Estruturas de Dados
Listas

Prof. Tales Nereu Bogoni
tales@unemat.br

UNEMAT
Universidade do Estado de Mato Grosso

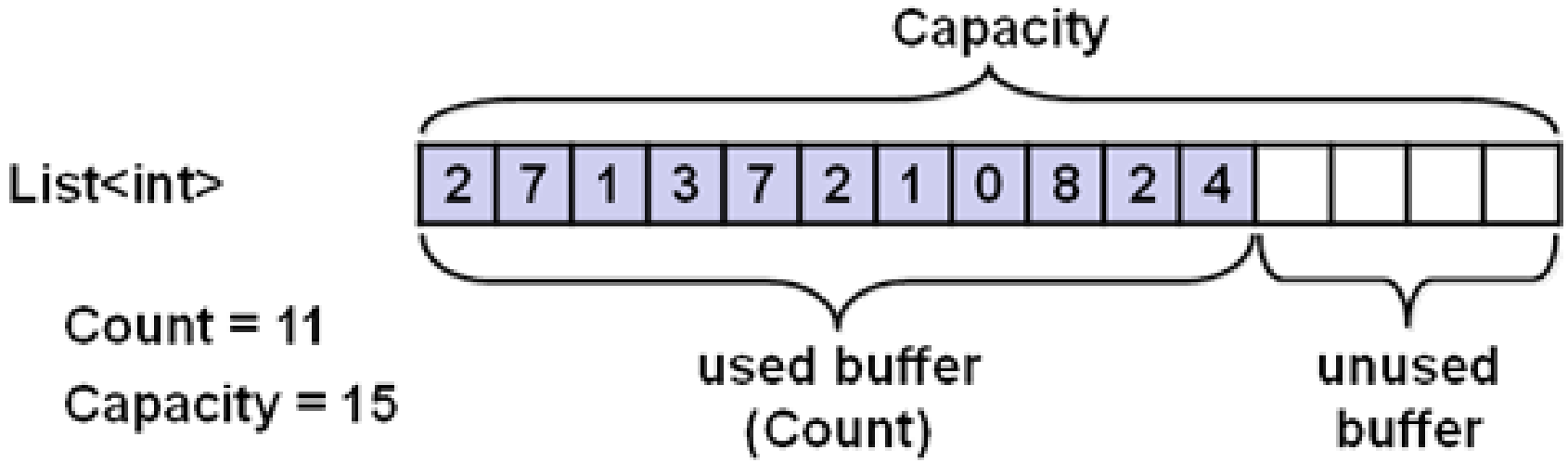
Listas

UNEMAT
Universidade do Estado de Mato Grosso

- Listas são estruturas extremamente flexíveis que possibilitam uma ampla manipulação das informações uma vez que inserções e remoções podem acontecer em qualquer posição
- As listas são estruturas compostas, constituídas por dados de forma a preservar a relação de ordem linear entre eles
- Podem ser de qualquer tipo de dado
- Em geral, uma lista segue a forma a1, a2, a3, ..., an
  - onde n determina o tamanho da lista
- Quando n = 0 a lista é chamada nula ou vazia.
- Para toda lista, exceto a nula, a_{i+1} segue (ou sucede) a_i (i < n), e a_{i-1} precede a_i (i > 1)
- O primeiro elemento da lista é a_1, e o último a_n

Definição de Lista

- Em geral, uma lista segue a forma a1, a2, a3, ..., an
  - onde n determina o tamanho da lista
- Quando n = 0 a lista é chamada nula ou vazia.
- Para toda lista, exceto a nula, a_{i+1} segue (ou sucede) a_i (i < n), e a_{i-1} precede a_i (i > 1)
- O primeiro elemento da lista é a_1, e o último a_n
- A posição correspondente ao elemento a_i na lista é *i*



Características

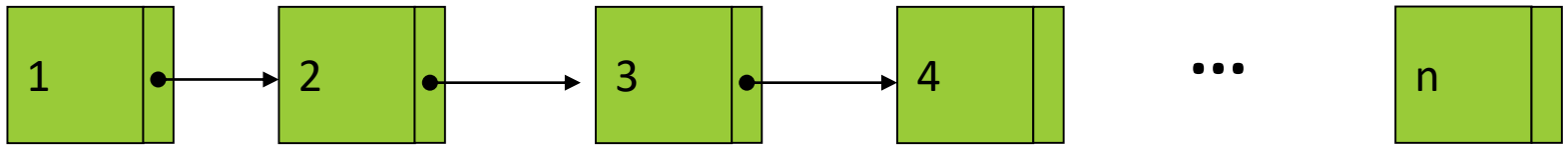
UNEMAT
Universidade do Estado de Mato Grosso

- Homogênea. Todos os elementos da lista são do mesmo tipo
- A ordem nos elementos é decorrente da sua estrutura linear, no entanto os elementos não estão ordenados pelo seu conteúdo, mas pela posição ocupada a partir da sua inserção
- Para cada elemento existe anterior e seguinte, exceto o primeiro, que não possui anterior, e o último, que não possui seguinte
- É possível acessar e consultar qualquer elemento na lista
- É possível inserir e remover elementos em qualquer posição

Alocação Dinâmica de Memória

UNEMAT
Universidade do Estado de Mato Grosso

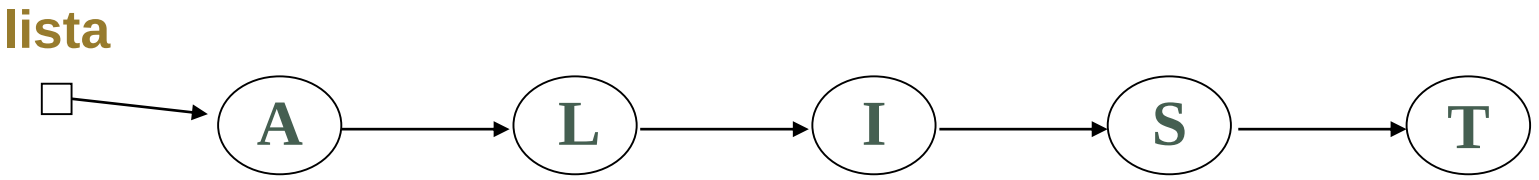
- Chamada de **Lista Encadeada**
- Posições de memória são alocadas a medida que são necessárias
- Cada elemento é chamado de **nó da lista**
- Nós encontram-se aleatoriamente dispostos na memória e são interligados por ponteiros, que indicam o próximo nó
  - Nós precisam de um campo a mais



Listas Encadeadas

UNEMAT
Universidade do Estado de Mato Grosso

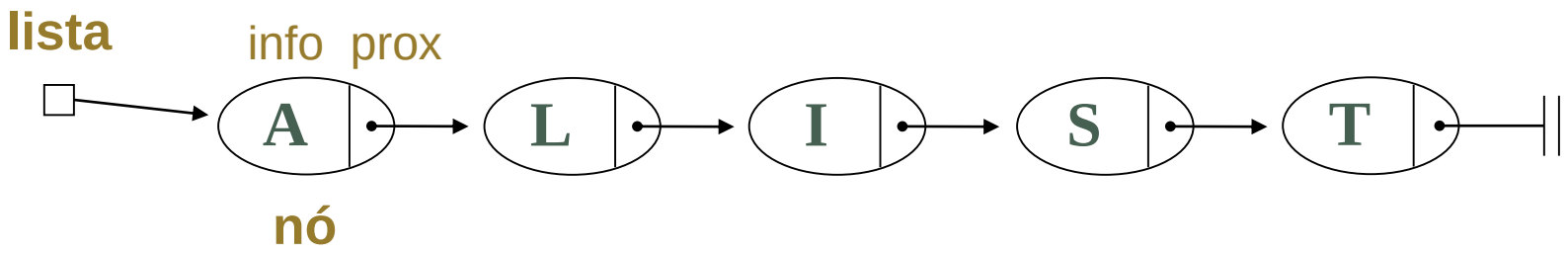
- A sequência de elementos é especificada explicitamente, onde cada um contém um ponteiro para o próximo da lista (*link*) → **Encadeamento**
- A lista é representada por um ponteiro para o primeiro elemento (ou nó)
- Do primeiro elemento, pode-se alcançar o segundo seguindo o encadeamento e assim sucessivamente
- Para cada novo elemento inserido na estrutura, um espaço na memória é alocado dinamicamente



Listas Encadeadas

UNEMAT
Universidade do Estado de Mato Grosso

- Cada elemento possui pelo menos dois campos: um para armazenar a **informação** e outro para o endereço do **próximo** (ponteiro)
- Deve haver um ponteiro especial para o 1º da lista
- O ponteiro do último elemento tem que especificar algum tipo de *final* (aponta para si próprio ou **nulo**)
- Uma lista vazia (ou nula) é uma lista sem nós



Operações Básicas com Listas

UNEMAT
Universidade do Estado de Mato Grosso

- Criar a lista
- Verificar se a lista está vazia
- Inserir/remover nó no início da lista
- Inserir/remover nó no final da lista
- Inserir/remover nó no meio da lista
- Pesquisar um elemento dentro da lista
- Exibir um Nó da lista
- Exibir toda a lista

Criar uma lista

UNEMAT
Universidade do Estado de Mato Grosso

- Para implementar uma lista serão necessárias duas classes
  - No : informações que deseja armazenar e uma referência para um outro nó
  - Lista : que armazenará apenas uma referência de um Nó, que será a cabeça da lista
    - Possui todos os métodos para trabalhar com a lista
- Inicialmente uma lista está vazia

```
public class No {
   public int info;
   public No proximo;
 }
```

```
public class Lista {
   public No lista;
}
```

```
public static void
    main(String[] args) {
       Lista l = new Lista();
}
```

Verificar se a lista está vazia

UNEMAT
Universidade do Estado de Mato Grosso

- Um método bastante utilizado quando se trabalha com listas é verificar se ela está vazia.
- Basicamente este método é auxiliar para os demais métodos que são implementados nas listas
- Consiste em apenas verificar se o primeiro Nó da lista existe ou não

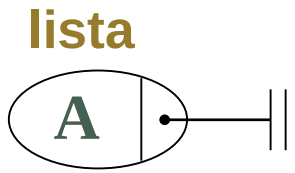
```
public boolean vazia(){
        return lista==null;
    }
```

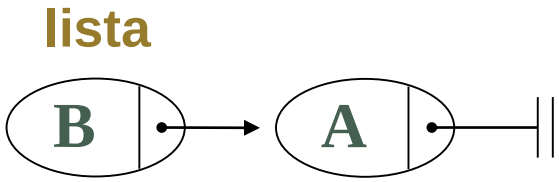
Inserir um elemento na Cabeça da lista

UNEMAT
Universidade do Estado de Mato Grosso

- Quando se está trabalhando com listas simplesmente encadeadas a inserção pela Cabeça da Lista é a operação mais simples
- Consiste em criar um novo Nó e coloca-lo como sendo o nó inicial da lista, e, se existirem outros valores, fazer com que este novo Nó aponte para o Nó que estava na cabeça da lista anteriormente.

```
public void addFirst(int info){
        No n = new No();
        n.info = info;
        n.proximo = lista;
        this.lista = n;
    }
```





Mostrar a lista

UNEMAT
Universidade do Estado de Mato Grosso

- Este método é utilizado para exibir o conteúdo da lista, utilizado didaticamente, em implementações de listas normalmente ele não existe
- Consiste em percorrer todos os Nós da lista a partir da Cabeça e exibir seu conteúdo
- Aqui são usados dois métodos, um na Classe Nó (que mostra os dados de um nó) e outro na classe Lista (que percorre toda a lista mostrando os Nós)

Mostrar a lista

UNEMAT
Universidade do Estado de Mato Grosso

```
public void showList(){
        No aux = lista;
        if(this.vazia())
            System.out.println("Lista Vazia");
        else {
            System.out.println("Cabeça");
            while(aux!=null){
                aux.showNo();
                aux = aux.proximo;
            }
            System.out.println("Cauda");
        }
    }
```

```
public void showNo(){
        System.out.println
       ("Info: "+info+ "   Próximo->“
       +proximo);
    }
```

Remover um elemento na Cabeça da lista

UNEMAT
Universidade do Estado de Mato Grosso

- Em listas simplesmente encadeadas a remoção também é mais simples de ser feita a partir da cabeça da lista
- É necessário verificar se a lista não está vazia
- A nova cabeça da lista passa a ser elemento que a antiga cabeça estava apontando

```
public void removeFirst(){
        if(!this.vazia()){
            System.out.println("Removendo...");
            lista.showNo();
            lista = lista.proximo;
        }
    }
```

Adicionar um elemento no final da lista (cauda)

UNEMAT
Universidade do Estado de Mato Grosso

- Para executar esta operação em uma lista simplesmente encadeada é necessário percorrer toda a lista para saber qual é o último nó
- Depois de encontrar o último Nó, este deve apontar para o Nó que está sendo inserido

```
public void addLast(int info){
        if(this.vazia())
            addFirst(info);
        else {
            No n = new No();
            n.info = info;
            No aux = lista;
            while (aux.proximo!=null)
                aux = aux.proximo;
            aux.proximo = n;
        }
    }
```

Remover um elemento no final da lista (cauda)

UNEMAT
Universidade do Estado de Mato Grosso

- É uma operação bastante complexa quando se trata de uma lista simplesmente encadeada
- É necessário descobrir quem é o Nó antecessor do último Nó da lista para fazer com que ele aponte para um valor nulo, eliminando assim o último elemento

```
public void removeLast(){
        if(!this.vazia()){
            No anterior = this.lista;
            No aux = this.lista;
            while (aux.proximo != null){
                anterior = aux;
                aux = aux.proximo;
            }
            System.out.println("Removendo...");
            aux.showNo();
            anterior.proximo = null;
            if (aux == lista)
                lista = null;
        }
    }
```

Adicionar um elemento no meio de uma lista

UNEMAT
Universidade do Estado de Mato Grosso

- Esta é uma das grandes vantagens de trabalhar com listas com relação ao uso de arrays
- Primeiro deve-se localizar a posição onde o elemento deve ser inserido
- Em seguida fazer com que o novo Nó aponte para o mesmo local do Nó antecessor, e, o Nó antecessor deve apontar para o novo Nó

```
public void addLastValue(int value, int info){
      if(this.vazia())
         addFirst(info);
      else{
         No aux = this.lista;
         {
            while(aux.proximo != null && aux.info!=value)
               aux = aux.proximo;
            if(aux.info == value)
            {
               No n = new No();
               n.info = info;
               n.proximo = aux.proximo;
               aux.proximo = n;
            } else {
               addLast(info);
            }        }     }    }
```

Remover um elemento no meio de uma lista

UNEMAT
Universidade do Estado de Mato Grosso

- Outra grande vantagem de trabalhar com listas com relação ao uso de arrays é a remoção de elementos do meio da lista
- Primeiro deve-se localizar a o elemento dentro da lista
- Em seguida fazer com que o Nó antecessor aponte para o local onde o Nó que será eliminado está apontando

```
public void removeInfo(int info){
      if(!this.vazia()){
         No aux = lista;
         No anterior = lista;
         while(aux.proximo!=null && aux.info!=info){
            anterior = aux;
            aux = aux.proximo;
         }
         if(aux.info == info){
            System.out.println("Removendo...");
            aux.showNo();
            if(aux == lista) //1o elemento
               lista=aux.proximo;
            else // outros elementos
               anterior.proximo = aux.proximo;
         } else
            System.out.println(info+" não existe na lista");
      }   }
```

Exercício

UNEMAT
Universidade do Estado de Mato Grosso

- Fazer uma lista para inserir livros
  - String codigo, titulo, autor
- Executar as seguintes operações
  - Inserir na cabeça (“AB123”, “A volta dos que não foram”, “Filisbina”)
  - Inserir na cabeça (“XY234”, ”As longas tranças de um careca”, “Mefistófeles”)
  - Inserir na cauda(“GH897”, ”Tarzan atravessa o deserto a nado”, ”Minhocoleta”)
  - Inserir após “AB123” (“JK564”, “Matando a morte”, “Coveiro”)
  - Mostrar a lista de livros
  - Remover o livro “AB123”
  - Remover o livro da Cauda
  - Remover o livro da Cabeça
  - Mostrar a lista de livros

Implementação de métodos

UNEMAT
Universidade do Estado de Mato Grosso

- Criar um método que faça a inserção de dados ordenada pelo código do livro dentro da lista
- Faça um método que conte quantos elementos tem na lista
- Faça um método que retorne uma segunda lista com a metade da lista original (os primeiros elementos) e que a lista original fique somente com os últimos elementos
- Faça um método que concatene duas listas
- Faça um método que retorne a posição (valor numérico) de uma determinado elemento dentro da lista