UCS1712 - GRAPHICS AND MULTIMEDIA LAB

EX - 1 : Study of Basic output primitives in OpenGL

NAME: KEERTHANA T
REGISTER NUMBER: 185001074
CLASS-SEC: CSE-B
DATE: 23/7/2021

AIM :

To study the basic output primitives in OpenGL by performing the following:

1. Create a window using OPENGL and to draw the following basic output primitives – POINTS, LINES, LINE_STRIP, LINE_LOOP, TRIANGLES, TRIANGLE STRIP, TRIANGLE FAN, QUADS, QUAD_STRIP, POLYGON.

2. Create a window and draw a simple House using OpenGL (shapes and colours can be as per your preferences)

ALGORITHM :

1. Algorithm for output primitives:
    - Start
    - Import GL library as a header file
    - Create a function void display()
    - Begin GL_POINTS

- Add three points of vertices(310.0, 410.0),(330.0, 410.0),(320.0, 420.0)
- End GL_POINTS
- Begin GL_POLYGON
- Add vertices (120.0, 100.0),(180.0, 100.0),(180.0, 10.0),(120.0, 10.0) to the polygon
- End GL_POLYGON
- Begin GL_LINES
- Add lines of vertices (5, 100),(75, 100),(35, 150),(35, 50)
- End GL_LINES
- Begin GL_LINE_STRIP
- Add lines of vertices (60, 360),(80, 350) ,(100, 360) ,(120, 350) ,(140, 360) ,(160, 350) ,(180, 360) ,(200, 350) ,(220, 360) ,(240, 350) ,(260, 360) ,(280, 350) ,(300, 360)
- End GL_LINE_STRIP
- Begin GL_LINE_LOOP
- Add vertices (110, 420) ,(145, 380) , (120, 465) ,(175, 445)
- End GL_LINE_LOOP
- Begin GL_TRIANGLES
- Add vertices (220.0, 10.0) ,(260.0, 100.0) ,(280.0, 10.0)
- End GL_TRIANGLES
- Begin GL_TRIANGLE_STRIP
- Add vertices (220.0, 150.0) ,(260.0, 200.0) ,(280.0, 150.0) ,(280.0, 150.0) ,(300.0, 200.0) ,(320.0, 150.0) ,(320.0, 150.0) ,(360.0, 200.0) ,(380.0, 150.0)
- End GL_TRAINGLE_STRIP
- Begin GL_TRIANGLE_FAN
- Add vertices (20, 240) ,(70, 290) ,(120, 240) ,(70, 190)
- End GL_TRIANGLE_FAN
- Begin GL_QUADS
- Add vertices (280, 220) ,(220, 220) ,(240, 280) ,(260, 280)
- End GL_QUADS
- Begin GL_QUAD_STRIP
- Add vertices (320, 220) , (380, 220) ,(360, 280) ,(340, 280)
- End GL_QUAD_STRIP

- Add red color to init
- Give basic details for the output window in the main function
- Call it in main function
- End

2. Algorithm for drawing a house:
    - Start
    - Import GL library as a header file
    - Create void display() function
    - Add vertices for stars using GL_POINTS
    - Add vertices for a triangle for the front roof using GL_TRIANGLES
    - Add vertices for chimney using GL_QUADS
    - Add vertices for side roof using GL_QUADS
    - Add vertices for a polygon representing a front wall using GL_POLYGON
    - Add vertices for a polygon representing a door using GL_POLYGON
    - Add vertices for a point representing a door knob using GL_POINTS
    - Add vertices for a polygon representing a side wall using GL_POLYGON
    - Add vertices for a polygon representing the ground using GL_POLYGON
    - Add vertices for lines representing the borders of the diagrams using GL_LINES and GL_LINE_LOOP and GL_LINE_STRIP.
    - Add background color in init function
    - Give basic details for the output window in the main function
    - Call display and init function in the main function
    - End

CODE:

1. Output primitives:

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

void display() {
   //POINTS
   glBegin(GL_POINTS);
   glVertex2f(310.0, 410.0);
   glVertex2f(330.0, 410.0);
   glVertex2f(320.0, 420.0);
   glEnd();
   //POLYGON
   glBegin(GL_POLYGON);
   glVertex2f(120.0, 100.0);
   glVertex2f(180.0, 100.0);
   glVertex2f(180.0, 10.0);
   glVertex2f(120.0, 10.0);
   glEnd();
   //LINES
   glBegin(GL_LINES);
        glVertex2f(5, 100);
        glVertex2f(75, 100);
        glVertex2f(35, 150);
        glVertex2f(35, 50);
   glEnd();
   //LINES-STRIP
   glBegin(GL_LINE_STRIP);
        glVertex2f(60, 360);
        glVertex2f(80, 350);
        glVertex2f(100, 360);
        glVertex2f(120, 350);
        glVertex2f(140, 360);
```

```
        glVertex2f(160, 350);
        glVertex2f(180, 360);
        glVertex2f(200, 350);
        glVertex2f(220, 360);
        glVertex2f(240, 350);
        glVertex2f(260, 360);
        glVertex2f(280, 350);
        glVertex2f(300, 360);
    glEnd();
    //LINE-LOOP
    glBegin(GL_LINE_LOOP);
        glVertex2f(110, 420);
        glVertex2f(145, 380 );
        glVertex2f(120, 465);
        glVertex2f(175, 445);
    glEnd();
    //TRIANGLE
    glBegin(GL_TRIANGLES);
        glVertex2f(220.0, 10.0);
    glVertex2f(260.0, 100.0);
    glVertex2f(280.0, 10.0);
    //glVertex2f(380.0, 10.0);
    //glVertex2f(400.0, 100.0);
    //glVertex2f(420.0, 10.0);
    glEnd();
    //TRIANGLE-STRIP
    glBegin(GL_TRIANGLE_STRIP);
    glVertex2f(220.0, 150.0);
        glVertex2f(260.0, 200.0);
    glVertex2f(280.0, 150.0);
    glVertex2f(280.0, 150.0);
    glVertex2f(300.0, 200.0);
    glVertex2f(320.0, 150.0);
    glVertex2f(320.0, 150.0);
    glVertex2f(360.0, 200.0);
```

```
    glVertex2f(380.0, 150.0);
    glEnd();
    //TRIANGLE-FAN
    glBegin(GL_TRIANGLE_FAN);
        glVertex2f(20, 240);
        glVertex2f(70, 290);
        glVertex2f(120, 240);
        glVertex2f(70, 190);
    glEnd();
    //QUADS
    glBegin(GL_QUADS);
        glVertex2f(280, 220);
        glVertex2f(220, 220);
        glVertex2f(240, 280);
        glVertex2f(260, 280);
        glEnd();
        //QUADS-STRIP
        glBegin(GL_QUAD_STRIP);
         glVertex2f(320, 220);
        glVertex2f(380, 220);
        glVertex2f(360, 280);
        glVertex2f(340, 280);

        glEnd();
    glFlush();
}

void myinit() {
    glClearColor(1.0, 1.0, 1.0, 1.0);
    glColor3f(1.0, 0.0, 0.0);
    glPointSize(5.0);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(0.0, 499.0, 0.0, 499.0);
}
```

```
int main(int argc, char** argv) {
   glutInit(&argc, argv);
   glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
   glutInitWindowSize(800, 800);
   glutInitWindowPosition(0, 0);
   glutCreateWindow("Output Primitives");
   glutDisplayFunc(display);

   myinit();
   glutMainLoop();
   return 0;
}
```

2. House:

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

void myInit (void)

{
        glClearColor(0.0,0.0,0.0,0.0);
         glPointSize(9.0);
       glMatrixMode(GL_PROJECTION);
       glLoadIdentity();
       gluOrtho2D(0.0, 500.0, 0.0, 400.0);
}


void myDisplay(void)

{

       glClear(GL_COLOR_BUFFER_BIT);
       //STARS
```

```
    glBegin(GL_POINTS);
glVertex2f(80.0, 250.0);
glVertex2f(40.0, 350.0);
glVertex2f(100.0, 320.0);
glVertex2f(180.0, 290.0);
glVertex2f(120.0, 350.0);
glVertex2f(155.0, 320.0);
glVertex2f(540.0, 350.0);
glVertex2f(245.0, 320.0);
glVertex2f(290.0, 345.0);
glVertex2f(220.0, 250.0);
glVertex2f(540.0, 350.0);
glVertex2f(400.0, 320.0);
glVertex2f(480.0, 290.0);
glVertex2f(520.0, 350.0);
glVertex2f(455.0, 320.0);
glEnd();
//FRONT ROOF
    glColor3f(0.627, 0.322, 0.176);
glBegin(GL_TRIANGLES);
glVertex2f(125.0, 170.0);
glVertex2f(175.0, 240.0);
glVertex2f(225.0, 170.0);
glEnd();
// CHIMNEY
glColor3f(0.545, 0.271, 0.075);
    glShadeModel(GL_SMOOTH);
glBegin(GL_QUADS);
glVertex2f(300.0, 240.0);
glVertex2f(300.0, 260.0);
glVertex2f(320.0, 260.0);
glVertex2f(320.0, 240.0);
glEnd();
//SIDE ROOF
glColor3f(0.545, 0.271, 0.075);
```

```
    glShadeModel(GL_SMOOTH);
glBegin(GL_QUADS);
glVertex2f(225.0, 170.0);
glVertex2f(175.0, 240.0);
glVertex2f(325.0, 240.0);
glVertex2f(375.0, 170.0);
glEnd();
    // FRONT WALL
    glColor3f(0.627, 0.322, 0.176);
glBegin(GL_POLYGON);
glVertex2f(125.0, 70.0);
glVertex2f(125.0, 170.0);
glVertex2f(225.0, 170.0);
glVertex2f(225.0, 70.0);
glEnd();
// DOOR
glColor3f(0.502, 0.000, 0.000);
glBegin(GL_POLYGON);
glVertex2f(165.0, 70.0);
glVertex2f(165.0, 110.0);
glVertex2f(185.0, 110.0);
glVertex2f(185.0, 70.0);
glEnd();
// DOOR KNOB
glColor3f(0.309804,0.184314,0.184314);
glBegin(GL_POINTS);
glVertex2f(180.0, 85.0);
glEnd();
// SIDE WALL
    glColor3f(0.545, 0.271, 0.075);
glBegin(GL_POLYGON);
glVertex2f(225.0, 70.0);
glVertex2f(225.0, 170.0);
glVertex2f(375.0, 170.0);
glVertex2f(375.0, 70.0);
```

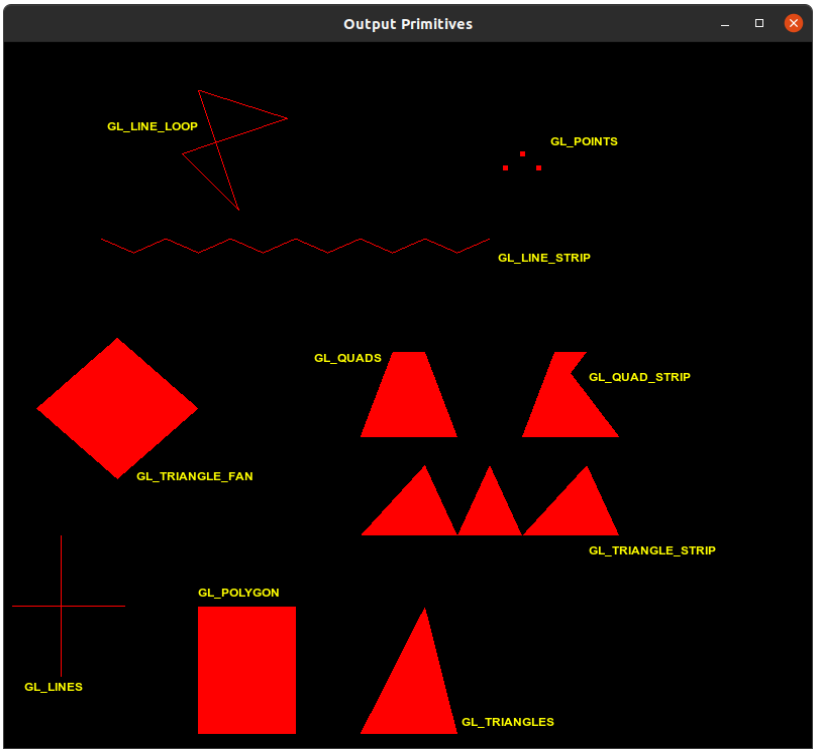
```
glEnd();

glColor3f(0.000, 0.749, 1.000);
glBegin(GL_POLYGON);
glVertex2f(275.0, 105.0);
glVertex2f(275.0, 145.0);
glVertex2f(325.0, 145.0);
glVertex2f(325.0, 105.0);
glEnd();

//GROUND
    glColor3f(0.137255,0.556863,0.137255);
glBegin(GL_POLYGON);
glVertex2f(0.0, 0.0);
glVertex2f(0.0, 70.0);
glVertex2f(600.0, 70.0);
glVertex2f(600.0, 0.0);
glEnd();
// BORDER LINES
glColor3f(0.502, 0.000, 0.000);
glLineWidth(9.0);
glBegin(GL_LINE_LOOP);
glVertex2f(125.0, 170.0);
glVertex2f(175.0, 240.0);
glVertex2f(225.0, 170.0);
glEnd();

glColor3f(0.502, 0.000, 0.000);
glLineWidth(9.0);
glBegin(GL_LINE_LOOP);
glVertex2f(225.0, 170.0);
glVertex2f(175.0, 240.0);
glVertex2f(325.0, 240.0);
glVertex2f(375.0, 170.0);
glEnd();
```

```
glColor3f(0.502, 0.000, 0.000);
glBegin(GL_LINE_STRIP);
glVertex2f(125.0, 170.0);
glVertex2f(125.0, 70.0);
glVertex2f(225.0, 70.0);
glVertex2f(225.0, 170.0);
glEnd();

glColor3f(0.502, 0.000, 0.000);
glBegin(GL_LINE_STRIP);
glVertex2f(225.0, 70.0);
glVertex2f(375.0, 70.0);
glVertex2f(375.0, 170.0);
glEnd();

glColor3f(0.502, 0.000, 0.000);
glBegin(GL_LINE_STRIP);
glVertex2f(300.0, 240.0);
glVertex2f(300.0, 260.0);
glVertex2f(320.0, 260.0);
glVertex2f(320.0, 240.0);
glEnd();

glColor3f(0,0,0);
glBegin(GL_LINE_LOOP);
glVertex2f(275.0, 105.0);
glVertex2f(275.0, 145.0);
glVertex2f(325.0, 145.0);
glVertex2f(325.0, 105.0);
glEnd();
glColor3f(0,0,0);

glBegin(GL_LINES);
glVertex2f(300.0, 105.0);
```

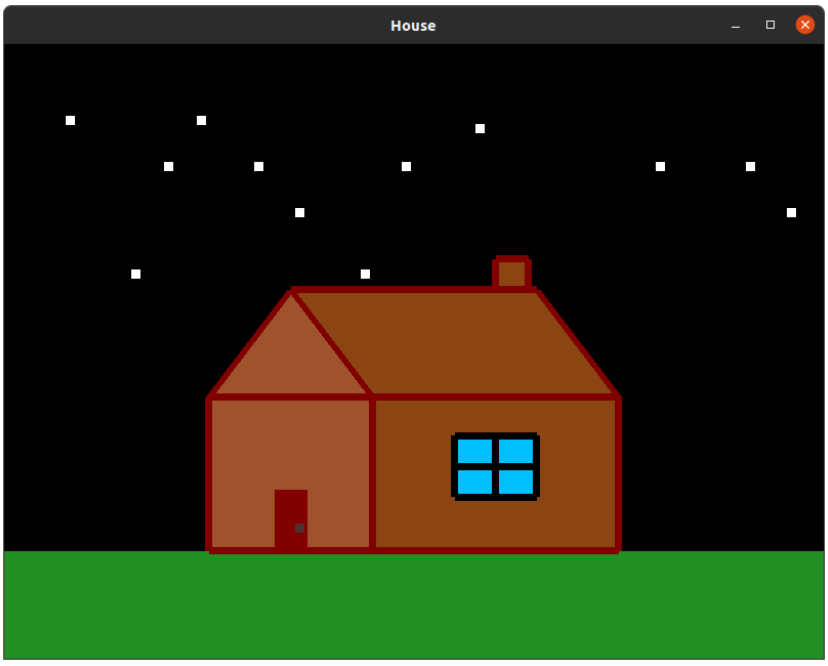
```
    glVertex2f(300.0, 145.0);
    glVertex2f(275.0, 125.0);
    glVertex2f(325.0, 125.0);
    glEnd();

    glFlush();

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

        glutInit (&argc, argv);
        glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);
        glutInitWindowSize (800, 600);
        glutInitWindowPosition (10, 10);
        glutCreateWindow ("House");
        glutDisplayFunc (myDisplay);
        myInit();
        glutMainLoop();
        return 0;
}
```

OUTPUT SCREENSHOTS:

1. Output primitives:



2. House:



RESULT:

Thus we have successfully implemented in drawing the output primitives and a house using openGL.