UCS1712 – GRAPHICS AND MULTIMEDIA LAB
MODEL PRACTICAL EXAMINATION OCT–2021

29-10-21

VENKATARAMAN N
CSE – C
185001192

QUESTIONS

31) (a) Construct a C++ Program using OpenGL to draw the greeting message in our mother tongue
(b) Draw a teapot and apply lighting

PROGRAM-1

AIM:

To construct a C++ Program using OpenGL to draw the greeting message in our mother tongue

CODE:

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

#include<iostream>
#include<vector>
#include<algorithm>
#include<cmath>
#include<utility>
#include<stdlib.h>

using namespace std;

typedef double ld;
typedef long long ll;

const int WINDOW_WIDTH = 1400;
const int WINDOW_HEIGHT = 700;

const int X_LIMIT = 200;
const int Y_LIMIT = 100;

const int SCREEN_FPS = 60;

enum Dir{LEFT, RIGHT, UP};

void myInit();
void myDisplay();

void MidPointCircleAlgo(ld x, ld y, ld r, Dir dir);
void BresenhamLineAlgo(ld X1, ld Y1, ld X2, ld Y2);

int main(int argc, char* argv[]) {
      glutInit(&argc,argv);
      glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB|GLUT_DEPTH);
      glutInitWindowSize(WINDOW_WIDTH, WINDOW_HEIGHT);
      glutCreateWindow("Greeting in Mother Toungue");
```

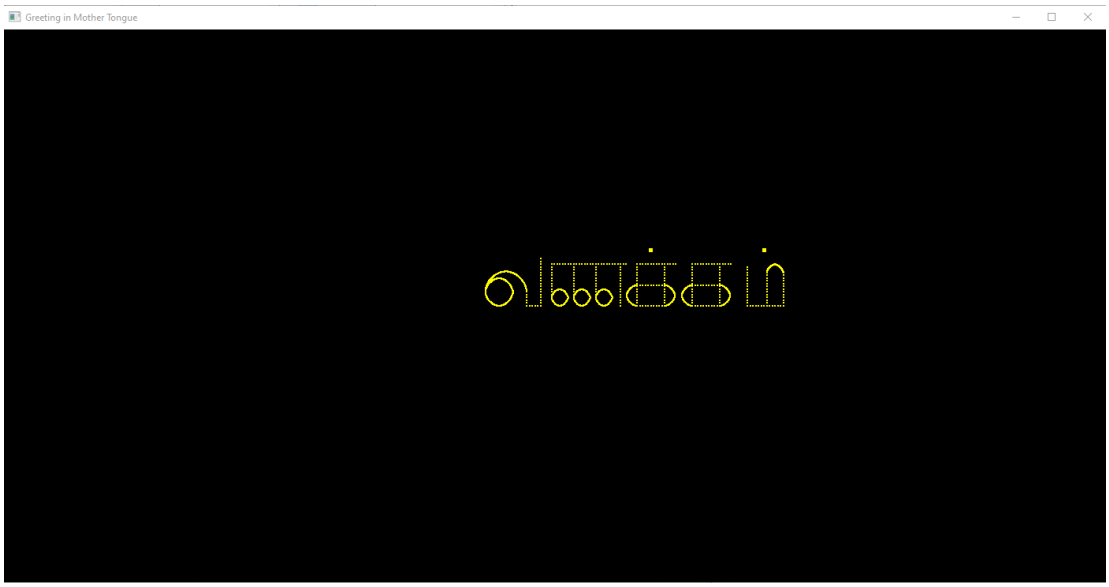
```
	myInit();

	glutDisplayFunc(myDisplay);
	glutMainLoop();
	return 0;
}

void myInit() {
	glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
	glClearColor(0,0,0,1);

	glPointSize(2);
	glLineWidth(3);

	glMatrixMode(GL_PROJECTION);
	glLoadIdentity();
	gluOrtho2D(-X_LIMIT,X_LIMIT,-Y_LIMIT,Y_LIMIT);

	glEnable(GL_BLEND);
	glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
	glEnable(GL_DEPTH_TEST);
}

void myDisplay() {
	glColor3f(1,1,0);

	//Draw "va"
	MidPointCircleAlgo(-20,5,5, LEFT);
	MidPointCircleAlgo(-20,5,5, RIGHT);

	MidPointCircleAlgo(-17.5, 5, 7.5, UP);
	vector<pair<ld,ld>> points;

	//points.push_back(make_pair(-25,5));
	//points.push_back(make_pair(-25,15));
	points.push_back(make_pair(-10,5));
	points.push_back(make_pair(-10,0));
	points.push_back(make_pair(-5,0));
	points.push_back(make_pair(-5,18));



	for(int i=1;i<points.size();i++) {
		BresenhamLineAlgo(points[i-1].first, points[i-1].second,
points[i].first, points[i].second);
	}

	// Draw "na"
	MidPointCircleAlgo(2,3,3,LEFT);
	MidPointCircleAlgo(2,3,3,RIGHT);

	MidPointCircleAlgo(10,3,3,LEFT);
	MidPointCircleAlgo(10,3,3,RIGHT);

	MidPointCircleAlgo(18,3,3,LEFT);
	MidPointCircleAlgo(18,3,3,RIGHT);

	BresenhamLineAlgo(-1,3,-1,15);
	BresenhamLineAlgo(7,3,7,15);
	BresenhamLineAlgo(15,3,15,15);
	BresenhamLineAlgo(-1,15,27,15);
	BresenhamLineAlgo(24,0,24,15);

	//Draw "ik"
```

```
	BresenhamLineAlgo(30,0, 30, 15);
	BresenhamLineAlgo(30,15, 45, 15);
	BresenhamLineAlgo(40,0, 40, 15);
	BresenhamLineAlgo(30,7.5, 40, 7.5);
	BresenhamLineAlgo(30,0, 40, 0);

	MidPointCircleAlgo(30, 3.75, 3.75, LEFT);
	MidPointCircleAlgo(40, 3.75, 3.75, RIGHT);

	glPointSize(5);
	glBegin(GL_POINTS);
		glVertex2d(35, 20);
	glEnd();
	glPointSize(2);

	//Draw "ka"
	BresenhamLineAlgo(50,0, 50, 15);
	BresenhamLineAlgo(50,15, 65, 15);
	BresenhamLineAlgo(60,0, 60, 15);
	BresenhamLineAlgo(50,7.5, 60, 7.5);
	BresenhamLineAlgo(50,0, 60, 0);

	MidPointCircleAlgo(50, 3.75, 3.75, LEFT);
	MidPointCircleAlgo(60, 3.75, 3.75, RIGHT);

	//Draw "him"
	BresenhamLineAlgo(70, 0, 70, 15);
	BresenhamLineAlgo(70, 0, 83, 0);
	BresenhamLineAlgo(77, 0, 77, 12);
	BresenhamLineAlgo(83, 0, 83, 12);

	MidPointCircleAlgo(80, 12, 3, UP);

	glPointSize(5);
	glBegin(GL_POINTS);
		glVertex2d(76, 20);
	glEnd();
	glPointSize(2);
	glFlush();
}

void BresenhamLineAlgo(ld X1, ld Y1, ld X2, ld Y2) {
	ld dx = abs(X2-X1);
	ld dy = abs(Y2-Y1);

	ld stepX, stepY;

	if(X2 > X1) stepX = 1;
	else        stepX = -1;

	if(Y2 > Y1) stepY = 1;
	else        stepY = -1;

	ld x = X1, y = Y1, xEnd = X2, yEnd = Y2, p;

	if(dx > dy) {
		p = 2*dy - dx;

		glBegin(GL_POINTS);
		while(x != xEnd) {
			glVertex2d(x, y);

			x += stepX;
```

```
                if(p < 0)
                        p += 2*dy;
                else {
                        p += 2*(dy-dx);
                        y += stepY;
                }
        }
        glEnd();
    } else {
        p = 2*dx - dy;

        glBegin(GL_POINTS);
        while(y != yEnd) {
                glVertex2d(x, y);

                y += stepY;

                if(p < 0)
                        p += 2*dx;
                else {
                        p += 2*(dx-dy);
                        x += stepX;
                }
        }
        glEnd();
    }
}

void plotLeftPoints(ld x0, ld y0, ld x, ld y) {
    glBegin(GL_POINTS);
        glVertex2d(x0-x, y0+y);
        glVertex2d(x0-x, y0-y);
        glVertex2d(x0-y, y0+x);
        glVertex2d(x0-y, y0-x);
    glEnd();
}

void plotRightPoints(ld x0, ld y0, ld x, ld y) {
    glBegin(GL_POINTS);
        glVertex2d(x0+x, y0+y);
        glVertex2d(x0+x, y0-y);
        glVertex2d(x0+y, y0+x);
        glVertex2d(x0+y, y0-x);
    glEnd();
}

void plotTopPoints(ld x0, ld y0, ld x, ld y) {
    glBegin(GL_POINTS);
        glVertex2d(x0+x, y0+y);
        glVertex2d(x0-x, y0+y);
        glVertex2d(x0+y, y0+x);
        glVertex2d(x0-y, y0+x);
    glEnd();
}



void MidPointCircleAlgo(ld x0, ld y0, ld r, Dir dir) {
    ld p = 1 - r;
    ld x = 0, y = r;

    while(x < y) {
        if(dir == LEFT) plotLeftPoints(x0, y0, x, y);
        if(dir == RIGHT) plotRightPoints(x0, y0, x, y);
```

```
            if(dir == UP) plotTopPoints(x0, y0, x, y);

            x+=0.1;

            if(p < 0)
                  p += 1 + 2*x;
            else {
                  p += 1 + 2*(x-y);
                  y-=0.1;
            }
      }
}
```

OUTPUT:



PROGRAM-1

AIM:

To draw a teapot and apply Lighting

CODE:

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

#include<iostream>
#include<vector>
#include<algorithm>
#include<cmath>
#include<utility>
#include<stdlib.h>

using namespace std;

typedef double ld;
typedef long long ll;

const int WINDOW_WIDTH = 700;
const int WINDOW_HEIGHT = 700;

const int X_LIMIT = 2;
const int Y_LIMIT = 2;
const int Z_LIMIT = 2;

const int SCREEN_FPS = 60;

void myInit();
void myDisplay();

int main(int argc, char* argv[]) {
	glutInit(&argc,argv);
	glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB|GLUT_DEPTH);
	glutInitWindowSize(WINDOW_WIDTH, WINDOW_HEIGHT);
	glutCreateWindow("Teopot Lighting");

	myInit();

	glutDisplayFunc(myDisplay);
	glutMainLoop();
	return 0;
}

void myInit() {
	glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
	glClearColor(1,1,1,1);

	glPointSize(5);
	glLineWidth(3);

	glMatrixMode(GL_PROJECTION);
	glLoadIdentity();
	glOrtho(-X_LIMIT,X_LIMIT,-Y_LIMIT,Y_LIMIT,-Z_LIMIT,Z_LIMIT);

	glEnable(GL_BLEND);
	glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
	glEnable(GL_DEPTH_TEST);
```

```
        glShadeModel(GL_SMOOTH);

        GLfloat light_diffuse[] = {0.7, 0.7, 0.7, 1};
        GLfloat light_position[] = {0,0,1,0};

        glLightfv(GL_LIGHT0, GL_DIFFUSE, light_diffuse);
        glLightfv(GL_LIGHT0, GL_POSITION, light_position);

        glEnable(GL_LIGHTING);
        glEnable(GL_LIGHT0);
}

void myDisplay() {

        glPushMatrix();
        glRotated(180, 0, 1, 0);
        glutSolidTeapot(1);
        glPopMatrix();

        glFlush();
}
```

OUTPUT: