Assignment 13

Class - SE IV
Batch - F4

Roll NO - 21430
DOS.

Title - Implementation of openGL function

Problem statement -:
Write C++ program to draw 3-D cube and perform following transformation on it using openGL i) Scaling ii) Translation iii) Rotation about an axis

Learning objective -:
To implement openGL functions.

Theory -:
* OpenGL Basic -:
open graphics Library (OpenGL) is a cross language, cross platform API for rendering 2D and 3D vector graphics. OpenGL is a low-level widely supported modeling and redering software package, available across all platforms. It can be used in arrange of graphics applications such as games, CAD design or modeling. OpenGL API is designed mostly in hardware.

* Pre-requisities for OpenGL -:
Since openGL is a graphics API and not

platform its own it, requires a language to operate in and the language of choice is C++

* Overview of an OpenGL program
i. Main
ii. Open window and Configure frame Buffer.
iii. Initialize GL states and display.
iv. Loop
v. check for events.
vi. Redraw
vii. Clear the Screen
viii. Change states
ix. Render
x. Swap Buffers

* OpenGL Syntax -:
All functions have the form, gl*
glvertex3f() - 3 means that this function take three arguments and f means that the type of those argument float

In OpenGL program it is better to use OpenGL variable type.

Algorithm/ Pseudocode -:

```
procedure init(void)

    glclearcolor (0.0, 0.0, 0.0)
    glEnable( GL_Depth_TEST)
end procedure.

procedure reshape ( int w, int h)

    glviewport (0, 0, (GLsize) w, (GLsize)h) ;
    glmatrixMode ( GL PROJECTION)
    glLoad Identity ()
    glutperspective (60, 1, 2.0, 90.0)
    glmatrixMode ( GL_MODEVIEW)
end procedure

procedure Display (void)

    glcolour (GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)

    glLoadIdentity ()
    glTranslatef ( 0.0, 0.0, -7)
    glscalef ( 1.2, 1.2, 1.2)
    glRotate ( angle, 1, 0, 0)
    glRotate ( angle, 0, 1, 0)
    glRotate ( angle, 0, 0, 1)

    glBegin (GL_QUADS)
```

```
glcolor3f(0.0f,1.0f,0.0f)
glvertex(1.0f,1.0f,-1.0f)
glvertex(-1.0f,1.0f,-1.0f)
glvertex3f(-1.0f,1.0f,1.0f)
glvertex3f(1.0f,1.0f,1.0f)

//similarly form all remaining 5 faces. we have
//to write in code

glEND()
glutSwapBuffers()

end procedure

procedure main(int argc, char ** argv)

glutInit(&argc,argv)
glutInitDisplayMode(GLUT_DOUBLE|GLUT_RGB|
                    GLUT_DEPTH)
glutInitWindowposition(0,0)
glutInitwindowsize(600,600)
glutcreatewindow("3D CUBE")
init()
glutDisplayFunc(display)
glutReshapeFun(reshape)
glutTimefunc(1000/50,timer,0)
glutMainloop()

end procedure
```

* Conclusion :-

In this assignment I have implemented open GL Concepts and I was able to perform some transformation operation on 3D cube

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

typedef float Matrix4[4][4];

Matrix4 theMatrix;
static GLfloat input[8][3] =
{
    {40,40,-50},{90,40,-50},{90,90,-50},{40,90,-50},
    {30,30,0},{80,30,0},{80,80,0},{30,80,0}

};

float output[8][3];
float tx, ty, tz;
float sx, sy, sz;
float angle;

int choice, choiceRot;

void setIdentityM(Matrix4 m)
{
    for (int i = 0; i < 4; i++)
        for (int j = 0; j < 4; j++)
            m[i][j] = (i == j);
}

void translate(int tx, int ty, int tz)
{

    for (int i = 0; i < 8; i++)
    {
        output[i][0] = input[i][0] + tx;
        output[i][1] = input[i][1] + ty;
```

Report ...

```
    }
}
void scale(int sx, int sy, int sz)
{
    theMatrix[0][0] = sx;
    theMatrix[1][1] = sy;
    theMatrix[2][2] = sz;
}
void RotateX(float angle) //Parallel to x
{

    angle = angle * 3.142 / 180;
    theMatrix[1][1] = cos(angle);
    theMatrix[1][2] = -sin(angle);
    theMatrix[2][1] = sin(angle);
    theMatrix[2][2] = cos(angle);

}
void RotateY(float angle) //parallel to y
{

    angle = angle * 3.14 / 180;
    theMatrix[0][0] = cos(angle);
    theMatrix[0][2] = -sin(angle);
    theMatrix[2][0] = sin(angle);
    theMatrix[2][2] = cos(angle);

}
void RotateZ(float angle) //parallel to z
{

    angle = angle * 3.14 / 180;
    theMatrix[0][0] = cos(angle);
    theMatrix[0][1] = sin(angle);
    theMatrix[1][0] = -sin(angle);
    theMatrix[1][1] = cos(angle);
```

Report ...

```
84
85  void multiplyM()
86  {
87      //We Don't require 4th row and column in scaling and rotation
88      //[8][3]=[8][3]*[3][3] //4th not used
89      for (int i = 0; i < 8; i++)
90      {
91          for (int j = 0; j < 3; j++)
92          {
93              output[i][j] = 0;
94              for (int k = 0; k < 3; k++)
95              {
96                  output[i][j] = output[i][j] + input[i][k] * theMatrix[k][j];
97              }
98          }
99      }
100
101 }
102 void Axes(void)
103 {
104     glColor3f(1.0, 1.0, 1.0);                // Set the color to BLACK
105     glBegin(GL_LINES);                        // Plotting X-Axis
106     glVertex2s(-1000, 0);
107     glVertex2s(1000, 0);
108     glEnd();
109     glBegin(GL_LINES);                        // Plotting Y-Axis
110     glVertex2s(0, -1000);
111     glVertex2s(0, 1000);
112     glEnd();
113 }
114 void draw(float a[8][3])
115 {
116     glBegin(GL_QUADS);
117     glColor3f(0.7, 0.4, 0.5); //behind
118     glVertex3fv(a[0]);
119     glVertex3fv(a[1]);
```

Report ...

```
118     glVertex3fv(a[0]);
119     glVertex3fv(a[1]);
120     glVertex3fv(a[2]);
121     glVertex3fv(a[3]);
122
123     glColor3f(0.8, 0.2, 0.4);  //bottom
124     glVertex3fv(a[0]);
125     glVertex3fv(a[1]);
126     glVertex3fv(a[5]);
127     glVertex3fv(a[4]);
128
129     glColor3f(0.3, 0.6, 0.7); //left
130     glVertex3fv(a[0]);
131     glVertex3fv(a[4]);
132     glVertex3fv(a[7]);
133     glVertex3fv(a[3]);
134
135     glColor3f(0.2, 0.8, 0.2);  //right
136     glVertex3fv(a[1]);
137     glVertex3fv(a[2]);
138     glVertex3fv(a[6]);
139     glVertex3fv(a[5]);
140
141     glColor3f(0.7, 0.7, 0.2); //up
142     glVertex3fv(a[2]);
143     glVertex3fv(a[3]);
144     glVertex3fv(a[7]);
145     glVertex3fv(a[6]);
146
147     glColor3f(1.0, 0.1, 0.1);
148     glVertex3fv(a[4]);
149     glVertex3fv(a[5]);
150     glVertex3fv(a[6]);
151     glVertex3fv(a[7]);
152
153     glEnd();
```

Report ...

```
157 {
158     glClearColor(0.0, 0.0, 0.0, 1.0); //set background color to white
159     glOrtho(-454.0, 454.0, -250.0, 250.0, -250.0, 250.0);
160     // Set the no. of co-ordinates along X & Y axes and their gappings
161     glEnable(GL_DEPTH_TEST);
162     // To Render the surfaces Properly according to their depths
163 }
164
165 void display()
166 {
167     glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
168     Axes();
169     glColor3f(1.0, 0.0, 0.0);
170     draw(input);
171     setIdentityM(theMatrix);
172     switch (choice)
173     {
174     case 1:
175         translate(tx, ty, tz);
176         break;
177     case 2:
178         scale(sx, sy, sz);
179         multiplyM();
180         break;
181     case 3:
182         switch (choiceRot) {
183         case 1:
184             RotateX(angle);
185             break;
186         case 2: RotateY(angle);
187             break;
188         case 3:
189             RotateZ(angle);
190             break;
191         default:
192             break;
```

Report ...

```
        break;
    }

    draw(output);
    glFlush();
}

int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB | GLUT_DEPTH);
    glutInitWindowSize(1362, 750);
    glutInitWindowPosition(0, 0);
    glutCreateWindow("3D_Transformations...");
    init();


    cout << "Choose Transformation: \n\t 1. Translation \n\t 2. Scaling \n\t 3. Rotation \n\t 4. Exit \n\nChoice =  ";
    cin >> choice;
    cout << endl;

    if (choice == 1)
    {
        cout << "\n\t Enter tx  ty  tz =  ";
        cin >> tx >> ty >> tz;
    }

    else if (choice == 2)
    {
        cout << "\n\t Enter Sx  Sy  Sz =  ";
        cin >> sx >> sy >> sz;
    }

    else if (choice == 3)
    {
        cout << endl;
```

Report ...

```
        {
            cout << "\n\t Enter Rotation angle =  ";
            cin >> angle;
        }

        else if (choiceRot == 2)
        {
            cout << "\n\t Enter Rotation angle =  ";
            cin >> angle;
        }

        else if (choiceRot == 3)
        {
            cout << "\n\t Enter Rotation angle =  ";
            cin >> angle;
        }

        else
        {
            cout << "\t Wrong Choice...!!!" << endl;
        }
    }


    else if (choice == 4)
    {
        return 0;
    }


    glutDisplayFunc(display);
    glutMainLoop();

    return 0;
}
```

Report ...

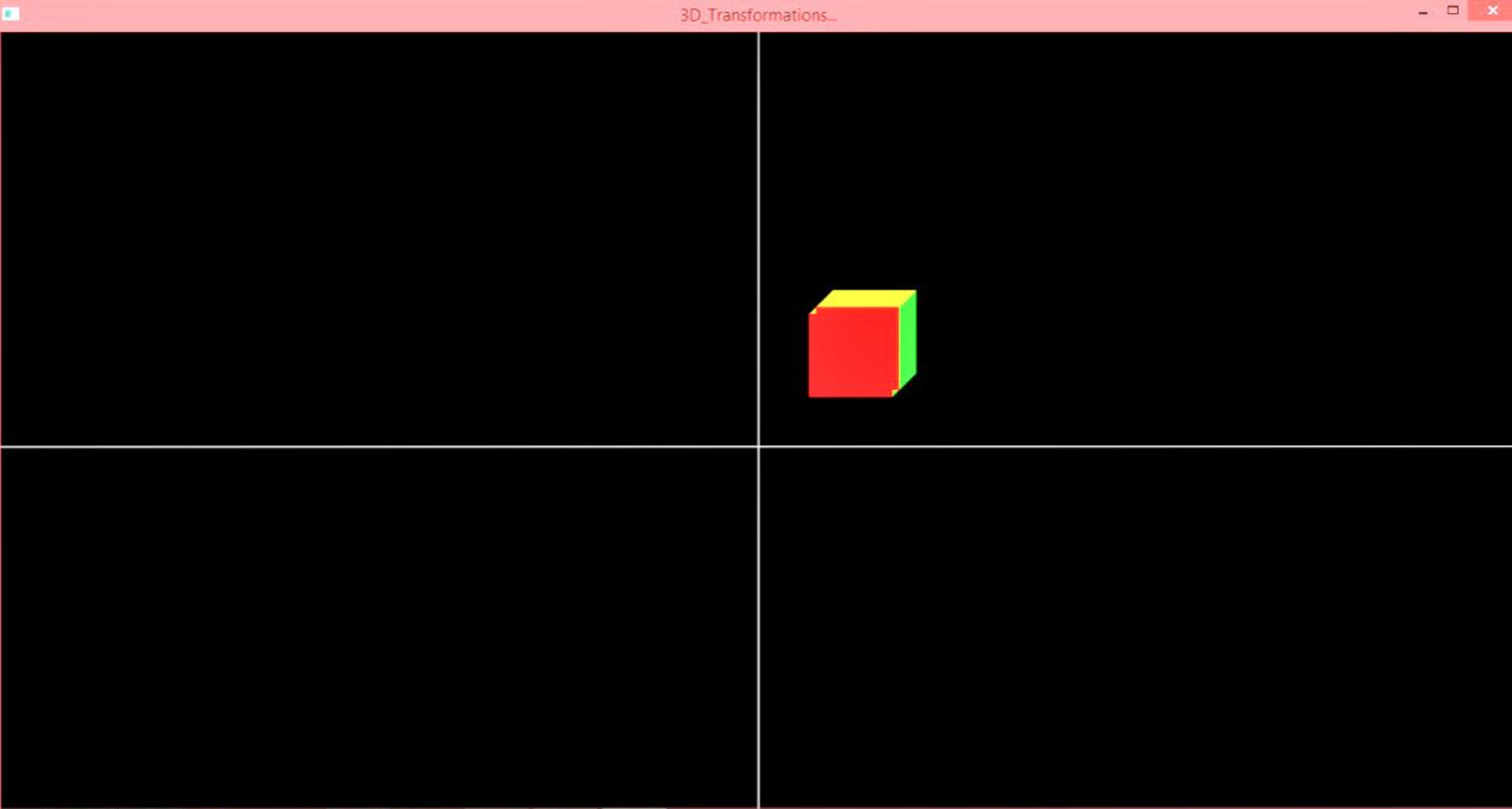
3D_Transformations...

C:\Users\Gauri\Documents\oop13.exe

```
Choose Transformation:
        1. Translation
        2. Scaling
        3. Rotation
        4. Exit

Choice =  1


        Enter tx  ty  tz =  4 4 4
```


3D_Transformations...

3D_Transformations...

C:\Users\Gauri\Documents\oop13.exe

```
Choose Transformation:
        1. Translation
        2. Scaling
        3. Rotation
        4. Exit

Choice =  3


Choose Axis For Rotation:
        1. Parallel to X-axis.(y& z)
        2. Parallel to Y-axis.(x& z)
        3. Parallel to Z-axis.(x& y)
 Choice =  1

        Enter Rotation angle =  60
```

3D_Transformations...

C:\Users\Gaun\Documents\oop13.exe

```
Choose Transformation:
        1. Translation
        2. Scaling
        3. Rotation
        4. Exit

Choice =  2


        Enter Sx  Sy  Sz =  3 3 3
```