PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
Instituto de Ciências Exatas e Informática
Computação Gráfica - Profa.: Rosilane Mota

Prova 01 (Valor: 30 pontos)

Nome: Rafael Amaury Diniz Augusto - 651047

Essa é uma atividade avaliativa individual, sem consulta à internet, celular ou à terceiros, que aborda os conteúdos apresentados na disciplina de Computação Gráfica até Preenchimento (inclusive). As respostas devem ser coerentes com o conteúdo apresentado em sala de aula. **Qualquer conteúdo divergente daquele apresentado em sala de aula será desconsiderado.**

Todas as anotações, memórias de cálculo, rascunhos podem ser entregues com a solução para cada questão. Com acesso a essas informações, a avaliação da questão pode considerar parte do raciocínio apresentado. Essas informações extras devem conter apenas o conteúdo pertinente à questão.

Considere essa avaliação como uma forma de você verificar se o seu processo de assimilação do conteúdo ocorreu sem nenhuma pendência em termos de dúvidas. Em caso de constatar que ainda existe dúvida, você terá a oportunidade de esclarecê-las com o professor. Lembre-se de que essa ação é importante para assimilar conteúdos futuros que dependem do conhecimento avaliado.

Preste atenção ao tempo programado para a realização da prova! Você terá até o final da aula para finalizar todas as questões. No caso das questões dissertativas (abertas), a resposta deverá ser preenchida de modo legível, ou será desconsiderada. No caso de lápis, não será aceita revisão das correções. As questões em branco serão indicadas na própria prova.

Desejo a vocês uma boa avaliação!

Atenciosamente,

Profa. Rosilane Mota

QUESTÃO 01 (05 PONTOS)

As equações que fundamentam as transformações geométricas 2D e 3D foram adaptadas durante a evolução do conteúdo apresentado em sala para uma forma geral que viabilizasse otimizações. Suponha uma transformação 3D que atenda ao seguinte sistema linear:

$$\boxed{\begin{aligned} x &= ax + 2ay + 4az \\ y &= 4ax + ay + 2az \\ z &= 2ax + 4ay + az \\ h &= 1 \end{aligned}}$$

a) (2 pontos) Como essa nova transformação pode ser representada matricialmente? Indique a equação matricial.

b) (3 pontos) Como aplicar em uma única equação essa nova transformação e uma translação, nessa ordem? Indique a equação envolvendo as duas transformações.

Questão 02 (06 pontos)

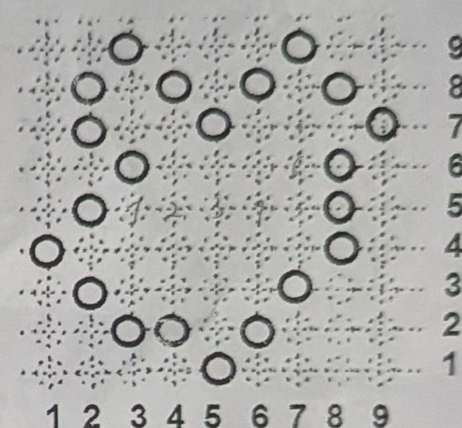
O algoritmo de rasterização de circunferências de Bresenham não calcula toda a primitiva gráfica. Explique como é tratada na implementação do algoritmo a generalização para quaisquer posições que esteja a primitiva gráfica, assim como para que toda a primitiva seja exibida.

Questão 03 (4 pontos)

O algoritmo de *Sutherland-Hodgeman* (recorte de polígonos) estabelece algumas restrições para atender aos objetivos do algoritmo. Cite cada objetivo e como cada um deles é implementado no algoritmo.

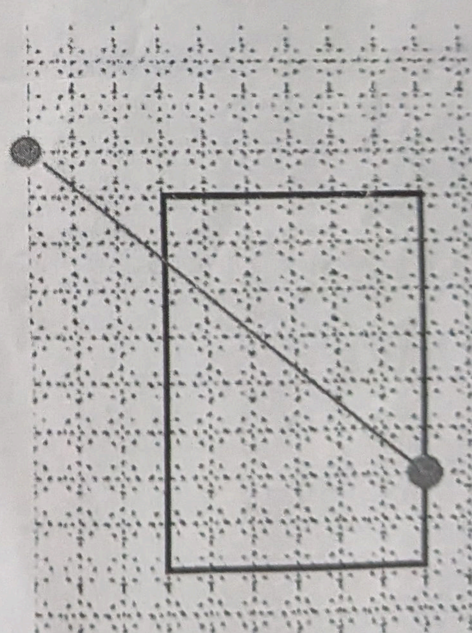
Questão 04 (5 pontos)

Para a imagem a seguir da área a ser preenchida pelo algoritmo de *Boundary com conectividade 4*, indique a ordem de preenchimento a partir do ponto inicial (3,5). Mostre a ordem utilizada de análise dos vizinhos.



Questão 05 (10 pontos)

Para a imagem ao lado, considere os limites do retângulo como (xmin, ymin) = (0,0) e (xmax, ymax) = (6,8) e os pontos que definem a reta como o ponto inicial (-3,9) e o final (6, 2).



Aplique e indique os valores das variáveis em cada iteração para:

c) **(5 pontos)** Algoritmo de *Liang-Barsky (paramétrica)* para a reta;
d) **(5 pontos)** Algoritmo de *DDA* para a reta.

Mostre os valores das variáveis para cada um dos algoritmos a cada iteração.

25.5

Rafael Amaury Diniz Augusto - 651047

(3) São 4 restrições: → Polígono fechado e mesmo sentido/ordem dos vértices

2

* A → B, onde A e B são ambos vértices dentro da área de recorte = Salva na lista o ponto B.

* A → B, onde A e B são vértices fora da área de recorte = ignora os dois vértices.

* A → B, onde A está dentro e B está fora = acha interseção de B com o recorte (B') e salva B'.

* A → B, onde A está fora e B está dentro = acha interseção para A com o recorte (A') e salva A' e salva B.

(1) A) $\begin{bmatrix} a & 2a & 4a & 0 \\ 4a & a & 2a & 0 \\ 2a & 4a & a & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ Eq?

B) $\begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} a & 2a & 4a & 0 \\ 4a & a & 2a & 0 \\ 2a & 4a & a & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

4

$= \begin{bmatrix} a & 2a & 4a & t_x \\ 4a & a & 2a & t_y \\ 2a & 4a & a & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$ Eq?

4) boundary fill com ordem de preenchimento direita → cima → esquerda → baixo.

Ponto inicial = (3, 5)

Pseudo código:

```
func boundaryFill(ponto, cor, corBorda) {
    ponto.cor = cor
    // Verificando se a cor dos vizinhos é diferente de corBorda e de cor
    if ponto.direita.cor != [corBorda, cor]
        boundaryFill(ponto.direita, cor, corBorda)
    if ponto.cima.cor != [corBorda, cor]
        boundaryFill(ponto.cima, cor, corBorda)
    if ponto.esquerda.cor != [corBorda, cor]
        boundaryFill(ponto.esquerda, cor, corBorda)
    if ponto.baixo.cor != [corBorda, cor]
        boundaryFill(ponto.baixo, cor, corBorda)
    return
}
```

Ordem de preenchimento com ponto inicial (3,5):

5

(3,5), (4,5), (5,5), (6,5), (7,5), (7,6), (7,7), (8,7),
(7,8), (7,6), (6,6), (5,6), (4,6), (4,7), (3,7), (3,8),
(7,4), (6,4), (5,4), (4,4), (3,4), (2,4), (3,3), (4,3),
(5,3), (6,3), (5,2)

⑤ A) $X_{MIN}, Y_{MIN} = (0,0)$
$X_{MAX}, Y_{MAX} = (6,8)$

$P_{inicial} = (-3,9)$
$P_{final} = (6,2)$

$dX = 9$
$dY = -7$
$U1 = 0 \quad U2 = 1$

se YYY(-9, -3, 0, 1) → Retorna TRUE!
{
$r = 1/3$
$U1 = 1/3$
}
5 se YYY(9, 9, 1/3, 1) { → Retorna TRUE!
$r = 1$
}
se YYY(7, 9 1/3, 1) { → Retorna TRUE!
$r = 9/7$
}
se YYY(-7, -1, 1/3, 1) { → Retorna True!
$r = -1/-7 = 1/7$
}

U1 > 0, logo...
$X1 = -3 + 1/3 * 9 = -3 + 3 = 0$
$Y1 = 9 + 1/3 * -7 = 9 - 7/3 = 27/3 - 7/3 = 20/3 = 6+2/$

b) P inicial = (-3, 9)
P final = (6, 2)

YYY(-3, 9, 6, 2)

dx = 9
dY = -7
passos = 9

X inc = 1
y inc = -7/9 = -0.777...

5

X = -3
Y = 9

no loop temos os valores seguintes a cada iteração:

X = {-2, -1, 0, 1, 2, 3, 4, 5, 6}
Y = {8.23, 7.46, 6.69, 5.92, 5.15, 4.38, 3.61, 2.84, 2.07}

4. 5

(1) Essa generalização é feita após o cálculo do primeiro ~~quadrante~~ octante do círculo! Como todos os outros pontos do círculo podem ser representados como reflexões em X, Y, ou X e Y do primeiro quadrante que já foi calculado, é possível generalizar e otimizar essa conta ao apenas derivar os outros pontos a partir desses que já foram calculados, ao invés de refazer todas as contas. Isso economiza muitos gastos computacionais e deixa o código mais eficiente!

E qdo centro não for origem?