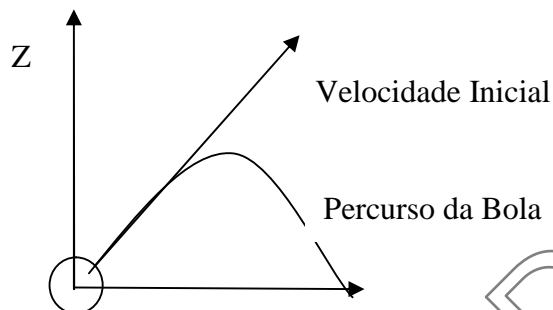


Engenharia Multimédia Curso de Informática	2º Ano - 1º Semestre
Técnicas de Animação Gráfica Multimédia II	Exame
Data : 25/7/2007	Duração : 60 Minutos
Parte Teórica	Prof. : Jorge Mota

Numero : _____ Nome : _____

Pergunta 1

(3 valores) Implemente em Flash e ActionScript um programa que permita simular o disparo de uma “bala de canhão”, usando as seguintes formulas:



Notas para implementação:

Formulas para calculo do movimento da bola:

NT – “Normalized Time” – Varia no intervalo [0,1], se a animação dura 200 frames, então num dado momento $NT = \text{Frame_actual} / 200$.

$G = 9,8 \text{ m/s}^2$

$X = \text{velocidade} * \sin(\text{angulo}) * NT * 10$

$Y = 0$

$Z = \text{velocidade} * \cos(\text{angulo}) * NT * 10 - G (NT * 10)^2 / 2$

Ângulo, é a inclinação do vector velocidade inicial.

Considere que existe um movieClip chamado “bala” na biblioteca de símbolos.

Todos os elementos adicionais ficam ao arbítrio do aluno.

Pergunta 2

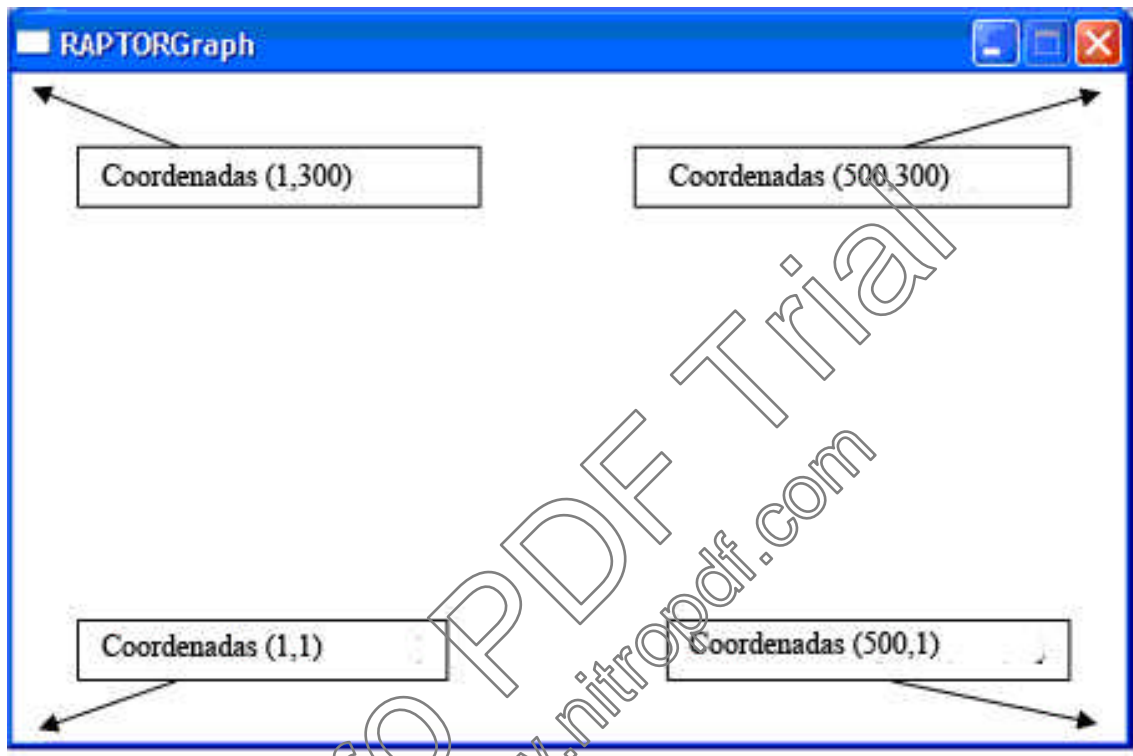
Elabore um pequeno programa em Raptor que permita fazer o traçado de um gráfico da função $\sin(x)$ no intervalo de [0;360] graus (3 Valores).

Notas para implementação:

Em raptor a biblioteca gráfica chama-se raptorGRAPH e é chamada por:

`Open_Graph_Window (X_Size, Y_Size)`

`Open_Graph_Window
(500, 300)`



Para fechar a janela usamos:

`Close_Graph_Window`

`Close_Graph_Window`

Funções básicas de desenho em raptorGRAPH:

<code>Put_Pixel</code>	Ilumina um pixel com a cor desejada
<code>Draw_Line</code>	Desenha uma linha entre dois pontos
<code>Clear_Window</code>	Limpa a janela gráfica

Exemplos de uso:

↓
Put_Pixel(20, 30,
Black) →

↓
Draw_Line(15, 99, 12,
13, Light_Blue) →

↓
Clear_Window(Black) →

Resposta:

Nesta implementação consideraram-se as seguintes variáveis e funções complementares:

Factor_escala → que representa o factor de escala para os valores a representar no eixo dos yy.

Origem_y → que representa a origem vertical para a representação do gráfico da função sin()

Converter_radiano → Função que permite converter em radianos os valores do intervalo [0,360]

É implementado um ciclo contado no intervalo pretendido em graus [0,360], que de cada vez calcula para cada valor de x o respectivo y e o majora do factor_escala pretendido e faz o seu offset para a posição origem_y.

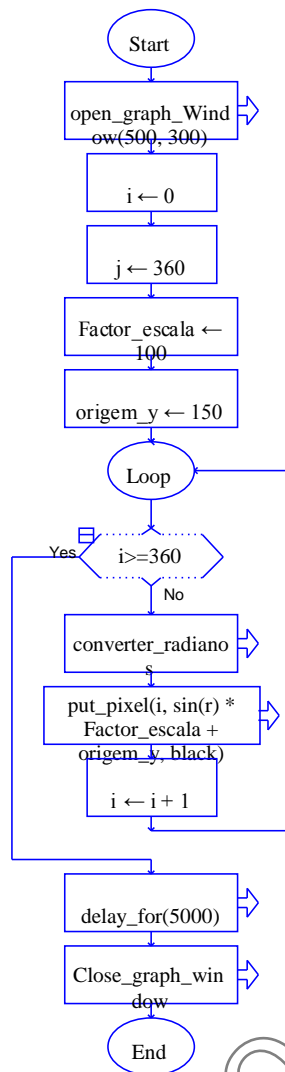
Esta implementação faz uso das funções de desenho em RaptorGRAPH:

Open_Graph

Close_Graph

Put_Pixel

Uma implemetação diferente poderia usar a função Draw_line em substituição da função Put_pixel.



Pergunta 3

Explique como converter uma imagem bitmap colorida usando um padrão 24 bits (RGB), numa imagem Bitonal (Preto e Branco). (3 Valores).

Resposta:

Para converter uma imagem bitmap colorida a 24 bits (RGB) numa imagem bitonal ou seja apenas a preto e branco, podemos proceder da seguinte forma:

-Criar um procedimento que leia pixel a pixel cada elemento da imagem bitmap e converta as suas componentes R, G e B em níveis de cinzento usando uma média pesada semelhante à usada no sistema televisivo NTSC/PAL para converter imagem cor em PB :

Valor do nível de cinzento para cada pixel= $R*0,2+G*0,6+B*0,2$

- Em seguida aplicar comparar o valor calculado com os seguintes intervalos [0,127] e [128,255]. Se o valor estiver no primeiro intervalo então colocamos o

valor de 0 em cada componente RGB se o valor estiver no segundo intervalo colocamos o valor 255 nas componentes 255.

A imagem assim obtida é bitonal possui apenas o Branco Branco ou o Preto preto.

Pergunta 4

Analise o seguinte código em ActionScript, que implementa o comportamento de uma bola, num campo, usando um vector velocidade representado pelas componentes xv e yv:

0: // Programação do Comportamento da Bola.

```
1: onClipEvent(load){
2:     //Criar as variáveis que representam o vector velocidade
3:     var xv=5;
4:     var yv=5;
5: }
6: //Este código é executado sempre que entra numa frame
7: onClipEvent(enterFrame){
8:     //Verificar colisões com os limites do campo
9:     if((this._y+this._height)>Stage.height){
10:         //Recolocar a bola no fim do campo
11:         this._y=Stage.height-this._height;
12:         //Activar movimento da Bola para cima
13:         yv=yv*(-1);
14:     }
15:     // Verificar colisões com os limites do campo
16:     if(this._y<0){
17:         //Colocar a bola no canto superior
18:         this._y=0;
19:         //Activar o movimento da bola para cima
20:         yv=yv*(-1);
21:     }
22:     //Verificar colisões com os limite lateral
23:     if((this._x+this._width)>Stage.width){
24:         //Recolocar a bola no fim do campo
25:         this._x=Stage.width-this._width;
26:         //Activar movimento da Bola para cima
27:         xv=xv*(-1);
28:     }
29:     //verificar se ultrapassa os limites laterais do campo
30:     if((this._x+this._width<0) || (this._x>Stage.width)){
31:         //Somar um ponto ao jogador
32:         if(this._x+this._width<0){_root.score++;}
33:         //Colocar a bola a meio campo
34:         this._x=Stage.width/2;
35:         //Alterar a direcção da bola
```

```

36:         xv=xv*(-1);
37:     }
38:     //Incrementar a posição da Bola
39:     _x=_x+xv;
40:     _y=_y+yv;
41: }
42: // Programação do Comportamento da Raquete
43: onClipEvent(enterFrame){
44:     // Se a tecla UP é premida move a raquete da esquerda para cima
45:     if(Key.isDown(Key.UP)){
46:         this._y= this._y -5;
47:     }
48:     // Se a tecla DOWN é premida move a raquete da esquerda para Baixo
49:     if(Key.isDown(Key.DOWN)){
50:         this._y= this._y +5;
51:     }
52:     //Se a Raquete ultrapassa a área do Campo, então é colocada
    novamente no 53: //fundo do campo
54:     if((this._y+this._height)>Stage.height){
55:         this._y=Stage.height-this._height;
56:     }
57:     //Se ultrapassa o campo parte superior então move para a parte superior
58:     if(this._y<0){
59:         this._y=0;
60:     }
61:     //Verificar a colisão da bola com a raquete
62:     if(this.hitTest(_root.Bola)){
63:         //Altera a direcção da bola
64:         _root.Bola.xv= _root.Bola.xv*(-1);
65:         //Move a bola para os limites da raquete
66:         _root.Bola._x=this._x+this._width;
67:     }
69: }

```

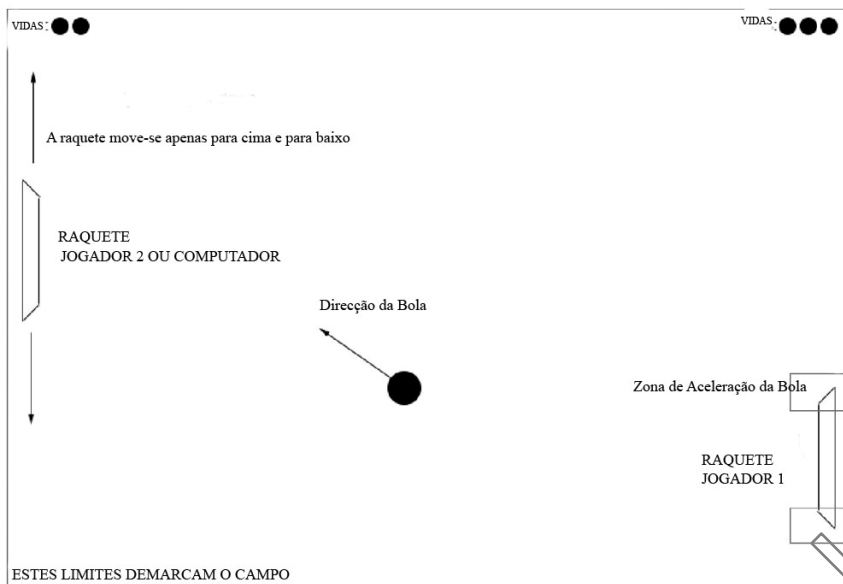
Implemente o efeito de Zona de Aceleração e Variação de direcção da bola nas extremidades da raquete conforme representado na figura.

Resposta:

```

61: //Verificar a colisão da bola com a raquete
62: if(this.hitTest(_root.Bola)){
    If ((_root.Bola._x>this._x-this._height/4) or (_root.Bola._x>this._x-
this._height/4*3)){
        //Altera a direcção da bola
        _root.Bola.xv= _root.Bola.xv*(-random()*3);
65: //Move a bola para os limites da raquete
66:     _root.Bola._x=this._x+this._width;
67: }

```



Notas para facilitar a implementação :

- Numere as linhas de código para melhor se perceber as ALTERAÇÕES QUE FIZER

Pergunta 5

Represente a matriz genérica de transformação, de **ROTAÇÃO em 2D** ($MTRX(\theta)$), de forma que esta permita por uma simples operação de transformação, rodar na posição actual o ponto $P(x,y,1)$ de um ângulo θ . (2 valores):

Resposta:

Matriz de transformação

$$\begin{bmatrix} 1 & 0 & d_x \\ 0 & 1 & d_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(a) & -\text{sen}(a) & 0 \\ \text{sen}(a) & \cos(a) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -d_x \\ 0 & 1 & -d_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos(a) & -\text{sen}(a) & d_x(1-\cos(a))+d_y\text{sen}(a) \\ \text{sen}(a) & \cos(a) & d_y(1-\cos(a))-d_x\text{sen}(a) \\ 0 & 0 & 1 \end{bmatrix} = T(d_x, d_y).R(a).I(-d_x, -d_y)$$

Em vez θ usamos α para representar o ângulo de rotação.

Pergunta 6

Relacione o conceito de animação em flash com o conceito de **easing in** e **easing out** da animação tradicional, explicando o que é isto de easing in e out em termos de animação de movimentos. Dê exemplos, e/ou use gráficos para ilustrar a sua resposta(2 valores)

Pergunta 7

Compare as características da API gráfica avançada como DirectX 9C e o ambiente gráfico do Flash/ActionScript_ em termos de aplicações gráficas no

domínio dos videojogos 2D. Enumere vantagens e inconvenientes nas diversas vertentes que abordar.(2 valores)

Resposta:

O ambiente gráfico do Flash/ActionScript está associado ao Flash [versão 8 é a mais actual a data] e aos seus players (que são gratuitos e vêm embebidos na generalidade dos browsers para pc e Mac, em dispositivos moveis como telemóveis e consolas de jogos) [versão 9.0] a data.

O flash usa gráficos vectoriais 2D como base do seu ambiente multimédia, permitindo “footprints” para as aplicações/programas muito pequenos comparativamente a outros ambientes. Este ambiente com dez anos de existência sofreu uma evolução muito grande desde o seu surgimento e é considerado já um standard de facto para animações 2D na Internet (web).

As últimas versões do Flash permitem trabalhar com vídeo, som (mp3) e interagir com ambientes servidor na Internet de forma cada vez mais avançada estando a Adobe (empresa detentora do Flash) a promover ambientes programáticos baseados na tecnologia Flash com muitas funcionalidades e com elevada interoperabilidade com outros programas e ambientes, caso do Flex.

O flash é fácil de aprender, possui um ambiente de desenvolvimento interactivo de fácil aprendizagem e as aplicações geradas podem ser distribuídas para os mais diversos tipos de plataformas com muito poucas adaptações.

Como os gráficos e animações geradas são pseudo-interpretadas a sua velocidade é limitada comparativamente a ambientes gráficos nativos como o DirectX e OpenGL. É um ambiente gráfico 2D que apresenta como pontos fortes a portabilidade, estabilidade e facilidade de desenvolvimento.

O API DirectX9 é desenvolvido e distribuído pela Microsoft para ser utilizado nas suas linhas de sistemas operativos (Win98, win2000, WinXP, Windows Vista, Xbox ...) só podendo ser utilizado em projectos destinados a estas plataformas. É um API multi-dispositivo (ou seja possui recursos para a parte gráfica, som, jogos multiplayer, dispositivos de input – como joystick, ou outros). Tem evoluído regularmente e actualmente é usado não só em PC's, como em consolas (xbox) é GRATUITO mas possui como elementos a desfavor o não ser ‘open source’ ou Multi-Plataforma fora dos sistemas operativos da Microsoft. É uma api gráfica muito avançada quer para 2D quer para 3D que exige que se saiba programar num ambiente que possua capacidade de usar o DirectX, caso da generalidade dos ambientes de desenvolvimento para PC. O facto de não ser open-source limita a sua generalização a outros ambientes operativos. Apresenta como pontos fortes ser um ambiente robusto a 3D, possuir recursos para som, vídeo, ambientes ligados em rede, e a generalidade dos fabricantes de placas de vídeo avançadas para PC e portáteis optimizarem as suas placas para este API.

Esta pergunta vale 2 valores em 20.

Critérios para correcção: Se apresenta três características diferenciados para cada API e faz a

enumeração de exemplos de aplicação dos API com contextualização da origem e tipo de software (100%). Se apresenta deficiência na caracterização de um deles mas globalmente demonstra entender bem o que são, para que servem e como se utilizam (-20%). Se apenas descreve um dos API (apenas 40% da cotação). Qualquer outra situação deve ser valorada usando como princípio o entendimento que o aluno demonstra de conhecimento desde tipo de recurso gráfico.

Pergunta 8

Crie um procedimento em actionscript que permita carregar um textura Jpeg da pasta .imagens e a mostre no palco do Flash. (2 valores)

Nitro PDF Trial
www.nitropdf.com