Construções de Computação Gráfica - 1ª Entrega

Nome: Matheus Pimentel Cardoso – 1820786BCC

Construção 1

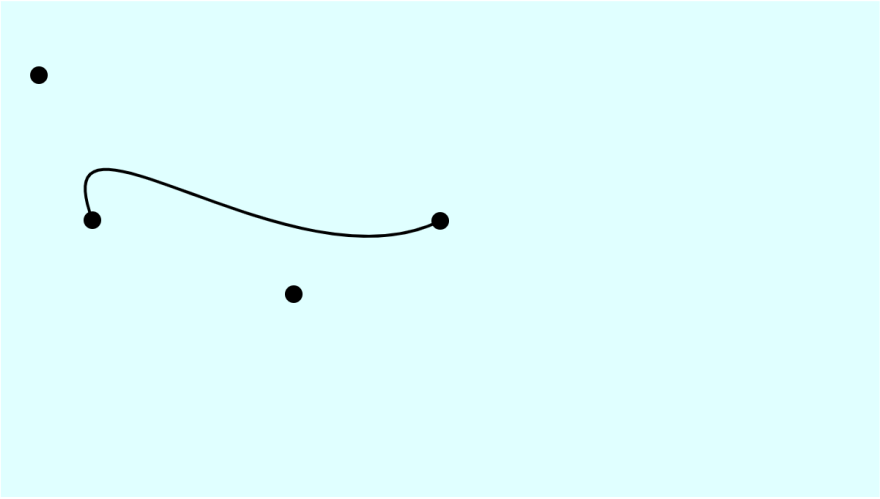
Link para o GITHUB: https://github.com/MatheusPCardoso/Computacao-Grafica

Código Principal:

```
function draw() {
  background(224, 255, 255); //fundo

  if (mouseIsPressed) {//verifica se o mouse está pressionado
    if (Math.abs(fDot.x - mouseX) <= 14 && Math.abs(fDot.y - mouseY) <= 14) {//movendo o primeiro ponto
      fDot.x = mouseX
      fDot.y = mouseY
    }
    else if (Math.abs(sDot.x - mouseX) <= 14 && Math.abs(sDot.y - mouseY) <= 14) {//movendo o segundo ponto
      sDot.x = mouseX
      sDot.y = mouseY
    }
    else if (Math.abs(tDot.x - mouseX) <= 14 && Math.abs(tDot.y - mouseY) <= 14) {//movendo o terceiro ponto
      tDot.x = mouseX
      tDot.y = mouseY
    }
    else if (Math.abs(foDot.x - mouseX) <= 14 && Math.abs(foDot.y - mouseY) <= 14) { //movendo o quarto ponto
      foDot.x = mouseX
      foDot.y = mouseY
    }
  }
```

Print da tela:



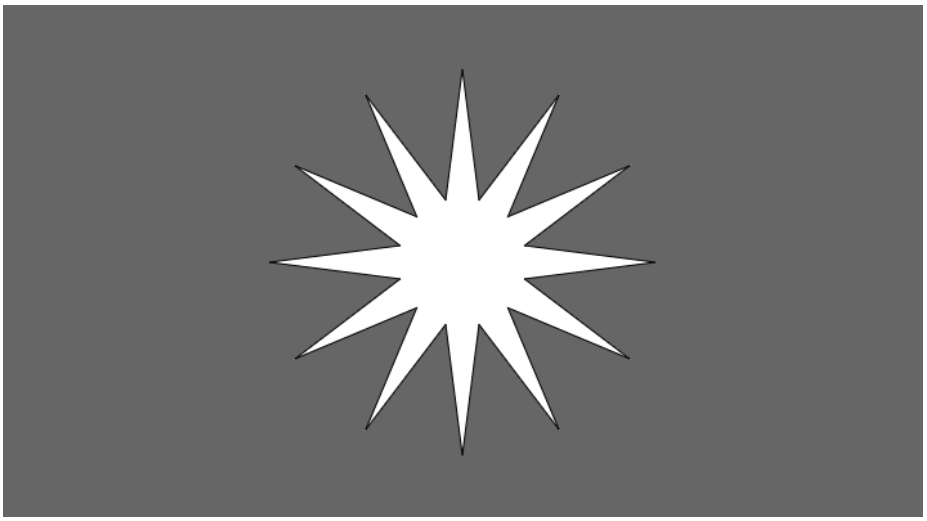
Construção 2

Link para o GITHUB: https://github.com/MatheusPCardoso/Computacao-Grafica

Código Principal:

```
function star(x, y, r1, r2, pontos) {
    let angulo = TWO_PI / pontos;
    let meioAngulo = angulo / 2.0;
    beginShape();
    for (let a = 0; a < TWO_PI; a += angulo) {
        let sx = x + cos(a) * r2;
        let sy = y + sin(a) * r2;
        vertex(sx, sy);
        sx = x + cos(a + meioAngulo) * r1;
        sy = y + sin(a + meioAngulo) * r1;
        vertex(sx, sy);
    }
    endShape(CLOSE);
```

Print da tela:



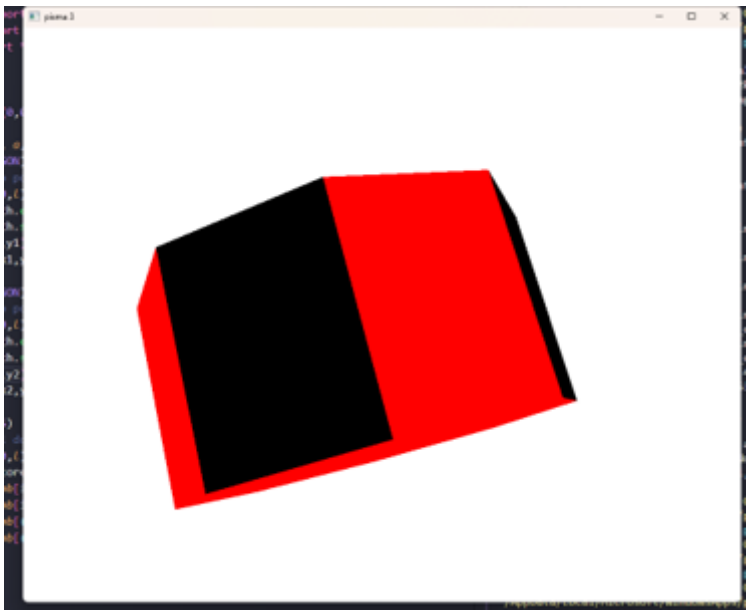
Construção 3

Link para o GITHUB: https://github.com/MatheusPCardoso/Computacao-Grafica

Código Principal:

```
def calcPrisma(r, l, a, pb, pt, an):
    glBegin(GL_POLYGON)
    #Retorna base do poligono
    for i in range(0,l):
        x1 = r * math.cos(i*an)
        y1 = r * math.sin(i*an)
        pb += [ (x1,y1) ]
        glVertex3f(x1,y1,0.0)
    glEnd()
    glBegin(GL_POLYGON)
    #Retorna topo do poligono
    for i in range(0,l):
        x2 = r * math.cos(i*an)
        y2 = r * math.sin(i*an)
        pt += [ (x2,y2) ]
        glVertex3f(x2,y2,a)
    glEnd()
    glBegin(GL_QUADS)
    #Retorna lateral do poligono
    for i in range(0,l):
        glColor3fv(cores[(i+1)%len(cores)])
        glVertex3f(pb[i][0],pb[i][1],a)
        glVertex3f(pb[i][0],pb[i][1],0.0)
        glVertex3f(pb[(i+1)%l][0],pb[(i+1)%l][1],0.0)
        glVertex3f(pb[(i+1)%l][0],pb[(i+1)%l][1],a)
    glEnd()
```

Print da tela:



Construção 3.1

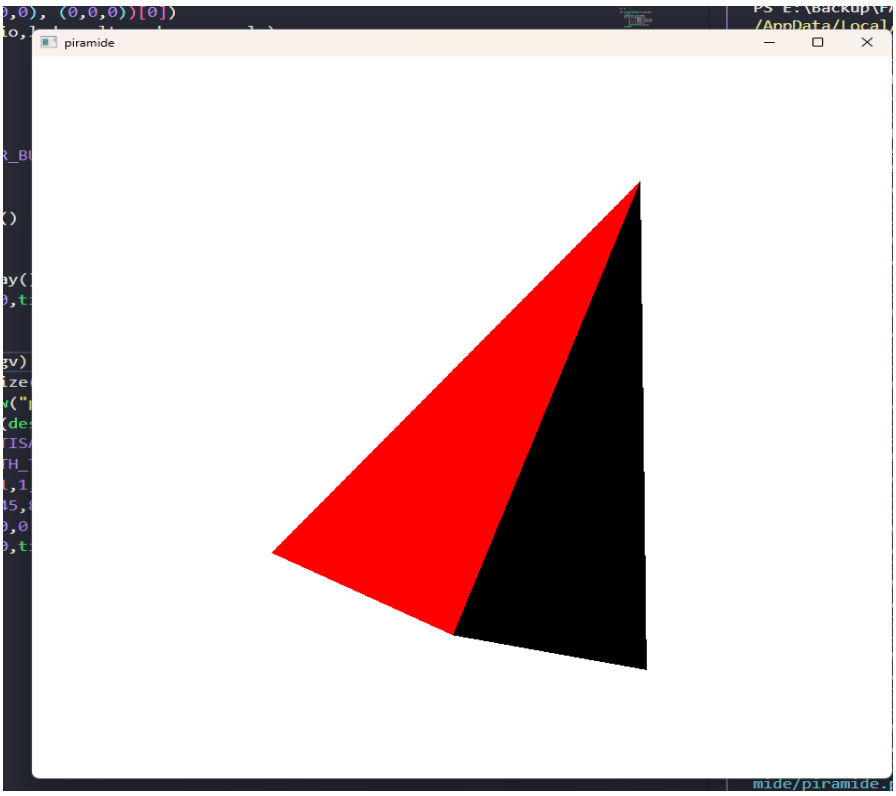
Link para o GITHUB: https://github.com/MatheusPCardoso/Computacao-Grafica

Código Principal:

```
def calcPiramide(r,l,a,pb,an):
    # BASE
    glBegin(GL_POLYGON)
    for i in range(0,l):
        x = r * math.cos(i*an)
        y = r * math.sin(i*an)
        pb += [ (x,y) ]
        glVertex3f(x,y,0.0)
    glEnd()

    # LATERAL
    glBegin(GL_TRIANGLES)
    for i in range(0,l):
        glColor3fv(((1,0,0), (0,0,0))[(i+1)%len(((1,0,0), (0,0,0)))])
        glVertex3f(0.0,0.0,a)
        glVertex3f(pb[i][0],pb[i][1],0.0)
        glVertex3f(pb[(i+1)%l][0],pb[(i+1)%l][1],0.0)
    glEnd()
```

Print da tela:



Construção 4

Link para o GITHUB: https://github.com/MatheusPCardoso/Computacao-Grafica

Código Principal:

```
# funcao responsavel pelo calculo do plot
def return_L(i,j):
    k = (math.pi*i/(n1-1))-(math.pi/2)
    p = 2*math.pi*j/(n2-1)
    x = r*math.cos(k)*math.cos(p)
    y = r*math.sin(k)
    z = r*math.cos(k)*math.sin(p)
    s = p/(2*math.pi)
    t = (k + math.pi/2)/(math.pi)
    return x,y,z,s,t

# funcao que gera a figura
def figure():
    glPushMatrix()
    glRotatef(local_x_start, 1.0, 1.0, 0.0)
    glRotatef(local_y_start, 0.0, 0.0, 1.0)
    glBindTexture(GL_TEXTURE_2D, texture[0])
    glBegin(GL_QUAD_STRIP)
    for i in range(0,n1):
        for j in range(0,n2):
            x,y,z,s,t = return_L(i,j)
            glTexCoord2f(s,t)
            glVertex3f(x,y,z)
            x,y,z,s,t = return_L(i+1,j)
            glTexCoord2f(s,t)
            glVertex3f(x,y,z)
    glEnd()
    glPopMatrix()
```

Print da tela:

Construção 5

Link para o GITHUB: https://github.com/MatheusPCardoso/Computacao-Grafica

Código Principal:

```
# funcao responsavel pelo calculo do plot
def return_L(i,j):
    k = (math.pi*i/(n1-1))-(math.pi/2)
    p = 2*math.pi*j/(n2-1)
    x = r*math.cos(k)*math.cos(p)
    y = r*math.sin(k)
    z = r*math.cos(k)*math.sin(p)
    return x,y*y,z

# funcao que gera a figura
def figure():
    glPushMatrix()
    glRotatef(local_start,1.0,0.0,0.0)
    glBegin(GL_QUAD_STRIP)
    for i in range(0,n1):
        for j in range(0,n2):
            x,y,z = return_L(i,j)
            glVertex3f(x,y,z)
            x,y,z = return_L(i+1,j)
            glVertex3f(x,y,z)
    glEnd()
    glPopMatrix()
```

Print da tela: