UNIVERSIDAD NACIONAL

AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERIA

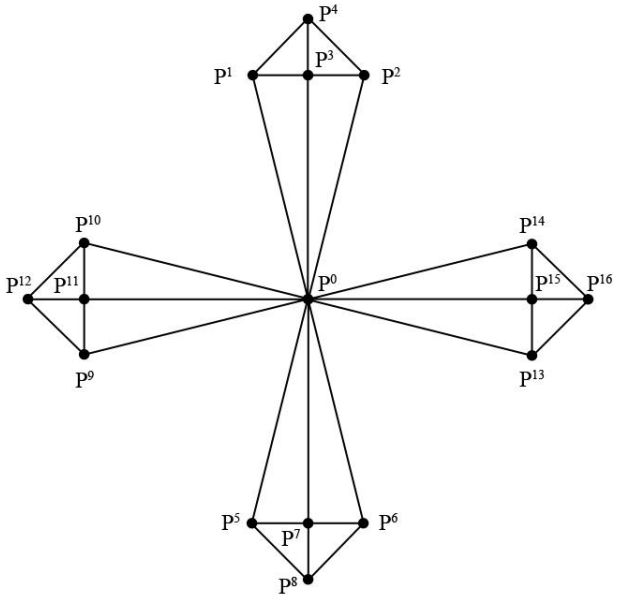
Tarea 2

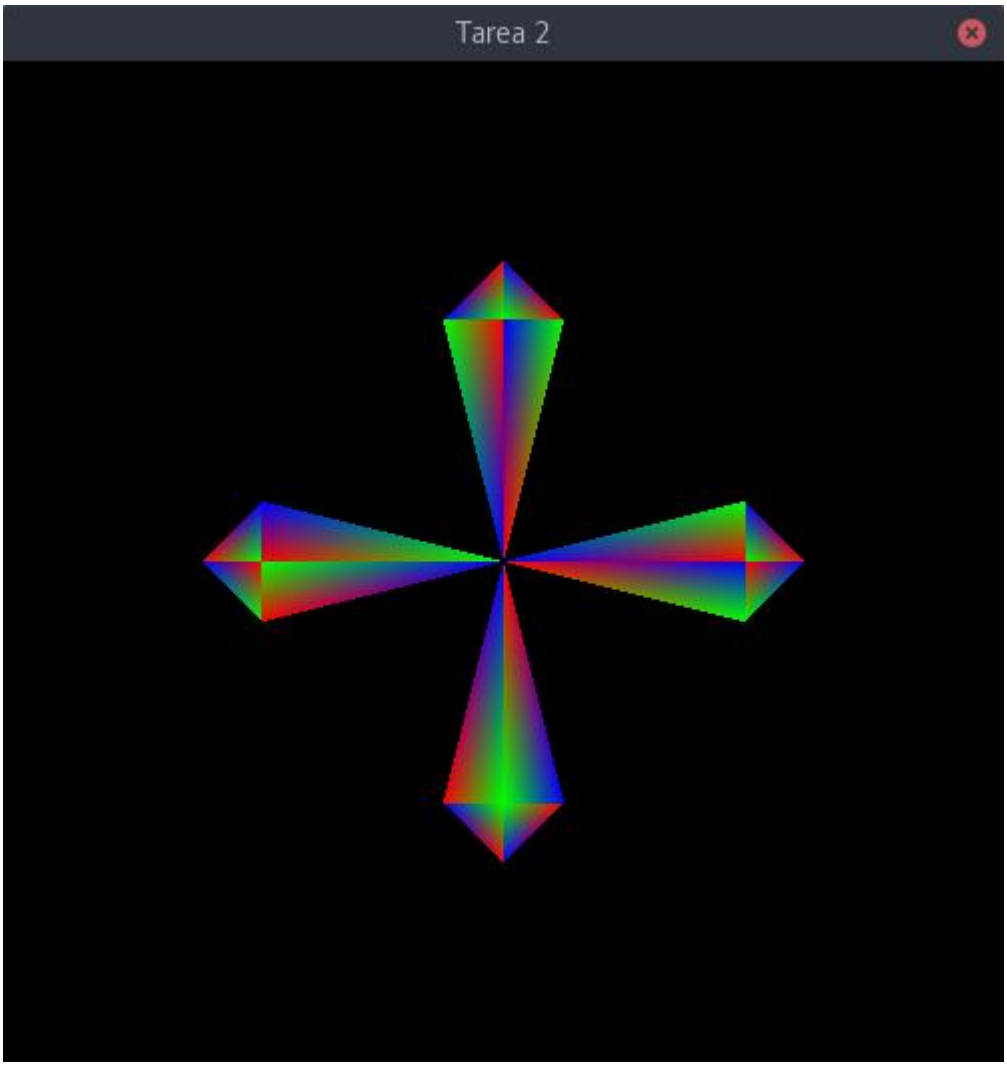
Computación gráfica

Grupo: 05

Cabrera López Oscar Emilio

Realizar un programa que dibuje la siguiente geometría, utilizar triángulos para dibujar la forma.





```
void dibujaEjercicio1() {
    // 1
    glBegin(GL_TRIANGLES);
    glColor3f(0.0f, 0.0f, 1.0f);
    glVertex3f(0.0f, 0.0f, 0.0f);
    glColor3f(0.0f, 1.0f, 0.0f);
    glVertex3f(-1.0f, 4.0f, 0.0f);
    glColor3f(1.0f, 0.0f, 0.0f);
    glVertex3f(0.0f, 4.0f, 0.0f);
    glEnd();

    glBegin(GL_TRIANGLES);
    glColor3f(1.0f, 0.0f, 0.0f);
    glVertex3f(0.0f, 0.0f, 0.0f);
    glColor3f(0.0f, 1.0f, 0.0f);
    glVertex3f(1.0f, 4.0f, 0.0f);
    glColor3f(0.0f, 0.0f, 1.0f);
    glVertex3f(0.0f, 4.0f, 0.0f);
    glEnd();

    // 2
    glBegin(GL_TRIANGLES);
    glColor3f(0.0f, 0.0f, 1.0f);
    glVertex3f(-1.0f, 4.0f, 0.0f);
    glColor3f(0.0f, 1.0f, 0.0f);
    glVertex3f(0.0f, 4.0f, 0.0f);
    glColor3f(1.0f, 0.0f, 0.0f);
    glVertex3f(0.0f, 5.0f, 0.0f);
    glEnd();

    glBegin(GL_TRIANGLES);
    glColor3f(1.0f, 0.0f, 0.0f);
    glVertex3f(1.0f, 4.0f, 0.0f);
    glColor3f(0.0f, 1.0f, 0.0f);
    glVertex3f(0.0f, 4.0f, 0.0f);
    glColor3f(0.0f, 0.0f, 1.0f);
    glVertex3f(0.0f, 5.0f, 0.0f);
    glEnd();

    // 3
    glBegin(GL_TRIANGLES);
    glColor3f(0.0f, 0.0f, 1.0f);
    glVertex3f(0.0f, 0.0f, 0.0f);
    glColor3f(0.0f, 1.0f, 0.0f);
```

```
    glVertex3f(0.0f, -4.0f, 0.0f);
    glColor3f(1.0f, 0.0f, 0.0f);
    glVertex3f(-1.0f, -4.0f, 0.0f);
    glEnd();

    glBegin(GL_TRIANGLES);
    glColor3f(1.0f, 0.0f, 0.0f);
    glVertex3f(0.0f, 0.0f, 0.0f);
    glColor3f(0.0f, 1.0f, 0.0f);
    glVertex3f(0.0f, -4.0f, 0.0f);
    glColor3f(0.0f, 0.0f, 1.0f);
    glVertex3f(1.0f, -4.0f, 0.0f);
    glEnd();

    // 4
    glBegin(GL_TRIANGLES);
    glColor3f(0.0f, 0.0f, 1.0f);
    glVertex3f(-1.0f, -4.0f, 0.0f);
    glColor3f(0.0f, 1.0f, 0.0f);
    glVertex3f(0.0f, -4.0f, 0.0f);
    glColor3f(1.0f, 0.0f, 0.0f);
    glVertex3f(0.0f, -5.0f, 0.0f);
    glEnd();

    glBegin(GL_TRIANGLES);
    glColor3f(1.0f, 0.0f, 0.0f);
    glVertex3f(1.0f, -4.0f, 0.0f);
    glColor3f(0.0f, 1.0f, 0.0f);
    glVertex3f(0.0f, -4.0f, 0.0f);
    glColor3f(0.0f, 0.0f, 1.0f);
    glVertex3f(0.0f, -5.0f, 0.0f);
    glEnd();

    // 5
    glBegin(GL_TRIANGLES);
    glColor3f(0.0f, 0.0f, 1.0f);
    glVertex3f(0.0f, 0.0f, 0.0f);
    glColor3f(0.0f, 1.0f, 0.0f);
    glVertex3f(-4.0f, 0.0f, 0.0f);
    glColor3f(1.0f, 0.0f, 0.0f);
    glVertex3f(-4.0f, -1.0f, 0.0f);
    glEnd();

    glBegin(GL_TRIANGLES);
```

```
    glColor3f(0.0f, 1.0f, 0.0f);
    glVertex3f(0.0f, 0.0f, 0.0f);
    glColor3f(1.0f, 0.0f, 0.0f);
    glVertex3f(-4.0f, 0.0f, 0.0f);
    glColor3f(0.0f, 0.0f, 1.0f);
    glVertex3f(-4.0f, 1.0f, 0.0f);
    glEnd();

    // 6
    glBegin(GL_TRIANGLES);
    glColor3f(0.0f, 0.0f, 1.0f);
    glVertex3f(-4.0f, 1.0f, 0.0f);
    glColor3f(0.0f, 1.0f, 0.0f);
    glVertex3f(-4.0f, 0.0f, 0.0f);
    glColor3f(1.0f, 0.0f, 0.0f);
    glVertex3f(-5.0f, 0.0f, 0.0f);
    glEnd();

    glBegin(GL_TRIANGLES);
    glColor3f(0.0f, 1.0f, 0.0f);
    glVertex3f(-4.0f, -1.0f, 0.0f);
    glColor3f(1.0f, 0.0f, 0.0f);
    glVertex3f(-4.0f, 0.0f, 0.0f);
    glColor3f(0.0f, 0.0f, 1.0f);
    glVertex3f(-5.0f, 0.0f, 0.0f);
    glEnd();

    // 7
    glBegin(GL_TRIANGLES);
    glColor3f(0.0f, 0.0f, 1.0f);
    glVertex3f(0.0f, 0.0f, 0.0f);
    glColor3f(0.0f, 1.0f, 0.0f);
    glVertex3f(4.0f, 1.0f, 0.0f);
    glColor3f(1.0f, 0.0f, 0.0f);
    glVertex3f(4.0f, 0.0f, 0.0f);
    glEnd();

    glBegin(GL_TRIANGLES);
    glColor3f(0.0f, 1.0f, 0.0f);
    glVertex3f(4.0f, -1.0f, 0.0f);
    glColor3f(1.0f, 0.0f, 0.0f);
    glVertex3f(0.0f, 0.0f, 0.0f);
    glColor3f(0.0f, 0.0f, 1.0f);
    glVertex3f(4.0f, 0.0f, 0.0f);
```

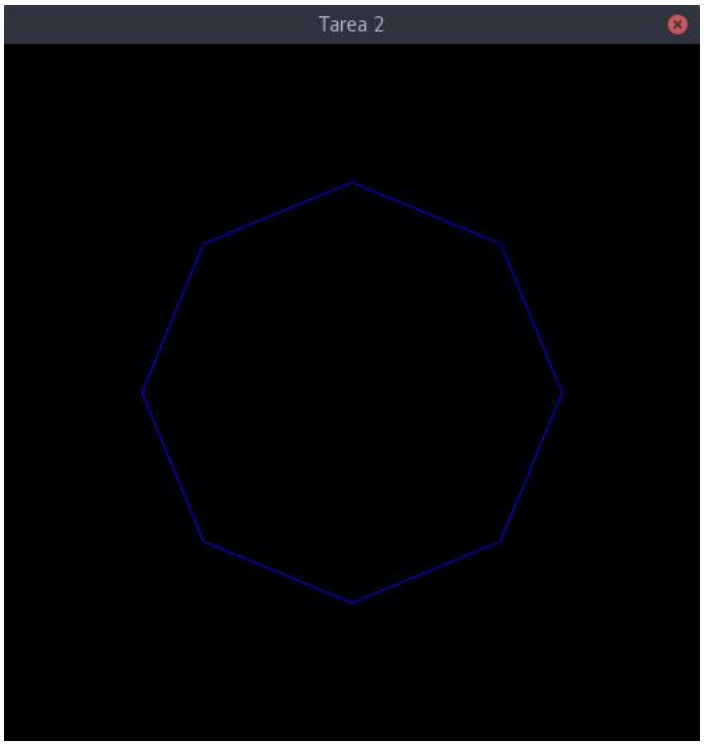
```
    glEnd();

    // 8
    glBegin(GL_TRIANGLES);
    glColor3f(0.0f, 0.0f, 1.0f);
    glVertex3f(4.0f, 1.0f, 0.0f);
    glColor3f(0.0f, 1.0f, 0.0f);
    glVertex3f(4.0f, 0.0f, 0.0f);
    glColor3f(1.0f, 0.0f, 0.0f);
    glVertex3f(5.0f, 0.0f, 0.0f);
    glEnd();

    glBegin(GL_TRIANGLES);
    glColor3f(0.0f, 1.0f, 0.0f);
    glVertex3f(4.0f, -1.0f, 0.0f);
    glColor3f(1.0f, 0.0f, 0.0f);
    glVertex3f(4.0f, 0.0f, 0.0f);
    glColor3f(0.0f, 0.0f, 1.0f);
    glVertex3f(5.0f, 0.0f, 0.0f);
    glEnd();

}
```

Realizar un programa que se dibuja un octógono, utilizar la primitiva GL_LINE_LOOP de tal forma que no se rellene y únicamente se renderice el contorno.



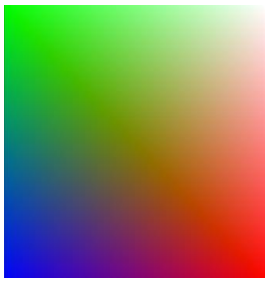
```
void dibujaPoligono(int lados, float radio) {
    double ang = 2 * M_PI / lados;
    glBegin(GL_LINE_LOOP);
    for (int i = 1; i <= lados;  i++) {
        glVertex3d(radio * cos(ang * (i - 1)), radio * sin(ang * (i - 1)), 0.0);
        glVertex3d(radio * cos(ang * i), radio * sin(ang * i), 0.0);
    }
    glEnd();
}
```

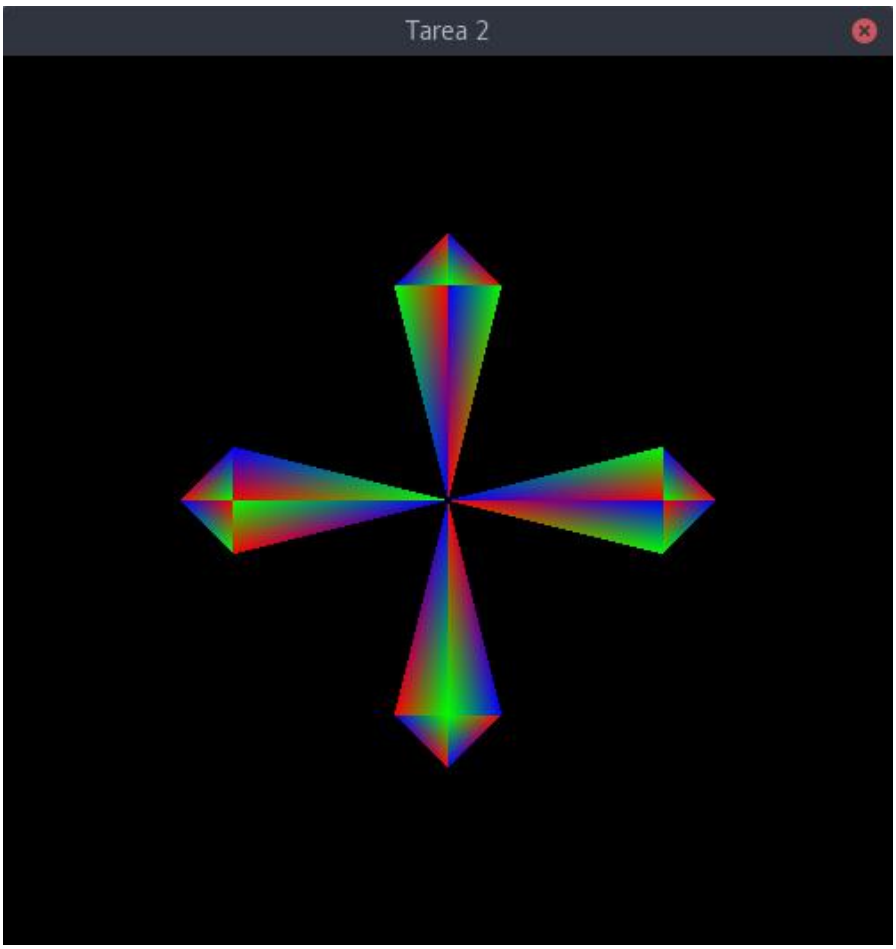
Con ayuda del ejemplo 04-Teclado, realizar un programa que cuando se apriete la tecla E, solo se dibuje el ejercicio 1, si se aprieta la tecla P, únicamente se dibuje el polígono del ejercicio anterior.

```
void keyboard(unsigned char key, int x, int y) {
    switch (key) {
        case 'e':
        case 'E':
            ejercicio = 1;
            break;
        case 'p':
        case 'P':
            ejercicio = 2;
            break;
        case 27:
            exit(0);
            break;
        default:
            break;
    }
}
```

```
void display() {
    glClear(GL_COLOR_BUFFER_BIT);
    glClearColor(0.0f, 0.0f, 0.0f, 0.0f);
    glMatrixMode(GL_MODELVIEW_MATRIX);
    glLoadIdentity();
    glTranslatef(0.0f, 0.0f, -20.0f);
    if (ejercicio == 1) {
        dibujaEjercicio1();
    } else if (ejercicio == 2) {
        dibujaPoligono(8, 5);
    }
    glFlush ();
}
```

Para el ejercicio 1, colocar un color a cada vértice del octágono, de tal forma que se visualice como a continuación se muestra.





En base al ejemplo 05-Mouse. Diga la diferencia que existe entre glutPassiveMotionFunc, glutMouseFunc y glutMotionFunc.

GlutPassiveMotionFunc: Muestra la posición del mouse en la ventana, cuando ningún botón del mouse es presionado.
GlutMouseFunc: Muestra la posición del cursor cuando se presiona un botón, el botón que se presiona, y el estado, si es presionado o liberado.
GlutMotionFunc: Muestra la posición del cursor mientras algún boton esta presionado.