Ex. No. 1	**OpenGL Primitives**	**Karthik D**
September 8, 2022	**UCS1712 - Graphics and Multimedia Lab**	**195001047**

Aim

To develop a C++ program using the OpenGL framework to study the basic output primitives.

Question

1. To create an output window using OPENGL and to draw the following basic output primitives — POINTS, LINES, LINE_STRIP, LINE_LOOP, TRIANGLES, QUADS, QUAD_STRIP, POLYGON.
2. To create an output window and draw a checkerboard using OpenGL.
3. To create an output window and draw a house using POINTS, LINES, TRIANGLES and QUADS/POLYGON.

OpenGL Algorithm to Generate Primitives

Procedure plotPrimitive(shape_identifier, vertice_xs, vertice);

```
// Initiate the target primitive (POINTS, LINES, etc.)
var shape_identifier;
glBegin(shape_identifier);

// Enumerate all the vertices of the figure being drawn
var l = length(vertice_xs);
for i=0:(l-1)
Begin
        glVertex2d(vertice_xs[i], vertice_ys[i]);
End

// Terminate the target primitive plotting
glEnd();
.
```

End {plotPrimitive}

Implementation using C++ Program Code

1. main.cpp - Driver and Handler to render the primitives, plot the checkerboard, and draw the house

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

#define BUFFER_SIZE 200


void renderSpacedBitmapString(float x, float y, void *font, char
*string) {
   char *c;
   int x1 = x;
   for (c = string; *c != '\0'; c++) {
       glRasterPos2f(x1, y);
       glutBitmapCharacter(font, *c);
       x1 = x1 + glutBitmapWidth(font, *c);
   }
}


void markString(char *string, int x, int y, int x_offset, int y_offset,
short large) {
   void *size = (large ? GLUT_BITMAP_HELVETICA_12 :
GLUT_BITMAP_HELVETICA_10);
   glColor3f(255.0, 0, 0.0); // red color
   printf("\n%s", string);
   renderSpacedBitmapString(x+x_offset, y+y_offset, size, string);
   glFlush();
}


// POINTS, LINES, LINE_STRIP, LINE_LOOP, TRIANGLES, TRIANGLE STRIP,
// TRIANGLE FAN, QUADS, QUAD_STRIP, POLYGON

void plotPoints()   {
```

```
glBegin(GL_POINTS);
// line
glVertex2d(20, 30);
glVertex2d(130, 40);
glVertex2d(170, 50);
glVertex2d(120, 140);
// line strip
glVertex2d(20, 190);
glVertex2d(130, 200);
glVertex2d(50, 230);
glVertex2d(120, 300);
// line loop
glVertex2d(20, 350);
glVertex2d(130, 360);
glVertex2d(170, 390);
glVertex2d(120, 460);
// triangle
glVertex2d(230, 350);
glVertex2d(230, 450);
glVertex2d(370, 400);
// triangle strip
glVertex2d(230, 190);
glVertex2d(230, 240);
glVertex2d(380, 210);
glVertex2d(300, 280);
// triangle fan
glVertex2d(230, 30);
glVertex2d(230, 120);
glVertex2d(380, 150);
glVertex2d(300, 30);
// quads
glVertex2d(500, 450);
glVertex2d(400, 400);
glVertex2d(450, 350);
glVertex2d(600, 400);
// quad strip
glVertex2d(500, 320);
glVertex2d(400, 290);
```

```
   glVertex2d(450, 190);
   glVertex2d(600, 240);
   glVertex2d(620, 350);
   // polygon
   glVertex2d(480, 160);
   glVertex2d(560, 160);
   glVertex2d(600, 70);
   glVertex2d(560, 40);
   glVertex2d(480, 40);
   glVertex2d(410, 90);
   glEnd();
}


void plotLabels()   {

   markString("POINTS and LINES", 10, 150, 0, 0, 1);
   markString("A(20, 30)", 20, 30, -5, 5, 0);
   markString("B(130, 40)", 130, 40, -5, 5, 0);
   markString("C(170, 50)", 170, 50, -5, 5, 0);
   markString("D(120, 140)", 120, 140, -5, 5, 0);

   markString("LINE STRIP", 30, 310, 0, 0, 1);
   markString("E(20, 190)", 20, 190, -5, 5, 0);
   markString("F(130, 200)", 130, 200, -5, 5, 0);
   markString("G(50, 230)", 50, 230, -5, 5, 0);
   markString("H(120, 300)", 120, 300, -5, 5, 0);

   markString("LINE LOOP", 30, 460, 0, 0, 1);
   markString("A(20, 350)", 20, 350, -5, 5, 0);
   markString("B(130, 360)", 130, 360, -5, 5, 0);
   markString("C(170, 390)", 170, 390, -5, 5, 0);
   markString("D(120, 460)", 120, 460, -5, 5, 0);

   markString("TRIANGLES", 300, 460, 0, 0, 1);
   markString("A(230, 350)", 230, 350, 5, -10, 0);
   markString("B(230, 450)", 230, 450, 5, 5, 0);
   markString("C(370, 400)", 370, 400, -20, 10, 0);
```

```
   markString("TRIANGLE STRIP", 240, 310, 0, 0, 1);
   markString("A(230, 190)", 230, 190, 5, -10, 0);
   markString("B(230, 240)", 230, 240, -20, 15, 0);
   markString("C(380, 210)", 380, 210, -10, 10, 0);
   markString("D(300, 280)", 300, 280, -20, 10, 0);

   markString("TRIANGLE FAN", 240, 150, 0, 0, 1);
   markString("A(230, 30)", 230, 30, 5, -10, 0);
   markString("B(230, 120)", 230, 120, -20, 15, 0);
   markString("C(380, 150)", 380, 150, -10, 10, 0);
   markString("D(300, 30)", 300, 30, 10, 10, 0);

   markString("QUADS", 500, 460, 0, 0, 1);
   markString("A(500, 450)", 500, 450, 20, -5, 0);
   markString("B(400, 400)", 400, 400, -40, -30, 0);
   markString("C(450, 350)", 450, 350, 25, 0, 0);
   markString("D(600, 400)", 600, 400, -20, 10, 0);

   markString("QUAD STRIP", 500, 310, 0, 0, 1);
   markString("A(500, 320)", 500, 320, 10, -5, 0);
   markString("B(480, 230)", 480, 230, -55, 10, 0);
   markString("C(550, 190)", 550, 190, -70, 0, 0);
   markString("D(600, 310)", 600, 310, -20, 0, 0);
   markString("E(620, 350)", 620, 220, -35, -25, 0);

   markString("POLYGONS", 500, 180, 0, 0, 1);
   markString("A(480, 160)", 480, 160, 10, 5, 0);
   markString("B(560, 160)", 560, 160, 20, -5, 0);
   markString("C(600, 70)", 600, 70, -15, -20, 0);
   markString("D(560, 40)", 560, 40, -5, -20, 0);
   markString("E(480, 40)", 480, 40, 0, -20, 0);
   markString("F(410, 90)", 410, 90, -25, -30, 0);
}


void plotLines()    {
   glBegin(GL_LINES);
```

```
   glVertex2d(20, 30);
   glVertex2d(130, 40);
   glVertex2d(170, 50);
   glVertex2d(120, 140);
   glEnd();
}


void plotLineStrip()    {
   glBegin(GL_LINE_STRIP);
   glVertex2d(20, 190);
   glVertex2d(130, 200);
   glVertex2d(50, 230);
   glVertex2d(120, 300);
   glEnd();
}


void plotLineLoop() {
   glBegin(GL_LINE_LOOP);
   glVertex2d(20, 350);
   glVertex2d(130, 360);
   glVertex2d(170, 390);
   glVertex2d(120, 460);
   glEnd();
}


void plotTriangles()    {
   glBegin(GL_TRIANGLES);
   glVertex2d(230, 350);
   glVertex2d(230, 450);
   glVertex2d(370, 400);
   glEnd();
}


void plotTriangleStrip()    {
```

```
   glBegin(GL_TRIANGLE_STRIP);
   glVertex2d(230, 190);
   glVertex2d(230, 240);
   glVertex2d(380, 210);
   glVertex2d(300, 280);
   glEnd();
}


void plotTriangleFan()  {
   glBegin(GL_TRIANGLE_FAN);
   glVertex2d(230, 30);
   glVertex2d(230, 120);
   glVertex2d(380, 150);
   glVertex2d(300, 30);
   glEnd();
}


void plotQuads()    {
   glBegin(GL_QUADS);
   glVertex2d(500, 450);
   glVertex2d(400, 400);
   glVertex2d(450, 350);
   glVertex2d(600, 400);
   glEnd();
}


void plotQuadStrip()    {
   glBegin(GL_QUAD_STRIP);
   glVertex2d(500, 320);
   glVertex2d(480, 230);
   glVertex2d(540, 300);
   glVertex2d(550, 190);
   glVertex2d(600, 310);
   glVertex2d(620, 220);
   glEnd();
```

```
}


void plotPolygon()    {
   glBegin(GL_POLYGON);
   glVertex2d(480, 160);
   glVertex2d(560, 160);
   glVertex2d(600, 70);
   glVertex2d(560, 40);
   glVertex2d(480, 40);
   glVertex2d(410, 90);
   glEnd();
}


void plotBlackBox(int topleft_x, int topleft_y, int dimension) {
   glBegin(GL_QUADS);
   glVertex2d(topleft_x, topleft_y);
   glVertex2d(topleft_x+dimension, topleft_y);
   glVertex2d(topleft_x+dimension, topleft_y+dimension);
   glVertex2d(topleft_x, topleft_y+dimension);
   glEnd();
}


void plotWhiteBox(int topleft_x, int topleft_y, int dimension) {
   glBegin(GL_LINE_LOOP);
   glVertex2d(topleft_x, topleft_y);
   glVertex2d(topleft_x+dimension, topleft_y);
   glVertex2d(topleft_x+dimension, topleft_y+dimension);
   glVertex2d(topleft_x, topleft_y+dimension);
   glEnd();
}


void plotCheckboard(int n_rows, int n_cols, int topleft_x, int
topleft_y, int dimension) {
   int x_posn = topleft_x;
```

```
    int y_posn = topleft_y;
    for(int i=0; i<n_rows; i++)    {
        for(int j=0; j<n_cols; j++) {
            if((i+j)%2==0){
                plotBlackBox(x_posn, y_posn, dimension);
            }
            else{
                plotWhiteBox(x_posn, y_posn, dimension);
            }
            y_posn += dimension;
        }
        x_posn += dimension;
        y_posn = topleft_y;
    }
}


void plotHouse()    {
    // Roof
    glColor3f(0.3, 0.5, 0.8);
    glBegin(GL_POLYGON);
    glVertex2d(200, 500);
    glVertex2d(600, 500);
    glVertex2d(700, 350);
    glVertex2d(300, 350);
    glEnd();
    // Top of Front Wall
    glColor3f(0.1, 0.5, 0.0);
    glBegin(GL_TRIANGLES);
    glVertex2d(200, 500);
    glVertex2d(100, 350);
    glVertex2d(300, 350);
    glEnd();
    // Front Wall
    glColor3f(0.7, 0.2, 0.3);
    glBegin(GL_POLYGON);
    glVertex2d(100, 350);
    glVertex2d(300, 350);
```

```
  glVertex2d(300, 100);
  glVertex2d(100, 100);
  glEnd();
  // Front Door
  glColor3f(0.7, 0.2, 0.9);
  glBegin(GL_POLYGON);
  glVertex2d(150, 250);
  glVertex2d(250, 250);
  glVertex2d(250, 100);
  glVertex2d(150, 100);
  glEnd();
  // Front Door Lock
  glColor3f(0.3, 0.7, 0.9);
  glPointSize(15);
  glBegin(GL_POINTS);
  glVertex2d(170, 170);
  glEnd();
  //  Side Wall
  glColor3f(0.1, 0.2, 0.3);
  glBegin(GL_POLYGON);
  glVertex2d(300, 350);
  glVertex2d(700, 350);
  glVertex2d(700, 100);
  glVertex2d(300, 100);
  glEnd();
  // Window one
  glColor3f(0.2, 0.4, 0.3);
  glBegin(GL_POLYGON);
  glVertex2d(330, 320);
  glVertex2d(450, 320);
  glVertex2d(450, 230);
  glVertex2d(330, 230);
  glEnd();
  // line of window one
  glColor3f(0.1, 0.7, 0.5);
  glLineWidth(5);
  glBegin(GL_LINES);
  glVertex2d(390, 320);
```

```
    glVertex2d(390, 230);
    glVertex2d(330, 273);
    glVertex2d(450, 273);
    glEnd();
    // Window two
    glColor3f(0.2, 0.4, 0.3);
    glBegin(GL_POLYGON);
    glVertex2d(530, 320);
    glVertex2d(650, 320);
    glVertex2d(650, 230);
    glVertex2d(530, 230);
    glEnd();
    // lines of window two
    glColor3f(0.1, 0.7, 0.5);
    glLineWidth(5);
    glBegin(GL_LINES);
    glVertex2d(590, 320);
    glVertex2d(590, 230);
    glVertex2d(530, 273);
    glVertex2d(650, 273);
    glEnd();
    // Entrance Path
    glColor3f(0.3, 0.5, 0.7);
    glLineWidth(3);
    glBegin(GL_POLYGON);
    glVertex2d(150, 100);
    glVertex2d(250, 100);
    glVertex2d(210, 0);
    glVertex2d(40, 0);
    glEnd();
}


void display_primitives()    {
    glClear(GL_COLOR_BUFFER_BIT);
    plotPoints();
    plotLabels();
    glColor3f(0.0f, 0.0f, 0.0f);
```

```
   plotLines();
   plotLineStrip();
   plotLineLoop();
   plotTriangles();
   plotTriangleStrip();
   plotTriangleFan();
   plotQuads();
   plotQuadStrip();
   plotPolygon();
   glFlush();
}


void display_checkboard()   {
   glClear(GL_COLOR_BUFFER_BIT);
   plotCheckboard(8, 8, 160, 80, 40);
   glFlush();
}


void display_house()    {
   glClear(GL_COLOR_BUFFER_BIT);
   plotHouse();
   glFlush();
}


void init()   {
   glClearColor(1.0, 1.0, 1.0,0.0);
   glColor3f(0.0f, 0.0f, 0.0f);
   glPointSize(10);
   glMatrixMode(GL_PROJECTION);
   glLoadIdentity();
   gluOrtho2D(0.0, 800.0, 0.0, 600.0);
}


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

```
   glutInit(&argc,argv);
   glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
   glutInitWindowSize(640, 480);

   // primitives
   // glutCreateWindow("Ex1A - OpenGL Primitives");
   // glutDisplayFunc(display_primitives);

   // checkboard
   // glutCreateWindow("Ex1B - Checkboard Pattern");
   // glutDisplayFunc(display_checkboard);

   // house
   glutCreateWindow("Ex1C - House");
   glutDisplayFunc(display_house);

   init();
   glutMainLoop();
   return 1;
}
#include <GL/glut.h>
#include <stdio.h>

#define BUFFER_SIZE 200


void renderSpacedBitmapString(float x, float y, void *font, char
*string) {
   char *c;
   int x1 = x;
   for (c = string; *c != '\0'; c++) {
       glRasterPos2f(x1, y);
       glutBitmapCharacter(font, *c);
       x1 = x1 + glutBitmapWidth(font, *c);
   }
}
```

```
void markString(char *string, int x, int y, int x_offset, int y_offset,
short large) {
   void *size = (large ? GLUT_BITMAP_HELVETICA_12 :
GLUT_BITMAP_HELVETICA_10);
   glColor3f(255.0, 0, 0.0); // red color
   printf("\n%s", string);
   renderSpacedBitmapString(x+x_offset, y+y_offset, size, string);
   glFlush();
}


// POINTS, LINES, LINE_STRIP, LINE_LOOP, TRIANGLES, TRIANGLE STRIP,
// TRIANGLE FAN, QUADS, QUAD_STRIP, POLYGON

void plotPoints()   {
   glBegin(GL_POINTS);
   // line
   glVertex2d(20, 30);
   glVertex2d(130, 40);
   glVertex2d(170, 50);
   glVertex2d(120, 140);
   // line strip
   glVertex2d(20, 190);
   glVertex2d(130, 200);
   glVertex2d(50, 230);
   glVertex2d(120, 300);
   // line loop
   glVertex2d(20, 350);
   glVertex2d(130, 360);
   glVertex2d(170, 390);
   glVertex2d(120, 460);
   // triangle
   glVertex2d(230, 350);
   glVertex2d(230, 450);
   glVertex2d(370, 400);
   // triangle strip
   glVertex2d(230, 190);
```

```
   glVertex2d(230, 240);
   glVertex2d(380, 210);
   glVertex2d(300, 280);
   // triangle fan
   glVertex2d(230, 30);
   glVertex2d(230, 120);
   glVertex2d(380, 150);
   glVertex2d(300, 30);
   // quads
   glVertex2d(500, 450);
   glVertex2d(400, 400);
   glVertex2d(450, 350);
   glVertex2d(600, 400);
   // quad strip
   glVertex2d(500, 320);
   glVertex2d(400, 290);
   glVertex2d(450, 190);
   glVertex2d(600, 240);
   glVertex2d(620, 350);
   // polygon
   glVertex2d(480, 160);
   glVertex2d(560, 160);
   glVertex2d(600, 70);
   glVertex2d(560, 40);
   glVertex2d(480, 40);
   glVertex2d(410, 90);
   glEnd();
}


void plotLabels()   {

   markString("POINTS and LINES", 10, 150, 0, 0, 1);
   markString("A(20, 30)", 20, 30, -5, 5, 0);
   markString("B(130, 40)", 130, 40, -5, 5, 0);
   markString("C(170, 50)", 170, 50, -5, 5, 0);
   markString("D(120, 140)", 120, 140, -5, 5, 0);
```

```
  markString("LINE STRIP", 30, 310, 0, 0, 1);
  markString("E(20, 190)", 20, 190, -5, 5, 0);
  markString("F(130, 200)", 130, 200, -5, 5, 0);
  markString("G(50, 230)", 50, 230, -5, 5, 0);
  markString("H(120, 300)", 120, 300, -5, 5, 0);

  markString("LINE LOOP", 30, 460, 0, 0, 1);
  markString("A(20, 350)", 20, 350, -5, 5, 0);
  markString("B(130, 360)", 130, 360, -5, 5, 0);
  markString("C(170, 390)", 170, 390, -5, 5, 0);
  markString("D(120, 460)", 120, 460, -5, 5, 0);

  markString("TRIANGLES", 300, 460, 0, 0, 1);
  markString("A(230, 350)", 230, 350, 5, -10, 0);
  markString("B(230, 450)", 230, 450, 5, 5, 0);
  markString("C(370, 400)", 370, 400, -20, 10, 0);

  markString("TRIANGLE STRIP", 240, 310, 0, 0, 1);
  markString("A(230, 190)", 230, 190, 5, -10, 0);
  markString("B(230, 240)", 230, 240, -20, 15, 0);
  markString("C(380, 210)", 380, 210, -10, 10, 0);
  markString("D(300, 280)", 300, 280, -20, 10, 0);

  markString("TRIANGLE FAN", 240, 150, 0, 0, 1);
  markString("A(230, 30)", 230, 30, 5, -10, 0);
  markString("B(230, 120)", 230, 120, -20, 15, 0);
  markString("C(380, 150)", 380, 150, -10, 10, 0);
  markString("D(300, 30)", 300, 30, 10, 10, 0);

  markString("QUADS", 500, 460, 0, 0, 1);
  markString("A(500, 450)", 500, 450, 20, -5, 0);
  markString("B(400, 400)", 400, 400, -40, -30, 0);
  markString("C(450, 350)", 450, 350, 25, 0, 0);
  markString("D(600, 400)", 600, 400, -20, 10, 0);

  markString("QUAD STRIP", 500, 310, 0, 0, 1);
  markString("A(500, 320)", 500, 320, 10, -5, 0);
  markString("B(480, 230)", 480, 230, -55, 10, 0);
```

```
   markString("C(550, 190)", 550, 190, -70, 0, 0);
   markString("D(600, 310)", 600, 310, -20, 0, 0);
   markString("E(620, 350)", 620, 220, -35, -25, 0);

   markString("POLYGONS", 500, 180, 0, 0, 1);
   markString("A(480, 160)", 480, 160, 10, 5, 0);
   markString("B(560, 160)", 560, 160, 20, -5, 0);
   markString("C(600, 70)", 600, 70, -15, -20, 0);
   markString("D(560, 40)", 560, 40, -5, -20, 0);
   markString("E(480, 40)", 480, 40, 0, -20, 0);
   markString("F(410, 90)", 410, 90, -25, -30, 0);
}


void plotLines()    {
   glBegin(GL_LINES);
   glVertex2d(20, 30);
   glVertex2d(130, 40);
   glVertex2d(170, 50);
   glVertex2d(120, 140);
   glEnd();
}


void plotLineStrip()    {
   glBegin(GL_LINE_STRIP);
   glVertex2d(20, 190);
   glVertex2d(130, 200);
   glVertex2d(50, 230);
   glVertex2d(120, 300);
   glEnd();
}


void plotLineLoop() {
   glBegin(GL_LINE_LOOP);
   glVertex2d(20, 350);
   glVertex2d(130, 360);
```

```
   glVertex2d(170, 390);
   glVertex2d(120, 460);
   glEnd();
}


void plotTriangles()    {
   glBegin(GL_TRIANGLES);
   glVertex2d(230, 350);
   glVertex2d(230, 450);
   glVertex2d(370, 400);
   glEnd();
}


void plotTriangleStrip()    {
   glBegin(GL_TRIANGLE_STRIP);
   glVertex2d(230, 190);
   glVertex2d(230, 240);
   glVertex2d(380, 210);
   glVertex2d(300, 280);
   glEnd();
}


void plotTriangleFan()  {
   glBegin(GL_TRIANGLE_FAN);
   glVertex2d(230, 30);
   glVertex2d(230, 120);
   glVertex2d(380, 150);
   glVertex2d(300, 30);
   glEnd();
}


void plotQuads()    {
   glBegin(GL_QUADS);
   glVertex2d(500, 450);
```

```
   glVertex2d(400, 400);
   glVertex2d(450, 350);
   glVertex2d(600, 400);
   glEnd();
}


void plotQuadStrip()    {
   glBegin(GL_QUAD_STRIP);
   glVertex2d(500, 320);
   glVertex2d(480, 230);
   glVertex2d(540, 300);
   glVertex2d(550, 190);
   glVertex2d(600, 310);
   glVertex2d(620, 220);
   glEnd();
}


void plotPolygon()    {
   glBegin(GL_POLYGON);
   glVertex2d(480, 160);
   glVertex2d(560, 160);
   glVertex2d(600, 70);
   glVertex2d(560, 40);
   glVertex2d(480, 40);
   glVertex2d(410, 90);
   glEnd();
}


void plotBlackBox(int topleft_x, int topleft_y, int dimension) {
   glBegin(GL_QUADS);
   glVertex2d(topleft_x, topleft_y);
   glVertex2d(topleft_x+dimension, topleft_y);
   glVertex2d(topleft_x+dimension, topleft_y+dimension);
   glVertex2d(topleft_x, topleft_y+dimension);
   glEnd();
```

```
}


void plotWhiteBox(int topleft_x, int topleft_y, int dimension) {
   glBegin(GL_LINE_LOOP);
   glVertex2d(topleft_x, topleft_y);
   glVertex2d(topleft_x+dimension, topleft_y);
   glVertex2d(topleft_x+dimension, topleft_y+dimension);
   glVertex2d(topleft_x, topleft_y+dimension);
   glEnd();
}


void plotCheckboard(int n_rows, int n_cols, int topleft_x, int
topleft_y, int dimension) {
   int x_posn = topleft_x;
   int y_posn = topleft_y;
   for(int i=0; i<n_rows; i++)    {
       for(int j=0; j<n_cols; j++) {
           if((i+j)%2==0){
               plotBlackBox(x_posn, y_posn, dimension);
           }
           else{
               plotWhiteBox(x_posn, y_posn, dimension);
           }
           y_posn += dimension;
       }
       x_posn += dimension;
       y_posn = topleft_y;
   }
}


void plotHouse()    {
   // Roof
   glColor3f(0.3, 0.5, 0.8);
   glBegin(GL_POLYGON);
   glVertex2d(200, 500);
```

```
  glVertex2d(600, 500);
  glVertex2d(700, 350);
  glVertex2d(300, 350);
  glEnd();
  // Top of Front Wall
  glColor3f(0.1, 0.5, 0.0);
  glBegin(GL_TRIANGLES);
  glVertex2d(200, 500);
  glVertex2d(100, 350);
  glVertex2d(300, 350);
  glEnd();
  // Front Wall
  glColor3f(0.7, 0.2, 0.3);
  glBegin(GL_POLYGON);
  glVertex2d(100, 350);
  glVertex2d(300, 350);
  glVertex2d(300, 100);
  glVertex2d(100, 100);
  glEnd();
  // Front Door
  glColor3f(0.7, 0.2, 0.9);
  glBegin(GL_POLYGON);
  glVertex2d(150, 250);
  glVertex2d(250, 250);
  glVertex2d(250, 100);
  glVertex2d(150, 100);
  glEnd();
  // Front Door Lock
  glColor3f(0.3, 0.7, 0.9);
  glPointSize(15);
  glBegin(GL_POINTS);
  glVertex2d(170, 170);
  glEnd();
  //  Side Wall
  glColor3f(0.1, 0.2, 0.3);
  glBegin(GL_POLYGON);
  glVertex2d(300, 350);
  glVertex2d(700, 350);
```

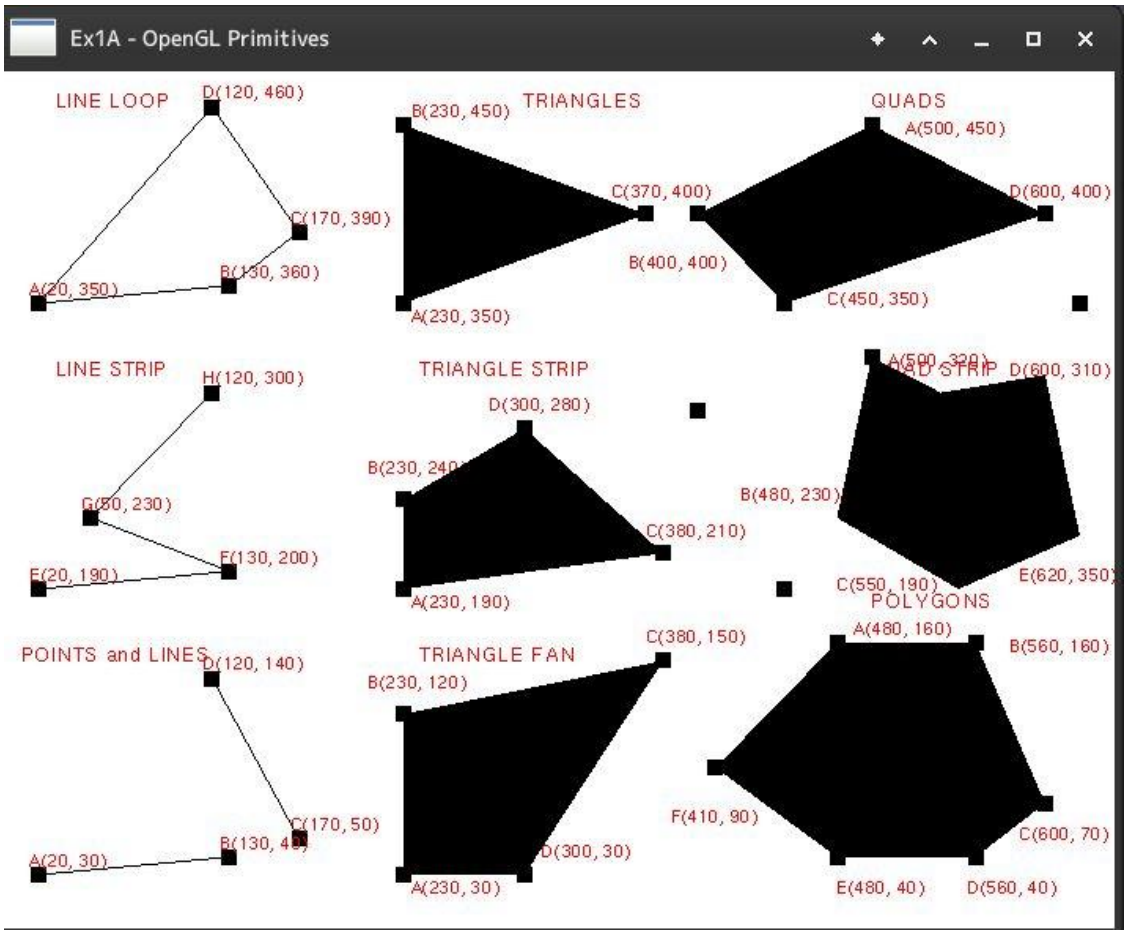
```
  glVertex2d(700, 100);
  glVertex2d(300, 100);
  glEnd();
  // Window one
  glColor3f(0.2, 0.4, 0.3);
  glBegin(GL_POLYGON);
  glVertex2d(330, 320);
  glVertex2d(450, 320);
  glVertex2d(450, 230);
  glVertex2d(330, 230);
  glEnd();
  // line of window one
  glColor3f(0.1, 0.7, 0.5);
  glLineWidth(5);
  glBegin(GL_LINES);
  glVertex2d(390, 320);
  glVertex2d(390, 230);
  glVertex2d(330, 273);
  glVertex2d(450, 273);
  glEnd();
  // Window two
  glColor3f(0.2, 0.4, 0.3);
  glBegin(GL_POLYGON);
  glVertex2d(530, 320);
  glVertex2d(650, 320);
  glVertex2d(650, 230);
  glVertex2d(530, 230);
  glEnd();
  // lines of window two
  glColor3f(0.1, 0.7, 0.5);
  glLineWidth(5);
  glBegin(GL_LINES);
  glVertex2d(590, 320);
  glVertex2d(590, 230);
  glVertex2d(530, 273);
  glVertex2d(650, 273);
  glEnd();
  // Entrance Path
```

```
   glColor3f(0.3, 0.5, 0.7);
   glLineWidth(3);
   glBegin(GL_POLYGON);
   glVertex2d(150, 100);
   glVertex2d(250, 100);
   glVertex2d(210, 0);
   glVertex2d(40, 0);
   glEnd();
}


void display_primitives()    {
   glClear(GL_COLOR_BUFFER_BIT);
   plotPoints();
   plotLabels();
   glColor3f(0.0f, 0.0f, 0.0f);
   plotLines();
   plotLineStrip();
   plotLineLoop();
   plotTriangles();
   plotTriangleStrip();
   plotTriangleFan();
   plotQuads();
   plotQuadStrip();
   plotPolygon();
   glFlush();
}


void display_checkboard()   {
   glClear(GL_COLOR_BUFFER_BIT);
   plotCheckboard(8, 8, 160, 80, 40);
   glFlush();
}


void display_house()    {
   glClear(GL_COLOR_BUFFER_BIT);
```

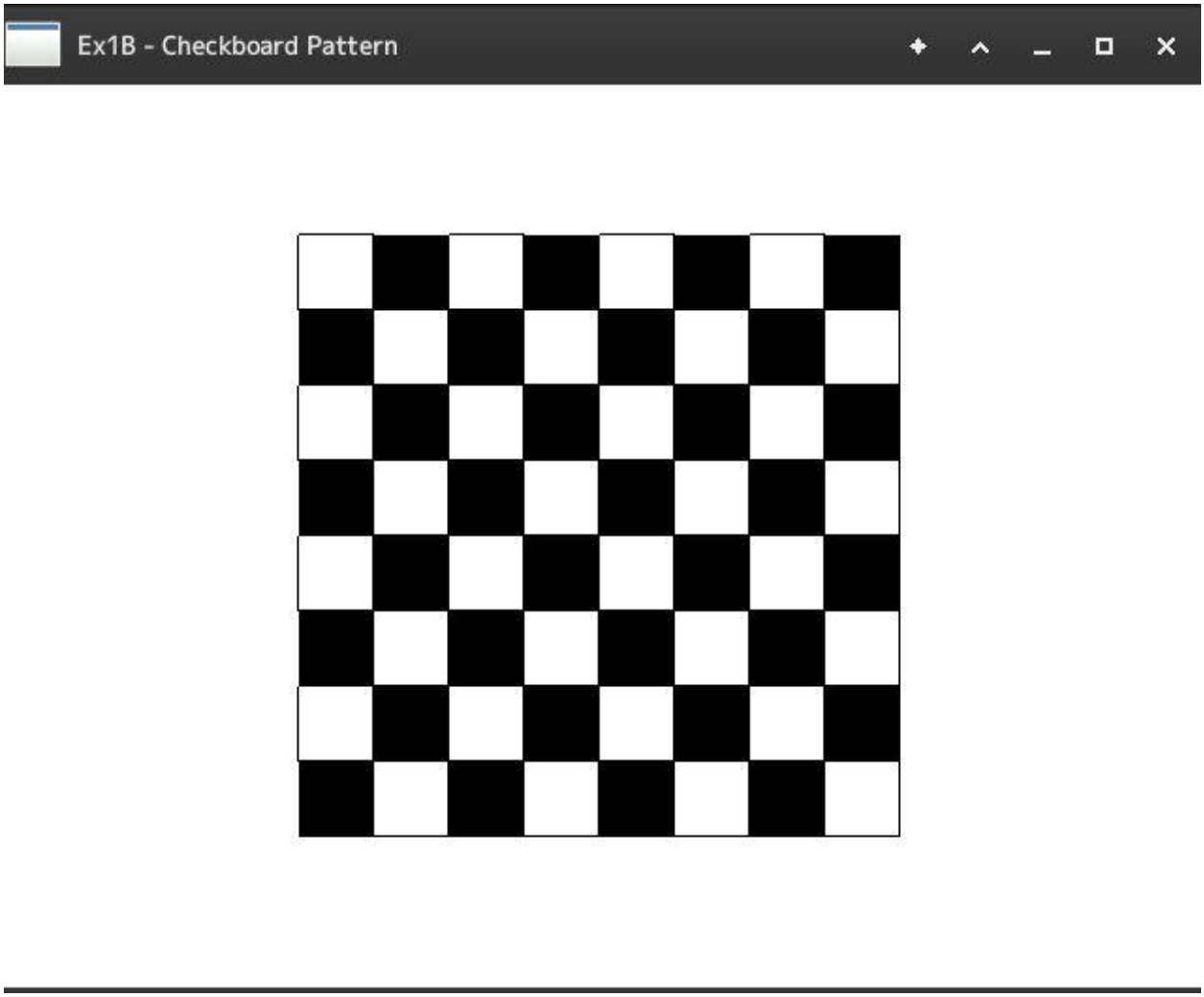
```
   plotHouse();
   glFlush();
}


void init()   {
   glClearColor(1.0, 1.0, 1.0,0.0);
   glColor3f(0.0f, 0.0f, 0.0f);
   glPointSize(10);
   glMatrixMode(GL_PROJECTION);
   glLoadIdentity();
   gluOrtho2D(0.0, 800.0, 0.0, 600.0);
}


int main(int argc,char* argv[]) {

   glutInit(&argc,argv);
   glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
   glutInitWindowSize(640, 480);

   // primitives
   // glutCreateWindow("Ex1A - OpenGL Primitives");
   // glutDisplayFunc(display_primitives);

   // checkboard
   // glutCreateWindow("Ex1B - Checkboard Pattern");
   // glutDisplayFunc(display_checkboard);

   // house
   glutCreateWindow("Ex1C - House");
   glutDisplayFunc(display_house);

   init();
   glutMainLoop();
   return 1;
}
```

Sample Output

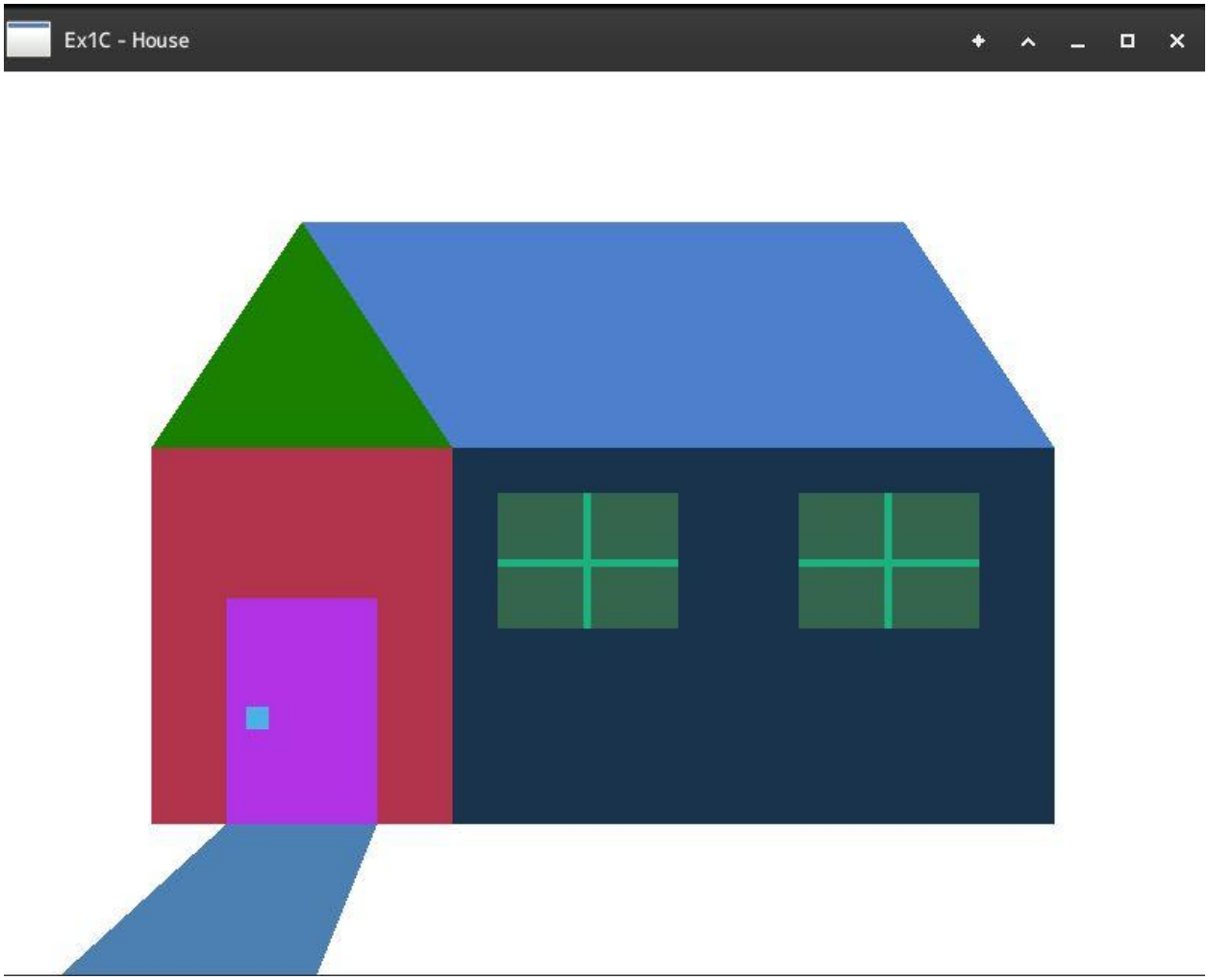
a. Primitive Shapes



b. Checkerboard Pattern



c. House drawn using primitive shapes



Learning Outcomes

Through this implementation of primitive shape based constructions using the OpenGL framework and C++ programming language, the following concepts were learnt:

1. The working of plotting in the OpenGL framework
2. The workflow and procedure to construct a complex shape using primitives
3. General understanding of the OpenGL framework and its APIs