Listas

Giselle Lopes Ferrari Ronque
ferrari@eletrica.ufpr.br

Listas Lineares

- A lista linear é a estrutura que permite representar um conjunto de dados de forma a preservar a relação de ordem existente entre eles.
- Uma lista linear, ou tabela, é um conjunto de n>= 0 nós L[1], L[2], ..., L[n], onde:
  - Se n>0, L[1] é o primeiro nó,
  - Para 1 < k <= n, o nó L[k] é precedido por L[k-1].

Listas Lineares

- Observe que existe uma ordenação implícita: se n > 0 temos que:
  - L[1] é o primeiro nó da lista.
  - L[n] é o último nó da lista.
  - se 1 < k < n L[k] é precedido por (k-1) e seguido por (k+1).
  - se n = 0 dizemos que a lista é vazia.
- Seqüência de nós

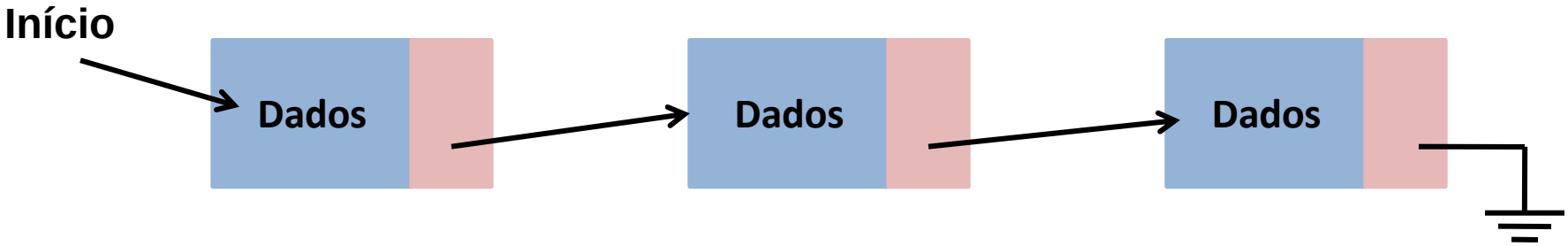
Tipo de Armazenamento

- O tipo de armazenamento de uma lista linear pode ser classificado de acordo com a posição relativa na memória de dois nós consecutivos na lista.
- Lista linear com **alocação estática** de memória
  - Também chamada de **Lista Seqüencial**
  - Nós em posições contínuas de memória
  - Geralmente representado por vetores
  - Útil para implementar filas e pilhas (variáveis para controlar fim e início)

Tipo de Armazenamento

- Lista linear com **alocação dinâmica** de memória
  - Também chamada de **Lista Encadeada**.
  - Posições de memória são alocadas a medida que são necessárias.
  - Nós encontram-se aleatoriamente dispostos na memória e são interligados por ponteiros, que indicam a próxima posição da tabela.
    - Nós precisam de um campo a mais: campo que indica o endereço do próximo nó.

Lista Simplesmente Encadeada




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

Lista Simplesmente Encadeada

- Início da lista
  - Endereço do primeiro nó definido:

```
struct lista L1, *ini;
L1.prox = NULL;
ini = &L1;
```

    - Ou:

```
struct lista *ini;
ini = malloc (sizeof (lista));
ini->prox = NULL;
```

Lista Simplesmente Encadeada

- Início da lista
  - Definição de lista vazia

```
lista *ini;
ini = NULL;
```

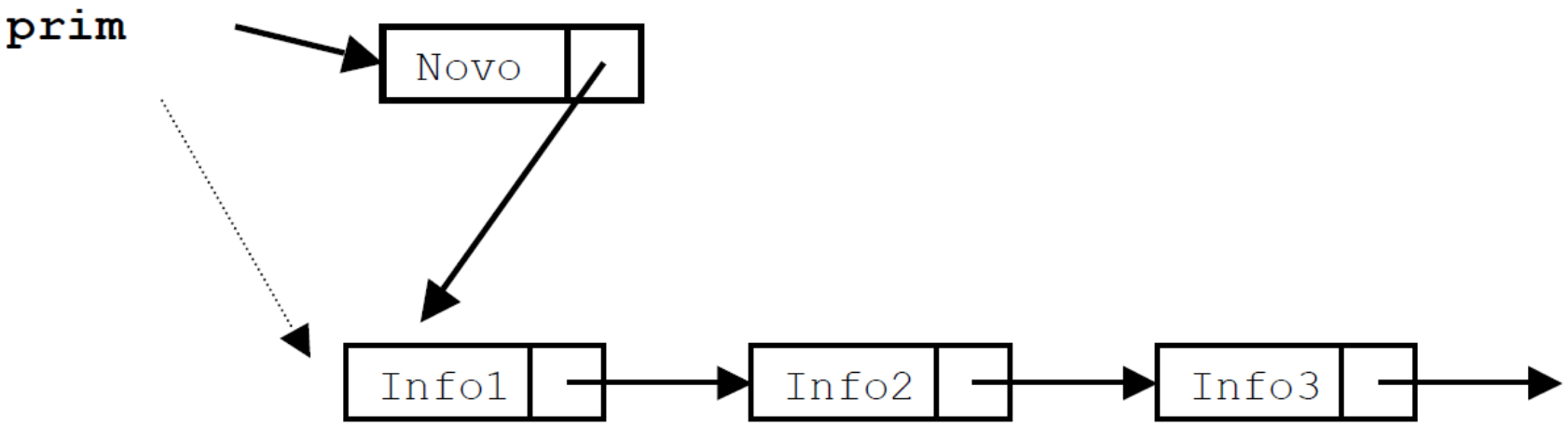
  - Função de inicialização → retorna uma lista vazia

```
struct lista * inicializa (void){
        return NULL;
}
```

Lista Simplesmente Encadeada

- Função de inserção no início da lista

```
struct lista * insere (struct lista * L, int i){
        struct lista *novo = (struct lista *) malloc (sizeof (struct lista);
        novo->info = i;
        novo->prox = L;
        return  novo;}
```



Lista Simplesmente Encadeada

- Exemplo:

```
int main (void){
        struct lista *l;            /* declara uma lista não inicializada */
        l = inicializa();           /* inicializa lista como vazia */
        l = insere(l, 23);          /* insere na lista o elemento 23 */
        l = insere(l, 45);          /* insere na lista o elemento 45 */
        ...
        return 0;
}
```

Lista Simplesmente Encadeada

- Função que percorre os nós de uma lista

```
/* função imprime: imprime valores dos elementos */
void imprime (struct lista *l) {
        struct lista *p;                /* variável auxiliar para percorrer a lista */
        for (p = l; p != NULL; p = p->prox)
                printf("info = %d\n", p->info);
}
```

Lista Simplesmente Encadeada

- Função que verifica se a lista está vazia

```
/* função vazia: retorna 1 se a lista está vazia ou 0 se não está vazia */
int vazia (struct lista *l) {
        if (l == NULL)
                return 1;
        else
                return 0;
}
```

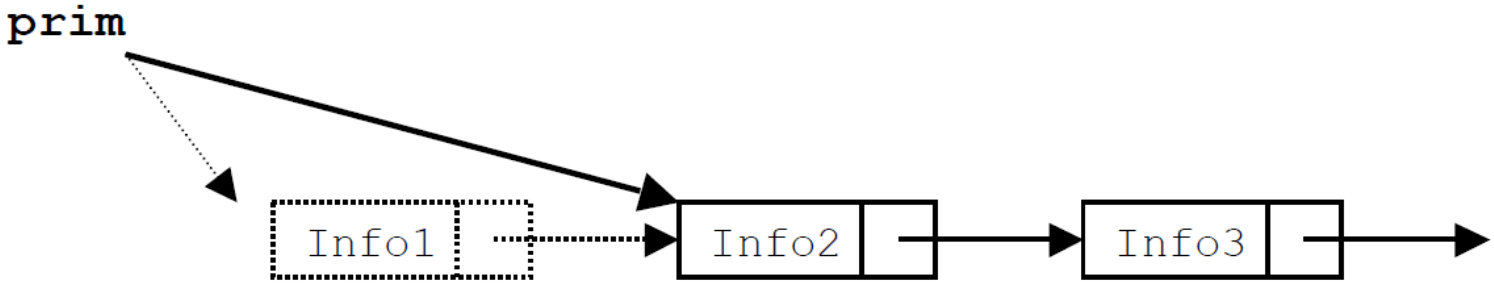
Lista Simplesmente Encadeada

- Função de busca

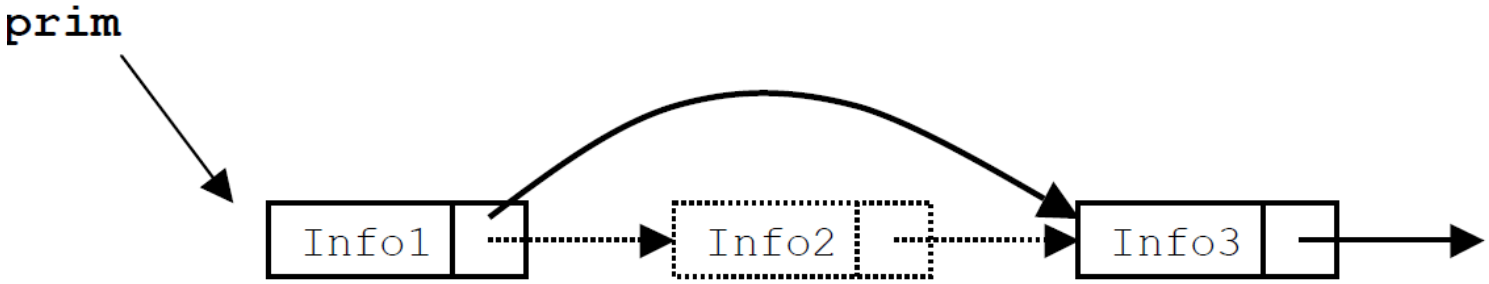
```
/* função busca: busca um elemento na lista */
struct lista * busca (struct lista * l, int v){
        struct lista * p;
        for (p=l; p!=NULL; p=p->prox)
                if (p->info == v)
                        return p;
        return NULL;                            /* não achou o elemento */
}
```

Lista Simplesmente Encadeada

- Função de remoção de um nó



Remoção do primeiro elemento da lista.



Remoção de um elemento no meio da lista.

Lista Simplesmente Encadeada

- Função de remoção de um nó

```
struct lista * retira (struct lista *l, int v) {
     struct lista *ant = NULL;          /* ponteiro para elemento anterior */
     struct lista *p = l;               /* ponteiro para percorrer a lista*/
    /* procura elemento na lista, guardando anterior */
    while (p != NULL && p->info != v) {
          ant = p;
          p = p->prox;}
    /* verifica se achou elemento */
    if (p == NULL)
          return l; /* não achou: retorna lista original */
```

Lista Simplesmente Encadeada

- Continuação ...

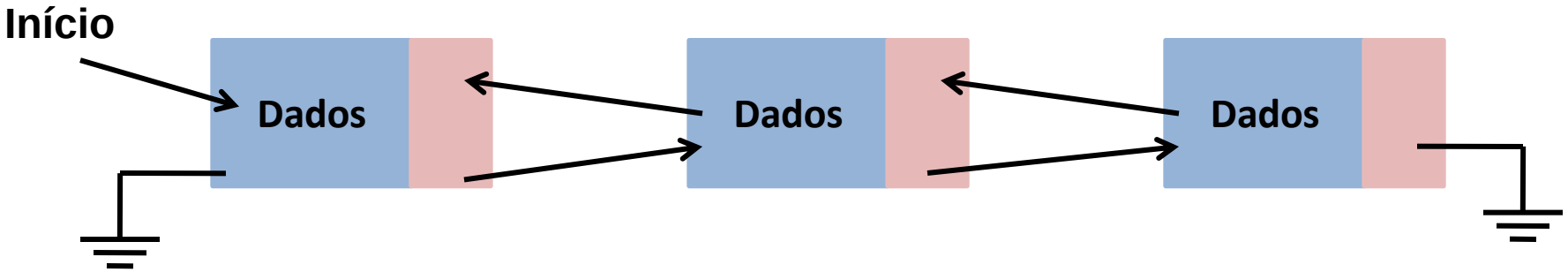
```
 /* retira elemento */
if (ant == NULL) {
        /* retira elemento do início */
        l = p->prox;
}
else {
        /* retira elemento do meio da lista */
        ant->prox = p->prox;
}
free(p);
return l;
}
```

Lista Simplesmente Encadeada

- Função para liberar uma lista

```
void libera (struct lista * l) {
        struct lista * p = l;
        struct lista * t;
        while (p != NULL) {
            t = p->prox;              /* referência para o próximo elemento*/
            free(p);                  /* libera a memória apontada por p */
            p = t;                    /* faz p apontar para o próximo */
        }
}
```

Lista Duplamente Encadeada




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