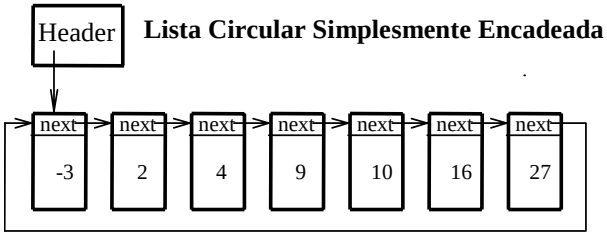
CIV 2801 – FUND. DE COMPUTAÇÃO GRÁFICA APLICADA – 2004.2

Prova Final 22 de novembro de 2004
Duração: 2:00 hs Sem consulta

1ª Questão (2,5 pontos)
A figura abaixo ilustra um tipo de estrutura de dados chamada de lista circular simplesmente encadeada. No caso é uma lista em cujos elementos estão sendo armazenados números inteiros de forma ordenada. A lista é circular pois o último elemento aponta para o primeiro elemento (aquele que é apontado pelo ponteiro cabeça – *header*).



A implementação desta lista em C pode ser feita utilizando a seguinte definição:

```
typedef struct _sll {
  struct _sll    *next;
  int            n;
 } Sll;
```

Considere que existe uma variável global **`header`** que é o ponteiro para o primeiro elemento da lista, isto é, este ponteiro sempre aponta para o elemento que contém o menor número:

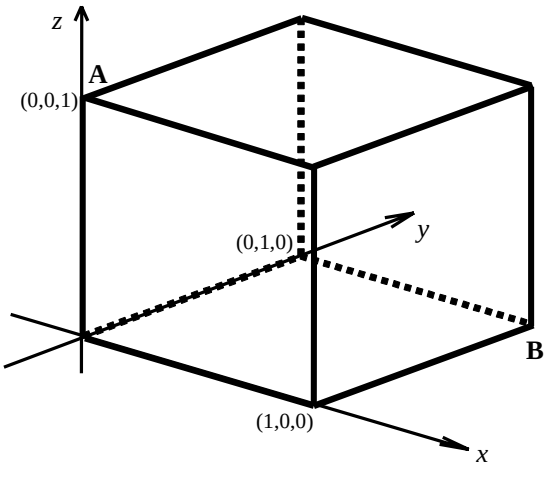
```
static Sll *header;
```

Pede-se a implementação em C das seguintes funções:

- **`Sll *SllFind( int n );`** Esta função retorna um ponteiro para o elemento da lista que contém um número inteiro **`n`** fornecido. A função deve tratar do caso de lista vazia (ponteiro **`header`** nulo) e do caso de o número fornecido não estar na lista. Em ambos os casos, o valor de retorno da função é nulo.
- **`void SllDelete( int n );`** Esta função remove da lista um elemento que é dado pelo seu número inteiro. A função libera a memória utilizada pelo elemento e faz a concatenação dos ponteiros dos elementos adjacentes ao removido para manter a consistência da lista. O caso de só existir um elemento na lista (o que é removido) também deve ser tratado. Caso o número fornecido não esteja na lista, nada deve ser feito.

2ª Questão (2,5 pontos)
(a) Para o cubo com lados unitários, encontre os parâmetros do modelo de câmera (posições dos pontos do olho e de referência e componentes do vetor $\boldsymbol{v_{up}}$ que define o plano vertical da câmera) sabendo que:

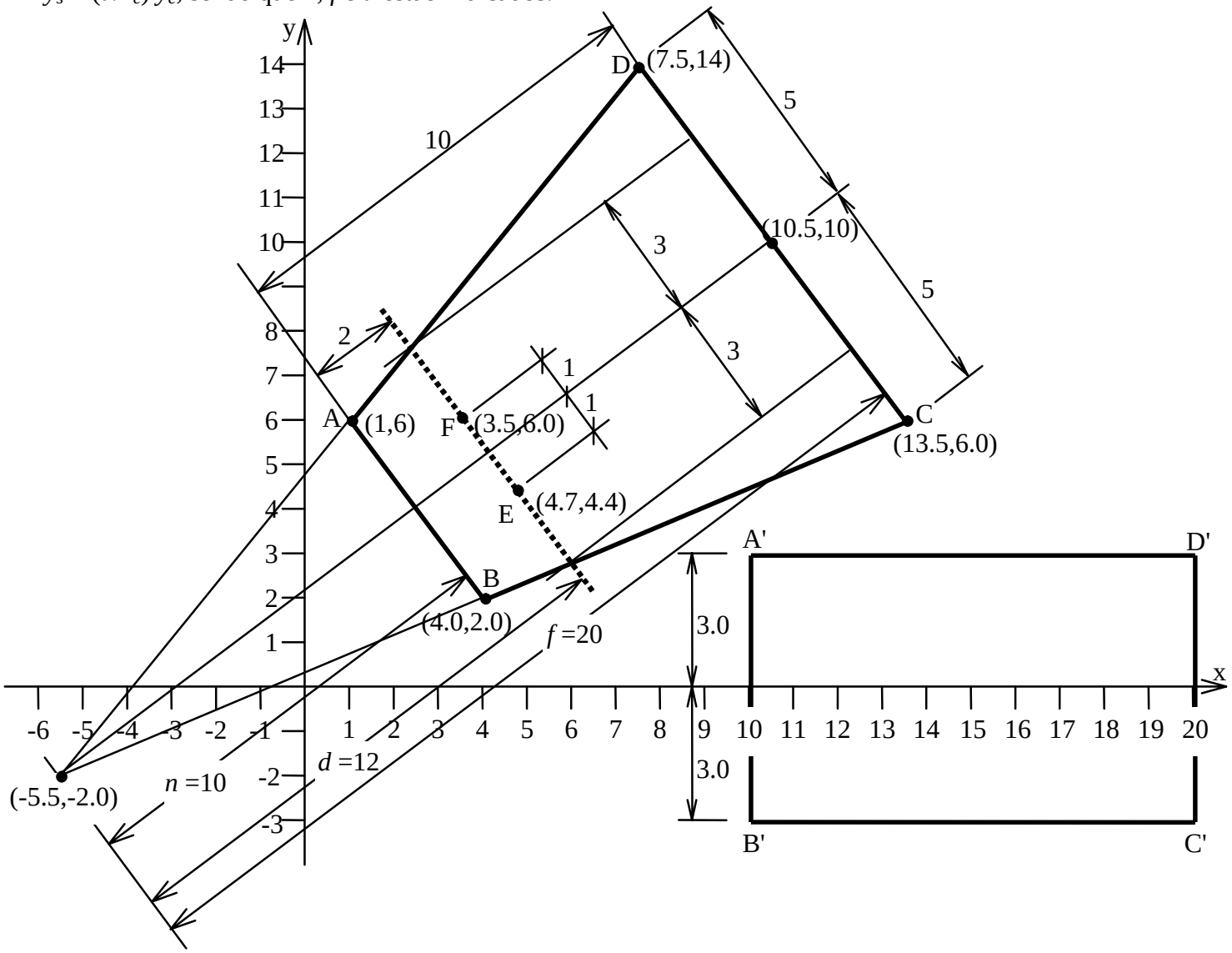
- A linha de visão (reta entre o olho e o ponto de referência) passa pelos pontos **A** e **B**. Os dois pontos ficam coincidentes na imagem projetada.
- O ponto de referência fica no centro do cubo.
- O ponto do olho fica a uma distância de 3 diagonais do cubo em relação ao ponto **A** e a 4 diagonais do ponto **B**.
- O eixo *z* aparece vertical na imagem projetada na tela do computador.



(b) Quais as componentes, no sistema de coordenadas *xyz*, dos vetores unitários $\boldsymbol{x_e}$, $\boldsymbol{y_e}$ e $\boldsymbol{z_e}$ do sistema de coordenadas do olho?

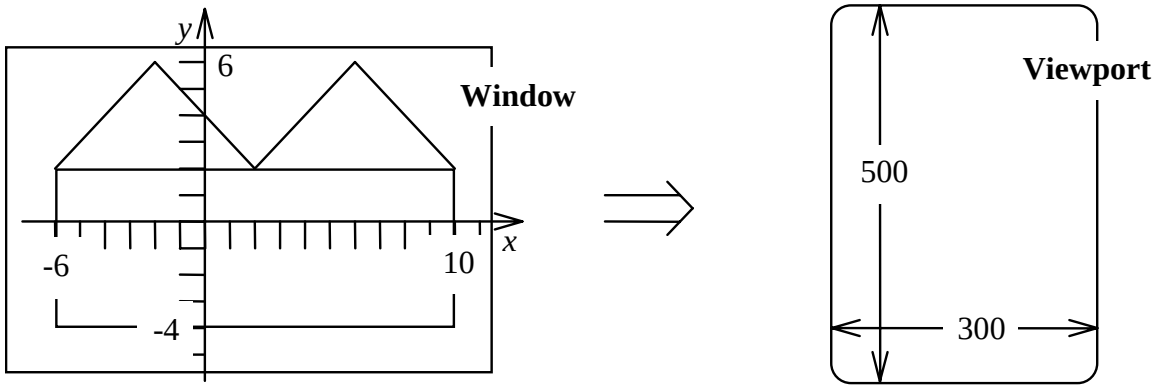
3ª Questão (2,5 pontos)

Determine uma possível concatenação de matrizes que transforma os pontos do $\mathbf{R^2}$ de tal forma que o trapézio ABCD é transformado no retângulo A'B'C'D', como indicado no desenho abaixo. Observe a analogia com o modelo de câmera da visualização 3D e observe também que os pontos E e F se situam no "plano de projeção", isto é, a distância entre E e F não se altera. Nesta analogia, $x_s = (n+f) - (n \cdot f)/x_e$ e $y_s = (d/x_e) \cdot y_e$, sendo que n, f e d estão indicados.



4ª Questão (2,5 pontos)

Considere a transformação *window-viewport* mostrada abaixo. Determine a janela (*window* – retângulo no espaço *xy*) necessária para posicionar o objeto da figura da esquerda de forma centrada no *viewport* da figura da direita, sem que a imagem fique distorcida (escala na direção horizontal igual à escala na direção vertical). Também deve existir uma folga de no mínimo 5% do tamanho do objeto em relação à janela.



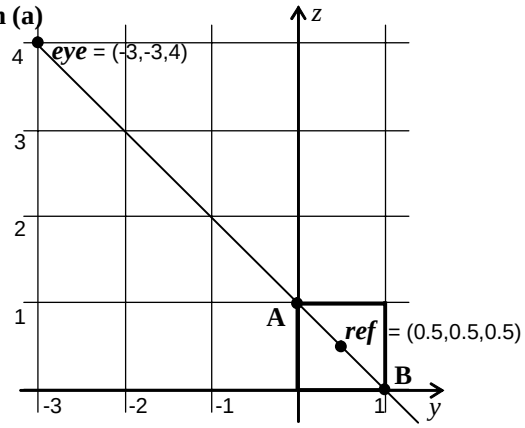
1ª Questão

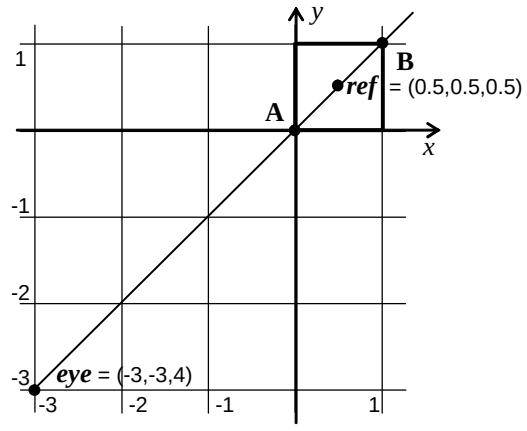
```
#include <stdlib.h>

typedef struct _sll {
  struct _sll *next;
  int n;
 } Sll;

static Sll *header;

Sll *SllFind( int n )
{
 Sll *elem;

 if( header == NULL )
 {
  return( NULL );
 }

 elem = header;
 do
 {
  if( elem->n == n )
  {
   return( elem );
  }
  elem = elem->next;
 } while( elem != header );

 return( NULL );
}
```

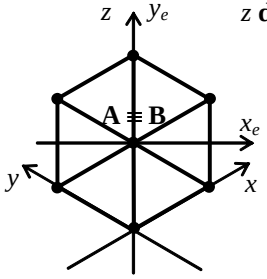
```
void SllDelete( int n )
{
 Sll *elem;
 Sll *prev = header;

 elem = SllFind( n );
 if( elem == NULL )
  return;

 if( elem->next == elem )
 {
  header = NULL;
 }
 else
 {
  while( prev->next != elem )
  {
   prev = prev->next;
  }
  prev->next = elem->next;
  if( header == elem )
   header = elem->next;
 }
 free( elem );
}
```

2ª Questão

Item (a)







z define o sentido de vup

⇓

vup = (0,0,1)

(***vup*** pode ser paralelo a *z* pois só serve para definir o plano vertical da câmera)

Item (b) (coordenadas *xyz*)

$\boldsymbol{z_e}$ **é paralelo à reta BA** $\Longrightarrow$ $\boldsymbol{z_e} = (-\sqrt{3}/3, -\sqrt{3}/3, \sqrt{3}/3)$

$$\boldsymbol{x_e} = \frac{\boldsymbol{vup} \times \boldsymbol{z_e}}{\|\boldsymbol{vup} \times \boldsymbol{z_e}\|} \Longrightarrow \boldsymbol{x_e} = (\sqrt{2}/2, -\sqrt{2}/2, 0)$$

$$\boldsymbol{y_e} = \boldsymbol{z_e} \times \boldsymbol{x_e} \Longrightarrow y_e = (\sqrt{6}/6, \sqrt{6}/6, \sqrt{6}/3)$$

3ª Questão

Translação do olho para origem:

$$T := \begin{pmatrix} 1.0 & 0.0 & 5.5 \\ 0.0 & 1.0 & 2.0 \\ 0.0 & 0.0 & 1.0 \end{pmatrix}$$

Rotação dos eixos:
(componentes dos vetores unitários nas linhas da submatriz de transformações lineares)

$$R := \begin{pmatrix} \frac{4}{5} & \frac{3}{5} & 0.0 \\ \frac{-3}{5} & \frac{4}{5} & 0.0 \\ 0.0 & 0.0 & 1.0 \end{pmatrix}$$

Ponto qualquer no sistema do olho:

$$\begin{pmatrix} xe \\ ye \\ w \end{pmatrix} := R \cdot T \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Distorção do trapézio em um retângulo:
(análogo à distorção do frustum em um paralelepípedo)

$n := 10 \qquad f := 20 \qquad d := 12$

$$P := \begin{bmatrix} (n+f) & 0 & -(n \cdot f) \\ 0 & d & 0 \\ 1 & 0 & 0 \end{bmatrix} \qquad \begin{pmatrix} xh \\ yh \\ w \end{pmatrix} := P \cdot \begin{pmatrix} xe \\ ye \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} xh \\ yh \\ w \end{pmatrix} := P \cdot R \cdot T \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \qquad \begin{pmatrix} xs \\ ys \end{pmatrix} := \begin{pmatrix} \frac{xh}{w} \\ \frac{yh}{w} \end{pmatrix}$$

Verficação do ponto A:

$xA := 1.0 \qquad yA := 6.0$

$$\begin{pmatrix} xhA \\ yhA \\ w \end{pmatrix} := P \cdot R \cdot T \cdot \begin{pmatrix} xA \\ yA \\ 1 \end{pmatrix} \qquad \begin{pmatrix} xsA \\ ysA \end{pmatrix} := \begin{pmatrix} \frac{xhA}{w} \\ \frac{yhA}{w} \end{pmatrix} \qquad \begin{pmatrix} xsA \\ ysA \end{pmatrix} = \begin{pmatrix} 10 \\ 3 \end{pmatrix}$$

Verficação do ponto C:

$xC := 13.5 \qquad yC := 6.0$

$$\begin{pmatrix} xhC \\ yhC \\ w \end{pmatrix} := P \cdot R \cdot T \cdot \begin{pmatrix} xC \\ yC \\ 1 \end{pmatrix} \qquad \begin{pmatrix} xsC \\ ysC \end{pmatrix} := \begin{pmatrix} \frac{xhC}{w} \\ \frac{yhC}{w} \end{pmatrix} \qquad \begin{pmatrix} xsC \\ ysC \end{pmatrix} = \begin{pmatrix} 20 \\ -3 \end{pmatrix}$$

Verficação do ponto E:

$xE := 4.7 \qquad yE := 4.4$

$$\begin{pmatrix} xhE \\ yhE \\ w \end{pmatrix} := P \cdot R \cdot T \cdot \begin{pmatrix} xE \\ yE \\ 1 \end{pmatrix} \qquad \begin{pmatrix} xsE \\ ysE \end{pmatrix} := \begin{pmatrix} \frac{xhE}{w} \\ \frac{yhE}{w} \end{pmatrix} \qquad \begin{pmatrix} xsE \\ ysE \end{pmatrix} = \begin{pmatrix} 13.333 \\ -1 \end{pmatrix}$$

Verficação do ponto F:

$xF := 3.5 \qquad yF := 6.0$

Distância entre E e F não se altera

$$\begin{pmatrix} xhF \\ yhF \\ w \end{pmatrix} := P \cdot R \cdot T \cdot \begin{pmatrix} xF \\ yF \\ 1 \end{pmatrix} \qquad \begin{pmatrix} xsF \\ ysF \end{pmatrix} := \begin{pmatrix} \frac{xhF}{w} \\ \frac{yhF}{w} \end{pmatrix} \qquad \begin{pmatrix} xsF \\ ysF \end{pmatrix} = \begin{pmatrix} 13.333 \\ 1 \end{pmatrix}$$

4ª Questão

```
vpr = 500.0/300.0 = 1.67;          /* razão de aspecto do viewport */
cx = (-6.0+10.0)/2.0 = 2.0;        /* posição x do centro da window */
cy = (-4.0+ 6.0)/2.0 = 1.0;        /* posição y do centro da window */
sizex = (10.0+6.0)*1.05 = 16.8;    /* tamanho horizontal da window */
sizey =  (6.0+4.0)*1.05 = 10.5;    /* tamanho vertical inicial da window */

Como sizey < (vpr*sizex), então
sizey = sizex * vpr = 28.0;        /* tamanho vertical corrigido da window */

xmin = cx - (sizex / 2.0) =  -6.4; /* limite esquerdo da window */
xmax = cx + (sizex / 2.0) = +10.4; /* limite direito da window */
ymin = cy - (sizey / 2.0) = -13.0; /* limite inferior da window */
ymax = cy + (sizey / 2.0) = +15.0; /* limite superior da window */
```