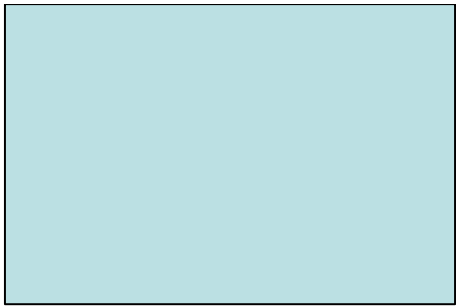
Alocação Sequencial - Exercício

Implemente, no TAD LISTA, a seguinte operação:

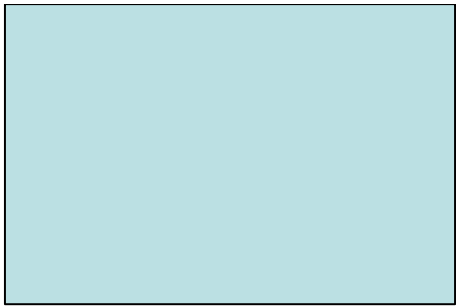
```
int pertence (LISTA *l, int v);
```

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

```
int pertence (LISTA *l, int v)
{
   int i;
   for (i=0; i<l->N; i++)
      if (l->val[i]==v)
         return (1);
   return (0);
}
```



Alocação Sequencial - Exercício

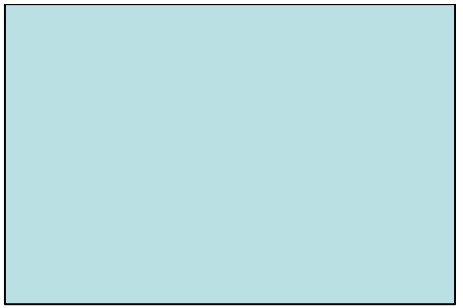


Implemente, no TAD LISTA, a seguinte operação:

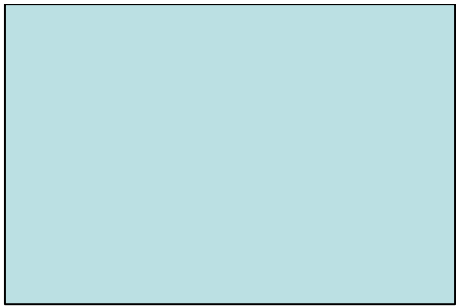
int eh_ord (LISTA *l);

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

```
int eh_ord (LISTA *l)
{
   int I;
   for (i=0; i<l->N-1; i++)
      if (l->val[i] > l->val[i+1])
         return (0);
   return (1);
}
```



Alocação Sequencial - Exercício



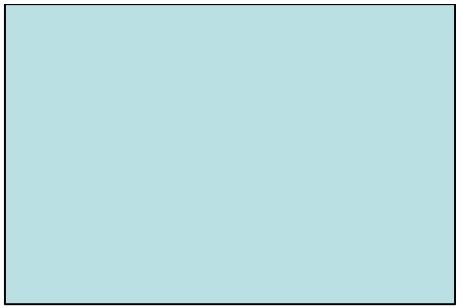
Implemente, no TAD LISTA, utilizando recursividade, a seguinte operação:

void gera_lista (LISTA *l, int m, int n);

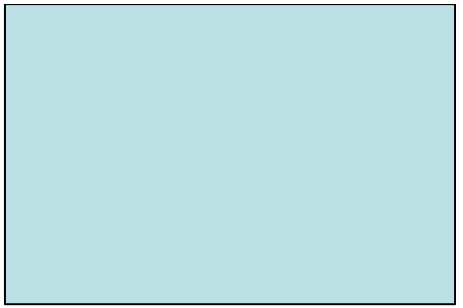
a qual utilizando-se das operações do TAD LISTA produz uma lista de inteiros correspondente a [m..n].

UNIVASF

```
void gera_lista (LISTA *l, int m, int n) {
   if (m>n) {
      printf ("\nERRO! Intervalo invalido.\n");
      exit (5);
   }
   else
      if (m==n) {
         cria_lista (l);
         ins (l, m, 1);
      }
```



```
    else
    {
       gera_lista (l, m+1, n);
       ins (l, m, 1);
    }
}
```



Listas – Alocação Encadeada

Alocação Encadeada

Desconsiderando as questões associadas ao armazenamento estático em memória, você consegue identificar alguma desvantagem em usar o armazenamento sequencial para representar listas?

Alocação Encadeada

Na alocação encadeada:

- a posição relativa dos elementos na memória não tem que corresponder estritamente à sua posição lógica na estrutura;

- os nós de uma lista encontram-se aleatoriamente dispostos na memória;

- sendo interligados por referências, que indicam a posição do próximo elemento da lista.

UNIVASF

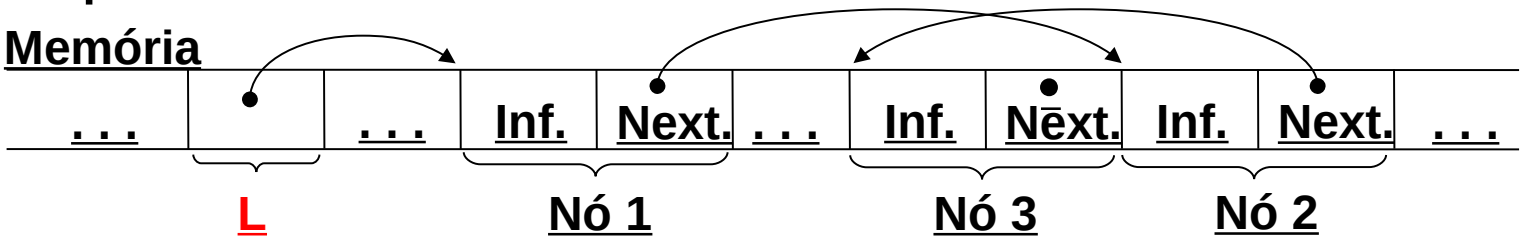
Alocação Encadeada

Logo:

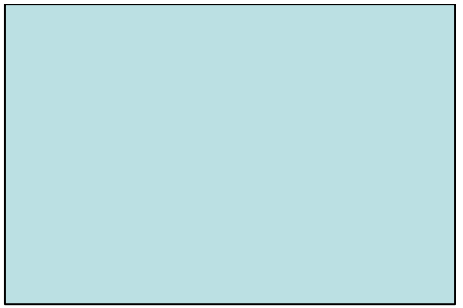
- aumento no custo de armazenamento x flexibilidade;

- acréscimo de um campo em cada nó para armazenar uma referência da posição de seu sucessor.

Esquema:



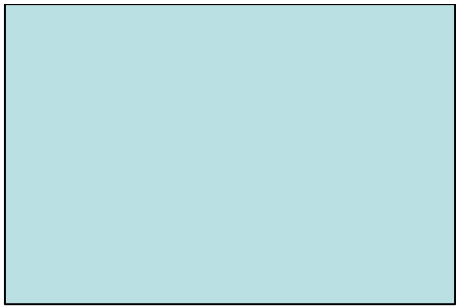
Alocação Encadeada



Uma lista encadeada pose ser armazenada em um vetor alocado estaticamente.

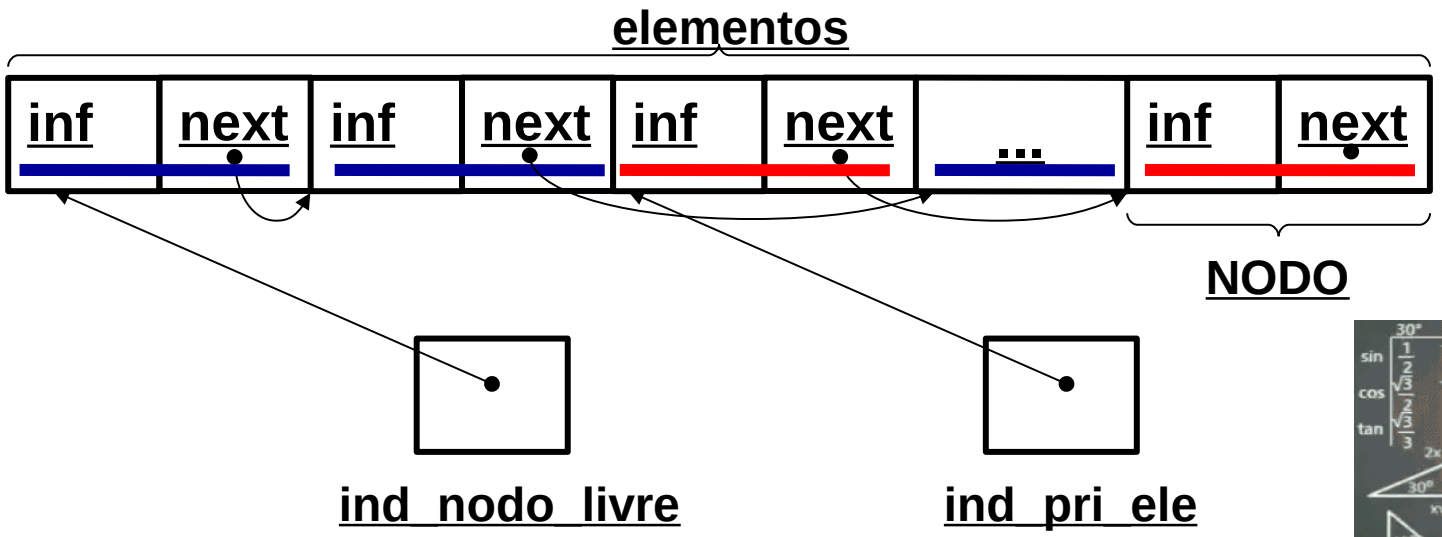
Para uma compreensão mais adequada vamos definir as estruturas necessárias para este processo.

UNIVASF



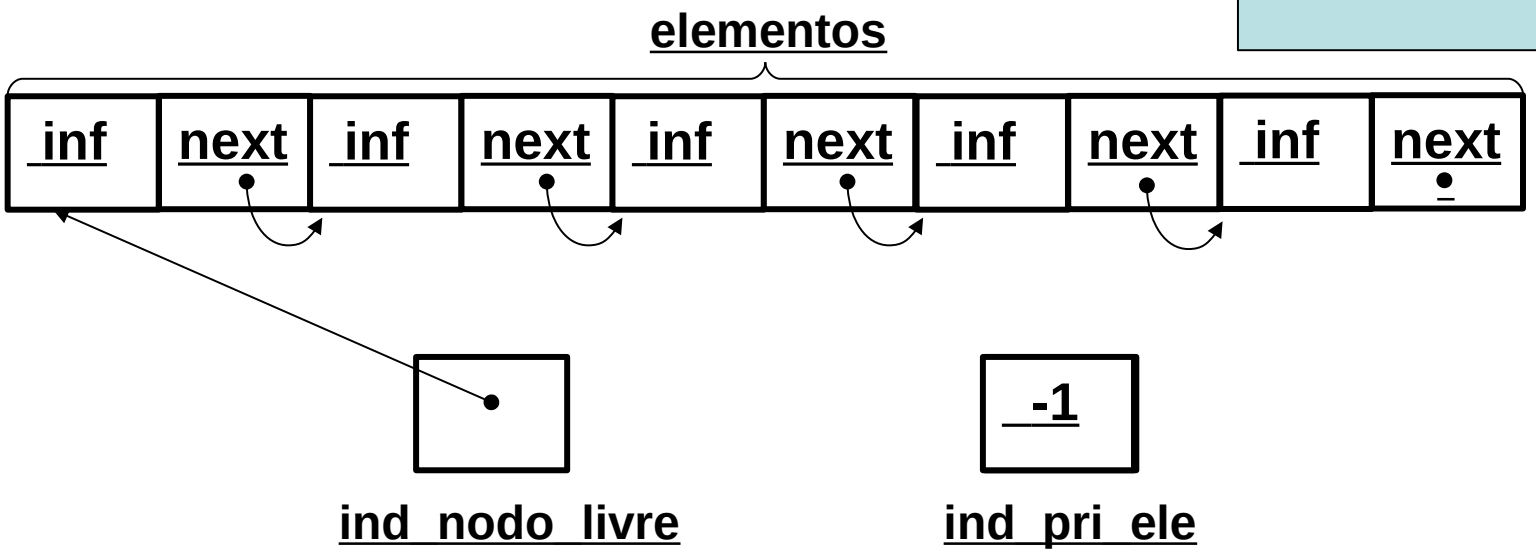
```
/*estruturas do TAD LISTA_ENC_EST*/
typedef struct
{
   float inf;
   int next;
}NODO;
typedef struct
{
   int ind_pri_ele;
   int ind_nodo_livre;
   NODO elementos[max];  /*#define max 100, por exemplo*/
}LISTA_ENC_EST;
```

Alocação Encadeada



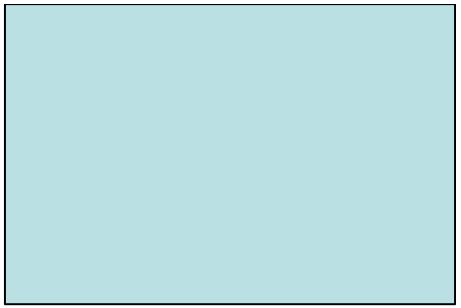
Com base no que foi dito, quais passos constituiriam a operação de criação de uma lista do TAD LISTA_ENC_EST.

Alocação Encadeada



Implemente esta operação.

```
void cria_lista (LISTA_ENC_EST *l)
{
   int i;
   l->ind_pri_ele = -1;
   l->ind_nodo_livre = 0;
   for (i=0; i<max-1; i++)
      l->elementos[i].next = i+1;
   l->elementos[i].next = -1;
}
```



Alocação Encadeada

Com base no que foi visto, implemente a operação que verifica se a lista é vazia no TAD LISTA_ENC_EST.

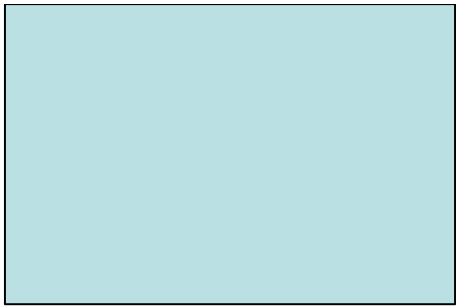
```
int eh_vazia (LISTA_ENC_EST *l)
{
        return (l->ind_pri_ele == -1);
}
```

Alocação Encadeada

Com base no que foi visto, implemente a operação que retorna o tamanho da lista no TAD LISTA_ENC_EST.

```
int tam (LISTA_ENC_EST *l) {
   int cont=0, ind=l->ind_pri_ele;
   while (ind!=-1) {
      cont++;
      ind = l->elementos[ind].next;
   }
   return cont;
}
```

Alocação Encadeada



Com base no que foi visto, implemente a operação que efetua a inserção de um elemento na lista representada pelo TAD LISTA_ENC_EST.

Dicas:

A posição é válida?

Tem espaço na memória para armazenar mais um elemento?

Todas as situações de inserção são tratadas da mesma forma?

UNIVASF