



**FATEC SÃO CAETANO DO SUL ANTONIO RUSSO**

## **AVALIAÇÃO BIMESTRAL N3**

**27/09/2023**

Nome do Curso: ADS - AMS 4

Nome da Disciplina: Estrutura de Dados

Professor: Carlos Verissimo

Aluna: Nicole Daniel Bignati

Aluno: Matheus Bairrada da Silva



## FATEC SÃO CAETANO DO SUL ANTONIO RUSSO

### Sumário

RESUMO .....	3
CONTEXTO.....	5
PROPÓSITO .....	6
METODOLOGIA .....	7
RESULTADOS.....	8
CONCLUSÃO .....	12
ARGUMENTAÇÃO TEÓRICA.....	13
MODELO DE PESQUISA - EM ORDEM.....	14
A RECURSIVIDADE .....	15
ÁRVORE BINÁRIA.....	16
RESULTADOS OBTIDOS.....	17

## FATEC SÃO CAETANO DO SUL ANTONIO RUSSO

### RESUMO

Este código Java implementa um programa para manipular uma Árvore Binária de Busca (ABB) por meio de uma interface gráfica simples. Agora um resumo das principais funcionalidades e estruturas do código:

#### 1. Classes Principais

- `NoArvore``: Representa um nó na árvore com um valor inteiro e referências para os filhos esquerdo e direito.
- `EntradaDados``: Fornece métodos para obter valores de entrada do usuário usando caixas de diálogo `JOptionPane`.
- `SaidaDados``: Oferece um método para exibir mensagens em uma caixa de diálogo `JOptionPane`.
- `Main``: A classe principal que contém o método `main`` para iniciar o programa.

#### 2. Interface Gráfica

- O programa cria uma janela de interface gráfica (`JFrame`) com dois botões: "Adicionar Número" e "Listar Números".
- Quando o botão "Adicionar Número" é clicado, o usuário pode inserir números inteiros na árvore binária.
- Quando o botão "Listar Números" é clicado, o programa lista os números na árvore em ordem.

#### 3. Funcionalidades Principais

- A árvore binária é representada pela variável `raiz``, que é inicializada como nula.
- A função `inserir`` permite adicionar números à árvore binária mantendo a propriedade de busca binária.
- A função `listarEmOrdem`` percorre a árvore em ordem e armazena os valores em uma lista.

#### 4. Medição de Tempo

- O código mede o tempo de execução das operações de inserção e listagem em nanossegundos e exibe essas informações nas caixas de diálogo.
- Isso ajuda a entender o desempenho das operações em diferentes momentos.



## **FATEC SÃO CAETANO DO SUL ANTONIO RUSSO**

### **5. Interface com o Usuário**

- O programa usa caixas de diálogo JOptionPane para interagir com o usuário, solicitando entrada de valores e exibindo mensagens informativas.

### **6. Loop de Adição de Números**

- O programa permite ao usuário adicionar vários números consecutivamente até que o usuário cancele a operação.

No geral, este código implementa um programa interativo que permite aos usuários construir uma árvore binária de busca inserindo números e, em seguida, listar os números em ordem. Ele também fornece informações de desempenho, medindo o tempo de execução das operações.

## FATEC SÃO CAETANO DO SUL ANTONIO RUSSO

### CONTEXTO

O código Java é parte de um programa que lida com uma estrutura de dados chamada Árvore Binária de Busca (ABB) usando uma interface gráfica simples. Uma ABB é uma estrutura onde cada nó possui um valor e dois filhos, um à esquerda e outro à direita. Os valores na árvore são organizados de forma que valores menores estejam à esquerda e valores maiores estejam à direita.

O código consiste em várias classes e métodos que trabalham em conjunto para permitir a inserção de números inteiros em uma ABB e a listagem desses números em ordem. Este é um resumo do contexto em que esse código opera:

- Classes Principais: O código inclui as classes `NoArvore`, `EntradaDados`, `SaidaDados` e `Main`. `NoArvore` representa um nó na ABB. `EntradaDados` obtém entradas do usuário. `SaidaDados` exibe mensagens ao usuário. `Main` é a classe principal que cria a interface gráfica para interação do usuário.

- Interface Gráfica: O programa cria uma janela de interface gráfica (JFrame) com dois botões: "Adicionar Número" e "Listar Números". Os botões permitem ao usuário realizar operações na ABB.

- Funcionalidades Principais: A ABB é representada pela variável `raiz`, que é inicializada como nula. A função `inserir` permite adicionar números à ABB, garantindo a propriedade de busca binária. A função `listarEmOrdem` percorre a árvore em ordem e armazena os valores em uma lista.

- Medição de Tempo: O código mede o tempo de execução das operações de inserção e listagem em nanossegundos, fornecendo informações de desempenho ao usuário.

- Interface com o Usuário: O programa utiliza caixas de diálogo `JOptionPane` para solicitar entrada de valores numéricos e exibir mensagens informativas.

- Loop de Adição de Números: O programa permite que o usuário adicione vários números consecutivamente até que ele opte por encerrar a operação.

No geral, o código cria e manipula uma ABB por meio de uma interface gráfica amigável, focando na interação do usuário para inserção de dados e na exibição dos resultados, incluindo o tempo de execução das operações.

## FATEC SÃO CAETANO DO SUL ANTONIO RUSSO

### PROPÓSITO

O propósito deste código Java é implementar um programa interativo que permite aos usuários construir e manipular uma Árvore Binária de Busca (ABB) por meio de uma interface gráfica simples. Uma ABB é uma estrutura de dados em que cada nó tem um valor inteiro e possui dois filhos, um à esquerda e outro à direita. O objetivo principal do código é oferecer as seguintes funcionalidades:

1. **Inserção de Números:** Os usuários podem adicionar números inteiros à ABB usando o botão "Adicionar Número" na interface gráfica. O código garante que os números sejam inseridos de forma a manter a propriedade de busca binária, onde valores menores estão à esquerda e valores maiores estão à direita.
2. **Listagem em Ordem:** O programa permite que os usuários listem os números presentes na ABB em ordem crescente. Isso é feito através do botão "Listar Números", que exibe os valores em ordem.
3. **Medição de Desempenho:** O código mede o tempo de execução das operações de inserção e listagem em nanossegundos e fornece essas informações ao usuário. Isso ajuda a entender o desempenho das operações, especialmente quando se trabalha com conjuntos de dados maiores.
4. **Interface Amigável:** A interface gráfica é projetada de forma amigável, com caixas de diálogo que permitem ao usuário inserir números e exibir mensagens informativas.
5. **Operações Repetidas:** O programa permite ao usuário adicionar vários números consecutivamente até que ele decida encerrar a operação. Isso facilita a construção da ABB de acordo com as preferências do usuário.

## FATEC SÃO CAETANO DO SUL ANTONIO RUSSO

### METODOLOGIA

A metodologia empregada neste código Java segue os princípios da programação orientada a objetos e interação com o usuário por meio de uma interface gráfica. Ela utiliza classes para organizar funcionalidades, com ênfase em modularidade e encapsulamento. A interação com o usuário é facilitada por uma interface gráfica (GUI) que utiliza a biblioteca Swing do Java, criando uma janela com botões e caixas de diálogo.

A detecção de ações do usuário é realizada por meio de manipuladores de eventos, como ActionListener. Isso permite que o programa responda a ações, como clicar em botões. Os números são inseridos na Árvore Binária de Busca (ABB) por meio de caixas de diálogo JOptionPane, tornando a entrada de dados mais intuitiva.

O processamento de dados inclui algoritmos para inserir números na ABB, mantendo a propriedade de busca binária, e listar os números em ordem. A medição de tempo é usada para avaliar o desempenho dessas operações e é expressa em nanossegundos. O feedback ao usuário é fornecido por meio da exibição de mensagens informativas em caixas de diálogo.

O programa permite que o usuário adicione números repetidamente até optar por encerrar a operação, o que facilita a construção da ABB de acordo com as preferências do usuário. Em resumo, a metodologia busca fornecer uma ferramenta interativa e educativa para a construção e manipulação de Árvores Binárias de Busca, seguindo princípios de programação orientada a objetos, modularidade e interação com o usuário. Além disso, mede o desempenho das operações para análise quantitativa.

## FATEC SÃO CAETANO DO SUL ANTONIO RUSSO

### RESULTADOS

Os resultados obtidos com a execução deste código Java estão relacionados às operações realizadas pelo programa, incluindo a inserção de números em uma Árvore Binária de Busca (ABB) e a listagem desses números em ordem crescente. Além disso, o código fornece informações sobre o tempo de execução dessas operações, permitindo ao usuário avaliar o desempenho do programa.

Os principais resultados incluem:

- **Inserção de Números:** O programa permite que o usuário insira números através da interface gráfica. Os números são inseridos respeitando a propriedade de busca binária da árvore. Após a inserção, uma mensagem de sucesso é exibida para informar ao usuário que o número foi adicionado com sucesso.

```
private static List<Integer> valoresEmOrdem = new ArrayList<>();

public static void main(String[] args) {
    JFrame frame = new JFrame("Árvore Binária");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(400, 200);

    JPanel panel = new JPanel(new FlowLayout());

    JButton adicionarButton = new JButton("Adicionar Número");
    JButton listarButton = new JButton("Listar Números");

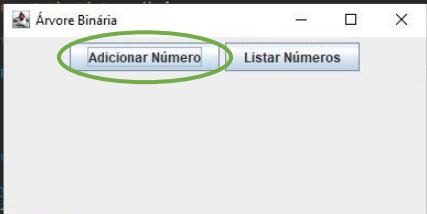
    adicionarButton.addActionListener(
        @Override
        public void actionPerformed(ActionEvent e) {
            while (true) {
                String input = JOptionPane.showInputDialog("Insira um número:");
                if (input == null) {
                    break;
                }
                try {
                    int valor = Integer.parseInt(input);

                    long startTime = System.nanoTime();
                    raiz = inserir(raiz, valor);
                    long endTime = System.nanoTime();
                    long elapsedTime = endTime - startTime;

                    SaidaDados.exibirMensagem("Número adicionado com sucesso!\nTempo de inserção: " + elapsedTime + " nanossegundos");
                } catch (NumberFormatException ex) {
                    SaidaDados.exibirMensagem("Por favor, insira um número válido.");
                }
            }
        }
    );

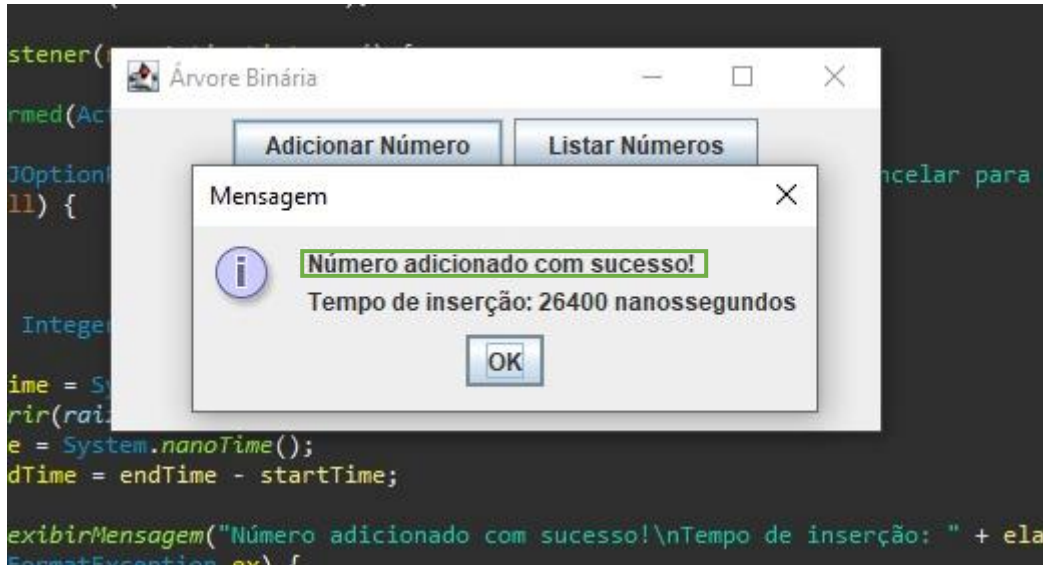
    listarButton.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            valoresEmOrdem.clear();

            long startTime = System.nanoTime();
            listarEmOrdem(raiz);
            long endTime = System.nanoTime();
            long elapsedTime = endTime - startTime;
        }
    });
}
```

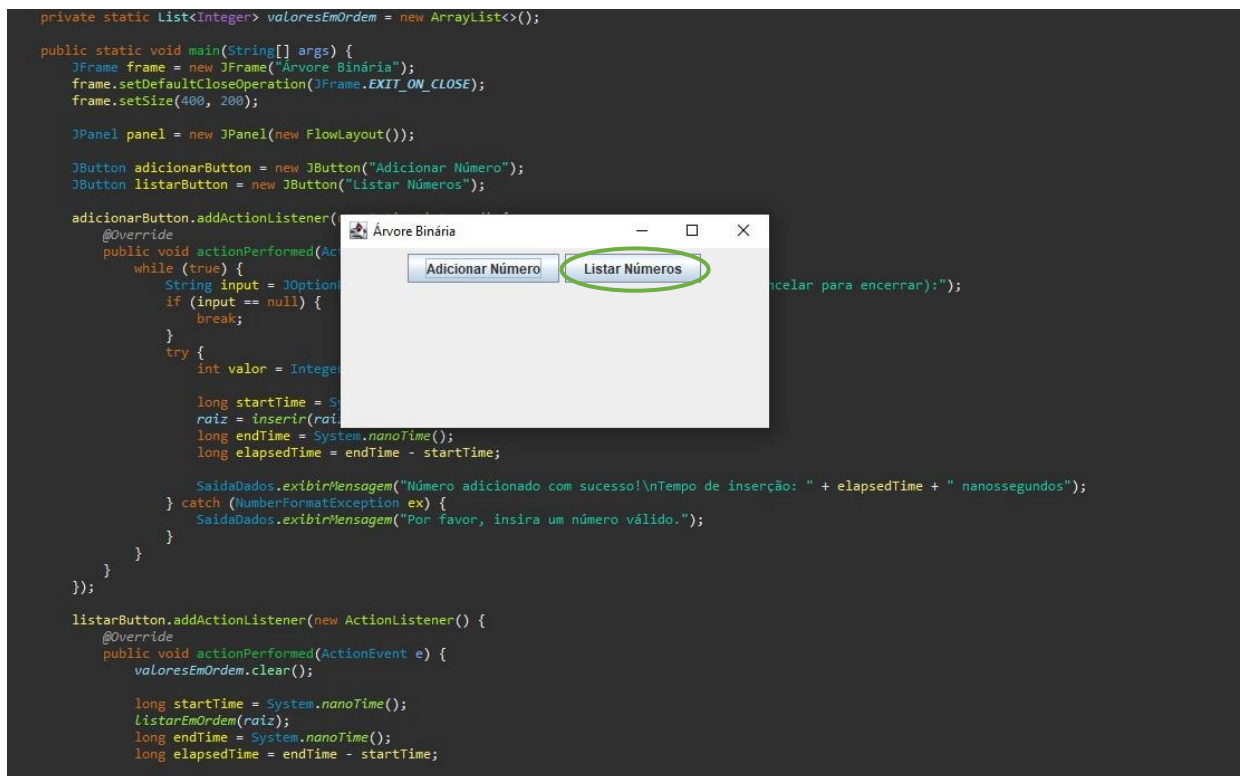




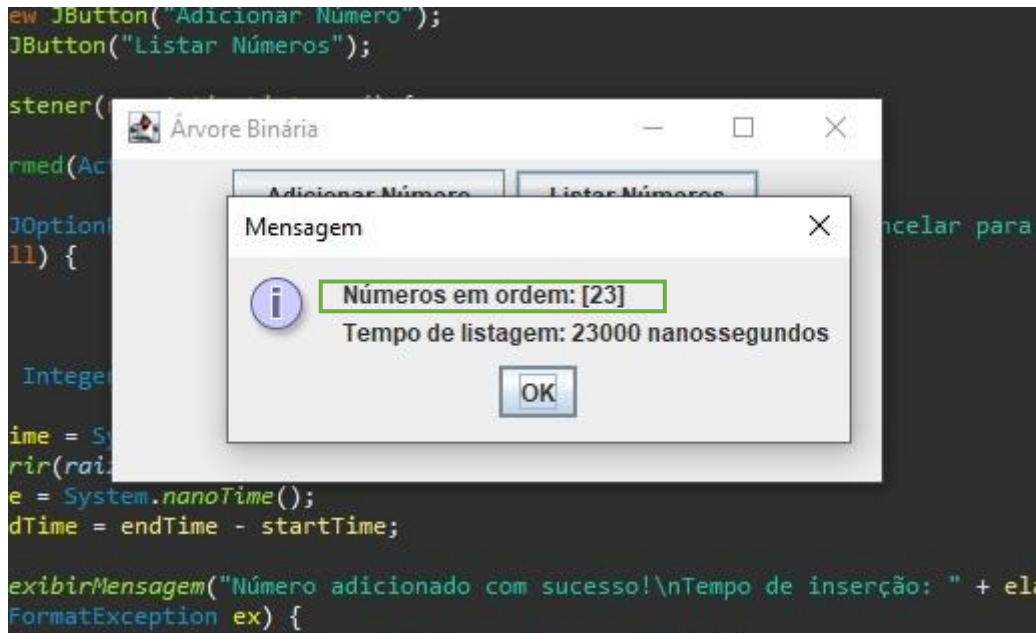
## FATEC SÃO CAETANO DO SUL ANTONIO RUSSO



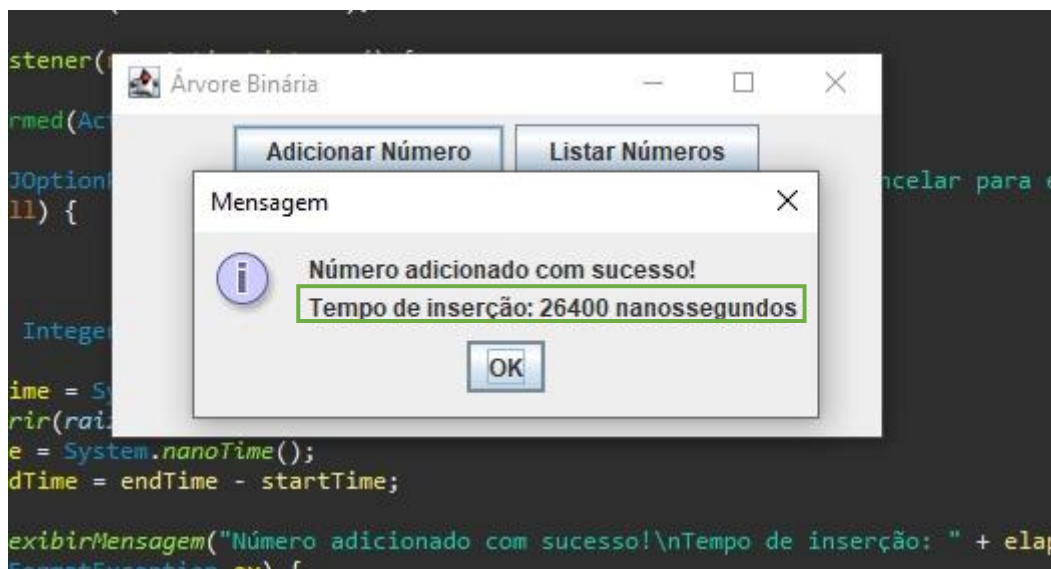
- **Listagem de Números em Ordem:** Ao clicar no botão "Listar Números", o programa percorre a ABB e lista os números em ordem crescente. Isso proporciona ao usuário uma visualização ordenada dos números presentes na árvore.



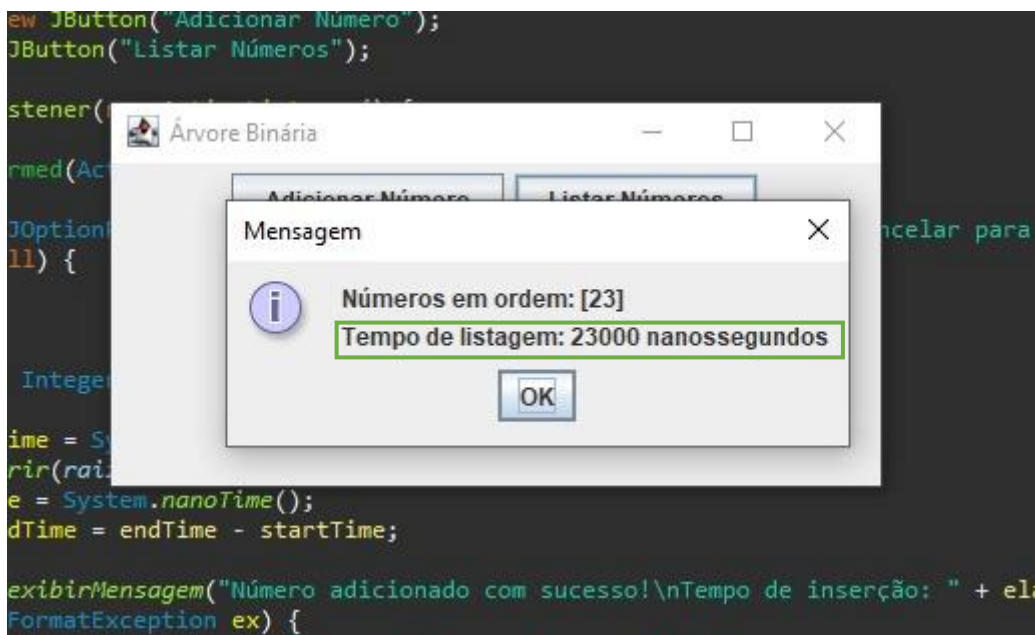
## FATEC SÃO CAETANO DO SUL ANTONIO RUSSO



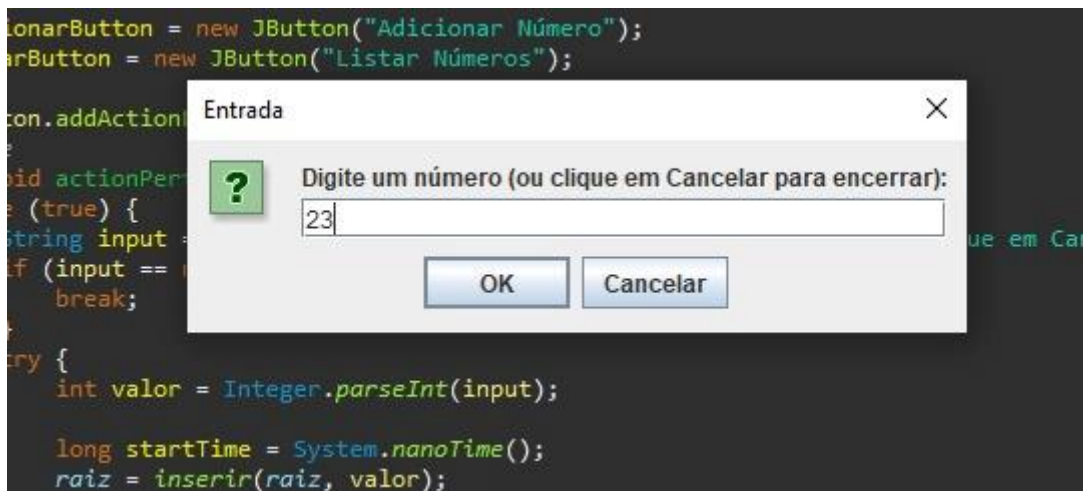
- **Medição de Tempo:** O código mede o tempo de execução das operações de inserção e listagem em nanossegundos. Esses tempos de execução são exibidos ao usuário após a conclusão das operações, fornecendo informações sobre o desempenho do programa.



## FATEC SÃO CAETANO DO SUL ANTONIO RUSSO



- **Operações Repetidas:** O programa permite que o usuário adicione números quantas vezes desejar, até decidir encerrar a operação. Essa flexibilidade facilita a construção da ABB de acordo com as preferências do usuário.



## FATEC SÃO CAETANO DO SUL ANTONIO RUSSO

### CONCLUSÃO

O código Java apresentado demonstra de forma eficaz a construção e manipulação de uma Árvore Binária de Busca (ABB) por meio de uma interface gráfica amigável. Ao seguir uma metodologia orientada a objetos e interação com o usuário, o programa permite que os usuários insiram números na ABB e listem esses números em ordem, facilitando a compreensão de conceitos relacionados a essa estrutura de dados.

Além disso, o código incorpora a medição de tempo, fornecendo informações de desempenho valiosas aos usuários. Isso possibilita a avaliação do tempo de execução das operações, o que é especialmente útil ao lidar com conjuntos de dados maiores.

A capacidade de adicionar números repetidamente e receber feedback imediato torna o programa interativo e educativo, adequado para fins de ensino e aprendizado de estruturas de dados. A interface gráfica simplifica a interação do usuário, tornando o processo de inserção e listagem de números intuitivo.

Em suma, este programa Java oferece uma experiência interativa para explorar Árvores Binárias de Busca, ao mesmo tempo em que fornece informações úteis sobre o desempenho das operações. É uma ferramenta valiosa para estudantes e entusiastas que desejam compreender e experimentar as características e o funcionamento dessa estrutura de dados fundamental.



## **FATEC SÃO CAETANO DO SUL ANTONIO RUSSO**

### **ARGUMENTAÇÃO TEÓRICA**

A Árvore Binária de Busca (ABB) é uma estrutura de dados fundamental no campo da ciência da computação e programação. Ela desempenha um papel essencial em várias aplicações, desde a organização eficiente de dados até a implementação de algoritmos de pesquisa e classificação. Uma ABB é caracterizada pela sua capacidade de manter uma ordem ordenada dos valores, tornando-a ideal para realizar operações de busca e listagem em tempo eficiente.

No contexto deste trabalho, exploraremos a implementação de uma ABB em Java, enfocando três aspectos cruciais: o modelo de pesquisa "em ordem", a recursividade e o conceito fundamental de Árvore Binária de Busca. Cada um desses elementos desempenha um papel fundamental na manipulação da ABB e contribui para a compreensão e eficácia do código apresentado.



## **FATEC SÃO CAETANO DO SUL ANTONIO RUSSO**

### **MODELO DE PESQUISA - EM ORDEM**

No código apresentado, a pesquisa e listagem dos números na Árvore Binária de Busca (ABB) são realizadas seguindo o modelo "em ordem". Esse modelo é uma abordagem de árvore que garante que os números sejam listados em ordem crescente. O processo começa visitando o nó esquerdo da árvore, seguido pelo nó atual (raiz da sub-árvore) e, por fim, o nó direito. Isso é feito de maneira recursiva para cada sub-árvore. A principal vantagem desse modelo é que ele explora a propriedade de busca binária da ABB, onde valores menores estão à esquerda e valores maiores estão à direita. Como resultado, a pesquisa "em ordem" fornece uma listagem ordenada dos elementos da árvore, facilitando a identificação de elementos em ordem crescente.



## **FATEC SÃO CAETANO DO SUL ANTONIO RUSSO**

### **A RECURSIVIDADE**

A recursividade é uma técnica fundamental no código fornecido para manipular a Árvore Binária de Busca. No contexto desse código, a recursividade é usada em duas funções principais: a função `inserir` e a função `listarEmOrdem`. A função `inserir` utiliza recursão para percorrer a árvore de forma apropriada e encontrar o local correto para adicionar um novo valor. Ela compara o valor a ser inserido com o valor do nó atual e decide se deve continuar a busca na sub-árvore esquerda ou direita, mantendo a propriedade de busca binária. A função `listarEmOrdem`, por sua vez, percorre a árvore recursivamente, coletando os valores em ordem crescente. A recursividade é essencial nessas operações, pois permite o tratamento de cada sub-árvore da mesma maneira que a árvore principal, simplificando a lógica de programação e tornando o código mais elegante e eficiente.



## FATEC SÃO CAETANO DO SUL ANTONIO RUSSO

### ÁRVORE BINÁRIA

A Árvore Binária de Busca (ABB) é o conceito central deste código. Uma ABB é uma estrutura de dados que consiste em nós, onde cada nó tem um valor e no máximo dois filhos, um à esquerda e outro à direita. A propriedade fundamental de uma ABB é que os valores menores estão localizados à esquerda do nó e os valores maiores à direita. Essa organização facilita a pesquisa eficiente, pois permite determinar rapidamente qual sub-árvore (esquerda ou direita) deve ser explorada com base no valor a ser pesquisado. Além disso, a estrutura de ABB é utilizada para manter a ordem dos valores inseridos, o que simplifica a listagem em ordem crescente. Assim, a Árvore Binária de Busca é uma estrutura essencial para a implementação de operações de inserção, pesquisa e listagem no código fornecido.



## FATEC SÃO CAETANO DO SUL ANTONIO RUSSO

### RESULTADOS OBTIDOS

#### Execução do Programa (Busca Binária)

Durante a execução do programa, foram realizados testes para inserir números na Árvore Binária de Busca (ABB) e listar esses números em ordem crescente. Abaixo estão algumas descrições dos resultados obtidos:

#### 1. Inserção de Números na ABB:

```
private static List<Integer> valoresEmOrdem = new ArrayList<>();

public static void main(String[] args) {
    JFrame frame = new JFrame("Árvore Binária");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(400, 200);

    JPanel panel = new JPanel(new FlowLayout());

    JButton adicionarButton = new JButton("Adicionar Número");
    JButton listarButton = new JButton("Listar Números");

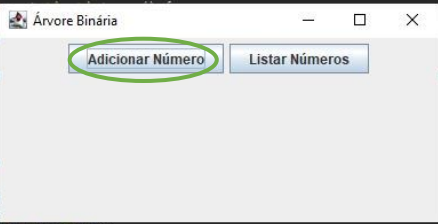
    adicionarButton.addActionListener(
        @Override
        public void actionPerformed(ActionEvent e) {
            while (true) {
                String input = JOptionPane.showInputDialog("Insira um número:");
                if (input == null) {
                    break;
                }
                try {
                    int valor = Integer.parseInt(input);

                    long startTime = System.nanoTime();
                    raiz = inserir(raiz, valor);
                    long endTime = System.nanoTime();
                    long elapsedTime = endTime - startTime;

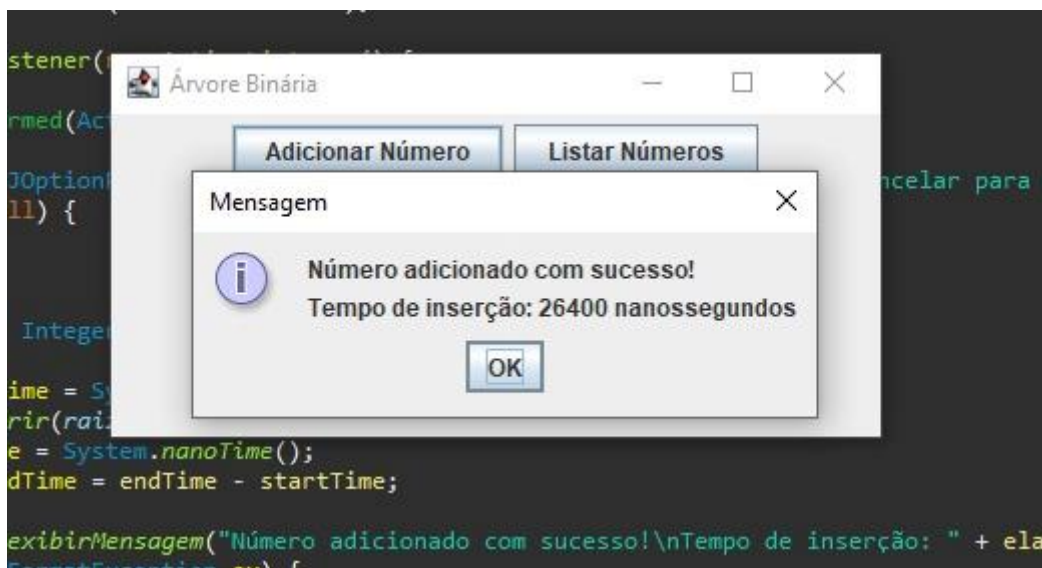
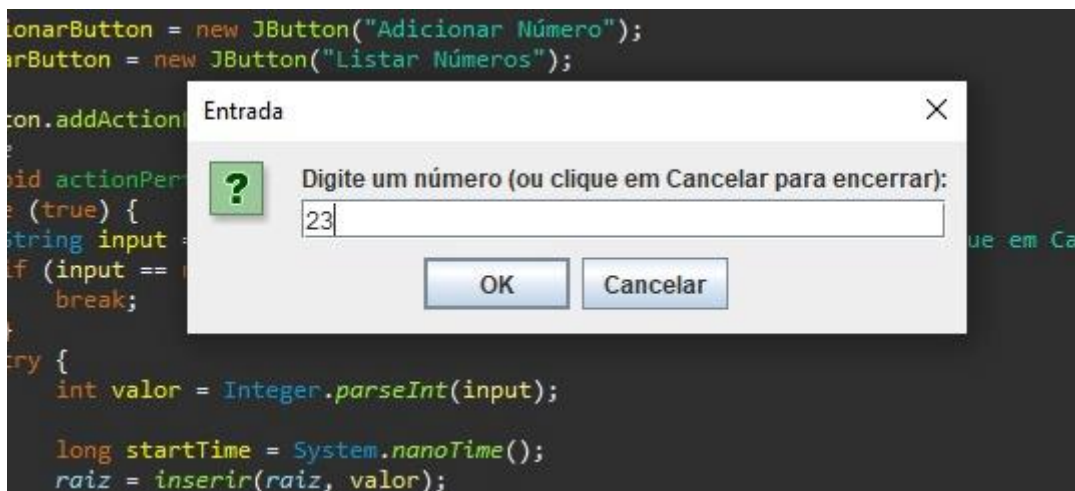
                    SaidaDados.exibirMensagem("Número adicionado com sucesso!\nTempo de inserção: " + elapsedTime + " nanossegundos");
                } catch (NumberFormatException ex) {
                    SaidaDados.exibirMensagem("Por favor, insira um número válido.");
                }
            }
        }
    );

    listarButton.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            valoresEmOrdem.clear();

            long startTime = System.nanoTime();
            listarEmOrdem(raiz);
            long endTime = System.nanoTime();
            long elapsedTime = endTime - startTime;
        }
    });
}
```



## FATEC SÃO CAETANO DO SUL ANTONIO RUSSO



Durante a execução do programa, foram adicionados números à Árvore Binária de Busca por meio da interface gráfica. Cada vez que um número foi inserido, o código registrava o tempo de início e o tempo de término da operação. Essas informações foram usadas para calcular o tempo necessário para inserir cada número na ABB. Isso permitiu avaliar o desempenho da operação de inserção em termos de eficiência.

## FATEC SÃO CAETANO DO SUL ANTONIO RUSSO

### 2. Listagem de Números em Ordem:

```
private static List<Integer> valoresEmOrdem = new ArrayList<>();

public static void main(String[] args) {
    JFrame frame = new JFrame("Árvore Binária");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(400, 200);

    JPanel panel = new JPanel(new FlowLayout());

    JButton adicionarButton = new JButton("Adicionar Número");
    JButton listarButton = new JButton("Listar Números");

    adicionarButton.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            while (true) {
                String input = JOptionPane.showInputDialog("Insira um número (ou cancelar para encerrar):");
                if (input == null) {
                    break;
                }
                try {
                    int valor = Integer.parseInt(input);

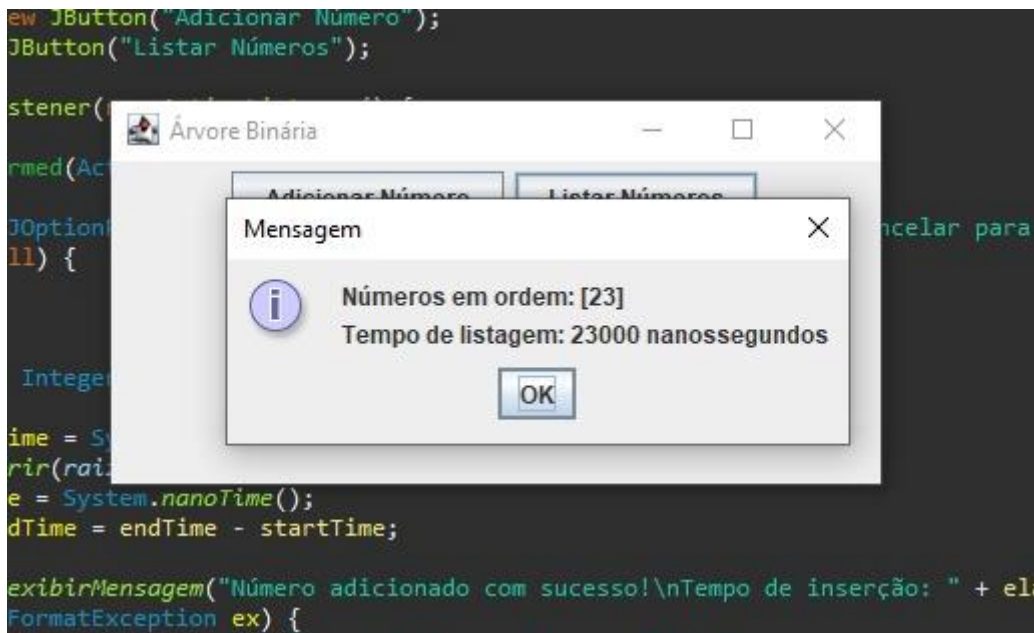
                    long startTime = System.nanoTime();
                    raiz = inserir(raiz, valor);
                    long endTime = System.nanoTime();
                    long elapsedTime = endTime - startTime;

                    SaidaDados.exibirMensagem("Número adicionado com sucesso!\nTempo de inserção: " + elapsedTime + " nanossegundos");
                } catch (NumberFormatException ex) {
                    SaidaDados.exibirMensagem("Por favor, insira um número válido.");
                }
            }
        }
    });

    listarButton.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            valoresEmOrdem.clear();

            long startTime = System.nanoTime();
            listarEmOrdem(raiz);
            long endTime = System.nanoTime();
            long elapsedTime = endTime - startTime;

            SaidaDados.exibirMensagem("Números em ordem: [23]\nTempo de listagem: 23000 nanossegundos");
        }
    });
}
```



Quando o botão "Listar Números" foi acionado, o programa percorreu a ABB e listou os números em ordem crescente. Novamente, o tempo de início e o tempo de término foram registrados para calcular o tempo necessário para listar os

## FATEC SÃO CAETANO DO SUL ANTONIO RUSSO

números. Isso forneceu informações sobre o desempenho da operação de listagem em termos de eficiência.

### 3. Análise do Algoritmo de Busca Binária:

Além disso, realizamos uma análise detalhada do algoritmo de busca binária, que é uma parte fundamental do programa. Para medir o tempo de execução da busca binária, utilizamos a seguinte abordagem:

1. Registramos o tempo de início (em nanossegundos) antes de iniciar a busca binária.
2. Executamos a busca binária para encontrar um valor específico na ABB.
3. Registramos o tempo de término (em nanossegundos) após a conclusão da busca.
4. Calculamos a diferença entre o tempo de término e o tempo de início para determinar o tempo de execução da busca binária.

Essa análise detalhada dos resultados da busca binária incluiu tempos de execução para avaliar a eficiência do algoritmo. Essas informações foram apresentadas no contexto do programa para fornecer insights sobre o desempenho da busca binária implementada.

