UNIVERSIDADE FEDERAL DE ITAJUBÁ

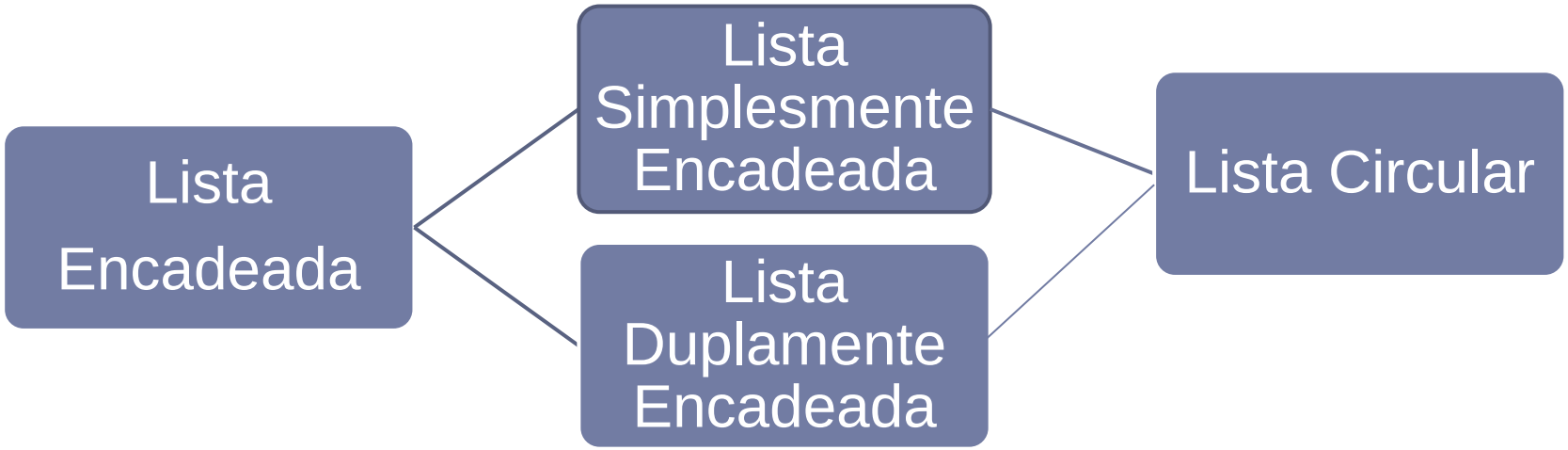
Algoritmos e Estrutura de Dados I

COM 111

Variações do TAD Lista
Lista Circular

Vanessa Cristina Oliveira de Souza

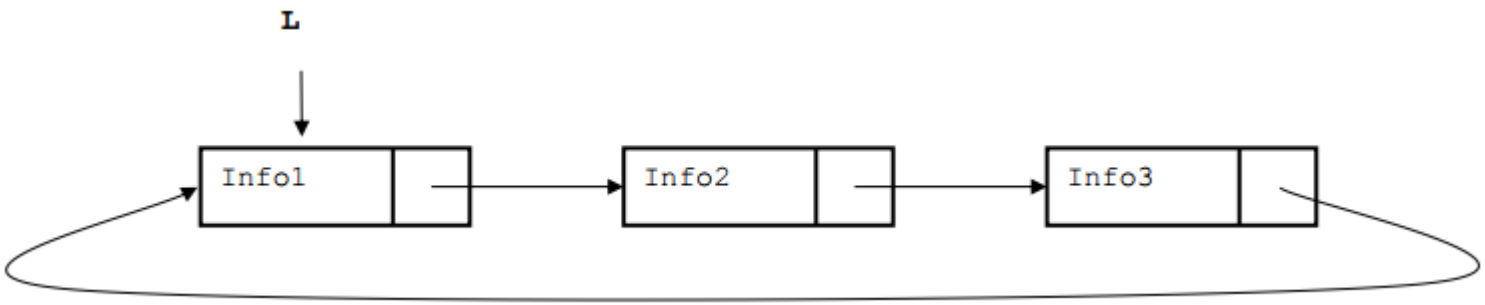
Introdução


Lista Encadeada
Lista Simplesmente Encadeada
Lista Duplamente Encadeada
Lista Circular

Lista Circular

Lista Circular

- O último elemento tem como próximo o primeiro elemento da lista, formando um ciclo
  - a lista pode ser representada por um ponteiro para um elemento inicial qualquer da lista
  - "**ela não tem fim**"



Lista Circular

- Para tornar a lista interminável, o ponteiro **prox** do último elemento apontará para o **primeiro elemento** da lista, em vez do valor NULL, como vimos no caso das listas simplesmente e duplamente encadeadas.

- Nas listas circulares, nunca chegaremos a uma posição a partir da qual não poderemos mais nos mover.

- Chegando ao último elemento, o deslocamento vai recomeçar no primeiro elemento.

Lista Circular

- Dada uma estrutura de lista circular (simplesmente encadeada) vazia:
  - Insira os seguintes elementos em ordem crescente:
    - 5, 2, 3, 7, 10, 1
  - Remova o elemento 10
  - Remova o elemento 1

TAD LISTA CIRCULAR – Insere Ordenado

```
//noListavoNo é o primeiro elemento da lista
if (aux == NULL || aux->valor > valor)
{
    novoNo->prox = aux;
    l->tamanho++;
    //se mudou o início da lista, precisa mudar
    if (aux == NULL)
    {
        novoNo->prox = novoNo;
        l->inicio = novoNo;
        return;
    }
    while (aux->prox != l->inicio)
    {
        aux = aux->prox;
    }
    aux->prox = novoNo;
    l->inicio = novoNo;
    return;
}
while ((aux->prox != l->inicio) && (aux->prox->valor < valor))
{
    aux = aux->prox;
}
novoNo->prox = aux->prox;
aux->prox = novoNo;
l->tamanho++;
return;
```

```
//noListavoNo é o primeiro elemento da lista
if (aux == NULL || aux->valor > valor)
{
    l->inicio = novoNo;
    novoNo->prox = aux;
    l->tamanho++;
    return;
}
while ((aux->prox != NULL) && (aux->prox->valor < valor))
{
    aux = aux->prox;
}
novoNo->prox = aux->prox;
aux->prox = novoNo;
l->tamanho++;
return;
```

TAD LISTA CIRCULAR – Remove elemento

```
//elemento a ser removido é o primeiro da lista
if (aux->valor == valor)
{
    while (aux->prox != l->inicio)
    {
        aux = aux->prox;
    }
    aux->prox = l->inicio->prox;
    aux = l->inicio;
    l->inicio = aux->prox;
    free(aux);
    return;
}
```

```
//elemento a ser removido é o primeiro da lista
if (aux->valor == valor)
{
    l->inicio = aux->prox;
    free(aux);
    return;
}
```

TAD LISTA CIRCULAR – Consulta elemento

```
int consultaElemento(lista *l, int valor)
{
    noLista *aux = l->inicio;
    if (aux == NULL)
    {
        printf("\nLista Vazia\n");
        return;
    }
    while (aux->prox !=l->inicio && aux->valor < valor)
    {
        aux = aux->prox;
    }
    if ((aux == NULL) || (aux->valor != valor))
        return 0;
    else
        return 1;
}
```

TAD LISTA CIRCULAR

- Exercícios
  - Alterar as demais funções (se necessário) do TAD Lista para uma Lista Circular
  - Desenvolva um procedimento que recebe um ponteiro para um nó qualquer de uma lista circular e retorna o número de nós dessa lista.
    - *int tamanhoCalc(lista *l, noLista *no)*
    - Será necessário alterar a função consultaElemento :
      - noLista *consultaElemento2(lista *l, int valor);

TAD LISTA CIRCULAR

- Exercícios
  - Nas adaptações feitas para a lista circular, foi preciso percorrer a lista para encontrar seu fim e atualizar o ponteiro para o início todas as vezes que um elemento foi inserido ou removido do início da lista.

```
//elemento a ser removido é o primeiro da lista
if (aux->valor == valor)
{
    while (aux->prox != l->inicio)
    {
        aux = aux->prox;
    }
    aux->prox = l->inicio->prox;
    aux = l->inicio;
    l->inicio = aux->prox;
    free(aux);
    return;
}
```

  - Essa complexidade pode ser removida acrescentando um ponteiro para o final da lista.

```
struct lista
{
    struct no *inicio;
    struct no *fim;
    int tamanho;
};
```

TAD LISTA CIRCULAR

- Exercícios
  - Dada uma estrutura de lista circular (simplesmente encadeada) vazia, com ponteiro de início e fim da lista :
    - Insira os seguintes elementos em ordem crescente:
      - 5, 2, 3, 7, 10, 1
    - Remova o elemento 10
    - Remova o elemento 1
  - Alterar as funções (se necessário) do TAD Lista Circular para uma Lista Circular com ponteiro para o fim da Lista.

TAD LISTA CIRCULAR

- Exercícios
  - Execute o seguinte programa usuário no seu TAD Lista Circular:
    - Com um ponteiro no início
    - Com um ponteiro no início e um no fim

```
int main()
{
    lista *l = criaLista();
    noLista *no;
    insereOrdenado(l, 5);
    insereOrdenado(l, 7);
    insereOrdenado(l, 2);
    insereOrdenado(l, 3);
    insereOrdenado(l, 1);
    insereOrdenado(l, 10);
    imprimeLista(l);
    removeElemento(l, 10);
    removeElemento(l, 1);
    imprimeLista(l);
    no = consultaElemento2(l, 5);
    printf("\n O tamanho da lista eh : %d\n" ,tamanhoCalc(l, no));
    insereOrdenado(l, 4);
    imprimeLista(l);
    no = consultaElemento2(l, 5);
    printf("\n O tamanho da lista eh : %d\n" ,tamanhoCalc(l, no));
    return (EXIT_SUCCESS);
}
```

Lista Circular

- Dada uma estrutura de lista circular (simplesmente encadeada) vazia:
  - Insira os seguintes elementos em ordem crescente:
    - 5, 2, 3, 7, 10, 1
  - Remova o elemento 10
  - Remova o elemento 1