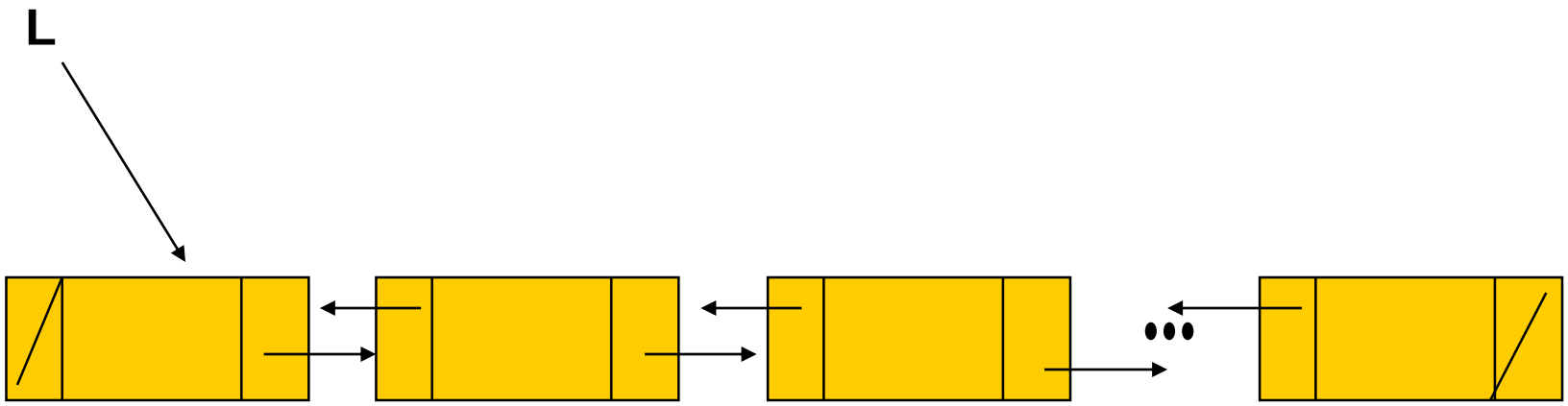
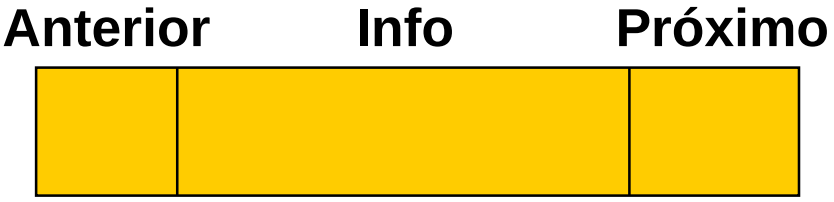
Listas Lineares Duplamente Encadeadas - LLDE

Estruturas de Dados

LL duplamente encadeadas





Nodo

Declaração

```
typedef struct temp{
        struct temp * ant;
        int info;
        struct temp * prox;
    }tLLDE;
```

Operações sobre LLDE

- criar lista, lendo dados de arquivo / teclado
- listar todos os nodos da lista
- listar de trás para diante
- inserir um nodo antes / depois de determinado nodo
- remover o primeiro / último / k-ésimo nodo
- trocar a ordem de 2 nodos
- ...

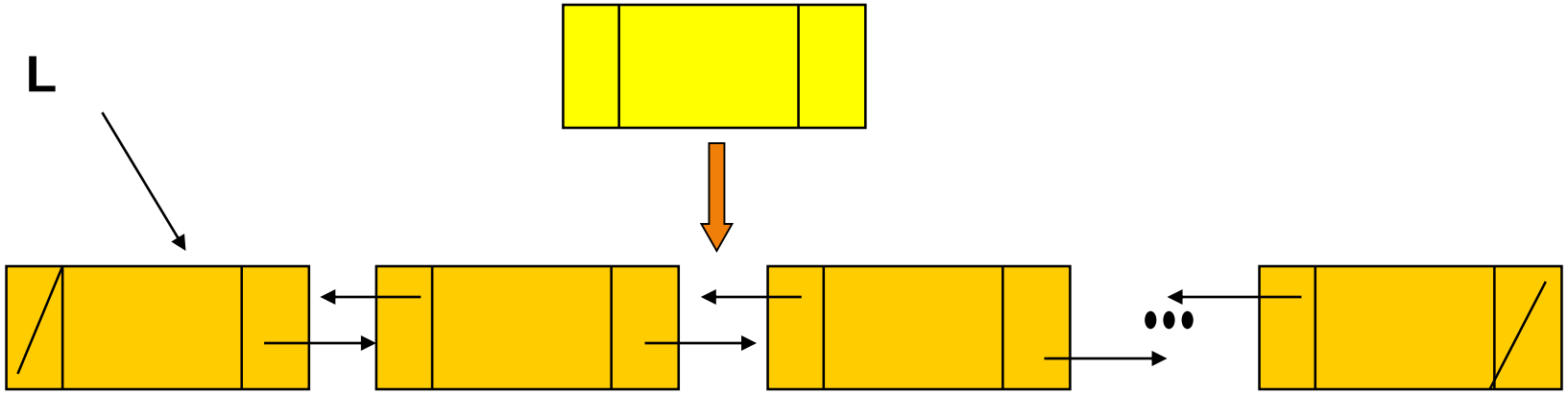
Imprimir invertido as Info da LLDE

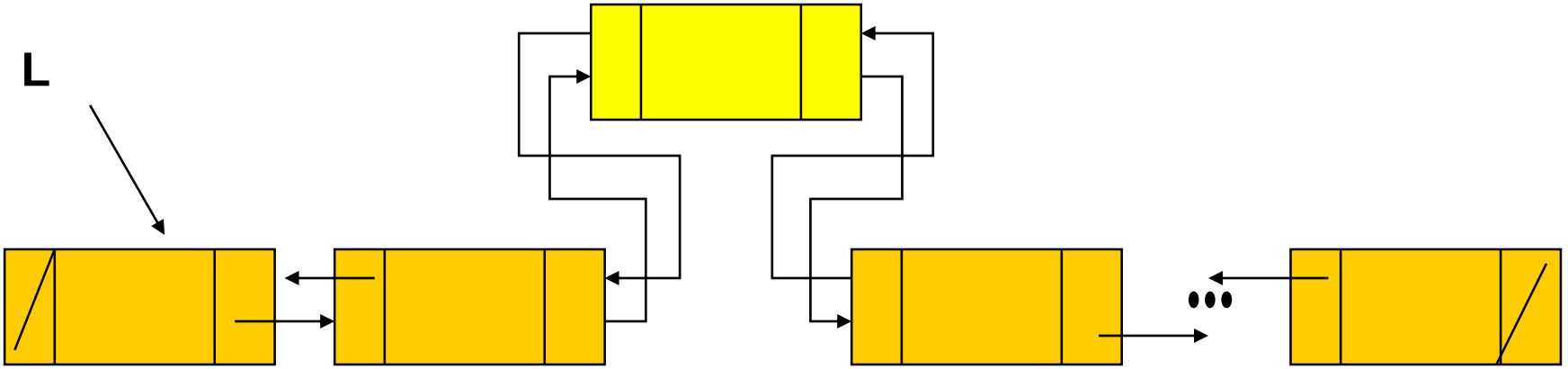
```
void ImprimeLLDEInvertida (tLLDE * L)
{
 tLLDE * ptAux ;
 if ( L = NULL)
    printf ( "Lista vazia ! \n")
 else {
          ptAux = L;          // ptAux percorre a lista até o final
          while (ptAux->prox != NULL)
             ptAux = ptAux ->prox;
           printf(ant \t <- valor ->\t prox\n");
           while (ptAux != NULL)
            {
              printf ("%d \t<-%d ->\t%d\n", ptAux->ant, ptAux->info, ptAux->prox );
              ptAux = ptAux->ant     // volta para nodo anterior
            }
       }
}
```

Inserir um novo nodo no final

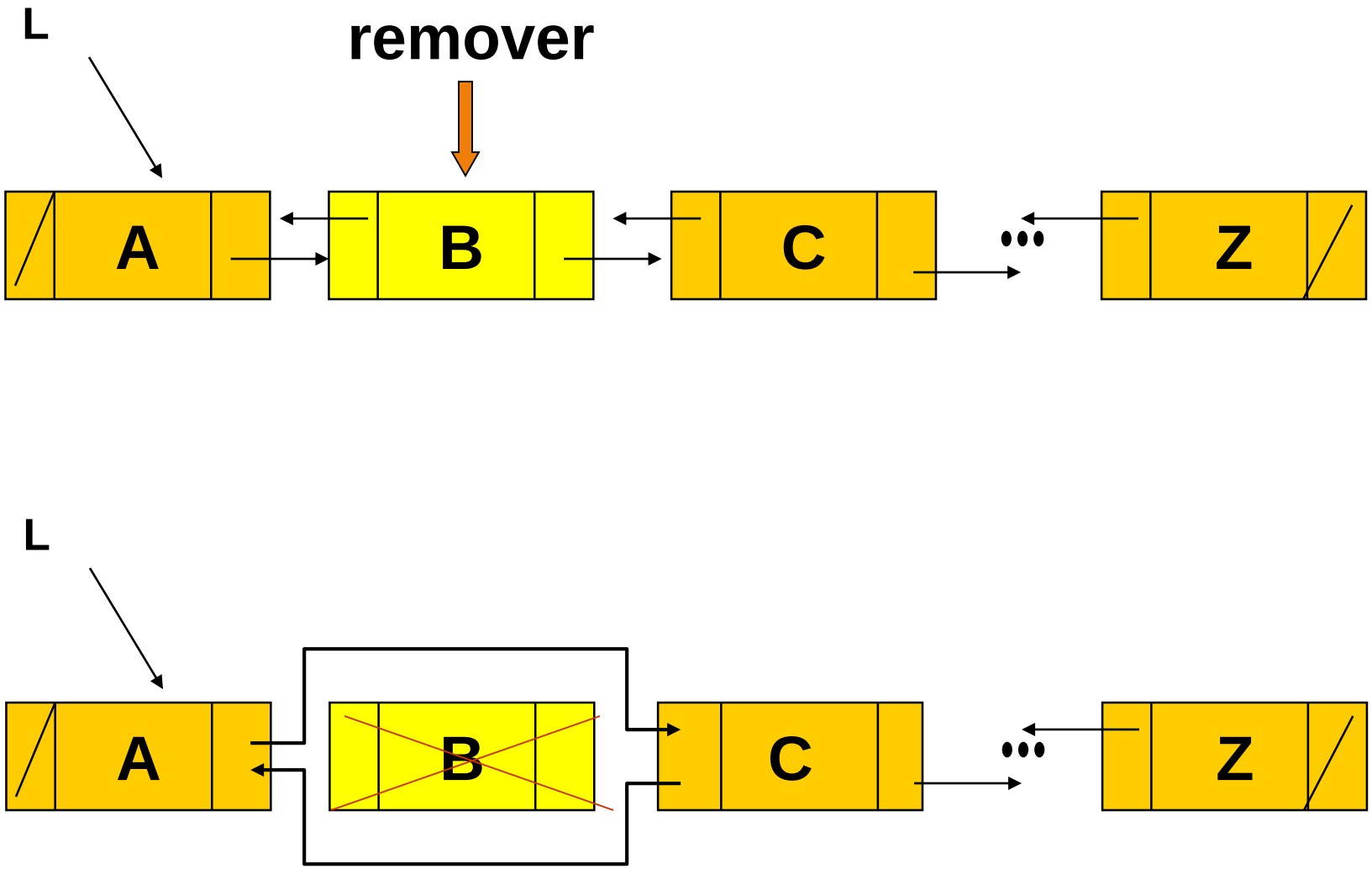
```
tLLDE * InserirNodoFinal (tLLDE *  L, int dado)
{
 tLLDE *novo, *ptAux;
 novo = (tLLDE *) malloc (sizeof (tLLDE)); //aloca novo nodo
 novo->info = dado;            //insere dado no novo nodo
 novo->prox = NULL;
 if (L == NULL)                // primeiro nodo
   {
        L = novo;
        novo->ant = NULL;
    } else {                          // mais de um nodo
           ptAux = L;        //posiciona auxiliar no início da lista
           while (ptAux->prox != NULL)
                 ptAux = ptAux->prox; // ptAux pára no último nodo
            ptAux->prox = novo;        //encadeia novo com ptAux
            novo->ant = ptAux;
            novo->prox = NULL;
          }
  return L;
}
```

Inserir um novo nodo no meio





Remover um novo nodo do meio



Algoritmos sugeridos :

- **incluir / remover um novo nodo no início de uma LL duplamente encadeada**

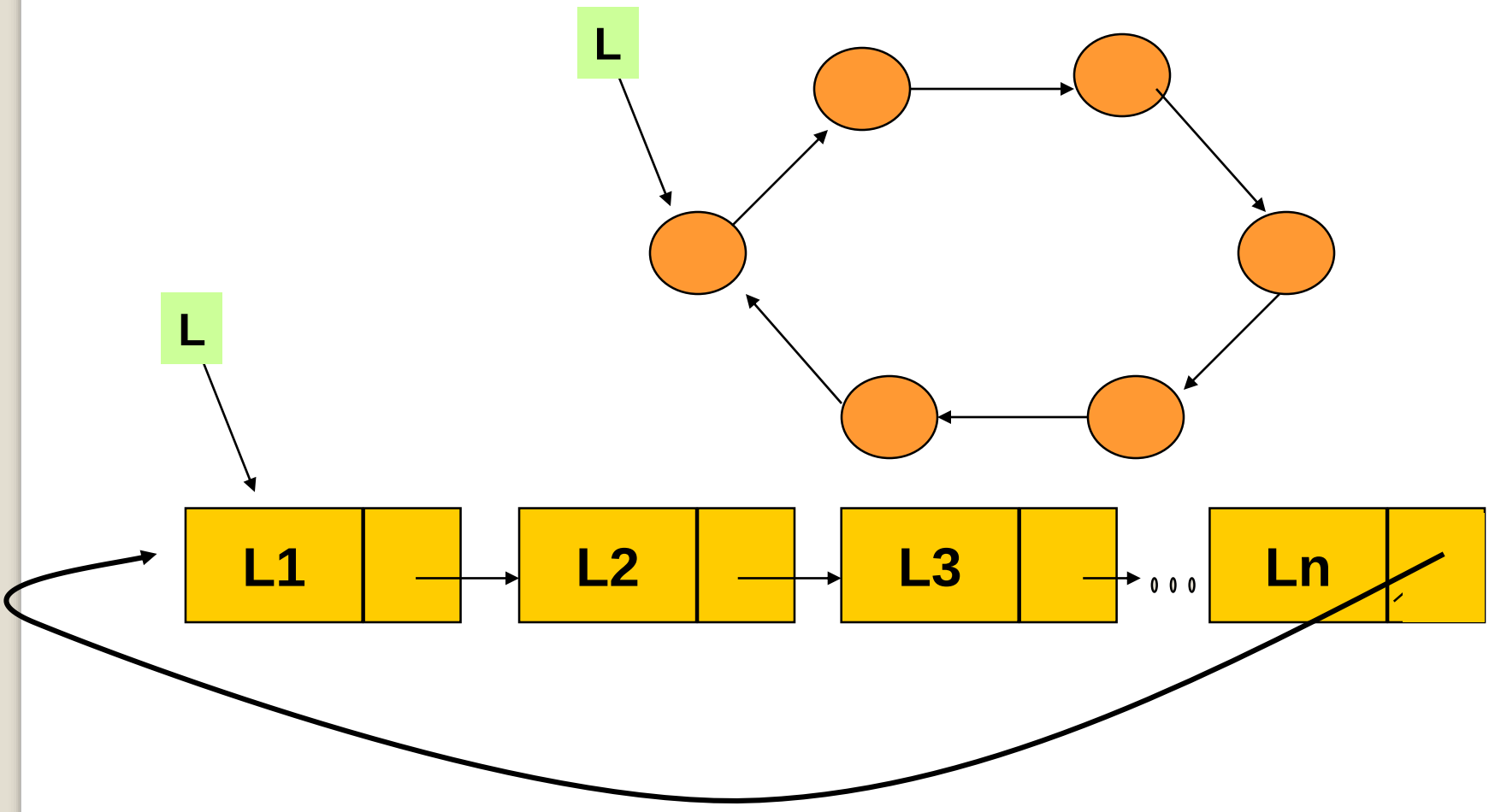
- **inserir um novo nodo na k-ésima posição da LL duplamente encadeada**

- **inserir um novo nodo antes daquele que apresentar uma determinada informação (*Info*)**

- **remover o nodo que contém uma determinada informação**

Lista Circular

Lista encadeada circular



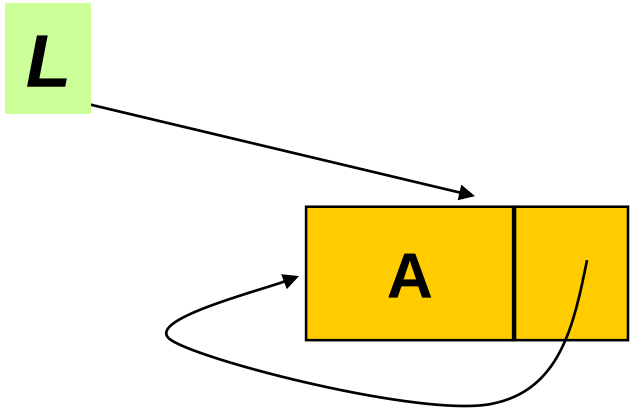
apontador L - qualquer nodo da lista

Lista encadeada circular

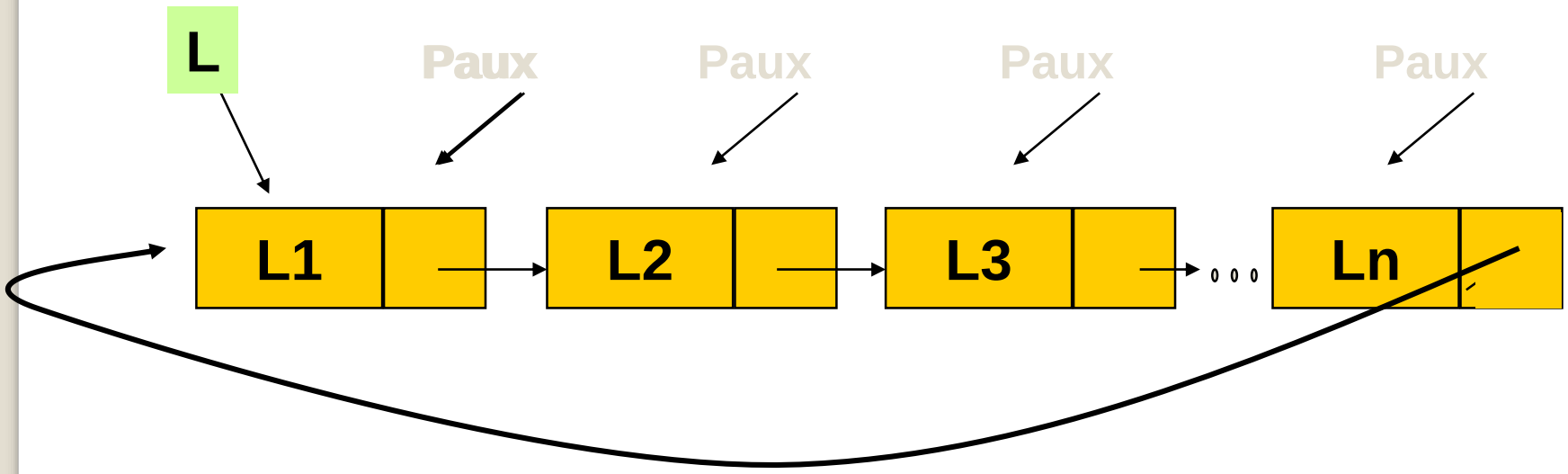
- **lista vazia**

L = NULL

- **lista com 1 só nodo**



Percorrer lista encadeada circular

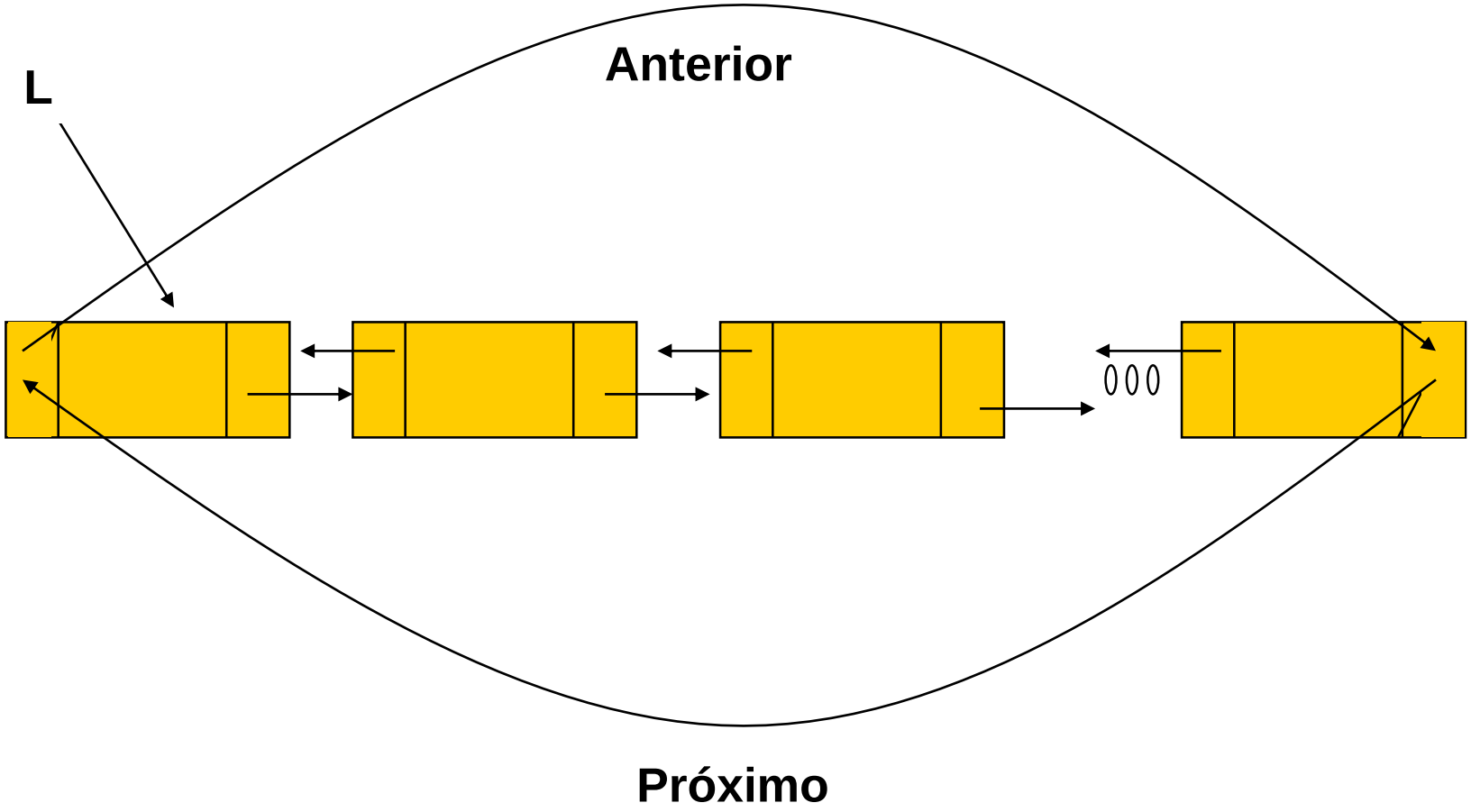


- parar quando o nodo inicial for novamente alcançado

Imprimir dados de LL circular encadeada

```
void ImprimeLLCircular (Lista * L)
{
 Lista * pAux;
 if (L == NULL)
   printf ("lista vazia!\n");
 else
 {
    pAux = L;
    do{
         printf("%d  \n", pAux->info);
         pAux = pAux->elo; //avança para o próximo nó
       }while(pAux !=  L)  //repete até voltar ao ini
  }
}
```

Lista circular duplamente encadeada



Algoritmos sugeridos :

- contruir lista circular duplamente encadeada, lendo dados do teclado
- inserção de novo nodo em LL circular simplesmente encadeada, logo após um nodo que contém um determinado campo de informação
- mesmo anterior, no caso de duplamente encadeada
- remoção de um nodo que contém uma determinada informação, nos dois casos (simplesmente e duplamente encadeada)
- Remoção de um nó dado seu endereço
- destruição de lista circular

Bibliografia

- Material dos professores:
  - Clesio S. Santos
  - Nina Edelweiss
- Introdução a Estruturas de Dados, Waldemar Celes, Renato Cerqueira José Lucas Rangel , Campus, 3ª Ed. ,2004