Listas encadeadas

Prof. Leandro O. Freitas
leandro@politecnico.ufsm.br

Listas encadeadas

Sistemas para Internet
UFSM

- Pré-requisitos:
  - Registros;
  - Ponteiros;
  - Alocação dinâmica de memória.

Lista encadeada

Sistemas para Internet
UFSM

- Refere-se a uma estrutura de dados mais dinâmica que os vetores pois a alocação/liberação de memória é realizada conforme a criação ou retirada de elementos.
  - Assim, não existe uma quantidade predefinida de memória para o total de elementos. Não é necessário saber o total de elementos.

- Composta de ***nós (ou nodos)*** que guardam informações sobre cada elemento.

- Define-se cada elemento como um registro que possui:
  - campos de informações;
  - ponteiro para o próximo elemento da lista.

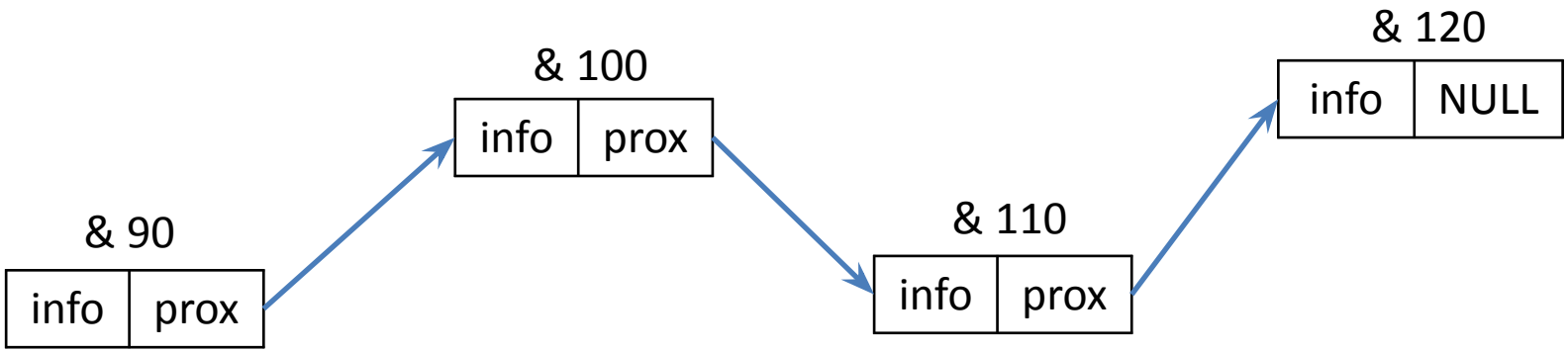
& 90

info (int, float, char, struct, etc)	prox (endereço de memória do próximo nó)

Lista encadeada

Sistemas para Internet
UFSM

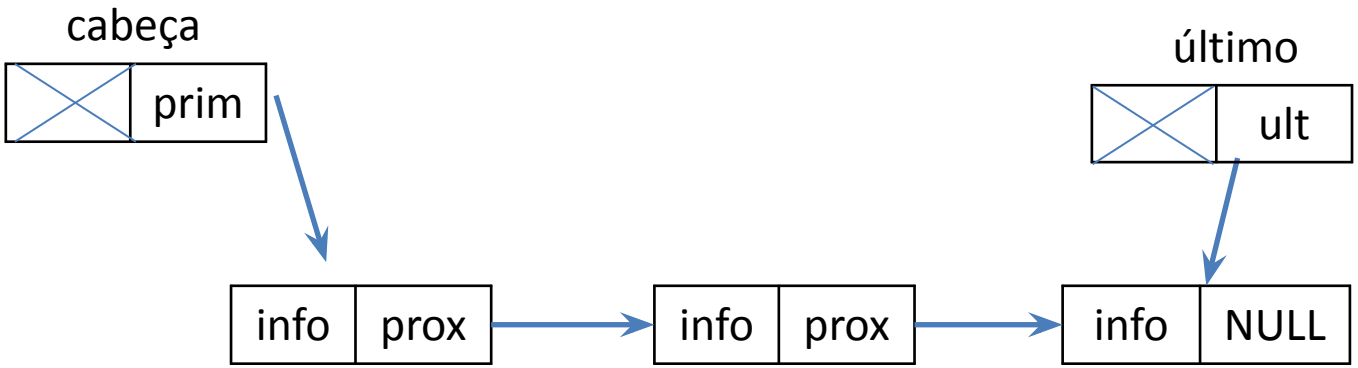
- Características:
  - Tamanho da lista não é pré-definido;
  - Espaço utilizado em memória é proporcional ao número de elementos;
  - Elementos ocupam posições não contíguas na memória;
  - O que os caracteriza como um conjunto de elementos (lista)?
    - Cada elemento guarda endereço de memória do próximo.



Lista encadeada

Sistemas para Internet
UFSM

- Deve ter uma célula cabeça:
  - Não guarda informações, apenas o endereço de memória do primeiro nó;

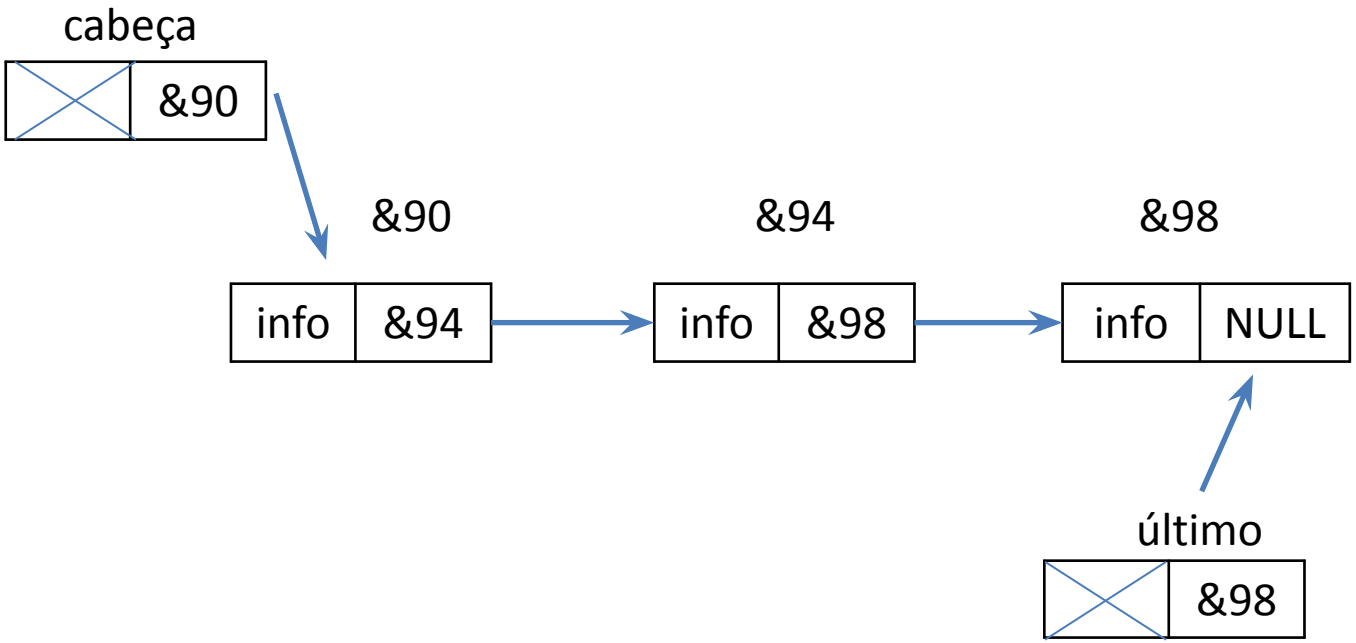


- Pode ter um apontador para o último elemento.

Lista encadeada

Sistemas para Internet
UFSM

- Exemplo:



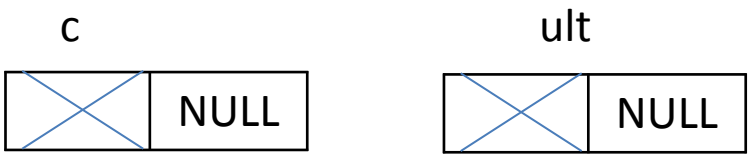
- Acesso aos elementos **não** é feito de forma independente!
- Para acessar o elemento alocado no endereço &98, deve-se passar pelos seus antecessores. Logo, deve-se sempre saber quem está no início da lista.

Lista vazia

Sistemas para Internet
UFSM

- Quando não existem elementos na lista o nó cabeça possui um valor nulo (NULL);
- Ponteiro do último elemento também possui valor nulo (NULL).

```
c = NULL;
ult = NULL;
```

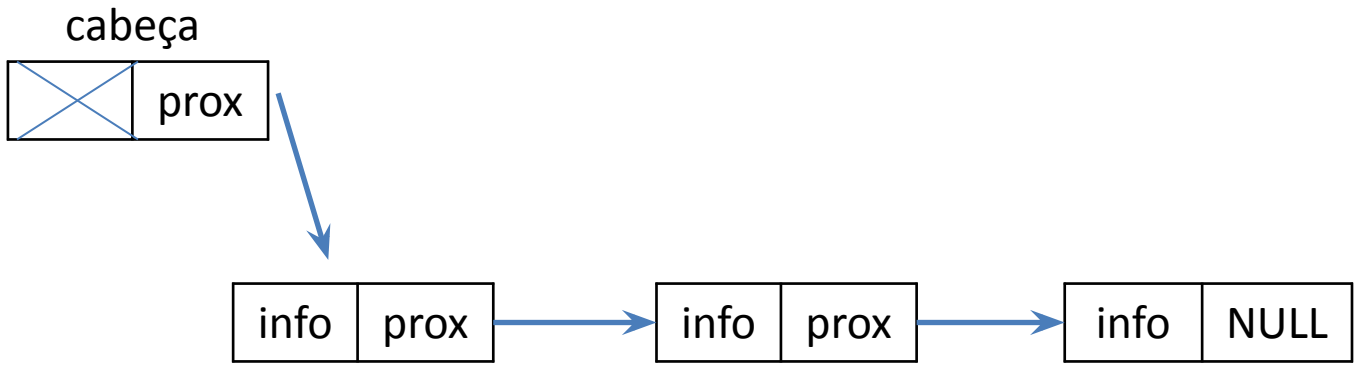


- Principais manipulações:
  - Inserção de novos elementos;
  - Remoção de elementos existente;
  - Acesso e manipulação de valores dos nós.

Inserir elementos na lista

Sistemas para Internet
UFSM

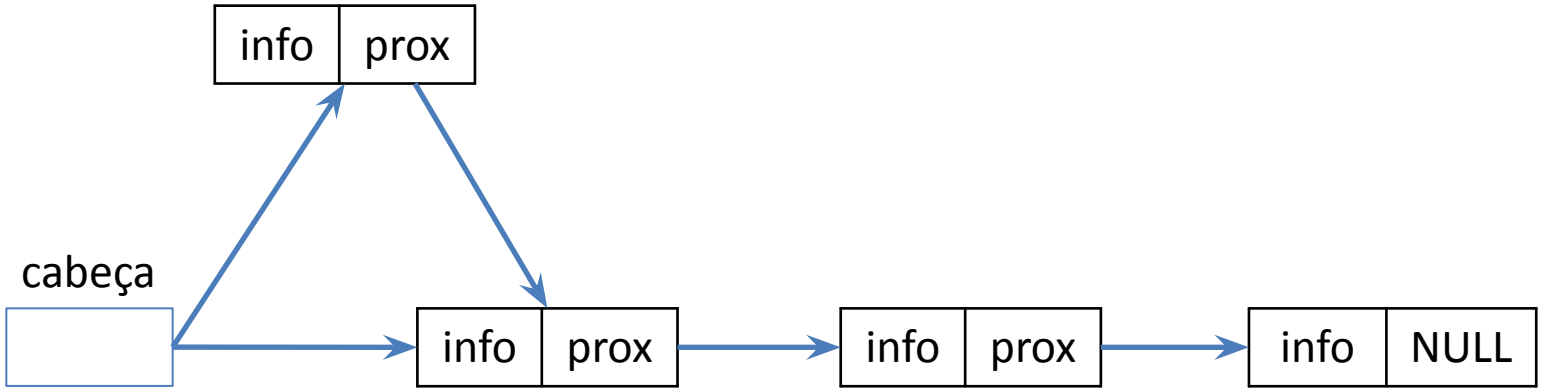
- 3 opções de inserção de elementos:
  - Início da lista: 1ª posição (posição mais à esquerda na imagem abaixo);
  - Final da lista: última posição (posição mais à direita na imagem);
  - Meio da lista: entre dois elementos quaisquer.



Inserção na 1ª posição

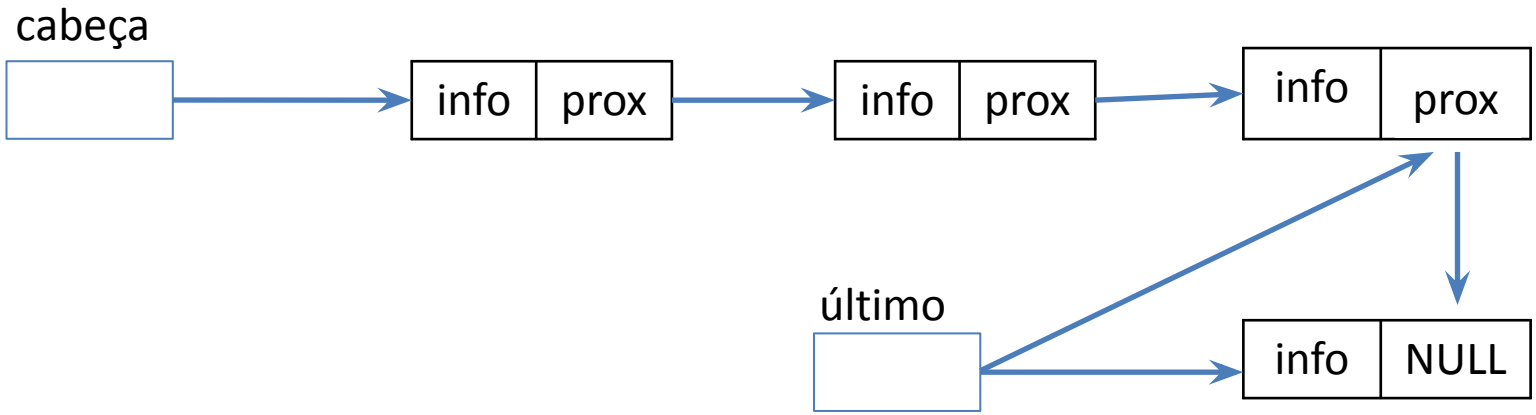
Sistemas para Internet
UFSM

- Nó cabeça SEMPRE deve apontar para o primeiro elemento!



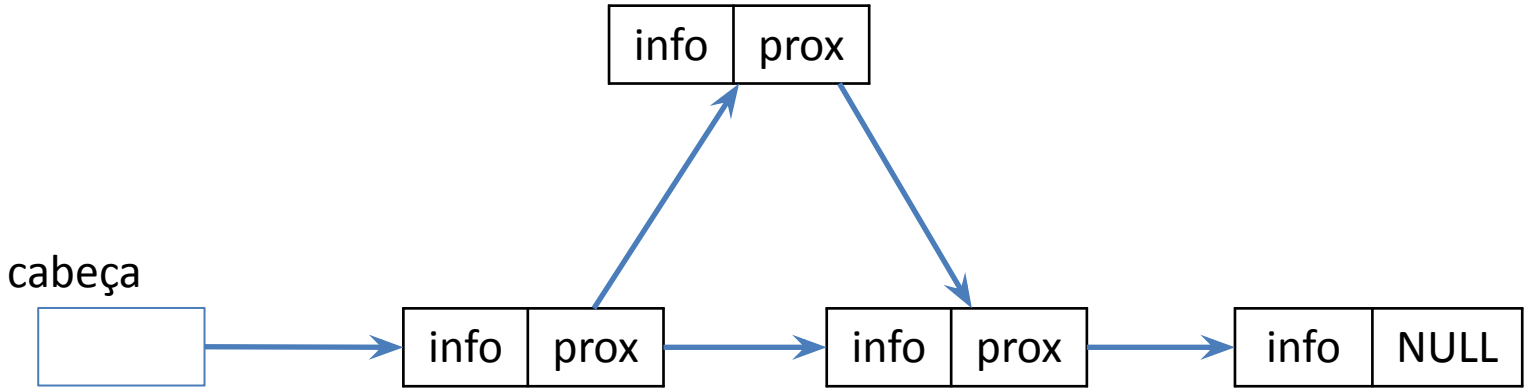
Inserção na última posição

Sistemas para Internet
UFSM



Inserção em posição específica

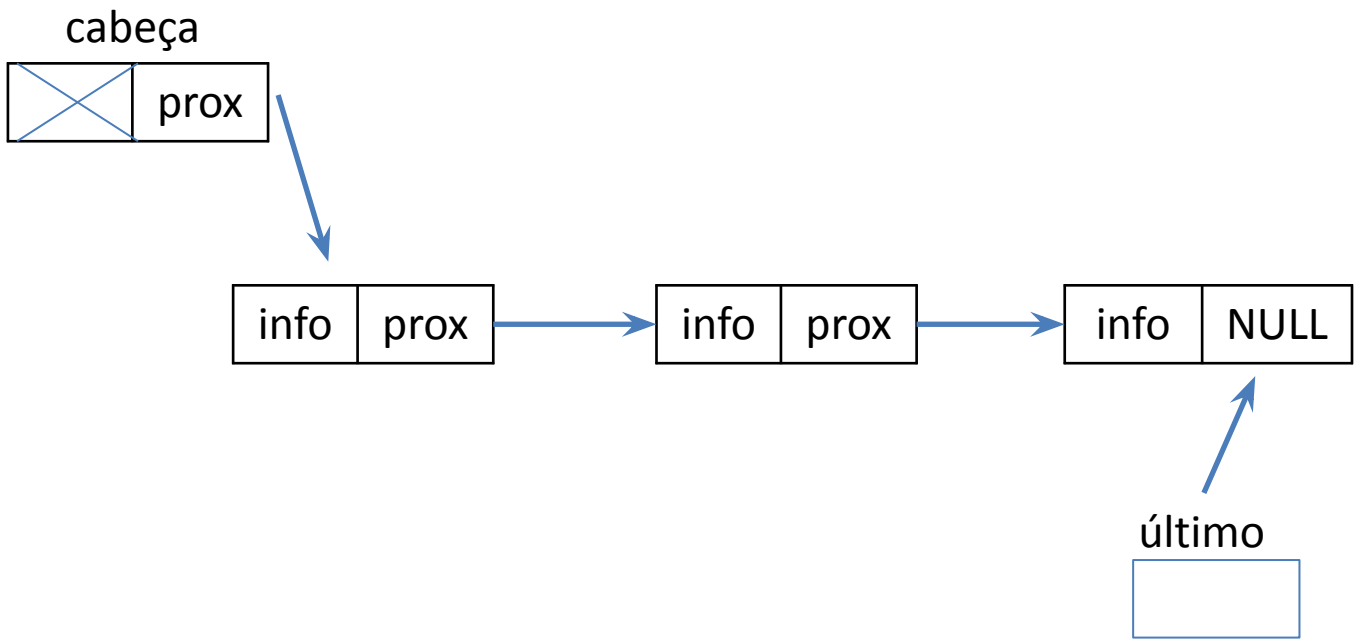
Sistemas para Internet
UFSM



Retirar elemento da lista

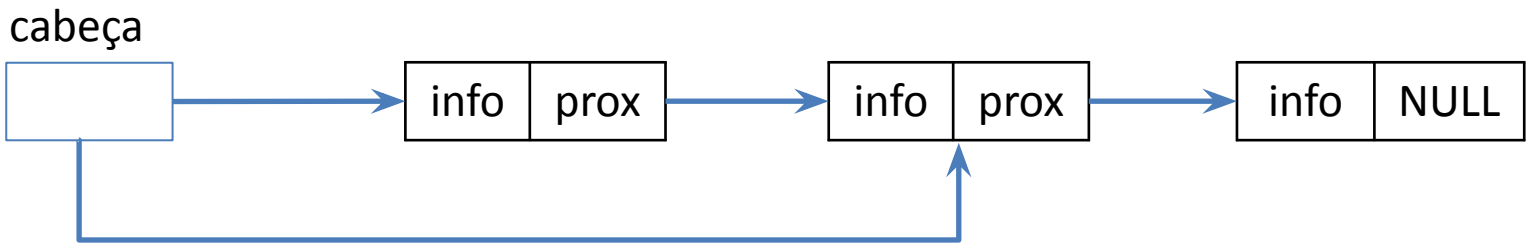
Sistemas para Internet
UFSM

- 3 opções para retirada de elementos:
  - 1ª. posição;
  - Última posição;
  - Posição específica.



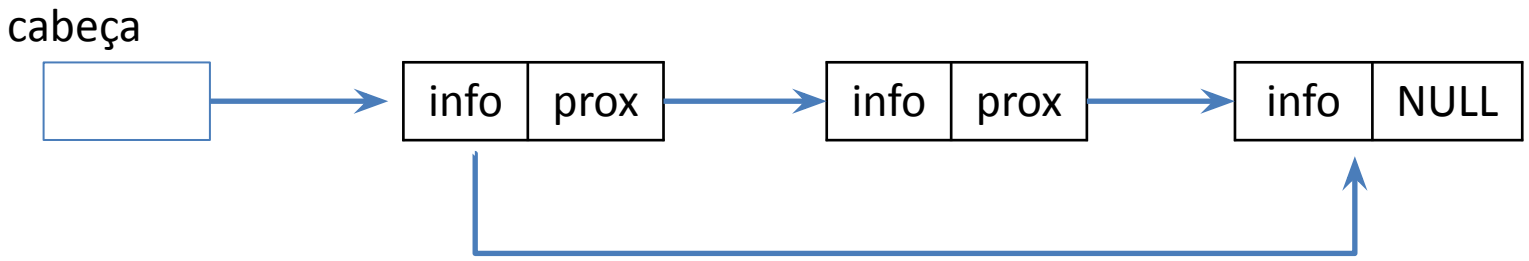
Retirar o 1º elemento da lista

Sistemas para Internet
UFSM



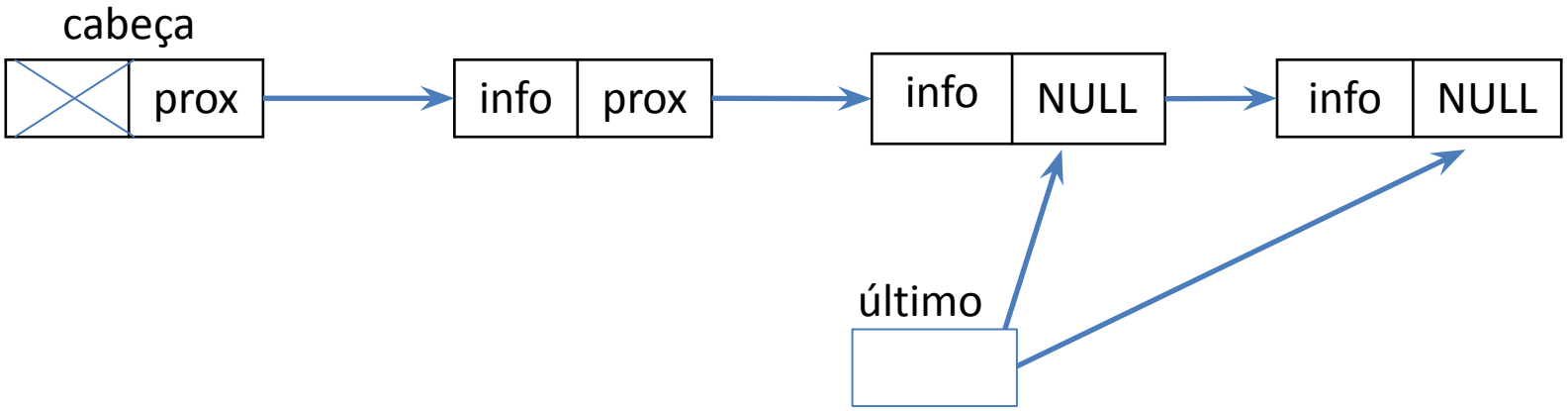
Retirar elemento específico

Sistemas para Internet
UFSM



Retirar último elemento

Sistemas para Internet
UFSM



Estrutura da lista encadeada

Sistemas para Internet
UFSM

```c
struct lista {
    int info;
    struct lista *prox;
};

typedef struct lista Lista;

// FUNÇÃO PARA INSERIR NO INÍCIO
Lista *insere(Lista *l, int i) {
    Lista *novo;
    novo = (Lista *) malloc (sizeof(Lista));
    novo->info = i;
    novo->prox = l;
    return novo;
}
```

Lista encadeada deve iniciar vazia!!

```c
int main() {
    Lista *l;
    l = NULL;
A   l = insere(l, 43);
B   l = insere(l, 75);
C   l = insere(l, 30);
D   l = insere(l, 12);
    imprime(l);
    l = retira(l, 75);
    printf("\n");
    imprime(l);

    return 0;
}
```

Símbolo ->: Para acessar os campos da struct

Operações sobre listas encadeadas

- Inserir no início da lista e imprimir:

```
19  // FUNÇÃO PARA INSERIR NO INÍCIO
20  Lista *insere(Lista *l, int i) {
21      Lista *novo;
22      novo = (Lista *) malloc (sizeof(Lista));
23      novo->info = i;
24      novo->prox = l;
25      return novo;
26  }
27
28  //FUNÇÃO PARA IMPRIMIR ELEMENTOS DA LISTA
29  void imprime (Lista *l)
30  {
31      Lista *p;
32      for (p=l; p!=NULL; p=p->prox)
33          printf("%d ", p->info);
34      }
```

```
61  int main() {
62      Lista *l;
63      l = NULL;
64      l = insere(l, 43);
65      l = insere(l, 75);
66      l = insere(l, 30);
67      l = insere(l, 12);
68      imprime(l);
69      l = retira(l, 75);
70      printf("\n");
71      imprime(l);
72
73      return 0;
74  }
```

Operações sobre listas encadeadas: Retirar elementos

Sistemas para Internet
UFSM

```c
//FUNÇÃO PARA RETIRAR UM VALOR DA LISTA
Lista *retira(Lista *l, int v) {
    Lista *ant = NULL;
    Lista *p = l;

    while (p != NULL && p->info != v) {
        ant = p;
        p = p->prox;
    }

    if (p == NULL){
        return l;
    }

    if (ant == NULL) {
        l = p->prox;
    }
    else {
        ant->prox = p->prox;
    }

    free(p);
    return l;
}
```

```c
int main() {
    Lista *l;
    l = NULL;
    l = insere(l, 43);
    l = insere(l, 75);
    l = insere(l, 30);
    l = insere(l, 12);
    imprime(l);
    l = retira(l, 75);
    printf("\n");
    imprime(l);

    return 0;
}
```

Realize um teste de mesa nesta função

Operações sobre listas encadeadas

- **Vantagens:**
  - Permite inserir ou retirar itens do meio da lista a um custo constante (importante quando a lista tem de ser mantida em ordem).
  - Bom para aplicações em que não existe previsão sobre o crescimento da lista.

- **Desvantagem:**
  - Utilização de memória extra para armazenar os apontadores.

Exercícios

1. Descreva textualmente as etapas para criação de uma lista vazia e a inserção de elementos nela;

2. Desenvolva um algoritmo para criar uma lista encadeada vazia. Depois crie uma função que permita que o inserir elementos nela. Por último, apresente os valores pares da lista;

3. Escreva uma função que receba por parâmetro um ponteiro para o primeiro elemento da lista e retira um elemento do início, do meio e do final da lista. Depois apagar da memória o elemento retirado;

4. Escreva uma função que receba por parâmetro um valor e um ponteiro para o primeiro elemento da lista e insere um nó na última posição;

Exercícios 2

Sistemas para Internet
UFSM

1. Faça um algoritmo com uma função que crie uma cópia de uma lista encadeada. Ou seja, uma nova lista de mesmo tamanho com o conjunto de valores na mesma ordem da primeira;

2. Faça um algoritmo com uma função para verificar se duas listas encadeadas são iguais. Duas listas são consideradas iguais se possuem a mesma sequência de valores;

3. Faça um algoritmo para criar uma lista encadeada e inserir elementos (nós) nela. Uma particularidade que esta lista deve ter é que o último elemento nunca aponta para NULL, mas sim para o primeiro elemento.
   a. Pesquise sobre listas circulares.