Estruturas de dados

Sistemas para Internet
UFSM

Listas duplamente encadeadas

Leandro O. Freitas
leandro@politecnico.ufsm.br

Retomando...

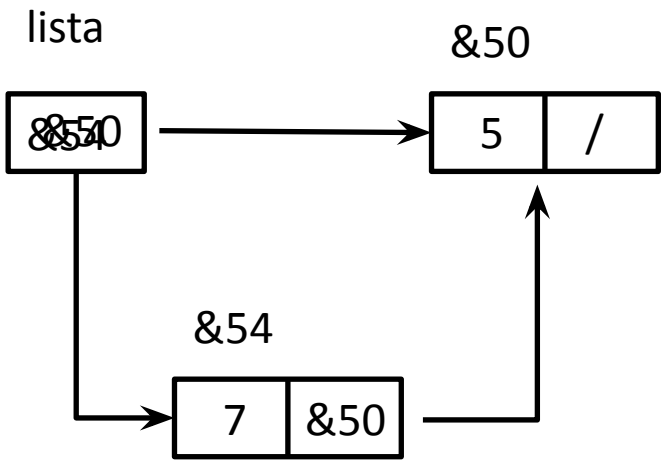
Sistemas para Internet
UFSM

- Listas encadeadas:
  - Permitem organização de dados em endereços de memória não sequenciais.
  - Cada elemento possui um ponteiro para o próximo elemento da lista.
  - Vantagem: não precisa de espaço de memória pré-definido, como em vetores.
  - Problemas:
    - Não permite percorrer elementos em ordem inversa;
    - Dificulta a retirada de elementos. Mesmo sabendo o endereço do elemento que será retirado, precisamos saber também o endereço do anterior, para apontá-lo para o próximo do que será removido.

Retomando...

Sistemas para Internet
UFSM

- Listas encadeadas:

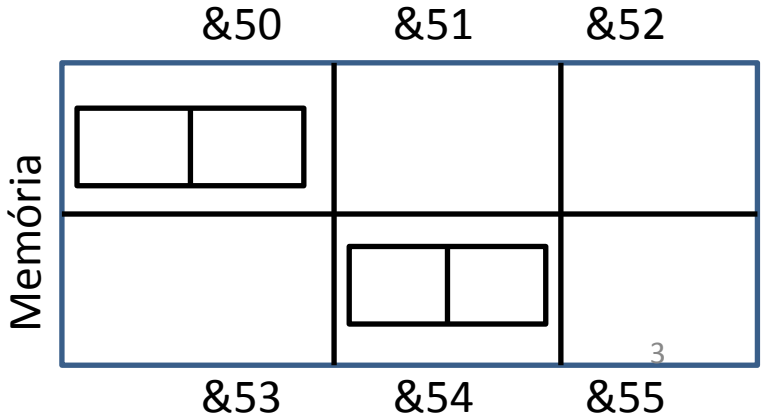



```
Lista *novo = (Lista*) malloc (sizeof(Lista));
novo->info = 5;
novo->prox = lista;
return novo;
```

```
Lista *novo = (Lista*) malloc (sizeof(Lista));
novo->info = 7;
novo->prox = lista;
return novo;
```

lista = ~~NULL~~ ~~&50~~ &54

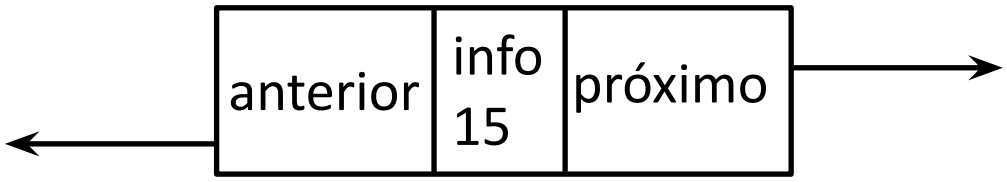
```
lista = insere (lista);
lista = insere (lista);
```



Listas duplamente encadeadas

Sistemas para Internet
UFSM

- Cada elemento tem um ponteiro para o próximo e outro ponteiro para o elemento anterior.



- A partir de um elemento, podemos acessar seus adjacentes, o próximo e o anterior.

- O ponteiro para o elemento anterior permite percorrer a lista em ordem inversa, até chegar no primeiro elemento.

- Ponteiro anterior do primeiro elemento é NULL.

Lista duplamente encadeada

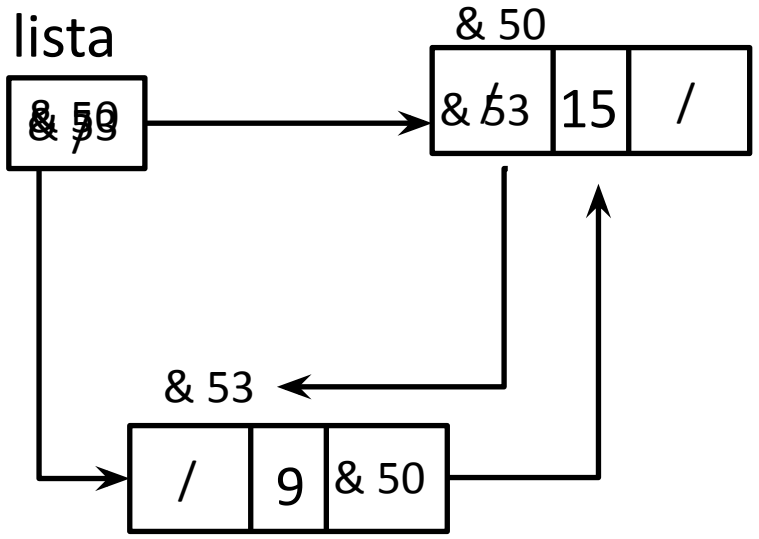
Sistemas para Internet
UFSM

```
struct dupla
{
   int info;
   struct dupla *ant;
   struct dupla *prox;
};
typedef struct dupla Lista;
```

```
main ()
{
   Lista *L;
   L = NULL;
   L = insere (L, 5);
}
```

Inserir elementos

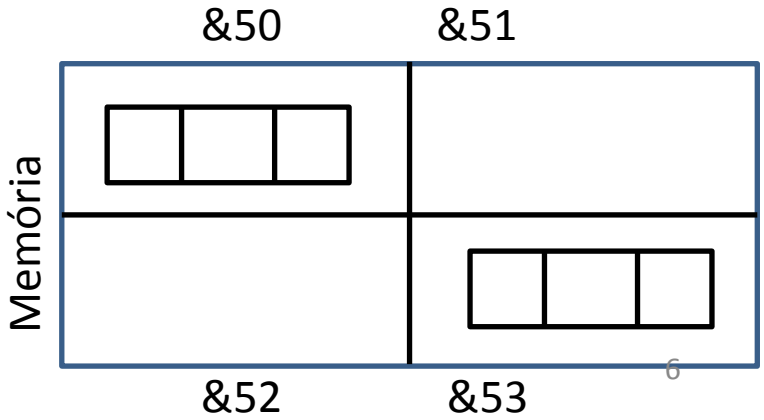
Sistemas para Internet
UFSM



1. Lista *novo = (Lista*) malloc (sizeof(Lista));
2. novo->info = x;
3. novo->prox = lista;
4. novo->ant = NULL;
5. if (lista != NULL)
6. lista ->ant = novo;
7. return novo;

lista = ~~NULL~~ ~~&50~~ &53

lista = insere (lista, 15);
lista = insere (lista, 9);



Busca de elementos

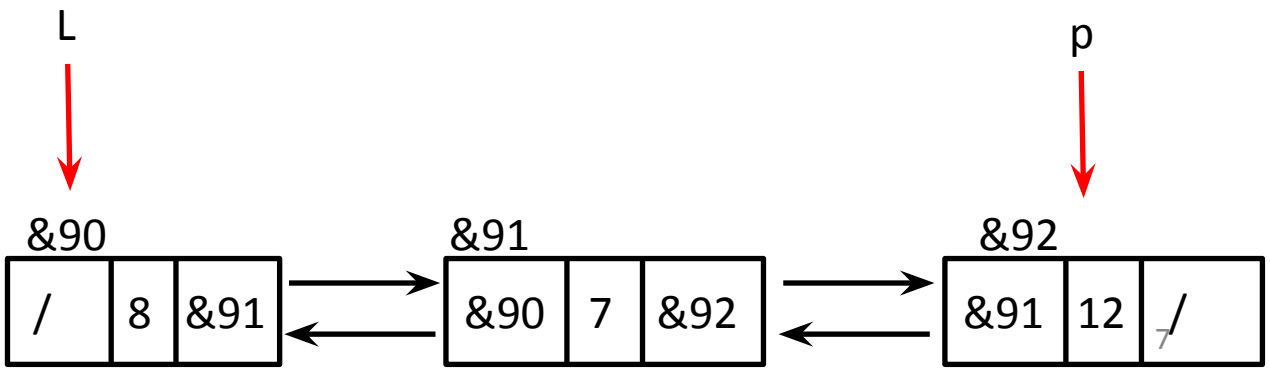
Sistemas para Internet
UFSM

- Função de busca recebe informações referentes ao elemento que procuramos e retorna o ponteiro para o nó onde ele está armazenado.
- Caso o elemento não esteja na lista o retorno é NULL.

```
1.  Lista *busca (Lista *L, int v) {
2.     Lista *p;
3.     for (p = L; p != NULL; p = p->prox) {
4.           if (p->info == v) {
5.                 return p;
6.           }
7.     }
8.     return NULL;
9.   }

main() {
   Lista *t;
   t = busca (L, 12);
}
```

L = &90
v = 12
p = ~~&90~~ ~~&91~~ &92
t = &92



Remover elementos

Sistemas para Internet
UFSM

- Mais complexa que a lista simplesmente encadeada pois é preciso ajustar os dois encadeamentos (anterior e próximo).
- É possível retirar um elemento conhecendo apenas o ponteiro para ele.

```
Lista2* retira (Lista2* L, int v) {
   Lista2* p;
   p = busca (L, v);
   if (p == NULL) {
      return L;
 }
```

Faça um teste de mesa neste trecho de código e insira os comentários que achar importante

L = &90
v = 7
p = &91

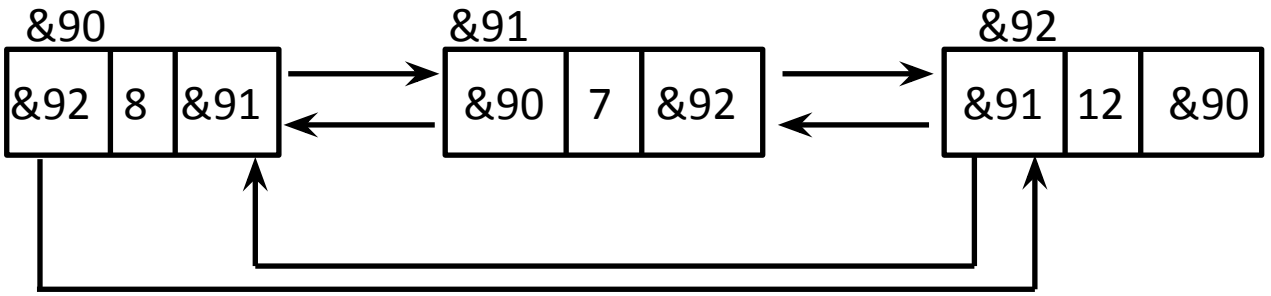
```
1.  if (L == p)
2.      L = p->prox;
3.  else
4.      p->ant->prox = p->prox;

5.  if (p->prox != NULL)
6.      p->prox->ant = p->ant;
7.  free(p);
8.  return L;
}
```

Lista duplamente encadeada

Sistemas para Internet
UFSM

- Lista circular:
  - O último elemento passa a ter como próximo o primeiro da lista.
  - O primeiro elemento da lista passa a ter como anterior o último.



Exercícios

- Desenvolva um algoritmo que permita inserir elementos na última posição de uma lista duplamente encadeada.
- Desenvolva um algoritmo que implemente uma lista circular duplamente encadeada.
- Desenvolva um algoritmo para permitir a inclusão ordenada de elementos na lista.
- Desenvolva um algoritmo para impedir inclusão de valores repetidos.
- Pesquise sobre aplicações de listas duplamente encadeadas