UNIVERSIDADE ABERTA
www.uab.pt

Ministério da Ciência Tecnologia e Ensino Superior

p-Fólio

U.C. 21020 – Licenciatura em Engenharia Informática

Computação Gráfica

5 de fevereiro de 2019

-- INSTRUÇÕES --

- O estudante deverá responder à prova na folha de ponto e preencher o cabeçalho e todos os espaços reservados à sua identificação, com letra legível.
- No caso de provas com escolha múltipla, **sem grelha de resposta**, deverá indicar a resposta correcta na folha de ponto, indicando o número da pergunta e a resposta que considera correcta.
- No caso de provas com escolha múltipla, **com grelha de resposta, tabela e/ou espaços para preenchimento**, deverá efectuar as respostas no enunciado, pelo que o mesmo deverá ser entregue ao vigilante, juntamente com a folha de ponto, **não sendo permitido ao estudante levar o enunciado.**
- Verifique no momento da entrega da(s) folha(s) de ponto se todas as páginas estão rubricadas pelo vigilante. Caso necessite de mais do que uma folha de ponto, deverá numerá-las no canto superior direito.
- Em hipótese alguma serão aceites folhas de ponto dobradas ou danificadas.
- Exclui-se, para efeitos de classificação, toda e qualquer resposta apresentada em folhas de rascunho.
- Os telemóveis deverão ser desligados durante toda a prova e os objetos pessoais deixados em local próprio da sala de exame.
- A prova é constituída por **2** páginas e termina com a palavra **FIM**. Verifique o seu exemplar e, caso encontre alguma anomalia, dirija-se ao professor vigilante nos primeiros 15 minutos da mesma, pois qualquer reclamação sobre defeito(s) de formatação e/ou de impressão que dificultem a leitura não será aceite depois deste período.
- Utilize unicamente tinta azul ou preta.
- Responda às questões de forma clara, sucinta, e apresente todos os cálculos.
- Quando solicitado, apresente ainda uma representação gráfica do resultado final obtido na questão.
- A cotação de cada uma das questões é indicada junto do enunciado da mesma.
- A prova é **SEM CONSULTA**. Todos os elementos necessários à resolução são fornecidos no enunciado.

Duração: 90 minutos

QUESTÃO 1 (3 valores)

Recorra ao algoritmo *Bresenham* para calcular as coordenadas de todos os *pixels* que representam a reta entre ***A***(20, 10) e ***B***(30, 18). Elabore uma tabela que mostre os ciclos interativos do início ao final do cálculo. Mostre graficamente o resultado final (grelha de pontos).

QUESTÃO 2 (3 valores)

Calcule com o algoritmo do **Ponto Médio** as coordenadas de todos os *pixels* necessários para representar uma circunferência com os seguintes valores: Raio (r) = 10 e Centro = (3, 4). Elabore uma tabela com todas as etapas de interação e respetivos resultados. Mostre graficamente o resultado.

QUESTÃO 3 (6 valores)

Codifique em JOGL um programa que desenhe círculos de cores e tamanhos diferentes com projeção ortográfica. Na apresentação do programa deverá minimamente:

- Especificar a funções *init*(), *display*() e *reshape*(), definindo todas as linhas de código necessárias para definição de janela-visor, projeção espacial, configuração da cor, entre outras, que julgue serem necessárias para conseguir que o programa execute o solicitado;
- Identificar as bibliotecas (*import*) necessárias para o código compilar;
- Especificar a função *main*(), definindo os comandos necessários para a instanciação do objeto e criação da visualização;
- Comentar o código, explicando o que ele faz em cada linha;

```
import java.awt.*;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;
import javax.media.opengl.*;
import javax.media.opengl.glu.GLU;

import java.util.Random;
import com.sun.opengl.util.Animator;
import com.sun.opengl.util.GLUT;

@SuppressWarnings("serial")
public class CirculosVariantes extends Frame implements GLEventListener {
 static int HEIGHT = 800, WIDTH = 800;
 static GL gl; // interface para o OpenGL
 static GLUT glut = new GLUT(); // interface para a GLUT
 static GLCanvas canvas; // uma frame desenhável
 static GLCapabilities capabilities;
 Random rand;
 static Animator animator;

 public CirculosVariantes() {
  capabilities = new GLCapabilities();
  canvas = new GLCanvas();
  canvas.addGLEventListener(this);
  add(canvas, BorderLayout.CENTER);
  gl = canvas.getGL();
  rand = new Random();
  addWindowListener(new WindowAdapter() {
   public void windowClosing(WindowEvent e) {
    animator.stop();
    System.exit(0);
   }
  });
 }

 public void init(GLAutoDrawable drawable) {
  animator = new Animator(canvas);
  animator.start(); // thread iniciado
 }

 public void reshape(GLAutoDrawable drawable, int x, int y, int width,int
height) {
  WIDTH = width;
  HEIGHT = height;
  gl.glClearColor(0.0f, 0.5f, 0.5f, 0.5f);
  gl.glMatrixMode(GL.GL_PROJECTION);
  gl.glLoadIdentity();
  gl.glOrtho(0, width, 0, height, -1.0, 1.0);
  gl.glMatrixMode(GL.GL_MODELVIEW);
  gl.glLoadIdentity();
  gl.glViewport(0, 0, width, height);
 }

 void drawCircle () {
  double r;
  r = rand.nextDouble()*100f;
  glut.glutWireSphere(r, 50, 50);
 }

 public void display(GLAutoDrawable drawable) {
  gl.glClear (GL.GL_COLOR_BUFFER_BIT|GL.GL_DEPTH_BUFFER_BIT);
```

```
   gl.glLoadIdentity();
   gl.glTranslated(100.0f, 150.0f, 0.0f);
   gl.glPushMatrix();
    for (int i = 0; i <=30; ++i)
    {
     gl.glColor3f(rand.nextFloat(),rand.nextFloat(),rand.nextFloat());
     gl.glTranslated(rand.nextFloat()*200.0f, rand.nextFloat()*300.0f,
0.0f);
     drawCircle();
    }
   gl.glPopMatrix();
   // torna refrescamento de desenho mais lento
   try {
    Thread.sleep(200);
   } catch (Exception ignore) {
    }
 }

 public static void main(String[] args) {
  CirculosVariantes pfolio = new CirculosVariantes();
  pfolio.setTitle("Pfólio");
  pfolio.setSize(WIDTH, HEIGHT);
  pfolio.setVisible(true);
 }

}
```

FIM