



Estrutura de Dados

Prof. Dr. Bruno Aguilar da Cunha
bruno.cunha@facens.br

Vamos praticar!

Exercícios!

1) Cadastre 5 números em uma fila e mais 5 em uma pilha. Em seguida, possibilite ao usuário escolher exibir as seguintes informações:

- 1- os números que estão nas duas estruturas.
- 2- os que estão na fila.
- 3- os que estão na pilha.
- 4- colocar os números pares da fila na pilha
- 5- colocar os números ímpares da pilha na fila

Vamos praticar!

Exercícios!

2) Usando a classe Pilha disponibilizada, desenvolva um algoritmo que realize a conversão de números decimais para binário.

Vamos praticar!

Exercícios!

3) Escreva um programa que simule a distribuição de senhas de atendimento a um grupo de pessoas. Cada pessoa pode receber uma senha prioritária ou normal. O programa deve obedecer os seguintes critérios:

- Existe apenas um atendente.
- 1 pessoa com senha normal deve ser atendida a cada 3 pessoas com senha prioritária
- Não havendo prioridades, as pessoas com senha normal podem ser atendidas.




01

LISTAS ENCADEADAS

LISTAS ENCADEADAS



Arranjos (Arrays) são **sequências de elementos de um mesmo tipo**, armazenados em posições contíguas de memória, e que possuem tamanho fixo, sendo úteis nos mais variados cenários. Entretanto, há situações em que desejamos maior flexibilidade, e as propriedades de arranjos não se adequam às nossas necessidades. Por exemplo, arranjos não são a estrutura de dados ideal se desejamos alguma das características a seguir:




LISTAS ENCADEADAS



Capacidade de inserir elementos no meio da estrutura de dados. Se usarmos um arranjo e precisarmos inserir um elemento no meio da sequência, precisamos mover todos os elementos do arranjo.

Capacidade de inserir um número arbitrário de elementos. Em outras palavras, deseja-se que a estrutura de dados possua um tamanho dinâmico, sem valor máximo pré-definido.



LISTAS ENCADEADAS

Listas encadeadas são a forma mais simples de estrutura de dados dinâmica. Em uma lista, os elementos não são armazenados contiguamente, como acontece em arranjos. Nelas, cada elemento é armazenado separadamente, e possui valor (o dado que desejamos acessar) e um ponteiro para o próximo elemento.

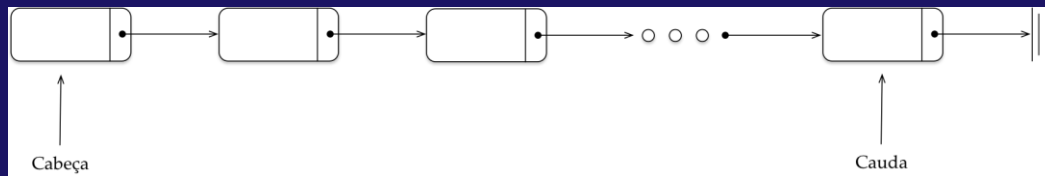
Um ponteiro (ou apontador) é uma variável que armazena o endereço de outra variável (eles referenciam outra variável).

LISTAS ENCADEADAS

Listas encadeadas são a forma mais simples de estrutura de dados dinâmica. Em uma lista, os elementos não são armazenados contiguamente, como acontece em arranjos. Nelas, cada elemento é armazenado separadamente, e possui valor (o dado que desejamos acessar) e um ponteiro para o próximo elemento.

Um ponteiro (ou apontador) é uma variável que armazena o endereço de outra variável (eles referenciam outra variável).

LISTAS ENCADEADAS



Listas encadeadas são a forma mais simples de estrutura de dados dinâmica. Em uma lista, os elementos não são armazenados contiguamente, como acontece em arranjos. Nelas, cada elemento é armazenado separadamente, e possui valor (o dado que desejamos acessar) e um ponteiro para o próximo elemento.

Um ponteiro (ou apontador) é uma variável que armazena o endereço de outra variável (eles referenciam outra variável).

LISTAS ENCADEADAS

No caso de listas encadeadas, um ponteiro armazena o endereço do próximo elemento da lista. Para percorrermos a lista, basta seguirmos os ponteiros ao longo dela.

A flexibilidade de listas encadeadas não vem de graça. Listas encadeadas possuem duas desvantagens principais. Primeiro, precisamos de espaço adicional para armazenar os ponteiros. Segundo, se quisermos acessar um elemento em uma dada posição da lista, precisamos percorrer todos os elementos anteriores a ele na lista.

LISTAS ENCADEADAS

A classe abaixo um nó de uma lista encadeada

```
public class IntNoSimples {  
    int valor;  
    IntNoSimples prox;  
    IntNoSimples(int ValorNo) {  
        valor = ValorNo;  
        prox = null;  
    }  
}
```

LISTAS ENCADEADAS

Para facilitar a implementação de alguns métodos importantes, criaremos uma nova classe para armazenar um ponteiro para o início (primeiro) e fim (último) da lista. Essa classe será nossa lista encadeada propriamente dita.

```
public class ListaEncadeada {  
    IntNoSimples primeiro, ultimo;  
    int numero_nos=0;  
  
    ListaEncadeada() {  
        primeiro = ultimo = null;  
    }  
}
```

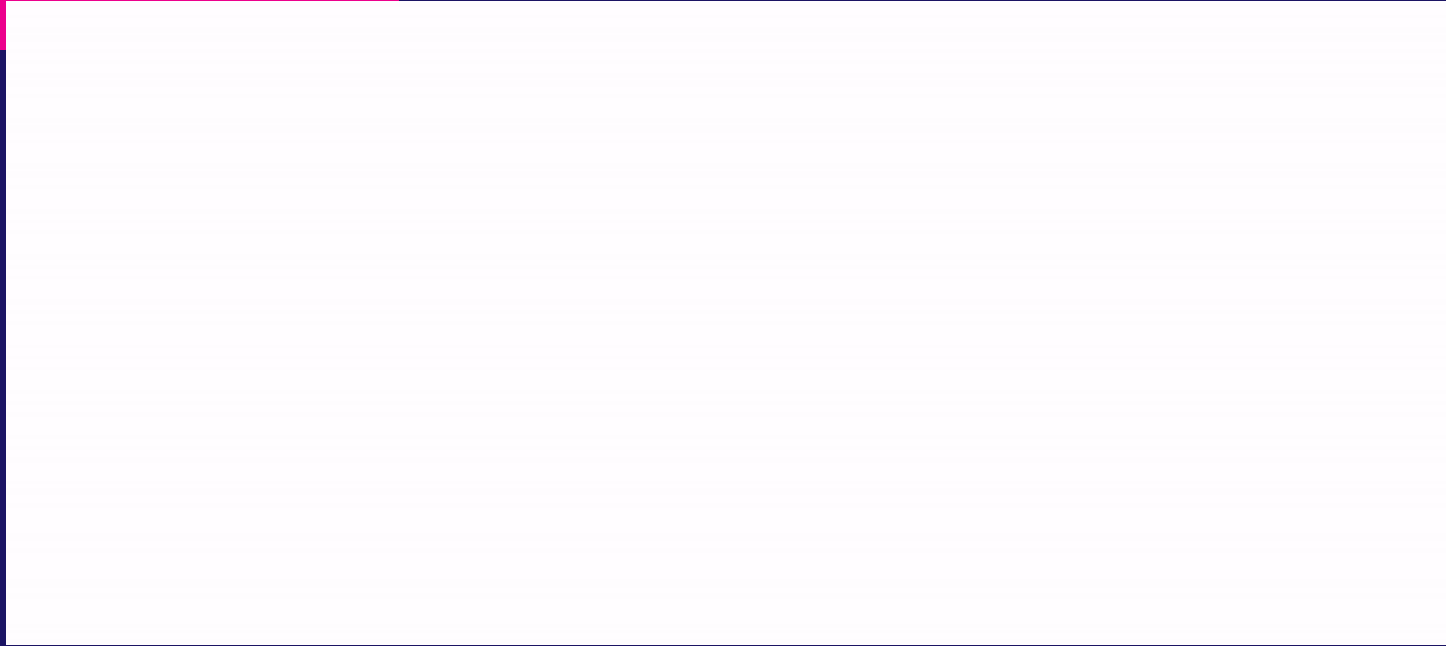
INSERIR NA LISTA (INICIO)

A forma mais simples e mais rápida de se inserir um elemento em uma lista encadeada é inseri-lo no começo da lista. O código abaixo estende nossa classe Lista definida anteriormente para conter uma função `insere_no_inicio`.

```
void insereNo_inicio (IntNoSimples novoNo) {  
    novoNo.prox = primeiro;  
    if(primeiro == null && ultimo==null)  
    {  
        ultimo = novoNo;  
    }  
    primeiro = novoNo;  
    numero_nos++;  
}
```

ILUSTRAÇÃO

INSERE_NO_INICIO



INSERIR NA LISTA (FIM)

```
void insereNo_fim (IntNoSimples novoNo) {  
    novoNo.prox = null;  
    if (primeiro == null)  
        primeiro = novoNo;  
    if (ultimo != null)  
        ultimo.prox = novoNo;  
    ultimo = novoNo;  
    numero_nos++;  
}
```


INSERIR NA LISTA (POSIÇÃO)

```
void insereNo_posicao(IntNoSimples novoNo, int posicao){
    IntNoSimples temp_no = primeiro;
    int numero_nos = ContarNos();
    int pos_aux;
    if(posicao == 0)
    {
        novoNo.prox = primeiro;
        if(primeiro == ultimo)
        {
            ultimo = novoNo;
        }
        primeiro = novoNo;
    }
    else
    {
        if (posicao <= numero_nos)
        {
            pos_aux = 1;
            while(temp_no != null && posicao > pos_aux)
            {
                temp_no = temp_no.prox;
                pos_aux ++;
            }
            novoNo.prox = temp_no.prox;
            temp_no.prox = novoNo;
        }
        else
        {
            if(posicao > numero_nos)
            {
                ultimo.prox = novoNo;
                ultimo = novoNo;
            }
        }
    }
}
```

BUSCA NA LISTA

```
IntNoSimples buscaNo (int buscaValor){  
    int i = 0;  
    IntNoSimples temp_no = primeiro;  
    while (temp_no != null)  
    {  
        if (temp_no.valor == buscaValor)  
        {  
            JOptionPane.showMessageDialog(null,  
                "No " + temp_no.valor + " posição " + i);  
            return temp_no;  
        }  
        i++;  
        temp_no = temp_no.prox;  
    }  
    return null;  
}
```

EXCLUSÃO NA LISTA

```
void excluNo (int valor){
    IntNoSimples temp_no = primeiro;
    IntNoSimples anterior_no=null;
    while (temp_no != null && temp_no.valor != valor){
        anterior_no = temp_no;
        temp_no = temp_no.prox;
    }
    if (temp_no == primeiro){
        if (temp_no.prox !=null)
            primeiro = temp_no.prox;
        else
            primeiro = null;
    }
    else{
        anterior_no.prox =temp_no.prox;
    }
    if (ultimo == temp_no)
        ultimo = anterior_no;
}
```

EXIBIÇÃO DA LISTA

```
void exhibeLista() {  
    IntNoSimples temp_no = primeiro;  
    int i = 0;  
    while (temp_no != null)  
    {  
        System.out.println(  
            "Valor" + temp_no.valor + " posição " + i);  
        temp_no = temp_no.prox;  
        i++;  
    }  
}
```

EXEMPLO DE USO (PARTE 1)

```
public class ExemploLista {
    int i = 0;
    IntNoSimples temp_no;
    int valor;
    public static void main(String[] args) {
        ListaEncadeada l = new ListaEncadeada();
        int opcao = 1, valor, posicao;
        while (opcao != 7) {
            opcao = Integer.parseInt (JOptionPane.showInputDialog(null,
                "Escolha uma Opção \n" + "1-Inserir Nó no início \n" +
                "2-Inserir Nó no fim \n" + "3-Inserir Nó em uma posição\n" + "4-Localizar Nó \n" +
                "5-Excluir Nó \n" + "6-Exibir lista \n" + "7-Sair"));
            switch (opcao) {
                case 1 :
                    valor = Integer.parseInt (JOptionPane.showInputDialog(null, "Inserir um Nó no início da lista\n" +
                        "Digite um valor"));
                    l.inserirNo_inicio(new IntNoSimples(valor));
                    break;
                case 2 :
                    valor = Integer.parseInt (JOptionPane.
                        showInputDialog(null, "Inserir um Nó no final da lista \n" +
                        "Digite um valor"));
                    l.inserirNo_fim(new IntNoSimples(valor));
                    break;
```

EXEMPLO DE USO (PARTE 2)


```
case 3 :
    valor = Integer.parseInt (JOptionPane.showInputDialog(null,
        "Inserir um Nó em uma posição \n" +
        "Digite um valor"));
    posicao = Integer.parseInt (JOptionPane.showInputDialog(null, "Digite a posição"));
    l.inserirNo_posicao(new IntNoSimples(valor),posicao);
    break;
case 4:
    valor = Integer.parseInt (JOptionPane.showInputDialog(null, "Localiza um valor \n" +
        "Digite um valor"));
    l.buscaNo(valor);
    break;
case 5:
    valor = Integer.parseInt (JOptionPane.showInputDialog(null,
        "Exclui um nó da lista \n" + "Digite um valor"));
    l.excluiNo(valor);
    break;
case 6:
    JOptionPane.showMessageDialog(null,"Exibe a lista");
    l.exibeLista();
    break;
default : JOptionPane.showMessageDialog(null,"Sair");
    }
}
```

BUSCAS EM LISTAS



A pesquisa por um valor em uma lista é uma tarefa relativamente fácil, se comparada à tarefa de inserção de elementos.

O único detalhe a ser observado é o fato de que precisamos percorrer a lista desde o começo para pesquisar por qualquer um dos elementos.





02

EXERCÍCIOS - LISTAS

LISTA ENCADEADA - EXERCÍCIOS

1) Escreva um programa em JAVA que leia números inteiros e armazene em uma LISTA ENCADEADA. A entrada de dados deve ser interrompida quando o usuário informar o número zero ou esgotar a quantidade definida de elementos a serem armazenados na estrutura. Por último, imprima os elementos existentes na lista encadeada criada.

LISTA ENCADEADA - EXERCÍCIOS

2) Escreva um programa em JAVA que leia números inteiros e só armazene aqueles que forem pares em uma LISTA ENCADEADA. A entrada de dados deve ser interrompida quando o usuário informar o número zero ou esgotar a quantidade definida de elementos a serem armazenados na estrutura. Por último, imprima os elementos existentes na lista encadeada criada.

LISTA ENCADEADA - EXERCÍCIOS

3) Escreva um programa em JAVA que crie e preencha uma pilha e remova os elementos da pilha e coloque na lista encadeada e na sequência imprima os elementos existentes na lista.

Dúvidas?

Dúvidas fora do horário de aula:
bruno.cunha@facens.br