# Algoritmos e Estruturas de Dados II

2º Período Engenharia da Computação

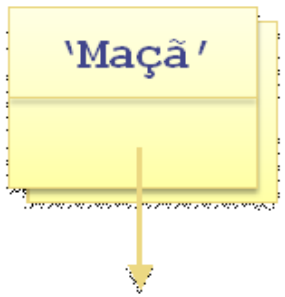
Prof. Edwaldo Soares Rodrigues
Email: edwaldo.rodrigues@uemg.br

UNIVERSIDADE
DO ESTADO DE MINAS GERAIS | UEMG
UNIDADE DIVINÓPOLIS

# Lista Encadeada (Dinâmica)

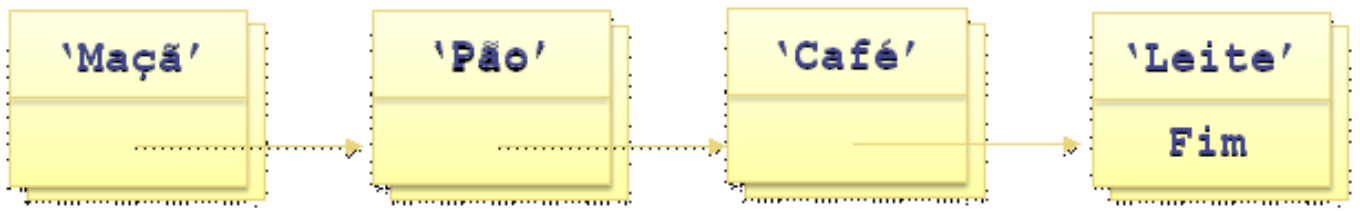
# Lista Encadeada (Dinâmica)

- Discussão intuitiva:
  - Ponteiros podem ser usados para construir estruturas, tais como listas, a partir de componentes simples chamados nó;



# Lista Encadeada (Dinâmica)

- Discussão intuitiva:
  - Um nó possui uma seta apontando para fora. Essa seta representa um ponteiro que aponta para outro nó, formando uma lista encadeada;

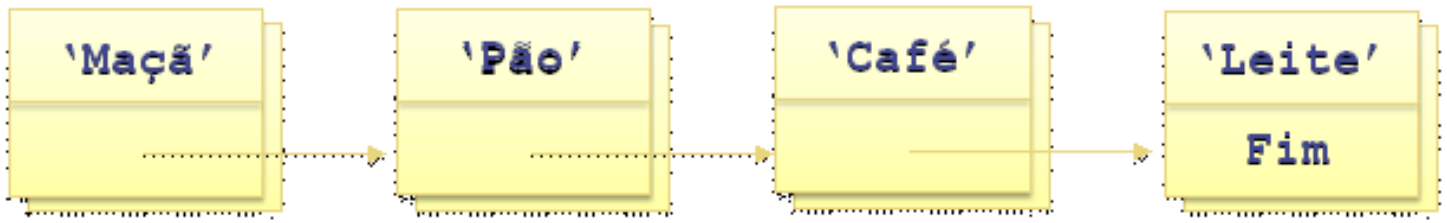


# Lista Encadeada (Dinâmica)

- Discussão intuitiva:
  - Listas encadeadas são úteis pois podem ser utilizadas para implementar o TAD lista. Nesse caso, as operações de inserção e remoção no meio da lista podem ser mais eficientes;
  - Uma segunda vantagem é o fato de não ser necessário informar o número de elementos em tempo de compilação;

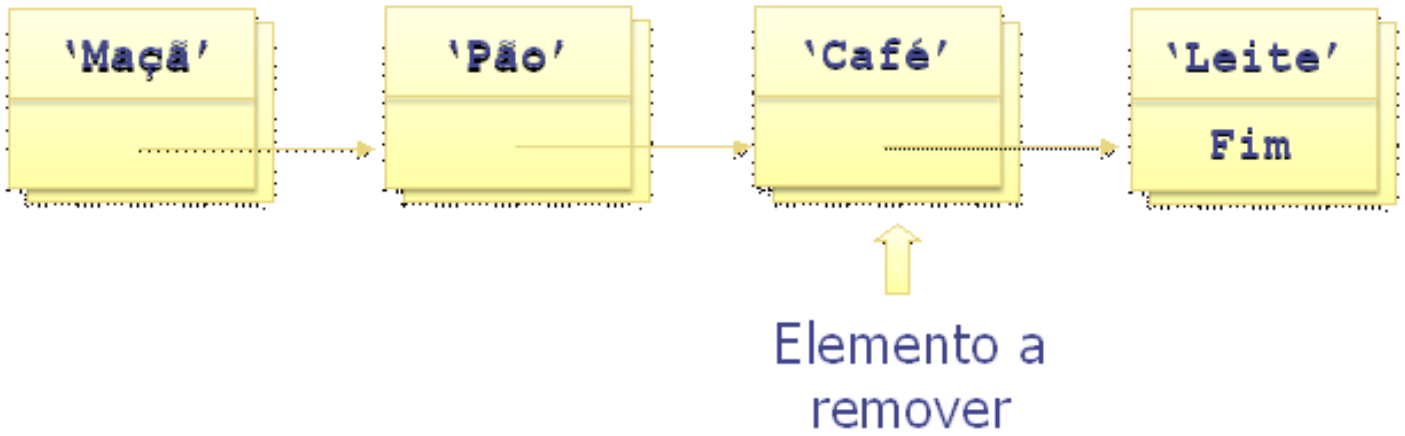
# Lista Encadeada (Dinâmica)

- Discussão intuitiva:
  - Por exemplo, uma operação de remoção pode ser feita da seguinte maneira:



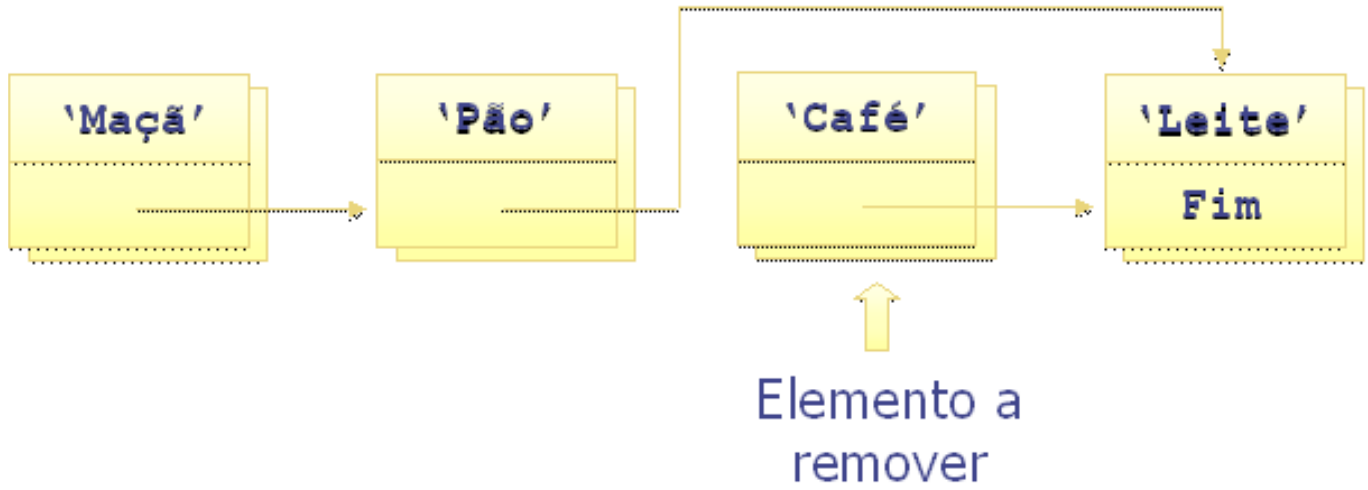
# Lista Encadeada (Dinâmica)

- Discussão intuitiva:
  - Por exemplo, uma operação de remoção pode ser feita da seguinte maneira:



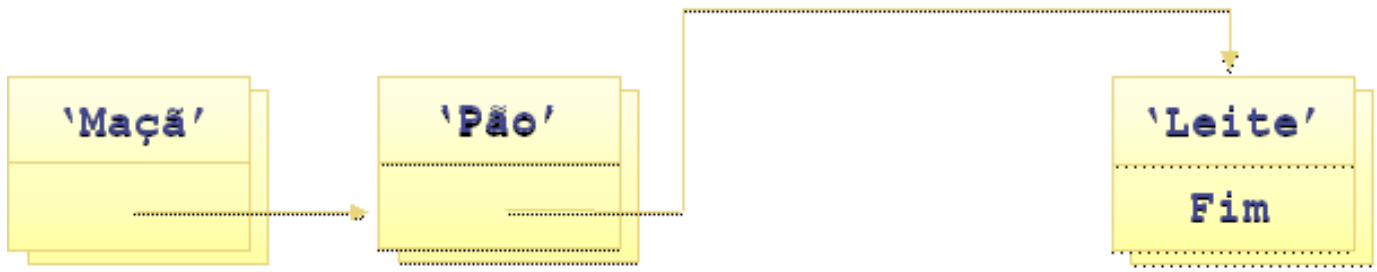
# Lista Encadeada (Dinâmica)

- Discussão intuitiva:
  - Por exemplo, uma operação de remoção pode ser feita da seguinte maneira:



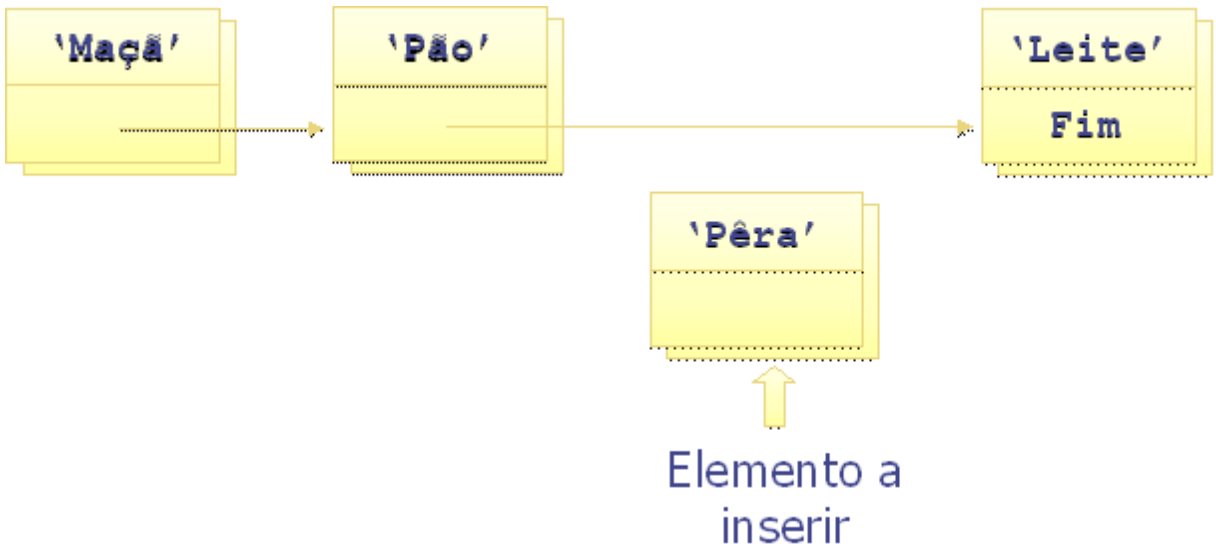
# Lista Encadeada (Dinâmica)

- Discussão intuitiva:
  - Por exemplo, uma operação de remoção pode ser feita da seguinte maneira:



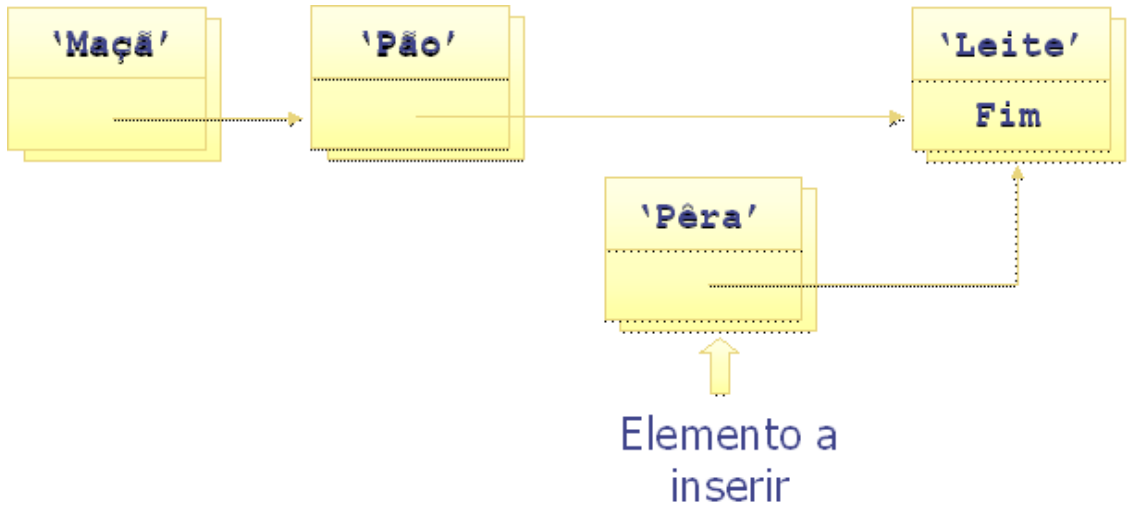
# Lista Encadeada (Dinâmica)

- Discussão intuitiva:
  - Por exemplo, uma operação de inserção pode ser feita da seguinte maneira:



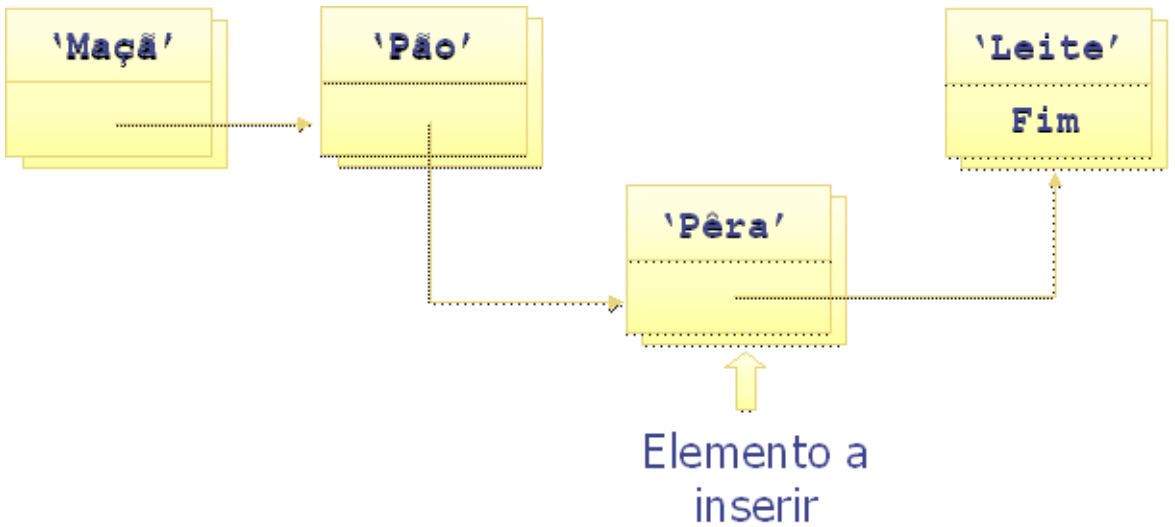
# Lista Encadeada (Dinâmica)

- Discussão intuitiva:
  - Por exemplo, uma operação de inserção pode ser feita da seguinte maneira:



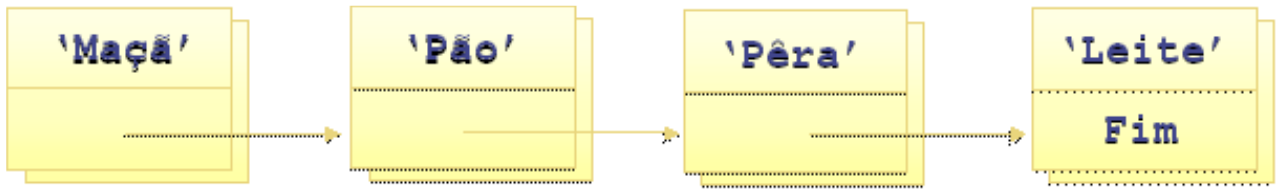
# Lista Encadeada (Dinâmica)

- Discussão intuitiva:
  - Por exemplo, uma operação de inserção pode ser feita da seguinte maneira:



# Lista Encadeada (Dinâmica)

- Discussão intuitiva:
  - Por exemplo, uma operação de inserção pode ser feita da seguinte maneira:



# Lista Encadeada (Dinâmica)

- TAD Lista Encadeada:

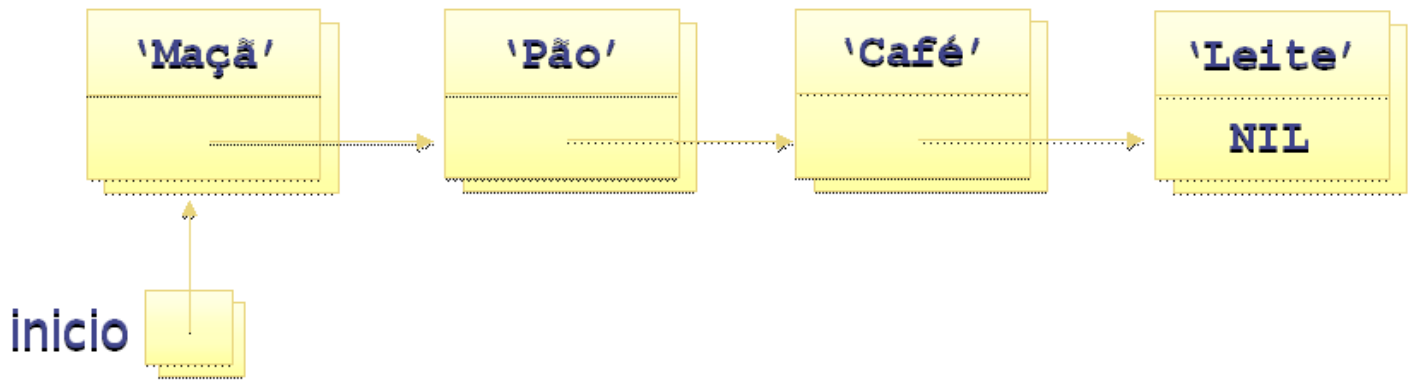
```
typedef struct elemento{
    int valor;
    struct elemento *proximo;
}Elemento;


typedef struct lista{
    Elemento *inicio;
}Lista;
```

# Lista Encadeada (Dinâmica)

- TAD Lista Encadeada – Principais operações:
  - Criar lista;
  - Limpar lista;
  - Inserir item (útlima posição);
  - Inserir item (por posição);
  - Remover item (última posição);
  - Remover item (por posição);
  - Recuperar item (dada uma chave);
  - Recuperar item (por posição);
  - Contar número de itens;
  - Verificar se lista está vazia;
  - Verificar se a lista está cheia;
  - Imprimir lista;

# Lista Encadeada (Dinâmica)

- TAD Lista Encadeada:
  - Antes de começarmos, precisamos definir como a lista será representada;
  - Uma forma bastante comum é manter uma variável ponteiro para o primeiro elemento da lista encadeada;



# Lista Encadeada (Dinâmica)

- TAD Lista Encadeada:
  - Convenciona-se que essa variável ponteiro deve ter valor NULL quando a lista estiver vazia;
  - Portanto, essa deve ser a iniciação da lista e também a forma de se verificar se ela se encontra vazia;

# Lista Encadeada (Dinâmica)

- TAD Lista Encadeada:
  - Outro detalhe importante é quanto as posições:
    - Na implementação com vetores, uma posição é um valor inteiro entre 0 e o campo fim;
    - Com listas encadeadas, uma posição passa ser um ponteiro que aponta um determinado nó da lista;
  - Vamos analisar cada uma das operações do TAD Lista:

# Lista Encadeada (Dinâmica)

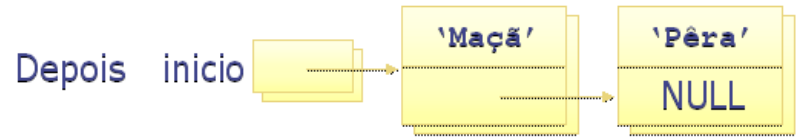
- Criar Lista:
  - **Pré-condição:** nenhuma;
  - **Pós-condição:** inicia a estrutura de dados;


- Limpar Lista:
  - **Pré-condição:** nenhuma;
  - **Pós-condição:** coloca a estrutura de dados no estado inicial;

# Lista Encadeada (Dinâmica)

- Inserir item (última posição):


  - **Pré-condição:** existência de memória disponível;
  - **Pós-condição:** insere um item na última posição, retorna true se a operação foi executada com sucesso, false caso contrário;




- Inserir item (por posição):

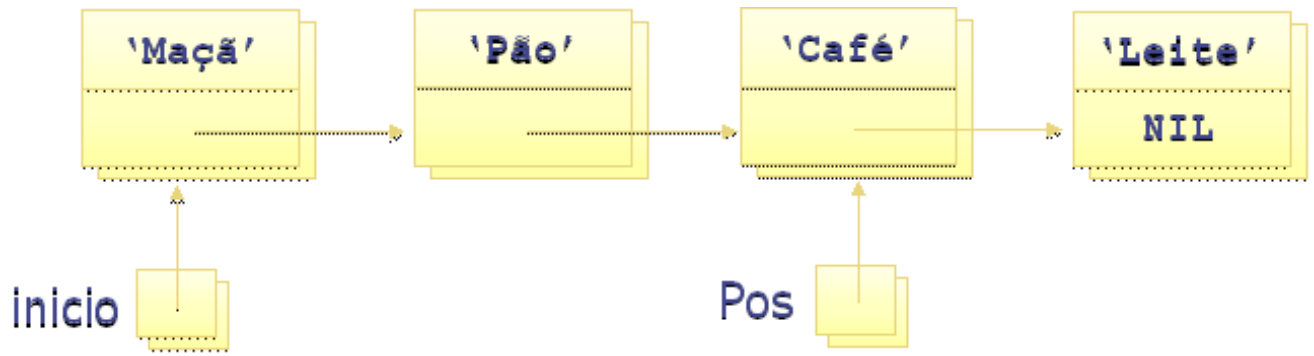
  - **Pré-condição:** existência de memória disponível;
  - **Pós-condição:** uma posição válida é passada e insere um item na posição indicada, retorna true se a operação foi executada com sucesso, false caso contrário;

# Lista Encadeada (Dinâmica)

- Remover item (última posição):
  - **Pré-condição:** a lista não está vazia;
  - **Pós-condição:** o último item da lista é removido, retorna true se a operação foi executada com sucesso;
- Remover item (por posição):
  - **Pré-condição:** uma posição válida é passada e a lista não está vazia;
  - **Pós-condição:** o último item da lista é removido, retorna true se a operação foi executada com sucesso;

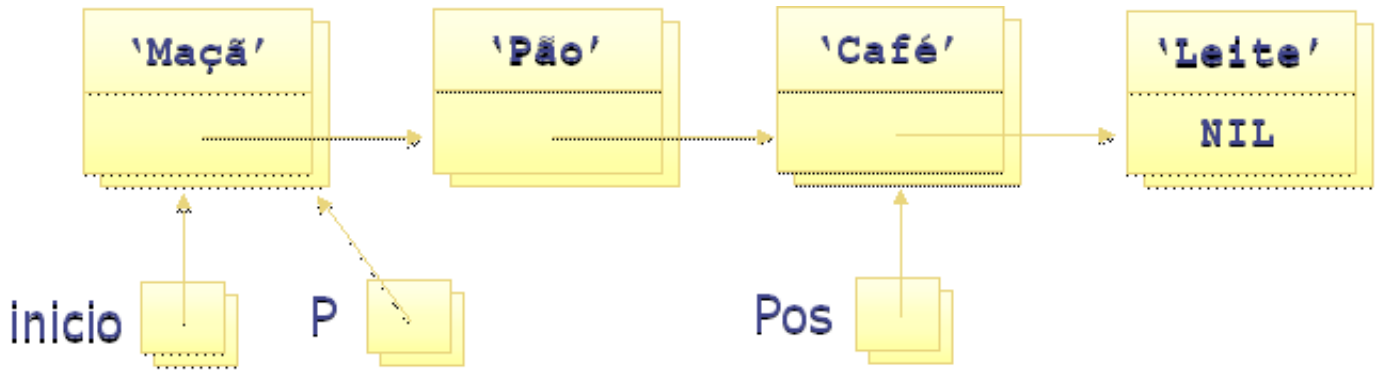
# Lista Encadeada (Dinâmica)

- Remover item (por posição):
  - **Pré-condição:** uma posição válida é passada e a lista não está vazia;
  - **Pós-condição:** o último item da lista é removido, retorna true se a operação foi executada com sucesso;



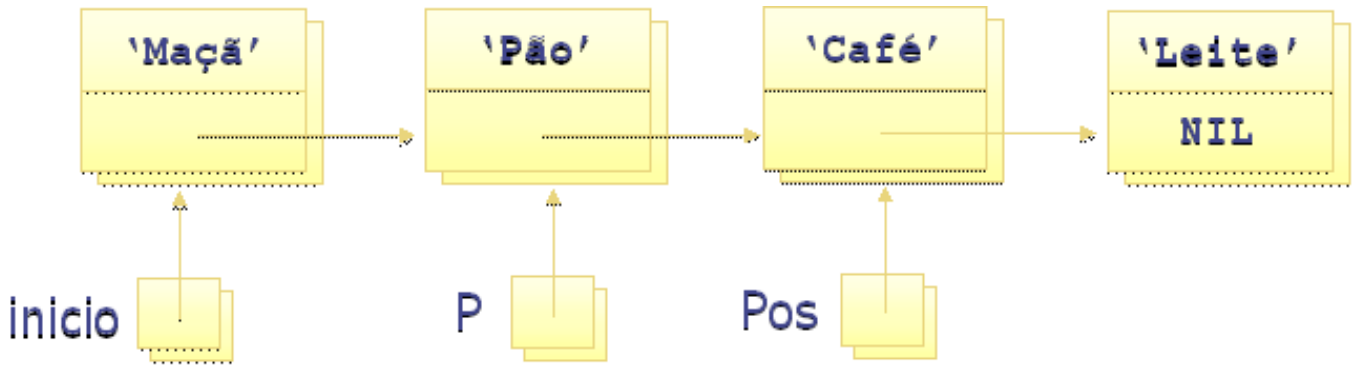
# Lista Encadeada (Dinâmica)

- Remover item (por posição):
  - **Pré-condição:** uma posição válida é passada e a lista não está vazia;
  - **Pós-condição:** o último item da lista é removido, retorna true se a operação foi executada com sucesso;



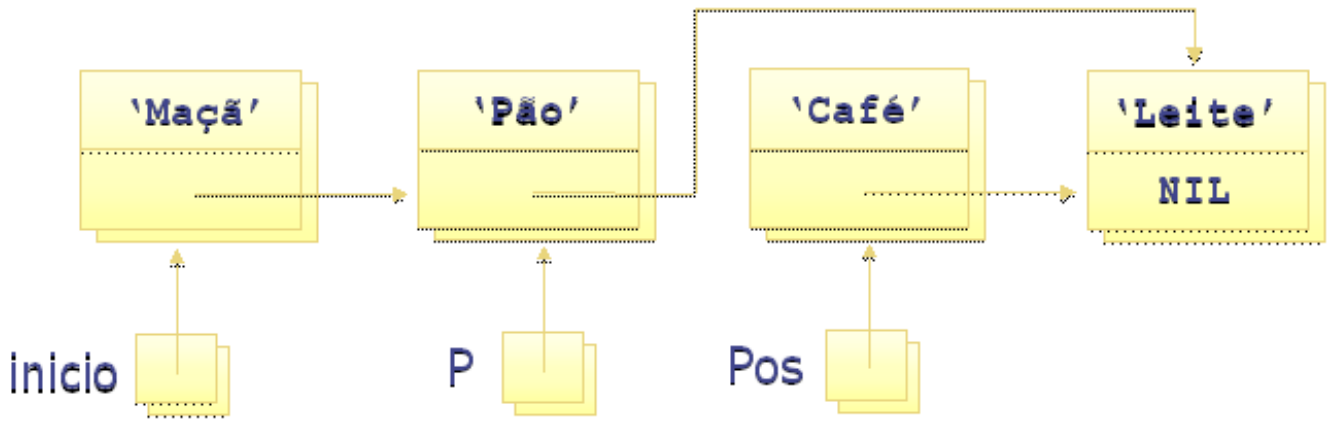
# Lista Encadeada (Dinâmica)

- Remover item (por posição):
  - **Pré-condição:** uma posição válida é passada e a lista não está vazia;
  - **Pós-condição:** o último item da lista é removido, retorna true se a operação foi executada com sucesso;



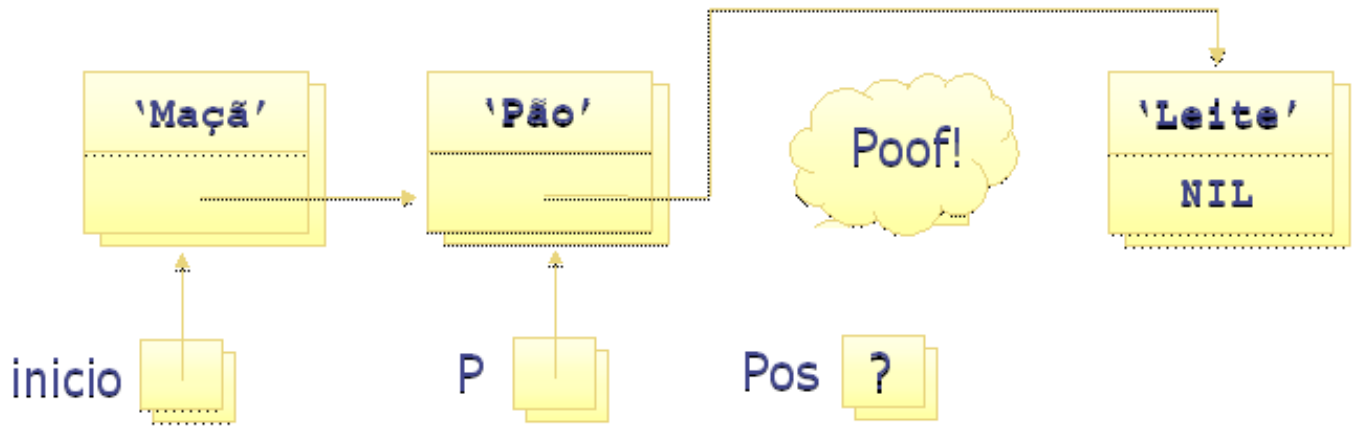
# Lista Encadeada (Dinâmica)

- Remover item (por posição):
  - **Pré-condição:** uma posição válida é passada e a lista não está vazia;
  - **Pós-condição:** o último item da lista é removido, retorna true se a operação foi executada com sucesso;



# Lista Encadeada (Dinâmica)

- Remover item (por posição):
  - **Pré-condição:** uma posição válida é passada e a lista não está vazia;
  - **Pós-condição:** o último item da lista é removido, retorna true se a operação foi executada com sucesso;

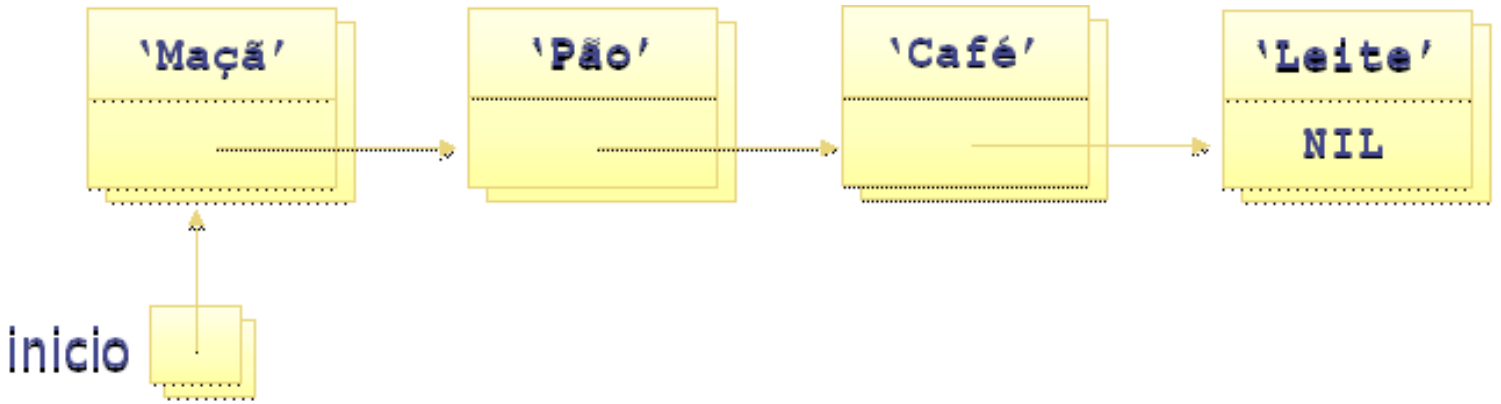


# Lista Encadeada (Dinâmica)

- Recuperar item (dada uma chave):
  - **Pré-condição:** nenhuma;
  - **Pós-condição:** verifica se a chave buscada está na lista, retorna true se a operação foi executada com sucesso, false caso contrário;
- Recuperar item (dada uma posição):
  - **Pré-condição:** nenhuma;
  - **Pós-condição:** recupera o item que se encontra na posição buscada, retorna true se a operação foi executada com sucesso, false caso contrário;

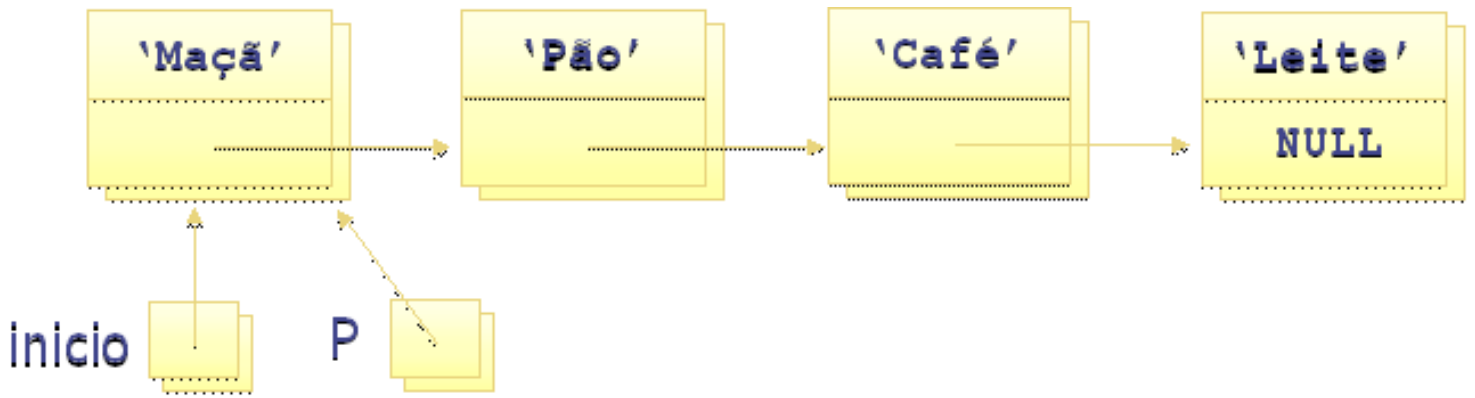
# Lista Encadeada (Dinâmica)

- Recuperar item (dada uma chave):
  - **Pré-condição:** nenhuma;
  - **Pós-condição:** verifica se a chave buscada está na lista, retorna true se a operação foi executada com sucesso, false caso contrário;



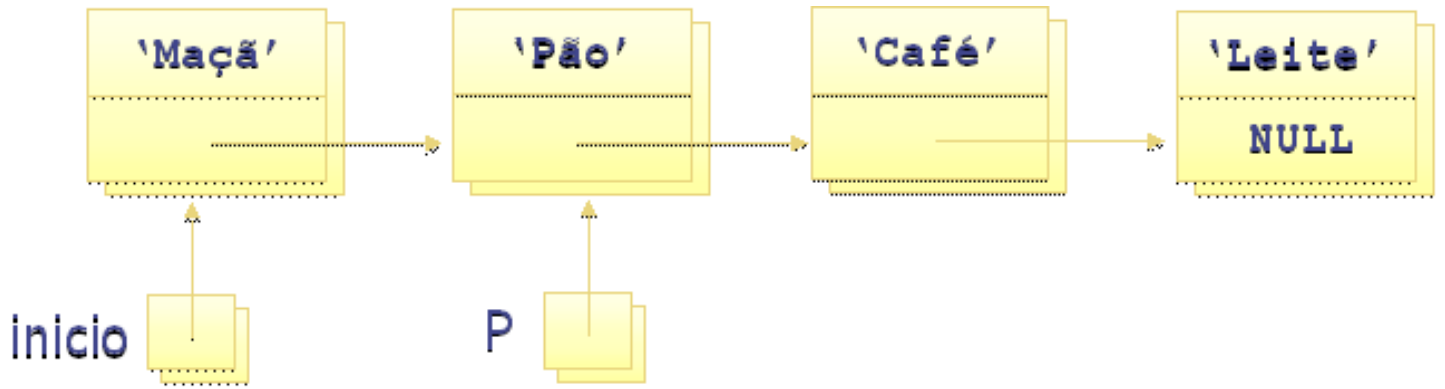
# Lista Encadeada (Dinâmica)

- Recuperar item (dada uma chave):
  - **Pré-condição:** nenhuma;
  - **Pós-condição:** verifica se a chave buscada está na lista, retorna true se a operação foi executada com sucesso, false caso contrário;



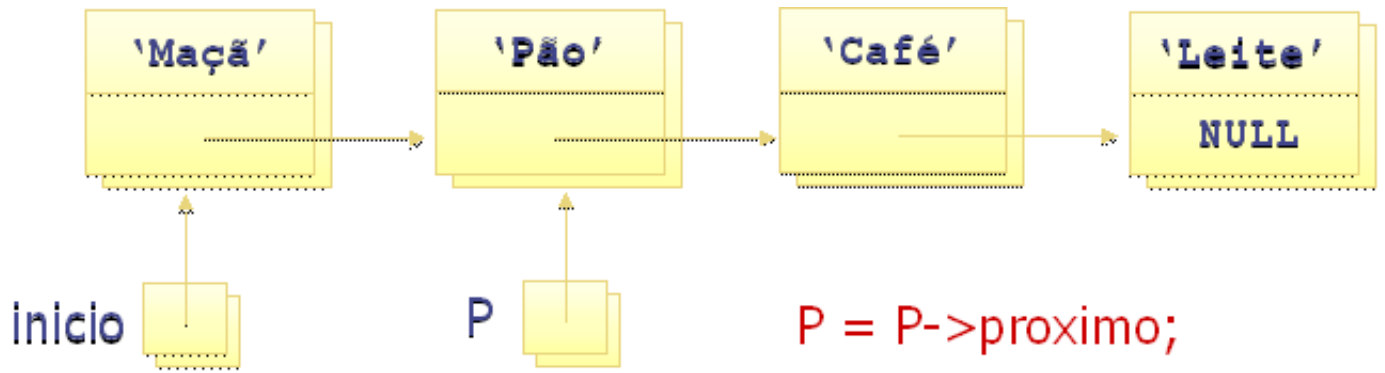
# Lista Encadeada (Dinâmica)

- Recuperar item (dada uma chave):
  - **Pré-condição:** nenhuma;
  - **Pós-condição:** verifica se a chave buscada está na lista, retorna true se a operação foi executada com sucesso, false caso contrário;



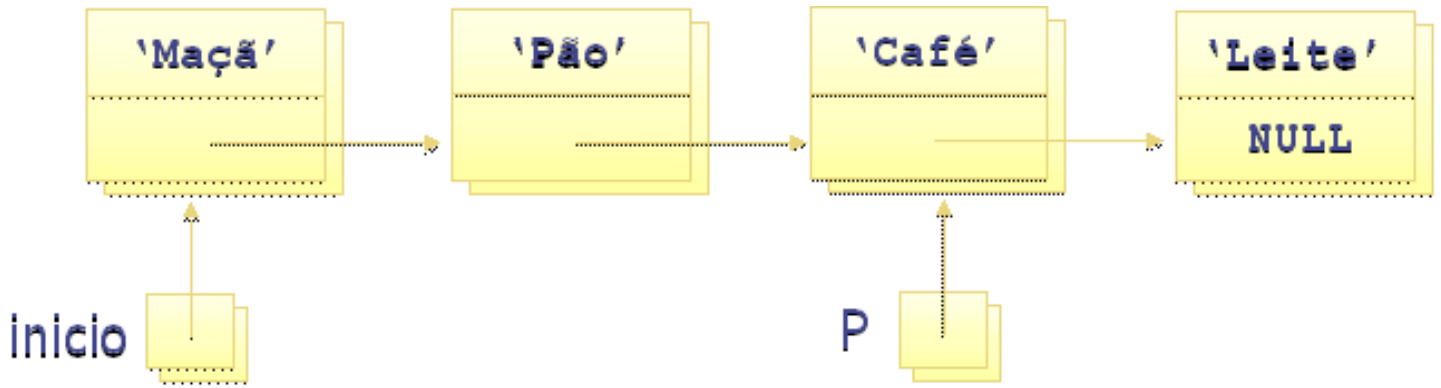
# Lista Encadeada (Dinâmica)

- Recuperar item (dada uma chave):
  - **Pré-condição:** nenhuma;
  - **Pós-condição:** verifica se a chave buscada está na lista, retorna true se a operação foi executada com sucesso, false caso contrário;



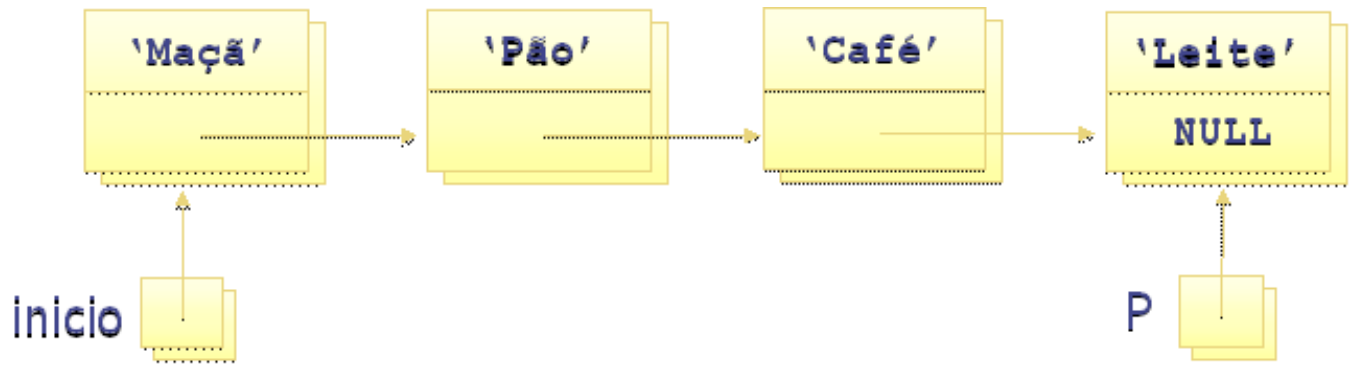
# Lista Encadeada (Dinâmica)

- Recuperar item (dada uma chave):
  - **Pré-condição:** nenhuma;
  - **Pós-condição:** verifica se a chave buscada está na lista, retorna true se a operação foi executada com sucesso, false caso contrário;



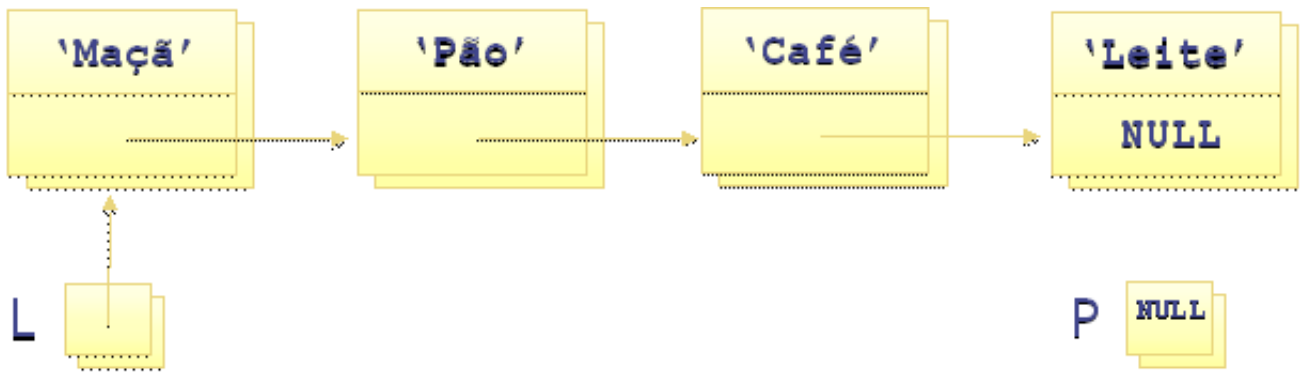
# Lista Encadeada (Dinâmica)

- Recuperar item (dada uma chave):
  - **Pré-condição:** nenhuma;
  - **Pós-condição:** verifica se a chave buscada está na lista, retorna true se a operação foi executada com sucesso, false caso contrário;



# Lista Encadeada (Dinâmica)

- Recuperar item (dada uma chave):
  - **Pré-condição:** nenhuma;
  - **Pós-condição:** verifica se a chave buscada está na lista, retorna true se a operação foi executada com sucesso, false caso contrário;



# Lista Encadeada (Dinâmica)

- Contar número de itens:
  - **Pré-condição:** nenhuma;
  - **Pós-condição:** retorna o número de itens da lista;
- Verificar se a lista está vazia:
  - **Pré-condição:** nenhuma;
  - **Pós-condição:** retorna true se a lista estiver vazia e false caso-contrário;

# Lista Encadeada (Dinâmica)

- Verificar se a lista está cheia:
  - **Pré-condição:** nenhuma;
  - **Pós-condição:** retorna true se a lista estiver cheia e false caso-contrário;
- Imprime lista:
  - **Pré-condição:** nenhuma;
  - **Pós-condição:** imprime na tela os itens da lista;

# Algoritmos e Estruturas de Dados II

- Bibliografia:
  - Básica:
    - CORMEN, Thomas, RIVEST, Ronald, STEIN, Clifford, LEISERSON, Charles. Algoritmos. Rio de Janeiro: Elsevier, 2002.
    - EDELWEISS, Nina, GALANTE, Renata. Estruturas de dados. Porto Alegre: Bookman. 2009. (Série livros didáticos informática UFRGS,18).
    - ZIVIANI, Nívio. Projeto de algoritmos com implementação em Pascal e C. São Paulo: Cengage Learning, 2010.
  - Complementar:
    - ASCENCIO, Ana C. G. Estrutura de dados. São Paulo: Pearson, 2011. ISBN: 9788576058816.
    - PINTO, W.S. Introdução ao desenvolvimento de algoritmos e estrutura de dados. São Paulo: Érica, 1990.
    - PREISS, Bruno. Estruturas de dados e algoritmos. Rio de Janeiro: Campus, 2000.
    - TENEMBAUM. Aaron M. Estruturas de dados usando C. São Paulo: Makron Books. 1995. 884 p. ISBN: 8534603480.
    - VELOSO, Paulo A. S. Complexidade de algoritmos: análise, projeto e métodos. Porto Alegre, RS: Sagra Luzzatto, 2001

# Algoritmos e Estruturas de Dados II