```
void ret (LISTA_ENC *pl, int k) {
   NODO *aux;
   if (k < 1 || k > tam(*pl)) {
       printf ("\nERRO! Posição invalida para retirada.\n");
       exit (4);
   }
   if (k==1) {
      aux = *pl;
      *pl = aux->next;
      free (aux);
   }
```

```
    else
    {
       NODO *aux2;
       for (aux=*pl; k>2; k--, aux=aux->next);
       aux2 = aux->next;
       aux->next = aux2->next;
       free (aux2);
    }
}
```

Alocação Encadeada

Com base no que foi visto implemente a operação ret() que compõem o TAD LISTA_ENC. Porém, implemente a operação utilizando recursividade.

```
typedef struct nodo
{
   int inf;
   struct nodo * next;
}NODO;
typedef NODO * LISTA_ENC;
void cria_lista (LISTA_ENC *);
int eh_vazia (LISTA_ENC);
int tam (LISTA_ENC);
void ins (LISTA_ENC *, int, int);
int recup (LISTA_ENC, int);
void ret (LISTA_ENC *, int);
```

Alocação Encadeada - Exercício

Implemente, no TAD LISTA_ENC, a seguinte operação:

int pertence (LISTA_ENC l, int v)

a qual retorna 1 (um) se v pertence a lista l e 0 (zero) caso contrário.

```
int pertence (LISTA_ENC l, int v)
{
   int t;
   for (t=tam(l); t>0; t--, l=l->next)
      if (l->inf==v)
         return (1);
   return (0);
}
```

```
int pertence (LISTA_ENC l, int v)
{
   while (l)
   {
      if (l->inf==v)
         return (1);
      l=l->next;
   }
   return (0);
}
```

Alocação Encadeada - Exercício

Implemente, no TAD LISTA_ENC, a seguinte operação:

int eh_ord (LISTA_ENC l)

a qual retorna 1 (um) se a lista l está em ordem crescente e 0 (zero) caso contrário.

```
int eh_ord (LISTA_ENC l)
{
   int t;
   for (t=tam(l); t>1; t--, l=l->next)
      if (l->inf > (l->next)->inf)
         return (0);
   return (1);
}
```

```
int eh_ord (LISTA_ENC l)
{
   if (eh_vazia(l) || tam(l)==1)
      return 1;
   do
   {
      if (l->inf > (l->next)->inf)
         return (0);
      l=l->next;
   }while (l->next);
   return (1);
}
```

```
int eh_ord (LISTA_ENC l)
{
   if (!l || (l && !(l->next)))
      return 1;
   do
   {
      if (l->inf > (l->next)->inf)
         return (0);
      l=l->next;
   }while (l->next);
   return (1);
}
```

Alocação Encadeada - Exercício

Implemente, no TAD LISTA_ENC, utilizando recursividade, a seguinte operação:

void gera_lista (LISTA_ENC *pl,int m,int n)

a qual utilizando-se das operações do TAD LISTA produz uma lista de inteiros correspondente a [m..n].

UNIVASF

```
void gera_lista (LISTA_ENC *pl, int m, int n)
{
   if (m>n)
   {
      printf ("\nERRO! Intervalo invalido.\n");
      exit (5);
   }
   else
```

```
    if (m==n)
    {
       cria_lista (pl);
       ins (pl, m, 1);
    }
    else
    {
       gera_lista (pl, m+1, n);
       ins (pl, m, 1);
    }
}
```

Alocação Encadeada

Com base no TAD LISTA_ENC, que acabamos de implementar, construa um programa que ofereça ao usuário a possibilidade de inserir, remover e consultar o valor de um elemento em uma lista. Bem como, imprimir a sequência de elementos contidos na mesma.

```
typedef struct nodo
{
   int inf;
   struct nodo * next;
}NODO;
typedef NODO * LISTA_ENC;
void cria_lista (LISTA_ENC *);
int eh_vazia (LISTA_ENC);
int tam (LISTA_ENC);
void ins (LISTA_ENC *, int, int);
int recup (LISTA_ENC, int);
void ret (LISTA_ENC *, int);
```