```
#include   <malloc.h>
#include   <stdio.h>
#include   "LISTA.h"



/* Inclusão do próprio módulo de definição */
#define MATRIZ_OWN
#include "MATRIZ.h"
#undef MATRIZ_OWN

/***** Protótipos das funções encapuladas no módulo *****/

static void DestruirElemento ( void* pValor );

/*****  Código das funções exportadas pelo módulo  *****/

/***************************************************************************
*
*  Função: MAT Cria Matriz
*  ****/

MAT_tpCondRet MAT_CriaMatriz(LIS_tppLista pLista,  int largura , int altura)
{
      int i;
      int j;
      LIS_tppLista a[100];



      pLista=LIS_CriarLista(DestruirElemento);
      for(i=0;i<altura;i++)
      {
            if(LIS_InserirElementoApos(pLista , &a[i] )!=LIS_CondRetOK)
            {
                  return MAT_CondRetNOT;
            }

            //IrInicioLista(pLista);

            //LIS_AvancarElementoCorrente(pLista , i ) ;

            a[i]=LIS_CriarLista(DestruirElemento);
            for(j=0;i<largura;j++)
            {
                  if(LIS_InserirElementoApos( a[i], NULL )!=LIS_CondRetOK)
                        {
                              return MAT_CondRetNOT;
```

```
                        }
                }
                IrInicioLista(a[i]);
        }

        IrInicioLista(pLista);
        return MAT_CondRetOK;

}

  MAT_tpCondRet MAT_InsereValor(LIS_tppLista pLista, void * pValor, int
largura , int altura)
  {


        LIS_tppLista pontLista;

        IrInicioLista(pLista);

        LIS_AvancarElementoCorrente(pLista , altura-1 );

        pontLista = (LIS_tppLista )LIS_ObterValor( pLista ) ;

        if(LIS_InserirElementoApos( pontLista , pValor )!=LIS_CondRetOK)
                {
                        return MAT_CondRetNOT;
                }


                return MAT_CondRetOK;
  }

  void * MAT_ObterValor( LIS_tppLista pLista ,int largura,int altura)
  {

        LIS_tppLista pontLista;

        IrInicioLista(pLista);

        LIS_AvancarElementoCorrente(pLista , altura-1 );

        pontLista = (LIS_tppLista )LIS_ObterValor( pLista ) ;

        LIS_AvancarElementoCorrente(pLista , largura-1 );

        return ;
```

```
  }
```

```
static void DestruirElemento ( void* pValor )
{
      if(pValor != NULL)
      {
            free(pValor);
      }
}
```