3);
glVertex3f(0.77, 0.63, 0.6);
glVertex3f(0.75, 0.5, 0.6);       //quad front window
glVertex3f(1.2, 0.5, 0.6);
glVertex3f(1.22, 0.63, 0.6);

glVertex3f(1.27, 0.63, .6);
glVertex3f(1.25, 0.5, 0.6);       //quad back window
glVertex3f(1.65, 0.5, 0.6);
glVertex3f(1.67, 0.63, 0.6);

glColor3f(r, g, b);
glVertex3f(0.7, 0.65, 0.6);
glVertex3f(0.7, 0.5, .6);       //first separation
glVertex3f(0.75, 0.5, 0.6);
glVertex3f(0.77, 0.65, 0.6);

glVertex3f(1.2, 0.65, 0.6);
glVertex3f(1.2, 0.5, .6);       //second separation
glVertex3f(1.25, 0.5, 0.6);
glVertex3f(1.27, 0.65, 0.6);
```

```
glVertex3f(1.65, 0.65, 0.6);
glVertex3f(1.65, 0.5, .6);
glVertex3f(1.7, 0.5, 0.6);
glVertex3f(1.7, 0.65, 0.6);
glEnd();


//************************************************************
glBegin(GL_QUADS);

/* top of cube*/
glColor3f(0.3, 0.3, 0.3);
glVertex3f(0.6, 0.5, 0.6);
glVertex3f(0.6, 0.5, 0.2);       //quad front window
glVertex3f(0.7, 0.65, 0.2);
glVertex3f(0.7, 0.65, 0.6);

glVertex3f(1.7, 0.65, .6);
glVertex3f(1.7, 0.65, 0.2);       //quad back window
glVertex3f(1.8, 0.5, 0.2);
glVertex3f(1.8, 0.5, 0.6);


//*****************************road and surrounding
development**********************************
if (flag1)
{
        glPushMatrix();
        glTranslatef(xw, 0, 0);
        glColor3f(0, 1, 0);
        glVertex3f(-100, 0.1, -100);
        glVertex3f(-100, 0.1, 0);        //a green surroundings
        glVertex3f(100, 0.1, 0);
        glVertex3f(100, 0.1, -100);

        glColor3f(0.7, 0.7, 0.7);
        glVertex3f(-100, 0.1, 0);
        glVertex3f(-100, 0.1, 0.45);        //a long road
        glVertex3f(100, 0.1, 0.45);
```

```
        glVertex3f(100, 0.1, 0);

        glColor3f(1.0, 0.75, 0.0);
        glVertex3f(-100, 0.1, 0.45);      //a median
        glVertex3f(-100, 0.1, 0.55);
        glVertex3f(100, 0.1, 0.55);
        glVertex3f(100, 0.1, 0.45);

        glColor3f(0.7, 0.7, 0.7);
        glVertex3f(-100, 0.1, 0.55);
        glVertex3f(-100, 0.1, 1);         //a long road
        glVertex3f(100, 0.1, 1);
        glVertex3f(100, 0.1, 0.55);

        glColor3f(0, 1, 0);
        glVertex3f(-100, 0.1, 1);
        glVertex3f(-100, 0.1, 100);       //a green surroundings
        glVertex3f(100, 0.1, 100);
        glVertex3f(100, 0.1, 1);
        glPopMatrix();
}
glEnd();

if (wheelflag)
{
        glPushMatrix();
        glTranslatef(xw, 0, 0);
        glColor3f(0.5, .2, 0.3);
        glBegin(GL_QUADS);
        for (i = 0; i < 200; i += 0.2)
        {
                glVertex3f(-100 + i, 0, 1);
                glVertex3f(-99.9 + i, 0, 1);
                glVertex3f(-99.9 + i, 0.2, 1);
                glVertex3f(-100 + i, 0.2, 1);
                i += 0.5;
        }
        for (i = 0; i < 200; i += 0.2)
        {
                glVertex3f(-100 + i, 0, 0);
```

```
                glVertex3f(-99.9 + i, 0, 0);
                glVertex3f(-99.9 + i, 0.2, 0);
                glVertex3f(-100 + i, 0.2, 0);
                i += 0.5;
            }
            glEnd();
            glPopMatrix();
        }

//*************************************************************************
**********************
            glBegin(GL_TRIANGLES);              /* start drawing the cube.*/

             /* top of cube*/
            glColor3f(0.3, 0.3, 0.3);
            glVertex3f(0.6, 0.5, 0.6);
            glVertex3f(0.7, 0.65, 0.6);      //tri front window
            glVertex3f(0.7, 0.5, 0.6);

            glVertex3f(0.6, 0.5, 0.2);
            glVertex3f(0.7, 0.65, 0.2);      //tri front window
            glVertex3f(0.7, 0.5, 0.2);

            glVertex3f(1.7, 0.65, 0.2);
            glVertex3f(1.8, 0.5, 0.2);      //tri back window
            glVertex3f(1.7, 0.5, 0.2);

            glVertex3f(1.7, 0.65, 0.6);
            glVertex3f(1.8, 0.5, 0.6);      //tri back window
            glVertex3f(1.7, 0.5, 0.6);

            glEnd();
            //************IGNITION SYSTEM**********************************
            glPushMatrix();
            glColor3f(0.7, 0.7, 0.7);
            glTranslatef(1.65, 0.2, 0.3);
            glRotatef(90.0, 0, 1, 0);
            gluCylinder(t, 0.02, 0.03, .5, 10, 10);
            glPopMatrix();
```

```
//********************WHEEL*********************************************

                glColor3f(0.7, 0.7, 0.7);
                glPushMatrix();
                glBegin(GL_LINE_STRIP);
                for (theta = 0; theta < 360; theta = theta + 20)
                {
                        glVertex3f(0.6, 0.2, 0.62);
                        glVertex3f(0.6 + (0.08 * (cos(((theta + angle) * 3.14) / 180))), 0.2 + (0.08
* (sin(((theta + angle) * 3.14) / 180))), 0.62);
                }
                glEnd();

                glBegin(GL_LINE_STRIP);
                for (theta = 0; theta < 360; theta = theta + 20)
                {
                        glVertex3f(0.6, 0.2, 0.18);
                        glVertex3f(0.6 + (0.08 * (cos(((theta + angle) * 3.14) / 180))), 0.2 + (0.08
* (sin(((theta + angle) * 3.14) / 180))), 0.18);
                }
                glEnd();

                glBegin(GL_LINE_STRIP);
                for (theta = 0; theta < 360; theta = theta + 20)
                {
                        glVertex3f(1.7, 0.2, 0.18);
                        glVertex3f(1.7 + (0.08 * (cos(((theta + angle) * 3.14) / 180))), 0.2 + (0.08
* (sin(((theta + angle) * 3.14) / 180))), 0.18);
                }
                glEnd();

                glBegin(GL_LINE_STRIP);
                for (theta = 0; theta < 360; theta = theta + 20)
                {
                        glVertex3f(1.7, 0.2, 0.62);
                        glVertex3f(1.7 + (0.08 * (cos(((theta + angle) * 3.14) / 180))), 0.2 + (0.08
* (sin(((theta + angle) * 3.14) / 180))), 0.62);
                }
                glEnd();
```

```
                glTranslatef(0.6, 0.2, 0.6);
                glColor3f(0, 0, 0);
                glutSolidTorus(0.025, 0.07, 10, 25);

                glTranslatef(0, 0, -0.4);
                glutSolidTorus(0.025, 0.07, 10, 25);

                glTranslatef(1.1, 0, 0);
                glutSolidTorus(0.025, 0.07, 10, 25);

                glTranslatef(0, 0, 0.4);
                glutSolidTorus(0.025, 0.07, 10, 25);
                glPopMatrix();
                //*************************************************************
                glPopMatrix();
                glEnable(GL_DEPTH_TEST);
                glutPostRedisplay();
                glutSwapBuffers();
        }
}


/*  The function called whenever a "normal" key is pressed. */
void NormalKey(GLubyte key, GLint x, GLint y)
{
        switch (key) {
        case ESCAPE: printf("escape pressed. exit.\n");
                glutDestroyWindow(window);          /* Kill our window */
                exit(0);
                break;

        case ' ':view = 1;
                DrawGLScene();
                break;

        case 'x': xangle += 5.0;
                glutPostRedisplay();
                break;

        case 'X':xangle -= 5.0;
```

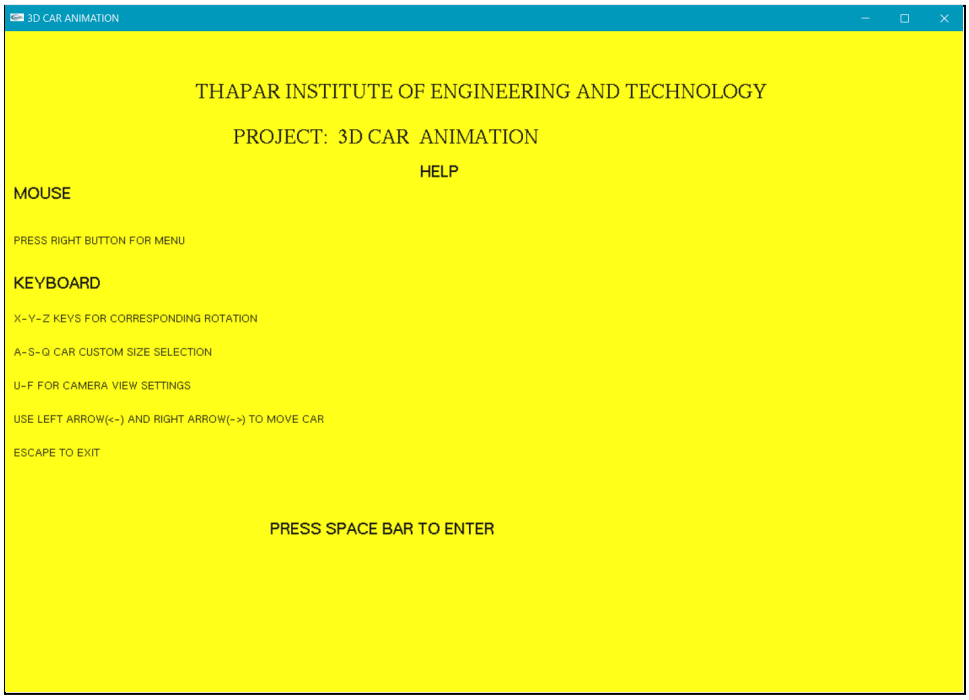
```
            glutPostRedisplay();
            break;

    case 'y':
            yangle += 5.0;
            glutPostRedisplay();
            break;

    case 'Y':
            yangle -= 5.0;
            glutPostRedisplay();
            break;

    case 'z':
            zangle += 5.0;
            glutPostRedisplay();
            break;

    case 'Z':
            zangle -= 5.0;
            glutPostRedisplay();
            break;

    case 'u':                   /* Move up */
            yt += 0.2;
            glutPostRedisplay();
            break;

    case 'U':
            yt -= 0.2;                  /* Move down */
            glutPostRedisplay();
            break;

    case 'f':                   /* Move forward */
            zt += 0.2;
            glutPostRedisplay();
            break;

    case 'F':
            zt -= 0.2;                  /* Move away */
```

```
                glutPostRedisplay();
                break;

        case 's':zs += .2;
                glutPostRedisplay();
                break;

        case 'S':zs -= 0.2;
                glutPostRedisplay();
                break;

        case 'a':ys += .2;
                glutPostRedisplay();
                break;

        case 'A':ys -= 0.2;
                glutPostRedisplay();
                break;

        case 'q':xs += .2;
                glutPostRedisplay();
                break;

        case 'Q':xs -= 0.2;
                glutPostRedisplay();
                break;


        default:
                break;
        }

}

static void SpecialKeyFunc(int Key, int x, int y)
{
        switch (Key) {
        case GLUT_KEY_RIGHT:
                if (!wheelflag)
                        xt += 0.2;
```

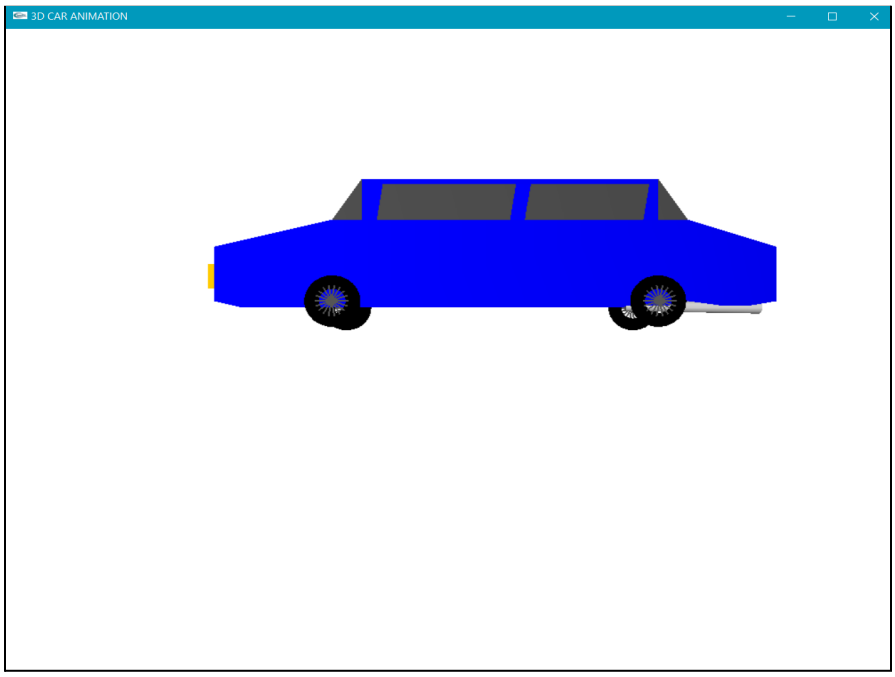
```
                if (wheelflag)
                {
                        angle += 5;
                        xw += 0.2;
                }
                glutPostRedisplay();
                break;

        case GLUT_KEY_LEFT:
                if (!wheelflag)
                        xt -= 0.2;
                if (wheelflag)
                {
                        angle += 5;
                        xw -= 0.2;
                }
                glutPostRedisplay();
                break;
        }
}


void myreshape(int w, int h)
{
        glViewport(0, 0, w, h);
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        if (w <= h)
                glOrtho(-2.0, 2.0, -2.0 * (GLfloat)h / (GLfloat)w, 2.0 * (GLfloat)h / (GLfloat)w,
-10.0, 10.0);
        else
                glOrtho(-2.0 * (GLfloat)w / (GLfloat)h, 2.0 * (GLfloat)w / (GLfloat)h, -2.0, 2.0,
-10.0, 10.0);
        glMatrixMode(GL_MODELVIEW);
        glutPostRedisplay();
}


//*************************** Main
************************************************************
```

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

        /* Initialisation and window creation */

        glutInit(&argc, argv);              /* Initialize GLUT state. */

        glutInitDisplayMode(GLUT_RGBA |      /* RGB and Alpha */
                GLUT_DOUBLE |     /* double buffer */
                GLUT_DEPTH);     /* Z buffer (depth) */

        glutInitWindowSize(Xsize, Ysize);     /* set initial window size. */
        glutInitWindowPosition(0, 0);         /* upper left corner of the screen. */

        glutCreateWindow("3D CAR ANIMATION"); /* Open a window with a title. */

 /* Now register the various callback functions */

        glutReshapeFunc(myreshape);
        glutDisplayFunc(DrawGLScene);        /* Function to do all our OpenGL drawing. */
        glutReshapeFunc(ReSizeGLScene);
        glutKeyboardFunc(NormalKey);         /*Normal key is pressed */
        glutSpecialFunc(SpecialKeyFunc);
        InitGL(Xsize, Ysize);


        /* Now drop into the event loop from which we never return */

        glutMainLoop();                   /* Start Event Processing Engine. */
        return 1;
}
```

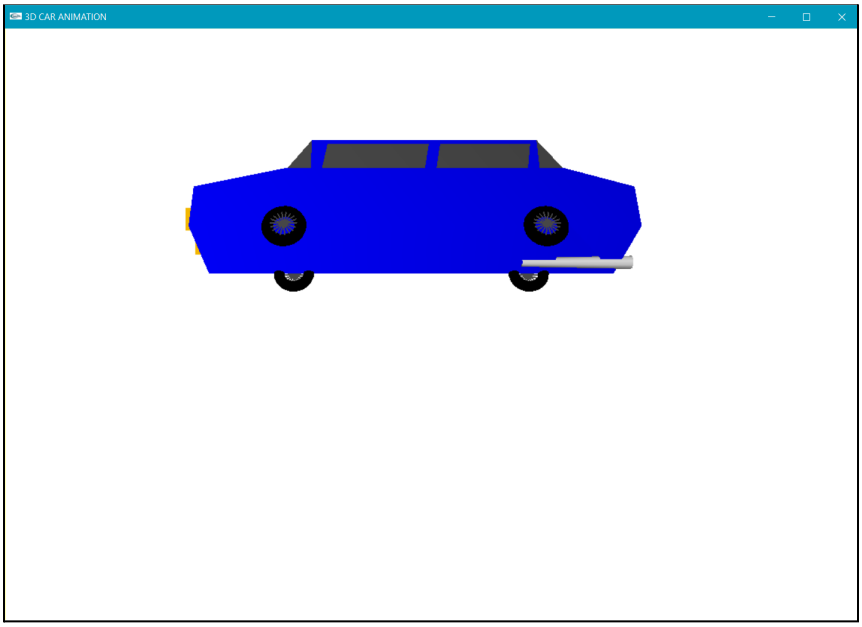
Snapshots



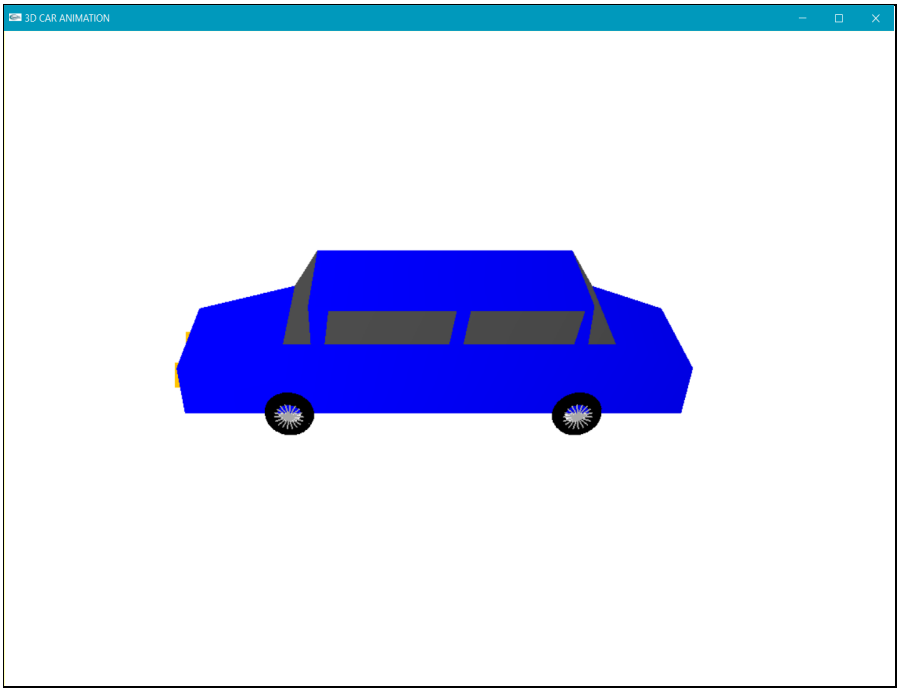
Starting window of the project



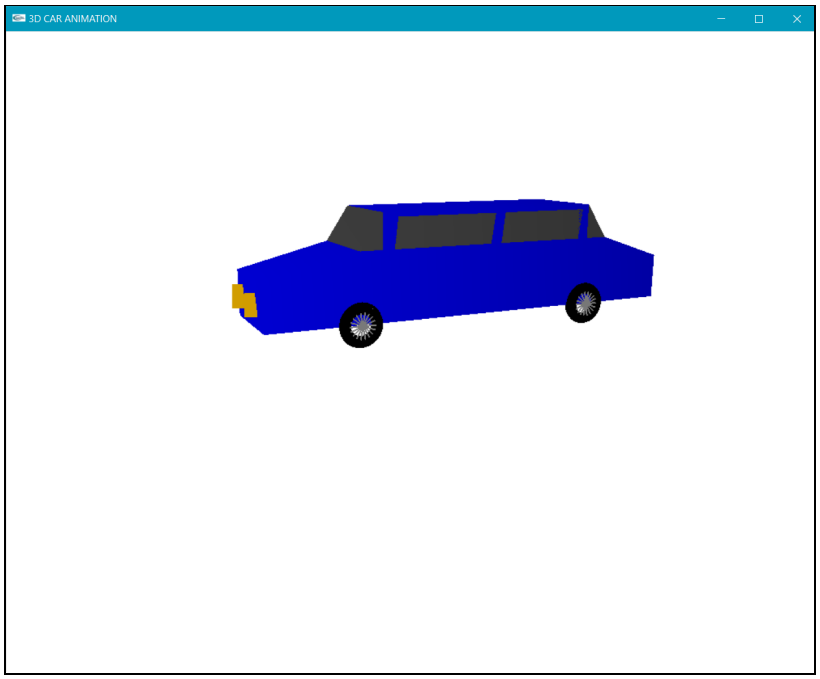
Initial image after clicking spacebar in starting window



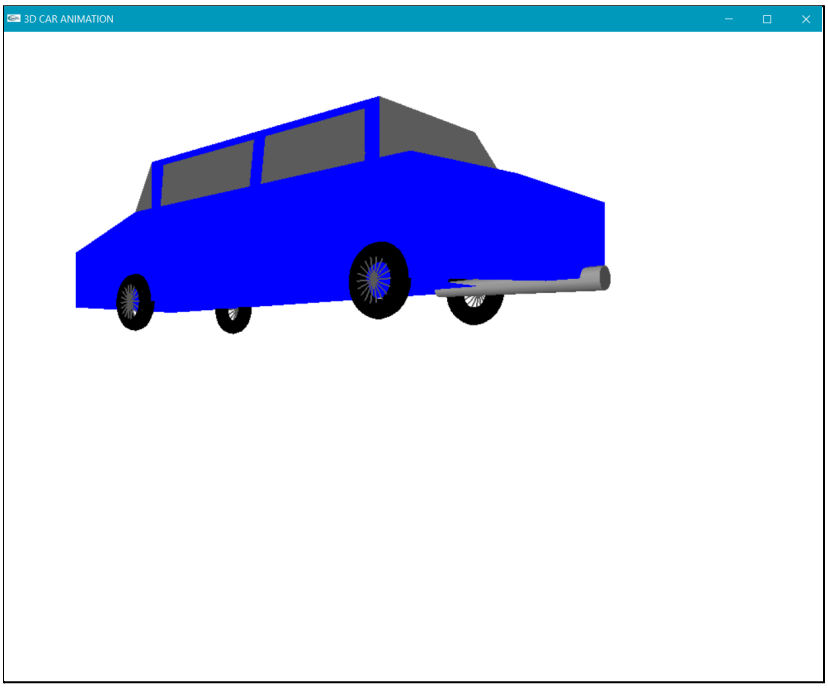
After clicking 5 times "X"



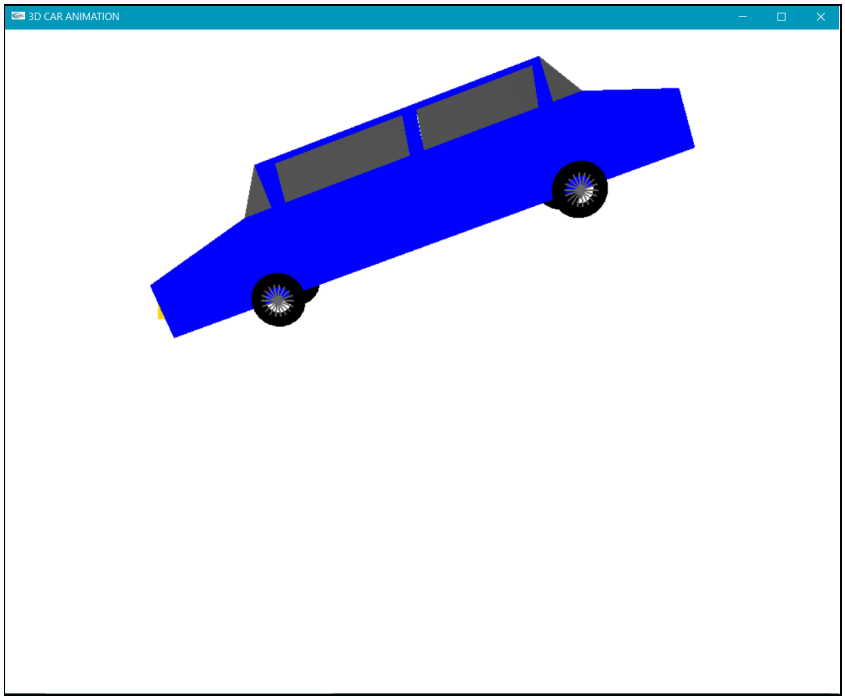
After clicking 5 times "x"



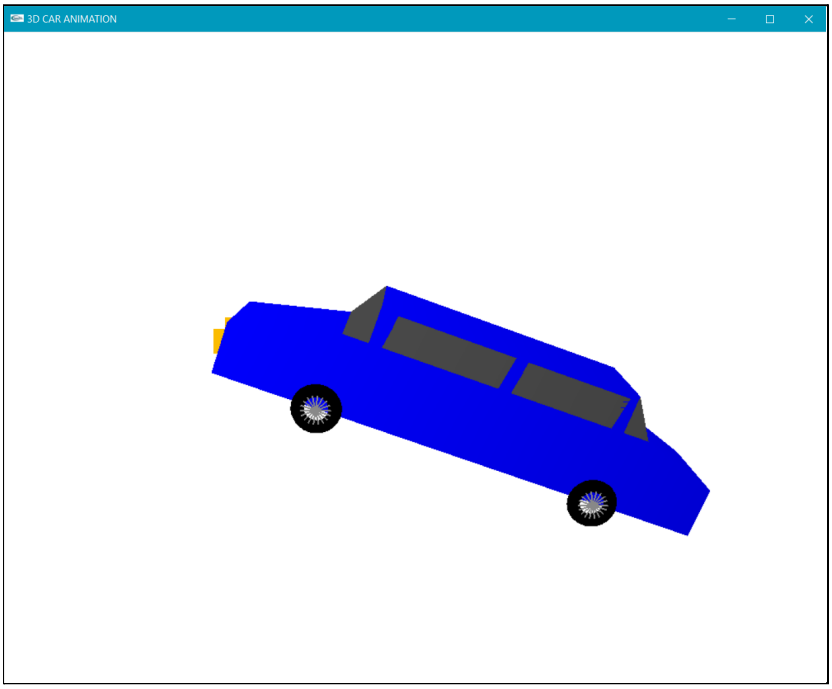
After clicking 5 times "y"



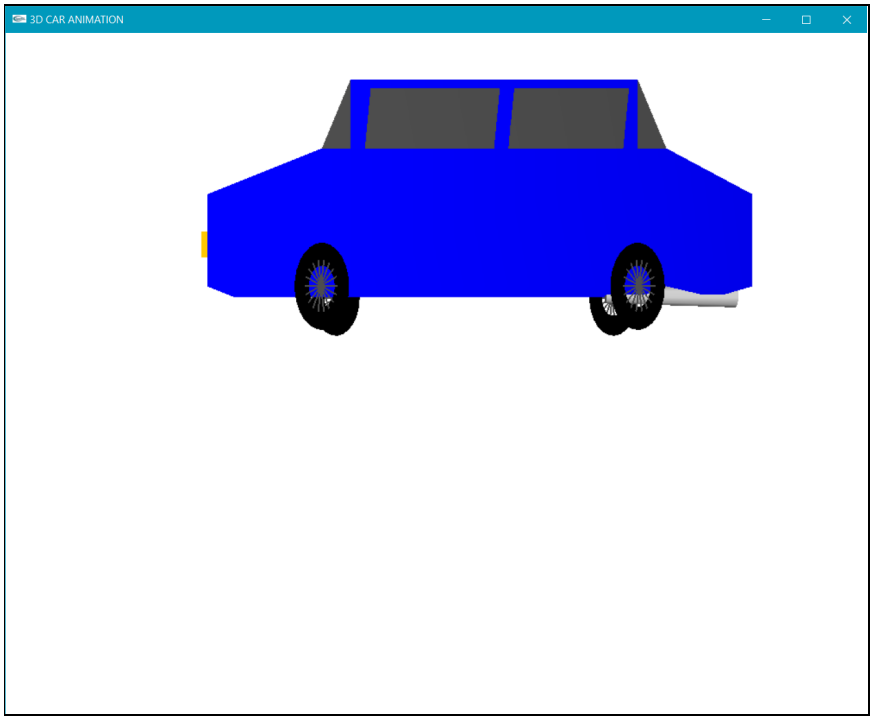
After clicking 5 times "Y"



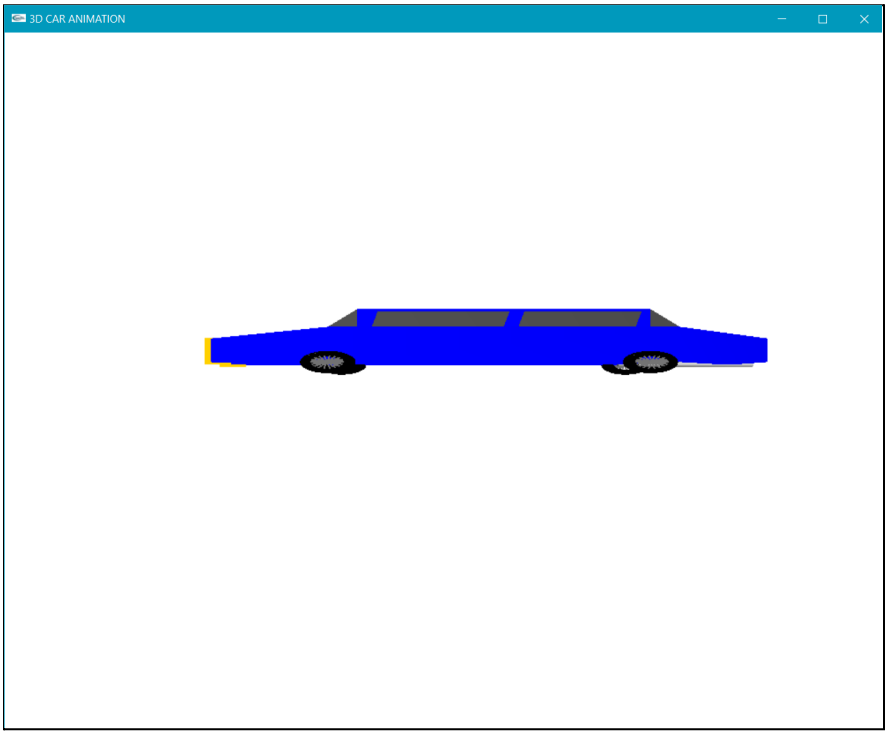
After clicking 5 times "z"



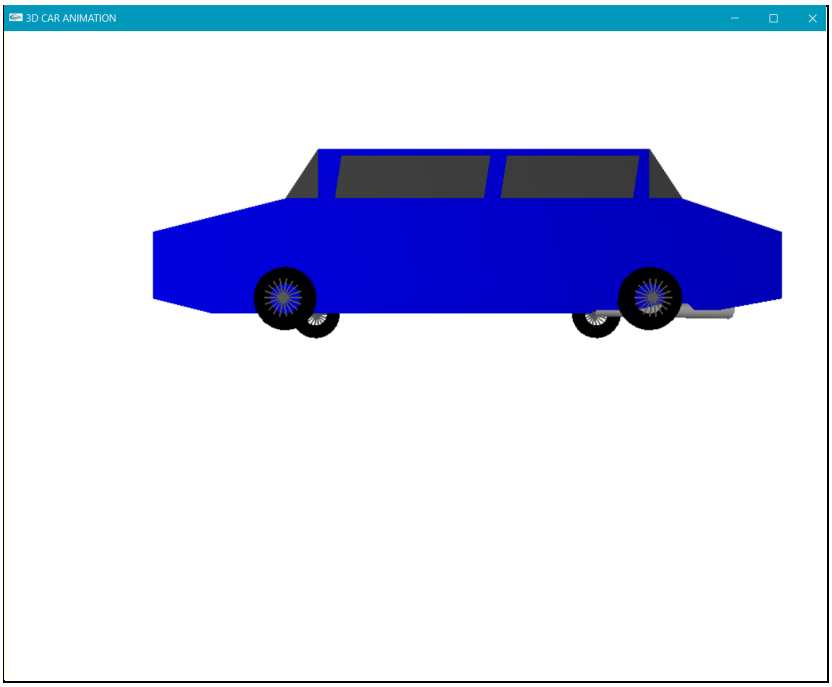
After clicking 5 times"Z"



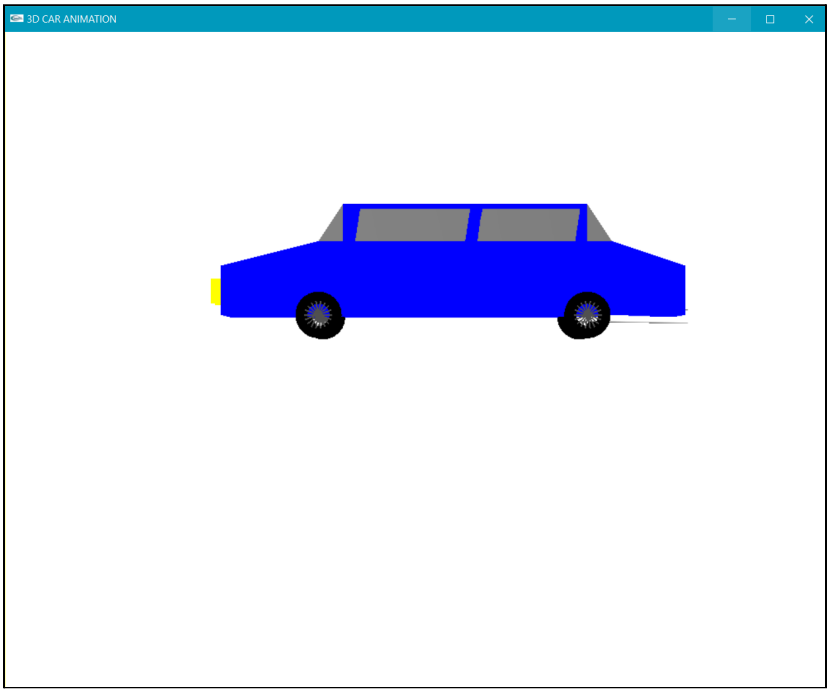
After clicking 3 times "a"



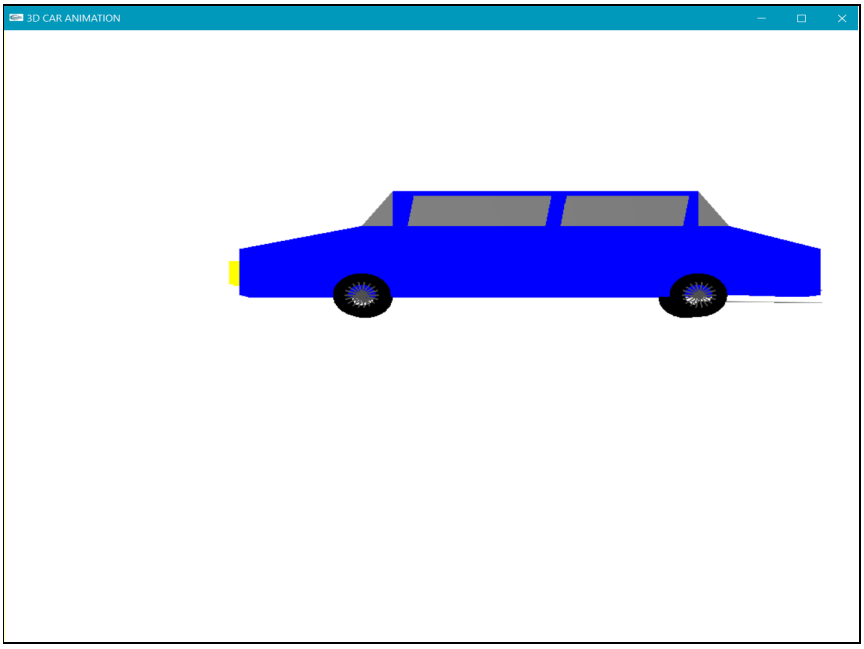
After clicking 3 times "A"



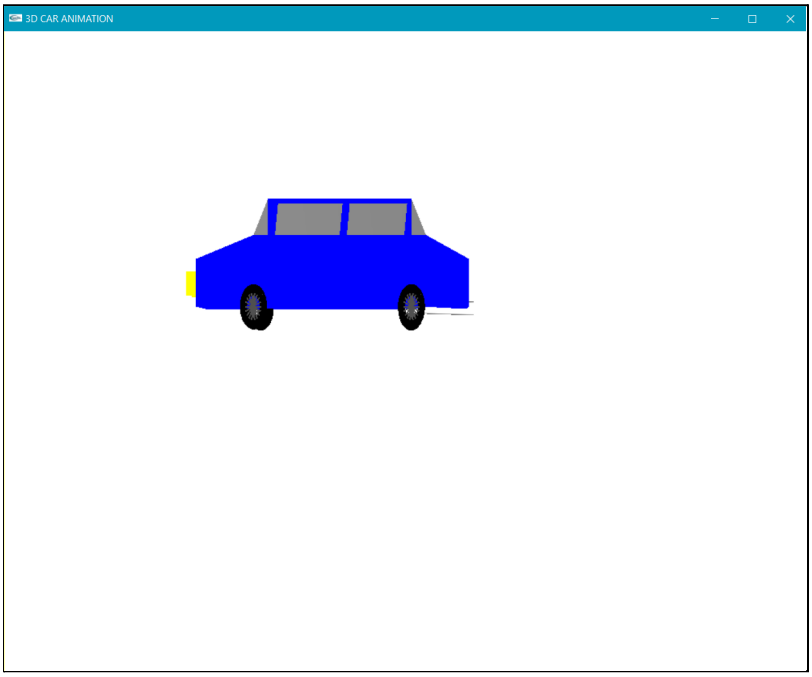
After clicking 3 times "s"



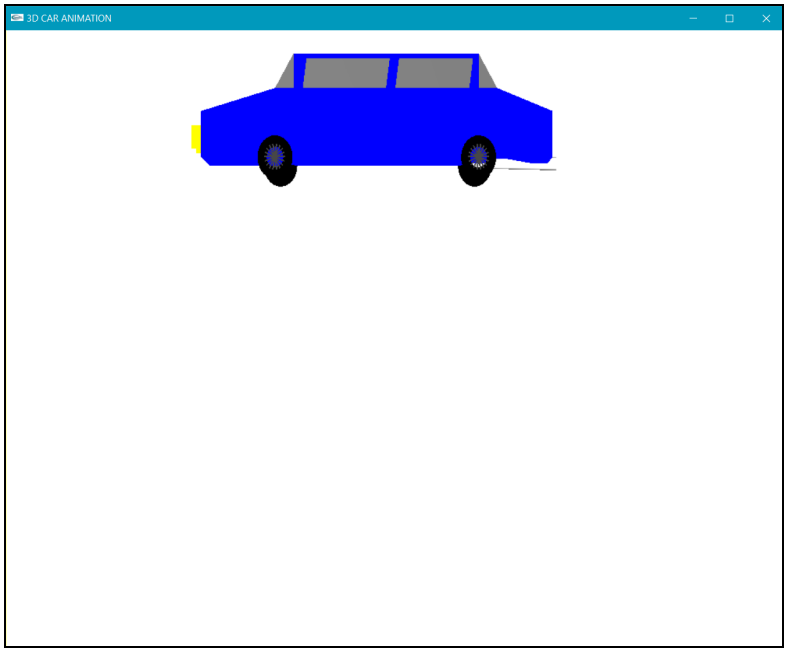
After clicking 3 times "S"



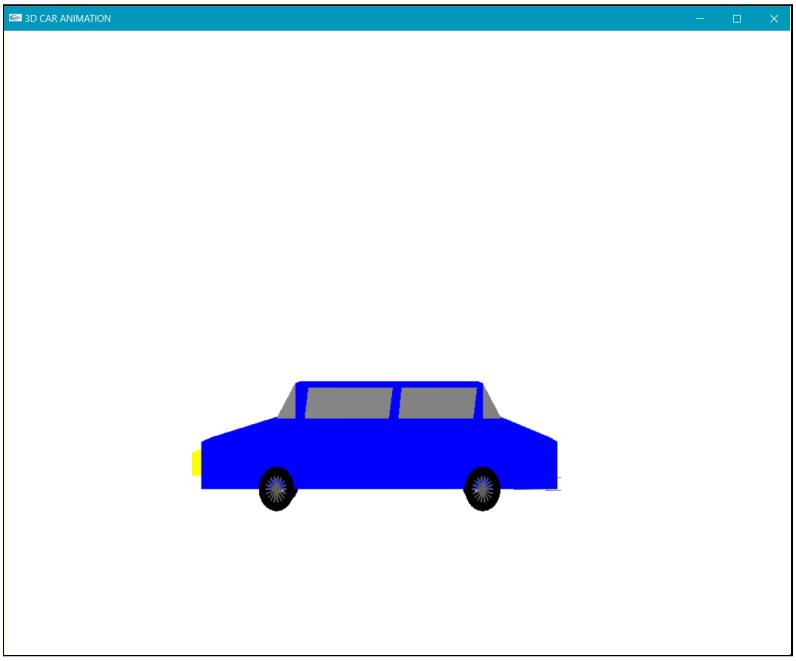
After clicking 2 times “q”



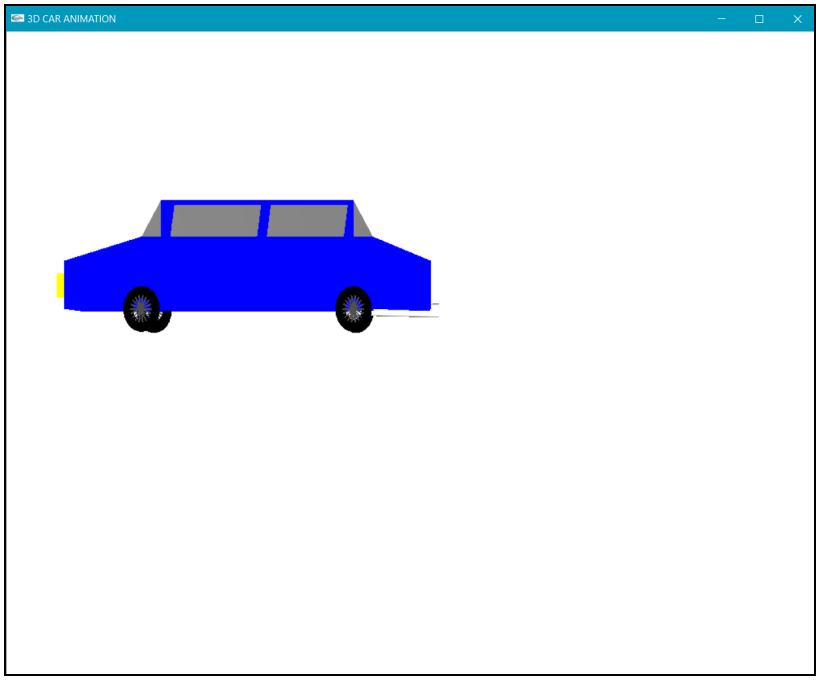
After clicking 2 times“Q”



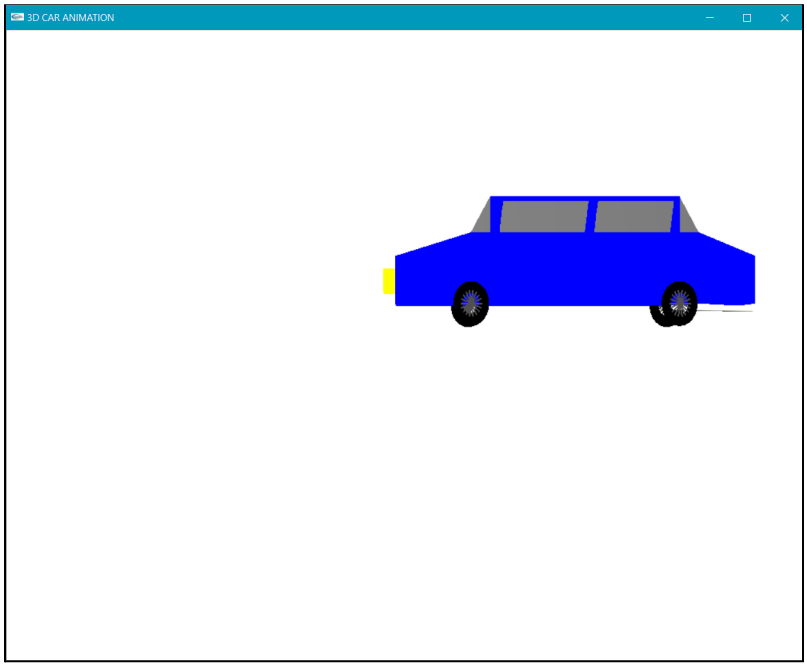
After clicking 2 times "u"



After clicking 2 times "U"



After clicking 2 times "<-"



After clicking 2 times "->"